Manufacturing Systems Interoperability in Dynamic Change Environments

By

Neil Hastilow

Under the Supervision of

Dr. R. I. M. Young

A Doctoral Thesis

Submitted in partial fulfilment of the requirements

for the award of

Doctor of Philosophy of Loughborough University

June 2013

Certificate of Originality
Thesis Access Conditions and Deposit Agreement

Students should consult the guidance notes on the electronic thesis deposit and the access conditions in the University's Code of Practice on Research Degree Programmes

Author: Neil Hastilow

Title: Manufacturing Systems Interoperability in Dynamic Change Environments

I *Neil Hastilow,* Yew Tree House, High St., Marchington, Uttoxeter, "the Depositor",
would like to deposit '*Manufacturing Systems Interoperability in Dynamic Change Environments*', hereafter referred to as the "Work", once it has successfully been examined in Loughborough University Institutional Repository

Status of access OPEN

Moratorium Period……years, ending……../……20…….

Status of access approved by (CAPITALS):……

Supervisor (Signature)……...……...

School of……...……

Author's Declaration *I confirm the following :*

CERTIFICATE OF ORIGINALITY
This is to certify that I am responsible for the work submitted in this thesis, that the original work is my own except as specified in acknowledgements or in footnotes, and that neither the thesis nor the original work therein has been submitted to this or any other institution for a degree

NON-EXCLUSIVE RIGHTS
The licence rights granted to Loughborough University Institutional Repository through this agreement are entirely non-exclusive and royalty free. I am free to publish the Work in its present version or future versions elsewhere. I agree that Loughborough University Institutional Repository administrators or any third party with whom Loughborough University Institutional Repository has an agreement to do so may, without changing content, convert the Work to any medium or format for the purpose of future preservation and accessibility.

DEPOSIT IN LOUGHBOROUGH UNIVERSITY INSTITUTIONAL REPOSITORY
I understand that open access work deposited in Loughborough University Institutional Repository will be accessible to a wide variety of people and institutions - including automated agents - via the World Wide Web. An electronic copy of my thesis may also be included in the British Library Electronic Theses On-line System (EThOS). I understand that once the Work is deposited, a citation to the Work will always remain visible. Removal of the Work can be made after discussion with Loughborough University Institutional Repository, who shall make best efforts to ensure removal of the Work from any third party with whom Loughborough University Institutional Repository has an agreement. Restricted or Confidential access material will not be available on the World Wide Web until the moratorium period has expired.

- That I am the author of the Work and have the authority to make this agreement and to hereby give Loughborough University Institutional Repository administrators the right to make available the Work in the way described above.
- That I have exercised reasonable care to ensure that the Work is original, and does not to the best of my knowledge break any UK law or infringe any third party's copyright or other Intellectual Property Right. I have read the University's guidance on third party copyright material in theses.
- The administrators of Loughborough University Institutional Repository do not hold any obligation to take legal action on behalf of the Depositor, or other rights holders, in the event of breach of Intellectual Property Rights, or any other right, in the material deposited.

*The statement below shall apply to **ALL** copies:*

This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

Restricted/confidential work: All access and any copying shall be strictly subject to written permission from the University Dean of School and any external sponsor, if any.

Author's signature……..Date……...……...……

user's declaration: for signature during any Moratorium period (Not Open work): ***I undertake to uphold the above conditions:***			
Date	Name (CAPITALS)	Signature	Address

Abstract

The benefits of rapid i.e. nearly real time, data and information enabled decision making at all levels of a manufacturing enterprise are clearly documented: the ability to plan accurately, react quickly and even pre-empt situations can save industries billions of dollars in waste. As the pace of industry increases with automation and technology, so the need for accurate, data, information and knowledge increases. As the required pace of information collection, processing and exchange change so to do the challenges of achieving and maintaining interoperability as the systems develop: this thesis focuses on the particular challenge of interoperability between systems defined in different time frames, which may have very different terminology. This thesis is directed to improve the ability to assess the requirement for systems to interoperate, and their suitability to do so, as new systems emerge to support this need for change.

In this thesis a novel solution concept is proposed that assesses the requirement and suitability of systems for interoperability. The solution concept provides a mechanism for describing systems consistently and unambiguously, even if they are developed in different timeframes. Having resolved the issue of semantic consistency through time the analysis of the systems against logical rules for system interoperability is then possible. The solution concept uses a Core Concept ontology as the foundation for a multi-level heavyweight ontology. The multiple level ontology allows increasing specificity (to ensure accuracy), while the heavyweight (i.e. computer interpretable) nature provides the semantic and logical, rigour required.

A detailed investigation has been conducted to test the solution concept using a suitably dynamic environment: Manufacturing Systems, and in particular the emerging field of Manufacturing Intelligence Systems. A definitive definition for the Manufacturing Intelligence domain, constraining interoperability logic, and a multi-level domain ontology have been defined and used to successfully prove the Solution Concept. Using systems from different timeframes, the Solution concept testing successfully identified systems which needed to interoperate, whether they were suitable for interoperation and provided feedback on the reasons for unsuitability which were validated as correct against real world observations.

Keywords: ontology, manufacturing intelligence, semantics, foundation ontology, core concept ontology, manufacturing systems, interoperability.

Acknowledgements

I would like to thank my supervisor Dr Bob Young for his patient, thought provoking and insightful support, encouragement and guidance. Working with Bob over the last 3 years has resulted in a significant change in my thought processes and perception of the world around me for which I am a better engineer and extremely grateful. Bobs good humoured but challenging input have not only ensured this work is something I am very proud of, but also something I have enjoyed a great deal.

I would also like to thank Prof Keith Case for acting as my independent reviewer, Dr. Nitishal Chungoora (Tish) for his extensive support in learning the programming and ontology structuring skills and Dr. George Gunendran for is support with the IODE toolset.

I would like to thank Dr Nigel Bird and Dr Mark Turner for proposing, authorising and then supporting my part time studies as part of my professional development. I would also like to thank the subject matter experts who contributed to the MI definition activities, whose combined knowledge was crucial to this work.

I would like to thank my parents for their untiring love, support and drive to ensure I achieve a potential I did not necessarily know I had.

Finally I would like to thank my wife Helen. Without her love and support this would not have been possible. She has run our home, our lives and our family for 3 years. Not only has she never begrudged the loss of weekends, evenings, holidays, and our life revolving around my research schedule but her constant, unwavering support whether it be re-assurance, a timely cup of tea or the knowledge that she will sort ‘everything else out’ is what has given me the strength to see this through. This, like everything I do is because of and for her.

List of Acronyms

AI	Analysis Indicator
BPMN	Business Process Modelling Notation
CAA	Civil Aviation Authority
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
CAPP	Computer Aided Process Planning
CIM	Computer Integrated Manufacturing
CL	Common Logic
CNC	Computer Numeric Control
EASA	European Aviation Safety Agency
ECLIF	Extended Common Logic Interchange Format
EIC	Engineering Improvement Centre
EMI	Enterprise Manufacturing Intelligence
ERP	Enterprise Resource Management
GD&T	Geometric Design and Tolerancing
HR	Human Resources
IC	Integrity Constraint
ICT	Information and Communications Technology
IMKS	Interoperable Manufacturing Knowledge System
IODE	Integrates Ontology Development Environment
IR	Inference Rule
ISO	International Standards Organisation
IT	Information Technology
KBE	Knowledge Based Engineering
KFL	Knowledge Framework Language
KIF	Knowledge Interchange Framework
KPI	Key Performance Indicator
KPV	Key Process Variable

LAN	Local Area Network
MDI	Model Driven Interoperability
MES	Manufacturing Execution System
MESA	Manufacturing Execution Systems Association
MI	Manufacturing Intelligence
MLO	Medium Level Ontology
MRP	Material Requirement Planning
MSCoC	Manufacturing Systems Centre of Competence
OEE	Overall Equipment Effectiveness
OEM	Original Equipment Manufacturer
PDM	Product Data Management
PFMEA	Process Failure Mode Effect Analysis
PI	Performance Indicator
PIM	Platform Independent Model
PLM	Product Lifecycle Management
PPM	Parts Per Million
PSL	Process Specification Language
QMS	Quality Management System
SCADA	Supervisory Control and Data Acquisition
SFDM	Shop Floor Data Management
SFIT	Shop Floor Information Technology
SMIF	Semantic Manufacturing Interoperability Framework
UML	Unified Modelling Language
WIP	Work In Progress
XML	Extensible Mark-up Language

Table of Contents

Table of Figures

1 Introduction

1.1 Context

As business enterprises become more data and information rich they have developed systems and processes to help them manage this data and information. As these systems have become more powerful and flexible they have also become crucial repositories for the information of those businesses. In the 'information age' in which these enterprises are currently operating this data, information and knowledge is as critical an asset as the capital assets on the company accounts (Gunendran, Young 2006). The importance of the systems to manage data, information and knowledge has been quickly recognised as a massive commercial opportunity leading to an explosion in the number of systems to provide this capability over the last 20 years.

The development of data, information and knowledge systems, hereafter referred to as Knowledge Systems or KS, was initially driven by the individual functions within business enterprises leading to finance, material scheduling, process planning, design and drafting, human resources, quality, marketing systems etc that were entirely separate with limited information flow between them. This approach was reinforced by the technology and architecture requirements of multi-domain systems; prior to fast, flexible networks, the solutions would have required a large mainframe which would have been impractical for many small to medium enterprises. As technology has developed as an enabler, so too has the recognition of the power of combining these KS domains to allow the re-use of the system contents across the enterprise, and when combined, the contents of the systems can achieve a synergy due to the wider perspective on information.

Over the last 10 years there has been a significant growth in the development of large unitary systems that control multiple aspects of a business, key examples being: Enterprise Resource Planning (ERP) systems such as SAP which can cover all aspects of customer requirements management, human resources, material planning, process planning, finance, and quality etc, and Product Lifecycle Management (PLM) systems which provide authoritative systems for all product related data such as product definition, configuration, process planning, validation etc. It can be seen that these large systems can overlap, requiring similar data to live in both systems, such as process planning information. For the enterprises that can afford to implement these large systems, there are still areas of requirement even these systems do not meet. Where enterprises have legacy systems from which they are not in a position to migrate, there is a need for the systems within the enterprise to interoperate. Significant levels of research have been carried out on methods of providing system interoperability within enterprises, however, there is an emerging domain

which poses some unique challenges when compared to other business function requirements; Manufacturing Intelligence (MI).

Historically, shop floor systems were used to deliver machine instructions (CNC programs) and collect and store output data (cycle times, numeric results etc). This data was manually controlled and processed, meaning that even with significant manpower dedicated to the task, only delayed, partially informed reaction to events could be managed. Manufacturing Intelligence aims to leverage modern Information Communication Technologies (ICT) to collect data from and monitor production systems in real time, and based on knowledge embedded within the system, react to occurrences or infer indications of an upcoming occurrence and pre-empt them. As an ideal, MI's core function is to provide appropriate responses to the information it has. It can be categorised as a knowledge system, as knowledge is required to respond appropriately to information. This knowledge can be shared with other systems such as PLM systems. MI is conventionally considered part of a Manufacturing Execution System (MES) but this knowledge and data is crucial to all aspects of the enterprise, included those that may have limited exposure to the MES system, and therefore there is a strong requirement for interoperability. Where MI differs to other domains of the enterprise's systems is in the number of different systems it interfaces with and the stability of the environment: each production area and resource within that area can be considered a separate 'system'. MI is a key challenge as the toolset for MI has not yet been fully developed, meaning that a number of application systems are required to carry out the function, and due to the rapid development of this field these systems are emerging, changing and becoming obsolete very rapidly. This gives rise to the requirement to manage the interoperability not just across current systems but across end of life and emerging and not yet developed systems.

1.2 The research strategy

1.2.1 Aims and objectives

This research aims to define a mechanism for evaluating system interoperability requirements and capability in environments which experience rapid change. This mechanism should:

- Identify where interoperation is required between defined systems and whether those systems meet the requirements for interoperability.
- Identify specifically how a system fails to meet the requirement of interoperability to enable the failure to be addressed.

1.2.2 Scope

This research focuses on understanding the mechanisms for interoperation between manufacturing intelligence (MI) systems and other systems and defines a mechanism for providing interoperability through time within the MI domain. MI systems are a sub-type of Manufacturing System. The MI domain has been chosen as it is an emerging domain subject to disparate and rapid rates of system and data change which provides a suitable challenge for this research to address. MI was identified as an area of increasing industrial focus, and therefore relevance, due to the potential operational efficiency benefits it could enable. This was confirmed through the literature review in Chapter 2 and the industrial investigation in Chapter 3.

This research used the aerospace discrete part manufacturing industry to create and test the proposed solution concept, however it was anticipated that the results would be applicable beyond this industry.

1.2.3 Research method

This research was carried out in conjunction with a large international Aerospace Company. The inclusion of the industrial aspect gave key insight into the industrial relevance of the research (as described in Chapter 3), including potential future applications and benefits, as well as providing a vital source of data with which to build the experimental solution.

This research used a hypothesis and test method. The initial literature review focused on 3 main related topics: Manufacturing Intelligence, Systems Lifecycles and Information Sharing. These topics were the initiation points for the literature research to establish the current state of the art regarding the understanding in the research domain. This was followed by industry based research into the challenges to interoperability to gain an understanding of the practical and real world implications of the issues this research aims to address. Against this

background work, the novel proposal for interoperability in dynamic change environments was then defined along with the relating research questions (see Section 4.3):

1. What is Manufacturing Intelligence?
2. What are the concepts in the MI Systems domain?
3. Can a domain foundational or core concept ontology be defined and formalised?
4. Can the proposed concept be proven through the formalisation of the ontology.

These research questions were the focus of the practical research which then resulted in the embodiment of the proposed solution. This solution was then tested to verify the solution and prove or disprove the research hypothesis.

1.2.4 Research hypothesis

The research hypothesis was stated as “Multiple systems can be defined using a consistent foundational ontology comprising constraining logic to highlight whether each system meets the requirements for interoperation with the others. Interoperability can then be ensured for and with future systems and versions of systems through the inherent process of system definition supporting the rapid development of interoperable systems.”

1.3 Solution development tools and techniques

The following sections describe the key tools and techniques that were selected, interpreted and developed for use in this research and the reason for their selection.

The approach can be summarised as:

- Using a heavyweight foundational ontology concept to provide consistency to multiple versions of specialised ontologies and exploring the concept as applied to Manufacturing Systems interoperability.
- Using a reinterpretation of the Noy, McGuiness (2001) approach to suit the development of a heavyweight ontology:
- Using Business Process Mapping Notation diagrams to capture real world systems, their 'agents', process flows, information flows, logic and objects (OMG 2012, OMG 2008). These could then be used to populate the 'term pool' as well as develop logic, relationships and eventually populate the knowledge base and create test cases which were captured in a diagrammatic style loosely based on IDEF0 (FIPS 1993).
- Using Unified Markup Language Class Diagrams to structure the term pool and visualize the relationships throughout the ontology development.
- Formalising the ontology by codifying the ontology using the IODE heavyweight ontology tool, and Common Logic (IODE 2010).

The use of foundational ontology concept is fundamental to the solution proposal, as the method for providing semantic consistency. The use of a heavyweight ontology provides greater rigour and enables the use of equally rigorous, machine executed logic(see Chapter 2). This ability to embed complex Common Logic into the ontology is a critical requirement of the solution proposal.

The Noy, McGuiness (2001) method has been demonstrated by a number of other projects to be an effective framework for ontology development in the literature review, and suitable for reinterpretation for the development of heavyweight ontologies. It is a manual method, which allows the user to focus on the most relevant concepts at the possible cost of domain concept coverage. However this risk was mitigated by the use of a number of concept enumeration approaches as detailed in Chapter 6.

BPMN was selected for its ability to clearly represent a process flow, the process agents, events and information flows. Although the toolset is potentially extremely comprehensive, the main strength of BPMN was that it is intuitive for individuals within the manufacturing industry due to is similarity to many common flow charting standard. To make the most of this attribute a small subset of the available chart elements were used. This resulted in no

loss of process fidelity but enabled the process owners to chart their own process, mitigating the risk of any loss of fidelity when communicating to an independent scribe or when learning to use the more complex compound chart elements.

IDEF0 (FIPS 1993) is an industry standard for system notation, so a basic derivation of its representation of a system was used due to the common understanding of the notation amongst the contributors.

UML class diagrams allow the creation of structured concept taxonomies including the nature and description of the relationships between the concepts e.g. binary, tertiary etc. The facet or slots of the concepts were not used in this case as the functionality of the ontology was intended to be provided by the constraining logic and relationships.

The IODE version 4.1.2 tool was selected based on its success in the Interoperable Manufacturing Knowledge Systems (IMKS) project, both because this indicated it was a suitable tool and also because this created the potential to reuse parts of the IMKS work if appropriate (Young et al. 2007, Young et al. 2010, Chungoora, Young 2010a, Chungoora, Young 2010b), as it would share the IODE systems in-built Medium Level Ontology (MLO) and system logic as a base.

For detailed descriptions of the tools and their merits see Chapter 2.

1.4 Thesis structure

This thesis has been structured into into the following chapters:

- Introduction
    - Introducing the aims and scope of the research as well as stating the research method and hypothesis.
- Literature review
    - A review of relevant current literature and current state of the art knowledge.
- An industrial investigation of key interoperability issues
    - An investigation of key issues within the research domain in an industrial context, including the declaration of some key models that have been used with this work.
- A novel approach to manufacturing systems interoperability in dynamic change environments
    - The exploration of the research domain leading to the research questions and solution concept proposal.
- Understanding the scope and concepts in manufacturing intelligence
    - Research into the definition of the MI domain and enumerating the concepts within it.
- Ontology term, relationship and logic definition
    - Exploring the domain terms, relationships, and the constraining logic and their relative structure and evolution.
- Formalising the ontology
    - Detailing the formalisation of the terms, relationships and logic of the ontology into a heavyweight ontology solution including testing and validation of the solution.
- Testing and experimental results
    - Results of the experimental application of the ontology solution to real world systems examples allowing validation of the solution concept against actual experience.
- Discussion, conclusions and further work
    - Review of the research, the solution concept and hypothesis and suggestions for further work.

Relevant detailed supporting information has been included in Appendices.

2 Literature review

2.1 Introduction

The aim of this literature review was to identify the current state of the art with respect to the research scope. This review discusses the concepts and conclusions of the current state of the art as well as identifying particularly relevant work.

This literature review has been structured around three core topics which represent the scope of this work:

- Manufacturing Intelligence
- Systems Lifecycle
- Information Sharing

These topics have then been expanded upon in two further levels of detail (see Literature Review Structure section).

The method used for this review was an iterative process with three main steps:

- Review of academic papers and published material using structured resources such as Loughborough University Library, Wolfston School, IMechE , Rolls-Royce Technical Library and focused internet resources.
- Expansive concept keyword searches based on the initial stage of the review.
- Focused concept using subject matter experts.

It was anticipated that the first iteration of this process would be based on the 3 core topics, with subsequent iterations expanding through the lower level detail.

From the cursory knowledge of the 3 core topics, it was also anticipated that significant levels of published work and information on Information Sharing should be returned using the term ‘Information sharing’ as a starting search expression, which can be expanded upon using related search terms. There should be published material on the Systems Lifecycle and Manufacturing topics, however, much of the core work relevant to these topics may be in other fields under other titles (e.g. in software development under the name ‘Application Lifecycle’ or business improvement under ‘Business Intelligence’), i.e. the initial research iteration (using these titles as the search terms) may return minimal information requiring more ‘lateral’ research.

2.2 Literature review structure

This review has been structured around a three core topics as shown in Figure 2-1. The review of literature was carried out at two levels. The 1st level covered the high level sub-topics, under which there may be a 2nd more detailed or specific level of research:

TOPIC	LEVEL 1	LEVEL 2
MANUFACTURING INTELLIGENCE	BUSINESS INTELLIGENCE	Business Intelligence, Manufacturing Intelligence and Dashboards
		Data Warehousing
	MANUFACTURING KNOWLEDGE/ INTELLIGENCE SYSTEMS	
	MANUFACTURING EXECUTION SYSTEMS	ISA 95 MES Model
		Manufacturing System Agility
	INTELLIGENT AUTOMATION	Agent Technology
		Holonic Manufacturing
SYSTEMS LIFECYCLE	SYSTEMS LIFECYCLE	Legacy systems
		Endurant vs. Perdurants
		Dynamism
		Lifecycle Timescales
		Application Lifecycle
		System Value and Condition Assessment
INFORMATION SHARING	DATA, INFORMATION AND KNOWLEDGE	Data, Information and Knowledge Maintenance
	INTEROPERABILITY	
	THE REQUIREMENT FOR INTEROPERABILITY	
	INTEGRATION VS INTEROPERABILITY	Integration Continuum
		Taxonomy
		Systems and Technology
		Information Systems Customisation for Integration
	PLM	PLM Information Exchange
	STANDARDS – SEMANTIC ISSUES	Standards Frameworks and Adoption
		Standards Adaptation
		Standards Flexibility
	ENTERPRISE INTEGRATION AND INTEROPERABILITY	Integrated, Unified and Federated Approaches
	INTERGRATION ARCHITECTURE	Model Driven Architecture
		Interoperability and Information Exchange Frameworks
		Semantic Interoperability frameworks
	ONTOLOGY	Ontologies and Interoperability
		Ontology Creation
		Ontology Merging
		Mapping
		Ontology Mapping
		Concept Mapping
		Heavyweight and Lightweight Ontologys
		Foundation Ontology
		Foundation Ontology Adoption
		Ontological vs Taxonomic approaches
		Descriptive and Common Logic
	CONTEXT OR VIEWPOINTS	
TOOLS	XML	
	XML Based Exchange	
	RDF and OWL	
	Common Logic Tools	
	PSL	
	IDEF0 and IDEF3	
	UML	
	Protégé	
	BPML	
	Business to Manufacturing Mark-up Language (B2MML)	

Figure 2-1 Literature review structure

2.3 Manufacturing intelligence

The majority of recently published material on manufacturing intelligence (also referred to as 'Enterprise Manufacturing Intelligence'(Siemens AG 2011, Unver 2012)) is technical or sales material (ISA 2000, ISA 2005), with however relatively little work published has been published on the future direction of the MI field. This literature focuses on what functionality specific MI products have without providing a coherent description of the purpose or scope of MI. These tools and literature meet the current basic understanding of Manufacturing Intelligence as described in the following section. This understanding is typified in published standards and literature descriptions of MI as a dynamic and rapidly changing and relatively unpredictable field (Unver 2012, ISA 2000, ISA 2005, Chen, Chien 2011).

Manufacturers need a comprehensive view of their operations at all times. They must be equipped with the most relevant information in the proper context so that holistic decisions can be made appropriately. Manufacturing Intelligence is the synthesis of the key elements (manufacturing performance, business intelligence, and real-time information) required for global manufacturers to compete in the current business environment. Manufacturing Intelligence is the next generation of decision support capabilities for global manufacturers. It is about making real-time manufacturing information, with "drill anywhere" capabilities, available to manufacturing executives and plant staff to facilitate decision-making and improve their supply chain performance (Unver 2012, Fulcher 2005, Toung 2006).

Brown(2007) comments on the recent heavy investment in MI which is being driven by the need to optimize operations and compliance with company goals, targets and corporate responsibilities. The author describes MI as the primary area of investment, projected to even exceed spending on ERP.

The key functions of MI have been described as (Jacobson, & Eriksen 2011):

- Aggregation: Making available data from many sources, most often databases.
- Contextualization: Providing a structure, or model, for the data that will help users find what they need, usually a folder tree utilizing a hierarchy such as the ISA-95 standard.
- Analysis: Enabling users to analyze data across sources and especially across production sites. This often includes the ability for true ad hoc reporting.
- Visualization: Providing tools to create visual summaries of the data to alert decision makers and call attention to the most important information of the moment. The most common visualization tool is the dashboard.

- Propagation: Automating the transfer of data from the plant-floor up to enterprise-level systems such as SAP, or vice versa.

Manufacturing Intelligence has been described as “the synthesis of three key elements (manufacturing performance, business intelligence, and real-time information) required for global manufacturers to compete in the current business environment. Manufacturing intelligence is the next generation of decision support capabilities for global manufacturers. However this raises unanswered questions of how real-time information systems and business level information system can not only be made to interoperate but how that interoperation can be sustained through time as they are subject to disparate systems lifecycles and rates of change(Toung 2006, Panetto, Molina 2008, Wang 2010, Mochal 2005).

The major PLM and Manufacturing Execution System (MES) system provider Siemens have stated that MES systems can be used to provide real-time information about what is happening on the shop floor for strategic (medium to long term) decision making as well as operational (immediate to short term) decisions (Siemens AG 2011, Panetto, Molina 2008). Within this scope, the function of informing manufacturing decision making and enabling appropriate reactions is termed 'Manufacturing Intelligence', a term which they describe as evolving from the more established field of 'Business Intelligence': the collection and use of business data for decision making. This description implies MI is a sub-function of MES.

The view of MI as a subset of is MES shared by Brown(2007), however elements of the emerging MI functions such as the extension in the real time/ control functions on the shop floor are extending the conventional limits of MES definitions.

Recent literature seems to confuse matters further (Siemens AG 2011, Unver 2012): the literature describes the term ‘Enterprise Manufacturing Intelligence’ showing the MES system crossing the ISA95 (ISA 2000, ISA 2005) Level 3 and 4 threshold to allow enterprise level reporting while also integrating with the controls level systems at levels 0,1 and 2 to provide real time data. This literature references the use of the ISA 95 model directly, but does not attempt to resolve the paradox that the standard defines Level 3 as the ‘Execution Level’ where the MES resides, hence having referenced ISA 95 to describe their proposed solutions, they then contravene the std. by having the MES operating at Levels 3 and 4, while also being integrated with levels 0,1 and 2; implying there is no reason to discriminate between these levels. The term Enterprise Manufacturing Intelligence seems to represent an aspiration to make MI more informed and powerful by pulling in information from many other areas of an enterprise, which in turn implies MI is outgrowing the ISA 95 definition of MI as a

sub function of MES. This ambiguity is referred to as 'flexibility' by MESA (MESA 2012), but it results in a risk of misalignments of interpretation between users of the standard.

It can be seen that despite the available and emerging standards describing manufacturing systems and data collection, the definition of MI is still highly ambiguous: lacking detail and specificity and with some key aspects of descriptive standards being contradictory. There is therefore a need for work to create a description of MI which defines both its scope and purpose in order to understand the interoperability requirements of the processes and related systems.

2.3.1 Business intelligence

Business Intelligence is a term for a large number of knowledge management products and functions that turn data into useful information that can support business decisions. These include Data Warehousing, Decision Support Systems (which enable forecasting) and Executive Information Systems (which provide direct access to domestic and external data sources for executive analysis)(Hodges 2007).

Some of the key concepts in Business Intelligence are common to other knowledge management initiatives such as PLM: these include:

- A single version of the truth (authoritative data sources such as the PLM Master Model)
- High levels of collaboration, enabled by global collaboration tools with version and access management.
- Flexible reporting carried out directly from the data storage source to ensure maximum use or re-use of the gathered data and information without the risk of unintended non-standard data manipulation.

Some of the commonly available tools used to provide these capabilities are products such as Microsoft's 'SharePoint', 'SQLServer' and 'Reportbuilder'(Jones).

2.3.1.1 Business intelligence, manufacturing intelligence and dashboards

MESA presents a hierarchical model consisting of Performance Indicators (PI also referred to as Key Performance Indicators or KPI) and Analysis Indicators (AI). At the top level are the enterprise indicators such as profit. If these deviates from expectation, the PI's at site level will be reviewed and it may be idenitified that a specific site is underperforming, hence a PI is an AI for the layer above it. Production is at the bottom of the Business Intelligence hierarchy and it is at this level that it can be referred to as Manufacturing Intelligence in manufacturing enterprises. Dashboards are near real time displays of PIs or KPIs at an operational level, however, they generally require user interpretation and action based on

experience so are also often referred to Data Dashboards. Business Intelligence systems refresh their data on a much lower frequency, such as once a day or shift, meaning that data can be out of date and there is a delay before feedback on any action taken (Scholten 2009). The reviewed literature describes these systems but does not describe the emerging field of Manufacturing Intelligence which merges the concepts of real time data collection and display with automated or highly informed decision making and actioning.

MESA has commissioned a study into the most commonly used KPIs (where a KPI is defined as the most important PI). Discounting the ones that are legislative requirements such as Health and Safety Incidents, it shows that the KPIs used by most successful large and small business are (Scholten 2009):

- On Time Delivery
- Manufacturing Cycle Time
- Total Inventory
- First Pass Yield
- Capacity Utilisation
- Customer reject rate (PPM)
- Batch/ Lot/ Unit Right 1st Time
- WIP inventory

These can be processed with Manufacturing Execution Systems (MES) data (and each other) to provide other calculated metrics and PIs such as:

- Quality
- Throughput
- Compliance
- Utilisation
- Inventory
- Customer service

It can be seen that the KPIs require significant data collection and a level of data manipulation, and the second list of PIs require further data manipulation. As these metrics are defined in Natural Language they can be interpreted in many ways, and then processed in varying ways. Each level of process provides opportunity for further levels of interpretation hence the comparison of KPIs across an enterprise could be misleading. Even an industry standard such as Overall Equipment Effectiveness (OEE) which is calculated from Quality x

Availability x Performance is subject to this problem with each element needing local interpretation for specific applications (See Ontology Section).

2.3.1.2 Data standards

Significant work has been carried out on the definition of standards for data collection, which have been increasingly focused on the challenge of manufacturing shop floor data collection such as the MANDATE standard (ISO 2010). This work references many other standards that provide guidance on nonclementure for manufacturing organisations. It also defines that certain aspects are out of scope limiting is applicability in the MI field:

- Product information
- Component information
- Cutting tools
- Technical maintenance information
- Enterprise modelling (including tools and techniques)

The purpose of the standard is stated as aiding the collection of data and information at Level 2 of the organisation and providing to Level 3 for operations management (ISA 2000, ISA 2005). While this represents beneficial progress, it is in conflict with the idea that MI may be displayed and used at the control or process level (using modern systems that enable a data driven business) or the MI may require data from all levels of the organisation(Siemens AG 2011) so serves to confuse the definition of MI rather than clarify it.

2.3.1.3 Data warehousing

Data warehousing is a approach that combines data and information into a single database suitable for analysis. The warehouse is not a unitary system (ie it is made up of a number of systems) and is designed to be used on an enterprise-wide basis. The system stores data, meta data and summary information. Data warehousing has evolved since its introduction in the 1990's from systems that took years to build and weeks or months to update (and hence were only used for long term decision making) to data marts which are smaller more agile versions that are interfaced to provide wider coverage. The literature states that current systems use a common meta data storage systems to overcome the problem of inconsistent meta data across the data marts. This, however, this is just the imposition of a global standard rather than a resolution (see Standards section). Current data warehouses require near real time updating and are used by a much increased number and variety of people (Hodges 2007).

2.3.2 Manufacturing knowledge/ intelligence systems

Work carried out on Industrial Product Services Systems shares similar requirements to MI systems (Meier, Roy & Seliger 2010). This work proposes a system that captures in-service data and information from the process and using data mining algorithms (such as pattern recognition or pattern classification) processes it to knowledge and uses these algorithms to create rules for process prognosis based on actual conditions. This work goes on to conclude that this capability needs developing for future PLM systems and for other life cycle stages such as Manufacture and Disposal.

Izza (2009) published a comprehensive list and review of discrete industrial information systems such as CAD, CAM, ERP, EDM, MES, PDM, PLM etc however it omits MI systems. This is a common issue across most of the manufacturing systems literature reviewed: very little has been published focusing on MI systems or their specific requirements and considerations for information sharing.

2.3.3 Manufacturing execution systems

Manufacturing Execution Systems have been characterised by MESA International (a trade association of MES vendors) as having these principal functions (ISA 2000, ISA 2005, Panetto, Molina 2008, Jr., Michel 2000):

- Planning system interface
- Data collection
- Exception management
- Work orders
- work stations
- Inventory / Materials and material movement

with MES supporting functions including:

- Genealogy
- Maintenance
- Time and Attendance
- Statistical Process Control
- Quality Assurance
- Process data
- Documentation Management

Some of these functions may be performed manually while others may be automated by individual software packages. The reviewed literature states that the increasing complexity of

intelligent closed loop control systems and the variations in functionality required from plant to plant has led to the view that MES systems should be modular with standard interfaces (Panetto, Molina 2008, Jr., Michel 2000). The relative limitations of using interfacing compared to integrating or interoperating are discussed in the information sharing section. It may not however, be appropriate to bring all functions within one system due to the increased inflexibility this can cause, requiring an ability for systems to interoperate, and to operate on separate but compatible lifecycles (Panetto, Molina 2008).

MES systems can be used to provide real-time information about what is happening on the shop floor for strategic (medium to long term) decision making as well as operational (immediate to short term) decisions (Siemens AG 2011, Panetto, Molina 2008). The field of informing manufacturing decision making and enabling appropriate reactions is termed 'Manufacturing Intelligence', a term which has evolved from the more established field of 'Business Intelligence': the collection and use of business data for decision making. This description implies MI is a sub-function of MES.

MES are characterised by a heterogeneous set of applications which operate between direct equipment control and enterprise systems (Jr., Michel 2000). A detailed MES and integration standard is provided in the ISA 95 group of standards (ISA 2000, ISA 2005)

Scholten (2009)describes at length the functions of MES and their relative strengths and applications. Some key points are:

- MES sub systems sub sector reporting capabilities require further development
- MES are often used to collect vast amounts of production data, but reporting is generally an offline activity on a daily weekly or monthly basis.
- MES dashboard near real time data to allow human operational decision making

2.3.3.1 ISA 95 MES model

The definitive definition of a Manufacturing Execution System (MES) is the ISA95 standard and supporting papers from MESA (ISA 2000, ISA 2005, Scholten 2009). ISA95 describes a Enterprise Control System Integration (not automation) and its functions in levels with associated timescales:

- Level 4: Business Planning in logistics - popularly called the ERP Layer. This is referred to as the Enterprise Domain in the standard to be platform independent and these activities are focused on the longer term i.e. weeks and months
- Level 3: Referred to as the MES Layer this is described as the Control Domain in ISA95. These activities focus on activities such as plant efficiency, product quality,

material storage locations, machine availability etc. These activities are usually in the timescales of days, hours or minutes.

- Levels 2,1,0: These lower Control Domain levels are where the production process takes place, with the help of PLCs, sensors, actuator, SCADA solutions and other control systems. These are either monitored in a timescale of minutes, seconds or milliseconds (Level 2), sensed and manipulated (Level 1) or the actual production process (Level 0).

This model clearly defines the time-scope of MES to be days to minutes and the role to be operational activities. Some activities that can be considered to be based on MI systems and information such as KPI reporting and control sit partially in Level 3 and MES. Nevertheless, the (near) real-time nature of MI data clearly aligns it to Levels 2,1 and 0. As such, the traditional view as MI as a subset of MES is called into question, with it appearing to bridge both MES and Controls systems levels. While the published literature talks extensively about functions that straddle the border between Level 3 and 4, using tools such as Business to Manufacturing Mark-up Language (B2MML), there is no specific mention of functions that straddle Level 3 and the lower levels or tools to do this. While B2MML could be used for this interaction it will be limited to the scope of ISA95 which itself does not cover MI terms in detail and relies on natural language descriptions of concepts and their relationships, which are open to ambiguous interpretation. This ambiguity is referred to as 'flexibility' by MESA (MESA 2012), but it results in a risk of misalignments of interpretation between users of the standard.

The levels described in ISA95 are also reflected in the ISO 15531 MANDATE standard (ISA 2000, ISA 2005, ISO 2010). However, this standard uses the IEC 62264-1 model which is entirely compatible with the ISA95 wording:

- Level 4: Business Planning & Logistics, Plant Production Scheduling, Operational Management.
- Level 3: Manufacturing Operations and Control
- Level 2,1,0: Batch, Continuous, Discrete Control

2.3.3.2 Manufacturing system agility

Agility is the ability of an enterprise to make a flexible, dynamic, reconfigurable and quick response to change in an unpredictable business environment (Qiao, Liu 2009). In this paper the authors discuss a method of providing cross-lifecycle data systems using a unified data model. While this work only defines a solution concept it does provide a compelling link between manufacturing system interoperability, manufacturing decision making,

manufacturing organisational agility and the potential route to a solution using a unified data model approach.

2.3.4 Intelligent automation

To cope with the increasing pace, variability, flexibility and requirement for interaction between elements of the manufacturing enterprise (often referred to a 'holons') the MI approach has been used as an enabler for Intelligent Automation. The conventional Computer Integrated Manufacturing (CIM) could lead to large monolithic, centralised systems which can take many months to implement and be inflexible due to their complexity. The actual implementation of CIM developed into more decentralised, hierarchical solutions (Schoop et al. 2002). This work describes a manufacturing model where the distributed intelligent manufacturing elements or systems dynamically interact to achieve global and local manufacturing objectives. The flexibility of the overall system will be limited by the individual elements (or holons) as well as the coordinating or supervisory system.

A description of Intelligent Automation is proposed as the integration of several emerging technologies and paradigms (Schoop et al. 2002):

- Agent technology
- Mechatronics
- Intelligent Manufacturing
- Holonic control systems.

Key areas of technology research to enable Intelligent Automation have been identified within this work as being:

- Distributed planning, dynamic re-scheduling, and intelligent supervision
    - Planning and re-scheduling capabilities included in an automation island
    - Monitoring, Diagnosis, Recovery (decision-making processes)
    - Remote monitoring, diagnosis and maintenance
- Interoperability between Local Control, Co-ordination Control Level and Intra-Enterprise Level
    - Functional, Logic and Physical System Architecture; Agent based MES systems
- Automation Design Environment
    - Support the optimization of costs during the life-cycle of Automation Solutions
    - Integration of Discrete-Event Control Simulation with Process- and Component-Simulation

2.3.4.1 Agent technology

"An agent is considered a software entity situated in a production environment, with enough intelligence that is capable of autonomous control actions in this environment and of co-operation relationships by participating in association's and agreements with other entities in order to meet its design objectives. An agent should be able to act without the direct intervention of humans or other agents, and should have control over its own actions and internal state" (Schoop et al. 2002). As per this description controller and scheduler systems can be considered agents. This approach is limited by the standardisation and coordination of the agents developed by different suppliers leading to:

- Duplication of functionality
- Lack of standard agent definitions or terms
- Limited reusability
- Limited ability to integrate with legacy systems

The review literature proposes that that the agent approach is not appropriate for production system due to the heterogeneous and dispersed nature of the physical agents in production as well as the requirement for dynamic decision rule capability (Schoop et al. 2002). It is these issues that the transition to holonic systems is attempting to resolve.

2.3.5 Supervisory control and data acquisition systems

Supervisory Control and Data Acquisition Systems (SCADA) are a widely used set of technologies and standards used to collect data from shop floor equipments. They are relatively low level technologies that allow the collection and storage of real time data feeds from control systems which are otherwise considered to be a very disparate technology group with which to communicate (Hak-Man Kim, Jong-Joo Lee & Dong-Joo Kang 2007).Most of the current development on these systems are focusing on developing the security robustness of these systems, which is currently seen as a weakness: IT system priorities are confidentiality, authentication, integrity, availability, and non-repudiation: SCADA systems emphasize reliability, real-time response, tolerance of emergency situations, personnel safety, product quality, and plant safety, usually to the exclusion of any security mechanism that might hinder these (Cesar 2008, Coates et al. 2010).

It has been proposed that SCADA systems can be enhanced through the addition of 'smart substations' which are clients that sit under the SCADA system and provide a flexible interface with the disparate control systems (Hak-Man Kim, Jong-Joo Lee & Dong-Joo Kang 2007). 'Smart' in this context refers to the ability to connect to many different types of equipment using wired and wireless networks. It does not however provide true context to

the data or suggest a framework under which this can be done, therefore the data cannot be acted upon and is not ‘intelligence’.

2.4 Systems lifecycle

2.4.1 Systems lifecycle

A large number of system lifecycles have been reviewed that have been defined by academic groups and consultants. While these were all incentivised to highlight the unique benefits of their model, and use significantly different levels of detail and number of steps in their models, they all conform to the same core process as defined within ISO 15288 (Willcocks, Lester 2003, Degler, Battle 2003, Rafinejad 2007, Taylor 2003, ISO 2002).

- Concept Stage
- Development stage
- Production Stage
- Utilisation Stage
- Support Stage
- Retirement Stage

There are a number of other steps that have been excluded from this summarised list such as post implementation, validation and stakeholder definitions etc as these can be seen as implied in the major steps listed. ISO 15288 provides a standard framework on which specific lifecycles for specific process can be built using these core stages. These core stages therefore represent the most widely applicable definition of the systems lifecycle.

2.4.1.1 Systems growth and complexity

Matt (2007) carries out an extensive review of the nature of complexity, summarising it as: " a complex system is one that has a large number of elements whose relationships are not simple.....the different nature of relations can be considered a driver of complexity". This work goes on to distinguish between: "structural complexity, which is caused by the number of elements and their interconnectedness, and dynamic complexity caused by feedback loops and highly dynamic and nonlinear behavior. Moreover, complexity can be understood as interaction between complicatedness and dynamics". This work was focused on solving the issue of maintaining key functional requirements during organisational growth, but its consideration of an organisation as a system means that many of its key findings and considerations are not directly relevant to other systems. This work proposes that the problem of growth in systems can be resolved by using axiomatic designed networks i.e. structures that are controlled by rules (in this case logical rules), as well as redefining the structure of the system as its growth exceed manageable limits. This holds a direct parallel to the approach of using common logic rule or axioms to structure an ontology about a system (Young et al. 2007, Young et al. 2010, Yu-Liang Chi 2010, Young et al. 2005, Young et al. 2009), with those rules controlling the rate and level of growth before a defined change

is required. The complexity in a system-focused ontology would be driven by the number of elements (concepts) and relationships along with the level of change in the domain which is consistent with the researched descriptions of complexity within this work (Matt 2007). This seems to support the hypothesis that heavyweight ontology could be used to support the development and sustainment of a system while maintaining its functionality and interoperability.

2.4.1.2 Legacy systems

Galliers (eds)(2003)describes how the inheritance of legacy IT and systems can deter investment in new systems. One reason for this is due to resources being consumed by operational and support issues, and the occurrence and severity of these issues will increase with time meaning that organisational and system agility can decay exponentially. This author proposes two ways to address this issue:

- Ongoing evaluation of exiting systems (to identify issues and opportunities and provide both a qualitative and quantitive assessment of the need for action)
- Benchmarking against external comparators (to re-align IT/ systems approach based on 'best in class')

Both of these activities rely on a constant mechanism of comparison and communication between new and old systems within the same domain or across domain (in the case of benchmarking): however, no solution to this issue is proposed.

A systems lifecycle approach has been proposed as a method of ensuring more effective systems development and implementation (Willcocks, Lester 2003). This work references high profile projects such as Taurus in the UK London financial markets, which was abandoned in 1993 at a cost of approx £0.5Bn, as an example of the difficulty of introducing new systems and distils the common core issues to be:

- Project size and complexity
- The 'newness' of the technology
- Level of structure within the project
- Human, political and cultural issues

It can be implied from this research that 'green field' systems implementations (i.e. those with no legacy systems in place) have a higher probability of success than those replacing legacy systems or processes, this is logical given the issues listed above, as the additional requirement to move an enterprise from one system to newer replacement significantly increases the impact of these issues. The research has shown that the post implementation phases of the systems lifecycle are the most neglected, and while a model has been

proposed that uses evaluation stages of subsequent lifecycle iterations to review the implemented systems, this does not provide any mechanism for pre-empting any compatibility issues or even enabling a constant basis of evaluation between the old and new systems. In this way any development or implementation of a replacement system or systems can be viewed as a post implementation iteration of the systems lifecycle for the legacy system.

2.4.1.3 Endurant vs. perdurants

A useful differentiation between manufacturing entities e.g. concepts or properties is whether they are enduring or perduring. Enduring entities (endurants) are not time limited such as the concept of a 'person', whereas perduring (perdurants) require a time description such as 'a person's life' i.e. an endurant would participate in a perdurant (Borgo, Leitão 2007, Masolo et al. 2003). This multiplicative approach would prove useful when describing entities across lifecycles as it allows the description of two entities within the same time-space, rather than two aspects of the same entity e.g. a machine tool and the amount of matter that forms that machine tool. The related topics of 3D and 4D ontologies i.e. whether entities should be described using a temporal dimension was adjudged out of scope as the temporal nature of objects is not the focus of this research.

2.4.1.4 Dynamism and flexibility

Izza (2009) refers to the concept of dynamism as a characteristic of enterprise applications. This is a useful concept as it links together the rate of change of the system with the level of change in the enterprise environment. This work also describes the need for system flexibility as being driven by the evolution of organisations, regulations, IT, business and cost containment which in turn requires the system integration to be equally flexible. This link between organisational evolution, system flexibility and the requirement for flexible integration is important as the conventional route to achieve integration or interoperation requires standardisation which constrains this flexibility (see Chapter 3**)**.

A key driving factor for system flexibility, in the context of this work is the need for flexibility to change or replace the system. This confirms that MI systems which operate at the lowest, fastest paced level of the organisation are themselves subject to fast paced change i.e. the development or replacement of systems. The industrial investigation (see Chapter 3) has shown that the difficulty of maintaining interoperation between systems during change often constrains the level of systems change possible. Izza's proposed solution must therefore be able to deal with fast paced change of systems while still enabling system design flexibility, which effectively rules out conventional approaches which rely on conformance to an explicit standard or framework.

2.4.1.5 Lifecycle timescales

The timescales for a systems lifecycle will depend on many factors, a simple example is the computer hardware required. Personal computers can be considered to have a maximum reliable life of 10 years. If the system is critical or operating in a harsh environment then this may be halved or quartered. The practical difficulties of ensuring older system work on newer hardware and operating systems means it is generally advisable to upgrade the system to a current version at the same time (Wiles 2002). In this example it can be seen that the environment in which a system operates can significantly reduce the system refresh lifecycle timescale and hence the frequency of system replacements. The fast evolution of information and communication technologies is causing new types of problem for integration and interoperability (Panetto, Molina 2008).

Software and systems tend to be designed around the capabilities and limits of the machines and data available at the time, and as these develop over time so solution providers develop system with more features and functions, and hence the accelerating pace of technology is reducing the system lifecycle timescale(Degler, Battle 2003).

An often trivialised issue affecting IT lifecycles is ‘fashion’ (Wang 2010). Capitalising on trends and fashions in IT and Management is vital for systems suppliers and consultants, whilst the ability to globally adopt systems once initial pilots or testing is proven successful is equally important to the customers in order to maximise the benefits they bring. The reviewed work shows that there is a negative benefit in the 1st year of adoption for early adopters which becomes a neutral affect in Year 2 moving to positive effect in the 3rd year (Wang 2010). This can be linked to the costs and complexity of early lifecycle adoption and so IT and systems suppliers as well as customers are striving to accelerate systems through the early lifecycle stages to minimise this effect. This in turn acts as an accelerating effect on the lifecycle iteration rate.

2.4.1.6 Application lifecycle

Application lifecycles (as there a many types) can be considered a lifecycle within, and similar to the systems lifecycle. Tradition 'waterfall' methods (analyze, design, code, test) have been replaced by the iterative Rapid Application Development (RAD) approach. The iterative, agile approach tend to be beneficial when the requirements are not well defined or are particularly complex, whereas the waterfall approach may be more appropriate where there is a requirement for a greater integration with current applications. Where the application is a development of extension of an existing application an enhancement lifecycle may be appropriate (Mochal 2005).

2.4.1.7 System value and condition assessment

Leading organizations periodically look at classes of applications and plot them based on their business value and technical condition. A suitable format for high level portfolio analysis can be conducted as shown below. This is a Boston Consulting Group type approach which was proposed for application analysis however it is equally applicable as a systems level as shown in Figure 2-2. (Maizlish, Handler 2005):

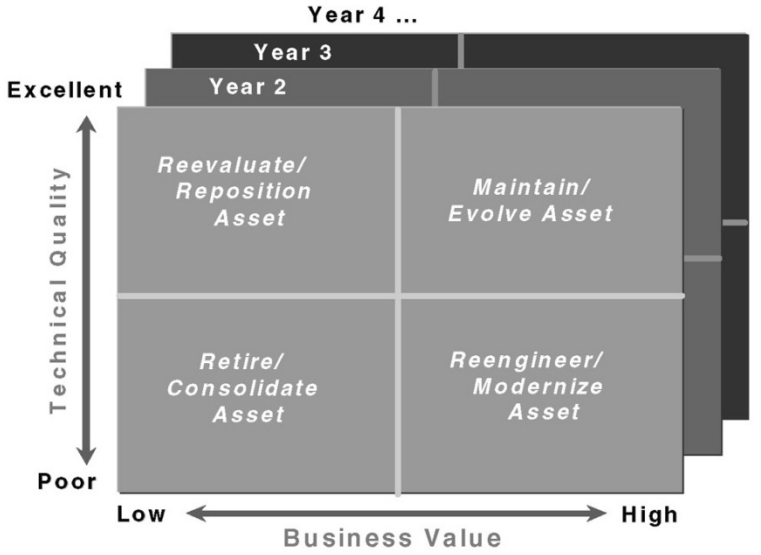


Figure 2-2 - View of technical quality and business value for applications (Maizlish, Handler 2005)

- Systems ending up in the lower left quadrant have low business value and are in poor technical condition. They are candidates for retirement, consolidation, or replacement.
- Systems in the lower right quadrant have high business value but are in poor technical condition. They are candidates for reengineering and modernization.
- Systems in the upper left quadrant are in excellent technical condition but have low business value. They are candidates for re-evaluation and repositioning.
- Systems in the upper right quadrant have high business value and are in excellent technical condition. Counter to conventional wisdom, these applications are still worth worrying about. They must be carefully maintained and evolved; otherwise, they will end up in the lower left quadrant over time.

This analysis still relies on a qualitative assessment process leading to consistency problems regarding the interpretation of the systems key concepts and performance.

2.5 Information sharing

2.5.1 Data, information and knowledge

Data is generally well understood as being words, numbers, pictures, the meaning of which may change with context (Young et al. 2005), and information being data in context so it has a particular meaning. Generally, knowledge defines the relationship between information within or across contexts (Gunendran, Young 2006, Young et al. 2005). Knowledge can also be considered to consist of relations between facts and decisions (Liu, Young 2007).

There is some difference of opinion on the exact nature of knowledge, albeit the underlying ethos is common e.g.:

- Knowledge is generated when information is combined with context and experience (Huang, Yang & Wang 1999)
- Knowledge, coupled with experience, leads to wisdom (Maizlish, Handler 2005)

High-quality data and information provide organizations with knowledge to enable effective actions and decision-making. Research indicates that 90% of all business decisions are suboptimal because of data quality. Ironically, the biggest data quality complaint does not pertain to the accuracy of the data but the completeness of the data. Incomplete data translates into incomplete information, which leads decision makers to rely on intuition with greater frequency than desired. Most organizations do not know what data or information they have. They have no idea about the value of their data or information. Most organizations are aware that their data are important, valuable, and imperfect (Maizlish, Handler 2005).

The quality of information depends on its accuracy and its relevance to the context. Some valuable information management approaches are being explored in Knowledge Management disciplines including (Degler, Battle 2003):

- Granularity of information-both in terms of storage and presentation
- The importance of metadata, particularly as it they relates to context
- Increased reusability of information, including the creation of "knowledge objects"
- Information structured in ways that move away from the 'document' and 'page' metaphors

2.5.1.1 Data, information and knowledge maintenance

Enterprise knowledge is dynamic, and so a knowledge base quickly becomes obsolete if it does not continue to incorporate new information quickly and does not have a mechanism for removing out-of-date or contradictory information (Degler, Battle 2003).

Any knowledge, information or data base requires maintenance, both of the systems and the contents. The design of any interoperation mechanism or process must not compromise this ability and ideally should enhance it by providing a unique authoritative source for any data type. Change control in an expensive interoperable system can be highly complex as the systems need to provide suitably rigorous but agile change mechanisms for a vast array of data types i.e. from orders, finance, technical product definitions and personnel details to local communication memos and work instructions (Young et al. 2007, Hodges 2007).

Structuring information and knowledge around products or product families such as the Product Range Model approach for variant product design, provides a mechanism for data re-use and could potentially allow the use of heuristics for knowledge maintenance as new designs ideas are accepted. (Costa, Young 2001)

2.6 Interoperability

An essential attribute of interoperability is seamless information transfer in all directions (Jr., Michel 2000). This work provides a summary of different aspects of an enterprise that can be integrated or interoperated. However this work does not mention the requirement to integrate systems through time (ie between old and new systems).

2.6.1 The Requirement for interoperability

Significant levels of work have been published on the requirement for interoperability as an enabler for collaboration. This work has covered many perspectives and areas of focus with some of the most relevant to this work being:

- Collaboration between Design and Manufacturing Functions (Gunendran, Young 2006, Young et al. 2007, Chungoora, Young 2010a, Chungoora, Young 2010b, Young et al. 2005).
- Collaboration across Enterprises (Panetto, Molina 2008, Chen, Doumeingts & Vernadat 2008, Lin, Harding 2007)
- Product Support Services Interoperability Requirements (Meier, Roy & Seliger 2010)
- Design and Product Lifecycle Information Exchange (Young et al. 2009, Kim, Manley & Yang 2006, Choi 2010, Skarka 2005, Kyoung-Yun Kim et al. 2009), Design Knowledge re-use (Costa, Young 2001)
- Supply Chain Integration (Yu-Liang Chi 2010, Deshayes, Foufou & Gruninger 2007, Chun-Che Huang, Shian-Hua Lin 2010, Taghaboni-Dutta, Trappey & Trappey 2010).
- Manufacturing Integration, Data Management, Decision Making and Knowledge Sharing (Young et al. 2010, Unver 2012, Chen, Chien 2011, Panetto, Molina 2008, Qiao, Liu 2009, Liu, Young 2007, Guerra-Zubiaga, Young 2008, Alsafi, Vyatkin 2010, Xiong et al. 2010, Zaremba 2003, Vincent Wang, Xu 2013)
- Specific industry examples such as Healthcare (Zhang, Xu & Ewins 2007), Automotive (Guo Wen-yue, Qu Hai-cheng & Hong 2010, Blomqvist, Öhgren 2008), Furniture Manufacture (Agostinho et al. 2009) and E-learning (Lagos, Setchi 2007).

The common themes within all of this work are:

- Interoperability between functions, domains and systems is an increasing requirement which is delivering massive benefits wherever it is achieved.
- The recognition that common basis of sharing meaning (both processes and systems) is a fundamental prerequisite for unambiguous communication.
- While standards have a part to play in allowing shared meaning, they require interpretation which leaves them prone to semantic miscommunication.

- There are a large number of approaches to achieving interoperability many of which focus on different aspects of the challenge.
- The use of an abstract or common model defining concepts, objects, rules and relationships, which are mapped to real world specific instances. Common focuses of this work are:
    - Using the common model to create conformal instances
    - Using instances to define a common model
    - Using existing instances and a common model to define rules or relationships.
- The recognition of the computational complexity of semantic reasoning and the significant time and effort required to represent the evolution of dynamically changing data (Kyoung-Yun Kim et al. 2009).
- ICT based solutions have traditionally resulted in systems which are limited in scope, complex, burdensome and inflexible to change (Young et al. 2007).

MESA International have published literature on a framework concept they refer to as 'P2E' or 'Plant to Enterprise' which recognises the importance of integration or interoperation of the shop floor and enterprise level systems within an organisation to enable better decisions e.g. regarding improvements etc (MESA 2012). This literature highlights the timescale related challenge of interfacing large enterprise system with dynamic shop floor system. It references the limitations of singular interface standard approaches such as middleware systems and proposes a loose, developing framework of approaches that might be used, but falls short of proposing a solution to this issue.

2.6.2 Integration vs interoperability

Interoperability can be described as the ability to exchange and use information. IEC standard IEC/65/290/DC describes it as the level of compatibility (Izza 2009). The IEC standard classifies the levels of compatibility as:

- Incompatible
- Coexistent
- Interconnectable
- Interworkable
- Interoperable
- Interchangeable.

Interchangeability includes the concept of interoperability and requires that the semantic and application functionality is defined such that should any system be substituted it will continue to operate albeit possibly with different dynamic responses (Izza 2009).

The work by Izza (2009) provides a thorough review of Integration approaches, which reviews much of the same material as the work reviewed on Interoperability Frameworks (see Interoperability Frameworks section). This work refers to the risk of confusing interoperability and integration; however, seems to occasionally blur the line between these concepts in its review. This work resulted in a useful 2D evaluation framework of integration techniques and tools using an axis with a scale form syntactic to semantic and the other axis static to dynamic.

2.6.2.1 Integration continuum

Panetto, Molina (2008) states that integration is generally considered to 'go beyond mere interoperability' to involve some level of function dependence. By this definition integrated systems must be interoperable but interoperable system may not be integrated. Interfacing can be considered a level of integration below interoperability. The implied description of fully integrated systems as being superior to interoperable systems does not take account of the issue of robustness; by this definition and failure of one system integrated with another is likely to result in a failure of the integrated systems, while interoperable systems are likely to be more robust.

Panetto, Molina (2008) goes on to provide a useful comparison of some key integration frameworks which can be summarised as:

- Levels of Information System Interoperability (LISI): Focuses on technical interoperability and the complexity of interoperation in 5 stages (Isolated, Connected, Functional Distributed, Domain Integrated, Universal)
- Organisational Interoperability Maturity (OIM): Extends the LISI model into the areas of Organisational Interoperability in 5 stages (Independent, Ad-hoc, Collaborated, Integrated, Unified)
- Levels of Conceptual Interoperability Model (LCIM): Extends the technical model of LISI in the area of Conceptual Interoperability in 5 stages (System Specific, Documented, Aligned Static, Aligned Dynamic, Harmonised)
- NATO C3 Technical Architecture (NC3TA): Focuses on the level of content structure in 4 levels (Unstructured data, Structured data, seamless data sharing, and seamless information sharing)

- European Interoperability Framework (EIF): This framework shows the progressive levels of challenge that must be overcome to achieve increasing levels of interoperability (Technical, Semantic and Organisational)

The review aligns these frameworks and allows the conclusion that to achieve organisationally integrated and seamless information sharing that domain integration and dynamic alignment of concepts must have been achieved which requires the Technical and Semantic challenges to have been overcome. To achieve unified information sharing the concepts must be harmonised, the technical interoperation must be universal and the organisational challenges must also be overcome.

2.6.2.2 Taxonomy

A major obstacle in information exchange is the differences in taxonomies used by various members of a supply chain. The work of Taghaboni-Dutta, Trappey et al. (2010) proposes an XML based framework for a supply chain integration hub that resolves this issue using an XML schema for interfacing. This work proposes that it resolves the interoperability issue, but it can be seen that while it only provides an advanced level of interfacing or integration, the data structures and hierarchies alone do not resolve the potential semantic inconsistencies which will prevent interoperability.

2.6.2.3 Systems and technology

Even if an enterprise uses current technology there is still complexity associated with interoperation between systems' competing software tools (Young et al. 2007).

A number of large enterprises have tried to overcome the challenge of integration or interoperation across systems by insisting that their suppliers use the same tools. This only results in the issue being pushed further down the supply chain, onto suppliers who are likely to be dealing with other enterprises who will use different systems. This issue is significant, and within the Automotive Industry alone the impact has been assessed as being in the order of $1Billion (Young et al. 2009, Borgo, Leitão 2007).

Web based technologies are improving global access to IT support solutions, they generally still focus on discrete areas of the business or systems, and in themselves do not have the capability to resolve the interoperability issues. Attempts to create inter-domain integrated systems have required significant overheads and result in inflexibility hence is not currently practically supported (Young et al. 2007).

The Information and Communication Technology requirements for interoperability have been identified as a key challenge by NIST, INTEROP and work by a number of authors. This work has also identified that the traditional approaches to integrated information sharing

have fallen short of the requirements for enterprise level decision making (Young et al. 2009) as do the approaches using interfacing systems or middleware (MESA 2012).

2.6.2.4 Information systems customisation for integration

The review by Izza (2009) of information systems integration provides some useful categorisations of systems which will need to be considered when defining an integration approach e.g.:

- Black box – unavailability of code or interfaces
- Grey box – unavailability of code but availability of interfaces
- White box – availability of code and interfaces

This work draws the important conclusion that even if the code is available it is generally inappropriate to modify it, advocating the use of application wrappers or interfacing systems. It can be inferred that any code/ system customisation would reduce the ability to migrate from one application to another.

2.6.2.5 PLM

The conventional description of PLM as "a strategic business approach for the collaboration, creation, management and dissemination and use of product definition information" (Anon 2008) does not fully represent the current developments in PLM Design and Manufacturing Information and Knowledge Management such as (Young et al. 2007, Chungoora, Young 2010a, Young et al. 2009, Liu, Young 2007, Choi 2010) which are developing wider information models and capabilities covering all aspects of the process and resources required to operate the enterprise. Much of the work reviewed in the literature review is either directly or indirectly contributing to this field.

2.6.2.6 PLM information exchange

Product Lifecycle Management tools are a key tool for product data and data storage and exchange. PLM systems have evolved from the conventional CAD and PDM tools hence much of the development on this toolset has focused on the Design context of the lifecycle and the management of product, process and resource information.(Young et al. 2009, Choi 2010). While these systems provide a method of information organisation and management they have significant limitations in terms of representing knowledge. The tight configuration control requirements of PLM systems do not fit well with the requirements of interoperable collaborative engineering (across domains or multiple system) (Young et al. 2010).

2.6.3 Standards – semantic issues

Deshayes, Foufou et al. (2007) describe standards as a "consensus on the semantics of terms" going on to describe how standards will have different semantics for the same

concepts and will express them in different ways. The multiplicity of standards is therefore highlighted as an integration challenge. This work goes on to describe an ontological approach to standards integration, which could also be used to integrate separate domain ontologies.

Syntactic integration using standards such as XML, ebXML or RossettaNet use (implicit) informal semantics (Izza 2009) and therefore are subject to the issues of inconstant semantic interpretation.

Young, Gunendran et al (2007) provided a compelling example how the same fundamental term such as 'Process' defined in several standards (ISO/CEN 19439, ISO 15531-1:ISO 18629-1, ISO 10303-49) demonstrating the lack of rigour when using a solely standards based approach.

2.6.3.1 Standards frameworks and adoption

In order to align product data, business and technology activities and enable interoperation, standards are a must. Specific standards are often defined within the framework of larger standards, an example being the FUNSTEP standard (ISO 10303-AP236) which is defined within the STEP group of standards (Jr., Michel 2000, Agostinho et al. 2009). This standard has been declared as a possible foundation for data exchange; however, it has not been widely adopted.

Having attempted to overcome the issues relating to defining a workable standard the next challenge is to manage the adoption of and compliance to that standard. In a diverse enterprise supply chain many of the other challenges described in this section will lead to conflicting attitudes and approaches to these standards because even if the standard is adopted, unless it is implemented in a uniform manner variation can result. A number of authors have described these issues related to the adoption of standards (Lin, Harding 2007, Choi 2010, Deshayes, Foufou & Gruninger 2007) as well as the issue of having to chose from conflicting standards (Jr., Michel 2000).

2.6.3.2 Standards adaptation

ISO 10303 AP224 fundamental concepts were used in the work to define the Semantic Manufacturing Interoperability Framework, which were adapted and formalised as part of the research (Chungoora, Young 2010a). This raises the questions of how far a standard can be 'adapted' while still being a generically applicable standard, and the risk introduced if other users adapt the same standard in a different way (i.e with the two adaptations be mutually compatible?).

Annex A of ISO/EIC 15288:2002 provides guidance on the tailoring of the international standard to allow its adoption in a variety of circumstances (ISO 2002).

It is common for standards to be built upon or 'extended' (Deshayes, Foufou & Gruninger 2007). These extensions are generally not limited to the use of the base standard's ontology meaning these domain or application specific extensions cannot be aligned or linked in the same way that domain ontology based on a common core (or standard) ontology can be (see Foundation Ontology section).

2.6.3.3 Standards flexibility

The standards used within the enterprise, be these design standards, communication standards, architecture standards etc, which are usually enacted to improve commonality and interoperability can result in organisations and functional inflexibility and an inability to adapt to new requirements. Any global standard that is defined would need to be suitably comprehensive in scope and coverage to accommodate any requirements from any function within the enterprise. Defining such a standard would be impractical, as its comprehensiveness would render the standard itself inflexible; compounding the constraints any standard applies to an organisation. The larger the organisation or data model the more the more likely it is that the standard coverage will be incomplete (Lin, Harding 2007). A one-by-one representation of information according to defined standards is not a simple job, and so is not worth the enormous cost and time needed to conform to the standards (Choi 2010).

2.6.4 Enterprise integration and interoperability

A definition of an enterprise in ISO 15704 (Chen, Doumeingts & Vernadat 2008) is given as "one or more organisations sharing a definite mission, goals and objectives to offer an output such as a product or service". This work goes on to summarise the difference between Enterprise Integration and Enterprise Interoperability as being the degree of coupling. If systems are tightly coupled, displaying interdependence, coherence and uniformity, they can be considered integrated. Interoperability, however, is characterised by looser coupling, coexistence, autonomy and a federated environment. The approach to interoperability is defined according to ISO 14258 as being:

- Integrated : where there exists a detailed, common, agreed format for all models
- Unified :where semantic equivalence is enabled by a common meta-level structure across constituent models
- Federated' :where there is no common format and models must dynamically accommodate (through a shared ontology) rather than having a predetermined meta-model.

2.6.4.1 Integrated, unified and federated approaches

As described in (ISO 1999) the approaches to interoperability can be categorised as:

- Integrated - With integrated models there is a standard model form. Diverse models are interpreted against the common template. Standard or reference models must be as rich as the constituent models. All models can be stored in standard form with information filtered or translated by the applications. Alternatively, standard models can be agreed to by constituent model owners, as in STEP. There are enormous difficulties associated with standardizing large numbers of models. Therefore, the ability to deal with something less than integrated models is most certainly necessary.
- Unified - The unified approach assumes that there exists a template that provides a common meta-level structure across constituent models, providing a means for establishing semantic equivalence. This template is then the basis for a meta-model. The meta-model is not in an executable form as it is in integrated situations. Using the meta-model, any model can be translated into any other. Loss of some semantics is possible. Normalized semantics is established by owners of constituent models.
- Federated - The federated model scenario exists if one assumes that no agent can impose requirements for semantic equivalence across all models of an enterprise. Models must be taken as encountered. The template is at the meta level and, as in the unified situation, the template is not executable. Some degree of integration or unification would help the communication process. Interoperability requires that models be dynamically accommodated rather than having a predetermined meta-model. This would be furthered with some sort of predetermined terminology system. Definitions are a particularly difficult problem for process interoperability when the namespace contains federated models.

Many different approaches and programs of work have been reviewed (Chungoora, Young 2010a, Panetto, Molina 2008, Izza 2009, Young et al. 2005, Young et al. 2009, Chen, Doumeingts & Vernadat 2008, Lin, Harding 2007, Zhang, Xu & Ewins 2007, Usman 2012, Shen, Carlson & Peter Tarczy-Hornoch 2009, Latif, Boyd & Hannam 1993, Ray, Jones 2006) and the general conclusions are:

- The Integrated approach is currently widely used, with its associated limitations of significant resource requirements and flexibility etc.
- The majority of work on interoperability solution development is following a Unified approach. While the details of the frameworks and modelling approaches differ, the common concept of meta-model definitions supported by formalised ontologies is

widely replicated. This approach is still generally in the concept and demonstrator development stage, and while credible routes to implementation have been proposed, implementation levels are still limited.

- The ISO (ISO 1999) description of the Federated approach is consistent with the findings of the reviewed research: this approach gives rise to significant ontology problems. The difficulties of ontology building, mapping and verification etc are discussed within the literature review, and therefore the problems arising in an approach where there is no semantic standard exacerbate the complexity and workload of any resolution approach. While Federated approaches have been described they have been declared impractical within the standard, and the Unified approach the preferred option.

2.6.5 Integration architecture

Architectures for integration and interoperation have been discussed in detail by Chen, Doumeingts et al. (2008). While a number of definitions for 'architecture' are offered, the most relevant is "the structure of components, their inter-relationships and the principles and guidelines governing their design and evolution over time". This definition is carried out at a high level of abstraction, allowing representation of the system in terms of features rather than detailed requirements on functions, data and resources. This work compares the key research on enterprise architectures such as the Computer Integrated Manufacturing Open System Architecture (CIMOSA), the Purdue Enterprise Reference Architecture (PERA) etc. The authors describe the key deficiency in the development of Interoperability Architectures as being one of the lack of a standard ontology for concepts, relationships and properties of enterprise architectures, which is common across other aspects of interoperability fields of research.

2.6.5.1 Model driven architecture

The Object Management Group (OMG) has proposed a Model Driven Architecture (MDA) using hierarchical common concept models which are defined by Meta and Meta-Meta models in turn. These models are (in descending hierarchical order): Computation Independent Model (CIM), Platform Independent Model (PIM) and Platform Specific Model (PSM). The CIM considers the environment and requirements without the detailed structure being defined), the PIM considers the operation of a system without focusing on requirements of any specific platform and the PSM takes the PIM and considers the requirements These common concept models can be mapped and transformed to each other using morphisms (either altering or non altering) or another transformation tool (Young et al. 2010, Young et al. 2009, Agostinho et al. 2009).

Chen, Doumeingts et al. (2008) described work which has been carried out within the framework of INTEROP and ATHENA on Model Driven Interoperability (MDI) based on MDA and enterprise interoperability concepts. This aims to allow automatic transformation between the CIM, PIM and PSM model levels and allow interoperation between models within different enterprises which use the same MDI architecture.

This MDA approach has been adopted by the Interoperable Manufacturing Knowledge Systems project (Young et al. 2010). As part of this work they have provided a compelling argument that where multiple PIMs are used there is a need for a heavyweight ontology which underlies these models as well as an ability to evaluate the differences in concepts across domains to verify the extent to which they are sharable.

2.6.5.2 Interoperability and information exchange frameworks

"The main purpose of an interoperability framework is to provide an organizing mechanism so that concepts, problems and knowledge on enterprise interoperability can be represented in a more structured way" (Jr., Michel 2000, Chen, Doumeingts & Vernadat 2008). Chen, Doumeingts et al. (2008) details the work carried out to develop interoperability frameworks. These frameworks show different ways of describing and categorising interoperability. This research describes how the work of various groups e.g. IDEAS, C4ISR AWG, ATHENA, NEHTA, has been built upon by the INTEROP Network of Excellence (NoE) framework. This framework appears to be a well rounded structure which is defined in 3 dimensions:

- Interoperability Barriers : these are categorised as Conceptual (syntactic and semantics), Technological and Organisation Barriers as per the E-Health (NEHTA) framework
- Interoperability Concerns (Enterprise Level) : Data, Service (application), Process (sequence of services), Business as per the ATHENA work
- Interoperability Approaches: Integrated, Unified, Federated as per ISO 14258 (ISO 1999).

This framework defines a useful three dimensional enterprise interoperability domain within which tools and activities can be categorised and compared.

Similar work has been carried out defining common knowledge frameworks to enable collaboration or interoperability(Liu, Young 2007). This work classifies the information and knowledge used to support manufacturing decision making into 3 domain specific knowledge models:

- Product Model

- Manufacturing (capability) Model
- Order Model

Two variants of each model are described: Local and Global. This work goes on to focus on the relationships within and between the models at the same and different organisation levels and is proposed for discrete decision making support. This work uses standardisation and explicit definitions to provide unambiguous structure, but as previously discussed this comes with a very high construction and maintenance overhead, which makes real world operation impractical.

It has been demonstrated that the product model approach is limited in its ability to capture the multiple viewpoint representations and the knowledge of relationships between such viewpoints (Gunendran, Young 2006). It is reasonable to infer that the other 2 models will be equally limited unless this multi-modal requirement is addressed (i.e. the ability to accommodate multiple domains). A method of achieving integration (i.e. not Interoperability) between product design viewpoints is proposed in this work using separate product information and knowledge layers and knowledge links, however there is a need to reduce the complexity of these links before this approach (or similar) could be expanded to other contexts and lifecycle stages.

2.6.5.3 Semantic interoperability frameworks

The use of frameworks based on semantic interoperability is an approach emerging from several areas, such as the ISO standards community (ISO TC184/SC4) (Young et al. 2010, Chungoora, Young 2010a). These papers propose the Semantic Manufacturing Interoperability Framework (SMIF) and the IMKS project. This work is built upon some of the strongest emerging ontological approaches i.e.:

- It uses common logic (CL) based ontological formalism
- It uses a heavyweight foundation ontology defined on the CL formalism
- It uses this foundation ontology to define domain specific ontologies

This approach provides more formal mechanisms for the capture of knowledge by starting from a low level of abstraction (such as GD&T semantics for product features). Heavyweight methods were used to provide interoperable knowledge sharing between domain models. This work was limited to aspects of the design and manufacturing phases of the (product) Lifecycle i.e. the complexities of applying this approach across multiple lifecycle phases with their multiple domains were not in their scope.

The Agile Manufacturing Data Management approach (Qiao, Liu 2009) displays a similar approach to those previously discussed in that it uses a multi-layered architecture which can be summarised as:

- 1st layer: Core concept and objects in a domain ontology (databases)
- 2nd Layer: Meta models describing the relationships between the objects (service components).
- The core concept and meta models together define a unified manufacturing data model combining Manufacturing Management, Enterprise Resource and Product & Process Information models.
- 3rd Layer: Service Layer, providing application Service Applications (web based)
- 4th Layer: Business Level services

The 3rd and 4th Layers of this model are predominantly detailed as part of the description of the use of different technology, but Layers 1 and 2 are an attempt (albeit loosely defined) to address the same issues as the more detailed MSIF

2.6.6 Ontology

It may be considered ironic that there are a number of definitions of an ontology such as that provided in (Noy, McGuinness 2001, Li, Yang & Ramani 2009). The definition that is used within this research is: 'a basis for shared meaning' . A lack of practical ontology results in semantic miscommunication: where the same term is used to define multiple concepts or objects and where multiple terms are used to define the same concept or object) (Young et al. 2007). Shared definitions of objects and concepts themselves do not ensure effective unambiguous communication, as this requires a wider lexicon of terms to enable the detailing and exchange of meta data and emerging data, information and knowledge types (Liu, Young 2007). An example of this ambiguous ontological interpretation is the understanding of what 'process' information may be; the work reviewed seems to use a generally common understanding of this information to be the process steps, activities and their relationships and requirements, however this could equally refer to Manufacturing Intelligence (MI) dynamic information, although there is little evidence of MI information requirements being included or accommodated.

Noy, McGuinness (2001) highlights the contradictory definitions of ontology (particularly within Artificial Intelligence literature) and proposes a clarification which appears to be strongly related to object based programming conventions:

- Ontology – Is a formal explicit description of concepts in a domain

- ‘Concepts’ are also sometimes called ‘classes’ which can also be divided in sub-classes
- The ‘roles’ or ‘properties’ of classes are defined within ‘slots’
- The restrictions on slots are referred to as ‘facets’ or ‘role restrictions’
- An ontology along with individual instances constitute a knowledge base (no clear definition of where a ontology ends and a knowledge base begins is provided)

The use of shared ontology has been proven to aid in the persistent representation of information across heterogeneous and design collaborations. Kim (2006) detailed a review of the development of Ontologies. Related tools such as PSL (Process Specification Language) and the semantic web describe how they were initially too high level to be practically applicable but have become increasingly detailed . This works goes on to propose an ontology based product development collaboration model but it does not address the issues of security or automatic ontology construction. It does show how structuring concepts using standard semantic terms and defined relationships provides the capability for computers to infer relationships for specific model instances.

Projects that operate within inter-enterprise environments or across multiple domains face the issue of different information models being used by different functions or areas. The local teams will develop models, vocabulary and terms that most effectively address their specific needs. While a standard vocabulary will help individuals within a domain to communicate it will not support flexibility from outside this domain due to the issues discussed in the ‘Standards’ section (Lin, Harding 2007).

2.6.6.1 Ontologies and interoperability

A number of authors have included useful reviews of the history and development of the field of Ontologies within the context of interoperability (Noy, McGuinness 2001, Young et al. 2007, Borgo, Leitão 2007, Kim, Manley & Yang 2006, Kyoung-Yun Kim et al. 2009, Li, Yang & Ramani 2009). However a suitably generic set of terms for evaluating an ontology have been proposed (Borgo, Leitão 2007, Blomqvist, Öhgren 2008):

- Expressiveness of the language (glossaries, natural/ formal languages)
- Purpose of the ontology (terminological ontologies, information ontologies, Knowledge sharing; where each level adds complexity)
- The domain covered (manufacturing, design, management, finance)
- Structural complexity (Degree of branching, depth of the hierarchy)

A number of manufacturing information systems have used ontologies to describe the structure an relationship between domain intra-organisational concepts e.g. MOSES, CIMOSA and MISSION (Lagos, Setchi 2007).

The reasons for creating an ontology have been summarised by Noy, McGuinness (2001) as:

- To share common understanding of the structure of information among people of software agents.
- To enable reuse of domain knowledge.
- To make domain assumptions explicit.
- To separate domain knowledge from the operational knowledge.
- To analyze domain knowledge.

The authors go on the describe these points in detail. These descriptions emphasise the relevance of ontology based knowledge sharing to providing consistency of understanding and enabling clear representation of domain knowledge.

2.6.6.2 Ontology creation

Noy, McGuiness (2001) describe a manual approach to defining an ontology. This approach is predicated by the assumptions that there is no one correct way to model a domain (as it depends on the intended use of the model). Ontology creation is an iterative process and the concepts in the ontology are likely to be nouns and verbs (describing objects and relationships). The defined approach relies on the creator of the ontology focusing on the most relevant concepts and properties, and using this focus on relevance to offset the inevitable incompleteness of an ontology. The approach is summarised in the following steps:

1. Determine the domain and scope of the ontology – The intended use of the ontology must be understood to ensure the resulting model is relevant. Techniques such as ‘Competency questions’ can be used to this end.
2. Consider reusing existing ontologies – Many ontologies have been defined (libraries of reusable ontologies are available) and many of them are in tools which allow them to be exported.
3. Enumerate important terms in the ontology – Writing down a list of all key terms, properties or statements to be used by or with the ontology in a ‘brainstorm’, focusing on being as comprehensive as possible while maintaining relevance.
4. Define the classes and the class hierarchy – Using either a top-down, or bottom-up or a communication of both the classes are organised into a hierarchical taxonomy.

5. Define the properties of classes – Intrinsic (e.g. flavour), extrinsic (e.g. name), parts (if the object is structured) and relationships are just some of the types of properties that can be defined within ‘slots’.
6. Define the facets of the slots – These describe the constraints on the slot values such as: number of values (cardinality), slot value type (e.g. string, number, boolean, enumerated (defined list) and instance) domain and range of a slot.
7. Create instances – populating the classes.

This work provides a significant level of detailed guidance on the structuring, process, nomenclature and logical considerations in ontology construction which summarises much of the legacy research and provides a very useful framework to follow.

Blomqvist, Öhgren (2008) Li, Yang et al (2009) provided a useful review of manual and automatic techniques for ontology creation. For manual methods the key factors compared were the level of definition, lifecycle coverage and level of reuse of existing ontologies. The outcome of this work can be summarised as:

- Automatic approaches provide a structured result but will not capture the most specific concepts without further development. This approach relies on pattern recognition and so is limited by the level of structure and pattern that is available to find.
- The manual approach gives a less structured result with less complex relations and axioms, but it captures the more specific and relevant concepts (corroborated by the work of Noy, McGuiness (2001)).
- A combination of the approaches in the future therefore appears the logical conclusion with manual being most appropriate for the high and low level of the ontology and automatic then being used to populate the middle levels.

The last point is corroborated by the findings of Li, Yang at al (2009).

Work has been published on the automation of the merging of separate ontologys, whether that is generic, domain, application or service ontologies (Wang et al. 2007).

2.6.6.3 Ontology merging

The ontology merging work of Wang et al (2007) was built on the use of OWL-S (as a non-natural language descriptor), Wordnet (as a disambiguating knowledge base) and its own ontology merging algorithm. This work reviews and discounts other approaches to this issue due to their requirement for manual input, limitation to lightweight ontologies and insufficient knowledge base development. This technique relies on the imported ontologies being fully defined within their own domains. In practice it is unlikely that the separate ontologies will be

fully documented as the work to document them all to generate a global ontology would be impractical.

2.6.6.4 Mapping

Mapping has been used as a method of integrating multiple viewpoint representations (Gunendran, Young 2006). The reviewed work highlights the limitations of mapping and translation, as they are limited to mapping between two models or viewpoints, and in the case of translation, are unidirectional. It also refers to other work that is looking to combine these approaches with agent based and ontology based approaches.

2.6.6.5 Ontology mapping

Ontology mapping has been proposed as a method of reconciling semantic structures across multiple domain ontologies (Chungoora, Young 2010a). The approach proposed in the SMIF uses logic/ rule based mapping, it is postulated that this resolves some of the current limitations of existing ontology mapping frameworks, although it does not mention explicitly which.

Ontology mapping concerns the establishment of ontology links (or mappings) between two ontologies. This means that for each ontology entity (concept, relation, attribute, etc) a corresponding entity in the second ontology is attempted to be found, with the same or the closest intended meaning; usually this correspondence is expressed by 1 to 1 functions (Izza 2009). This work provides a detailed review of mapping and matching techniques, finding mapping to be a key field of research for integration technologies.

2.6.6.6 Concept mapping

Concept mapping between supply chain members, using a XML schema has been proposed (Taghaboni-Dutta, Trappey & Trappey 2010) however it requires that those wishing to use the system must create a custom mapping interface to map their unique XML forms to a defined standard i.e. the interface uses the XML structures of the source and target files to create a visual structure for the user to manually assign mapping rules which are saved for future use. This system does not address semantic inconsistencies, and while the mapping rules a relatively simplistic, this could cause erroneous mappings for different uses of the same terms.

2.6.6.7 Heavyweight and lightweight ontologies

There are two commonly used categories of ontology: ‘Lightweight’ and ‘Heavyweight’. Lightweight ontologies require some level of interpretation leaving some opportunity for miscommunication e.g. a dictionary: the terms are explicitly defined but this is using a common but unscientific language. Lightweight ontologies allow shared discussions, but are not rigorously enough defined to allow systematic or automatic communication, action and

inference. Heavyweight ontologies by definition provide this capability and therefore share some basic structural requirements with machine code or programming languages. A heavyweight ontology uses formal (machine interpretable) logic in the form of axioms to avoid the terminological and conceptual ambiguities due to unintended interpretations (Borgo, Leitão 2007).

The complex topic of creating heavyweight ontology to enable systems integration (and their benefits over lightweight approaches) is the subject of significant published and ongoing review and research (Young et al. 2007, Young et al. 2010, Young et al. 2005, Young et al. 2009, Usman 2012, Usman et al. 2011). The reviewed literature consistently states that lightweight ontologys lack the rigour required to prevent ambiguity or the requirement for manual interpretation. It is also apparent that much of the work on heavyweight ontologies has been carried out for high level or foundational ontologies meaning they have been difficult to practically apply. This was overcome with the development of the Core Concept Ontology (Usman 2012, Usman et al. 2011).

2.6.6.8 Foundation and core concept ontologies

A ‘Foundation Ontology’ is one which is used as a general basis for building further domain specific (‘core’) ontologies (Borgo, Leitão 2007). Research in linguistics has suggested that as few as 60 semantic primitives are adequate to construct a very large number of concepts (Cassidy 2008). Work to define the ADACOR manufacturing core ontology highlighted the difficulties to build, maintain and modify proprietary ontologies to be used by heterogeneous manufacturing control applications, especially those built upon distributed approaches such as multi-agent systems. This problem pushed for new approaches in the development of manufacturing ontologies to simplify the effort to build, maintain and modify the ontologies. Following a review of the alternatives, the adoption of an established first order logic based foundational ontology (DOLCE) (Masolo et al. 2003) to structure the ADACOR concepts was suggested to overcome this problem and to improve the consistency of the overall system (Borgo, Leitão 2007). This work was limited to manufacturing scheduling and control entities and no real world testing was carried out. However, this work implied the approach that would be later explicitly defined and demonstrated by the work or the SMIF and IMKS research (Young et al. 2007, Young et al. 2010, Chungoora, Young 2010a, Chungoora, Young 2010b).

It has been proposed that it is possible to define an elemental foundation ontology that can be used to construct domain specific ontologies which, because of their shared root meanings can then provide that ability to interoperate. It has been proposed by Young, Gunendran et al (2007) that it may not be possible to create one foundation ontology that

can support the diversity of requirements for all possible domains. In this paper they suggest it may be possible to use multiple foundations ontologies: this would create an extra ontology layer that would require relationships between the foundation ontologies to be defined (a significant task). This work is part of the foundation work for the Interoperable Manufacturing Knowledge Systems (IMKS) research team at Loughborough University whose work is a key foundation for this research. The IMKS project focuses on interoperability across the manufacturing and design domains, and aims to identify a manufacturing foundation ontology through industrial research to identify key concepts and verification methods and uses Common Logic for system design and a PLM system as a source and repository for relevant product and manufacturing facility information (Young et al. 2010). Cassidy (2008) proposed an opposite scenario, where communities could operate using their own local ontology, and as long as they can translate it into a common foundation ontology they could exchange information without semantic mismatches. To simplify this translation task, Cassidy stated that the number of elements in the core ontology should be minimised. This approach is similar to the approach to standards integration proposed by Deshayes, Foufou et al. (2007) which holds the domain ontologies to be of primary importance which can be used to define a core ontology by classifying (manufacturing) concepts and relationships in a coherent modular architecture. This work was progressed further, leading to the development of the Core Concept Ontology (Usman 2012, Usman et al. 2011). The reviewed literature provides a comprehensive review of the previous work on light and heavyweight foundation ontology development and concludes that the level of abstraction or generalisation needed to make an ontology foundational leads to it being too generic to be practically applicable. The concept of a Core Concept Ontology is proposed and demonstrated, providing a foundational ontology for a particular domain which can then be used to develop further, more specialised, sub-domain ontologies with the Core Concept ontology providing semantic consistency. This domain specific foundational or Core Concept Ontology was adopted as a core principle of the solution concept for this research.

The Core Concept ontology adopts the IODE (IODE 2010) toolsets principles of ‘concepts’ and ‘Individuals’ where an individual is any concept that cannot be further instantiated. The concepts are related by Meta Logic which constrains and structures them (Chungoora, Young 2010a, Usman 2012).

The published information on the IMKS recognises the need for further work on greater Product Lifecycle coverage as well as knowledge maintenance. Two key areas not discussed are the ability to maintain interoperability through multiple system lifecycle iterations, and the foundation ontology extension requirements specifically for Manufacturing Intelligence

2.6.6.9 Foundation ontology adoption

The foundation ontologies OpenCyc, SUMO, DOLCE and BFO have been developed to have the expressivity of First Order Logic, meaning that they should enable semantic interoperability which enables computerised inference using rules expressing domain knowledge. None of these projects have adopted the approach of creating a common foundation ontology; in turn their adoption for use has been limited (Usman 2012, Cassidy 2008). The proposed solutions for this limited adoption can be summarised as:

- The number or core concepts should be as small as possible (for simplicity).
- The ontology should be public (for inclusiveness)
- There should be a intuitive natural language interface (to identify duplication)
- The ontology format should have the expressiveness of first order logic as a minimum.

The COSMO ontology project is intending to demonstrate some of these solutions, although it is not intended to be deployed beyond this demonstration activity (Cassidy 2008).

A key challenge for the use of Ontologies is the level of time and effort required to construct, verify and maintain them. Many of the papers reviewed identified automatic and semi-automatic mapping such as algorithms and heuristics as areas for further work (Lin, Harding 2007).

The principal of structured ontology based metadata and rules have been adopted and applied to web platforms, creating the ‘semantic web’ and ontology based query systems (Lin, Harding 2007).

Ontologies share the inheritance features with the object oriented (OO) programming languages, which are indeed suitable for implementing ontological procedures. However, in OO programming, the focus is on designing the operational properties, that is, the methods of a class, whereas ontology development is based on the structural properties, that is, relationships of a class. More importantly, the OO approach lacks the conceptual content of ontologies, and it is not sufficient for addressing rich knowledge modelling needs (Li, Yang & Ramani 2009).

2.6.6.10 Ontological vs taxonomic approaches

Currently, there is some confusion between taxonomy-based and ontology-based applications. One of the major differences between taxonomies and ontologies is that ontology represents much richer domain contexts than a taxonomy or a list of taxonomies. A taxonomy is a hierarchical classification of concepts in a sub domain. These concepts are connected only by domain-independent (i.e., taxonomic) relationships such as is-a. An

ontology, however, consists of several taxonomies, along with multiple domain specific (i.e., non-taxonomic) relationships to connect concepts across taxonomies (Li, Yang & Ramani 2009).

2.6.6.11 Descriptive and common logic

While the use of OWL's Description Logic (DL) goes some way to provide formal meanings to terms, it cannot provide the same level of ability as knowledge representation formalisms such as Common Logic (CL). While OWL allows the definition of binary relationships, CL allows more complex relationships such as the position within a process sequence. The use of CL then allows the use of standard process semantics from the Process Specification Language (PSL) which can then form the basis for a manufacturing ontology (Young et al. 2009). Comprehensive standards and descriptions of both PSL and CL have been published (ISO 2007, ISO 2004).

2.6.7 Context or viewpoints

Context can be described as the collection of relevant conditions and surrounding influences that makes a situation unique and comprehensible. The field of artificial intelligence has been exploring the implications of context in areas such as natural language processing for decades and has had to readjust its initial high hopes for machines that can mimic human intelligence. Context is enormously complex involving numerous interacting factors that people may not recognise on a conscious level, and many of which are outside the ability of machine input devices to capture (Degler, Battle 2003).

A product or process needs to be considered from multiple perspectives or viewpoints to satisfy the many conflicting requirements that must be addressed (Gunendran, Young 2006). In the case of a product or tool, examples of functional viewpoints might be: manufacturability, cost, maintainability etc. Most of the work reviewed discussing information viewpoints is based around product information, however, the principals are equally applicable to process and resource information etc. Approaches for information integration between viewpoints are reviewed in the 'Interoperability Frameworks' section.

The core issue of sharing meaning across functions has been described in many ways; one of the most common shorthand descriptions is that of 'context' as it is applicable to the many different barriers. It is the concept of context that was used to describe the limitation of the conventional geometric feature based approach to knowledge sharing. The feature is generally defined in one specific context e.g. design, which will not relate to the manufacturing or assembly contexts (Young et al. 2007). The two contexts may refer to the same geometry using different terms and parameters or there may be no direct geometric correlation between the features as defined in their contexts.

Young, Gunedran et al(2009) described three key contexts for knowledge structuring:

- Lifecycle
- Product
- System

These are all valid ways of perceiving the interoperability requirement. The Lifecycle context, as described in this work, considers the information and knowledge through the Design, Manufacturing, Operation and Disposal stages of the enterprise activity. The Product context considers the information and knowledge relating to product features, functions and associated activity, processes and most importantly, the relationships between them. It is within this context that design or manufacturing part families are useful, with these distinct types of part family groupings representing aspects of the Lifecycle context. The Systems context considers the software systems used. The Model Driven Architecture (MDA) approach is proposed as a suitable way to specify IT solutions that work across multiple platforms (see 'Architecture' section).

It is reasonable to infer from this work that to achieve enterprise level interoperability, methods for communication of information and knowledge must be able to be understood in a consistent, unambiguous way across contexts and viewpoints (see section on mapping), and that these contexts have significant overlaps and duplication.

A widely recognised way to resolve the issue of 'context for data, information and knowledge is to structure it around products or product families. Cost, Young (2001) discussed this approach and proposed a Product Range Model for variant product design that structures information about the product and its functions, design solutions and their interactions

2.7 Tools

This section reviews the relevant tools in the field of research.

2.7.1 XML

XML (extensible mark-up language) has been used by a number of different authors as suitably robust syntactic structure (Choi 2010, Taghaboni-Dutta, Trappey & Trappey 2010), but it can only be used to document and represent information structures that have been explicitly defined. XML defines the content of a page in a platform independent manner (suitable for PIMs) and is extensible in that it allows the creation of new tags for new and unforeseen purposes (Jr., Michel 2000)

2.7.1.1 XML based exchange

Choi (2010) describes the development of an XML based neutral file format for the management of PPR data using the PLM Services 1.0 specification for the definition of a PLM PIM and PSM (see Model Driven Architecture section). This work highlights that PLM Services are inflexible due to the inclusion of information extraneous for information exchange. A more fundamental issue is that this work relies on natural language to define concepts and relationships and so this work has similarities to that of Taghaboni-Dutta, Trappey et al.(2010) in that it describes the development of an XML schema for the exchange of information and XML/ adapters/ translators/ mapping approaches without first addressing the issue of ontology alignment. As such it is possible that these methods could provide a structured method for mapping concepts or relationships in separate domain models (see Mapping section) which turn out to be incorrectly or inconsistently interpreted leading to an ambiguous or erroneous exchange (see Ontology section)

Choi (2010)provides a detailed review of the development of product data exchange formats (many of which have been embodied in XML). Key points of this review are:

- Most exchange formats have focused on the exchange of product geometry information rather than PLM information.
- The PLM XML format that has been developed by Siemens is still biased towards product geometry and while it is an open schema, its development is limited by the requirement for commercial software (PLM XML SDK)
- Even if all encompassing information exchange encompassing standards were available, they could not be implemented due to the lack of a system which supports them.

2.7.2 RDF and OWL

RDF provides a simple data model and the RDF schema defines a simple ontology language with classes, sub-classes, properties and sub- properties and domain and range restrictions for expressing metadata. However, the RDF schema is not explicit (formal) enough when it comes to representing complex constraints. OWL (Web Ontology Language) has been developed as a vocabulary extension to RDF. (Kim, Manley & Yang 2006).

Lin (2007) used the expressiveness of the OWL primitives in the manufacturing taxonomy and axioms to enhance the information integration with an inter enterprise community. This work also includes a sound review of the development of the semantic web and associated standards (i.e. RDF and OWL).

By providing additional vocabulary along with a formal semantics, OWL facilitates greater machine interpretability of Web content than that supported by XML, Resource Definition Framework (RDF), or RDF Schema. In 2004, the World Wide Web Consortium made OWL a recommendation for Semantic Web technology. Semantic Web Rule Language is based on a combination of the OWL DL and OWL Lite sublanguages of OWL and is intended to be the rule language of the Semantic Web (Kyoung-Yun Kim et al. 2009).

2.7.3 Common logic tools

Knowledge Framework Language (KFL) is a specific example of a Common Logic based ontological formalism which provides expressive logic in which to encode subject matter ontology. First order logic based tools such as KFL provide much greater expressivity than taxonomic model based tools such as OWL (Chungoora, Young 2010a, Deshayes, Foufou & Gruninger 2007). The work on the COSMO ontology proposed the use of OWL with the intention of enabling automatic translation to a common logic compliant language (such as Knowledge Interchange Format KIF (Deshayes, Foufou & Gruninger 2007) or IKL) by representing rules, functions and higher-arity relations in the OWL format (Cassidy 2008).

Both the work to implement the Semantic Manufacturing Interoperability Framework (Young et al. 2010) and the IMKS project used KFL, therefore the Integrated Ontology Development Environment (IODE) was used as it is capable of handling Common Logic-based semantic frameworks (Chungoora, Young 2010a).

2.7.4 IODE, KFL and ECLIF

As described in Section 2.7.3 the IODE system uses Knowledge Framework Lanfuage (KFL) and Extended Comon Logic Interchange Format (ECLIF). In this research, KFL was used to declare relationships, logic and types, whilst ECLIF was used to declare instances

build the queries. The following section describe the basic declarations and formats used in this research.

2.7.4.1 KFL 'properties' or types

The following code is an example that declares a type of person called 'operator':

:Use MI

:Prop Operator

:Inst Type

:sup Person

:name "Operator"

:rem "An Operator is a Person responsible for operating or carrying out the Process."

The 'Use' statement declares the context for this declaration; in IODE the same term can be declared in multiple contexts, which can be explicitly called as by prefixing a term with the context. In this example when instantiating 'Operator' in this context it should be called as 'MI.Operator'. If another context 'Finance' existed then a finance operator would be instantiated using the term 'Finance.Operator' which would be separate from the 'MI.Operator' instance in the knowledge base.

The 'Prop' statement introduces the property of whatever is being declared, generally using its name, as described in the 'Name' statement, although when the name is impractically long or has spaces in it, this property will be a condensed version. It is this declaration that is used when referring to the declaration in the future e.g. a declaration with the name 'Group Capital Controller' may have the property 'GCController. The IMKS coding conventions have been followed so each word or initial of a property is capitalised.

The 'Inst' declares what is being instantiated, in this work the declarations are all 'Types'. An instance of a type cannot cease to be such while it exists.

The 'Sup' statement declares the super-type relationship for this type. In this example 'Operator' has the supertype 'Person' i.e. an operator is a type of person. A type may have a number of super-types, but it cannot be its own supertype.

The 'Rem' statement declares a text based description of the declared property to allow human interpretation of its meaning.

2.7.4.2 KFL relations

Unlike UML KFL does not use ‘attributes’, instead describes entities by relating them to each other or primitive data values. The following code declares two relations called ‘higherThan’ and ‘hasOpAuthorityLevel’

:Rel higherThan

:Inst BinaryRel

:Inst IrreflexiveBR

:Inst TransitiveBR

:Sig Top Top

:Rel hasOpAuthorityLevel

:Inst BinaryRel

:Sig Top OpAuthorityLevel

:name "has Operational Authority Level"

The ‘Rel’ statement declares the relation, whilst the ‘Inst’ statement declares what relationship it instantiates. The relation types used in this work were:

- Binary Relations: a relation linking two properties
- Irreflexive: the binary must have different properties for each of its arguments e.g. for the ‘higherThan’ relation, A can be higher than B or vice versa, but A cannot be higher than A.
- Transitive: If the relation applies to A and B, and B and C it can be inferred that it will also hold for A and C.

The ‘Sig’ statement declares the signature of the relation; for a binary relation this must be the two properties for which the relation can be valid. This effectively constrains the relationship validity so it is critical that it is defined at the right level of generalisation. In the ‘higherThan’ example shown above the relation holds for any properties so the signature is declared as ‘Top-Top’ with top being the most general property. For the ‘hasOpAuthorityLevel’ example the signature shows that the relation can apply to any property, but that the other property must be an ‘OperationalAuthorityLevel’.

It was through these sections of code as listed in Appendix E that the UML diagrams in Section 6.2.1 were formalised or codified.

2.7.4.3 KFL logic

The terms and syntax used within KFL use a similar structure to ECLIF (see Sections 2.7.4.4 and 2.7.4.5), and the IODE tool itself does use ECLIF for queries. KFL uses 6 logical operators of which 5 were used:

- Implication: following the basic 'if – then' form this operator uses a first argument (the antecedent) and a second (the conclusion). If the antecedent is true then the conclusion argument will also be true. The implication statement starts with '=>' and is followed by the two arguments, which look like ECLIF statements in parenthesis.
- Conjunction: this introduces the logical 'AND' operator to the logic. The conclusion argument is true if all the linked antecedents are true.
- Disjunction: this introduces the logical 'OR' operator. The conclusion argument is true if any of the antecedents are true.
- Negation: this introduces the 'NOT' operator. The conclusion is true if the antecedent is false.
- Existential Quantification: this introduces the 'EXISTS' statements. If valid instances for the terms in the antecedent can be found then the conclusion is true.

The Universal Quantification (FORALL) operator, which makes the conclusion true if the antecedent is true for all instances, was not required for this work.

This logic was used to form Inference Rules (IRs) that allow knowledge to be inferred about instances. The same logical operators were also used to form Integrity Constraints (ICs) which, if violated result in a reaction based on the strength of the IC. Two of the available four strengths were used: Soft, in which case a violation generates an error message but allows the transaction to complete, and Hard in which case a violation results in a transaction rollback (i.e. a failure to load).

2.7.4.4 ECLIF instances

ECLIF sentences are declared in parentheses with the predicate (the property or relation) first followed by the entity the predicate applies to:

(Operator Bob) – Bob is declared as an operator

(isTallerThan Bob John) – Bob is taller than John

2.7.4.5 ECLIF queries

Both KFL and ECLIF use '?' to identify an unknown variable e.g. ?x. By inserting variables in to declarations or logical statements, queries can be formed:

(isTallerThan Bob ?x) – Will return all things that are taller than Bob

(and(isTallerThan Bob ?x)(Person ?x)) – Will return all things that are taller than Bob but that are also declared as being a 'person'.

2.7.5 PSL

The Process Specification Language (PSL) (ISO 18629) is the result of work on heavyweight ontology for manufacturing and provides greater expressive rigour then lightweight ontology languages such as UML, IDEF0 and IDEF3 which have previously been used as structured tools for requirements capture and system design using less formal text-based descriptions to define classes and relationships. PSL is a useful example of heavyweight ontology but has limited support for concepts related to manufacturing resources and to work piece relationship to process, limiting its applicability for this research (Young et al. 2007).

An approach that has been used to overcome this limitation with respect to product design and manufacture is to use the Common Logic process semantics from PSL with the Core Product Model from ISO 10303 AP224 (Chungoora, Young 2010a). The work on heavyweight ontologies and PSL has been a key driver for the Interoperable Manufacturing Knowledge System (IMKS) project (Young et al. 2010) which focuses on interoperability across design and manufacturing domains.

2.7.6 IDEF0 and IDEF3

The integrated definition for function modelling was original developed by the US DoD and was subsequently released as an industry standard (FIPS 1993). It has subsequently become the basis for the generic systems models. The labelling nonclementure of the diagram elements varies, with the software based IDEF0 labels of Control and Mechanism/ Call being often replaced with 'Constraint' and Resource' in the systems engineering domain (see Figure 2-3).

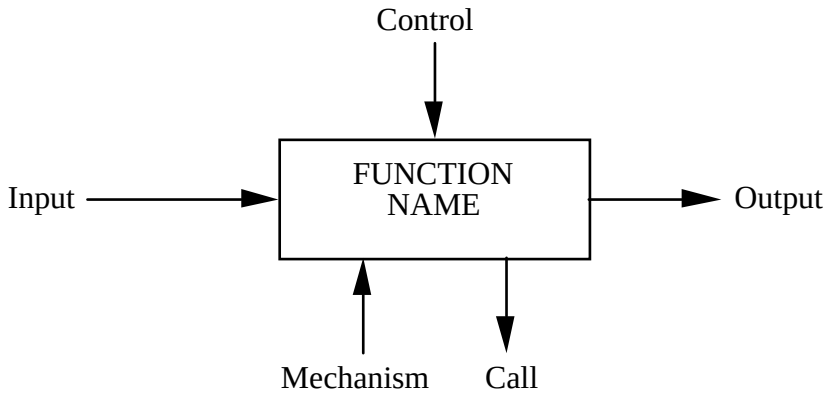


Figure 2-3 - IDEF0 system element representation

Due to the lack of a single integrated tool for the identification of information requirements and structure for a system, the STEP community have used a combination of IDEF0 (for functional requirements and information flow mapping) and EXPRESS (for information modelling). Some have also used IDEF3 capture process relationships and work flows. The effectiveness of these approaches is hampered by the requirement that the system designers to agree on terminology (see Ontology section) (Young et al. 2009).

Modelling processes with IDEF0 and IDEF3 can provide a clear view of interaction points and concurrency (Dorador, Young 2000).

2.7.7 UML

The Unified Modelling Language (UML) is a standard modelling language used in many methodologies, especially those that use object oriented techniques. UML can be used to model (or diagram) almost any entity, including programs, business processes, hardware, networks and architectures. It can be used to provide a common foundation for modelling in an organization (Mochal 2005). UML uses a number of different diagrams to describe a system, these diagrams are classified as either ‘structure’ or ‘behavioural’ diagrams. Structure diagrams show the static structure of the system being modelled and include class, component and object diagrams. Behavioural diagrams show the dynamic behaviour between the objects in the system such as activity, use case and sequence diagrams (Bell 2004).

UML has been used in conjunction with IDEF0 and IDEF3 in application design, where the IDEF tools were used to provide the top-down approach and UML the detailed bottom-up approach. This duel approach was necessary as UML is suitable for software design, but needed the extra support for requirement capture stage of the lifecycle (see Systems Lifecycle section) for the definition of the structure of the information models (Dorador, Young 2000).

The UML class diagrams have been selected for use as it is a widely understood notation and tool for describing the objects and relationships and is therefore a useful tool to develop a more detailed understanding of the objects and classes identified through BPMN construction activity.

This section describes the use and construction of Class diagrams within the standards set by OMGs UML 2 specification. Class diagrams are typically used to show classes, interfaces, data types and components, while other UML diagrams enable greater focus on some of these elements, the class diagram is particularly useful as it has an ability to represent them all (Bell 2004).

Figure 2-4 shows how classes are represented in UML.

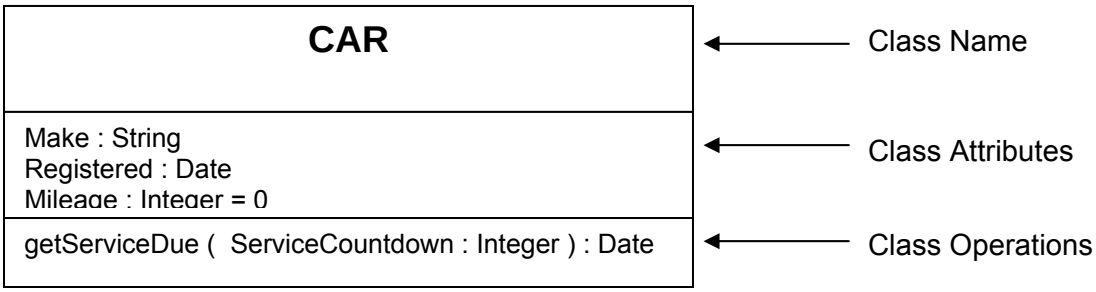


Figure 2-4 - Simple UML class figure for CAR

The class is shown as a rectangle with 3 compartments, the top compartment shows the class name (in bold type), the middle shows the class's attributes and the bottom the class's operations. The middle and bottom compartments are optional and while they may be used later in this work, initially only the class names will be used to ensure the correct level of focus on the basic structure before moving on to more detailed considerations.

The attribute list follows the convention 'name : attribute type' where the unit type description is an appropriate unit for the user of the diagram e.g. if the diagram is to be used to generate code, the units must be available within that programming language, whereas in a business class diagram the units can be less constrained. Adding an equals sign and a value to an attribute indicates a default value, e.g. in the CAR example a new car starts with a mileage of 0 miles.

The operations list uses the convention 'name (parameter list) : type of value returned' where '(parameter list) represents the input parameters for the function. In the CAR example the ServiceCountdown parameter is used in the getServiceDue operations to return a due date for the next service.

UML 2 includes a notation standard for showing instances of a class, which is largely similar to class notation with the constraint that the instance diagram relationships must match the class diagram relationships. Instance modelling instances is likely to provide an inappropriate level of detail in this work so is unlikely to be used to a significant level.

A key concept in object oriented design is that of inheritence: this is where a child class inherits the same functionality as the parent class (or super class). The child can then add specific functionality of it's own in addition to this.

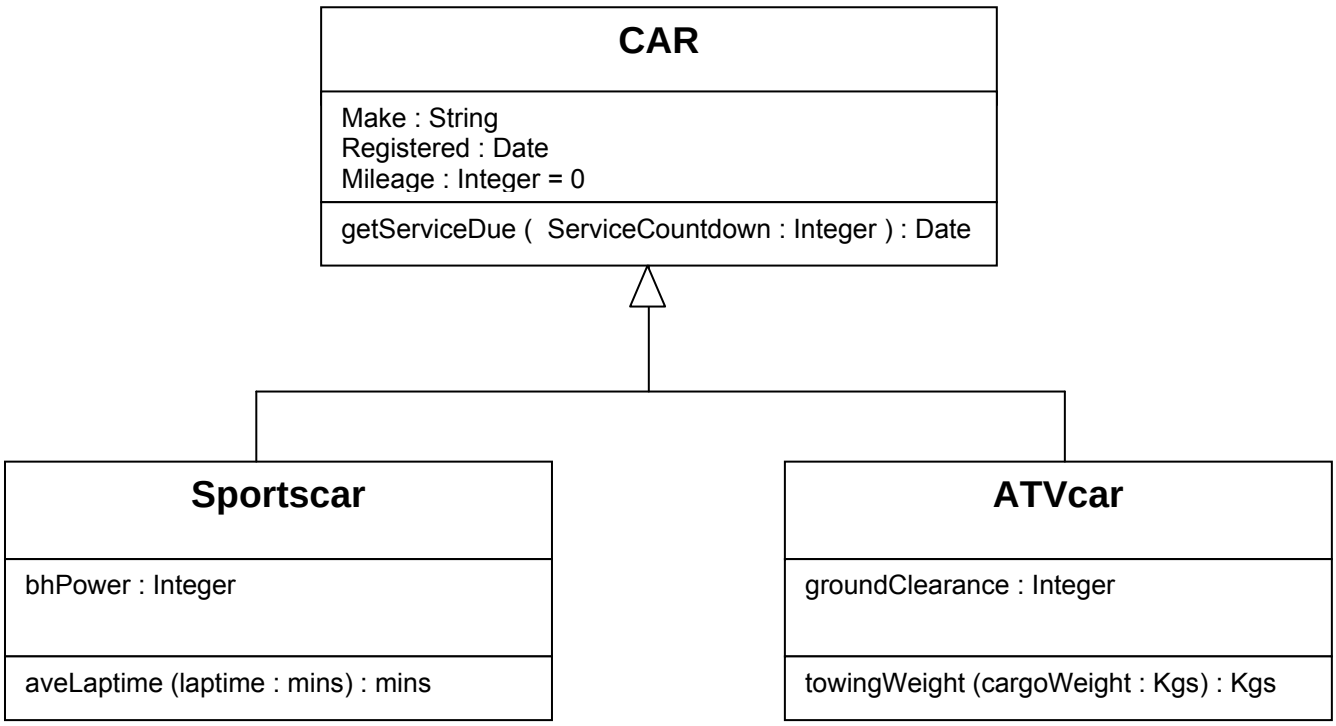


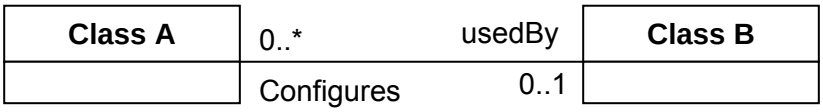
Figure 2-5 - Inheritance using tree notation

Figure 2-5 shows that two child classes (sportscar and ATVcar) have been defined beneath the CAR superclass. Each of these classes inherits the attributes and operations from the CAR class (these are not relisted at the child level) and incremental attributes are shown for each class. Inheritance is indicated by drawing a connector from the child to the superclass with a closed, unfilled arrowhead pointing to the superclass.

It is possible to define a superclass which is abstract which in turn defines abstract operations which can be implemented in different was in separate child classes. This is not envisioned to be a key requirement for this work.

The relationships between objects must be modelled for clarity. UML uses 5 types of association:

- **Bi-directional (standard):** A solid line between classes indicates that both classes are aware of each other and that there is a reciprocal relationship of some sort. The diagram will also indicate the multiplicity and the role the class takes. This information is added next to the class box. The role describes the nature of the association while the multiplicity describes whether this association is unique to specific instances or multiple instances (in either direction).



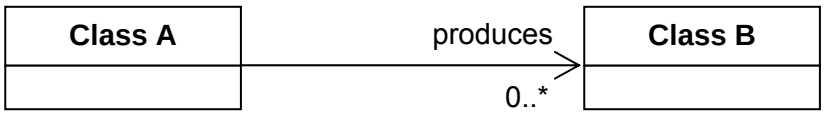
An instance of Class A is associated with a specific instance of Class B (or no instances), however that Class B instance may be also associated with other instances of Class A. Class A configures Class B, and Class B is used by Class A.

Figure 2-6 - Bi-directional association example

0..1	Zero or one
1 (or any other integer)	One (or other integer) only
0..*	Zero or more
*	Zero or more
2..8	Two to eight (inclusive)

Figure 2-7 - Multiplicity examples

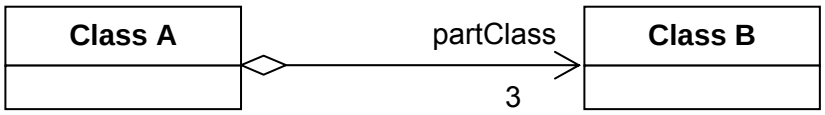
- **Unidirectional:** Only one of the related classes knows that the relationship exists. This is common when modelling processed flows as backflow in process such as manufacturing are often considered impractical or undesirable unless required for process feedback.



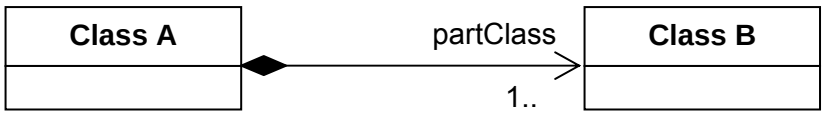
Class A produces (any number of) Class B.

Figure 2-8 - Unidirectional association example

- **Association Class :** It is possible to define a class which contains information purely relating to the relationship. In this case the class would exist for every instance of that relationship and would be shown as a class box connected by a dotted line to the association line between the two primary classes.
- **Aggregation :** Two forms of aggregation can be described; 'Basic' and 'Composition'. Basic composition shows one class to be part of another and is represented by showing an unfilled diamond at the superclass end of the association line. Composition aggregation is similar to basic except that the child class cannot exist independently of the parent.



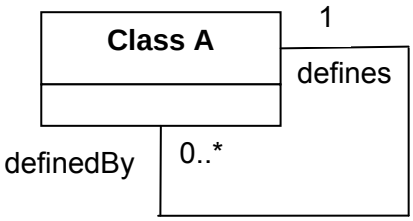
Basic Aggregation: Class A contains 3 instances of Class B, but if Class A ceases to exists the instances of class B persist



Composition Aggregation: Class A contains at least 1 instances of Class B, but if Class A ceases to exists the instances of class B are no longer valid/ exist.

Figure 2-9 - Basic and composition aggregation examples

- **Reflexive :** Rather than being a relationship between two classes an association can describe relationships between class instances.



Certain instances of Class A can be used to define other instances of Class A however not all instances will have another instance to define.

Figure 2-10 - Reflexive association example

For clarity it is sometimes desirable to show how classes are grouped. This grouping should be logical to the context of the diagram and the user of the information. UML supports a 'package' notation which can be displayed as a large folder representation within which the grouped classes are shown and which has the group title displayed on the folder tab.

UML 2 also has the ability to display the internal structure of a class. This can be used to simplify the interpretation of compositional diagrams, especially where there are a number of relationships. This is particularly useful for describing physical classes, but is unlikely to be required for this research. Further information on this and other aspects of UML not directly applicable to this work is available (Bell 2004).

2.7.8 Protégé

The open source 'Protégé' ontology editor and knowledge based framework tool is considered to be one of the most widely used ontology engineering tools and well supported by the medical informatics group at Stanford University (Noy, McGuinness 2001). It provides tools for ontology editing including concept, taxonomy and relationship building as well as ontology visualisation. Protégé supports multiple representation formats such as XML, RDF schema and OWL (Li, Yang & Ramani 2009). This view is supported by the use of the tool in (Blomqvist, Öhgren 2008).

2.7.9 BPML & BPMN

BPMN and BPML were both developed by the Business Process Management Initiative (BPMI), however, they are now supported by the Object Management Group (OMG)(OMG 2012, OMG 2008).

Business Process Modelling Notation is a standard for modelling business process and web services. BPMN provides a number of advantages to modelling business processes over the

Unified Modelling Language (UML). First, it offers a process flow modelling technique that is more conducive to the way business analysts model i.e. it is more user friendly and potentially more intuitive to understand for non analysts. Second, its solid mathematical foundation is expressly designed to map to business execution languages, whereas UML is not. BPMN can map to UML, and provides a solid business modelling front end to systems design with UML (White 2005).

BPML (Business Process Modelling Language) is a tool for business process description which defines a formal model for abstract and executable business processes. BPML is capable of expressing:

- Activities of varying complexity
- Transactions and their compensations
- Data management
- Concurrency
- Exception handling
- Operations Semantics

BPML also provides a grammar in the form of an XML schema for persistence and interchange of definitions across heterogeneous systems (Izza 2009).

BPMN has been chosen for the following reasons:

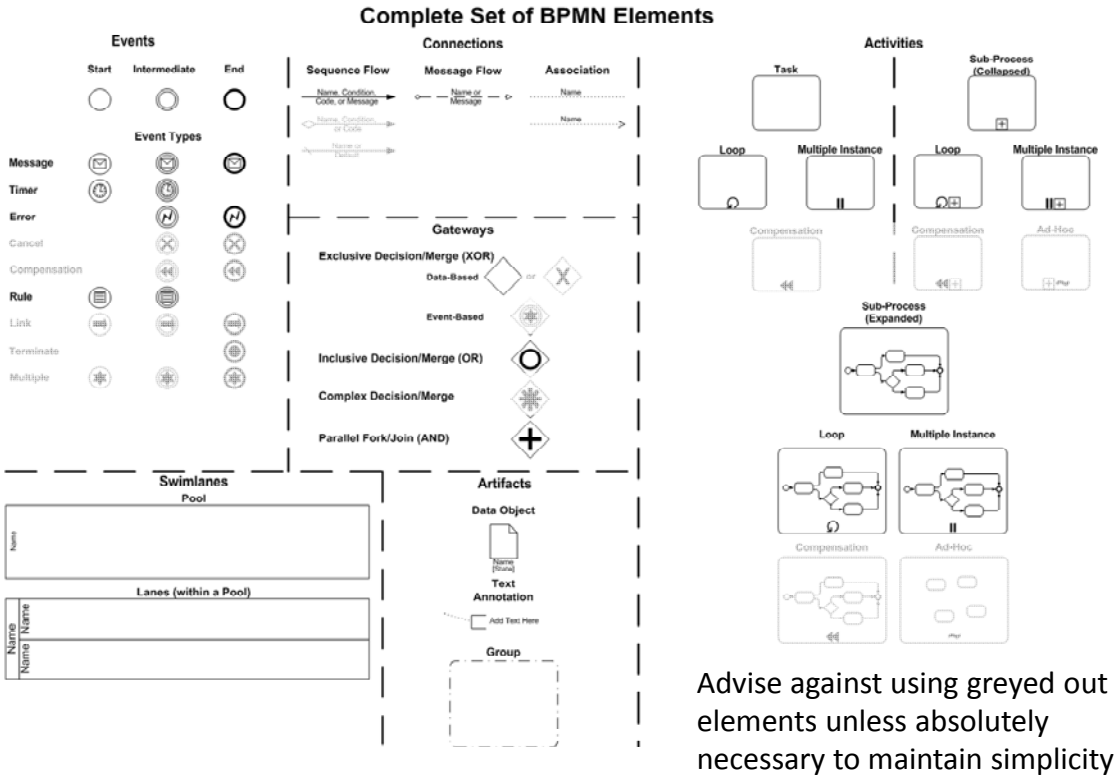
- It is capable of describing and specifying processes and systems from many 'perspectives' e.g.:
    - Process flow
    - Inputs/ outputs/ Variables
    - Data flow
    - Decisions/ reactions
    - Roles and responsibilities
    - Process boundaries/ scope
    - Timescales
- There are a number of specific tools, templates and processes for mapping a process from these perspectives; however, BPMN is particularly capable of providing an easily understood overview on a page.
- The Business Process Mapping Notation is a global standard which should be intuitive for engineers familiar with flow charting.
- This method can show the process information inputs and outputs of a System Input Process Output (SIPOC), the (complex if required) process flow and decisions and

key roles and events. As such it is a concise and clear way of representing a process and its requirements.

Some key aspects of the BPMN standard are:

- There are start, finish and intermediate events.
- Pools represent different organisations or groups, which can if required be broken down in to lanes represent individual participants.
- Process flow arrows cannot cross pools (the separate pools have their own process which may be joined/ triggered/ coordinated by messages or data flow).
- An event can be put on the edge of an activity to signify an ‘interrupt’ i.e. under certain circumstances follow this path.
- Gates can be considered like decision boxes in normal flow charting and are able to split or join process flows
- The diagram author needs to consider at what level to map the process, and it is likely that that once started they will change their mind, so it is recommended to sketch out the process before using electronic tools.

This section describes BPMN v1.1 however, v2 has recently been published. The new revision is consistent with the earlier version but develops more complex and capable coordination capabilities between parallel processes. One of the key aspects for which BPMN has been selected is its clarity and ease of use for those with basic flow charting experience, as such a small subset of the full elements available will be recommended for use (see Figure 2-11). Using a combination of these basic elements it is possible to represent many of the elements discounted. While this may not be as efficient as using the full element set, it makes deployment and use of the tool by inexperienced users easier and more consistent. This means that the changes in v2 are not particularly relevant in this case.



Advise against using greyed out elements unless absolutely necessary to maintain simplicity

Figure 2-11 - The subset of BPMN elements proposed for use (OMG 2008)

These elements can be used to describe a process, its flow, decisions, events, interruptions, messages/ data/ information items and flow as shown in the following examples. These diagrams are clear and intuitive but can be implemented in a number of ways depending on the intended outcome. This is particularly important when deciding the level of detail at which to map, and how the agents (i.e. swim lanes) are defined.

When mapping a number of processes, consistency in the level of detail and method of construction are vital. Like most standards, to enable its wide application it allows significant interpretation (see literature review 'Standards' section). This will be managed by using a core team of modelling facilitators who meet regularly to compare developments and one strong development coordinator.

Figure 2-12 and Figure 2-13 show example BPMN diagrams.

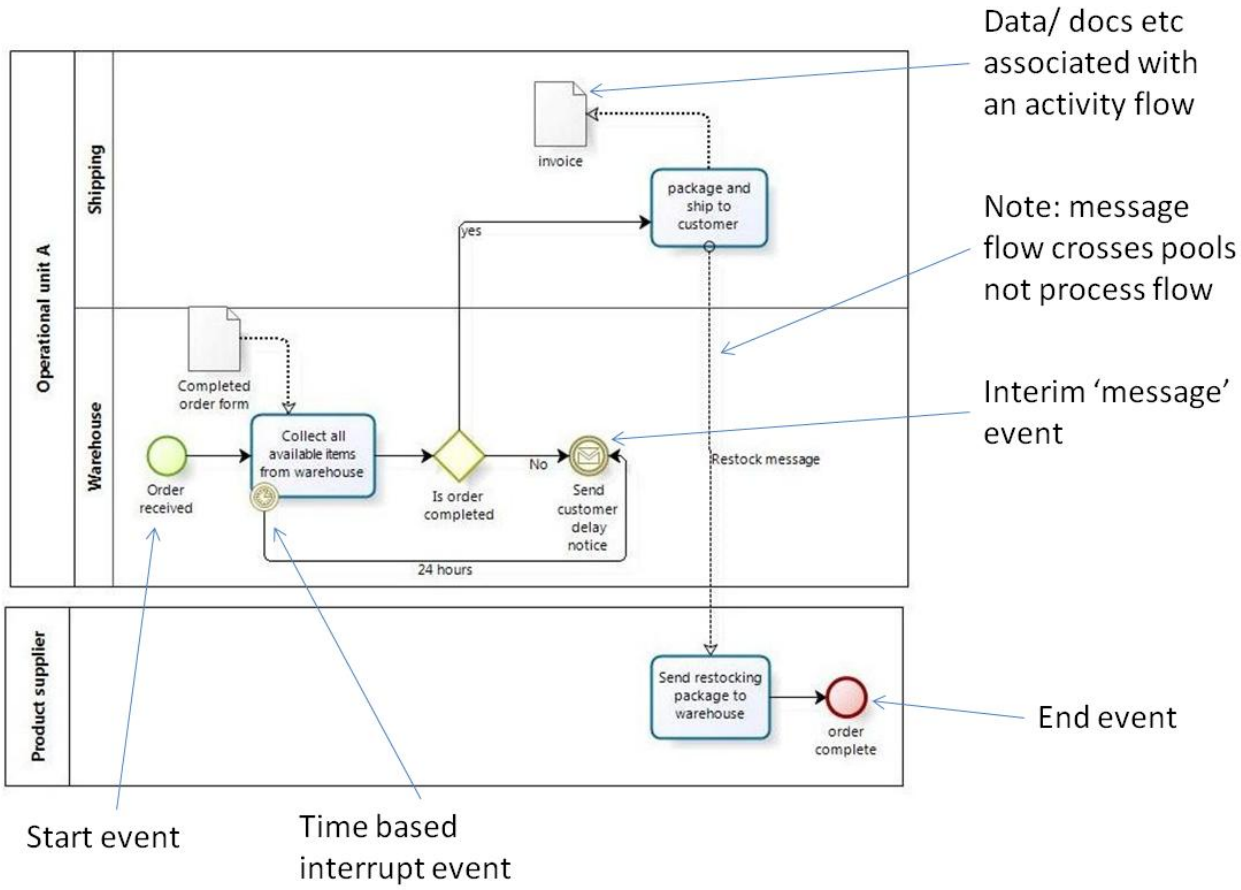


Figure 2-12 - A basic BPMN example

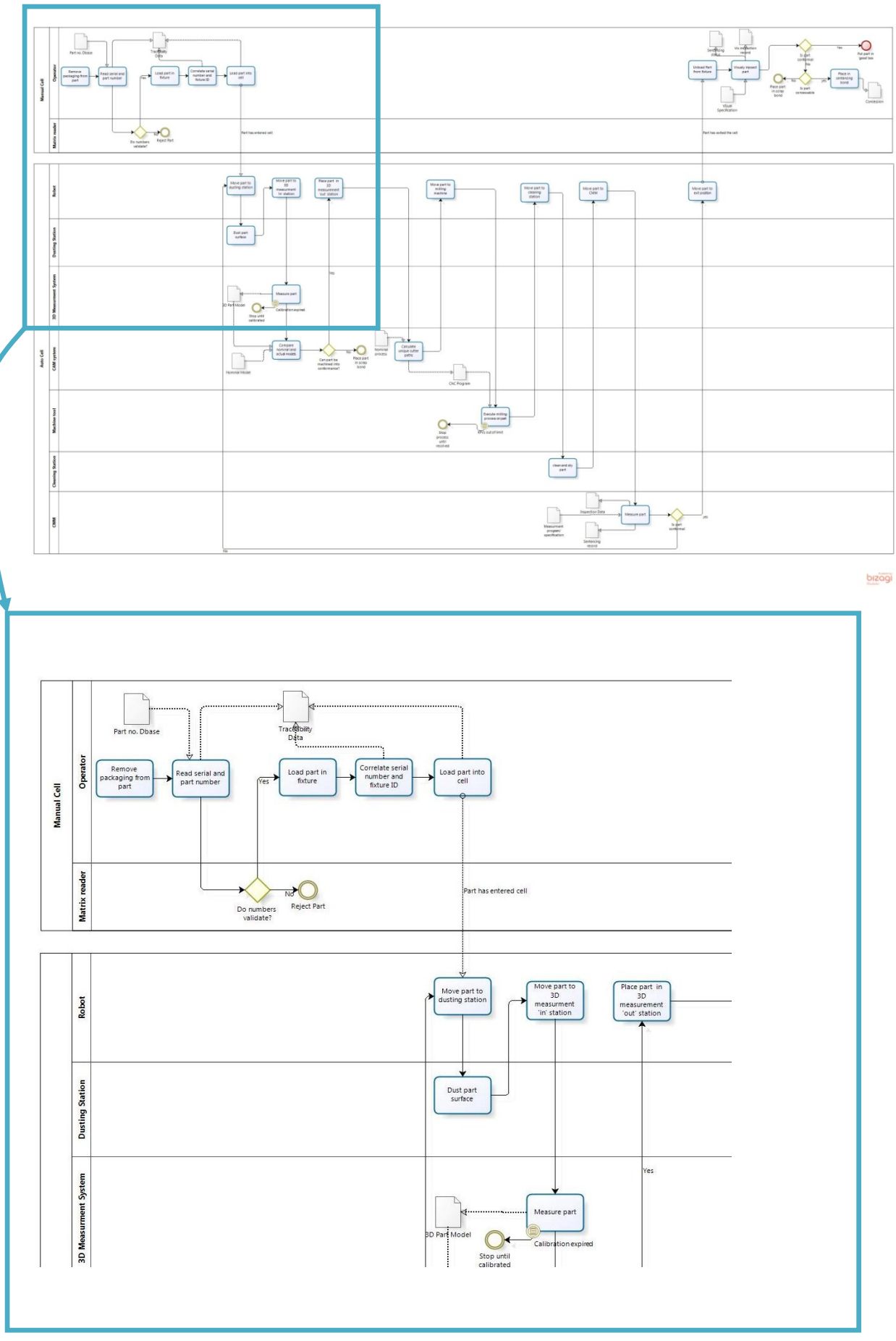


Figure 2-13 - A more complex BPMN example

2.7.10 Business to manufacturing mark-up language (B2MML)

B2MML is a set of XML schemas for exchanging information between MES level functions and Business level functions such as ERP. This is based on the object models defined in ISA95 (ISA 2000, ISA 2005, Scholten 2009).

2.7.11 Data dictionary

A Data Dictionary or metadata repository can be described as a "centralized repository of information about data such as meaning, relationships to other data, origin, usage, and format." (Lecky 2003, IBM 1993) : It describes in detail how a database was designed and all its key characteristics. This information allows developers and analysts to quickly access information about the tables, fields, procedures, processes and other information in the system.

The Data Dictionary defines the basic organisation of a database and collects together detailed information about database system components such as: data element definitions (tables, fields, key fields, primary keys, relationships etc), program elements used by the database to move data about or to manipulate it in some way, records (numbers of - not the actual records themselves), system parameters, system information, files and other system components, entity relationship diagrams.

A Data Dictionary can be produced:

- Automatically using a software tool to interrogate the database and map its structure
- Manually by combing through the code to determine the structure
- By a combination of automatic and manual processes.

The reviewed literature suggests that a combined automatic and manual process is best as the automatic systems can map the database schema but cannot gather the business context of each field or the underlying functional requirements such as storage length etc. The relative abilities of the manual and automatic data dictionary construction methods to gather large amount of data without significant qualitative weighting or smaller amounts but focusing on context and relevance holds a strong parallel with ontology creation methods (see Ontology Creation section). As described in the Ontology section, by many definitions a dictionary is an ontology, albeit a relatively simple one.

The Data dictionary is a detailed tool for explicitly documenting detailed data and information structure and is useful as such. However, while it can describe data stores in great detail, it still requires natural language descriptions and interpretation and is therefore subject to semantic inconsistencies. A data dictionary is also specific to an instance e.g. a database or

individual operation. While tools are being developed to map dictionaries/ ontology's, they still require manual validation (see Ontology Mapping section)

2.8 Summary

2.8.1 Manufacturing intelligence – key points

- Manufacturing Intelligence has been described as critical to business performance in modern manufacturing.
- MI deals with many heterogeneous functions, systems and holonic production units
- MI is rapidly evolving both conceptually and technologically and is influencing and affecting the systems and process with which it functions, in some cases interfacing, interoperating, integrating, or replacing entirely.
- MI has traditionally been described as a sub function of MES systems, however, the increasing scope of MI is now expanding beyond the traditional scope of MES system e.g. while the collection of production data is within the traditional scope of MES systems the use of embedded knowledge to inform the control of a unit of Holons merges this function with the control level of the enterprise according to the ISA95 definition.
- The business functions supported by MI data and information work on vastly different timescales, from discrete year-to-year planning at ERP level to (near) real time monitoring and interaction at the control level.
- MI requires systems which are agile and dynamic, with that ability to be rapidly re-engineered in timescales that are compatible with these functions.
- MI is used to generate, monitor and in some cases react to KPIs which are the crucial metrics used to monitor enterprise performance at all levels and timescales, however, these KPIs are loosely defined using natural language meaning that even within an enterprise they can be generated and interpreted inconsistently.
- SCADA systems can provide a technical solution to shop floor level supervisory control, but they are not capable of resolving domain semantic inconstancies which therefore have to be hard coded or fully mapped and designed into the control protocols, both of which are inflexible and cumbersome solutions.
- SCADA systems have not historically been directly integrated with MES systems and certainly not with PLM or ERP systems due to the potentially incompatible time bases and semantics of the systems.

2.8.2 Systems lifecycle – key points

- While there are clear systems lifecycle framework models available, these focus on the development of an individual system through to its retirement. This lifecycle may be iterative, generating many version of that system, but there is limited consideration of a migration lifecycle stage where functionality moves from one

system to another, and maintaining functional equivalence at the specification and functional levels.

- Organisational and Technology development
- As timescales of systems lifecycles are accelerating and becoming shorter, anything that hampers this pace can be considered a threat to business or enterprise development.
- System developments, which are replacing legacy systems, are the most complex and likely to fail.
- Concepts or properties within a system can be differentiated by whether they are time limited or not (endurant or perdurant), enabling and possibly requiring two styles of description of entities within the same times space.
- Proposed systems for assessing system conditions rely on qualitative assessment leading to consistency problems.

2.8.3 Information sharing – key points

- The requirement for interoperability between functions, domains and systems is increasing due to the significant benefits it can enable.
- Enterprise data, information and knowledge are dynamic as are the related systems; Knowledge sharing mechanisms must not constrain the ability to maintain this content and these systems.
- Significant levels of work have been published on information sharing as an enabler for collaboration, (particularly between design and manufacturing), however, this work generally focuses as collaboration within one timeframe (i.e. between domains at a particular time) rather than enabling interoperability across timeframes.
- The recognition that common basis of sharing meaning (both processes and systems) is a fundamental prerequisite for unambiguous communication.
- While standards have a part to play in allowing shared meaning, they require interpretation which leaves them prone to semantic miscommunication.
- There are a large number of approaches to achieving interoperability many of which focus on different aspects of the challenge.
- The use of an abstract or common model defining concepts, objects, rules and relationships, which are mapped to real world specific instances. Common focuses of this work are:
    - Using the common model to create conformal instances
    - Using instances to define a common model
    - Using existing instances and a common model to define rules or relationships.

- The recognition of the computational complexity of semantic reasoning and the significant time and effort required to represent the evolution of dynamically changing data.
- ICT based solutions have traditionally resulted in systems which are limited in scope, complex, burdensome and inflexible to change.
- Interchangability includes the concept of interoperability and requires that the semantic and application functionality be maintained if a system is substituted.
- Fully integrated systems, with their implied interdependency pose a greater operational risk than interoperable systems (systems which can exchange and use information together but function separately). Integration of individual systems can result in complexity and inflexibility.
- All reviewed models of interoperability share the view that achieve seamless information sharing the dynamic alignment of concepts across domains must be achieved which in turn required the technical and semantic challenges to be overcome.
- Enforcing the use of standard systems or tools to overcome the technical challenges of interoperation is ineffective unless the entire supply chain (internal and external) can adopt the standard tools. External suppliers will be unable to adopt the various systems their customers would be enforcing.
- Customisation of systems to achieve integration significantly reduces the ability to migrate to new versions or alternative systems in the future. This integration capability must be designed into the functionality and configuration in the early lifecycle stages using the systems core capabilities
- The tight configuration control requirements of PLM systems do not fit well with the requirements of interoperability across many systems.
- While standards are vital for interoperability, standards constructed using natural language, even those using tools such as XML, are subject to the issues of inconsistent semantic interpretation and the adaptation requirement for specific implementations. An integration standard which is totally comprehensive would be impractical to define and too inflexible to apply.
- The unified approach to interoperability is generally considered to be more appropriate and applicable than the inflexible and cumbersome integrated approach and impractical, ontologically challenging federated approach.
- The lack of standard ontology's for concepts, relationships and properties of enterprise architectures is the key deficiency in the development of interoperability architectures.

- Foundation ontology's using a model driven approach have been proposed to enable inter-domain interoperability. The use of common logic, heavyweight ontology's has been used to make these ontology's unambiguous.
- Interoperability work has generally been constrained to single product lifecycle stages and multiple domains or multiple product lifecycle stages (mainly design and manufacturing) and single domains due to the complexity of the concept relationships within a single stage or domain being exponentially increased over multiple stages or domains.
- The optimum ontology creation method uses a combination of manual and automatic techniques to gain the focus on concept relevance of a manual approach and the coverage and structure of an automatic approach.
- The challenges of maintaining interpretational consistency across domains and contexts is the key challenge for interoperability.
- Three key contexts for structuring knowledge have been defined as Lifecycle, Product and System.
- To achieve enterprise level interoperability, methods for communication of information and knowledge must be able to be understood in a consistent unambiguous way across contexts and viewpoints and that these contexts have significant overlaps and duplication.
- The Model Driven Architecture has been proposed as a suitable way to specify IT solutions that work across multiple platforms.
- The challenges of differing system and data timescales has been recognised within the Manufacturing domain, however, only a loose framework of approaches to meeting this challenge has been proposed rather than a specific solution.

2.8.4 Conclusions

The conventional or current definitions of Manufacturing Intelligence are out of date and insufficient. They fail to address the evolving requirements for near real time decisions and information for longer term planning from the shop floor along with the capability to take decisions and enact the outcome based on information and knowledge embedded within the systems as a collective.

While integration of MES and PLM/ ERP systems is being considered, this is poorly aligned with the nature of the new MI requirements. The technical requirements for MI are potentially aligned with the developments in data collection systems, however, without a comprehensive definition of MI it is unlikely that all the technical requirements will be met.

The benefits of rapid i.e. nearly real time, data and information enabled decision making at all levels of a manufacturing enterprise are clearly documented. The ability to plan accurately, react quickly and even pre-empt situations can save industries billions of dollars or pounds in waste. As the pace of industry increases with automation and technology, so the need for accurate data, information and knowledge increases. As the required pace of information collection and exchange increase the challenges and risks of applying context to data correctly and interpreting it increase exponentially leading to the risk of inconsistent or incorrect interpretation and subsequent incorrect decisions and actions being taken.

As new systems develop it is unlikely that it will be possible to maintain a one to one mapping between the new and old systems scope. This gives rise to the complexity of having to migrate data and information from and to and multiple systems. An interoperability method that is non system, format, context or perspective specific should allow this.

Applying global standards to systems and process on a manufacturing shop floor to provide consistent data is theoretically possible. However natural language-based standards are not sufficiently rigorous to provide semantic consistency, flexible enough to keep pace with the rapid new developments and changes or extensible enough to be able to describe emerging concepts and relationships without a level of local customisation. Therefore a framework standard could be used, but would require a more structured ontology to describe the objects, concepts and relationships within the standard. If this ontology is suitably extensible and flexible it may be possible to keeps pace with the changes within the MI field, maintaining consistent, unambiguous information exchange. If the basis for this ontology is consistent, it may also be possible to use it to describe or define future MI systems and implicitly ensure semantic and functions consistency.

The degree and rate of change and the number of systems, interfaces, concepts and relationships affected when considering MI systems, is increasing. The nature of the MI as a fast evolving field with many different functions/ domains, poorly defined concept standards and no standard ontology led to this being significant information sharing challenge. If this challenge is not addressed it is unlikely that MI will function across an enterprise (albeit there may be local islands of autonomation and success) or that any MI capabilities will be sustained over time.

The use of an ontological and in particular, a foundation ontology approach, as proposed for other areas of information sharing may go some way to resolve the challenges posed when maintaining and extending MI capabilities, but what is unclear is how this approach will cope with the rate and unpredictability of future changes and developments and the complexities of dealing with so many holonic manufacturing domains, which may be provided by different solution providers and so may have their own perspectives and semantics, as well as being used across the many enterprise functions and even shared between enterprises.

2.8.5 Key related areas of work

Key areas of related research were reviewed, highlighting gaps in the current research that are covered by this work are summerised and listed below. A full list of related topics reviewed can be found in the literature review:

Manufacturing intelligence definition: The reviewed material showed there is a lack of clarity regarding the definition of manufacturing intelligence, with some of the definitions being incompatible.

Dynamism: Highlights the need for flexibility through time for enterprise applications due to the rate of change in the systems but does not propose a solution.

System growth and complexity: Raises the issue of change, proposing using axioms to constrain growth in the desired way but does not make the link to heavyweight ontology development research.

Legacy systems: The reviewed work proposes a solution to the interoperability challenge of legacy system which relies on a consistent method of system comparison within or across domains, however no solution is proposed. A systems lifecycle based solution has also been proposed, however, this does not suggest a solution for the need to evaluate post-implementation and evaluation stages for legacy and emerging systems.

Timescales: This work linked the operating environment and the lifecycle rate, highlights the impact on system interoperability challenges, but does not propose a solution.

The requirement for interoperability: The requirement for enterprise to shop floor interoperability is described, along with the limitations of standards and interface-based solutions proposing a draft framework standard approach, which is still subject to semantic inconsistency and impedes flexibility and ability to change. This interoperability research focuses on interoperability within one timescale rather than between timescales or legacy and emerging systems.

Semantic Standards: This work develops and formalizes the Semantic Manufacturing Interoperability Framework standard proposing that the looser control of a framework allows adaptations to allow the standard to develop through time, however, no mechanism to ensure that the developments on the Semantic model are consistent through time is proposed.

Concept alignment: To achieve organisationally integrated and seamless information sharing, alignment of concepts must have been achieved, however, the solutions proposed have limited applicability in dynamic environments.

Foundation Ontology: Work on the development of foundation ontologies to provide semantic consistency was reviewed. This work identified several areas for further work including knowledge maintenance through time. Two key areas not discussed are the ability to maintain interoperability through multiple system lifecycle iterations, or the foundation ontology extension requirements specifically for Manufacturing Intelligence

2.8.6 Further work

This review has identified the need for further work in a number of areas:

- An approach for describing strategic manufacturing requirements, objectives and measures in a way which allows the automatic/ systematic flow down of these requirements and automatic configuration of sub-objectives, metrics and measures (such as KPIs/KPVs)
- A more robust, unambiguous description of key metrics and monitors e.g. (KPIs/KPVs) to provide a basis for consistent automation of reporting and knowledge based reaction.
- An up to date definition of Manufacturing Intelligence based less on technology and more on process and interactions with other functions and business processes.
- An approach to ensure flexible, timely, unambiguous information sharing between MI, MES, Control Systems and the production process including intelligent, data driven decisions, actions and automation.

- How the risk and complexity of legacy system replacement can be addressed using the techniques being developed in the field of information sharing (across domains), focusing in particular on the rapidly developing field of MI due to the rapid development lifecycle, differing timescales and number of system elements, concepts, domains etc.
- The most appropriate level of, and approach to Manufacturing Intelligence knowledge sharing between systems including the correct level of explicit, standard definitions versus dynamic information and knowledge management.
- Developing the principal of heavyweight foundation ontology for MI to provide cross lifecycle MI continuity and extendibility, including objects, concepts and relationships.
- Defining an updated definition of MI, MI standards and ontology that allows MI to be delivered through the configuration of off the shelf product functionalities rather than focused MI applications which in turn could provide a basis for consistent lifecycle development and migration i.e. flexing to adopt future requirements while maintaining required functionality.

3 An industrial investigation of key interoperability issues

3.1 Introduction

The purpose of this industrial investigation was to identify the relevance and impact of the issues relating to interoperability in dynamic change environments in an industrial environment. Due to the nature of the information the name of the company has been withheld. The investigation was carried out with the support of subject matter experts and technical authorities within the company.

The investigation focused on a large and diverse organisation which operates as a design and manufacturing solution provider over a large and complex supply chain. It's understanding of the power of integrating its many areas of knowledge has led the company to become a leader in the management of knowledge and information on a massive scale. The organisation has been leading and coordinating work at key institutions around the world to allow the relating of essential information and knowledge to its products using a feature based approach. It is also developing one of the world's most comprehensive PLM systems, which can be seen as an indicator of its commitment and approach to knowledge management.

The following subsections cover the need for, and key challenges to interoperability. The content has been grouped under subsection headings which reflect the significant topics identified in the literature review. Key concepts that were defined to enable a clear understanding of the research material have been declared in this chapter with these concepts being further developed in Chapter 4. In Section 3.4, fundamental concepts elicited through the industrial investigation are declared, such as what a 'system' is in relation to a 'process' which. These declarations have been used to ensure any underlying models or assumptions are explicitly described. The final section explores the industrial understanding of interoperability in the MI domain

3.2 The need for interoperability

The diagram below represents a simplified view of some key types of data and the systems in place to store it within the organisation. It shows that this data can be split between and in some cases duplicated across multiple systems. It includes a reference to the knowledge bases that the organisation access but does not necessarily own such as public, supplier and advanced research centres.

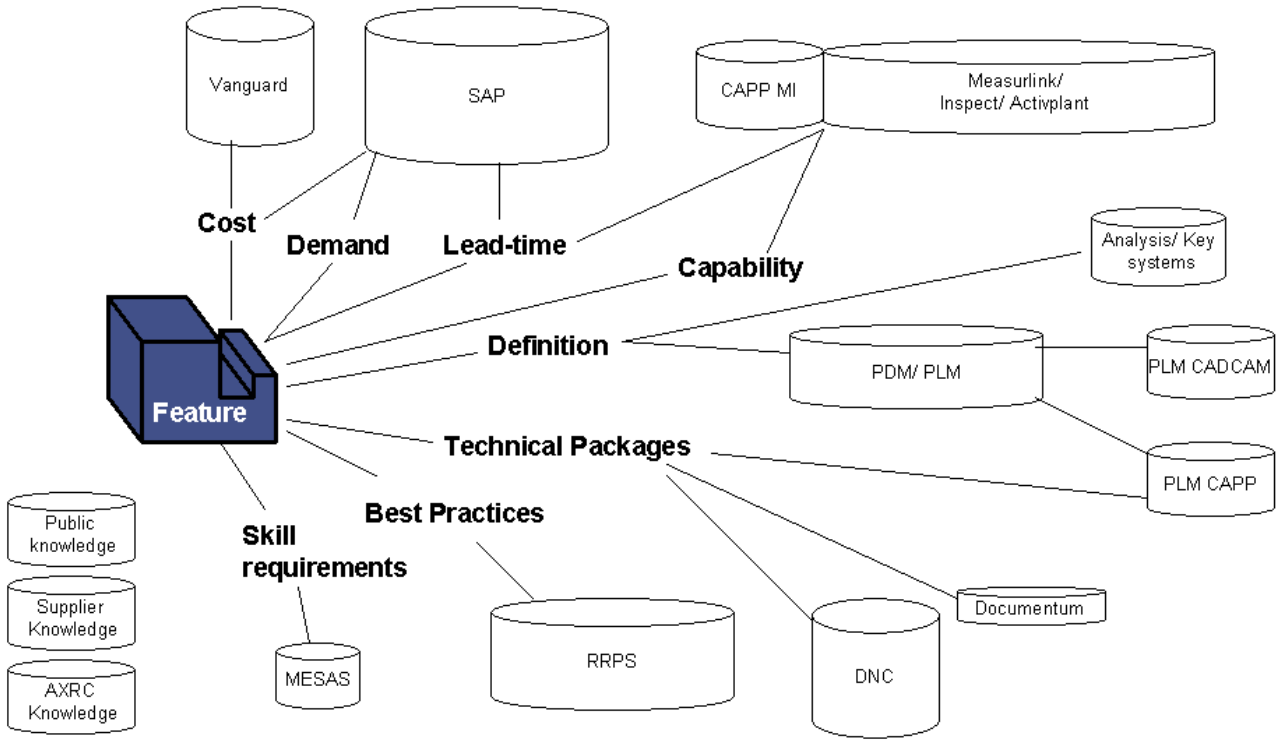


Figure 3-1 - A simplified representation of some of the organisation's knowledge and data- bases

Figure 3-2 shows the kinds of query that can be resolved by these knowledge/ data bases in the context of a particular feature. The feature context is critical to the query as the same query in a different context or different feature would possibly require a different answer. As a product and service engineering company, many of the organisations system are configured to store data and a product or feature context which is a strong basis for interoperability amongst those systems. The risk is when a query from a different domain e.g. HR, is put to these systems as it may be misinterpreted, or when a system that is not able to use the same feature context e.g. a design or manufacturing feature, is integrated. This diagram also shows that a query that spans several knowledge/ data bases cannot be resolved without the data being collated separately.

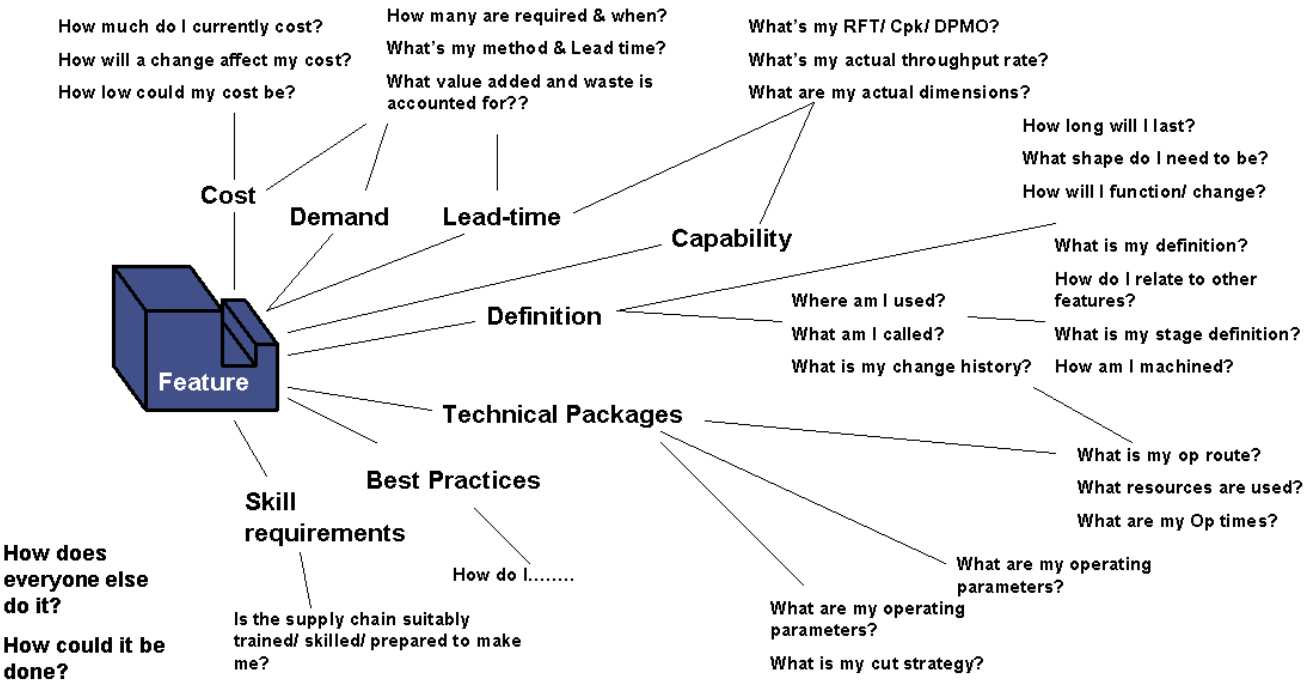


Figure 3-2 - Typical queries for the systems represented in Figure 3-1

The following sections will describe how the organisation approaches the challenges of increasing system interoperability.

3.3 Key interoperability issues

3.3.1 Standards

The organisation has a large and comprehensive system of standards, however, its Quality Management System is an intranet based systems which provides access to the mandatory and legal standards for the whole company. The organisational diversity that these standards have to cover, and the fact they are externally audited requirements, mean that they are defined at a high level. The QMS holds standards for systems procurement processes as well as portfolio management. The QMS also references other standards such as PLM Operating Procedures and Best Practices. The intranet also contains many process guides and other non mandatory documents for the operation of the ERP system (SAP), Product Data Management System (Metaphase) and the Execution System (SFDM). These systems are centrally configured and the company has undertaken significant programmes of work to converge the global instances to common configurations. This is coupled with very strong change and interface standards and governance due to the legal implications.

The usage of these 'core' systems is, however, very lightly guided rather than governed leading to significant variation in data population and completeness. While the company has focused heavily on reducing the number of SAP exception messages which are indicators of data issues, they are still prevalent as are production order variances due to incorrect part cost and detail assignments.

The organisation has a large number of other systems (over 1000), which have little or no configuration standard or enforced usage standard. There are some local data standards such as naming conventions, however, these cannot be globally enforced due to the inability of the large variety of systems across the company to adopt them along with the lack of willingness of the separate business units to adopt them. This is largely a legacy issue and standardisation programs for the systems versions, configurations and data standards are underway to improve the situation, which relies on business engagement with the central 'Centres of Competence' that are driving these standards. A community of 'Capability Leaders' is used to guide the development of new systems but their availability and authority is limited due to the organisation size and structure leading to control being a significant challenge.

The organisation is working towards component feature standardisation within its PLM system; these features are then to be used as the hub for related feature meta data including axioms (context driven limits) and attributes (costs and capabilities) which can be linked to manufacturing method data through the Computer Aided Process Planning system (part of

the corporate PLM system). There are a number of projects developing standard feature approaches within the organisation, working with partner companies such as ITP in Spain and Darmstadt University. The fact that there are disparate approaches to feature standardisation indicates the size of the challenge to developing standards across an organisation of this size.

3.3.2 Ontology

The organisation is very aware that it lacks a common ontology across its sites, businesses and functions at the fundamental product levels of communication. The same fundamental manufacturing documentation within the one business unit is referred to as; Manufacturing Instructions, Technical Instructions, Data Cards, Standard Operating Procedures, indicating the depth of this issue. Key Business metrics such as Right First Time, Scrap, Yield, Product Cost, Productivity are calculated in subtly different manners using inconsistent source data; while there are standard calculation definitions the terms of the calculations have different meanings in different areas. Automated reporting of these metrics e.g. feature 'right first time', without an effective heavyweight ontology to provide consistency has led to difficulty comparing measures across the company. Even within the core company systems there are domain semantic issues; material is an attribute of a product e.g. titanium, as well as a reference to a product assignment to a customer order within the ERP planning system e.g. 'material requirement planning'.

3.3.3 Legislation

Many of the components and technologies used by the organisation are considered to have sensitive or military application and as such they are subject to export control. As is common in high technology industries, the organisation relies heavily on its intellectual property and technology to maintain a competitive advantage. Failure to manage data and information systems and processes properly could have serious legal and commercial ramifications. Control cannot be achieved by total isolation or exclusion, as 80% of the organisation's manufacturing takes place in its external supply chain, as does a significant amount of its design engineering. Its total supply chain is truly global requiring real time integration between offices on different continents. Until recently the only mechanism for control of data were manual combined with standard security approaches such as secure networks and offices, however, the advent of the global PLM process has necessitated far more complex mechanisms. The organisation's PLM system segregates data between engine related groups which have varying levels of access control and for which all individuals need to be explicitly added for access. All data is subject to a 'Default Non Exportable' rule which ensures the natural state of data is that it cannot be removed from its native database instance. The only way to export data from an instance is to apply an export project to it. The

application of export projects is a manual process requiring knowledge of the export control levels for any dataset, with the ability to apply projects being a restricted privilege. Applying export and security meta data to each data set allows the global PLM instances to communicate and share data in a secure way. It is important to note that this raises significant levels of transactional bureaucracy, is still fundamentally reliant on individuals classifying every dataset correctly and has led to architectural inflexibility which has affected the performance of the system and its processes: The organisation has a PLM instance in Derby and another in Bristol along with other instances including North America, Germany and India. However the data control protocols that are required to comply with legal requirements cannot differentiate between instances in the same country and those in different countries, meaning that data has to be formally exported in the same way between Bristol and Derby domestic engineering teams. This has proven to be a significant burden and barrier to the implementation of the PLM working principals and practices. It is important to note that with significant compromises a level of control has been applied to the PLM system within the Siemens Teamcentre environment but this has not been achieved with other systems and the strategy for achieving that level of control is to bring more data and functions within the Teamcentre environment, making them subject to the PLM change control processes. This level of change control may not be appropriate for all data and information types and may become an unnecessary functional constraint due to the systems inability to identify and manage every data type in the most appropriate way.

A key aspect of the Civil Aviation Authority (CAA) and European Aviation Safety Agency (EASA) requirements for ensuring safety is that the method of manufacture and component validation history is retained and traceable. It is also a requirement that critical components are serialised and have full histories allowing any component to be traced back through the supply chain and allowing the identification of operators, inspectors and engineers that have made decisions affecting the parts as well as the operation sequence and resources used. Depending on the data and component type, this information may have to be stored for a fixed period or for the life of that engine or type of engine plus a fixed number of years. Unless this history is available for a component it may not be used. The engine configuration, component history etc within the organisation is stored in a number of systems including:

- SFDM – production and shop floor quality and yield history
- SAP – Manufacturing method and processes and finance data
- PLM – Product definition, validation and configuration
- PDM (Metaphase) – Product configuration.

- Archived Paper/ other media Documents – Component X-ray images, production documentation.
- Other electronic archives

This list is not exhaustive, even so there is clear overlap and duplication of systems and data and the interoperability of these systems is very limited. Maintaining the robust controls required for legal requirements across systems is an area requiring significant further work.

3.3.4 Organisation

The investigation focuses on a large and diverse organisation; its separate divisions and the disparity in division size can make balanced cross sector decision making complex. Many of the company systems are governed and configured by the Engineering and Technology (E&T) organisation which has to make the judgement as to whether all sectors or divisions should be given equal ranking when balloting changes or proposals. The organisation has grown significantly over the last 20 years, the level of growth across its different business sectors could not be achieved through organic growth alone so it has been accelerated by strategic acquisitions. The organisation has a number of Joint Ventures around the world as well as strategic embedded suppliers of both products and services Due to this conglomeration over time of many organisations, there has been an accumulation of many systems carrying out the same function and in some instances different configurations of the same system all of which will be on very different architectures.

Internally the organisation operates as a matrix organisation and uses functional ‘Centres of Competence’ within the global ‘Engineering Improvement Centre’ (EIC) to drive standardisation across the organisation. As a separate entity from the business units, it has a limited ability to enforce compliance to standards and methods and more fundamentally as a function its ability to understand the wider business requirements is entirely dependent on the engagement from the business units. The EIC function largely operates from one site in Derby and its level of influence can be seen to dissipate across organisation, functional and geographical barriers as shown in Figure 3-3.

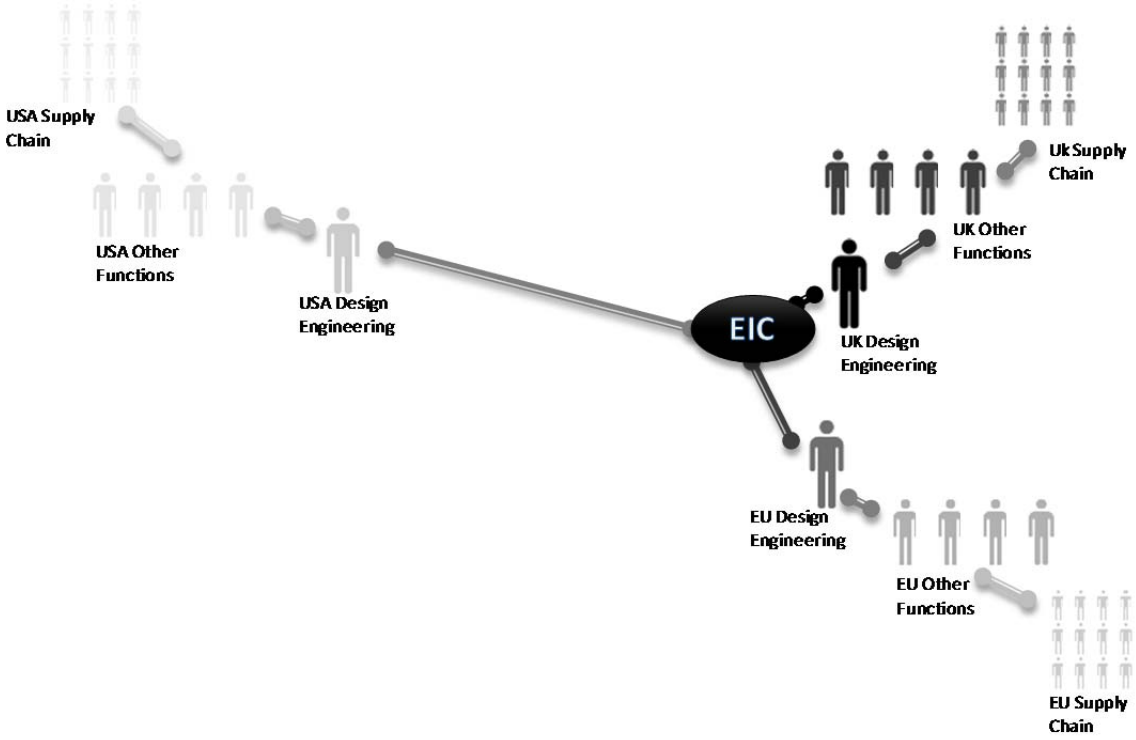


Figure 3-3 – Representation of the global and cross functional influence of the Engineering Improvement Centre (darker tone = stronger influence)

The organisation is world renowned for engineering excellence, and because of its history, reputation and core business it has a strong design engineering focus; 'Engineering' in the organisation is typically an abbreviation for 'Design Engineering'. This can manifest as an imbalance in some areas of understanding of the wider business requirements: where cross functional corporate systems such as PLM are being deployed, the balance between the traditional design and product configuration engineering activities requirements capture and the Manufacturing and logistical etc, may not be optimal.

The size and complexity of the enterprise, which is exacerbated by the acquired diversity along with the strong lead provided by the EIC organisation has resulted in a general strategy for information system interoperability of bringing all data and systems within the bounds of one unitary system; in this case PLM. This has also developed an approach to the deployment of PLM methods that does not always distinguish between the PLM methodology and the PLM core system (Teamcentre) i.e. as general assumption that if 'it' is not in Teamcentre 'it' is not PLM compliant.

3.3.5 Systems and technology

As previously discussed the organisation has a large diversity of acquired and developed systems. Many of its legacy systems are highly customised to meet specific requirements; this includes the legacy CAD system CADDS5 which was heavily customised by

Computervision (the original vendor) to meet the organisation's complex design and parametric modelling techniques. This customisation resulted in an inability to accept further upgrades to both hardware and software which in turn led to increasing difficulty to maintain the system and its interfaces with other systems that were changing such as the analysis systems that used the CAD data it produced.

The organisation uses a very wide range of versions of hardware and operating system; it maintains a basic capability to access data created on legacy systems such as VAXs systems and mainframes due to an inability to migrate these systems and data onto new platforms. Figure 3-4 shows the strategy to move from this position of significant complexity to one where the business uses a small number of unitary systems. The business is expending significant time and resource investigating how to move all its data into either a single, or one of a small number of databases.

The organisation's experiences maintaining a number of smaller systems without any clear interface standards, has shaped it approach to systems development. It is focusing on simplifying the management of data in one context at a time; the PLM system is being developed to manage product data, the ERP system manages enterprise planning etc. What has not yet been resolved is how to deal with the use of a unitary system across many domains. The functions within these domains will have different perspectives on the data and information within the systems due to the lack of a common and standard lexicon of terms or ontology. Without the ability to definitively describe the data and information within a system in an unambiguous way the reliability of the decisions taken upon the information will be dependent on whether the person making the decision happens to share the system designer's view of the data e.g. if a manufacturing engineering process owner sees a 'yield' figure in the ERP system of 70% against their process, he or she will assume that 70% of the parts processed will be conforming whereas a MRP controller will assume the cumulative yield of the process route to that point will result in a 70% yield. This relatively simple difference could result in excess WIP, which if replicated across the company would cost approximately £250M.

As well as pursuing a strategy of large centralised systems the organisation has also implemented strict controls on the development of systems dictating that systems can be configured in line with agreed internal standards that are supported by the OEM but must not be customised. Rather than customising the products after purchase, it is partnering with the systems suppliers i.e. Siemens, Hewlett Packard and SAP to influence the development of future product releases.

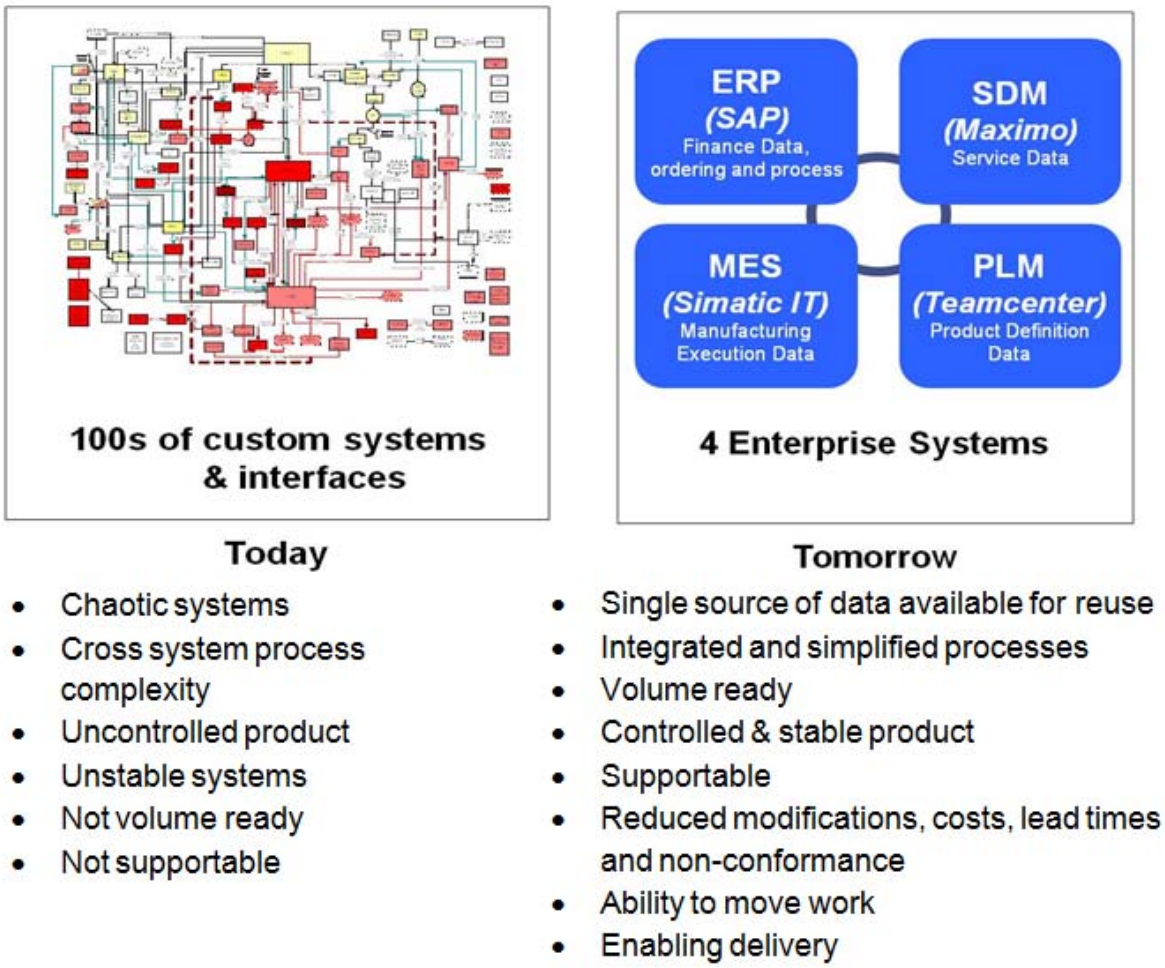


Figure 3-4 – The systems convergence strategy (provided by the organisation's Systems Executive)

3.3.6 Data, information and knowledge maintenance

The organisation has a strategy for the convergence of its systems (Figure 3-4), which is recognised as a long term endeavour. The current situation is that data and information is duplicated in many systems, in many cases using separate and un-correlated domain ontologys. This makes it difficult to identify or resolve this duplication and also makes it almost impossible to ensure the data does not diverge as it is updated. Significant levels of resource are being focused on this issue, and it recognises knowledge and information management as a key business function, and as such there are specialised functions within the organisation leading the development of its information management capabilities.

A common anecdotal example of data duplication and maintenance issues within the organisation is the maintenance of personnel listings: this organisation data is stored in the corporate web based HR Online system, as well as the legacy store in the People and Knowledge module of the ERP system SAP. This data is also in the corporate Excel based load and capacity tools, training planning tools and skills assessment tools as well as a

number of other systems. Some, but by no means all of these systems use a manual download of the data from either HR Online or SAP. However many rely on repeated manual entry. Virtually all these systems require manually auctioned or at least manually triggered updates. There are also numerous systems which use subsets of this data filtered by meta data such skill type, resource area or function. Unfortunately these meta tags are not standardised, globally available or populated in a standard way by the contributing functions across domains. The result is that this data is manually entered into many systems and maintained in a divergent manner leading to inconsistent data. A number of managers were approached and none felt that a request for this organisation data from the HR or Business functions would result in a consistent or accurate result. All stated that they would recreate the data from local knowledge rather than correct data stores being used. None of these managers would consider entering a request for a complex subset of this data as they did not believe the data was suitably structured or the meta-data available. If this anecdotal case is representative of other systems and processes there will be a massive benefit from central repositories that are suitably structured and accessible to allow different functional perspectives to interoperate with them.

In the areas of the business that do have a single authoritative data source e.g. the PDM system, change is managed by rigorous change control boards ultimately chaired by one or two individuals. This level of constraint is required due to the criticality of these systems but this does make them inflexible. It could seem that all of the change mechanisms for these systems are there to prevent uncontrolled change, however, there do not seem to be any mechanisms that promote or aid change ultimately resulting in lower organisational flexibility.

The organisation's legacy systems have generally been used as data stores, where data is manually managed, with varying levels of access and change control. These stores have little interoperability or cross referencing capability and what integration there is, is hard coded so that any changes to the data store structure will corrupt the communication mechanisms and any related reporting solutions.

The organisation has recently initiated research into knowledge maintenance which should help inform its future systems strategies. It is also developing and deploying web based centralised reporting solutions using the COGNOS system (see Section 3.3.8) which should provide far greater flexibility by providing an over-arching reporting mechanism.

3.3.7 Infrastructure

For over a decade the organisation has operated a core business network infrastructure. This infrastructure was originally targeted only at the Aerospace sector of the business and so the coverage is incomplete. This infrastructure is secure, robust and fully supported by

the company's IT solution provider Hewlett Packard including security and recoverability. The organisations sites around the world are not all directly linked by a Business LAN, albeit some of the key systems do have instances around the world that can communicated such as SAP and PLM. Direct connection across nation borders is complex due to the legal requirements to control the flow of knowledge and information. The organisation does operate a collaboration tool called 'ForumPass' which is authored and supported by one of its subdivisions company 'Exostar'. Forumpass is a web hosted, secure data store where users can create pages similar to those of social networking sites. These pages can be linked and used as message boards, balloting tools, public document stores, information 'wiki' sites etc. The system also provides the capability to launch teleconferencing 'Webex' sessions where users can remotely view and share files and even edit them collaboratively. While this toolset is useful for the manual exchange of discrete data and information, it does not provide a mechanism for a seamless and automated flow of information, meaning it is best suited to specific discussions on a focused topic and leading to the risk that it is incomplete or out of date. The organisation’s PLM programme has a work stream dedicated to collaboration, and it is through this system that the exchange of technical information is planned to be enabled and controlled. The company also has a central "IT Infrastructure" organisation which splits its scope into 6 work streams:

* Client Infrastructure
* Compute, Service Integration
* Network
* Factory and Facilities
* Programme Planning

However this function has a limited opportunity for interaction with the wide range of business units, which do not have their own IT functions with which to interface, and so the business communication flow in to this organisation tends to be centrally and functional led rather than holistic. There is also a risk that a central function such as this, with a depth of understanding of its area of work and the legal and technical requirements, may develop solutions to meet the technical infrastructure challenges without fully understanding or developing a global requirements vision, resulting in a centrally mandated solution that does not match the business expectations.

The organisation has historically operated local factory shop floor networks for the delivery of CNC programs and basic data capture which have not been connected to the business

infrastructure. These legacy systems were non standard and little more than cables and switches with little or no security. This 'fire break' approach has been necessary due to the complexity and lack of standardisation of the machine systems on the shop floor which due to their age and customisation are unstable or insecure. Connecting these systems to the businesses LAN infrastructure would be an unacceptable security risk and where this has been done has proven to jeopardise the stability of the business infrastructure. Over the last 5 years the organisation's Manufacturing Systems Centre of Competence has developed and deployed a standard shop floor technical 'Tech LAN' infrastructure which is still plant based but provides an improved level of security (class 3 or unsecure) at a local level, and also provides the ability to connect to the business LAN via an automated communication bridge using the Microsoft MQ to mirror files through a firewall which prevents manual transfer of data between the two networks. This is currently used to allow the DNC systems to store the master programs on a the secure business LAN where they can be recovered in an emergency and to pass them through a firewall using Microsoft MQ to a shop floor technical LAN to be deployed to the machine tools. While this is a significant improvement it means MQ is a single point of failure and the communication mechanism between the business LAN and Tech LAN is limited to the automatic transfer of files within specific folders. This is sufficient for the transfer of data but significantly constrains the transfer of information and knowledge as it is difficult to pass context and conditional informations with the data. The MSCoC is therefore starting to deploy an architecture called Tech LAN 2 which is a Class 2 network which allows dynamic interoperation between the 2 LAN domains as it is secure and supported. Tech LAN 2 does, however, push this requirement for security and standardisation further out onto the shop floor meaning the individual clients either need to be of an approved standard with sandard Virus software (F-Secure) and standardised builds with access control, or they must be connected to a buffer client which in turn is connected to the Tech LAN. Tech LAN 2 has also moved from using geographically dispersed or 'local' tech LAN servers to using a large campus server for the entire site. There is discussion regarding the use of this server for the within-region sites e.g. the whole of the UK, however even with high availability architecture design, it is felt by some business units that this represents a too significant business continuity risk. The move to Tech Lan 2 is creating a massive standardisation workload and cost and relies on the engagement of the local business unit, but the ongoing supportability and benefits on centralised systems lifecycle management will easily offset this cost.

3.3.8 Architecture strategy

As discussed earlier, the organisation has a clear strategy to reduce the number of systems interfaces. Figure 3-5 shows how all the elements of its PLM system are intended to be joined by a common backbone allowing the interaction of the elements of the PLM environment such as CAPP, CAD, CAM and data management. The interoperability of these PLM elements will still be dependent on the use of an ontology or ontologies across all

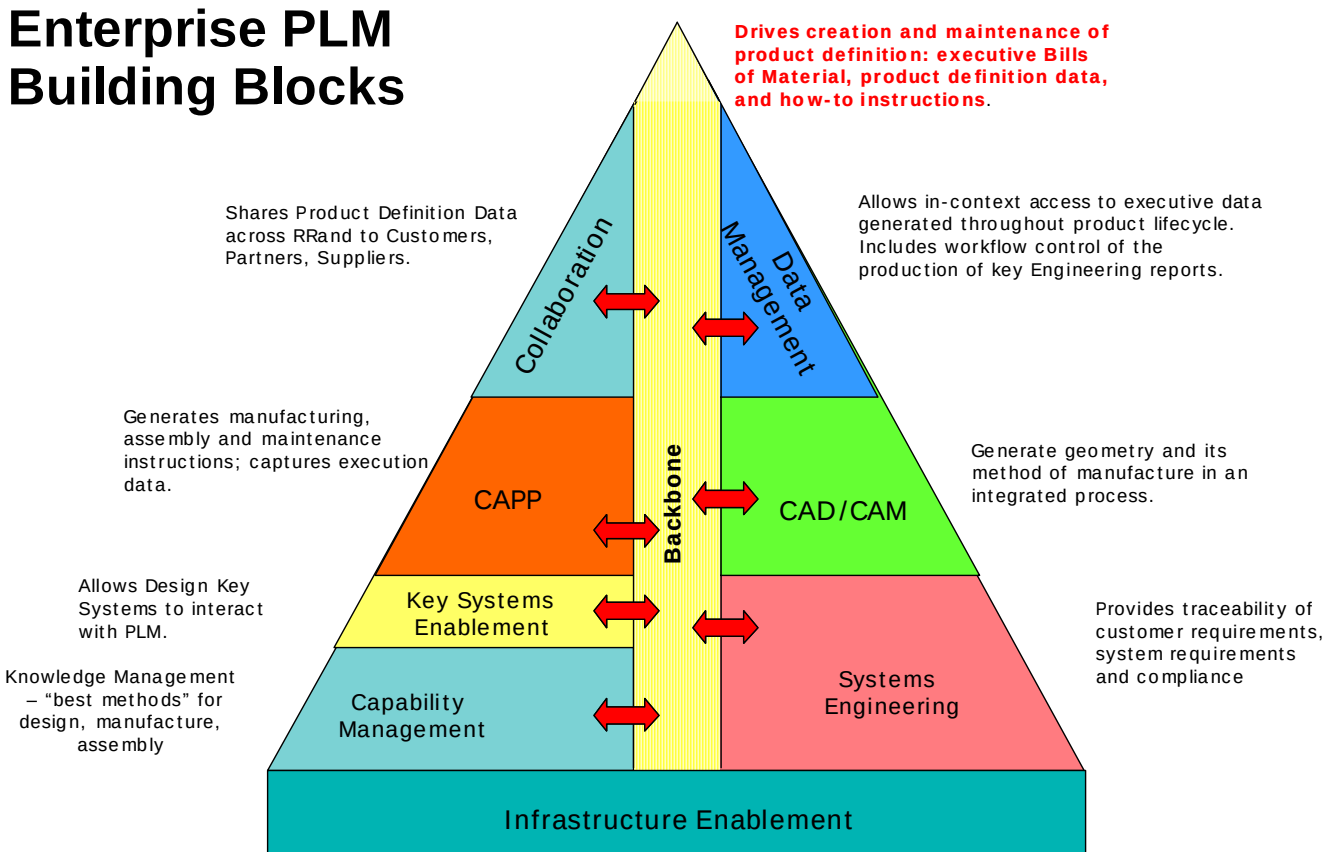


Figure 3-5 - Representation of the PLM 'backbone'

elements that will enable the unambiguous and automatic communication.

While Figure 3-6 clearly shows the organisation's strategy to move to a few well supported, carefully configured solutions. The organisation's Information Management Centre of Competence has identified a requirement to provide flexible and rapid reporting which is not constrained in its mode of data manipulation as it works on data extracts from authoritative systems and is configured in such a way as to provide a powerful, fast, cross system manipulation and reporting tool. It is to meet this need that it is developing its COGNOS solution which will be able to extract data from any of the organisations systems, and reconfigure it in to a star schema for rapid manipulation. While this system will still be limited by the effectiveness of the data standards, meta data availability and use of a common ontology, it effectively provides a matrix approach to systems architecture; the direct,

rigorous systems to system interfaces between the unitary, authoritative systems and some smaller satellite systems, and separate flexible reporting capability capable of extracting from any of the systems and manipulating the data as far as the data and knowledge content will allow.

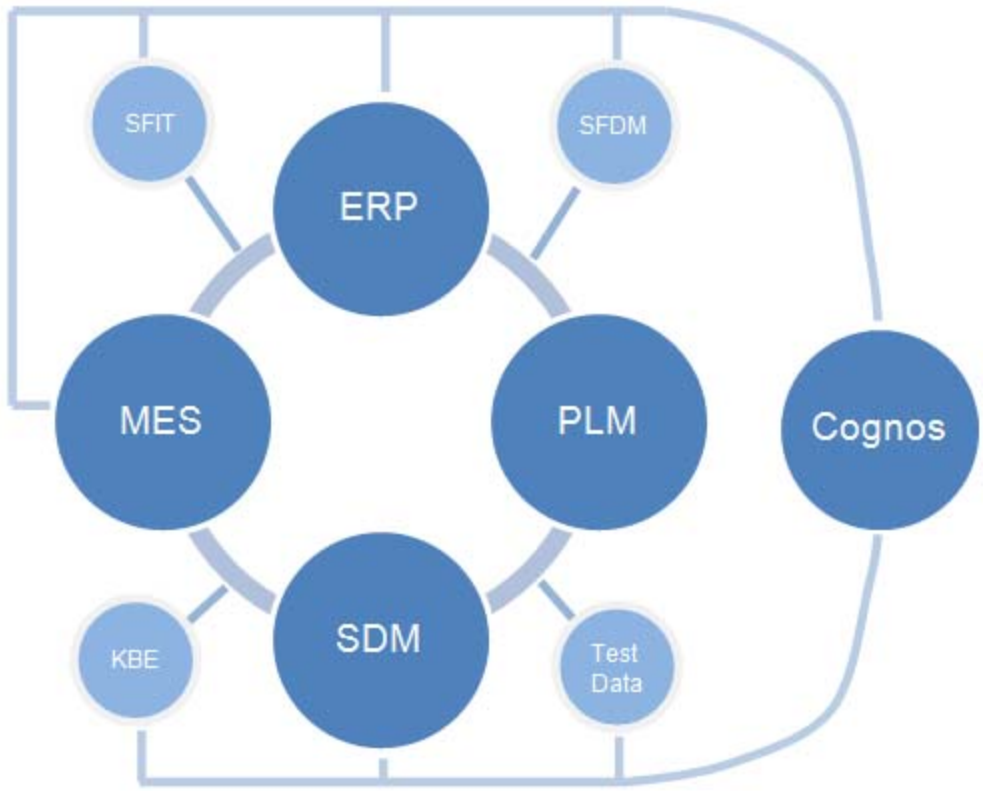


Figure 3-6 – The Organisation's core and overarching-reporting architectures.

3.4 Key declarations

3.4.1 Process, systems and information

The activities carried out within the Manufacturing domain, and the wider enterprise can be described as processes in that they take inputs and modify them, using resources and constraints, into the required outputs as shown in Figure 3-7. This process description can be used to define all physical and transactional aspects of an enterprise.

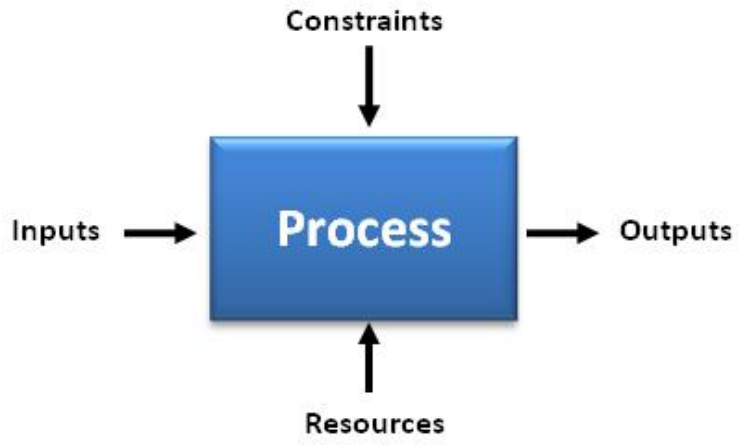


Figure 3-7 - Basic description of a process

'Systems' in the context of this work are instances of a process e.g. production scheduling is the process of interpreting the business output requirements based on customer demand and planning the required production launch schedules based on the manufacturing lead time and operational capacity. The production scheduling system is the system within a specific enterprise that carries out of supports this process using the defined instances of input data and constraints. Conversely a process can be considered the function of a system. As can be seen in Figure 3-8, using this conceptual linkage between a process and a system, a process can be represented using the same IDEF0 style representation (FIPS 1993) that would be used for a system (see Section 2.7.6)

The non-physical inputs and outputs of a system will take the form of information, as any input data would need appropriate contextual meta data applied either implicitly or explicitly, to ensure it is used appropriately. The rate at which systems take in, process and then output information can be considered their 'processing rate'.

The information which is input and output requires a structure. This structure allows the system to know where to find specific elements of the information and is necessary to ensure consistent system function or process.

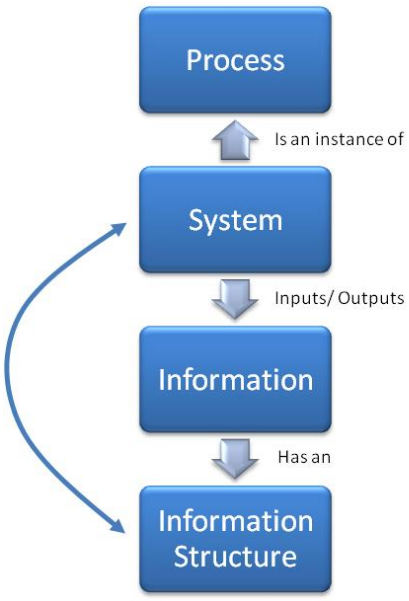


Figure 3-8 - The relationship between process, system, information and information structure

This description of a system as something that carries out a process using information which has a structure implies a connection between the system used and the structure of the information being input and output.

3.4.2 Process levels

The process within an enterprise can be defined as falling within 3 levels which were aligned to the ISA95 organisation levels (see Section 3.5.1) :

- 1st Order Process are those that directly interact with the Manufacturing Operation e.g. Maintenance, Logistics, Product, Process Improvement.
- 2nd Order Process are those that may interact directly with the Manufacturing Operation as well as with 1st order perspectives e.g. Research and Development, Marketing and Sales.
- 3rd Order Process are those that interact with 1st and 2nd order processes but not directly with the Manufacturing Operation e.g. Enterprise Executive, Supply/ Value chain Planning.

The first order process take operational information and use it to perform their function; as long as the information content and structure is as expected and the process functions as specified the output should be valid. Second order process function similarly to first order processes however they also take the output from first order processes. This introduces the risk that any issues with the structure or content of the first order process or the operational information can undermine the validity of the second order process. This risk is further compounded in third order process. This highlights the risk that the further removed from the

source operational information and process is, and the more that information is processed prior to input into a process the greater the risk of process inconsistency.

3.4.3 Rate of change of systems

The information processing rate of a system, shown as the organisational level timescales in the Section 3.5.1, and the rate of change of the system are linked to each other and the organisational level due to:

- The level of complexity and risk
- The level of regulatory impact
- The lead-time due to training, testing and development etc.

As the system and information structure are linked the rate of change of the system is linked the rate of change of the information structure.

3.5 Understanding the complexity of MI systems interoperability

3.5.1 Disparate organisation level timescales

An organisation can be defined as comprising three levels (see Chapter 2):

- Enterprise Level: At the highest organisation level, Enterprise Systems help inform and enact Enterprise wide coordination and strategic processes. These business processes involved tend to be long – in many cases greater than a year.
- Execution Level: At the middle level, the Execution Systems help inform and enact the process that result from the Enterprise processes. These processes tend to be either geographically or functionally limited and work on timescales of hours to weeks.
- Shop-Floor Level: At the Shop Floor level, the systems generate fast paced information about the operational processes as well as enacting control over the process with timescales ranging from milliseconds to minutes.

MI information is created and used across all levels of the organisation with the rate at which information is processed differing at different levels due to the rate of information input or availability and the process timescales. It is important to note that the conventional definition of MI systems describe them as bridging between the Execution and Shop-Floor levels, whereas this research resulted in and uses a definition where parts of the MI system can reside at any level of the organisation.

Figure 3-9 shows examples of the systems that may operate at the different levels. This figure is a simplification, as systems may support multiple processes which may span operational levels e.g. a CAD system can support either a drafting process (Execution Level) or PLM design process (Enterprise Level). While this complexity has been removed from this figure it is important to be aware that systems may support a number of processes across a number of these organisational levels and hence may support a number of different process timescales and processing rates.

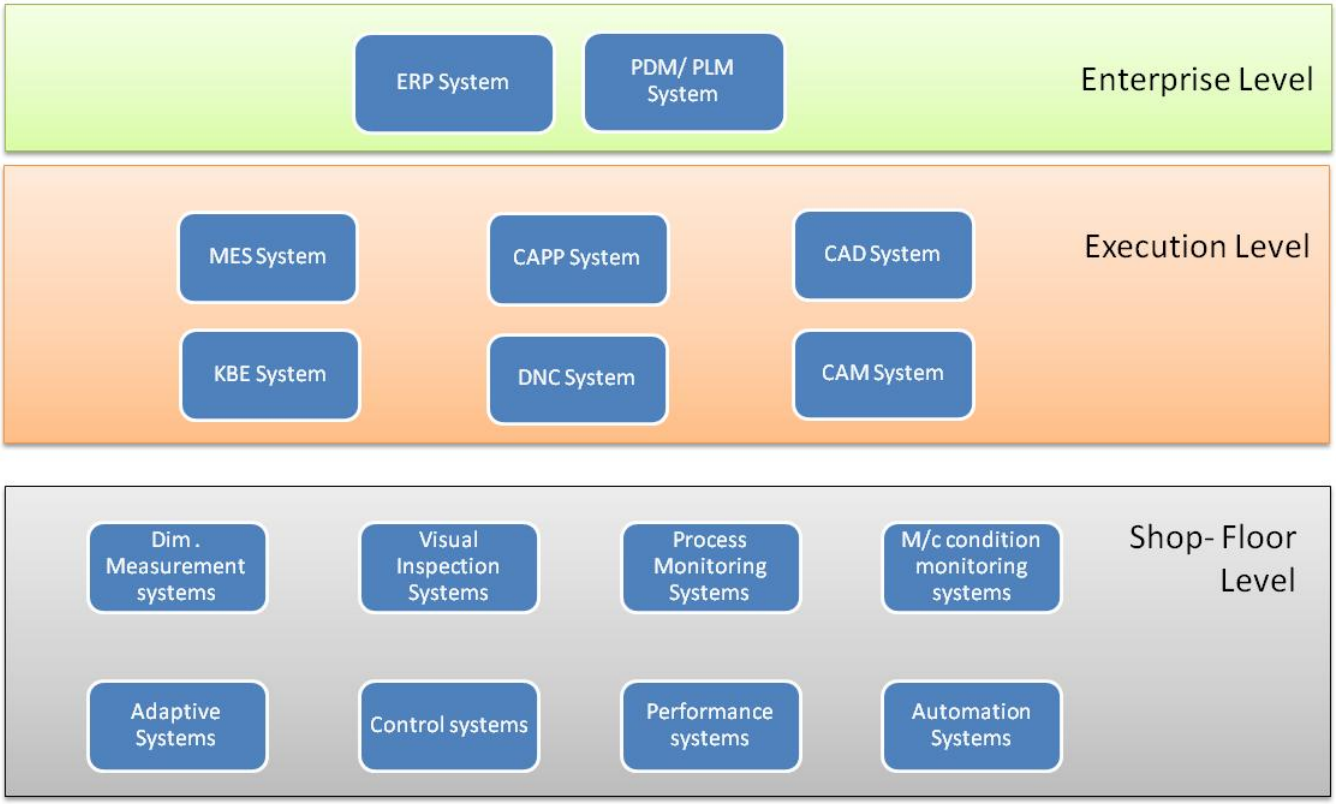


Figure 3-9 - MI systems at different organizational levels

Figure 3-10 shows the alignment of Process Levels (see Section 3.4.2) and Organisational levels and shows instances of information at each level: Performance Metric information types have been used in this example.

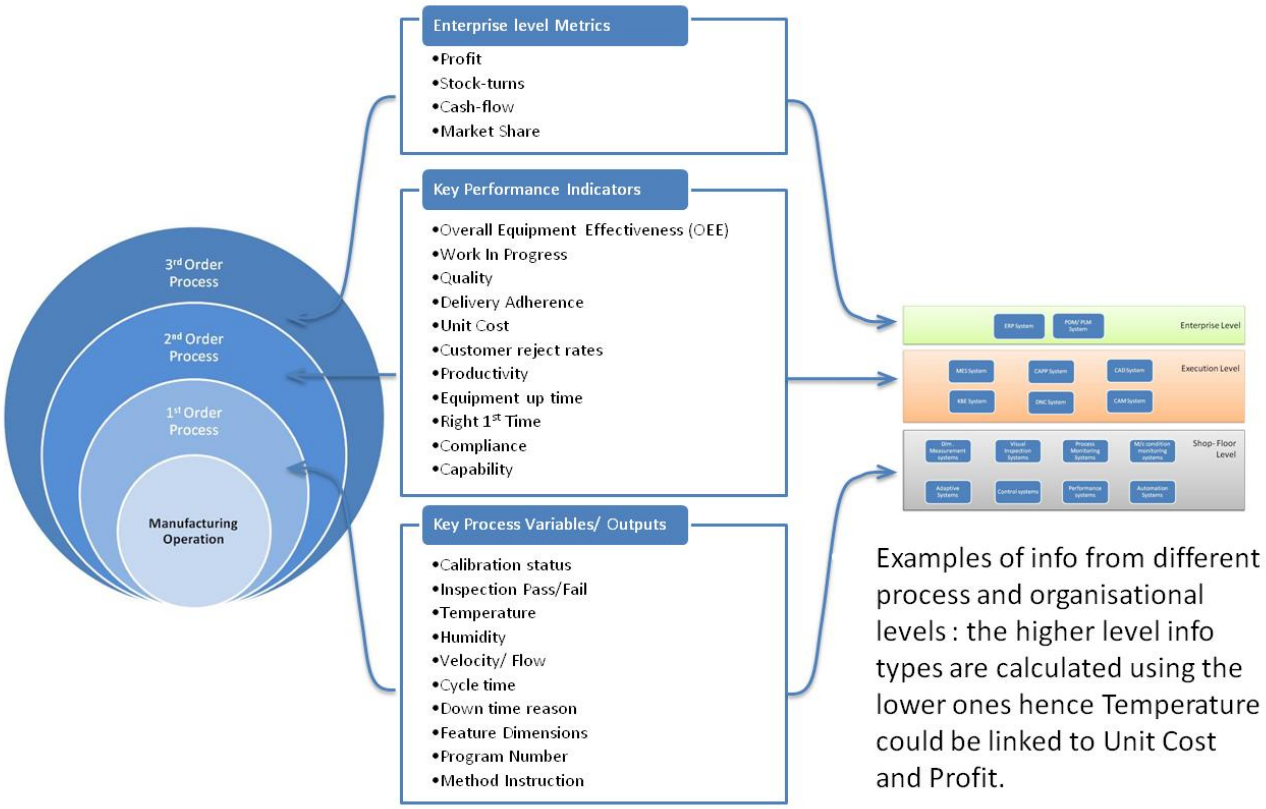


Figure 3-10 – Example instances of information at different organisational levels

3.5.2 MI system information flow

The flow of information between MI systems is highly complex, crossing functional, organisational, process and geographic domains. Figure 3-14 to Figure 3-14 show three examples of flow individually and then combined to show the complexity given just three instances of information type. These examples are:

- Product Configuration Information
- Product Variation to Specification Information
- Product Geometry Information

These information flows cross the organisational levels and therefore connect processes and systems which are operating on different timescales (see Section 3.5.1). Process and systems operating on shorter timescales which require inputs from those operating on longer timescales have to assume that that input information remains valid between updates. In some cases these faster operating process and systems feed and subsequently receive

information back from slower process and systems. This results in information latency despite the faster system providing the information at the required rate. This latency can be caused by any one of the many and disparate inputs which are required to generate a valid output. In an operational environment it is usually unacceptable to stop operations to wait for this latency, therefore at any time a number of systems may be operating with inputs which are out of date.

These figures allow a simple description of the full complexity of MI information flow given only 3 examples are shown and the potential numbers of information instances.

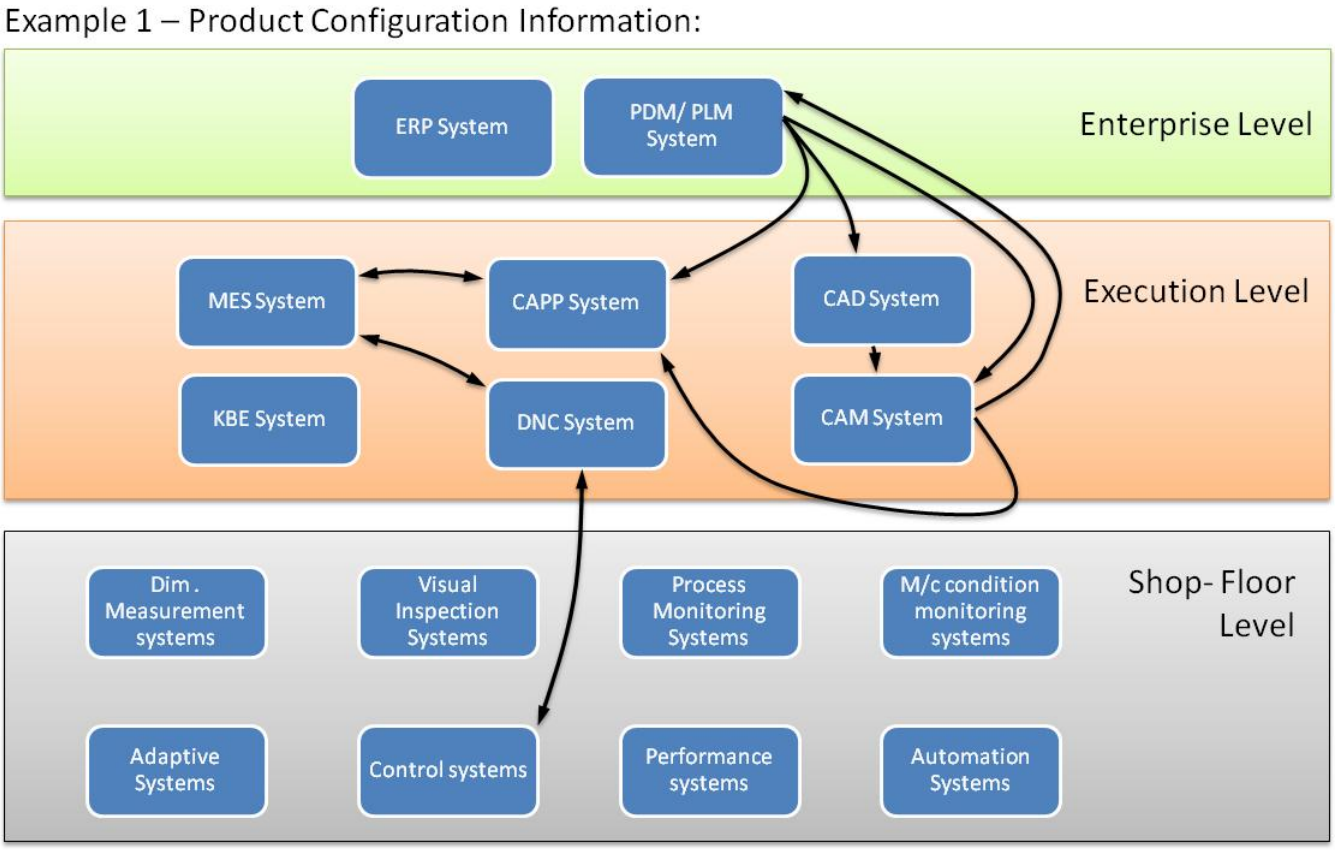


Figure 3-11 Example of Product Configuration information flow

Example 2 – Product Variation to Spec Information:

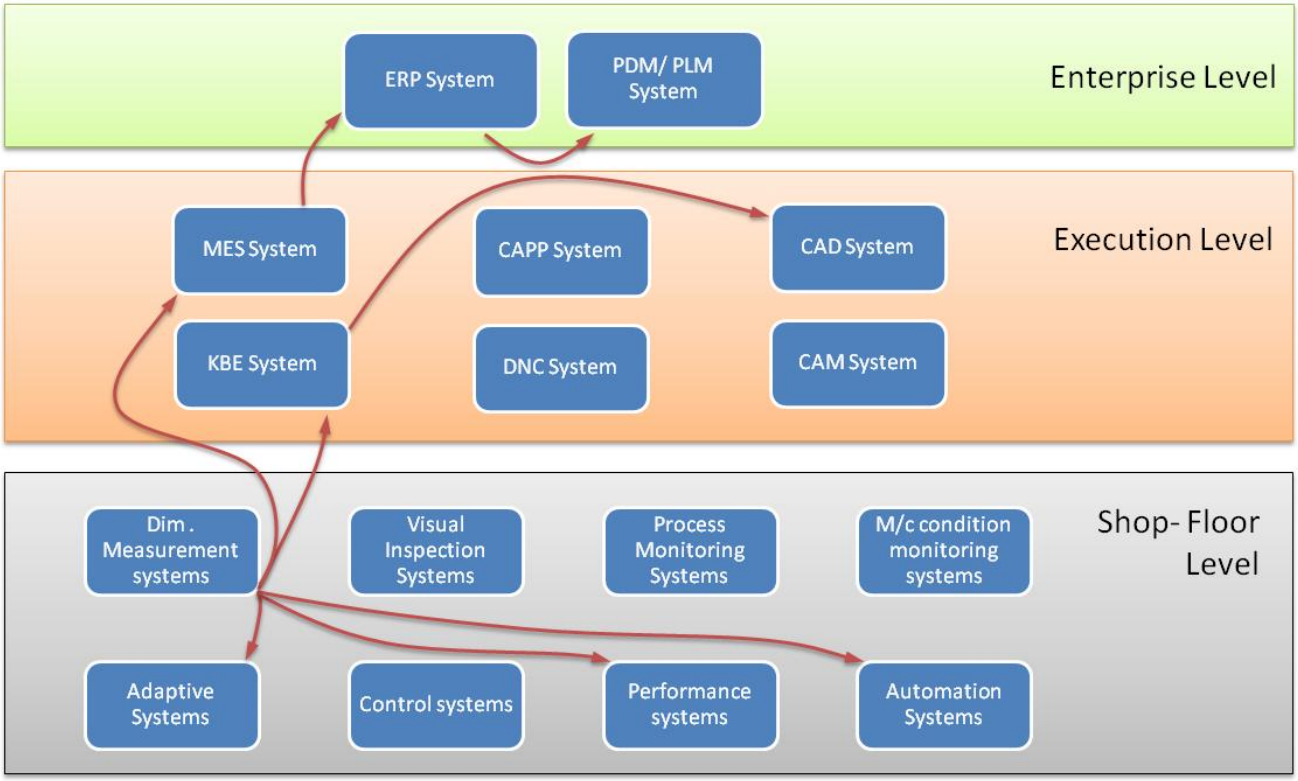


Figure 3-12 Example of Product Variation information flow

Example 3 – Product Geometry Information:

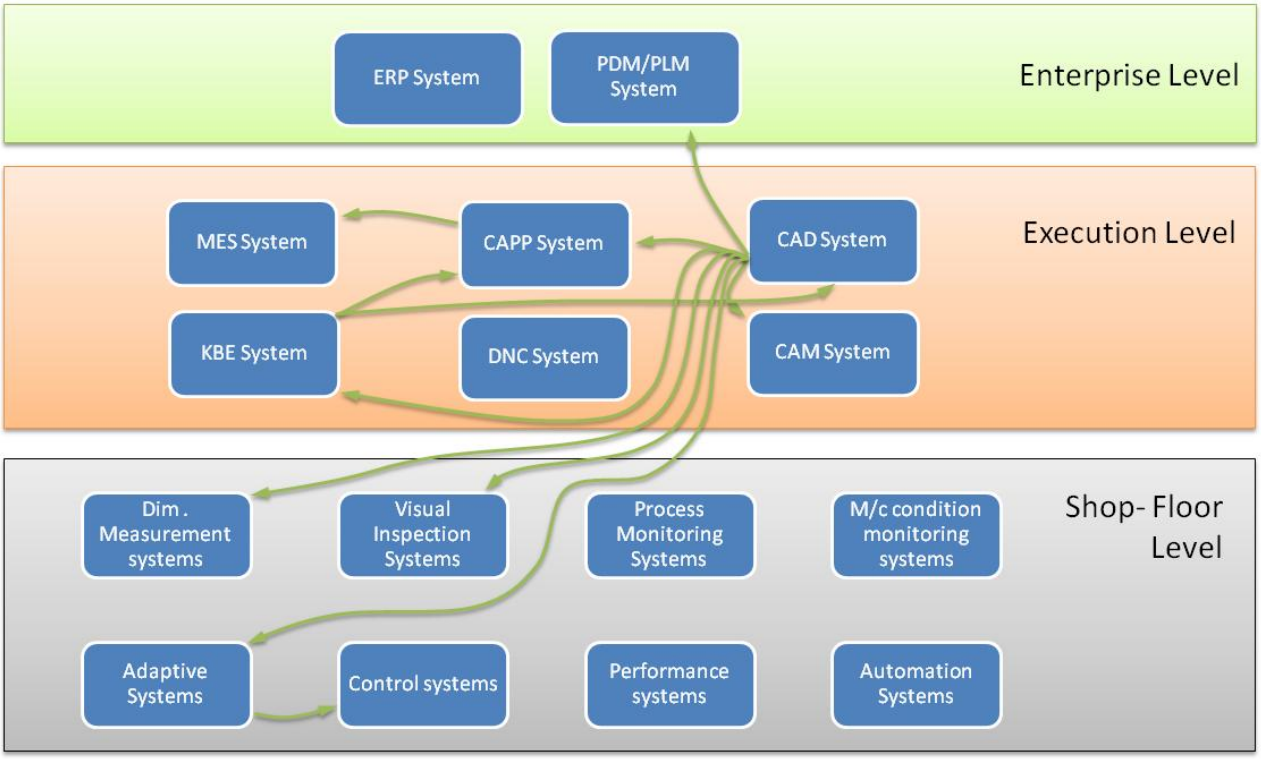


Figure 3-13 Example of Product Geometry information flow

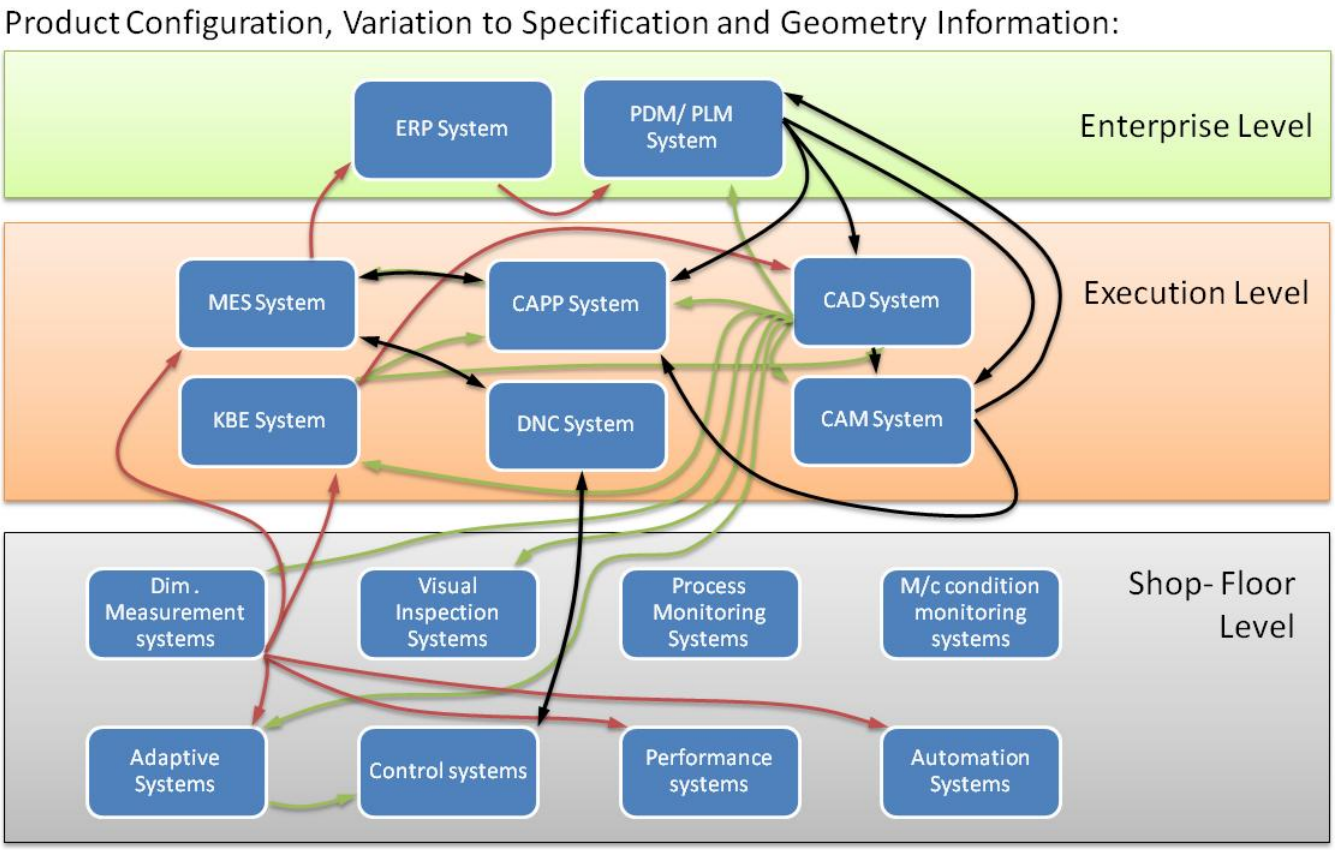


Figure 3-14 Combined, confusing information flow

The information that is input to these systems may be modified and output in a number of ways:

- With the content changed but the structure the unchanged
- With the content and structure changed
- With the content unchanged and the structure changed

3.5.3 Manufacturing intelligence information structure

As previously discussed the individual MI systems within an enterprise take in disparate information types, use it to perform their function or functions and then output information. These input and output information flows have a structure. The information structures may vary even within a particular information type. The same information type may be output from the system with the information structure changed or unchanged. It is also common for one information type to be processed and used with other information types to create a new type with a hybrid structure.

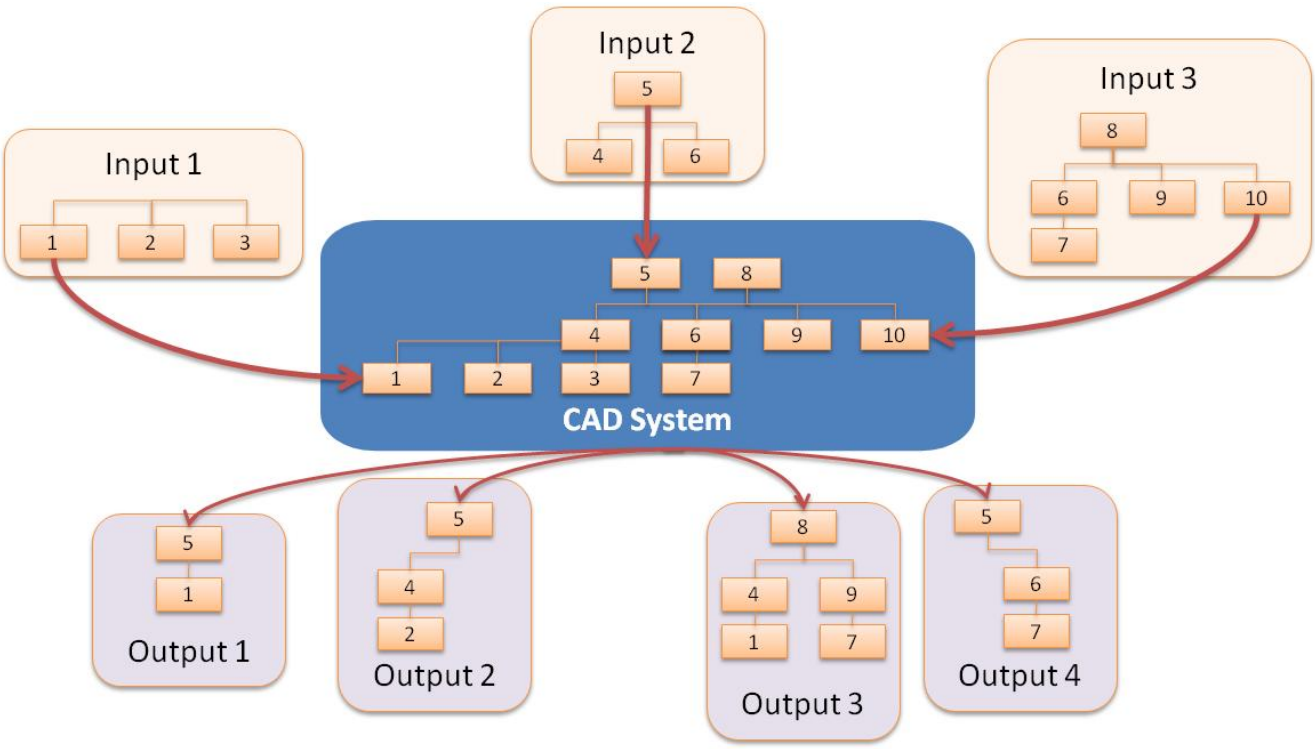


Figure 3-15 - Manufacturing information structure changing as it passes through a system

Figure 3-15 represents these data structure changes, with data elements 1 to 10 being input into a CAD system from 3 different inputs with their own unique data structures, which are represented by the links between the numbered data elements. The CAD system takes these data elements and constructs its own internal data model. The CAD system is then able to output the same 10 data elements in different data structures using the CAD system data model. In this example, data element 1 is input in a structure which shows a parallel relationship with elements 2 and 3. The CAD system data model has been designed such that it knows there is a relationship between element 4 in input 2 and the elements in input 1 as shown in the CAD system data model. Using this larger model the CAD system is then able to generate an output which contains data elements 1 and 5 in a valid structure inferred from the overall CAD systems data structure. In this way, data element structures are input, combined, inferred or transformed and then output from systems.

3.5.4 Inconsistent manufacturing systems definitions

The use of inconsistent semantics is a significant risk to MI systems. Figure 3-16 uses the industry standard Overall Equipment Effectiveness (OEE) metric as an example of inconsistency between two geographic locations within an organisation:

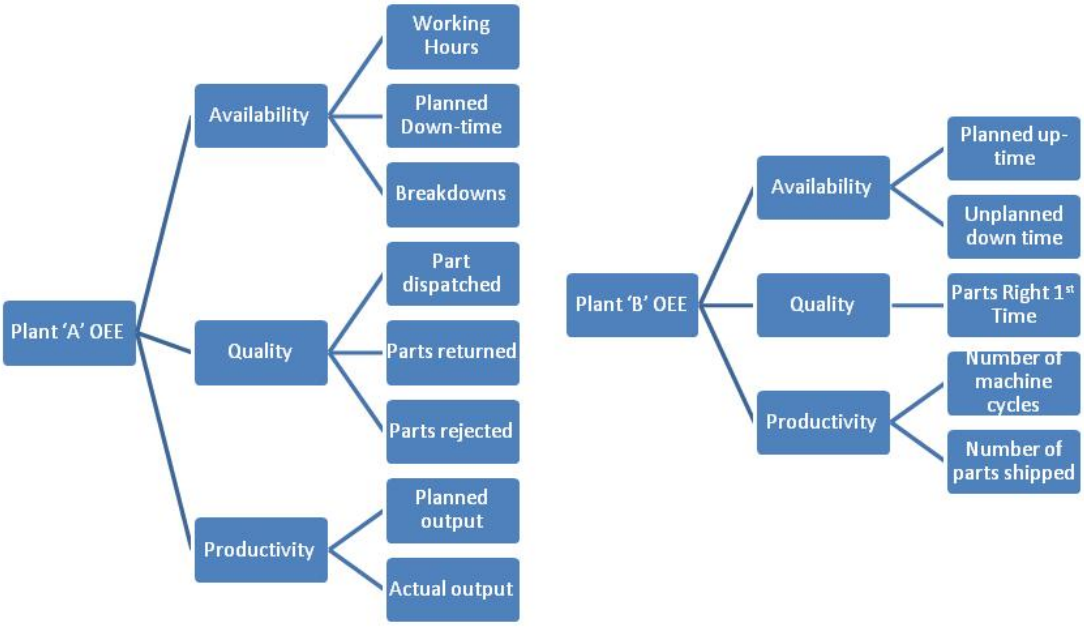


Figure 3-16 - Inconsistent MI system definitions: OEE example

Both operational plants use the industry standard definition of OEE: Availability x Quality x Productivity. However, each of these sub-metrics can be calculated using different source information. Each of these methods is valid and the information used will be driven by the data available within the plant. The result will be that the OEEs in these two plants will be inconsistent and incomparable. If decisions are made based on this comparison e.g. sourcing decisions based on plant effectiveness, the outcome will be unpredictable: if it is assumed that parts are being sourced at the most effective plant, but that plant turns out to be less effective, there could be significant impact on overall enterprise effectiveness and profit.

3.5.5 Inconsistent MI definitions through time

The previous example of inconsistent MI definitions (see Section 3.5.4) can be adapted to show a comparable issue with consistency through time rather than across and organisation:

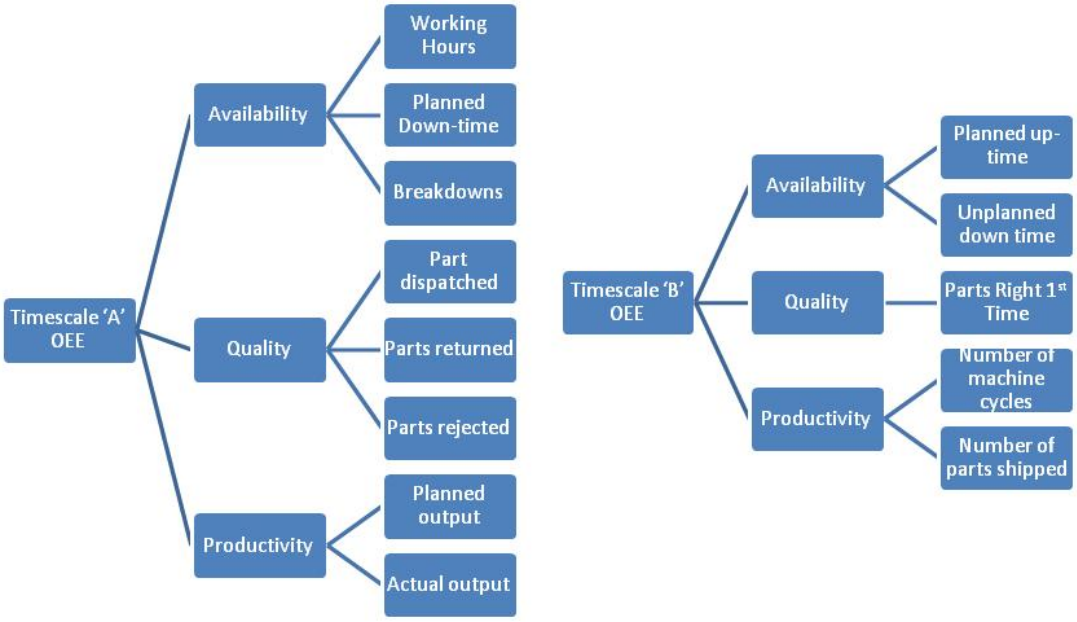


Figure 3-17 - Inconsistent MI definitions through time

Figure 3-17 shows the inconsistency between two time frames within the same operational plant. In this way the plant may report a change in OEE metric without any true change in operational effectiveness despite the use of a standard definition of the OEE metric.

The gap between the two timescales A and B i.e. the rate of change of the system, varies as per the previously described timescales and information processing rate models. The risk of MI definition inconsistency over time is proportional to the rate of system and Information structure change which in turn is related to the organizational level.

3.5.6 The complexity of systems & information structure change over time.

The previous section has described:

- The level of information flow within the MI domain (both types and complexity)
- The diverse information structures and the importance of structure compatibility and consistency.
- The rates of information processing, the levels at which it is processed.
- The potential risk and impact of inconsistent MI definitions.
- The different rates of change across an organization.

This section will describe why the heterogeneous nature of system and information structure rates of change is a significant challenge to maintaining MI system functionality and interoperability.

Figure 3-18 shows a simple set of 4 systems (systems A,B,C1 and C2) with simple information flows (Da,Db1 and Db2). This model is to transition to a future state where system B is split into two systems and systems C1 and C2 are combined into a single system. It can be seen that all of the data flows, including their content and structure, are required to change. It is usual, however, for there to be an interim state as the systems do not change at the same rate, as can be seen in the figure, and this creates a complex interim state where the outputs from Systems Bf1 and Bf2 need to provide a consistent flow of information (i.e. Db1 and Db2) to Systems C1 and C2 from the new inputs Daf1 and af2.

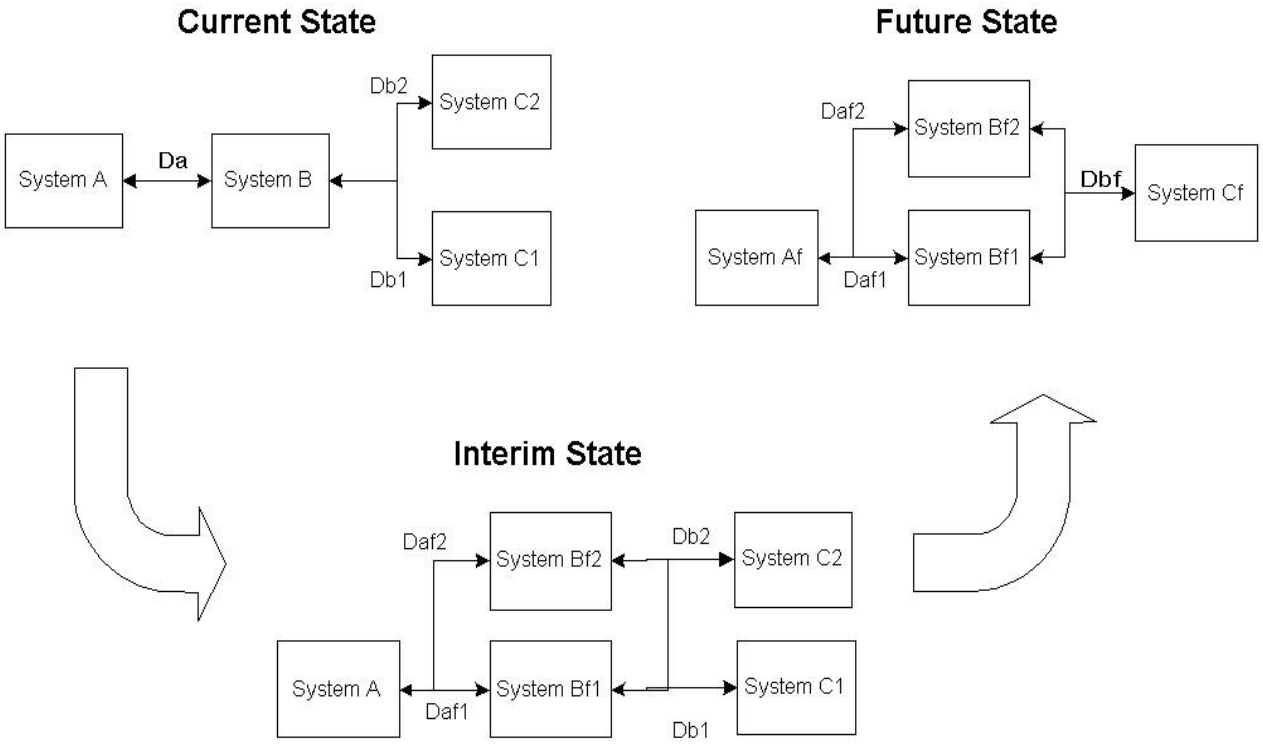


Figure 3-18 - The complexity of heterogeneous system change

When this model is considered in the context of the previous sections i.e. the levels of information sharing, systems and timescale disparity the full complexity and challenge of maintaining consistent processes, systems and information flow is apparent.

3.6 Summary

The key findings of the work to understand MI systems interoperability issues were:

- Manufacturing Intelligence Systems and information flows are complex and numerous and operate across all levels of the organisation
- These MI information flows are required to provide and support different info structures, even at a local level.
- The definitions of/ used in MI system can be inconsistent across an enterprise undermining the ability to share information
- Definition inconsistency is compounded the more the MI information is processed hence enterprise system are at the greatest risk.
- The defined MI system and information flow and the process level/ metric level definitions can be aligned to the ISA 95 system timescales model
- The same model of MI definition inconsistency used across an organisation can be used to describe inconsistency through time.
- The risk of MI system and information definition inconsistency over time is proportional to the rate of change of systems in Information structure, hence MI is a key area of hazard.
- A generic lifecycle can be used to describe systems and information structure change process. These lifecycles can be used to represent the non uniform rates of change.
- Heterogeneous system and information structure changes combined with the number of information types and systems and hence opportunities for inconsistency make MI systems information sharing over time a significant challenge.

The impact on an enterprise of these issues can be summarized as:

- An inability to create, update or maintain complex, integrated legacy systems resulting in potentially catastrophic business continuity risks. This also results in the failure to adopt systems improvements which can put the enterprise at a competitive disadvantage.
- A motivation to avoid systems interoperation or integration leading to ‘islands of information’ and making manufacturing decisions on incomplete information.
- Significant effort required on manual data collection and collation which due to its manual nature cannot meet the speed, accuracy or reliability requirements for MI decision making.
- An inability to consistently specify new MI systems, leading to system incompatibility or inconstancy, which in the worst case may be undiscovered i.e. the systems would

be misinterpreting each other and not functioning as intended without the users being aware.

- The reliance on standards to manage these issues means the systems can only develop as fast as the standards, and the wider the standards are used the more difficult consensus on updates is to achieve and so change becomes slower, as such a widely deployed standard would be unable to support the rate of change at the manufacturing operational level.
- Limited understanding or ability to express the requirements of systems as individuals and as a network leading to Enterprise system requirements over-riding manufacturing or operational level requirements due to them being generally recognized as more business critical without understanding either the impact of the alternative options.
- Lack of robust methods of ensuring system consistency results in the requirement for massive regression testing to mitigate the business continuity risk. This level of testing constrains ability to adopt change, as well as having a significant cost and resource impact i.e. if this testing is not practicable, the change cannot be addopted.

The impact summarised above results in an inability to create or maintain automated MI systems across a large organization. These systems are, and will be increasingly, required to optimize manufacturing enterprises and make good manufacturing decisions in a timely manner; this inability will therefore limit the effectiveness and competitiveness of these organizations.

4 A novel approach to manufacturing systems interoperability in dynamic change environments

4.1 Introduction

In this chapter the requirements for the solution concept are explored and the solution concept described. Based on this novel core concept ontology based solution, a set of research questions are defined. These research questions provide the framework for the research and testing activities.

4.2 System interoperability core concept heavyweight ontology

4.2.1 Ontology solution requirements

Based on the understanding generated in Chapter 3 the solution requirement can be defined as:

- A method to describe and specify future MI systems which ensures consistency and continuity of system function.
- This method must be able to consistently describe and specify any element of a current or future system or the systems in its entirety including its concepts, relationships, processes, interactions and interoperation and must work across the different organisation levels. This must include new concepts or types of relationship.
- The descriptions must be unambiguous and robust despite the nature of the future systems being unknown at the point the method is defined.
- Concepts must be machine interpretable.
- The solution must support the rapid and disparate nature of MI systems rates of change.

The requirement of the solution to deal with dynamic and developing systems interoperability implies the requirement for a unified approach (see section 2.6.4.1) where interoperability is provided through a common meta-structure.

Figure 4-1 represents the requirement within a single timeframe: each individual system must be understood as well as their configuration and interactions. This is represented by the analogy of explicitly defined geometrical elements which are assembled into a specific form with the cross systems interactions shown. The required solution must be able to provide this basic requirement within the MI domain, but it must also enable the overall form and the systems and their interactions to develop over time.

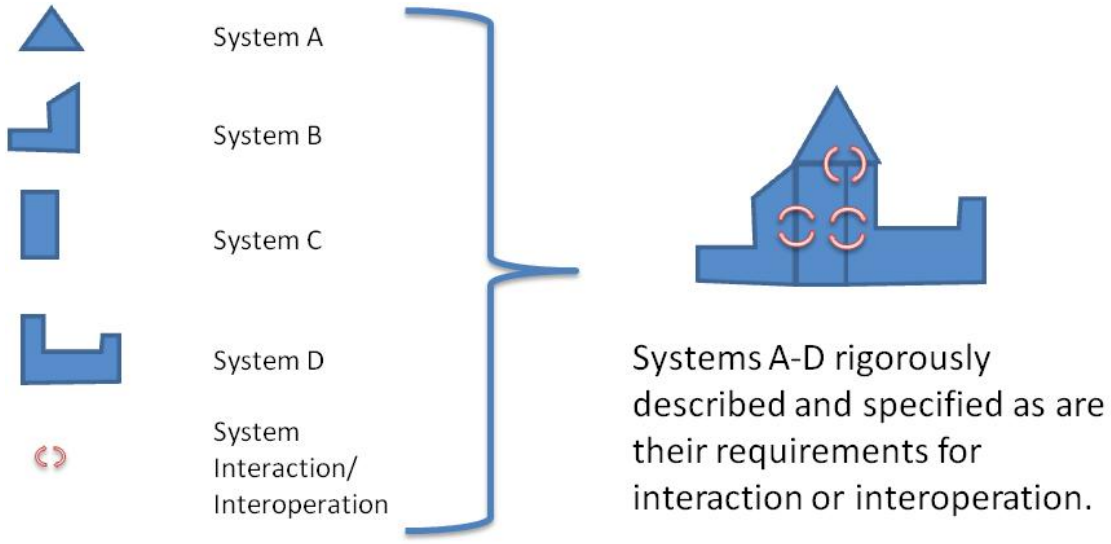


Figure 4-1 - A methodology of describing individual systems and their interactions and interoperations is required

In line with the selected, ontological approach, the solution must answer the ontology competency questions which in this case were defined to identify and ensure interoperability between systems. These competency questions shown in the Figure 4-2 were defined through the analysis of the literature review and industrial investigation (see Chapters 2 and 3).

Figure 4-2 - Ontology competency questions and supporting questions

4.2.2 Solution concept overview

Developing the identified gaps in the current knowledge, this research proposes the development of a mechanism for enabling and assessing system interoperability through time. The MI domain was chosen as a suitable domain to develop the solution, which also required work to define the domain to the required level of rigor.

Figure 4-3 describes how this approach meets the requirements detailed in the previous section by continuing the geometrical analogy: A MI domain Foundational Ontology or Core Concept Ontology (see section 2.6.6.8) is built upon a clear scope definition and used to define individual the systems, which in this case are represented by geometrical shapes. The foundational ontology, which may be supported by more specialised domain ontologies, in this example would be geometrical form descriptors. This ontology can be used to describe the individual systems and over all form as they change over time because they still conform to the foundational geometrical descriptors The individual systems can be designed to 'fit' together to form the overall system consistently due to this common foundation, and while the individual systems and processes may change over time, represented in this case

by the different scale of shape of the geometries, the functionality is retained and further developed.

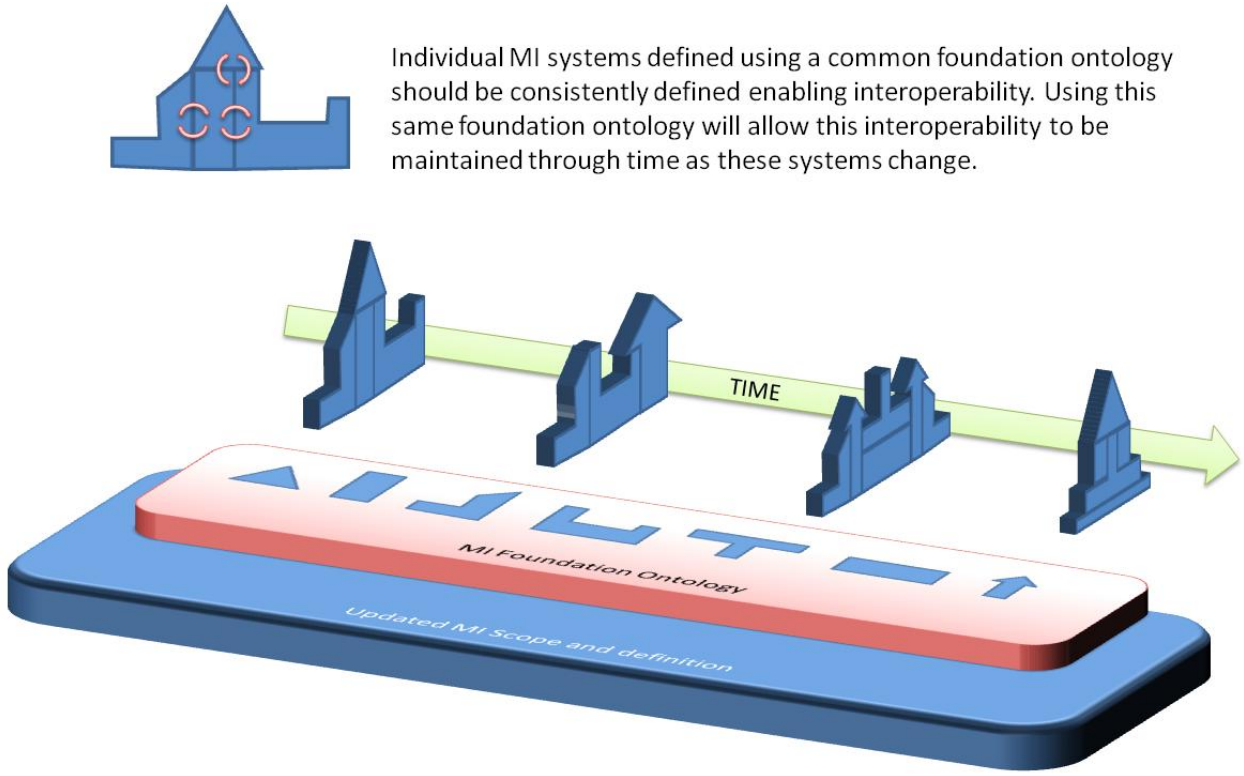


Figure 4-3 - Ontology based solution

The solution concept is able to identify where system interoperability is occurring or required and ensures interoperability by answering the competency questions defined in section 4.3.1. and constraining the system definition in line with explicit logic that defines the requirements for interoperability. The proposed solution answers these questions through a combination of core logic as new systems are declared to it, and user interaction or queries. As new systems are declared to the solution, it develops and expands both through explicit declarations from the user and inferred knowledge using the embedded system rules.

The proposed solution uses a common MI foundation ontology (which is a Core Concept Ontology) which is specialized to form various domain ontologys for the different areas across which the MI information flows for the purpose of specifying or describing the relevant systems and processes. These domain ontologies will be consistent because they use the same foundation, however, they will require the rigour and machine interpretability of a heavyweight ontology.

The proposition is: because the foundation ontology embodies the core foundational requirements of MI, even emerging MI concepts and relationships can be defined from this foundation in a manner consistent with the existing ontologies.

Figure 4-4 shows the capability of the proposed solution and how it extends the current state of the art.

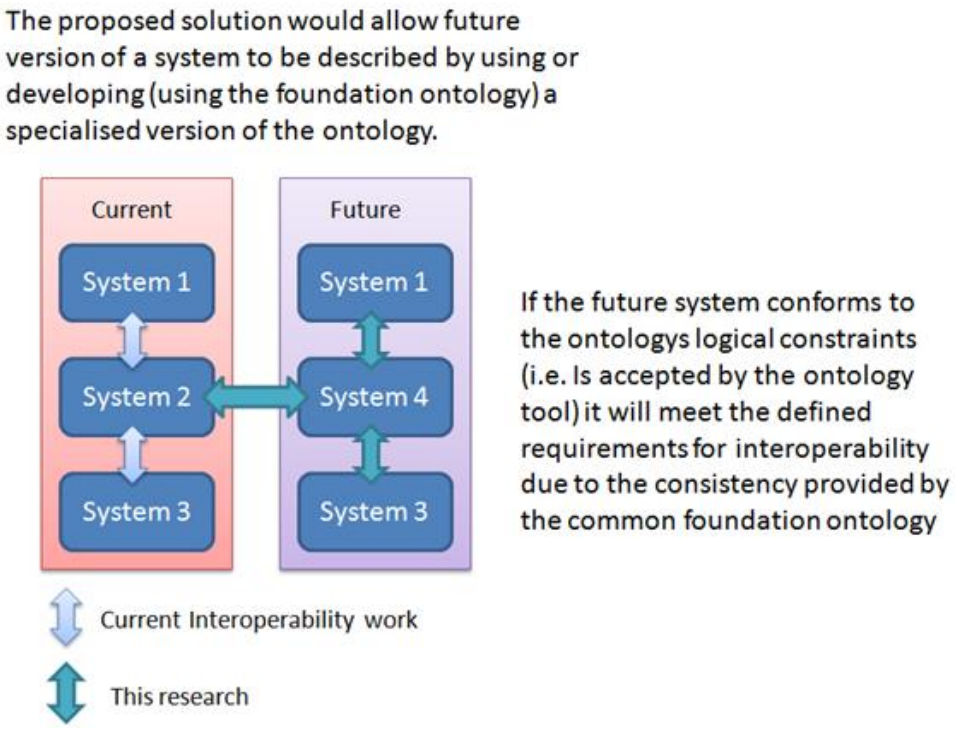


Figure 4-4 - The solution proposal capability

Figure 4-5 shows the proposed multi level ontology approach as proposed for MI. The foundation ontology is applicable on a much wider scope than Manufacturing intelligence systems due to its level of abstraction, This foundation ontology's scope can be described as 'Systems'. The more specialised core concept ontology developed from the foundation ontology narrows in scope, to focus on Manufacturing Systems. It is at the domain ontology level that the scope is specialised to Manufacturing Intelligence systems, with the final level of specialisation being the specific instances that populate the knowledge base. It is important to note that the instances may not conventionally be considered part of the ontology, but the proposed solution will rely on the populated instances to function therefore in this case they are an integral part of the ontology solution.

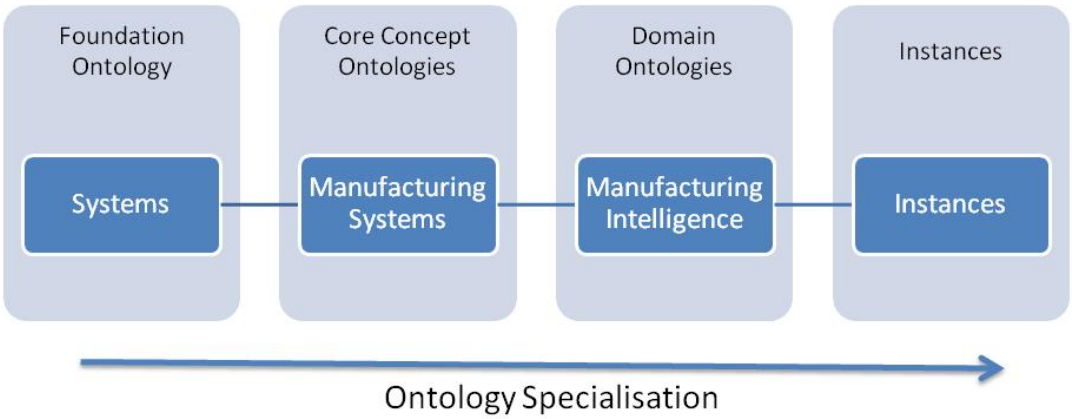


Figure 4-5 - The specialised ontology levels scope

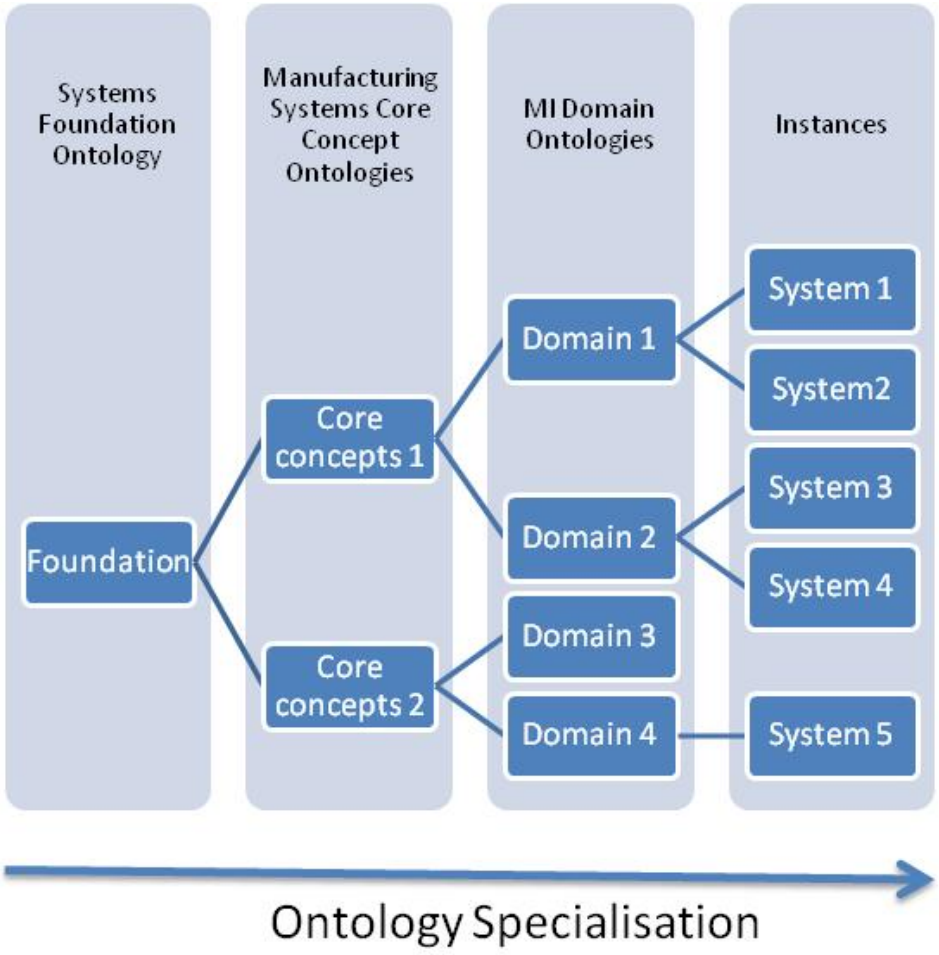


Figure 4-6 - The specialisation of ontologies from Foundation to Domain instances

Figure 4-6 shows the foundation ontology being used to defined two core concept ontologies, which are then used to define a number of domain ontologies. A potential example would be a Manufacturing System and Communication System core concept

ontology, which could both be defined from the same Systems foundation ontology. These two ontologies would be consistent due to the use of a common foundation.

This approach builds on research in the field of ontology based interoperation (see Chapter 2). The further development that will be required to meet the requirements of MI, will be the development of this method to address change over time i.e. applying an additional time axis to Figure 4-6.

Figure 4-7 shows the Foundation ontology being specialized, but also shows that the foundation ontology provides consistency and enables interoperation within a single time snapshot, and also that the persistence of this foundation ontology will provide this capability through time. The rate of change within the individual domains i.e. the rate of progress through versions does not need to be uniform or constant across the various domains.

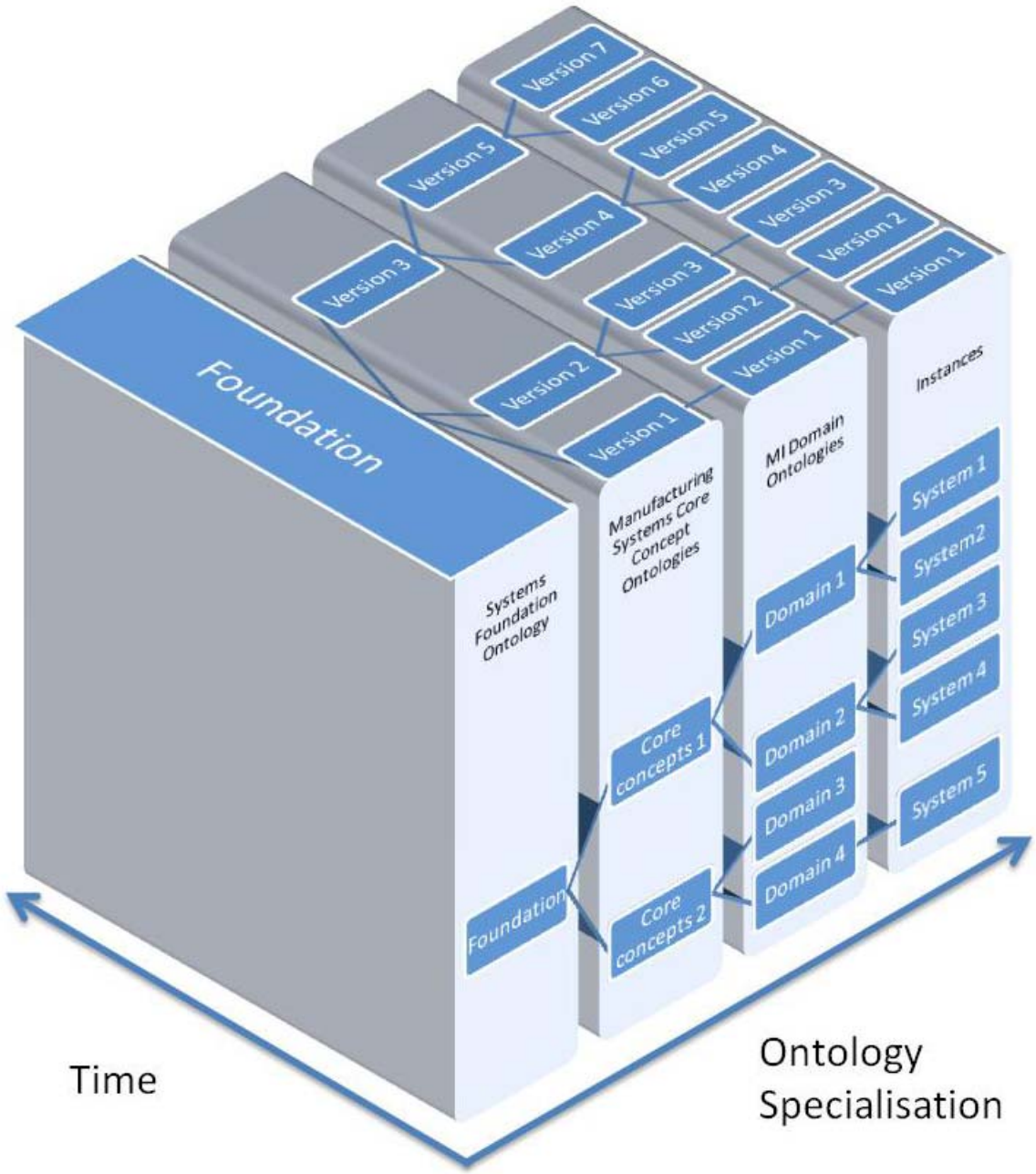


Figure 4-7 - The heavyweight Foundation ontology providing a consistent basis through time

Figure 4-7 shows the domain consistency being maintained through time. Figure 4-8 shows that either new systems can be added to the domain or new versions of any system can be added, with the heavyweight ontological rigor ensuring that the interactions with existing systems are understood and that system or version has been designed in line with the logical constraints for interoperability. If it is not, the solution will prompt the user for more information or flag the error.

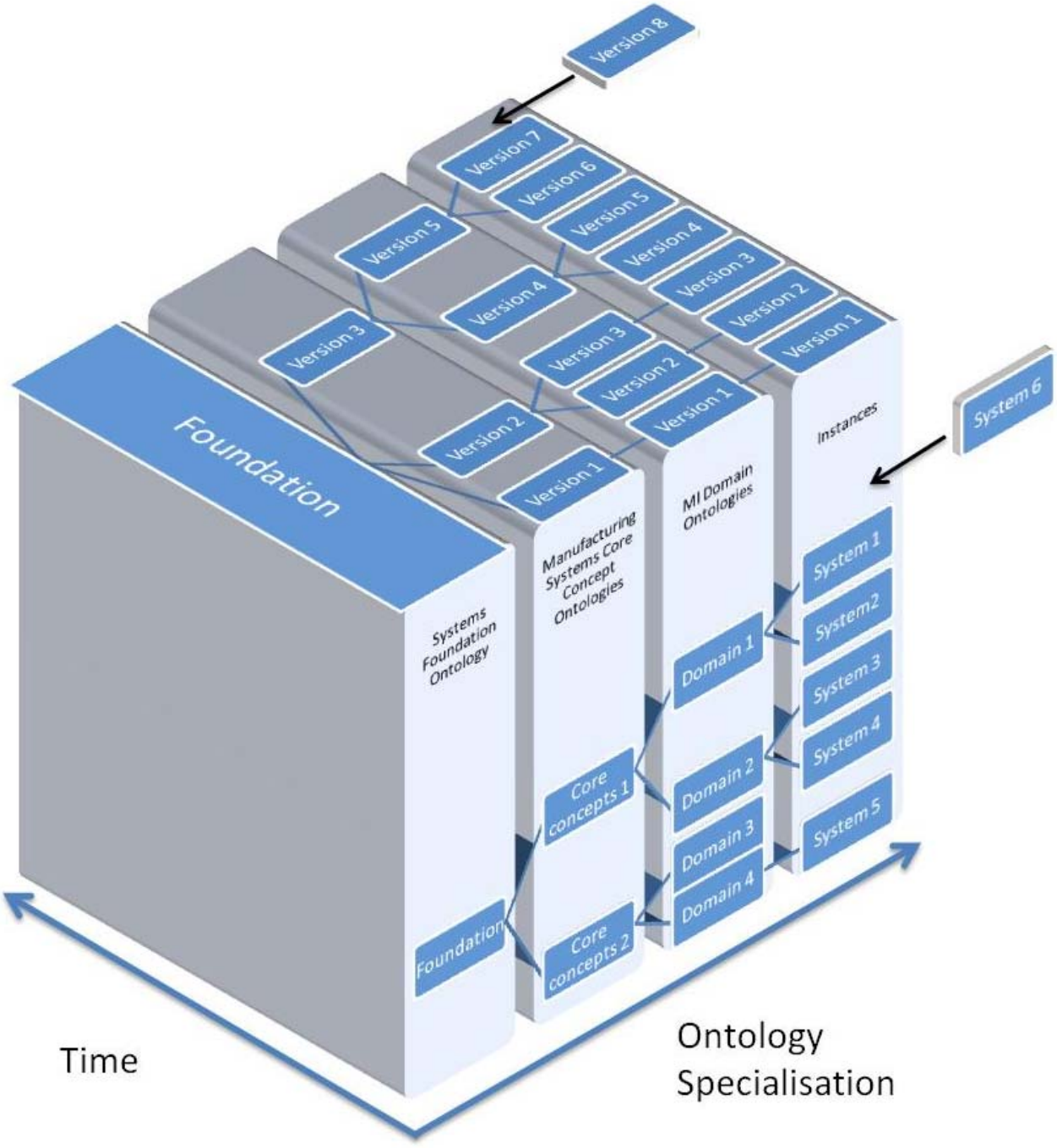


Figure 4-8 - The solution enables the addition of new systems or versions ontology or knowledge base.

4.3 Research questions

To confirm the validity of the proposed approach 4 research questions were defined:

5. What is Manufacturing Intelligence?
6. What are the concepts in the MI Systems domain?
7. Can a domain foundational or core concept ontology be defined and formalised?
8. Can the proposed concept be proven through the formalisation of the ontology.

The research work carried out to answer these questions is the subject of Chapters 5, 6, 7 and 8 respectively.

5 Understanding the scope and concepts in manufacturing intelligence

5.1 Introduction

This chapter details the research carried out to create a definition for ‘Manufacturing Intelligence’ in response to research question 1 (Section 4.3). The resulting definition resolves the inconsistencies and lack of clarity idenified in the literature review (Section 2.3).

This work used a combination of literature research and industry based research to create a clear understanding and definition of Manufacturing Intelligence. This work resolves potential contradictions in existing understanding resulting in a lightweight ontological model that allows a definition that is relevant for many perspectives and also identifies relevant terms and relationships within the domain.

‘Manufacturing Intelligence (MI)’ is a term which is increasingly common in industry and which has been referenced in standards for a number of years (ISA 2000, ISA 2005). Despite the prevalence of the term, there is a lack of clarity regarding the definition of MI with many aspects of the existing definitions conflicting with each other.

MI was chosen as a target domain as it is widely recognised as a key area for investment and rapid development through the literature review and industrial investigation in Chapter 2 and 3.

5.2 Method

Following a review of existing literature, open questions were defined that could be used to prompt answers from relevant industrial experts regarding their understanding of the term ‘Manufacturing Intelligence’ without leading or biasing their input. The respondents were chosen so as to provide a cross section of organisational levels and functional roles across a manufacturing landscape. The responses were then transcribed and key text terms gathered from them. These key text terms were grouped by synonyms and a tally created against each question and response to indicate key concepts of interest. These terms were also mapped using the UML class diagram convention to understand the relationship between them. The synonyms were then further interpreted with industry experts and grouped by meaning or function to provide a stronger indication of the expert’s views of the key functions of Manufacturing Intelligence. These results were then interpreted in the context of the existing literature based definitions and an updated definition of MI created. This definition was then assessed against the existing literature and individual inputs from the experts.

5.3 Literature summary

The key points of the reviewed literature are:

- MI supports manufacturing decision making and reactions
- While MI is defined in standards as a sub function of manufacturing execution systems, there is no attempt to resolve the issue that MES systems reside at Level 3 of the organisational model whereas MI information could come from other levels as implied in the emerging term ‘Enterprise Manufacturing Intelligence’.
- MI is loosely defined.
- MI has been described as the combination of Business Intelligence, manufacturing performance and real-time information.
- The links between Performance Indicators/Key Performance Indicators and MI is implied but not explicitly stated.
- MI systems do not yet seem to be given the same recognition as ERP, MES and CADCAM systems and there are few detailed MI systems requirements available.
- While MI is generally considered to involve distributed systems, connection (via SCADA systems) of ‘intelligent machines’ does not constitute MI due to the lack of overall coordination and orchestration.
- Key functions of MI are aggregations, contextualization, analysis, visualisation and propagation of manufacturing information.

While most of the material is compatible, the emergence of the term ‘Enterprise Manufacturing Intelligence’ (Siemens AG 2011, AMR Research 2011) seems to represent an aspiration to make MI more informed and powerful by pulling in information from many other areas of an enterprise, which in turn implies MI is outgrowing the ISA 95 definition of MI as a sub function of MES. This makes the definition of MI or its emerging form of EMI even less clearly defined.

5.4 Industrial survey & research

It has been recognised that MI is loosely defined as described in the literature review in Chapter 2. This has led to many different interpretations of the meaning and scope of MI across the manufacturing domain. The survey respondents were chosen to reflect the breadth of individuals with a view on MI. 30% of the respondents were aware of the term MI, but were unable to provide answers to the questions.

On this basis the individuals that were able to provide strong answers were considered subject matter experts. This was not solely on the basis of their knowledge of MI but also taking into account their knowledge of the manufacturing domain.

5.4.1 MI questions

The three open questions used to gather input were:

- In your view, what is 'Manufacturing Intelligence'/ what does it mean?
- What is it for/ what is its purpose?
- How will we know when we have it (what does success look like)?

The MI questions were sent to 40 individuals in a wide range relevant roles. Their responses are tabulated below against their role title.

The format of the responses was left open to avoid constraining the thought processes of the respondents.

While the responses display some key themes, no clear consistent answer could be inferred directly. It was therefore necessary to carry out a level of structured analysis of the response content to draw meaningful conclusions.

5.4.2 Feedback

Figure 5-1 to Figure 5-4 summarise the responses to the MI definition questions:

	In your view, what is 'Manufacturing Intelligence'/ what does it mean?
Senior Vice President	Data on our processes, inputs and outputs
Process Excellence Manager	To me Manufacturing Intelligence would provide a complete picture of the actual and potential performance of a manufacturing facility
Capability Acquisition Engineer	The capture and use of Manufacturing Data from actual manufacturing/inspection processes. E.g. Dimensional Data, Process KPV's etc
Process Team Leader	Manufacturing Intelligence is about getting the information to the right people in the right place at the right time. It is the "intelligence" of our manufacturing processes and their performance
Manufacturing Systems Architect	Business Intelligence applied to a manufacturing production context..it is therefore composed of two high-level elements: The interface with control systems and IT systems in order to collect/extract production data (actually a pre-requisite of MI, not strictly a component) The implementation of a "data warehouse" and reporting solution (including the development of manufacturing-specific reports) to enable analysis of the data Manufacturing Intelligence is NOT what Org X are doing - they are "enriching" the process definition in the router and developing the control systems to constrain the process to its authored definition - this is process control, not manufacturing intelligence..although some MI is generated as a by-product
Shop Floor Systems Super User	Using data to make informed decisions that reduce waste in Manufacturing processes
Head of Manufacturing Engineering	All data collected as input (e.g. KIPVs) as well as output during the course of a manufacturing process
Manufacturing Product Introduction & Technical Governance Executive	MI is data about a manufacturing process, including inputs in-process variables, controls, and outputs. The term includes the tools/systems for capturing such data and for processing it into useful information to control and improve current processes and design new processes which are capable and robust from day 1
Chief of Man Sys	Being able to collect data from manufacturing processes in a system independent of being created manually by an operator or automatically from the machine controller or any device connected to the machine tool, furnace, etc. Ability to analyse, report and store these data
Manufacturing Systems leader	MI should be an extended form of BI which is widely used to aid the allocation of scares resources in business. MI is a comprehensive set of tools any of which are used to answer the question where do I send my scare resources in order to maximise my profit/benefit
Manufacturing Systems Executive	MI is two things to me. First it is the collection of Key Process Variable data (i.e. feeds, speeds, pressure, temperatures), Product Attribute Data (i.e. actual geometry, surface condition) and Operational Data (Operator ID, start and stop times, OEE, downtime reasons). Second and more importantly it is the reporting and analytics of this information in such a way as to generate meaningful information (intelligence) that allows us to bake in process improvements and control the process to produce more consistent conforming product.
Data Driven Business Programme Manager	The combination of data and knowledge built into an entity ie a standard feature

Figure 5-1 MI questionnaire feedback

	What is it for/ what is its purpose?
Senior Vice President	improving business performance, both short term immediate tactical, and long term strategic
Process Excellence Manager	Providing: Process capability - actual measured at process and part level Quality performance Delivery performance Op by op real time processing time and quality (where are the parts and are they proceeding on time and to quality) Processing cost by part and Op System utilisation and capacity Resource utilisation and capacity Manufacturing risks and preventions/mitigations are maintained in near real time Energy and other overhead cost monitored tracked
Capability Acquisition Engineer	Used for monitoring of processes To look for deviations from expected behaviour in order to avoid production of non-conforming components To monitor the health of equipment and plan maintenance etc Used to build a knowledge of the process to enable identification of improvements and optimisations Used to drive design decisions for future component designs to maximise compliance with actual manufacturing capability
Process Team Leader	Answering qns such as: Do we have a dimensional quality problem? Do we have an emerging bottleneck? Do we have a spate of issues on one particular feature of our parts? Do we have a problem with the efficiency of one particular piece of kit? Are our furnaces running over temperature? Etc etc etc etc
Manufacturing Systems Architect	The purpose is to provide the capability to process the derived data in order to determine/prioritise/monitor process improvement initiatives
Shop Floor Systems Super User	To provide users (ME, Ops, Logistics etc) with knowledge to improve process capability and reduce cost
Head of Manufacturing Engineering	Req'd to understand inputs/outputs to maintain and further improve capable manufacturing process
Manufacturing Product Introduction & Technical Governance Executive	MI should help us to reach a point where manufacturing processes are self-controlling with minimum manual intervention, and where that manual intervention is informed by data rather than skill or opinion. It should provide us with the information to prioritise opportunities for improvement and to ensure such improvement actually delivers results. It should help us to design robust and capable processes for new parts and products.
Chief of Man Sys	MI is used as data base for improvement activities following the DMAIC process and ensuring that data are captured and stored where required for verification of compliance of an approved make process of a product, e.g. coolant flow, temperature limits, pressure, etc.
Manufacturing Systems leader	MI is a comprehensive set of tools any of which are used to answer the question where do I send my scare resources in order to maximise my profit/benefit
Manufacturing Systems Executive	MI is two things to me. First it is the collection of Key Process Variable data (i.e. feeds, speeds, pressure, temperatures), Product Attribute Data (i.e. actual geometry, surface condition) and Operational Data (Operator ID, start and stop times, OEE, downtime reasons). Second and more importantly it is the reporting and analytics of this information in such a way as to generate meaningful information (intelligence) that allows us to bake in process improvements and control the process to produce more consistent conforming product.
Data Driven Business Programme Manager	Definition of a capable and robust product design or process based on Manufacturing Intelligence in a "real-time environment"

Figure 5-2 MI questionnaire feedback (continued)

	How will we know when we have it (what does success look like)?
Senior Vice President	Right first time processes, that are stable and capable
Process Excellence Manager	We can accurately predict production times (launch to complete) We can track actual against predicted production times We can accurately predict factory capacity We can identify quality and delivery issues before they impact the customer Component cost build up is visible Problems can be identified and isolated within 60 mins. People at each level have easy visibility of information they need to understand the progress and issues with their task Everyone is utilising the same consistent base data The Manufacturing Information System data is a key driver for all management decisions
Capability Acquisition Engineer	Data is being used Both at the monitored process "duckboard" and at cell leadership level to monitor, report and correct deviations from expected behaviour By MTM to monitor health of machine tools to better plan maintenance and prevent failures and downtime By Manufacturing Engineers to drive, investigate and deliver process improvement activities. By Design Engineers to influence design decisions towards our actual manufacturing capability The key word here is "used", merely having the capability installed is not good enough.
Process Team Leader	Access to this data at headline and drill down levels to allow management, engineering, logistics, shopfloor to understand where our problems are as they happen. In systems that can be compared directly (not via excel) so you can see if machine KPV issues relate to part quality issues (for example).
Manufacturing Systems Architect	A full definition of the data to be collected from each process under change management (otherwise how do we know when we're "done"?) - A full definition of the reporting solution used to interpret the data (and hence the required data model) - (otherwise how do we know when the solution is complete?) - The active storage of such data in the "data warehouse" (got to be implemented/working right?) - The deployment of the reporting solution in accordance with the requirements derived from bullet #2 (as previous) - The active management of the processes using the MI toolset (Business Metrics - CPK etc.?) - (most important - how do we know we're using it to derive benefit?)
Shop Floor Systems Super User	When all users have easy access to knowledge that can be used to identify the route causes of waste
Head of Manufacturing Engineering	Overall measure must be feature-based Cpk values 1.33. Sub-milestones could include assessments against KPIVs and process outputs.
Manufacturing Product Introduction & Technical Governance Executive	When all our most critical processes have a level of control and knowledge sufficient to make them robust and capable.
Chief of Man Sys	My vision would be that the above mentioned is available without having still the need for paperwork. To enter data into a system for further analysis or reporting is no longer required since data once recorded can be transferred and used wherever needed.
Manufacturing Systems leader	Our scarce resource is often time and people so when you look at MI as a subset of BI we would look at understand where your pressure points are (often Non-conformance or delivery pressure) and use the intelligence derived for MI to better allocate our scare resources (people and time).
Manufacturing Systems Executive	success looks like a number of continuous improvement engineers per plant using the MI data in DMAIC projects to drive up quality, and that we see the plant KPI's improving in line with this.
Data Driven Business Programme Manager	When we have a closed loop system whereby products are designed on MI and the process feedback is maintained and updated the ongoing design process

Figure 5-3 MI questionnaire feedback (continued)

Shop Floor Systems
Technichal Manager

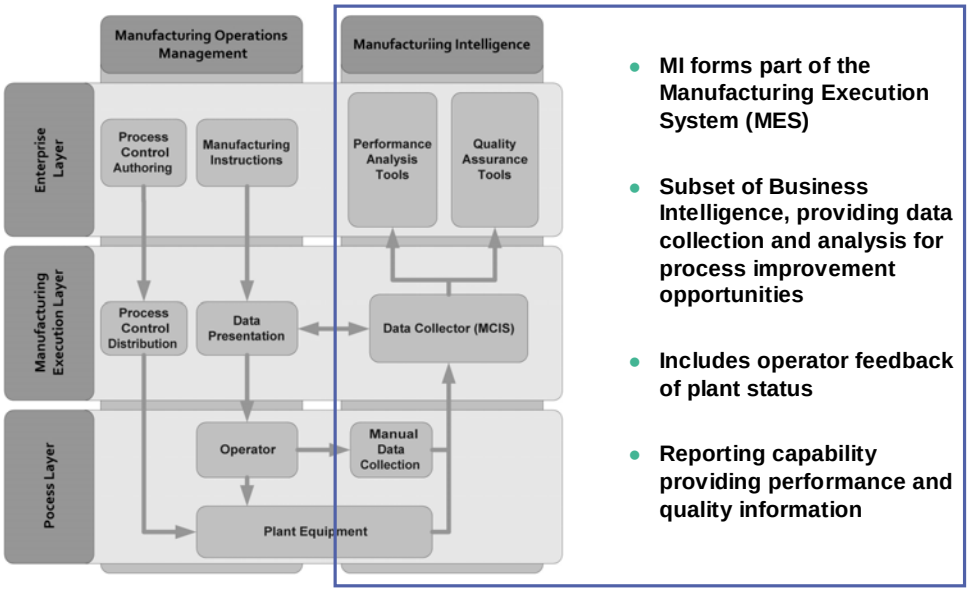


Figure 5-4 - MI questionnaire feedback (continued)

The feedback was analysed and key words within sentences extracted and tallied against each of the questions. This simply shows the number of times certain words were used and allows the inference of some key themes.

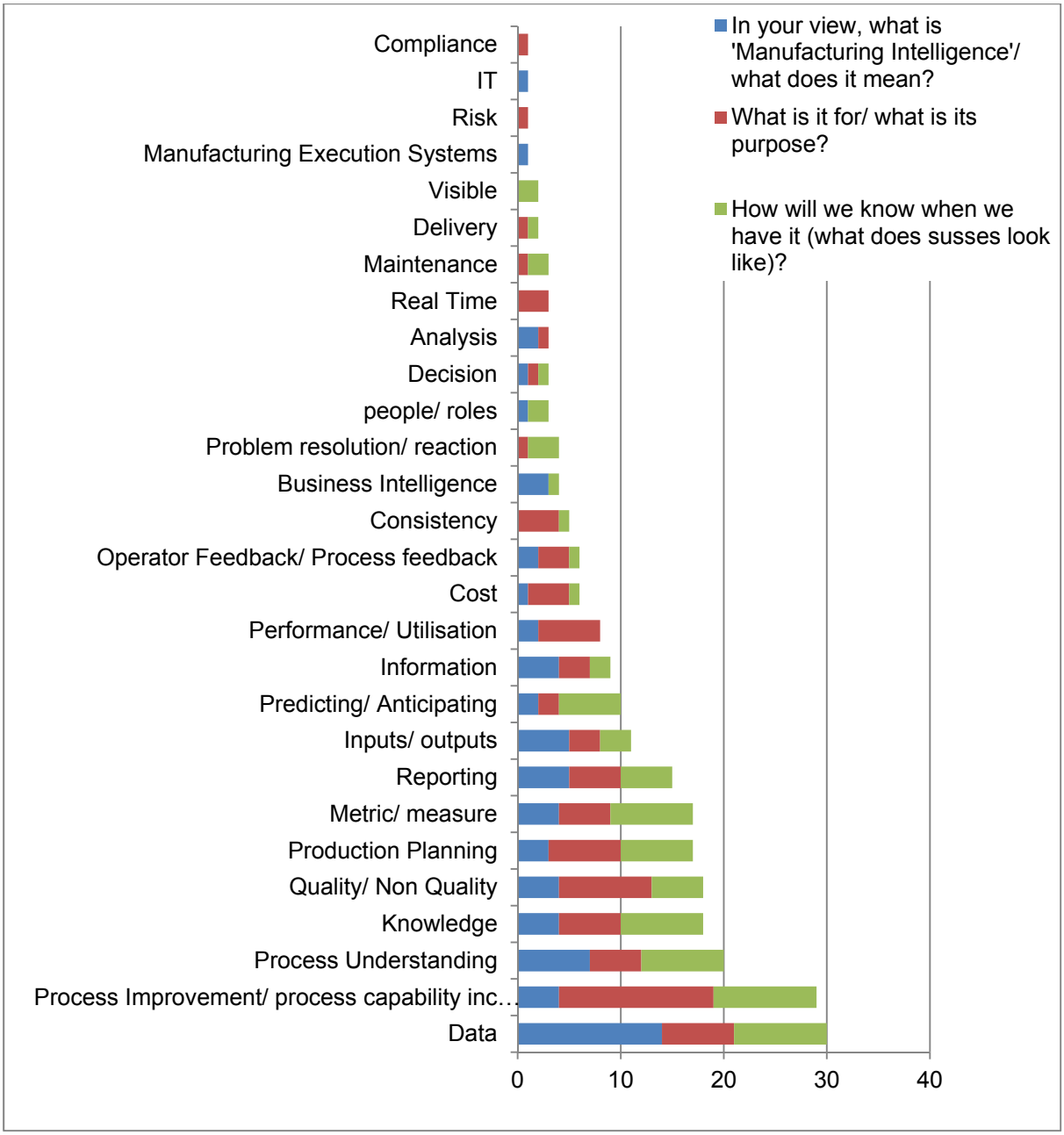


Figure 5-5 - Ordered tally results of feedback text mining

These words were then treated like concepts and using the input from the respondents and subject matter experts, relationships between them were defined within the MI context. This provides some level of structure to the gathered text based concepts and allows a further level of interpretation. Figure 5-6 shows a UML representation of these concepts and relationships. At this point it was unclear as to which level of the ontology concept shown in Figure 4-8 these terms would reside, however, the overall diagram allows the MI domain ontology scope to be defined.

Terms that were gathered, but that were unanimously agreed by subject matter experts on review not to be significant within the MI context were excluded from this diagram.

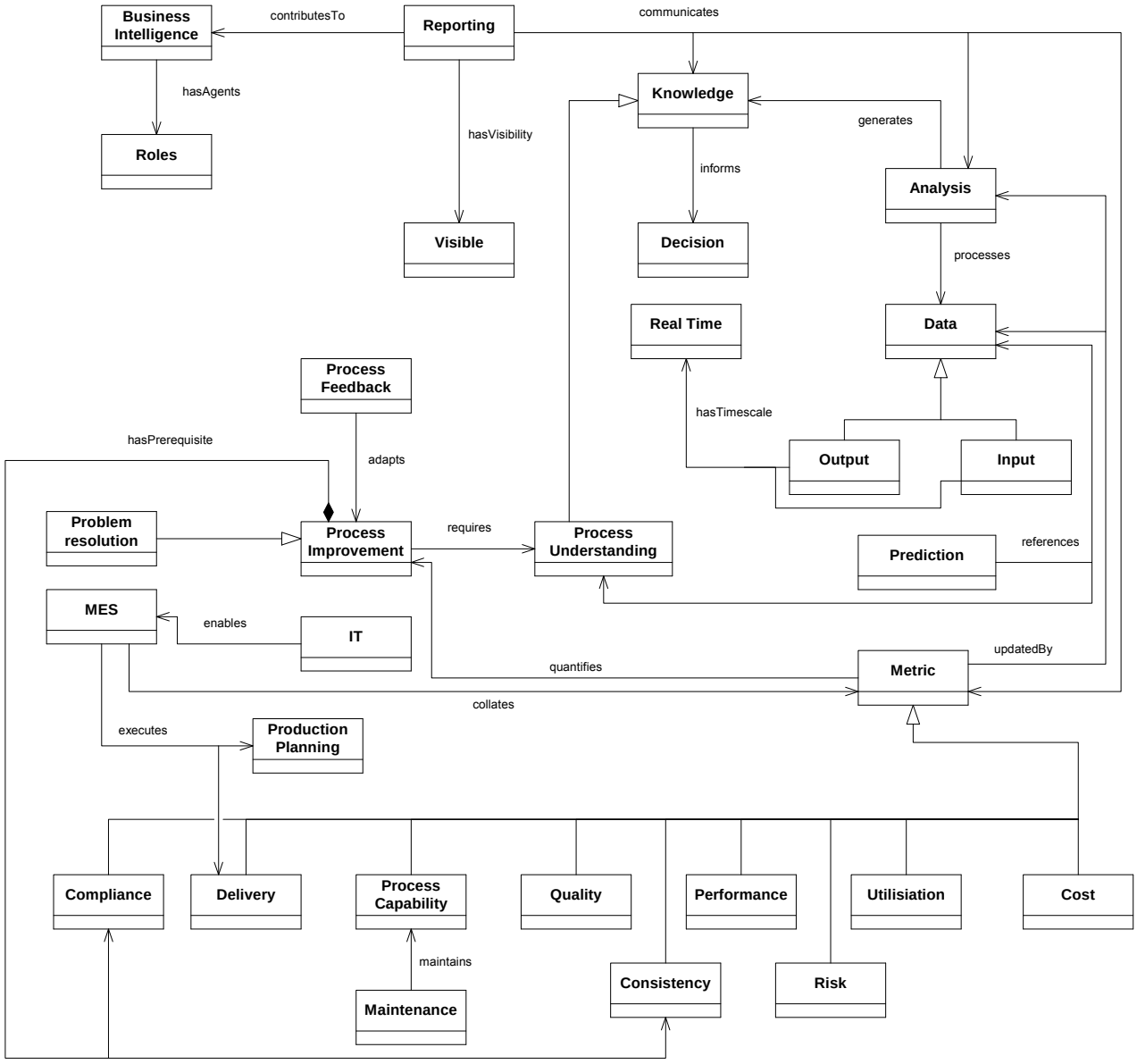


Figure 5-6 - Text analysis class diagram

Using this increased level of understanding and structure it was possible to combine some of the terms or concepts by intent or by shared meaning or purpose. This provided a stronger indication of the concepts that were more significant to the meaning of MI.

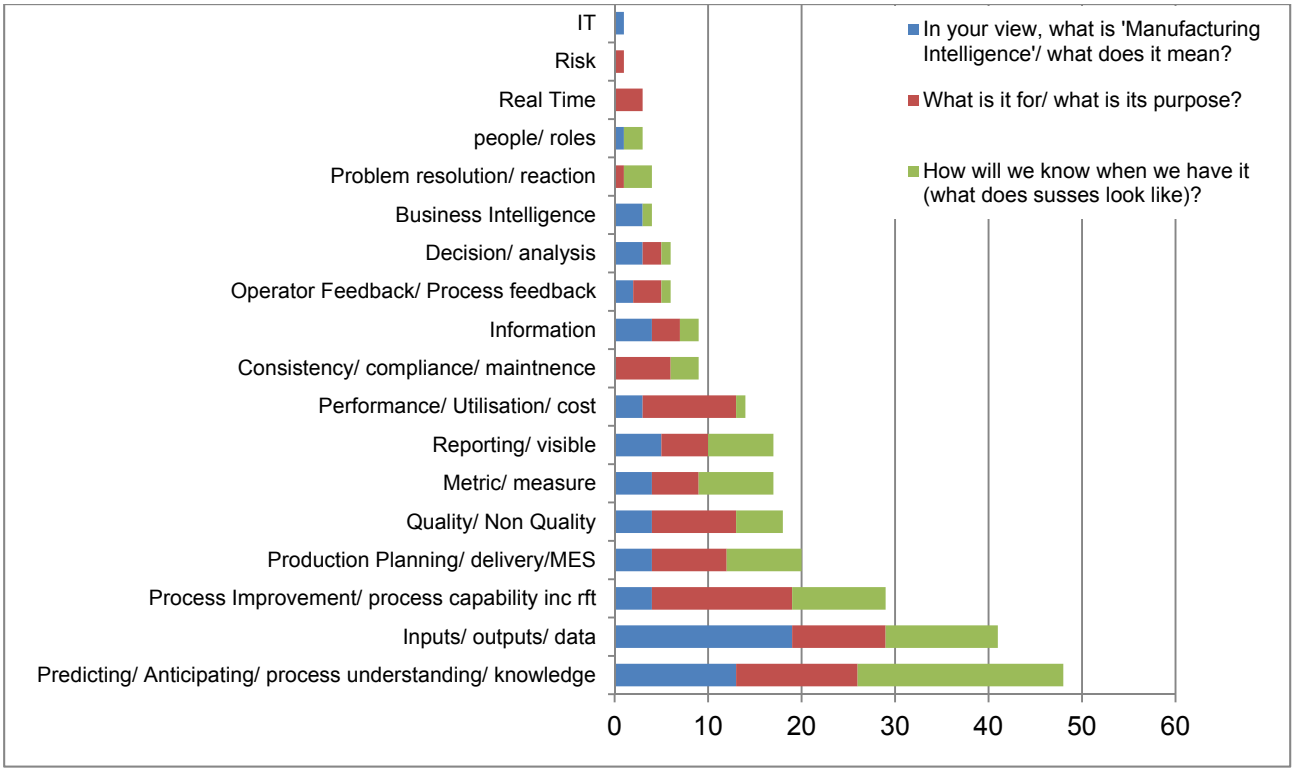


Figure 5-7 - Results grouped by 'intent'

5.5 Results

The class diagram constructed from the survey results was reviewed against the 'intent' concept tally to identify the key concepts according to the survey results. These were highlighted on the class diagram, and represent the key concerns of the respondents.

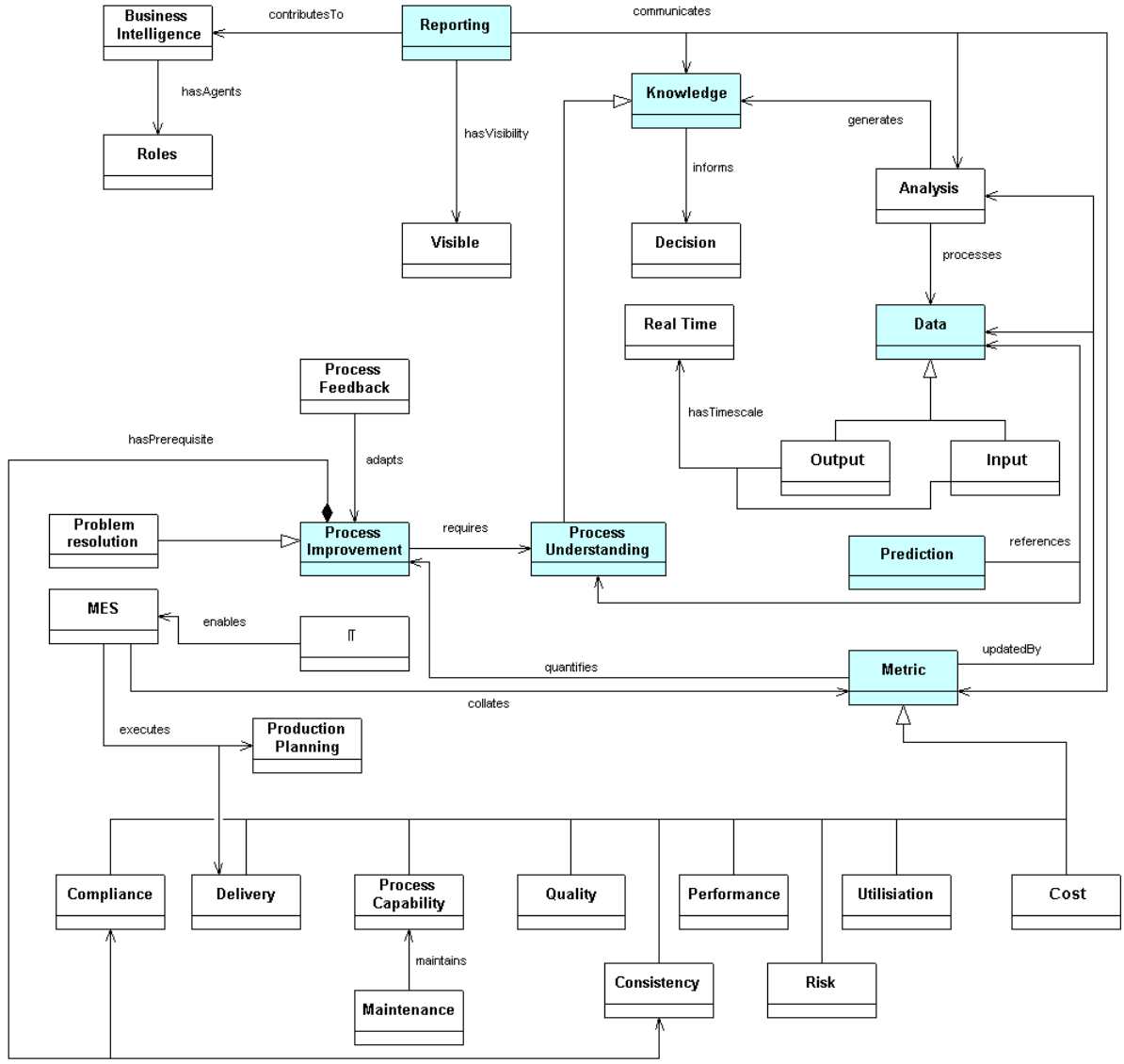


Figure 5-8 - Key concepts based on the 'intent' review

The class diagram was then reviewed against the literature review summary to identify those concepts which are also highlighted within the literature and these were highlighted on a separate version of the diagram.

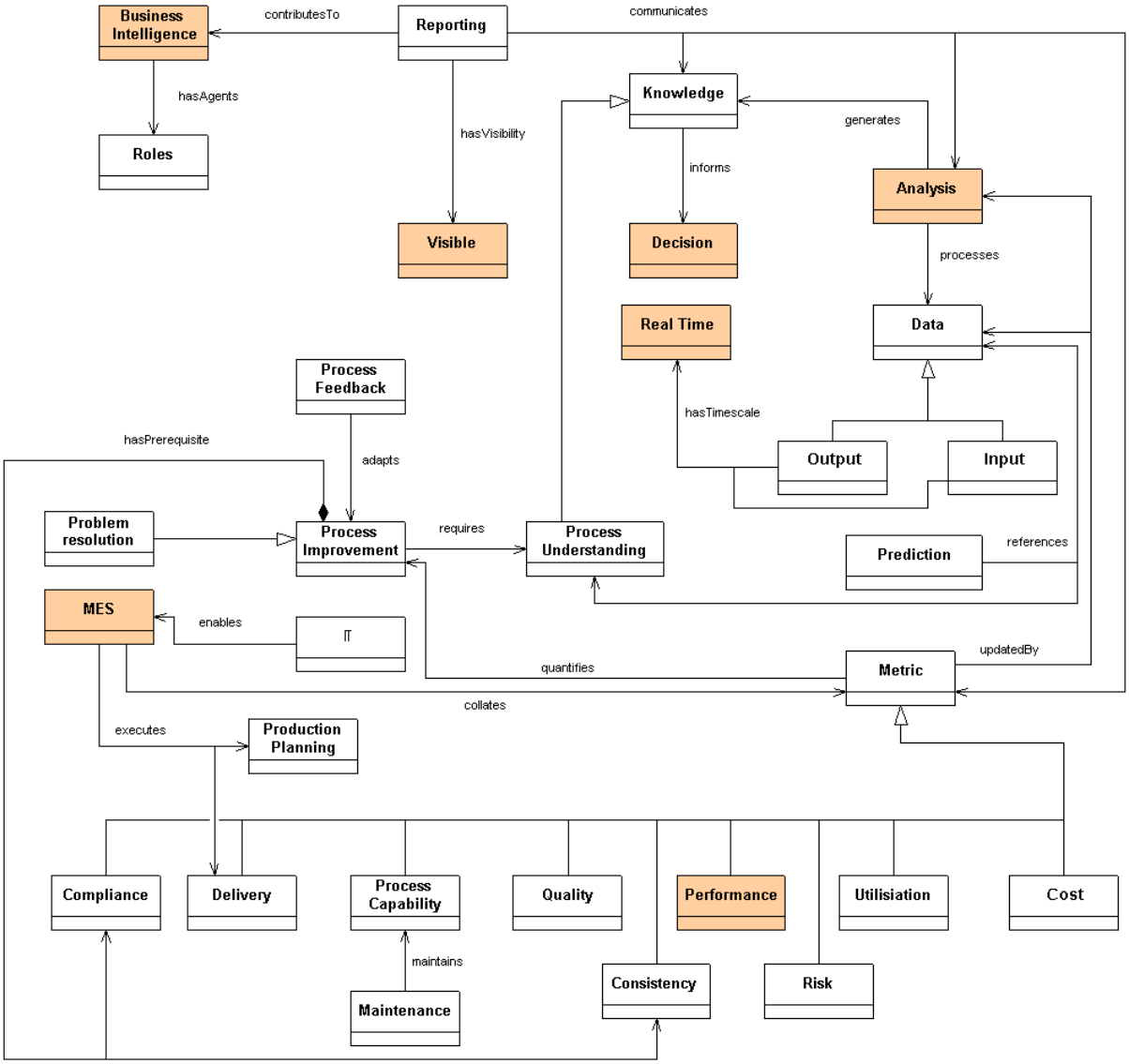


Figure 5-9 - Key concepts based on the literature review summary

It was anticipated that there would not be a significant overlap in these key concept sets, due to the stated need for this research. To ensure a meaningful outcome it was necessary to understand the links between the two apparently distinct key concept sets identified by the literature and survey results.

The key concepts from both the survey results and the literature review were tabulated and interpreted to find if they were:

- An enabler for another concept
- A descriptor of another concept
- A generalisation of another concept

It was clear that the two concept sets are closely linked and the apparent differences in specific terms could be due to slightly different but consistent perspectives arising from developments in the field of MI and knowledge management e.g. the literature focuses on the ability to make decisions, whilst the survey recognises the need for data, knowledge and understanding to make those decisions

This interpretation, along with the relationships allowed summary conclusion sentences to be constructed that use these concepts and example would be: "*MI involves the visual reporting of real time data*". It was also possible to infer some statements due to the concepts that appear to have weaker relationships or have not been referred to directly or indirectly by the key concepts.

Table 1 - Survey Key Concepts vs. Literature Key Concepts

Survey Key Concepts	**Literature Key Concepts**
Process Understanding	Business Intelligence
Process Improvement	Visible
Knowledge	Decision
Reporting	Real Time
Data	Analysis
Prediction	MES
Metric	Performance

The summary statements inferred from the key concepts and their relationships as shown in the class diagram were:

- MI involves the use of, and visual reporting of real time data.
- MES are involved in the support of MI.
- MI enables the prediction of future performance and how decisions will affect that performance.
- MI uses visual metrics to communicate information and trigger and inform actions and decisions.
- MI informs business intelligence but is distinct from it.

- MI generates process understanding knowledge using data and metric analysis.
- MI enables quantified process improvement using real time data, process understanding and knowledge.
- MI supports manufacturing decision making.
- MI can be used to drive manufacturing process and metric improvement
- MI takes data, including real time data and helps create information and knowledge and enables the re-use of this data, information and knowledge for performance and process improvement.
- MI can be applied to any identified manufacturing metric
- MI can be enabled by, but is not inherently IT or an IT system.
- MI relies on compliance and sustainment processes to enable process improvement.

From these statements it was possible to create a summary statement for the purpose and intent of MI which could be used to create answers to the research questions:

In your view, what is 'Manufacturing Intelligence'/ what does it mean?

Manufacturing Intelligence enables good manufacturing decisions based on understanding of the current status and the ability to predict and control the outcome of any given decision.

What is it for/ what is its purpose?

Communicating and improving manufacturing performance as quantified by appropriate metrics, with the appropriateness of metrics also being informed by MI understanding.

How will we know when we have it (what does success look like)?

The organisation has the right metrics and targets in place to achieve the organisations objectives. Everyone is aware of the current performance of the organisation and process against the target metrics, likely future trends in the metrics and how to either maintain or improve performance to achieve the short, medium and long term targets.

5.6 Conclusions

Using the published literature and structured analysis of industry subject matter experts it has been possible to create model answers to the questions " What is MI", "What is its purpose" and " What does an future state with MI look like". When these answers are reviewed against the literature and subject matter expert input it can be seen that the derived answers provide consistency with nearly all the apparently diverse views.

Two key assumptions or statements that do not align with the derived definitions are: "MI is a sub function of MES systems" and " MI is an IT system....". Invalidating these statements resolves the apparent conflict with the ISA95 model timescales, where MES systems reside within this model and the timescales involved in MI processes, data and information. It also ensures that MI is understood to be a function of systems and processes enabled by IT but not solely an IT solution.

The new definition of MI is a high level description that provides an outline scope of the MI function. The class diagrams were used purely to structure the concepts or terms and the relationships between them and should not be considered complete concept maps for MI. Similarly, the process used shares some similarities with ontology development processes (Noy, McGuinness 2001, Blomqvist, Öhgren 2008, Wang et al. 2007, Frankovic, Budinska 2006, Zahedi, Sinha 2010), but the output is not intended to be the full MI ontology and the level of specialisation in the terms is clearly inconsistent. These terms, relationships, logic statements and the understanding developed through this work would prove to be key in the development of the full and foundational MI Ontologies which are described in the following sections.

The clarified scope was used to focus the next phases of concept development to ensure the MI domain onotology's represented in Figure 4-8 remained focused on the correct scope.

6 Ontology term, relationship and logic definition

6.1 Introduction

This chapter describes the elicitation and development of the domain concepts for the Systems, Manufacturing Systems and Manufacturing Intelligence domains in response to research question 2 (Section 4.3). These concepts included the relationships, logic and the development and identification of the subgroup of core concepts which form the core and domain ontologies and would ultimately be used to populate the instances within the ontology solution as shown in Figure 4-8. Some of these terms would be used in the later formalisation and testing research to test the core function of the solution concept including the ability to add instances from different timeframes.

Following the MI definition work and as shown in Figure 6-1, two other methods were used to identify and develop the domain terms, relationships and logic: subject matter brainstorming and business process mapping. This combination of techniques was used to avoid unconscious bias in the research and to ensure sufficient domain coverage which was identified as a risk with manual techniques (Blomqvist, Öhgren 2008, Li, Yang & Ramani 2009). The use of standard tools and methods such as BPMN, UML class diagrams, brainstorming and the ontology development approach ensured that the method could be reused in other domains in the future. The ability to reuse the method is vital to allowing the solution concept developed in this research to be taken further or re-applied in subsequent research.

Sections 6.4 and 6.5 describe the logic development based on the ontology competency questions listed in Section 4.2.1, and the subsequent development and structuring of the concepts and relationships identified.

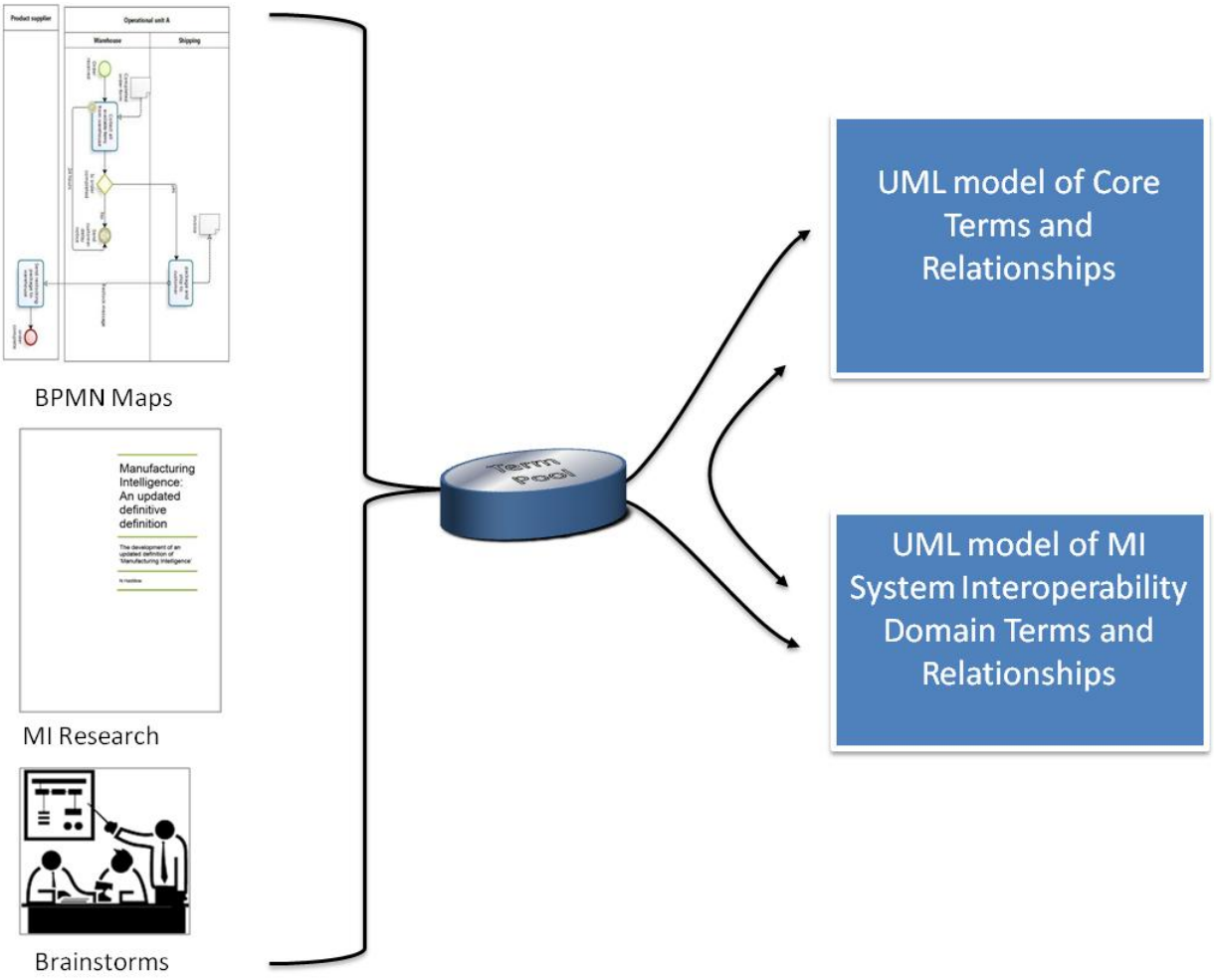


Figure 6-1 - summary of the domain term and relationship enumeration methods

6.2 Subject matter expert brainstorms

6.2.1 Unconstrained brainstorming and affinity mapping

A selection of 45 manufacturing systems industrial experts were gathered with representation from the Aerospace, Automotive, Electronic, Power Generation, Nuclear, Machine Tool and Naval industries with the experts representing large multinational to small (<5 person) enterprises. There was also input from manufacturing systems solutions providers as well as manufacturing systems forum organisers. Due to the number and diversity of input a number of different brainstorms were carried out. This was to allow free and open contribution without the risk of industrial intellectual property leakage between contributors, as well as the practical considerations of gathering such a diverse group in one event.

Following a briefing on the definition of MI used for this work to ensure the diverse views on the meaning of MI were resolved, the contributors were asked to volunteer words or concepts that they felt were strongly related to this field of activity. The standard rules of brainstorming were observed i.e. no input was discussed in detail, no criticism of ideas was permitted, everyone was expected to contribute, all input was recorded. After 2 hours over 200 terms had been volunteered. This input was an unstructured list of terms recorded on ‘sticky notes’ in no particular order. The attendees were then asked to group these terms into pools with similar meaning, functions or any other sort of linkage and where this linkage was not already present as a term, add it. This process was used to clarify the meaning of each term with the group as well as providing structure to the input. Finally they were asked to identify where terms could be linked to other terms by the relationship ‘is a’ link e.g. ‘dog’ is a ‘mammal’. This led the experts to define a basic taxonomy without unnecessary preconceptions of what the result of the work should be. The experts were deliberately not told about the whole process up front to avoid their own preconceived ideas of the MI domain biasing the resulting output.

Following this session this input was captured in the form of a UML class diagram. A list was constructed using the terms that represented the groupings or which had not been added to a grouping, this list would then be used as the prompting information for the prompted brainstorms. This list also included 1st level sub terms of the larger groupings. This was to avoid the prompts for the next stage being too open or high level, which could result in a failure to achieve an increase depth of detail in the prompted brainstorms.

It is important to note that while the author had views on the input e.g. some terms clearly seemed to be sub types of another term, no guidance or modifications were made to the

input at this stage. The input was not modified by the author until the later term structuring activity was started to ensure the input represented the subject matter experts input.

The following figures show a lightweight representation of the level 1 ontology. These diagrams represent the final results of the level 1 ontology development. Figure 6-2 shows on overview of the UML representation: the terms are structured around potential core concepts with the following figures showing the detail of each term grouping:

- Metric
- Data
- Constraint
- Target
- Response
- Timescale
- Manufacturing Method
- Interface
- Visualisation
- Collaboration
- Prediction
- Authority Level
- Standard
- Sustainment
- Person
- Analysis
- Traceability Item
- Status
- System
- Resource

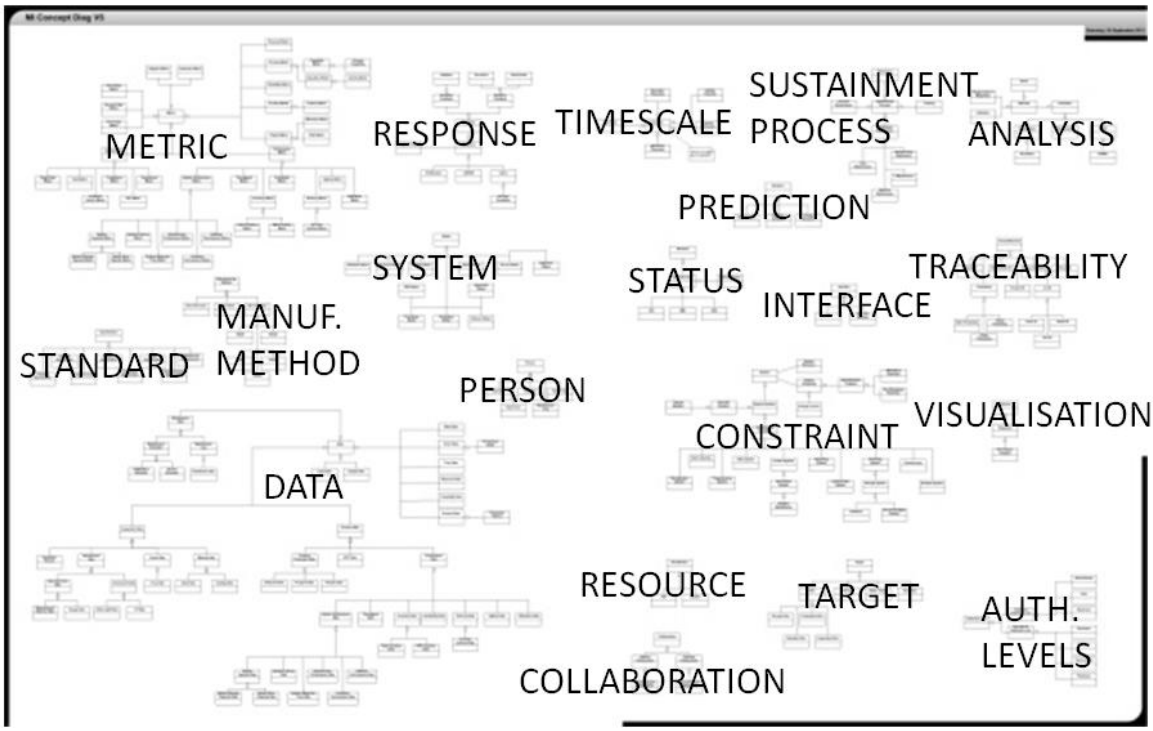


Figure 6-2 Overview of the lightweight representation of the level 1 ontology

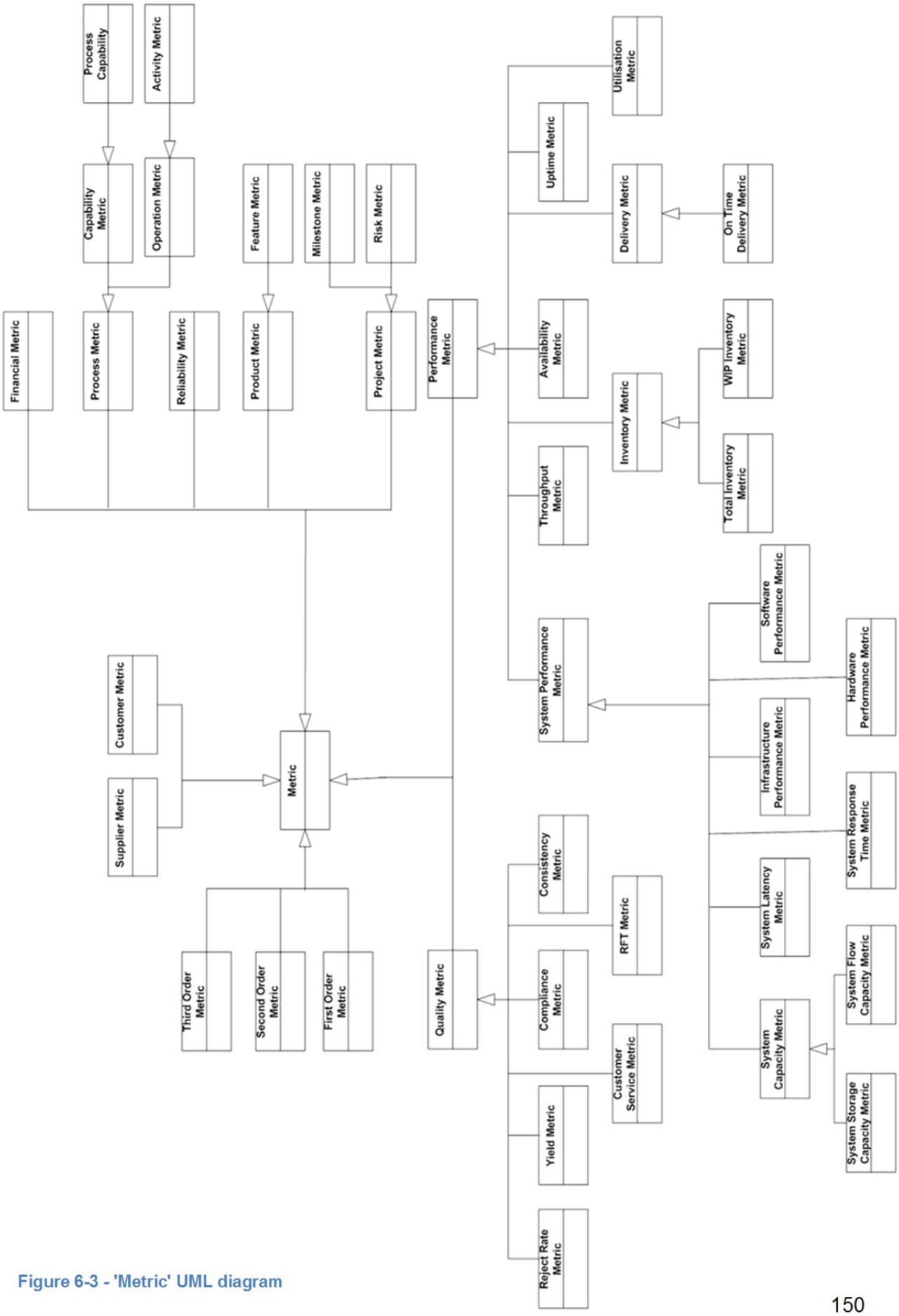


Figure 6-3 - 'Metric' UML diagram

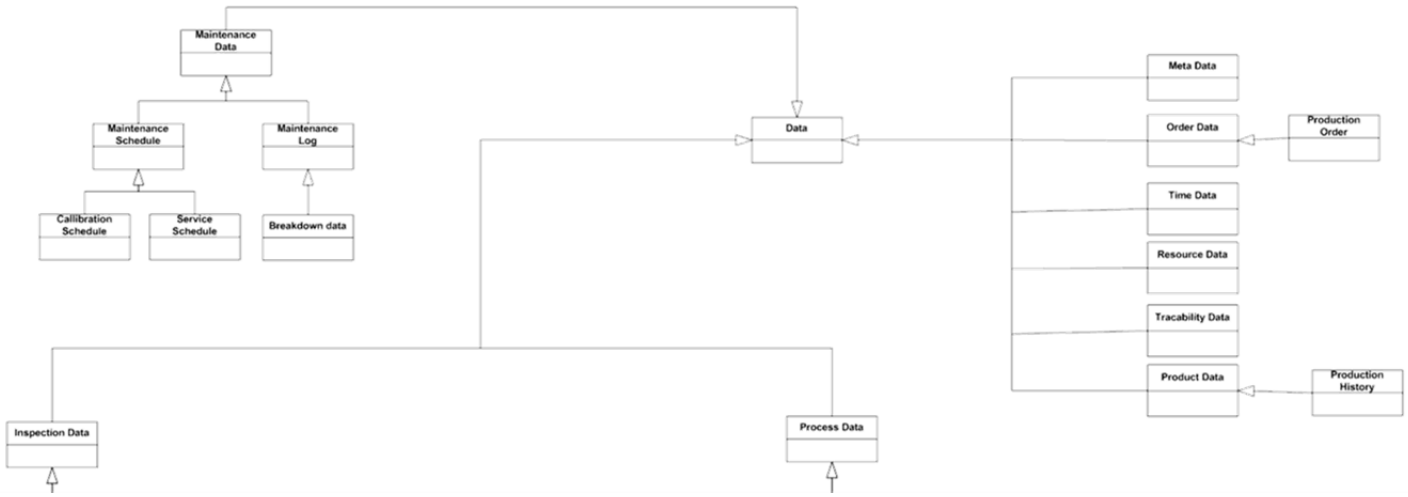


Figure 6-4 - 'Data UML diagram (part 1)

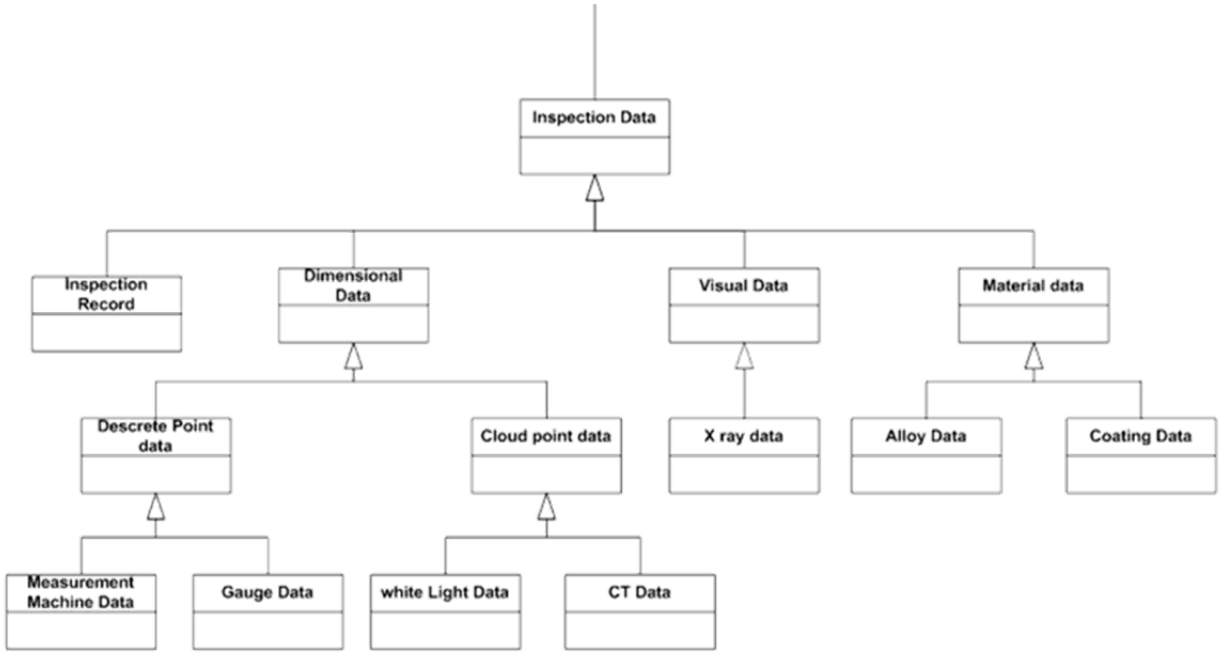


Figure 6-5 - 'Data UML diagram (part 2)

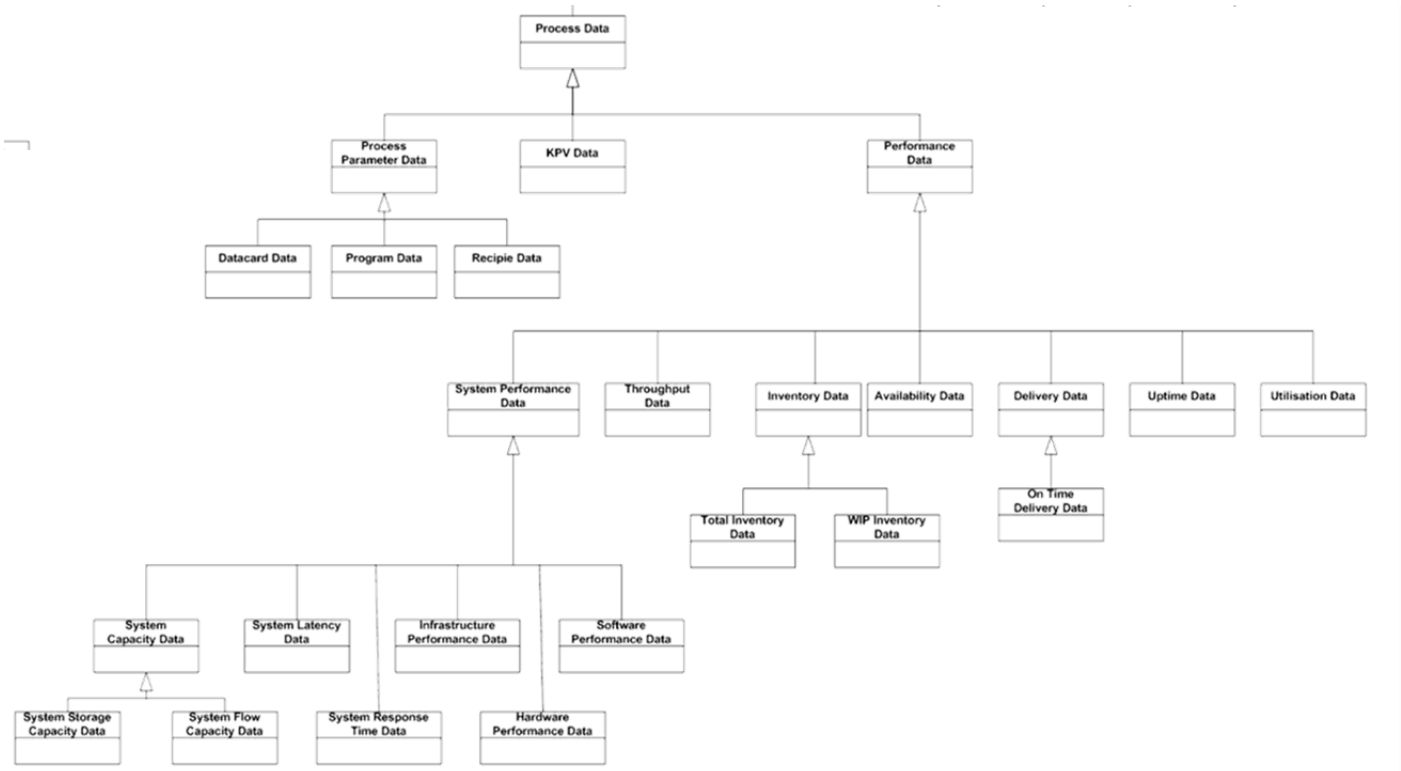


Figure 6-6 - 'Data UML diagram (part 3)

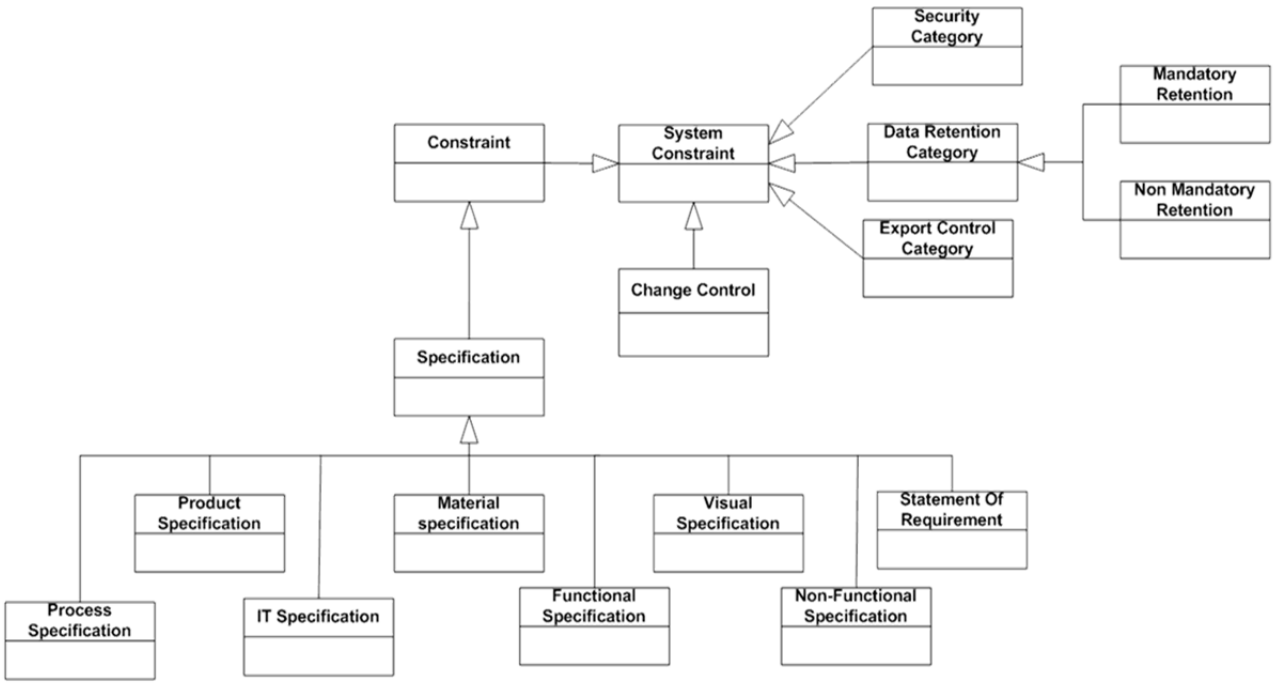


Figure 6-7 – 'Constraint' UML diagram

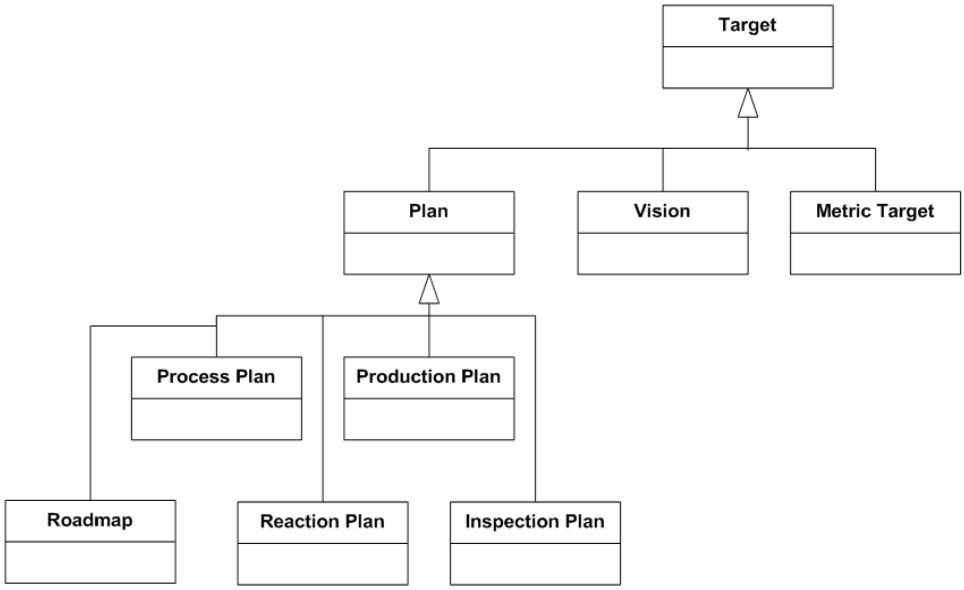


Figure 6-8 – 'Target' UML diagram

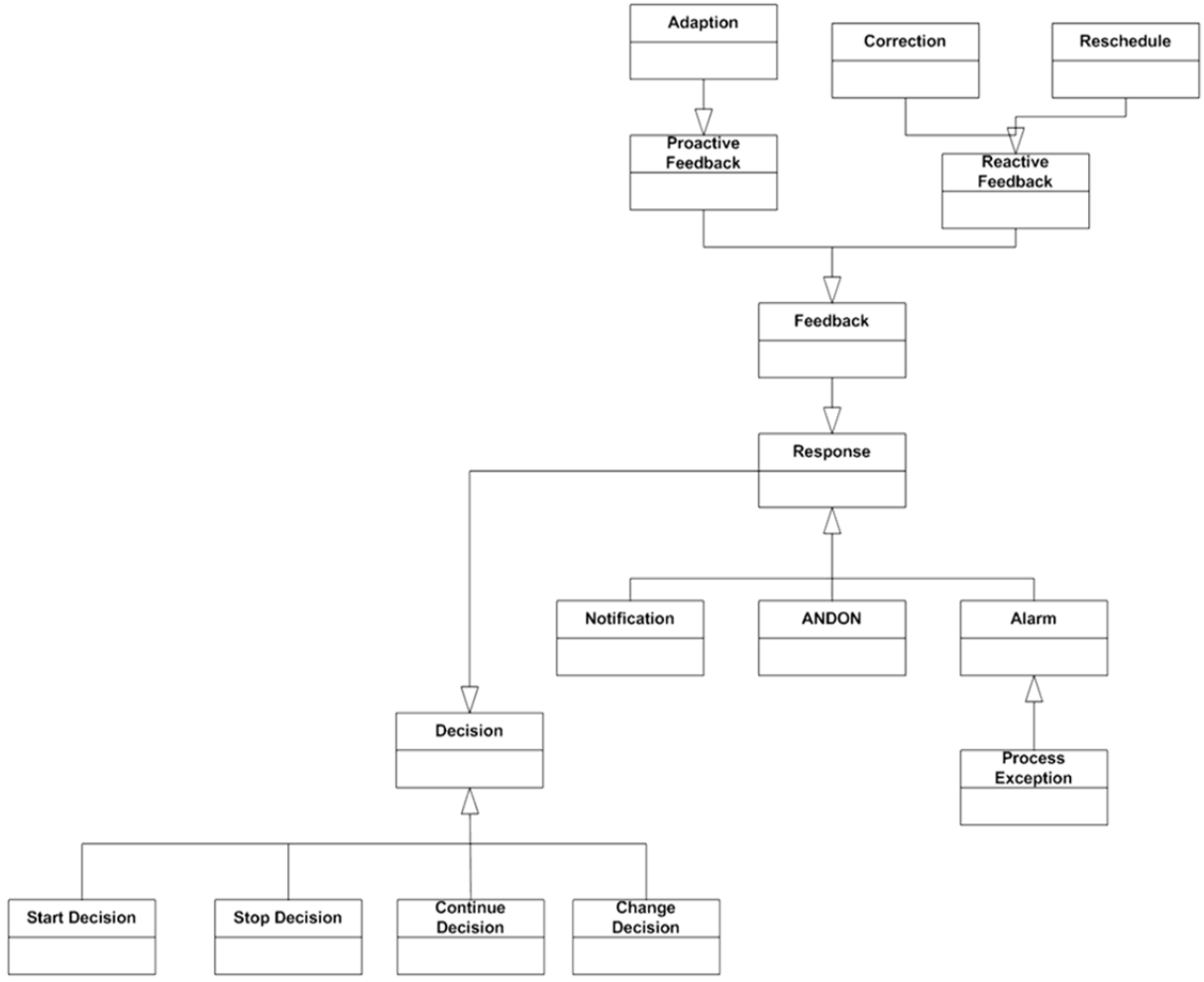


Figure 6-9 - 'Response' UML diagram

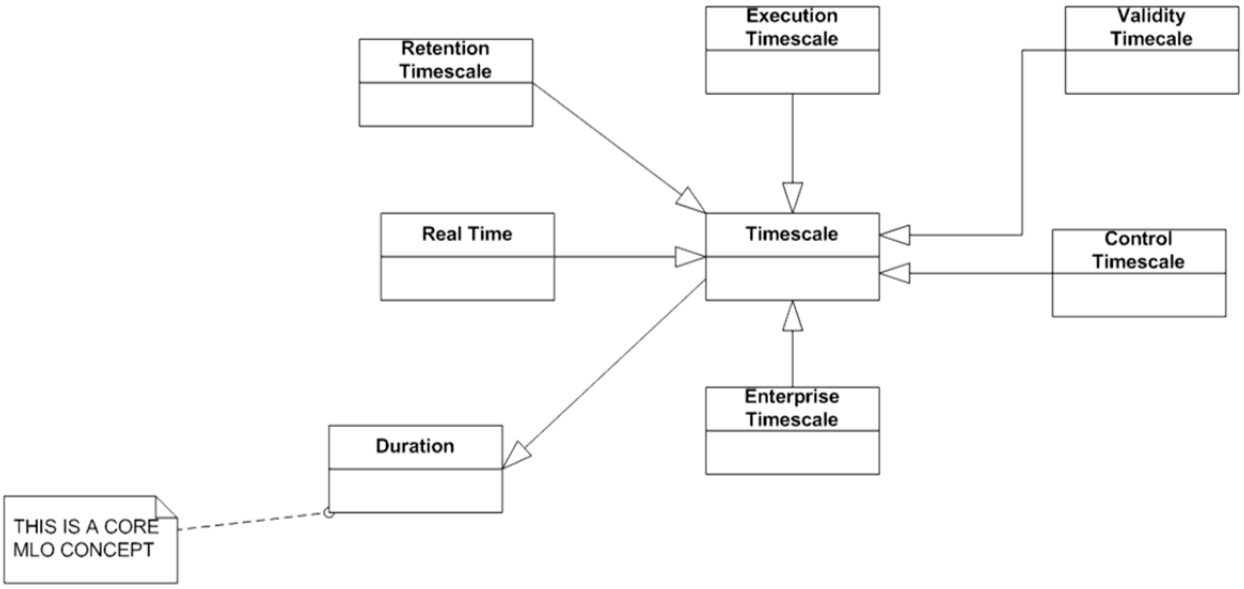


Figure 6-10 – 'Timescale' UML diagram

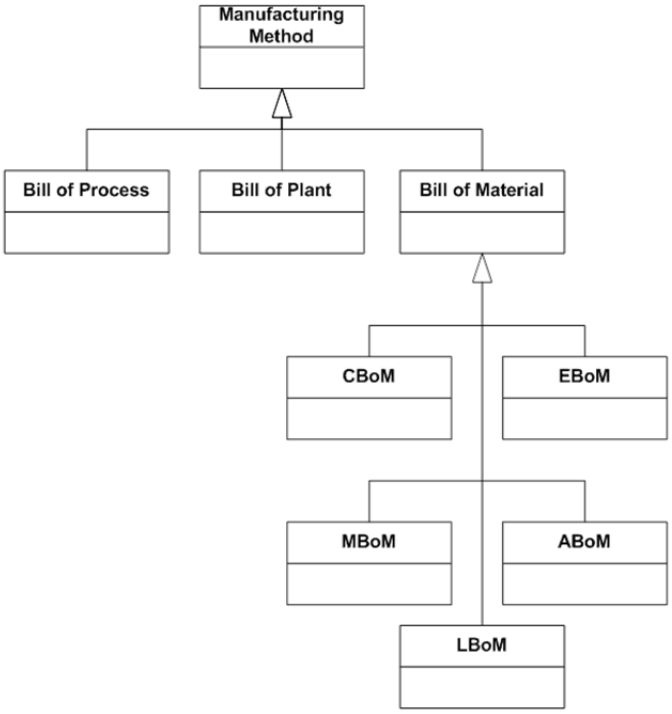


Figure 6-11 = 'Manufacturing Method' UML diagram

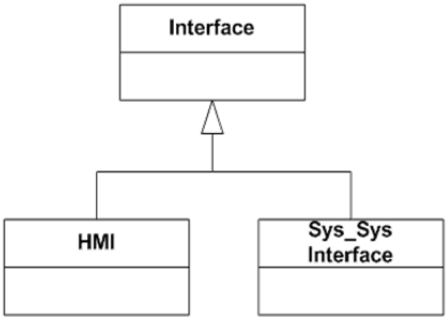


Figure 6-12 – 'Interface' UML diagram

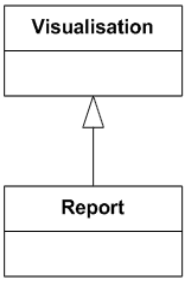


Figure 6-13 – 'Visualisation' UML diagram

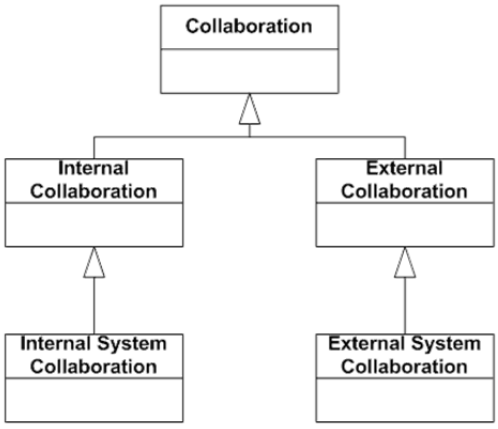


Figure 6-14 – 'Collaboration' UML diagram

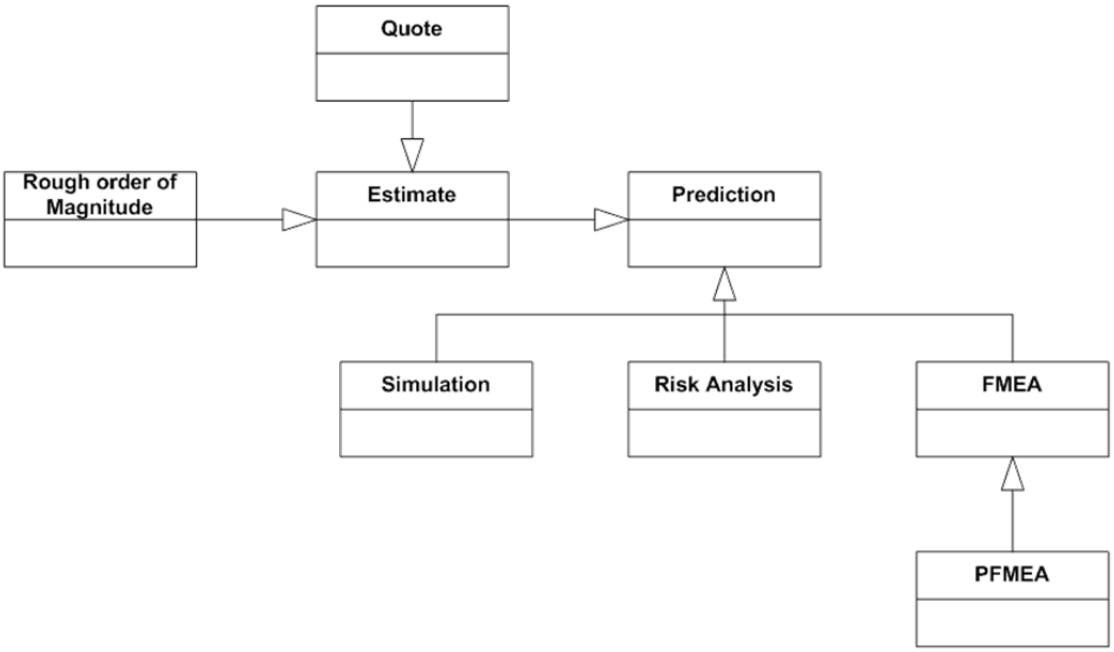


Figure 6-15 – 'Prediction' UML diagram

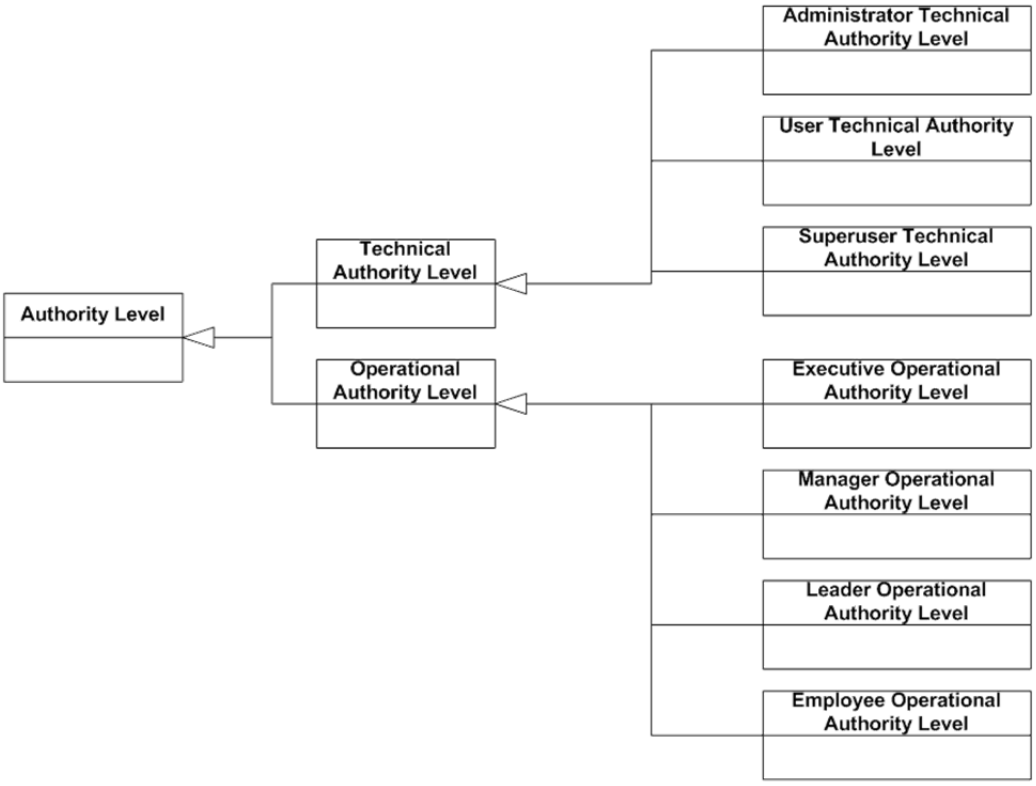


Figure 6-16 – 'Authority Level' UML diagram

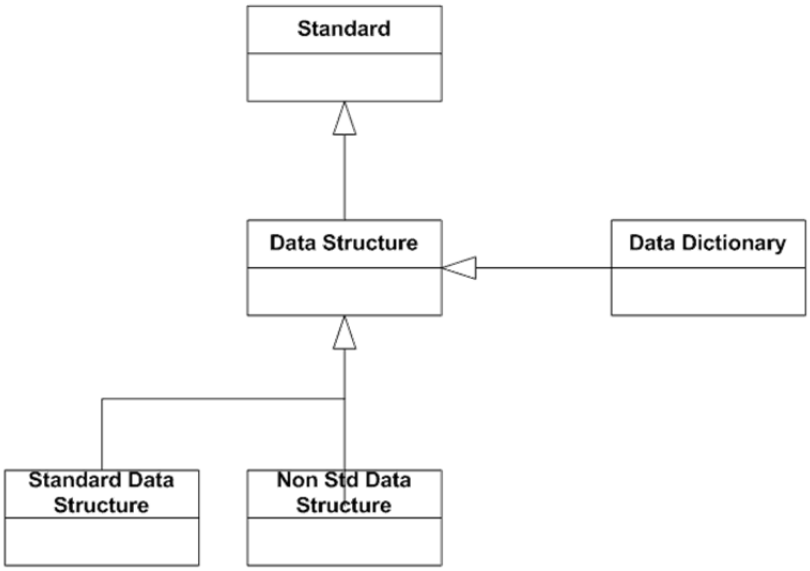


Figure 6-17 – 'Standard' UML diagram

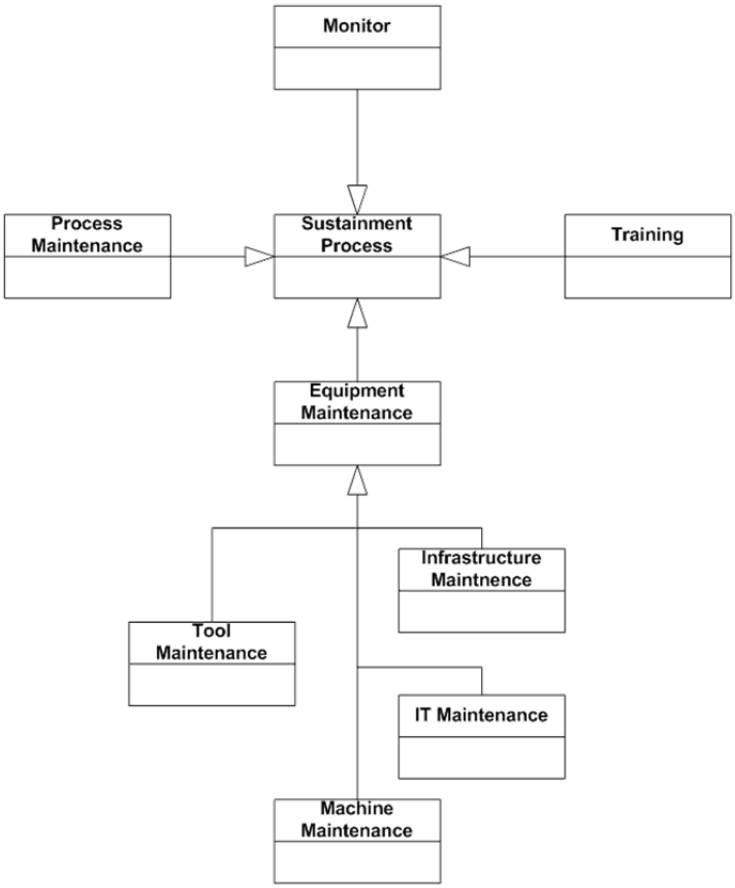


Figure 6-18 – Sustainment Process' UML diagram

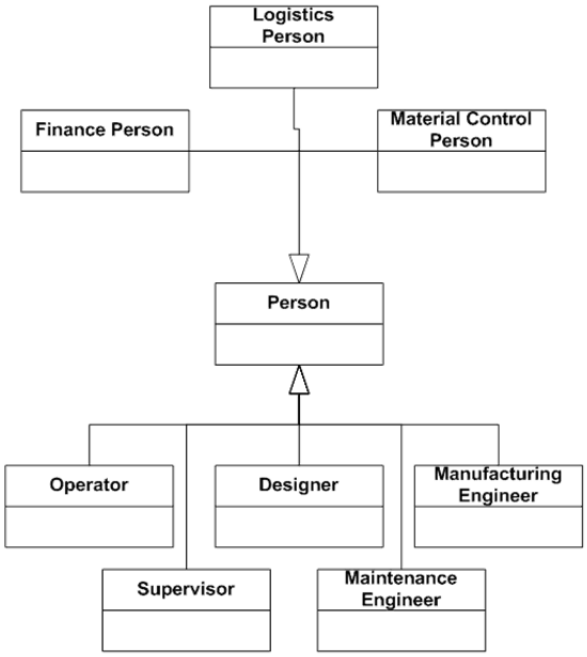


Figure 6-19 – 'Person' UML diagram

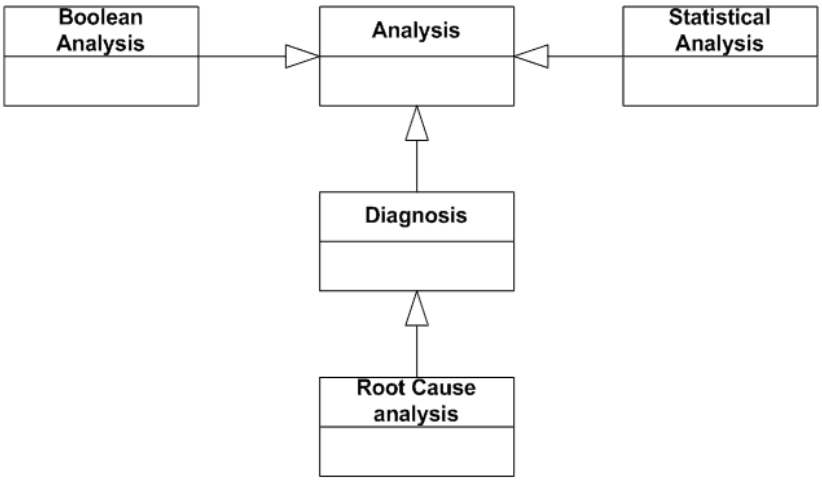


Figure 6-20 – 'Analysis' UML diagram

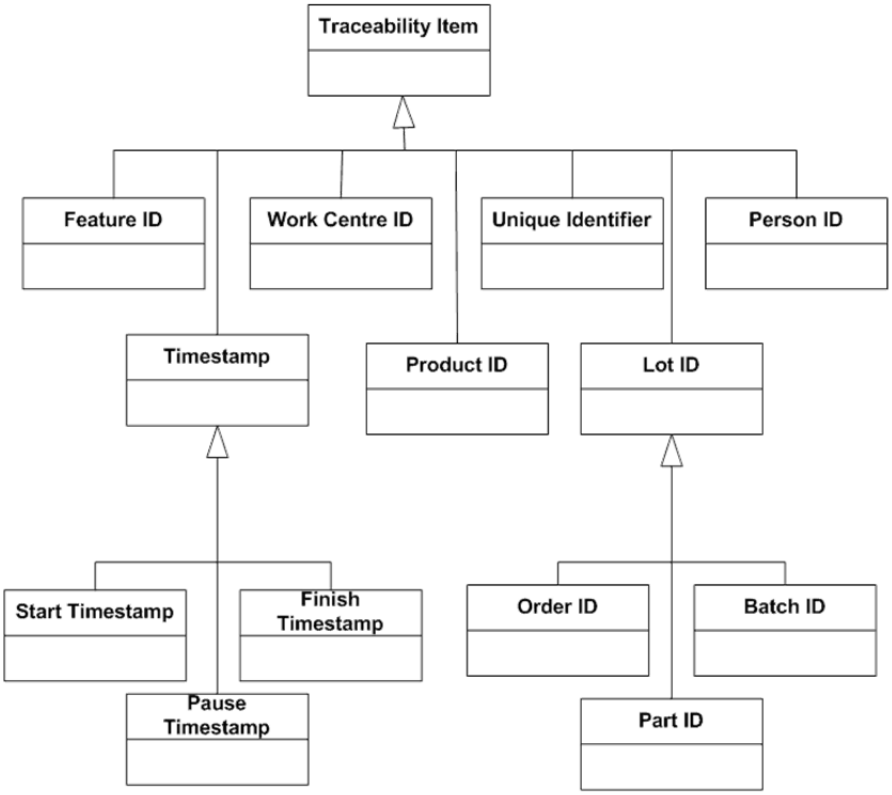


Figure 6-21 – 'Traceability Item' UML diagram

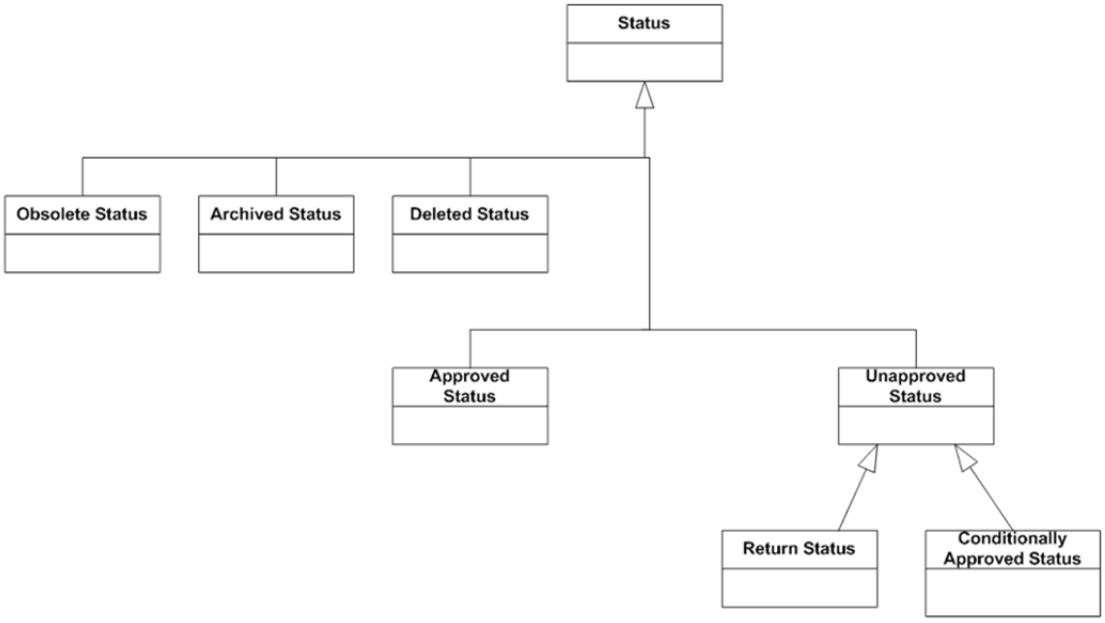


Figure 6-22 – 'Status' UML diagram

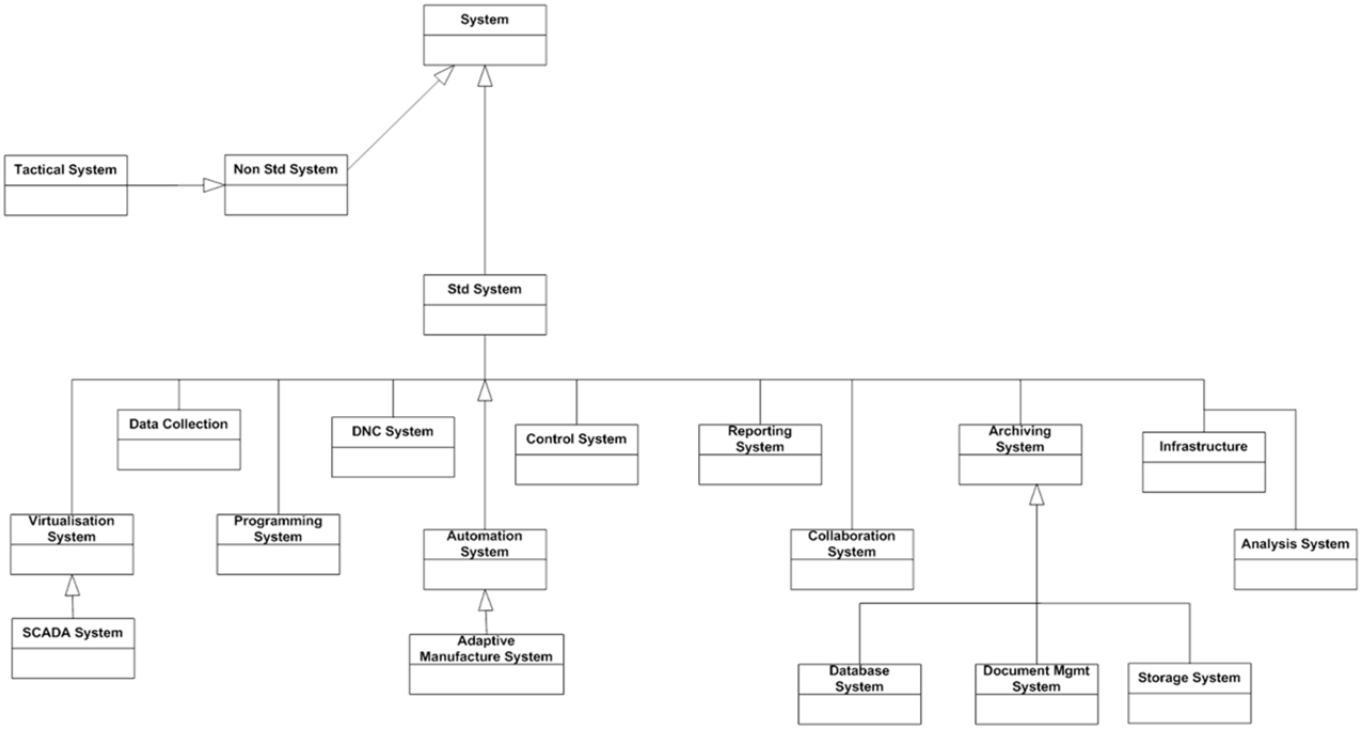


Figure 6-23 – 'System' UML diagram

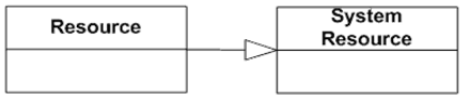


Figure 6-24 – 'Resource' UML diagram

6.2.2 Prompted brainstorming

15 of the subject matter experts described in Section 6.3.1 were gathered and presented with the UML diagram that represented the root terms from the term groupings generated in the unconstrained brainstorms. They were given the same brief on the MI domain definition and scope and were asked to generate term lists relating to each of the prompts. This group were not shown the full output from the previous group to avoid ‘leading’ or biasing their input.

This prompted brainstorming, in addition to the unconstrained brainstorming was used to try and ensure the breadth (unconstrained) and depth (prompted) of input.

Once the inputs were rationalised and combined with similar terms being referred to the experts for resolution, the resultant full diagram had over 350 terms and the prompt list had 42 terms.

The final part of this activity was to provide the subject matter experts a sheet with the 42 core terms printed on it and ask them to construct lines between the terms that have some form of relationship or interaction and try and define that relationship, of which an example is shown in Figure 6-39. It was interesting to note that the experts struggled with such an open brief and some required examples. Examples were given in a totally different domain (sports) to avoid biasing or directing their input. This input was used as part of the relationship development (see Section 6.5).

6.3 MI system business process mapping

In order to provide real world based input for the term pool and populate the solution ontology it was necessary to identify a source of instances and a method to capture them for this purpose. The approach taken was to use a structure process mapping standard (BPMN) that would allow the representation of real process and systems within a manufacturing facility. Figure 6-25 shows a simplified set of process steps that were used to identify relevant terms and logic in the BPMN diagrams. Although not shown, this process was highly iterative and required significant cross referencing of the identified terms with the subject matter experts and emerging UML class diagram taxonomies.

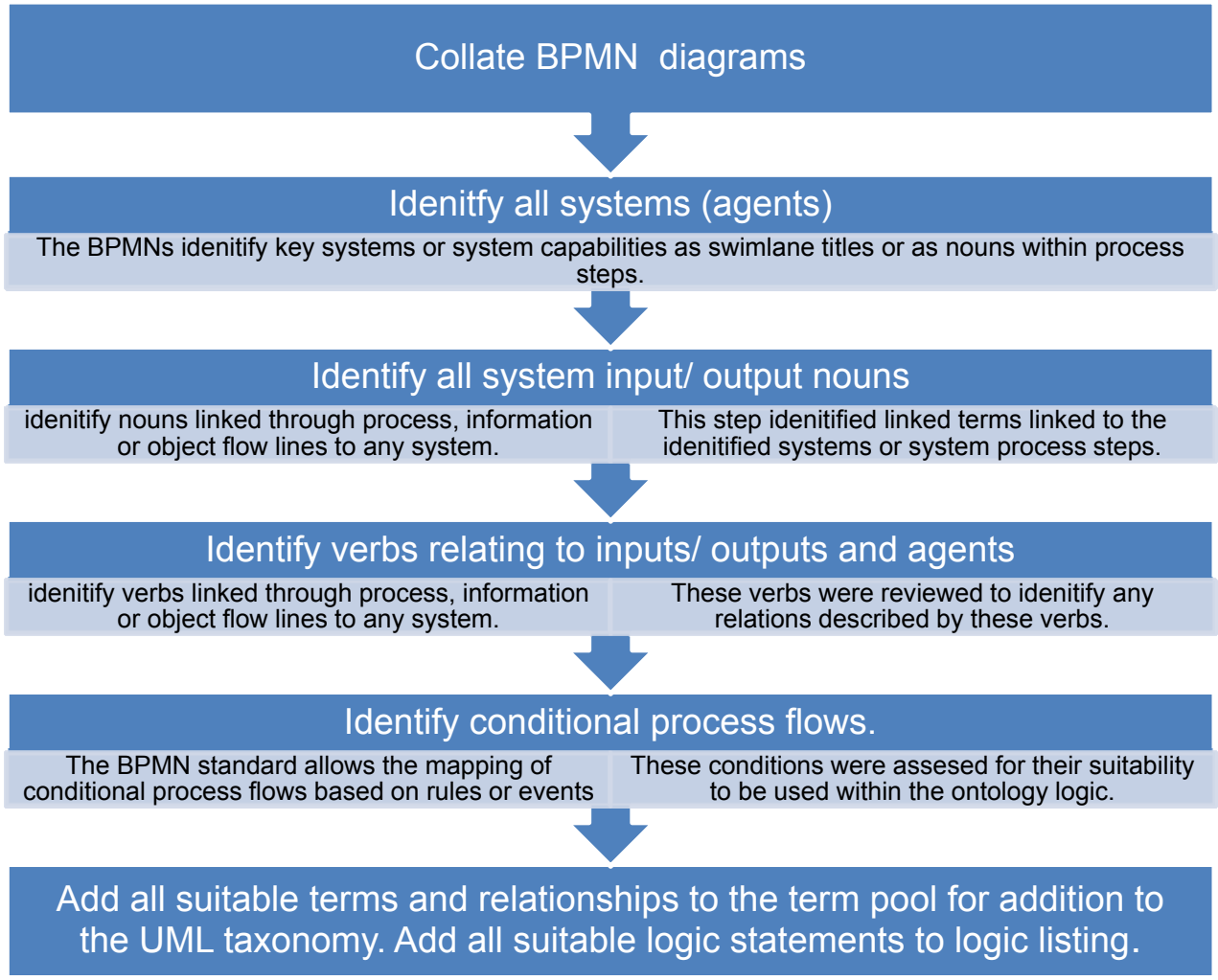


Figure 6-25 - The process steps used to identify concepts, relationships and logic within the BPMN diagrams

6.3.1 Term identification

The term pool was populated using 32 BPMN maps which cover all process steps within a specific state of the art manufacturing facility as represented in Figure 6-26:

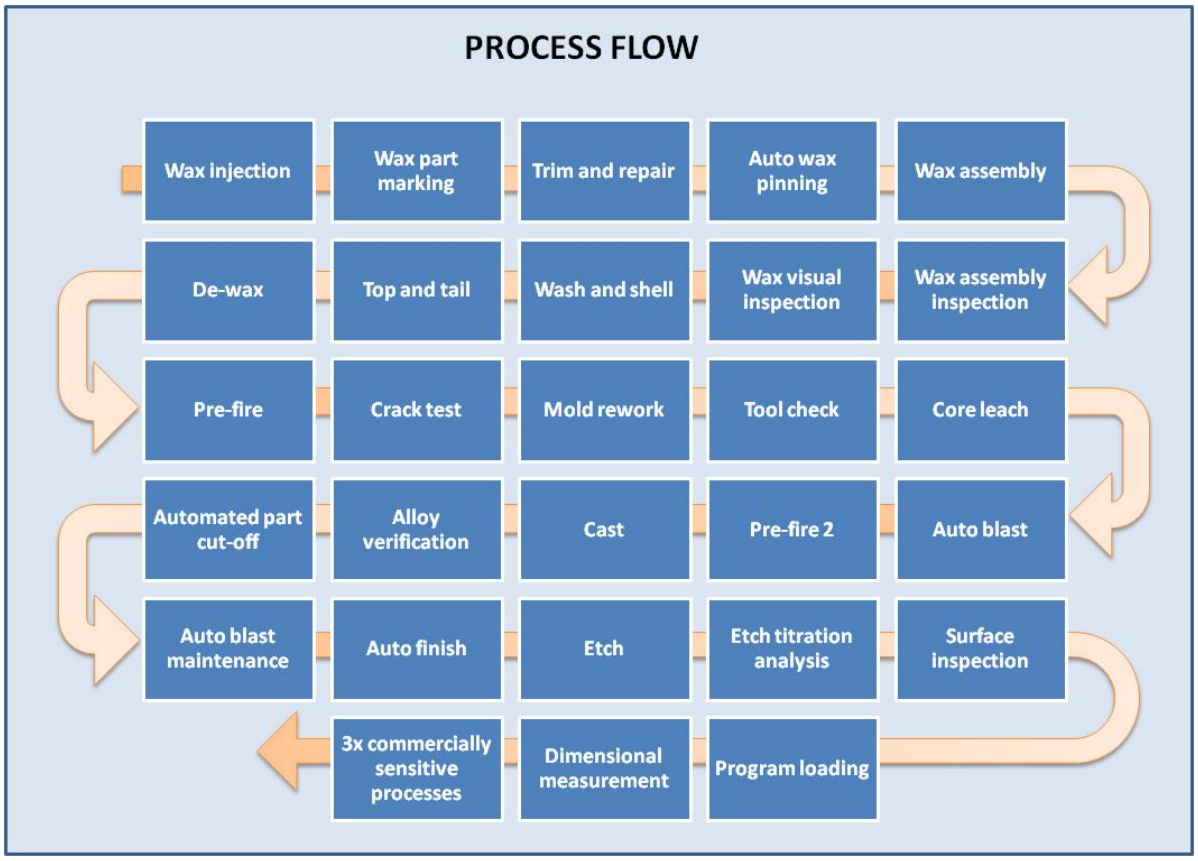


Figure 6-26 - The manufacturing process steps covered by the 32 BPMNs

A separate process map was created for every manufacturing process or defined sub process, with the 'agents' being defined as either the people, major equipment or systems. This allowed the information and data flows and activities to be clearly shown along with the affected agents. The BPMN maps are not included due to confidentially issues.

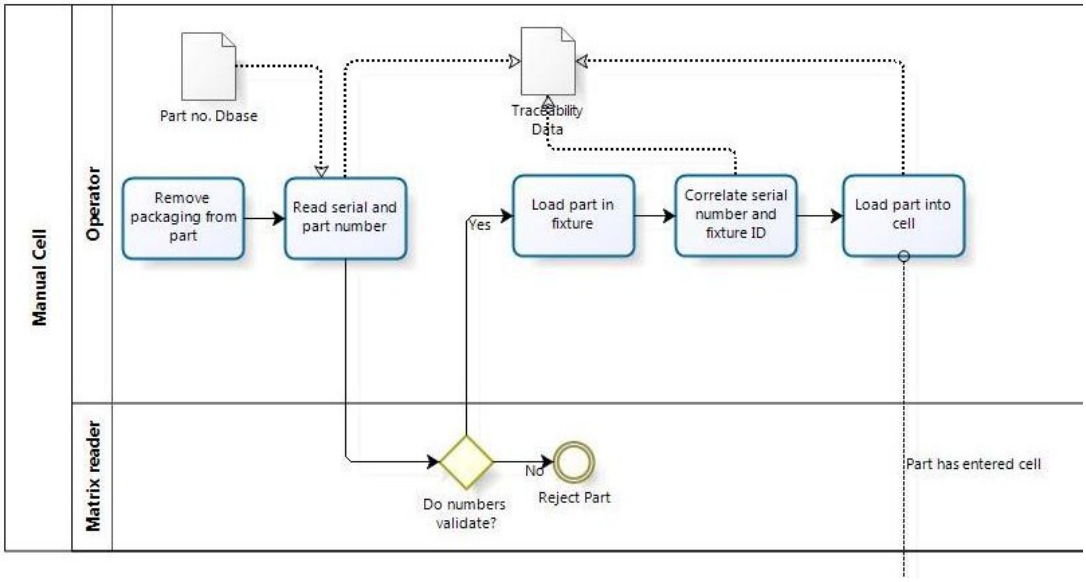


Figure 6-27 - A section of one of the BPMN diagrams

The relevant agents, information types, sources, flows and process described within the BPMNs were manually identified and added to the term pool. The BPMNs were also used to validate the defined logic (see Section 6.4) against real systems i.e. ensuring the logical rules are consistent with reality.

Figure 6-27 shows as section of a BPMN. This example shows the ‘matrix reader’ system as an agent with its own swimlane within the manual cell. The diagram shows the part serial and part numbers to be system inputs, a reject or accept signal the outputs, the operator as a resource and the question ‘do numbers validate’ as the system constraint.

Figure 4-5 described the solution concept as having 3 discrete ontology levels plus instances. That concept was modified based on a greater understanding of the BPMN and IODE tool functions, so that the second level of the ontology was actually the instances and the foundational level was clarified as being a core concept ontology as it was intended to be foundational within a specific domain. These changes are described in Figure 6-28 which shows how the core concept ontology is built upon by the more specialised Level 1 (manufacturing systems domain) ontology and the Level 2 ontology (which is comprised of the instances within the MI domain). When combined they are structured as shown, with the level one semantics extending the core concept ontology semantics and the Level 1 logic extending the core ontology logic. The logic is shown as being built on the semantics as the logic cannot be constructed without the semantics being defined. The Level 2 semantics (instances) and logic are then added to the taxonomy of the core and Level 1 ontologys.

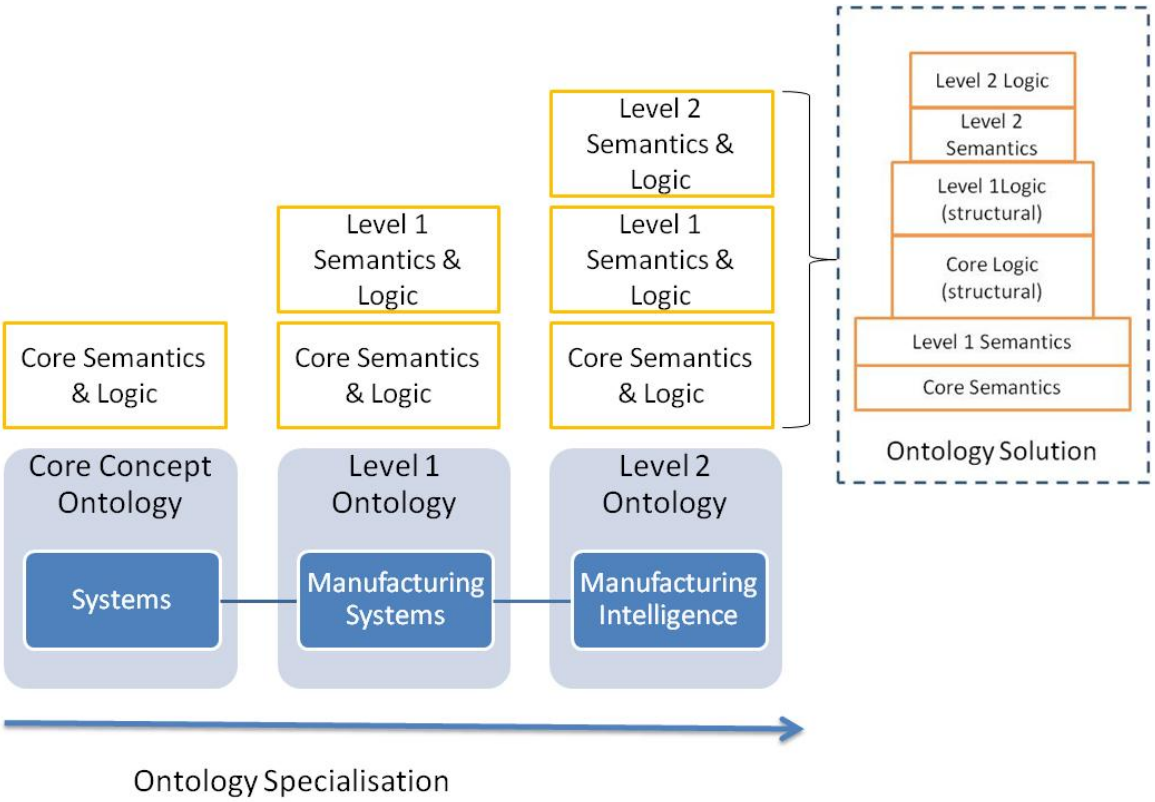


Figure 6-28 - Clarification of the ontology level model

The terms gathered from 32 BPMN diagrams, along with the brainstorming and MI definition research contributed to the creation of the core concept and Level 1 ontology term pool as described in Figure 6-1 and Section 6.3.1. leaving an outstanding requirement for suitable instances with which to populate the Level 2 (MI domain) ontology.

6.3.2 Instance identification

6 systems which were identified as ‘agents’ on the BPMN diagrams were selected to potentialy populate the knowledge base or ‘Level 2 ontology’ with facts about these systems. The 6 selected were chosen as they all operated in the initial cell of the facility. These 6 systems were:

- A Computer Aided Process Planning System (CAPP)
- A Manufacturing Execution System (MES)
- A Part Tracking System
- A Wax Injection Robot Control System
- A Wax Injection Machine Control
- A Wax Injection Cell Control

Figure 6-29 shows the process and systems which were identified within the initial manufacturing cell of the facility. Figure 6-30 shows how the BPMN information maps to the different ontology levels.

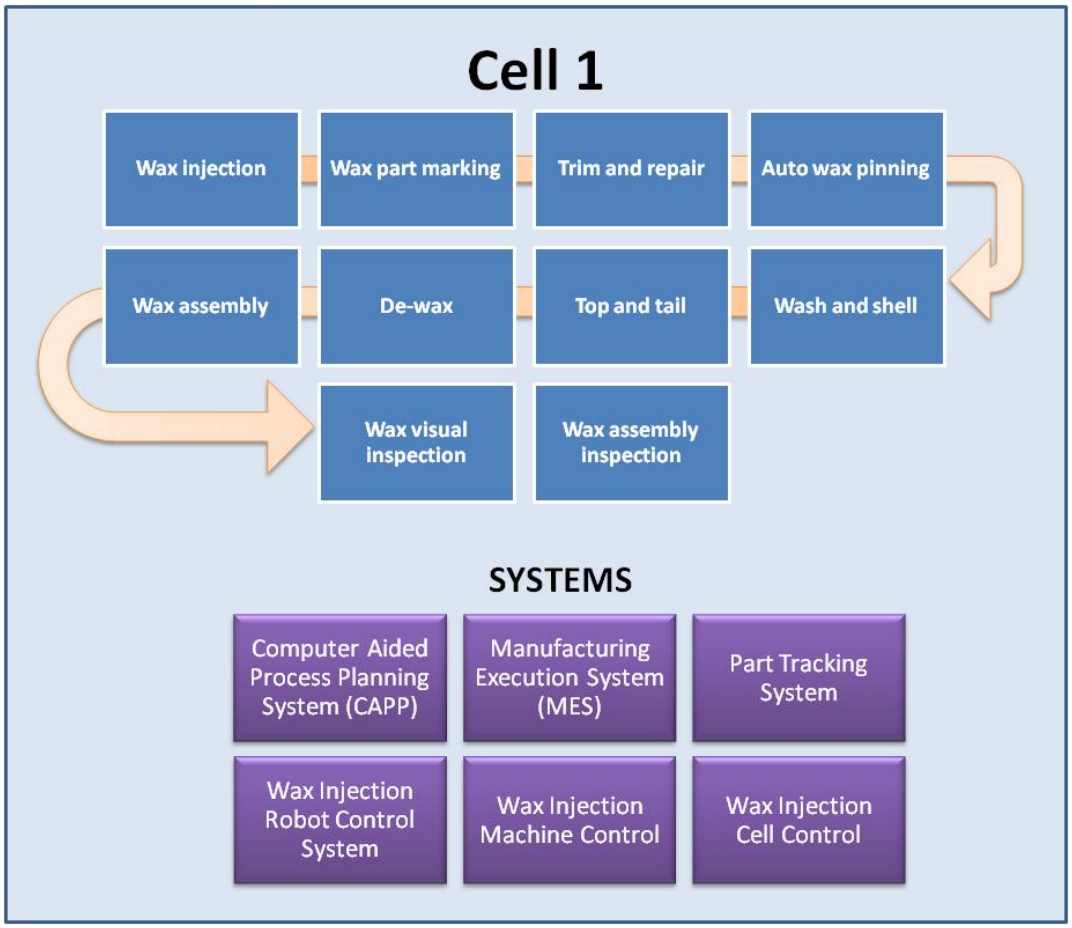


Figure 6-29 – The process and systems identified within the initial cell of the facility.

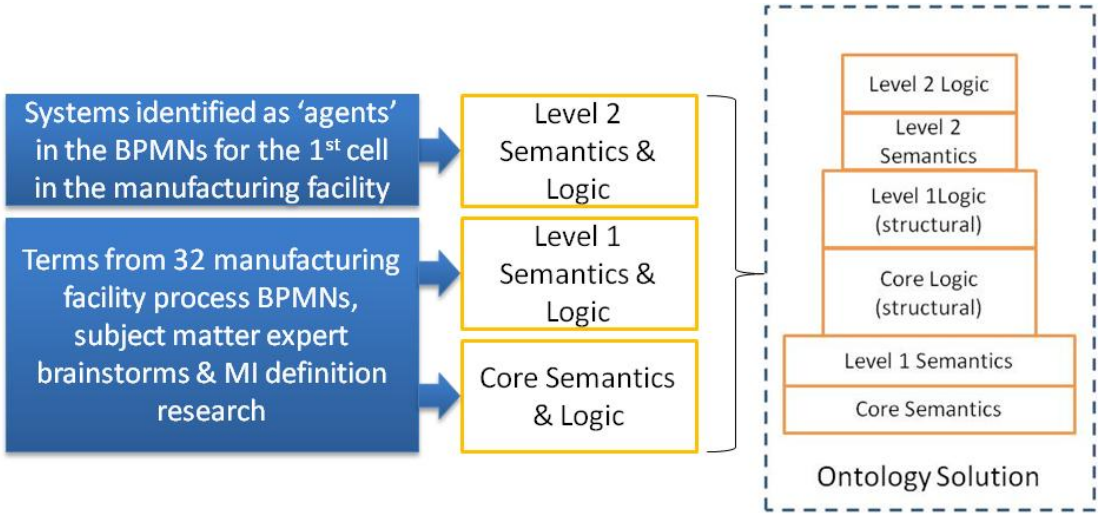


Figure 6-30 - The domain information is used to create the 3 ontology levels which when structured together form the proposed Ontology Solution

Following the identification of the core systems model as part of the core ontology UML diagram (see Section 6.5) it was decided to use this model to represent the 6 systems that would be captured for potential entry into the knowledge base (only 4 were eventually required). This visual representation allowed a simple overview of the systems inputs and outputs which could then be reviewed for the other applicable types against which they should be classified. The visual representations were also used with subject matter experts to prompt any missing facts. This proved highly effective as many of the facts were reclassified and a significant number of additional facts identified. The non system related facts and those related to the system inputs, outputs, resources and constraints could be declared. This phasing or fact declaration provided much needed structure to allow effective classification and coding of the facts.

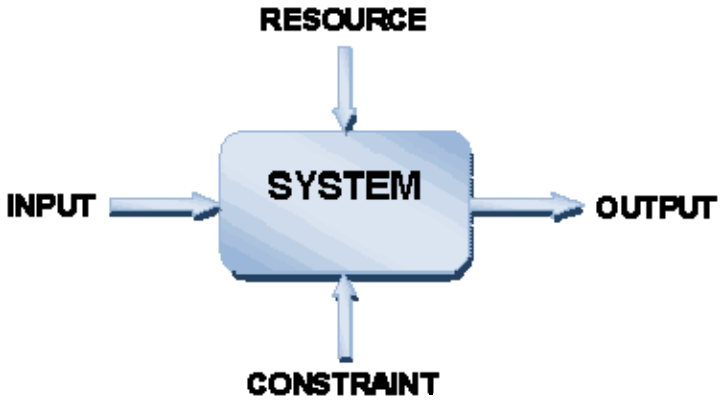


Figure 6-31 - Generic systems model

The following Figures detail the IDEF0 representations on the manufacturing cell systems.

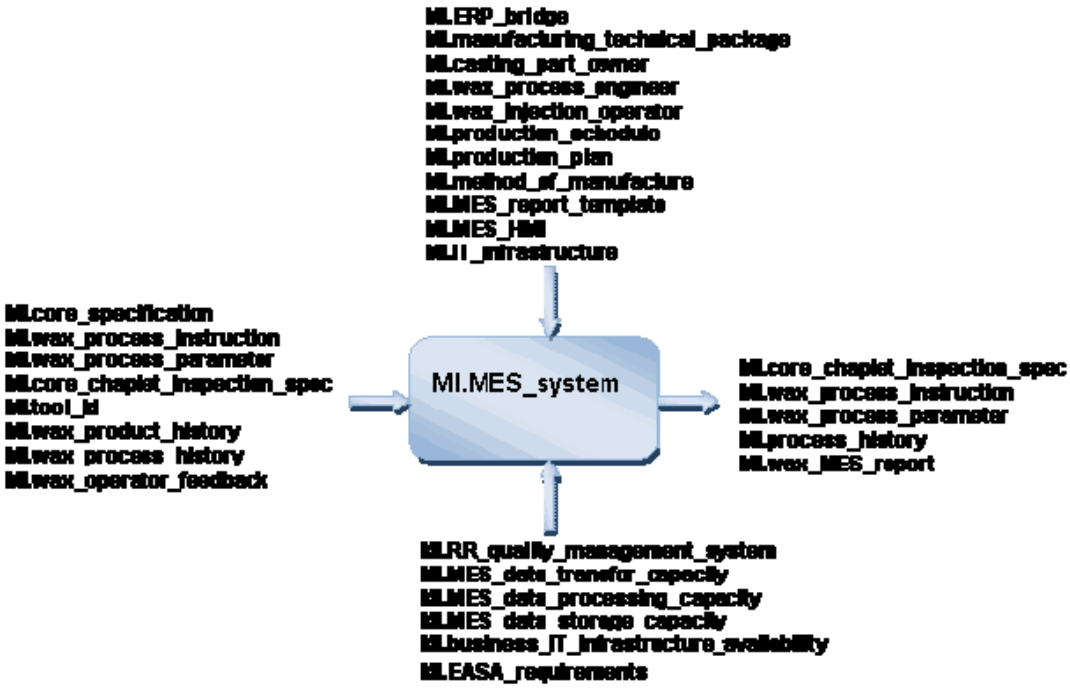


Figure 6-32 Generic systems model representations of the systems instances: MES System

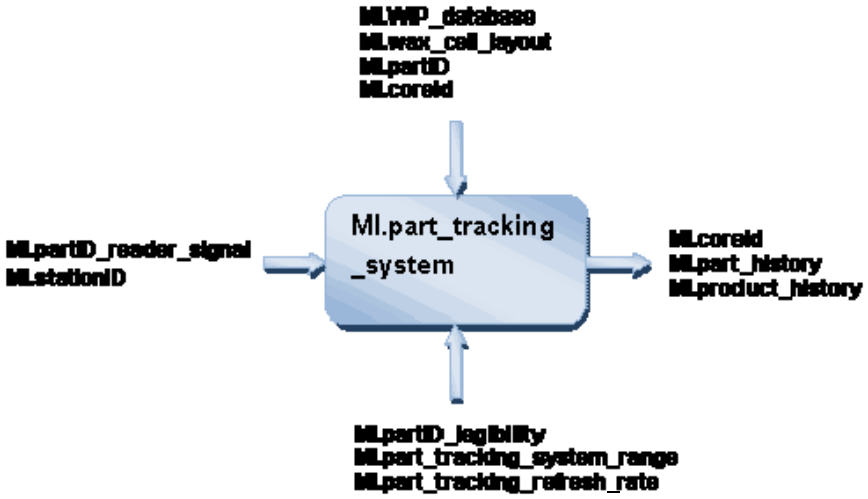


Figure 6-33 Generic systems model representations of the systems instances: Part Tracking System

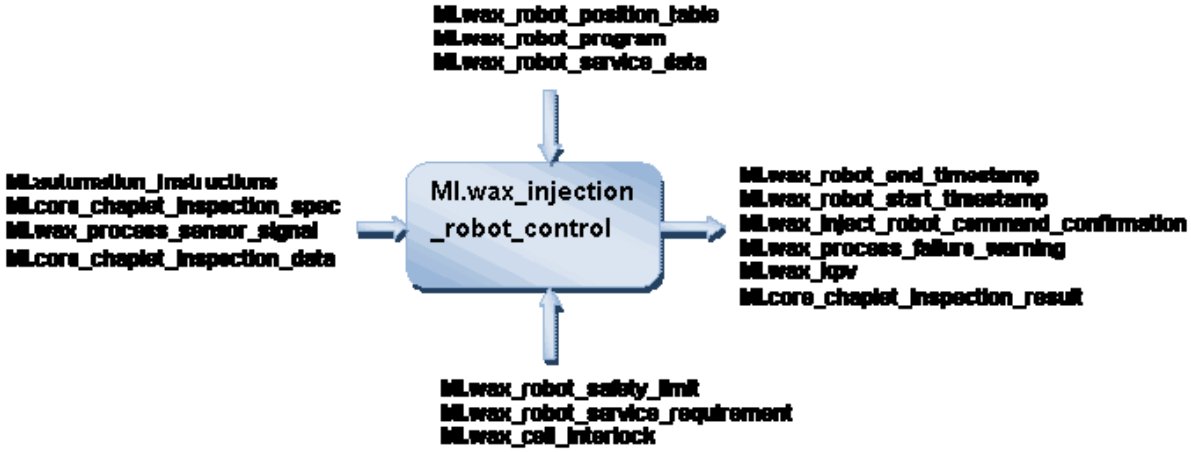


Figure 6-34 Generic systems model representations of the systems instances: Wax Injection Robot Control

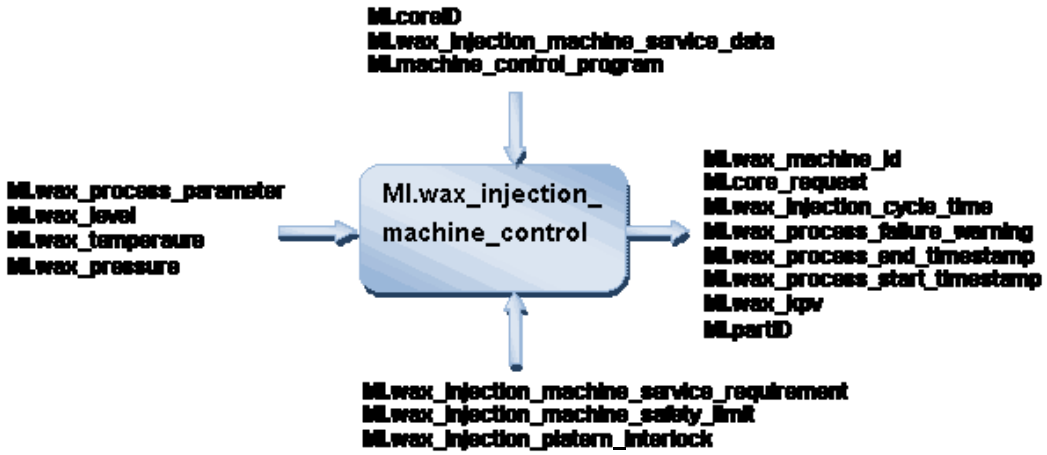


Figure 6-35 Generic systems model representations of the systems instances: Wax Injection Machine Control

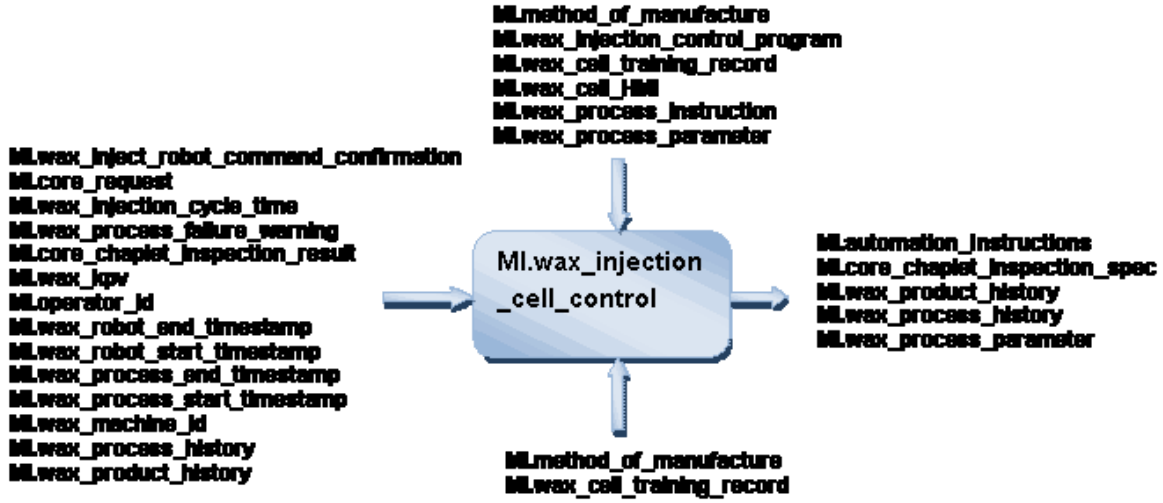


Figure 6-36 Generic systems model representations of the systems instances: Wax Injection Cell Control

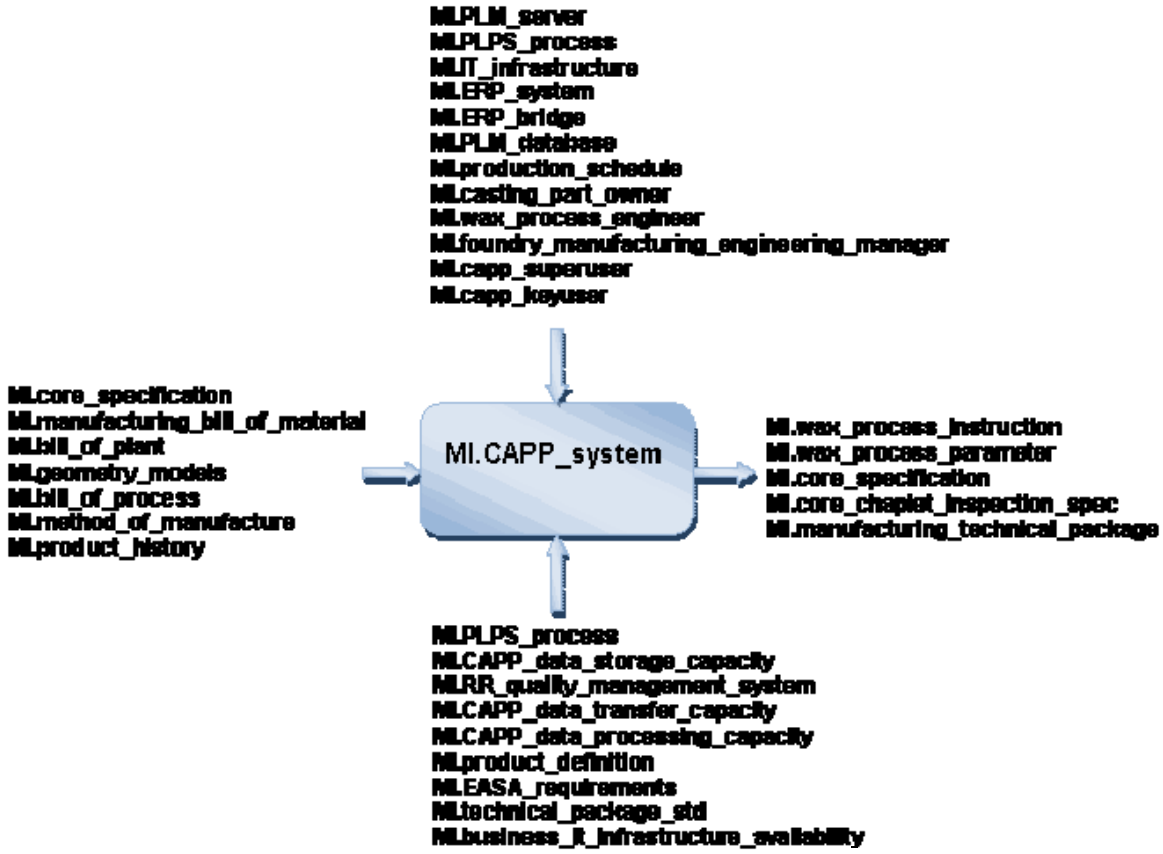


Figure 6-37 - Generic systems model representations of the systems instances: CAPP System

It was recognised early in this stage of the research that any systems identified and declared would be done so in the context of the wax injection cell. This means that any system that operates outside of this context would eventually need further facts adding. The ontology approach used in this research would allow further inputs, outputs, resources, constraints

and other related facts to be added within the standard structure of the extendable ontology in the same way that subject matter expert's additional facts were added.

6.4 Logic development

In order to constrain the emerging terms within the developing heavyweight ontology logical rules were required. These rules would have to constrain the terms within the combined contexts of both the MI systems domain and the requirements for interoperability.

The research described in Section 5.5 which resulted in the definition of MI generated some basic defining statements which could be used to help define logic and relationships later in the process. These were relatively loose descriptive statements that proved to be useful in creating a clear MI definition but required a level of interpretation to generate logic and relationships as described in Section 6.6.

Based on the findings from the literature review and industrial investigation (see Chapters 2 and 3) the challenges to systems interoperability could be grouped into 'themes' and these themes were then be used to identify constraining logic, to which a type or instance within the ontology must conform if it is to meet the requirements for interoperability. The logic themes and example logic are shown in Figure 6-38:

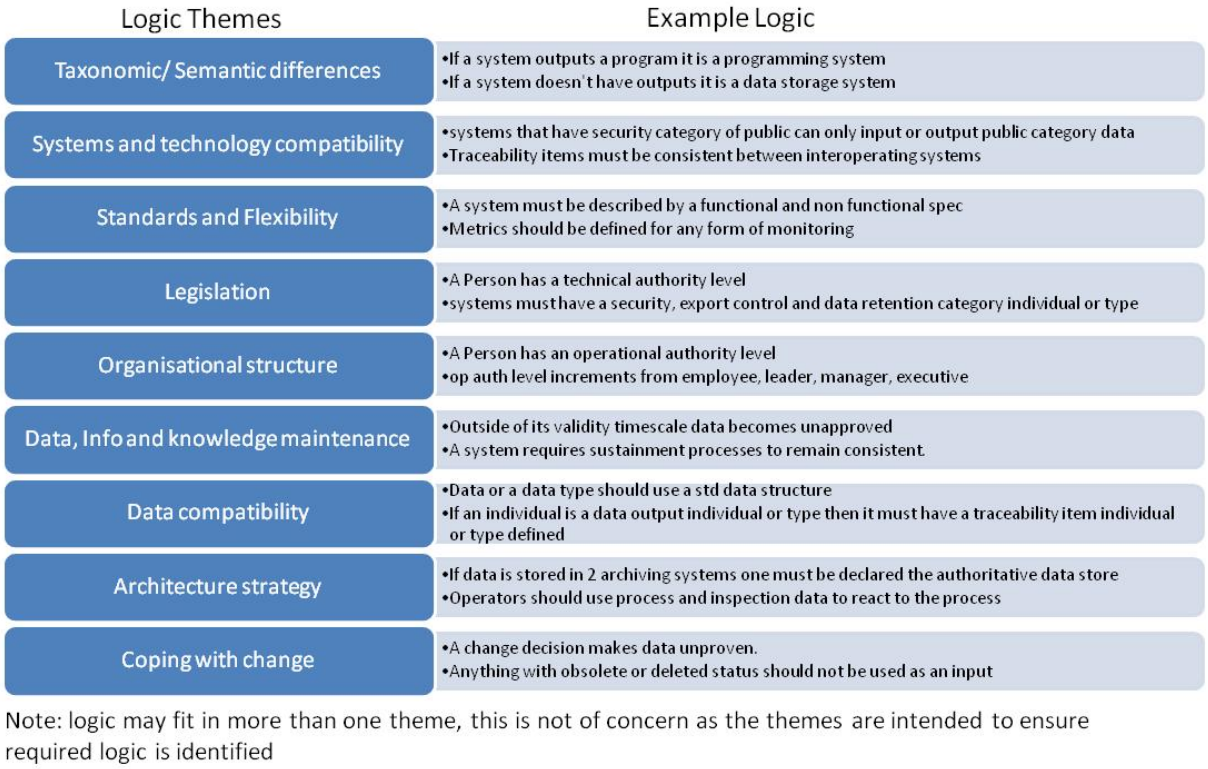


Figure 6-38 - Logic development themes developed from the challenges to interoperability research

The next approach was to focus the subject matter experts on the purpose of the ontology and list rules and relationships between individual terms using the UML taxonomies as a prompt. The development of these rules was structured around the key themes previously

identified as the challenges to system interoperability. This work resulted in the following statements:

- Each data instance has a technical authority level
- Each data instance has an operational authority level
- Decisions about systems can only be made by super users or administrators
- Non standard systems specifications require manager and super user approval
- A Person has a technical authority level
- A Person has an operational authority level
- If a person's operational authority level is less than the data authority level they cannot access the data
- If a person's technical authority level is less than the data authority level they cannot access the data
- Having technical authority does not give a person operational authority
- A system should use a standard data structure
- Anything with obsolete or deleted status should not be used
- For something to be approved, all data must be proven or approved.
- A change decision makes proven data unproven.
- If data is not approved it is be default unapproved
- The use of unapproved data within a system renders the system output unapproved
- Users or employees cannot change unproven to proven or unapproved to approved
- Outside of its validity timescale data becomes unapproved
- Equipment maintenance requires maintenance data
- Output data should must have a timestamp
- Finish data Timestamps cannot precede a start timestamp.
- Increasing a person's authority requires some sort of training
- For a person to work with a system there must be a HMI
- For two or more systems to work together there must be a system-system interface defined
- For 2 systems to measure the same metric they must use the same data types.
- For decisions to be consistent across people and systems the same data or metrics must be used.
- A system requires sustainment processes to remain consistent.
- People require training to remain consistent
- The durations between training to maintain person consistency must be defined.
- Maintenance intervals for a equipment must be defined

- The consistency of people and systems should be monitored.
- Metrics must be defined for any form of monitoring
- Targets must be defined for any form of monitoring.
- Reactions must be defined for any form of monitoring
- A system must be described by a functional and non functional spec
- If a system uses IT there must be an IT spec.
- Proactive feedback uses input data
- Reactive feedback uses output data
- Responses require output data
- Using non standard data structures requires super user authority
- Specifications must be approved by people with leaders and super users authority or higher
- System inputs, outputs must be specified for a system
- System resource and constraints should be specified for a system
- Metric targets have a validity timescale
- Metric data has a validity timescale
- A system has a data type if it is an input, output, resource or constraint
- If a system outputs a program it is a programming system
- If a system has data inputs but no data outputs it is a data storage system
- If a system is the only store of a data type it is an archiving system
- Data should not be stored in 2 archiving systems
- If data is stored in 2 archiving systems one must be declared the authoritative data store
- Data should have a data retention category (mandatory/ non mandatory)
- Data should have and export control category (export controlled/ non export controlled)
- Data should have a security category (public, private, secret, top secret)
- systems should have a security, export control and data retention category
- If an system doesn't have a security category if should be set to public
- If an system doesn't have an export control category it should be set to non export controlled
- If an system doesn't have a data retention category if should be set to non mandatory
- systems that have security category of public can only input or output public category data

- systems that have security category of private can only input or output public or private category data
- systems that have security category of secret can only input or output public, private or secret category data
- systems that have security category of top secret can only input or output public, private, secret or top secret category data
- systems with an export control category of non export control can only input or output non export control data
- systems with an data retention category of non mandatory control can only input or output non mandatory retention data
- If a data item is the input of one system and the output of another these systems are series associated
- If a system outputs data to another system it is an input for that other system
- A system functional spec must define a performance metric and metric target
- Systems that share the same input, outputs resources or constraints are parallel associated
- If a system outputs data that is not input into it, it is the data source for that data

The term pool, taxonomy and relationship development the logic development were all iterative and interlinked, with developments in one area requiring a form of regression assessment to ensure all logic and relationships still held true and comply to the constraining logic. This logic was also checked against the BPMNs (representing real systems and process) to ensure the correlation with reality. At this stage this was a manual activity, and the complexity due to the number of terms and relationships was highlighted by the number of issues that would be highlighted during the ontology coding activity, where these checks were carried out by the software in a very rigorous manner.

6.5 Concept and relationship development and structuring

The MI definition, subject matter expert brainstorming and process mapping activity resulted in a large term pool, and a smaller subset which represented parent terms for the terms groupings within the pool. This large pool was structured into a taxonomy although at this stage the subset was unstructured.

When the output from the relationship identification exercise carried out with the subject matter experts (see Section 6.3.2) was reviewed it was clear the area within the Manufacturing Systems domain that each expert worked, heavily influenced their input: the Computer Aided Process Planning experts represented the relationships relevant to them, which were different to those of the Manufacturing Execution System experts. Therefore all of the 16 sheets provided were reviewed simultaneously through a manual review and all identified links between terms were reviewed to identify whether common parent relationships existed. These common relationships were then manually checked to ensure none of the other defined relationships conflicted with them. It is noteworthy that despite the large number of relationships, no conflicts were found. These were considered to be the initial relationship list for the concept groups. The next, and very arduous activity was to apply these relationships to the groups and ensure that the relationships hold for all members of that grouping, and issues were identified, a decision had to be made as to whether the term had been incorrectly grouped or added to the taxonomy or whether the relationships was either invalid or only held for a sub branch of that group within the taxonomy.

Figure 6-39 shows two examples of relationships defined by different subject matter experts through the prompted brainstorming described in Section 6.2.2. The two experts identified that the terms ‘Decision’ and ‘Response’ have a relationship, however one defined this as ‘requires a’ and the other ‘will determine a’. In this case the relationships while different are compatible once interpreted. This interpretation was checked with the experts for accuracy, and the relationships both accepted for future consolidation or inclusion in the ontology.

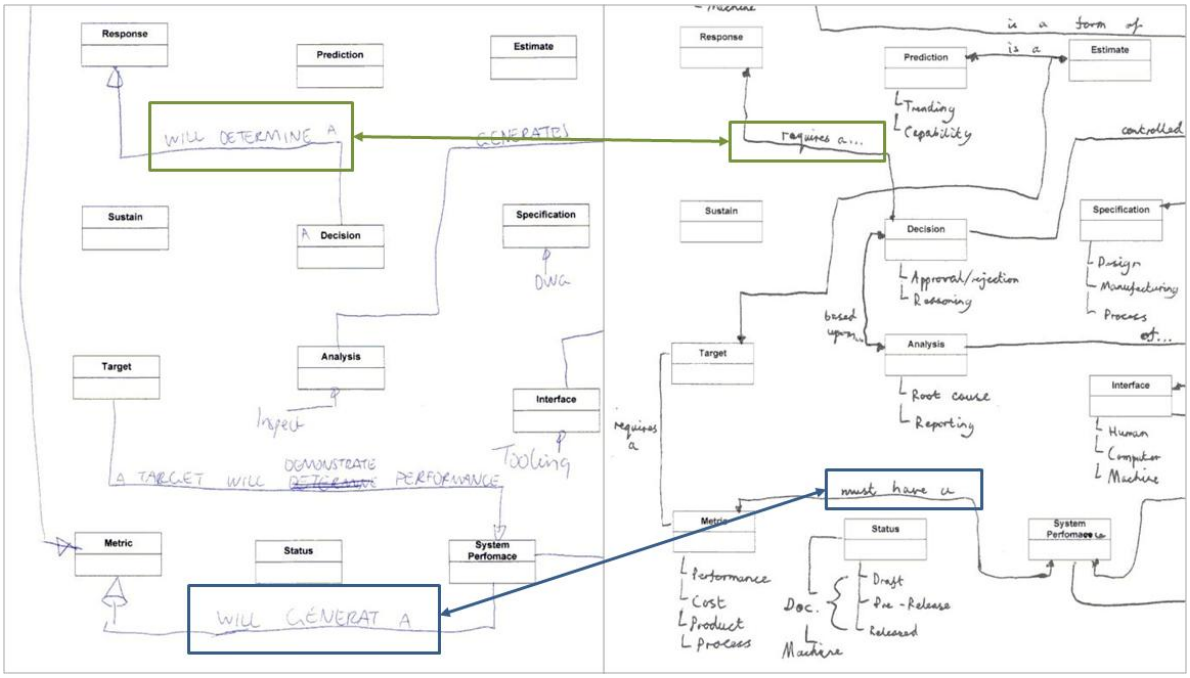


Figure 6-39 - Examples of differing subject matter expert generated relationships between 'parent' terms for the UML diagrams

The next phase of the research was to review the logic (see Section 6.4) and use this to define and then test the relationships between the term pools in addition to the previous work. Similarly to the previous relationship development exercise, this led to the development of generic relationships but also led to significant developments in the terms pools as the application of relationships necessitated splitting some of the initial groupings and the combining of other under a newly identified parent term. This activity was highly iterative and continued even through the coding activity as can be seen in the in the development change log (see Appendix A).

These relationships form a core or ‘structural’ part of the ontology logic, in that all future types or instances would have to conform to these relationships, as they are defined from the logic of Manufacturing Systems interoperability between the parent terms of the term groupings. These parent terms, through the ongoing development, would come to be the core or foundational terms described within the solution concept (Figure 4-8), as shown in Figure 6-40:

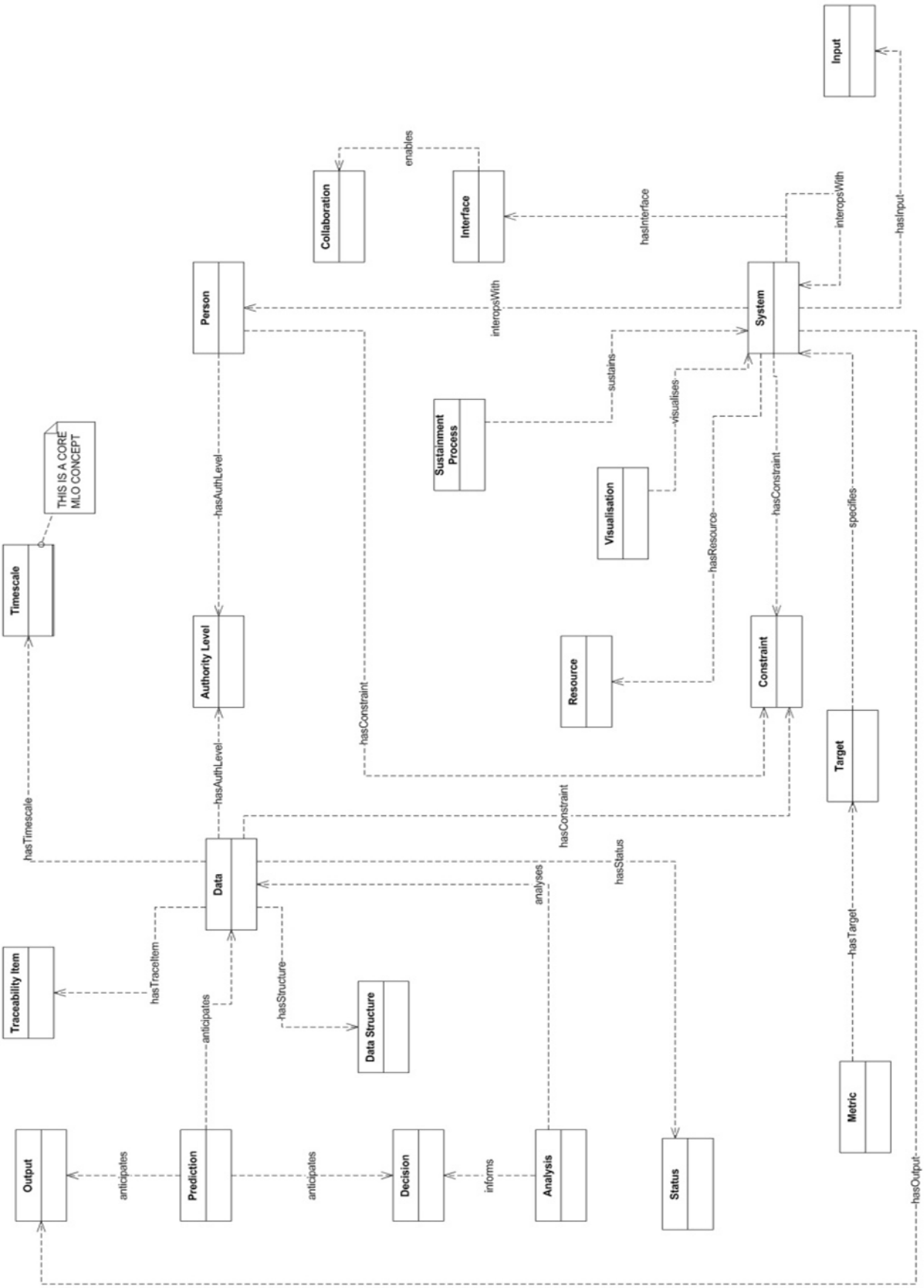


Figure 6-40 - The ontology core concepts UML diagram

A key finding of this stage of the work was the unconscious emergence of a standard systems model within the core ontology UML diagram shown in Figure 6-40. The development of the large number of terms and relationships was extremely complex and detailed, therefore it was not until a full review of the emerging results was carried out that the fact that the terms System, Input, Output, Resource and Constraint, were related in a conventional systems diagram format (see Section 6.3.2). This was taken as affirmation of the Core, Level 1 and Level 2 approach described in Figure 6-28, as through the development of a very large and complex Manufacturing Systems/ Manufacturing Intelligence Systems term pool (which would reside at Level 1) the generic system model emerged as a central part of the core concept model: this is consistent with the ontology domain (see Figure 6-40).

This phase of the research resulted in the large term pool taxonomy for Manufacturing Systems (Level 1 of the solution ontology), the core term model including core relationships within this pool, relationships that relate to Level 1 terms only and Level 1 and core terms, and a set of logic constraining the terms in the context of the requirements for interoperability.

7 Formalising the ontology

7.1 Introduction

This chapter details the work to create a formalised ontology and logic solution in support of the solution concept described in Section 4.2.2 in response to research question 3 (Section 4.3). The concepts, relationships and logic identified through the research explained in Chapter 6 were formalised using the IODE tool described in Section 2.7.4.

7.2 Solution design

Figure 7-1 shows a representation of the formalised multi-level ontology: The core and Level 1 ontology logic was built upon the declared core and Level 1 terms and relationships. This logic was described as ‘structural’ as it was explicitly coded through constraining logic or integrity constraints and inference rules which allow the inference of new relationships based on predefined logic. This structural logic in the Core and Level 1 ontologies forms the meta knowledge that as described in the unified approach to interoperability (Usman 2012). This logic can span both the core and Level 1 ontology levels, requiring the Core and Level 1 semantics to be declared prior to logic declaration; hence the logic is shown as being built on the semantic levels. The Level 2 ontology is declared using the full set of Core and Level 1 semantics and logic hence is shown at the top of the ontology structure. The detailed development and testing of the solution is shown in Section 7.4.

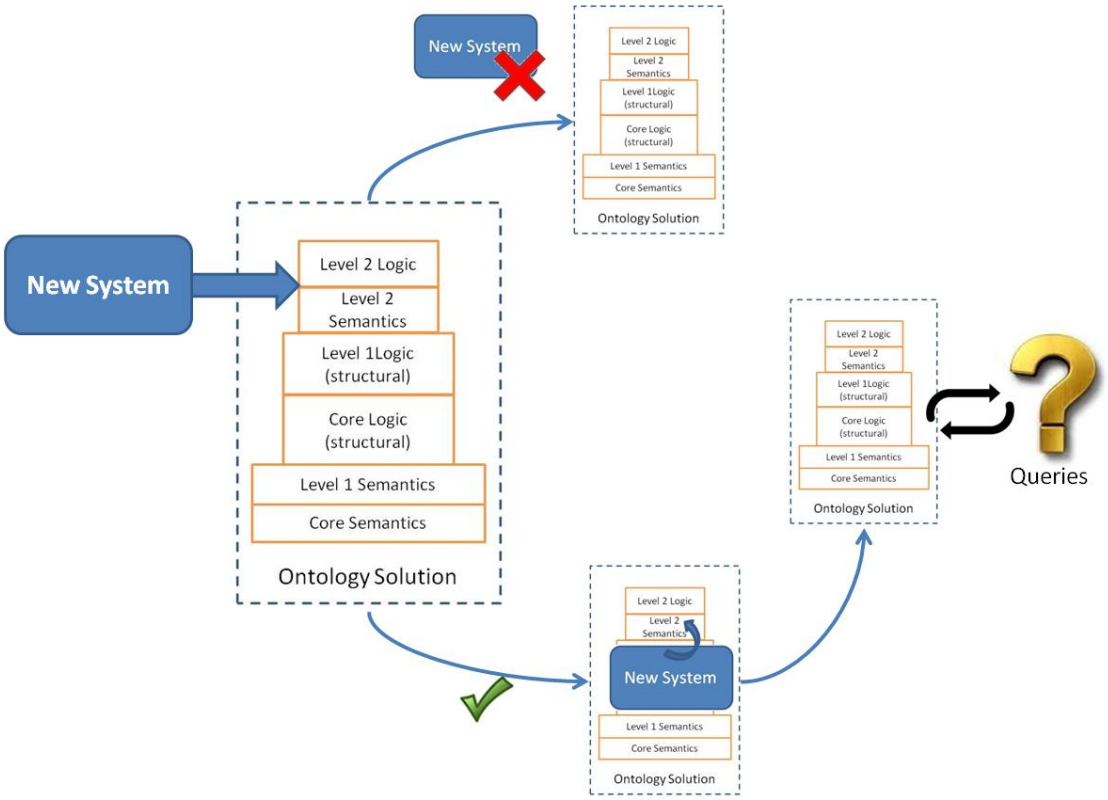


Figure 7-1 - The formalised solution ontology

Figure 7-1 shows how the solution functions:

- A new system is declared into the knowledge base (Level 2 ontology).
- If the new system is rejected by any of the structural logic i.e. it does not meet the requirements for interoperability as defined within the ontology solution logic, the declaration is rolled back and the system is rejected from the knowledge base with corresponding error message feedback.
- If the new system meets the requirements for interoperability it is incorporated into the knowledge base and becomes available for queries; the formalised ontology in itself is only a part of the solution as it only answers some of the ontology competency questions set out in Section 4.2.1. Formalised queries were required to fully answer the competency questions not answered by the function of the structural logic.

Section 7.4.6 lists the ontology competency questions, how the solution answers them, and the code for any queries.

7.3 Formalising the logical rules

The rules defined in Chapter 6 were rewritten using a more rigorous common logic format. Where possible the statements were written in the 'if', 'then' format to aid the process of coding them in the Knowledge Framework Language (KFL). It was found that logic statements not written in this format often required several logical statements to codify them, whereas those defined using this format were usually defined in a single code statement. This proved important at the testing stage, as it was difficult to keep track of which code statements affected which logical rules where there was not a one to one correlation. It was also found that the rules written in this format were easier to codify as it clearly structures the object and conditional statements in a manner suitable for coding.

At this early stage it was apparent that the syntactic, semantic and logical rigor that was required when coding the heavyweight ontology, while significantly increasing the complexity, would enable the solution to discern acceptable and non acceptable conditions in the knowledge base if the correct rules were formalised within it such as the example given in Section 7.4.1. Without this complexity, the solution could not resolve the issue of potential ontological ambiguity.

7.3.1 Integrity constraint and inference rule evolution

A list of the Inference Rule (IR) and Integrity Constraint (IC) logic for the 3 levels of the ontology was developed with examples being:

- If a system input is not approved the system output is not approved (IR)
- A system type must have some form of inputs, outputs, resource and constraints defined (IC)

A full list of the logic statements is available in Appendix B.

It can be seen that there are relatively few rules at the Core level, which is due to the limited number of terms available. There are also few rules for the Level 2 ontology as this level is more about providing the specialised terms for the MI domain and instances. The majority of the logic is created in the Level 1 ontology due to it having both the Level 1 and Core ontology terms and relationships to consider, and is not the instance level ontology (unlike Level 2). The different level of logic required at the ontology levels was consistent with the solution concept:

- A core concept ontology should have relatively low numbers of foundational concepts or terms, meaning the opportunity for logical rule is limited as this could result in the

ontology becoming overly specific or over-constrained and so limiting its applicability. This would be a key failing for a 'core' or foundational ontology.

- Domain ontologies, as shown in Figure 4-8, would be expected to contain a large number of domain specific concepts and terms, meaning the opportunity and requirement for constraining logic are much greater.
- The 'Level 2' ontology defined in this research, which is effectively the knowledge base, would be expected to contain a large and potentially rapidly growing number of terms, but these would be 'instances' of the terms defined in the higher level ontologies. As such they would be constrained by the structural logic of the ontologies on which is it built (i.e. IR, ICs and relationships) but would generally be at an inappropriately high level of specificity to define broadly applicable constraining logic hence little of no structural logic would be expected.

The logic statement numbers that are missing in Appendix B are those which were proposed but discounted prior to coding. The logic statements that are numbered with letter suffixes are those that have been broken down into a number of simple statements: this was due to the need to declare the statement at the right ontological level i.e. the right level of specificity, requiring logic to be declared explicitly at core concept (core ontology), Domain concept (domain ontology) or instance (knowledge base) level or at multiple levels. The statements that have been struck through with a statement in red have been removed or discounted through the ontology development process for the reason stated. The listing does not show that many of the statements that were initially written at one level were moved to another depending on the developing view of the level of generalisation of the logic .

There were a large number of changes made to the logic statements during the ontology development, excluding syntax errors and corrections. The key reasons for modifications can be summarised as:

- Logic removed/ modified as it was incorrectly stated at either the type or instance level (i.e. the wrong ontology level/ level of generalisation). This was the most common issue.
- Logic removed completely as it is not relevant to the core purpose of the solution i.e. is not related to system interoperability.
- Logic removed completely as the statement was disproved through the testing stage.
- Logic removed completely as it duplicates another statement.
- IC removed as the application of the rule has been automated through the use of an inference rule (IR).

Further issues were identified during the testing stages however, these are discussed throughout Chapter 8.

7.4 Coding and testing approach

7.4.1 Codify the core ontology

The core terms and relationships were declared in KFL forming the Core Semantics and common logic Integrity Constraints and Inference Rules used to accurately constrain the ontology in line with the UML model. This level of logic is referred to a ‘structural’ as it is required within the IODE tool to represent the relationships. Figure 7-2 represents this by showing the core logic being built in the ‘foundation’ of the Core semantics.

The following code shows examples of this activity. The first section of code declares something called ‘Data’ which is an subtype of ‘Object’ and an instance of a Type i.e. it is a superclass which will have subclasses. This is followed by a lightweight descriptive statement. For further detail see Section 2.7.4.

:Prop Data

:Inst Type

:sup Object

:name "Data"

:rem "Data is the lowest level of abstraction of information or knowledge and are numbers, characters or images that require a context to have meaning, at which point it becomes information."

The code shown below declares a relationship called ‘hasStructure’ which is a BinaryRelationship i.e. it relates two items, so if two items are not declared the relationship constraints will be invalidated. The relationship signature states that the two items joined by this relationship must be an instance or subtype of Data or DataStructure for the relationship to be valid.

:Rel hasStructure

:Inst BinaryRel

:Sig Data DataStructure

:name "has Structure"

The following code is an example of an Integrety Constraint. The IC states that if a thing called ‘t1’ and a second thing ‘t2’ interoperate, then an interface ‘i1’ will exist and be shared by both things.

(=> (and (interopsWith ?t1 ?t2)

(hasInterface ?t1 ?i1)

(hasInterface ?t2 ?i2))

(exists (?i)

(and (Interface ?i)

(hasInterface ?t1 ?i)

(hasInterface ?t2 ?i))))

:IC hard "If two arguments <code>?t1</code> <code>?t2</code> interoperate with each other they must have some associated interface <code>?i</code> in common."

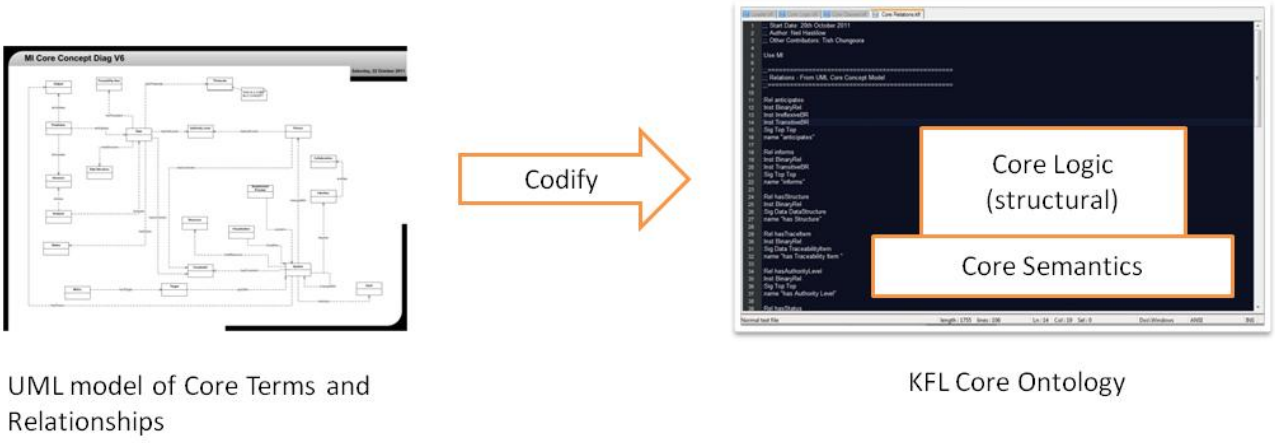


Figure 7-2 - Codifying the core ontology

Figure 7-3 shows the core ontology terms loaded into the IODE taxonomy. It can be seen that the MI context in which the terms are declared (shown as a MI. prefix) is built upon the IODE medium level ontology (MLO). The MLO term ‘Duration’ is highlighted in the figure and it can be seen that it is declared at the same level as the MI terms ‘Status’ and ‘AuthorityLevel’. There was a conscious effort to re-use MLO terms to avoid duplication of terms within the IODE tool, which could introduce semantic ambiguity.

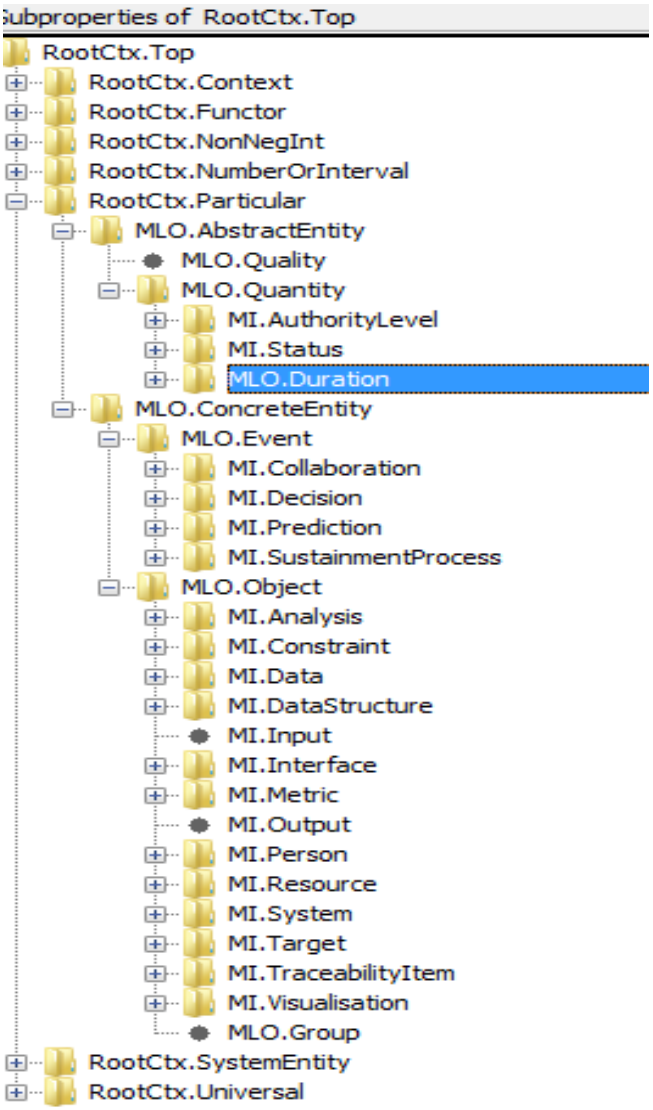


Figure 7-3 - IODE screenshot showing the Core Ontology terms

7.4.2 Test the core ontology

The test data set used with the core ontology comprised facts and types specifically designed to test the logic functions as expected i.e. accepting compliant data and triggering the logic to flag or reject non compliant data. The data test comprised the initial domain concepts (which would be developed further before being finally compiled into the Level 1 or domain ontology as listed in Appendix E). This test was relatively simple due to the limited number of concepts and relationships, involving the declaration of one sub type to each core term and is represented in Figure 7-4.

The Core Semantics and Logic were used to validate the IODE tool itself: due to its limited size it was possible to test all of the terms to ensure the tool itself behaves as expected.

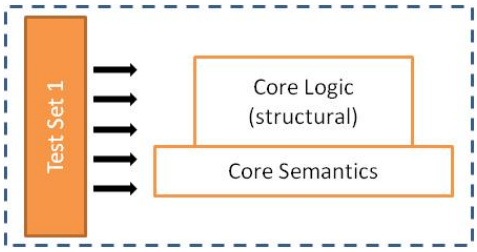


Figure 7-4 - Testing the core ontology

7.4.3 Codify the level 1 ontology

The Level 1 UML diagram was codified into ontology within the IODE tool in a similar manner to the Core ontology (see Section 7.4.1). This ontology is built on top of the Core ontology (semantics and logic) and is loaded incrementally to it as shown in Figure 7-5 i.e. without the Core ontology the Level 1 ontology is invalid.

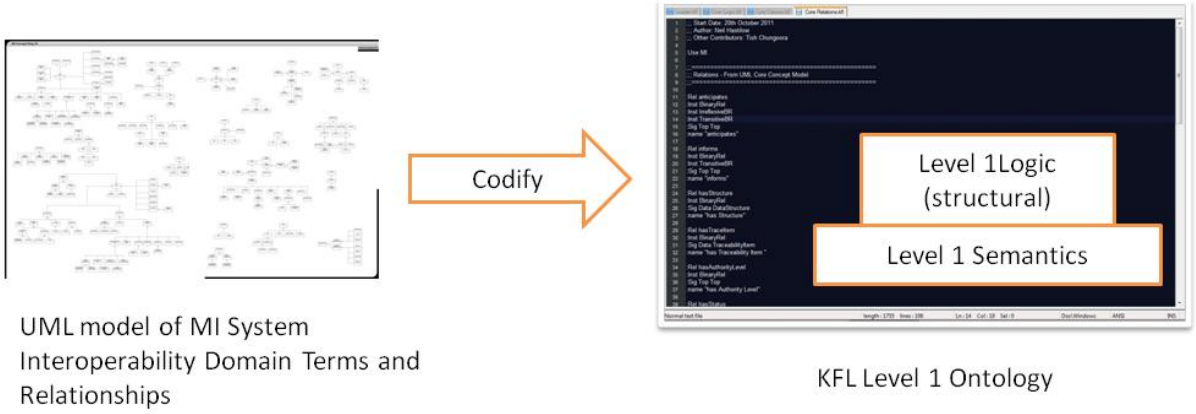


Figure 7-5 - Codifying the Level 1 ontology

Figure 7-6 shows the Level 1 term ‘OpAuthorityLevel’ and sub terms declared into the ontology, along with the lightweight ontology description. That these terms loaded showed that they met the structural logic requirements i.e. they met the defined constraints for something that is an operation authority level. Further testing was required to ensure that the terms were appropriately constrained by declaring instances to the ontology.

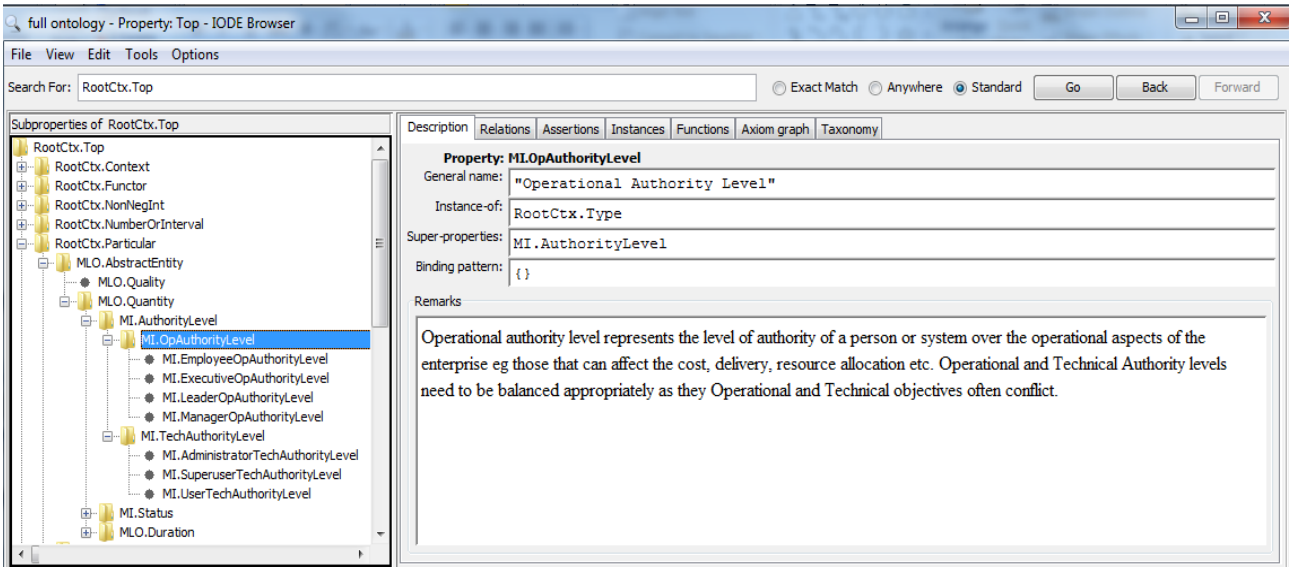


Figure 7-6 - IODE screenshot showing Level 1 ontology terms

7.4.4 Test the level 1 ontology

A second test data set, which is shown in Section 8.3.1 was created to test the combined Level 1 and Core Ontology (as previously described and in line with the solution concept described in Figure 4-7, to be valid the Level 1 ontology must be built on (i.e. include) the Core Ontology). Although this data set could not be fully comprehensive due to the number of terms, relationships and logical statements, a representative sample of facts and queries designed to give a thorough trial of the ontology was created.

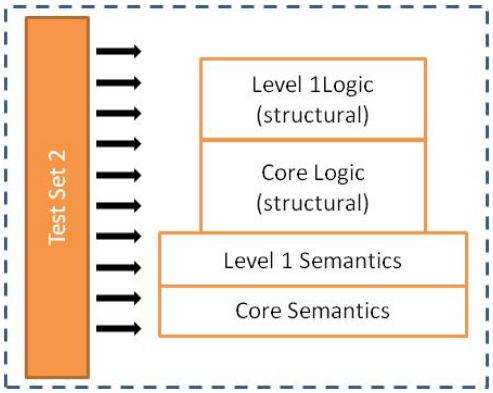


Figure 7-7 - Testing the Level 1 ontology

7.4.5 Develop the level 2 ontology

The method of developing the Level 2 (MI systems domain) ontology was different from Level 1: it involved the use of a test data set using a subset of the production data that would ultimately be used to populate the Level 2 ontology fully for the solution trials to fully declare a theoretical system. Manual interrogation of the Level 1, Core and developing Level 2 ontology focusing on the ontology development themes and competency questions was

used to further develop the Level 2 ontology. Figure 7-8 represents how the thought processes and lines of enquiry used to answer the competency questions were used to develop the Level 2 logic such as “If the output of one system is the input of another system the systems interoperate” (the full logic listing can be seen in Appendix B).

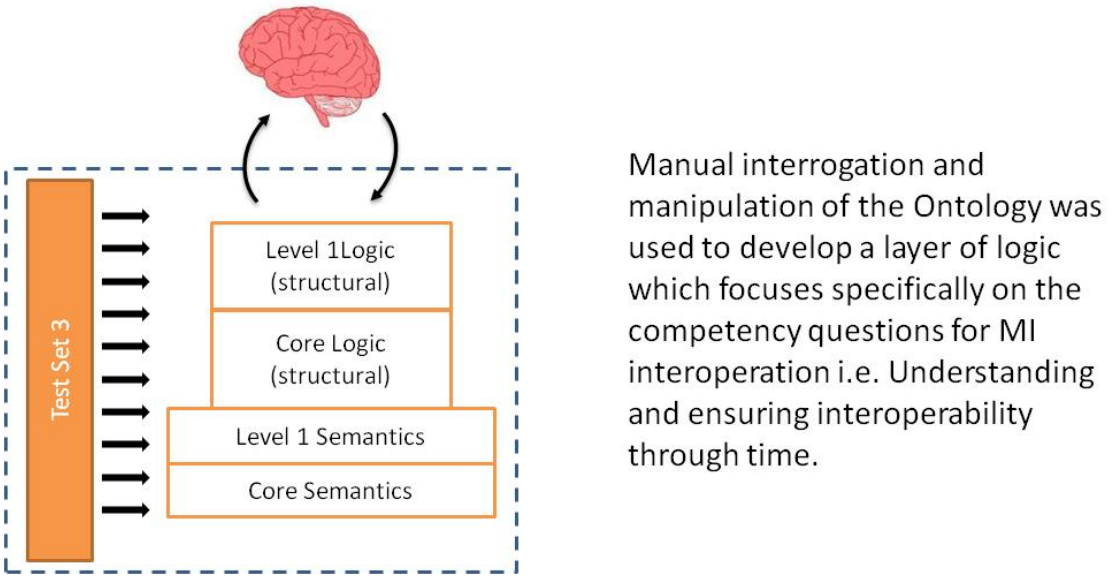


Figure 7-8 - Developing the Level 2 ontology using human interaction

7.4.6 Codify the level 2 ontology

The final stage of this development was the full population of the ontology to cover a specific set of BPMNs covering the 1st cell in the facility that was mapped, as described in Section 6.3.1, and to codify the manually developed queries required to answer the competency questions as representated in Figure 7-9.

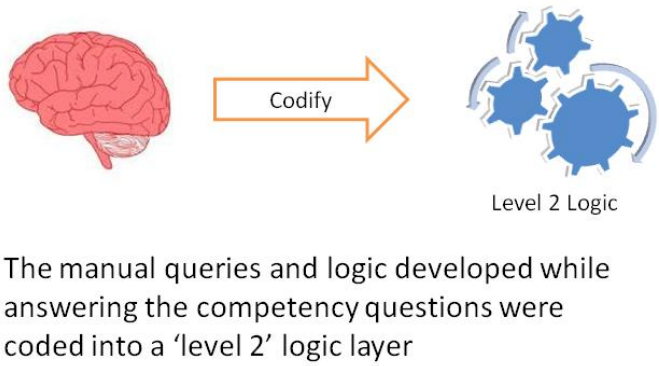


Figure 7-9 - Codifying the Level 2 ontology facts, logic and queries

The detail of this activity and the other testing activity is described in Chapter 8.

The following ECLIF queries were developed by focusing on the competency questions. These queries represent the code that was needed to interact with the ontology system and fact base to fully answer the onotology competency questions that were defined in Section 4.2.1. which were in turn aligned to the research hypothesis in Section 1.2.4.

Competency Question: Is the system allowed to interoperate with another system?

Answer: Partially answered by structural logic i.e. if it loads into the fact base and interoperates it is valid, but also requires the use of one of the following queries depending on the required level of specificity of the answer:

(interopsWith ?x ?y) – What entities interoperate?

(and(interopsWith ?x ?y) (system ?x)) – What entities do systems interoperate with?

(and(interopsWith ?x ?y) (system ?x)(system ?y)) – What systems interoperate?

(not(interopsWith ?x ?y) – What systems do not interoperate

Competency Question: What other systems are involved?

Answer: Inferring related systems by function and description (excluding those already declared as interoperating as they are covered by the previous question):

(and (System ?x)(System ?y)(hasInput ?x ?i)(hasOutput ?x ?o)(hasResource ?x ?r)(hasConstraint ?x ?c) (hasInput ?y ?i)(hasOutput ?y ?o)(hasResource ?y ?r)(hasConstraint ?y ?c)) – one of a number of ways of identifying systems with the same inputs, outputs, resources and constraints.

(and (hasInput ?x ?i)(hasInput ?y ?i)) – Identify entities with the same input.

(and (hasOutput ?x ?o)(hasOutput ?y ?o)) – Identify entities with the same output.

(and (hasResource ?x ?r)(hasResource ?y ?r)) – Identify entities with the same resource.

(and (hasConstraint ?x ?c)(hasConstraint ?y ?c)) – Identify entities with the same constraint.

(mayAccess ?x ?y) – Identify entities which may access each other.

(and(System ?x)(System ?y)(mayAccess ?x ?y)) – Identify systems which may access each other.

(and(System ?x)(System ?y)(analyses ?x ?y)) – Identify systems which related by an analysis process.

Competency Question: What data or information does the system create/ use?

Answer: List inputs, outputs, resources, constraints for a system.

(and(System ?x)(or(hasInput ?x ?i)(hasOutput ?x ?o)(hasResource ?x ?r)(hasConstraint ?x ?c)))

Competency Question: Does this system conform to my current understanding of interoperability requirements?

Answer: Functionally answered by the structural logic: if the system being added to the knowledge based does not meet the requirements for interoperability the logic will reject the declaration (see Section 7.2).

Competency Question: Is the system adequately specified or understood?

Answer: Functionally answered by the structural logic: the right level of specification is not available the solution will reject the system declaration. (see Section 7.2).

This chapter has described the creation of the formalised ontology solution, including the logic and queries that were believed to be required to prove the research hypothesis stated in Section 1.2.4. Chapter 8 goes on to detail the testing approach and results that were used to validate this solution and ultimately the research hypothesis.

8 Testing and experimental results

8.1 Introduction

Initial testing was carried out on each level of the ontology solution as they were coded, as described in Chapter 7. This testing verified the software and coding had been carried out correctly prior to ontology being fully populated as opposed to the following testing which verified the full ontology solutions ability to meet the function proposed by this research. The full system testing was carried out through 2 rounds of experimental testing, followed by the research experiment. The experiments each served the following functions, towards the aim of answering research question 4 (Section 4.3):

- Experiment 1 (Figure 8-1): Validate the ability of the ontology solution to function as expected using test data with known issues (error trapping). The queries were run against the database to ensure that the ICs, IRs and relationships were working correctly. Other manual queries were also run against the system to ensure the facts had loaded and the logic was working correctly
- Experiment 2 (Figure 8-2): Validate the ability of the ontology solution to correctly represent 4 of the real world systems based on the information from their BPMNs maps as described in Section 6.3.2
- Experiment 3 (Figure 8-2): Validate the ability of the ontology solution to represent a system from another timescale and identify the interoperability issues. Due to the impracticality of using a system from a future timescale (it would be possible to declare an emerging system, but by definition this would have no proven experience to benchmark against), a legacy system was used which has known interoperability issues. The success of the solution was judged against its ability to identify these known issues.

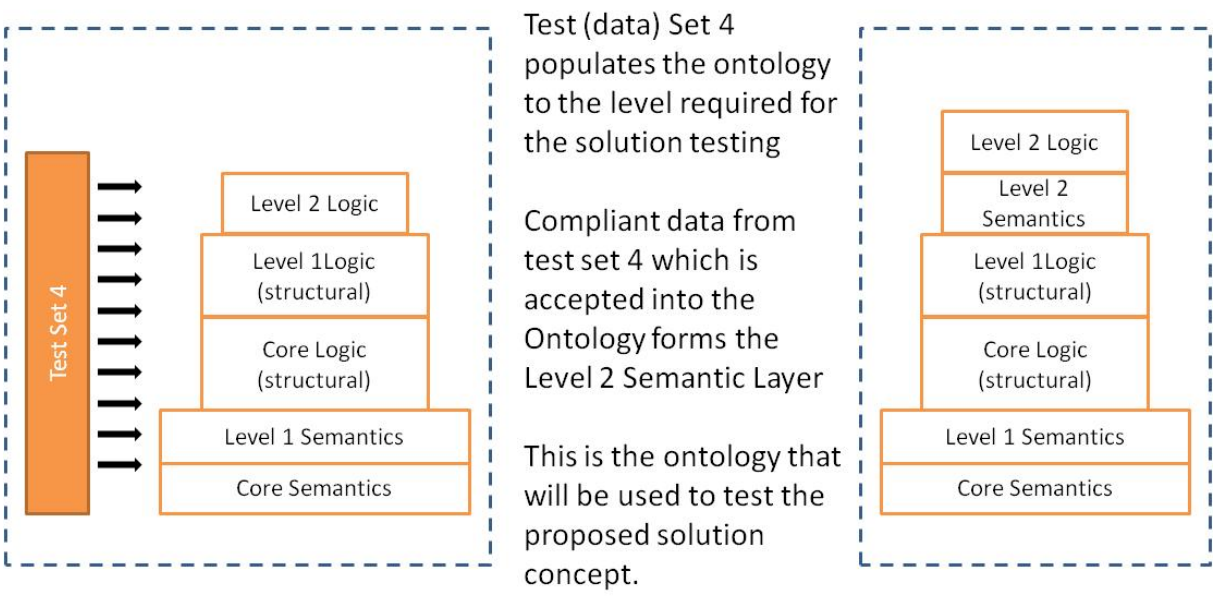


Figure 8-1 - Full ontology testing experiment 1

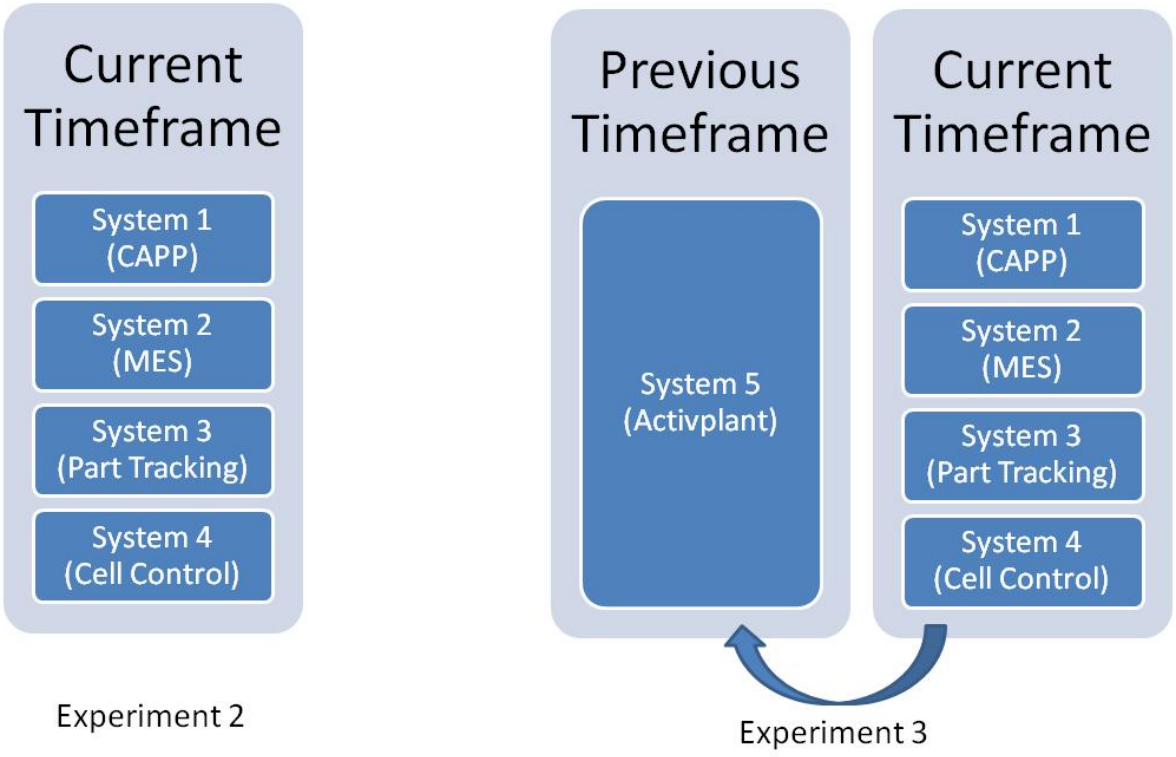


Figure 8-2 - Full ontology testing experiments 2 and 3

8.2 Core ontology testing results

The Core Ontology testing highlighted basic inconsistencies in type/ relationship declarations. Due to the small number of terms and relationships and logic in the Core ontology the testing was straightforward and demonstrated that the IODE tool was functioning as expected at a basic level constraining the declaration of concepts in line with the ontology logic.

8.3 Level 1 ontology testing results

The purpose of the testing at this stage was to ensure the ontology and logic behaved as expected, however a more in depth understanding of the IODE tool and the ontology also resulted. Some of the understanding generated through the testing phase can be seen in Appendix A.

The testing of the Level 1 Ontology was far more complex than the Core Ontology due to the much larger number of terms, and the potentialy complex interaction of the logic. It became very difficult to anticipate the interactions of the IC and IR logic as multiple relationships could be inferred for a single term any of which could trigger IC logic.

The findings of this testing stage could be catogrised into the topics of:

- Ontology: Increased understanding of the terms and relationships (in particular the structure of the terms and their level of generalization)
- Logic: Increased understanding of the logic
- Tool: Increased understanding of the IODE tool.

The increased understanding generated in these 3 areas helped clarify how the ontology solution concept could be formalised and validated, with examples being:

- With the emergence of a systems model in the core terms which is similar to the IDEF0 model (see Section 6.5), the 'system' prefixes and suffixes were removed from a large number of the ontology terms as it was considered unnecessary as the systemic nature is implicit. This was far more significant than a semantic or nomenclature clarification: the systemic nature of the entities was formalised within the ontology structure removing the need for manual interpretation of the term name as the entities were logically constrained such that they must have inputs, outputs, resources and constraints defined to be valid.
- Defining constraining logic at the right level of specificity or ontology level is crucial to the solution concept succeeding e.g. relationships may be used between multiple

classes meaning the relationship signatures have to be defined at the highest common level and then logic used to constrain the relationship back down to an appropriate level.

- Binary relationships were used in preference of uniary relations to maximize the logic capability and ontology flexibility. While uniary relationships for concepts such as ‘object status’ were possible, where status could be applied as a 'tag' to an object such as 'deleted' or 'obsolete', binary relations enabled the use of more complex logic to control the change of status and more flexible development of potential statuses by using a Binary relationship 'hasStatus' and creating status objects.
- The level of term abstraction referenced in ontology logic should be used to define at what ontological level the logic resides to aid logic development and control: logic that only uses the types in the core ontology should be considered the primary core logic, the logic that references domain Level 1 types will be the next level and instance logic the most specialised logic. Logic that refers to both types and instances may have to be split to allow this tiering to take place as per previous points.
- Differences in the performance of IODE v3.4 and v4.2.1 (when using supTC declarations) during early testing proved the importance of testing that the tool is performing as expected prior to testing a specific application of that tool and that using the most up to date version of the tools is crucial.
- Validation of the IODE tool highlighted that the context for a term (in KFL this is its prefix) should always be explicitly coded as the inference of context by IODE, while possible is not always accurate meaning IODE may recognize a single term as having two different meanings which undermines the solutions concepts structure logic.
- The structural logic is an important tool in the ontology development i.e. it can be considered both the ‘means’ and the ‘end’ e.g. Integrity constraint 37 used the relationship ‘hasConstraint’ for ‘Specifications’ which caused the logic to flag that ‘Specifications’ have been declared as a ‘Target’ rather than a ‘Constraint’. This raised the question of whether a specification is a target or a constraint or whether a target is a constraint. It can be expected that this type of learning will continue as the solution is applied, however how this will affect ongoing maintenance of this solution concept is unclear.

8.3.1 Example testing iteration

This section describes one iteration of the logic testing. The same approach was used many times identifying and resolving issues in different areas of the ontology:

A set of ECLIF assertions were made to add a system instance called ‘TestSystem1’ to the ontology. It is important to note that because they are instances they are effectively partial populations of the Level 2 ontology as defined in Section 4.2.2. These assertions were deliberately designed to contravene the Level 1 and core ontology logic to test its ability to govern additions. The initial test was to simply declare an instance using the code:

(System TestSystem1)

This was correctly rejected by IODE for breaching many logic integrity constraints (ICS) or rules. The next, far more complete input is shown below and was correctly rejected by the system due to lack of functional or non functional specification being declared:

(System TestSystem1)
(Input TestInput1)
(Output TestOutput1)
(Resource TestResource1)
(Constraint TestConstraint1)
(hasResource TestSystem1 TestResource1)
(hasConstraint TestSystem1 TestConstraint1)
(hasSecurityCategory TestSystem1 Public)
(hasInput TestSystem1 TestInput1)
(hasOutput TestSystem1 TestOutput1)
(hasSecurityCategory TestInput1 Public)
(hasSecurityCategory TestOutput1 Public)

This issue was resolved and a deliberate error added: declaring a resource as being a constraint (see below), which was correctly identified by the system and rejected with suitable feedback (the error is underlined):

(System TestSystem1)
(Input TestInput1)
(Output TestOutput1)
(Resource TestResource1)
(Constraint TestConstraint1)
(FunctionalSpecification FuncSpec1)
(NonFunctionalSpecification NonFuncSpec1)
(hasConstraint TestSystem1 FuncSpec1)
(hasConstraint TestSystem1 NonFuncSpec1)
(hasResource TestSystem1 TestResource1)
(hasConstraint TestSystem1 TestResource1)
(hasSecurityCategory TestSystem1 Public)
(hasInput TestSystem1 TestInput1)
(hasOutput TestSystem1 TestOutput1)
(hasSecurityCategory TestInput1 Public)
(hasSecurityCategory TestOutput1 Public)

This error was corrected and the code resubmitted, IODE then identified the lack of functional specification description of metrics and targets prompting the following code additions:

(System TestSystem1)
(Input TestInput1)
(Output TestOutput1)
(Resource TestResource1)
(Constraint TestConstraint1)
(FunctionalSpecification FuncSpec1)
(NonFunctionalSpecification NonFuncSpec1)
(Metric TestMetric1)
(MetricTarget TestMetricTarget1)
(specifies FuncSpec1 TestMetric1)
(specifies FuncSpec1 TestMetricTarget1)
(hasConstraint TestSystem1 FuncSpec1)
(hasConstraint TestSystem1 NonFuncSpec1)
(hasResource TestSystem1 TestResource1)
(hasConstraint TestSystem1 TestConstraint1)
(hasSecurityCategory TestSystem1 Public)
(hasInput TestSystem1 TestInput1)
(hasOutput TestSystem1 TestOutput1)
(hasSecurityCategory TestInput1 Public)
(hasSecurityCategory TestOutput1 Public)

This input failed as the metric does not target a metric target and the metric target does not have a validity timescale (the latter being only a warning as it is a soft IC).

The following code therefore loaded acceptable into the IODE system (with the soft IC warning)

(System TestSystem1)
(Input TestInput1)
(Output TestOutput1)
(Resource TestResource1)
(Constraint TestConstraint1)
(FunctionalSpecification FuncSpec1)
(NonFunctionalSpecification NonFuncSpec1)
(Metric TestMetric1)
(MetricTarget TestMetricTarget1)
(hasTarget TestMetric1 TestMetricTarget1)
(specifies FuncSpec1 TestMetric1)
(specifies FuncSpec1 TestMetricTarget1)
(hasConstraint TestSystem1 FuncSpec1)
(hasConstraint TestSystem1 NonFuncSpec1)

(hasResource TestSystem1 TestResource1)
(hasConstraint TestSystem1 TestConstraint1)
(hasSecurityCategory TestSystem1 Public)
(hasInput TestSystem1 TestInput1)
(hasOutput TestSystem1 TestOutput1)
(hasSecurityCategory TestInput1 Public)
(hasSecurityCategory TestOutput1 Public)

This acceptable input was then modified to deliberately cause IC61 ('a system cannot input or output data with a higher security category than the system') to reject the input on due to the security categories conflicting (see below). It therefore was a matter of concern that the following invalid code loaded acceptably (without the logic identifying the issue).

(System TestSystem1)
(Input TestInput1)
(Output TestOutput1)
(Resource TestResource1)
(Constraint TestConstraint1)
(FunctionalSpecification FuncSpec1)
(NonFunctionalSpecification NonFuncSpec1)
(Metric TestMetric1)
(MetricTarget TestMetricTarget1)
(hasTarget TestMetric1 TestMetricTarget1)
(specifies FuncSpec1 TestMetric1)
(specifies FuncSpec1 TestMetricTarget1)
(hasConstraint TestSystem1 FuncSpec1)
(hasConstraint TestSystem1 NonFuncSpec1)
(hasResource TestSystem1 TestResource1)
(hasConstraint TestSystem1 TestConstraint1)
(hasSecurityCategory TestSystem1 Public)
(hasInput TestSystem1 TestInput1)
(hasOutput TestSystem1 TestOutput1)
(hasSecurityCategory TestInput1 Public)
(hasSecurityCategory TestOutput1 Secret)

Queries were carried out against the ontology to identify whether TestSystem1 had the 'Secret' security category applied as per the assertion. The results showed that it had not. By specifically asserting: (hasSecurityCategory TestSystem1 MI.Public) following the loading of the erroneous code, the logic 'fired' and rejected the assertion due to IC61. It was noted the asserting (hasSecurityCategory TestSystem1 Public) did not cause the logic to fire implying there is a system issue resulting in IODE confusing types and instances? By explicitly declaring the MI context this was resolved and the logic shown to fire (rejecting the TestSystem1 instance). Subsequently by changing TestOutput1 to a security category of Public resolved the issue and it loaded correctly in line with expectations:

(System TestSystem1)
(Input TestInput1)
(Output TestOutput1)
(Resource TestResource1)
(Constraint TestConstraint1)
(FunctionalSpecification FuncSpec1)
(NonFunctionalSpecification NonFuncSpec1)
(Metric TestMetric1)
(MetricTarget TestMetricTarget1)
(hasTarget TestMetric1 TestMetricTarget1)
(specifies FuncSpec1 TestMetric1)
(specifies FuncSpec1 TestMetricTarget1)
(hasConstraint TestSystem1 FuncSpec1)
(hasConstraint TestSystem1 NonFuncSpec1)
(hasResource TestSystem1 TestResource1)
(hasConstraint TestSystem1 TestConstraint1)
(hasSecurityCategory TestSystem1 MI.Public)
(hasInput TestSystem1 TestInput1)
(hasOutput TestSystem1 TestOutput1)
(hasSecurityCategory TestInput1 MI.Public)
(hasSecurityCategory TestOutput1 MI.Secret)

8.3.2 Findings

This testing activity proved that following the feedback of the testing findings into the solution design the logic developed at this stage seems to be working as intended, constraining the definition of systems to ensure suitable levels of information are provided and that the content of that information does not contradict the system constraints which have been constructed in line with interoperability requirements.

This work identified a potentially significant issue with the automatic context inference ability of the latest versions of IODE (previous version have required the context to be explicitly declared) this means this functionality cannot be treated as reliable and contexts should be explicitly declared to avoid this bug in the ongoing development activities.

An additional outcome from this research as described in Section 8.3.1 was a strong indication that the heavyweight ontological approach is effective at guiding a user to define a system in a desired way and rejecting non compliant submissions to the ontology. This indicated that subject to the logic being robust and appropriately focused on the desired outcome this approach could be used to ensure a system was defined in an appropriate way to maximise the interoperability of the system which validates the concept shown in Figure 4-7, describing a multi level, (heavyweight) ontology defined within a single timeframe.

8.4 Full ontology testing results – Experiment 1

8.4.1 Initial fact declarations

The following facts were taken directly from the BPMN map for the first cell in the facility which used as the subject of the research as described in Sections 6.3.1 and 6.3.2. The facts attempt to describe a Computer Aided Process Planning (CAPP) system.

(Operator wax_injection_operator)
(Supervisor wax_production_leader)
(Supervisor foundry_manufacturing_engineering_manager)
(Supervisor foundry_manufacturing_manager)
(ManufacturingEngineer wax_process_engineer)
(ManufacturingEngineer casting_part_owner)
(MaintenenceEngineer foundry_maintenence_engineer)
(MaterialController foundry_material_requirements_planning_controller)
(Informs wax_injection_operator wax_production_leader)
(System CAPP_system)
(Input core_specification)
(Output core_specification)
(Output core_chaplet_inspection_spec)
(Output wax_process_instruction)
(Output wax_process_parameter)

8.4.2 Fact listing updated using MI ontology

The BPMN facts were rejected by the ontology system as they did not declare the required level of detail about the system. This was not a failure of the BPMN tool, but instead represented the incomplete information or knowledge that had been initially presented. As such this was viewed as the solution prompting that more information was required. Using the system experts and the prompts from the ontology systems (IODE) the fact listing was completed to allow the solution to accept the facts for this system to be accepted into the fact or knowledge base. It can be seen when comparing the facts listed in Sections 8.4.1 and 8.4.2, that the required level of data to fully declare the CAPP system to the fact base is significantly higher than was captured, even in a high quality BPMN diagram:

(Operator wax_injection_operator)
(Supervisor wax_production_leader)
(Supervisor foundry_manufacturing_engineering_manager)
(Supervisor foundry_manufacturing_manager)
(ManufacturingEngineer wax_process_engineer)
(ManufacturingEngineer casting_part_owner)
(MaintenanceEngineer foundry_maintenence_engineer)
(MaterialController foundry_material_requirements_planning_controller)
(informs wax_injection_operator wax_production_leader)
(StandardDataStructure XML)
(ValidityTimescale life_of_type)

(ValidityTimescale life_of_order)
(System CAPP_system)
(Input MI.core_specification)
(ProductSpecification MI.core_specification)
(Output MI.core_specification)
(Output MI.core_chaplet_inspection_spec)
(VisualSpecification MI.core_chaplet_inspection_spec)
(Output MI.wax_process_instruction)
(Output MI.wax_process_parameter)
(ProcessSpecification MI.wax_process_instruction)
(ProcessSpecification MI.wax_process_parameter)
(ProductSpecification MI.product_definition)
(Resource MI.PLM_server)
(Resource MI.PLPS_process)
(Resource MI.it_infrastructure)
(Resource MI.ERP_system)
(Resource MI.PLM_database)
(Resource MI.production_schedule)
(Resource MI.manufacturing_bill_of_material)
(Resource MI.bill_of_plant)
(Resource MI.geometry_models)
(Resource MI.bill_of_process)
(Resource MI.casting_part_owner)
(Resource MI.wax_process_engineer)
(Resource MI.foundry_manufacturing_engineering_manager)
(Constraint MI.PLPS_process)
(Constraint MI.data_storage_capacity)
(Constraint MI.RR_quality_management_system)
(Constraint MI.data_transfer_capacity)
(Constraint MI.data_processing_capacity)
(Constraint MI.ExportControlCategory)
(Constraint MI.RetentionCategory)
(Constraint MI.SecurityCategory)
(Constraint MI.product_definition)
(Constraint MI.EASA_requirements)
(NonFunctionalSpecification MI.capp_performance_requirements_spec)
(FunctionalSpecification MI.capp_functional_specification)
(FunctionalSpecification MI.RR_quality_management_system)
(hasConstraint MI.CAPP_system MI.PLPS_process)
(hasConstraint MI.CAPP_system MI.data_storage_capacity)
(hasConstraint MI.CAPP_system MI.RR_quality_management_system)
(hasConstraint MI.CAPP_system MI.data_transfer_capacity)
(hasConstraint MI.CAPP_system MI.data_processing_capacity)
(hasConstraint MI.CAPP_system MI.ExportControlCategory)
(hasConstraint MI.CAPP_system MI.RetentionCategory)
(hasConstraint MI.CAPP_system MI.SecurityCategory)
(hasConstraint MI.CAPP_system MI.product_definition)
(hasConstraint MI.CAPP_system MI.EASA_requirements)

(hasResource MI.CAPP_system MI.PLM_server)
(hasResource MI.CAPP_system MI.PLPS_process)
(hasResource MI.CAPP_system MI.PLM_database)
(hasResource MI.CAPP_system MI.production_schedule)
(hasResource MI.CAPP_system MI.manufacturing_bill_of_material)
(hasResource MI.CAPP_system MI.bill_of_plant)
(hasResource MI.CAPP_system MI.geometry_models)
(hasResource MI.CAPP_system MI.bill_of_process)
(hasResource MI.CAPP_system MI.casting_part_owner)
(hasResource MI.CAPP_system MI.wax_process_engineer)
(hasResource MI.CAPP_system MI.foundry_manufacturing_engineering_manager)
(SystemPerformanceMetric MI.CAPP_system_response_time)
(Metric MI.CAPP_system_response_time)
(MetricTarget MI.CAPP_system_response_time_target)
(hasTarget MI.CAPP_system_response_time MI.CAPP_system_response_time_target)
(hasSecurityCategory MI.CAPP_system MI.Private)
(hasExportControlCategory MI.CAPP_system MI.ExportControlled)
(hasRetentionCategory MI.CAPP_system MI.MandatoryRetention)

8.4.3 Examples of resulting errors

The following messages are examples of some of the typical error messages that required the fact declarations for the CAPP system to be modified and how the fact declarations and logic were updated to resolve them:

Systems <code>CAPP_system</code> with a retention category of non mandatory retention can only input <code>MI.core_specification</code> or output <code>MI.core_specification</code> non mandatory retention data.

This error means the CAPP system has been declared as having a non mandatory data retention category but it inputs and outputs mandatory data. Once the retention category of the data and system were confirmed as being correct the logic was. It was concluded that the retention category of a system is only relevant if an input is not subsequently output (implying the system retains the input). This error also highlighted a key difference between inputs and resources: resources are used by a system but implicitly are never retained by the system therefore the retention logic does not apply.

The disjoint properties (RootCtx.disjointWith) RootCtx.Particular and RootCtx.Universal should not have any instances in common, but they have MI.ExportControlCategory.

(See the development log) This error is due to an ECLIF declaration of logic that already exists in the structural KFL logic. The IODE system was found to be very intolerant of

duplication of logic declarations in the structural (KFL) logic and fact declarations for instances.

A system <code>CAPP_system</code> must be described by a functional and non functional spec.

This was a typical, if misleading error: the actual error is not a lack of spec but a lack of metrics within that spec, this error displayed correctly once the duplication of logic error was resolved. IODE seems to have a flaw where one significant and valid error message can cause many other erroneous errors which can in turn mask the root cause error.

Target individuals have a validity timescale individual <code>MI.CAPP_system_response_time_target</code>.

This error was a simple matter of the CAPP system response time metric target not having a validity timescale declared (how long should this target be considered valid by default)

These examples show the complex nature of the development of the fact declarations and logic.

8.4.4 Fact listing corrected to allow loading

This listing represents the CAPP system fact listing that allowed the CAPP system declaration to be accepted into the fact base effectively confirming that it meets the ontology structural logic criteria for being suitable for interoperation:

(Operator wax_injection_operator)
(Supervisor wax_production_leader)
(Supervisor foundry_manufacturing_engineering_manager)
(Supervisor foundry_manufacturing_manager)
(ManufacturingEngineer wax_process_engineer)
(ManufacturingEngineer casting_part_owner)
(MaintenanceEngineer foundry_maintenence_engineer)
(MaterialController foundry_material_requirements_planning_controller)
(informs wax_injection_operator wax_production_leader)

(hasTechAuthorityLevel wax_injection_operator MI.User)
(hasTechAuthorityLevel wax_production_leader MI.User)
(hasTechAuthorityLevel foundry_manufacturing_engineering_manager MI.Administrator)
(hasTechAuthorityLevel foundry_manufacturing_manager MI.User)
(hasTechAuthorityLevel wax_process_engineer MI.Superuser)
(hasTechAuthorityLevel casting_part_owner MI.Superuser)

(hasTechAuthorityLevel foundry_maintenence_engineer MI.Superuser)
(hasTechAuthorityLevel foundry_material_requirements_planning_controller MI.User)
(hasOpAuthorityLevel wax_injection_operator MI.Employee)
(hasOpAuthorityLevel wax_production_leader MI.Leader)
(hasOpAuthorityLevel foundry_manufacturing_engineering_manager MI.Manager)
(hasOpAuthorityLevel foundry_manufacturing_manager MI.Manager)
(hasOpAuthorityLevel wax_process_engineer MI.Employee)
(hasOpAuthorityLevel casting_part_owner MI.Employee)
(hasOpAuthorityLevel foundry_maintenence_engineer MI.Employee)
(hasOpAuthorityLevel foundry_material_requirements_planning_controller MI.Employee)
(StandardDataStructure MI.XML)
(ValidityTimescale MI.life_of_type)
(ValidityTimescale MI.life_of_order)
(ValidityTimescale MI.life_of_contract)
(System MI.CAPP_system)
(Input MI.core_specification)
(Output MI.wax_process_instruction)
(Output MI.wax_process_parameter)
(Output MI.core_specification)
(Output MI.core_chaplet_inspection_spec)
(Resource MI.PLM_server)
(Resource MI.PLPS_process)
(Resource MI.it_infrastructure)
(Resource MI.ERP_system)
(Resource MI.PLM_database)
(Resource MI.production_schedule)
(Resource MI.manufacturing_bill_of_material)
(Resource MI.bill_of_plant)
(Resource MI.geometry_models)
(Resource MI.bill_of_process)
(Resource MI.casting_part_owner)
(Resource MI.wax_process_engineer)
(Resource MI.foundry_manufacturing_engineering_manager)
(Constraint MI.PLPS_process)
(Constraint MI.data_storage_capacity)
(Constraint MI.RR_quality_management_system)
(Constraint MI.data_transfer_capacity)
(Constraint MI.data_processing_capacity)
(Constraint MI.product_definition)
(Constraint MI.EASA_requirements)
(MI.FunctionalSpecification MI.capp_functional_specification)
(MI.FunctionalSpecification MI.RR_quality_management_system)
(MI.NonFunctionalSpecification MI.capp_performance_requirements_spec)
(Metric MI.CAPP_system_response_time)
(MetricTarget MI.CAPP_system_response_time_target)
(hasTarget MI.CAPP_system_response_time MI.CAPP_system_response_time_target)
(specifies MI.capp_functional_specification MI.CAPP_system_response_time)

(specifies MI.capp_functional_specification MI.CAPP_system_response_time_target)
(hasConstraint MI.CAPP_system MI.capp_performance_requirements_spec)
(hasConstraint MI.CAPP_system MI.capp_functional_specification)
(hasConstraint MI.CAPP_system MI.RR_quality_management_system)
(hasConstraint MI.CAPP_system MI.PLPS_process)
(hasConstraint MI.CAPP_system MI.data_storage_capacity)
(hasConstraint MI.CAPP_system MI.RR_quality_management_system)
(hasConstraint MI.CAPP_system MI.data_transfer_capacity)
(hasConstraint MI.CAPP_system MI.data_processing_capacity)
(hasConstraint MI.CAPP_system MI.product_definition)
(hasConstraint MI.CAPP_system MI.EASA_requirements)
(hasResource MI.CAPP_system MI.PLM_server)
(hasResource MI.CAPP_system MI.PLPS_process)
(hasResource MI.CAPP_system MI.PLM_database)
(hasResource MI.CAPP_system MI.production_schedule)
(hasResource MI.CAPP_system MI.manufacturing_bill_of_material)
(hasResource MI.CAPP_system MI.bill_of_plant)
(hasResource MI.CAPP_system MI.geometry_models)
(hasResource MI.CAPP_system MI.bill_of_process)
(hasResource MI.CAPP_system MI.casting_part_owner)
(hasResource MI.CAPP_system MI.wax_process_engineer)
(hasResource MI.CAPP_system MI.foundry_manufacturing_engineering_manager)
(hasInput MI.CAPP_system MI.core_specification)
(hasOutput MI.CAPP_system MI.core_specification)
(hasOutput MI.CAPP_system MI.core_chaplet_inspection_spec)
(hasOutput MI.CAPP_system MI.wax_process_instruction)
(hasOutput MI.CAPP_system MI.wax_process_parameter
(hasSecurityCategory MI.CAPP_system MI.Private)
(hasExportControlCategory MI.CAPP_system MI.ExportControlled)
(hasRetentionCategory MI.CAPP_system MI.MandatoryRetention)
(VisualSpecification MI.core_chaplet_inspection_spec)
(ProcessSpecification MI.wax_process_instruction)
(ProcessSpecification MI.wax_process_parameter)
(ProductSpecification MI.product_definition)
(SystemPerformanceMetric MI.CAPP_system_response_time)
(hasTimescale MI.CAPP_system_response_time_target MI.life_of_order)
(hasRetentionCategory MI.core_specification MI.MandatoryRetention)
(hasRetentionCategory MI.wax_process_parameter MI.MandatoryRetention)
(ProductSpecification MI.core_specification)
(MI.hasSecurityCategory MI.core_specification MI.Private)
(MI.hasSecurityCategory MI.wax_process_instruction MI.Private)
(MI.hasSecurityCategory MI.wax_process_parameter MI.Private)
(MI.hasSecurityCategory MI.core_specification MI.Private)
(MI.hasSecurityCategory MI.core_chaplet_inspection_spec MI.Private)
(hasTimescale MI.CAPP_system_response_time_target MI.life_of_contract)

8.4.5 Experiment 1 summary

While the formalization and testing is presented as a sequential activity, it can be seen that they were parallel or iterative activities resulting in a much stronger understanding of the ontology, the logic and the IODE tool. This understanding resulted in the ontology and constraining logic being formalized in the IODE tool successfully.

The testing phases proved that the solution concept as described in Figure 4-7 functioned as anticipated within the Manufacturing Systems domain, constraining the knowledge base to accept only manufacturing systems that meet its understanding of being designed for interoperability and was ready to be populated and tested against real, deployed systems evidence and used to validate that the concept is applicable and holds true across time frames.

8.5 Declaring 4 systems in the current timeframe - Experiment 2 results

8.5.1 Introduction

The work described in Section 8.4 resulted in a formalised multi-level ontology into which it was proven that facts could be declared which could be processed and if necessary fire logical rules to infer new facts or to reject the declared fact if it contravenes the logical constraints. This chapter details the work to apply this capability to a set of data which would test its ability to answer the research questions and prove or disprove the research hypothesis.

Four systems which had been recently designed for a state of the art automated manufacturing facility with interoperability as a key priority and which have proven their interoperability were declared to the system, these being:

- A Computer Aided Process Planning system.
- A Manufacturing Execution system
- A Part Tracking system
- A Cell Control system.

These systems had been described using IDEF0 notation from the BPMN diagrams and the methodology used to derive the facts from these was the same as used in the development phase in Chapter 7. This represented the largest populated fact base that the system had seen and effectively populated the Level 2 ontology or knowledge base.

8.5.2 Declaring new systems

The facts describing the systems shown in Figure 8-2 were declared to the ontology as they were defined from the BPMN diagrams. The declaration of the 1st of the 4 systems from the current timeframe (the computer aided process planning system) required over 300 facts

and resulted in 400 error messages. This represented a large number of syntax errors due to the input method being manual text input, and a large number of expected but missing declarations. While the system returned 400 errors messaged, it was apparent that there were only really 3 types of error:

- Syntax errors, i.e. coding errors.
- Missing declarations i.e. insufficient information or missing facts
- Explicit contravention of constraining ontology logic i.e. declaring an unacceptable fact.

On detailed investigation it was found that there were 20 syntax errors and 85 missing declarations, the remaining error messages were repeated errors due to the erroneous terms or facts affecting multiple logic strings or relationships. Using the error messages, the BPMN and the detailed system specifications which were also made available, the 85 fact declarations the ontology tool had prompted through error messages were added. The syntax errors were then corrected and the system was successfully loaded into the knowledge base.

The 2nd system declared to the ontology required only 120 facts and resulted in 300 errors messages (of which 130 were valid errors). It was noticed at this point that the reduced number of facts was due to many of the facts declared for the first system e.g. most of the people and their relationships had to be declared initially, and many of these people were involved with the second system. This initially seemed a positive aspect as it reduced the number of required declarations for subsequent systems but while investigating some of the errors if became apparent that this actually increases the risk of facts being inconsistently declared: it is very easy to declare the same concept using two names at different points for example a cell team lead could be declared as ‘MI.cell_Lead’ or ‘MI.cell_TeamLead’, and duel declarations of the same entity like this can disrupt the function of the logic as some logic may fire against either of both the instances with confusing results. Very few of these errors were syntax errors which was attributed to increased user experience and knowledge. The reason for the high level of errors i.e. more errors than facts, was that as well as missing information or declarations about the 2nd system, with two interoperating systems now declared into the ontology far more of the logic was brought into action, so many of the facts the ontology tool was prompting for were also invalidating or ‘firing’ a number of logic statements. The addition of the missing facts (as prompted by the logic error messages) resolved these errors and the 1st and 2nd system loaded correctly. It was noted at this stage that once the systems were ‘fully’ declared i.e. all mandatory information was provided, that

they did not fire any logic that would indicate they could not or should not interoperate which reflected the real world experience.

The 3rd and 4th systems were subsequently declared, requiring only 140 facts in total and each system resulting in only 40 error messages which represented only 6 or 7 actual errors. The reduction in the number of facts was once again due to a number of the facts required having already been declared for systems 1 and 2. The reduction in errors and facts is linked, as the fewer missing facts the fewer errors and this results in less false errors. Once again once the facts the ontology tool prompted for were added, no interoperability issues were flagged which correlates with the real world understanding of these systems.
Figure 8-3 shows the relative number of errors reported as each system was declared.

Overall the manual coding of the facts was laborious, slow and complex. Each subsequent system addition required constant reviewing of all the existing fact declarations to avoid duel declarations creating semantic inconstancy for specific instances. Due to this issue, the addition of system 4, (which had a quarter of the facts and one tenth of the error messages of system 1) took slightly longer than system 1. Based on the increasing complexity of this manual checking process adding many more systems would become exponentially more difficult.

Once all four systems were successfully loaded into the ontology knowledge base, the Level 2 ontology was considered populated enough to add a system from a different timeframe to investigate the research questions and hypothesis.

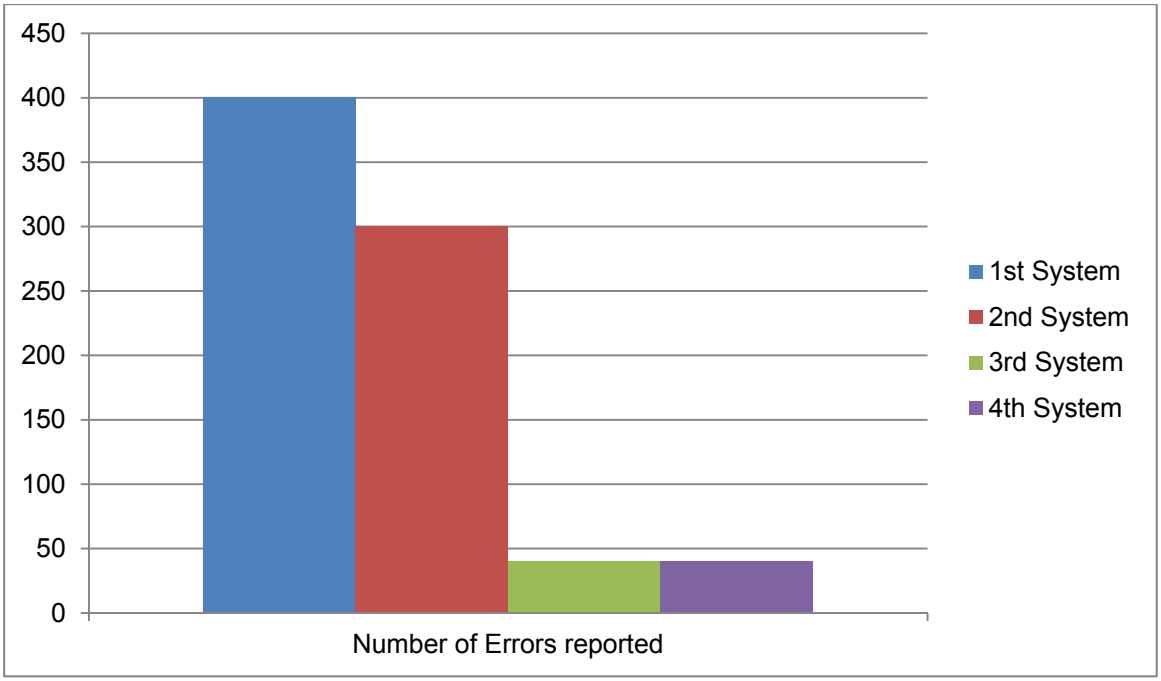


Figure 8-3 - Number of errors reported as each system was declared.

The process of loading the 4 real world systems into the knowledge base, reconfirmed the findings from Experiment 1 regarding the validity of the solution concept shown in Figure 4-7, but also validated that the rules used to test system interoperability within the solution concept were consistent with real world observations of deployed systems in that when fully declared the 4 systems were assessed as being interoperable which is in line with the observations of these systems in practice.

Figure 8-4 shows an IODE screenshot following the successful loading of 763 facts that were required to describe the 4 systems idenitified in the BPMN maps in suitable detail to meet to requirements of the ontology solution structural logic.

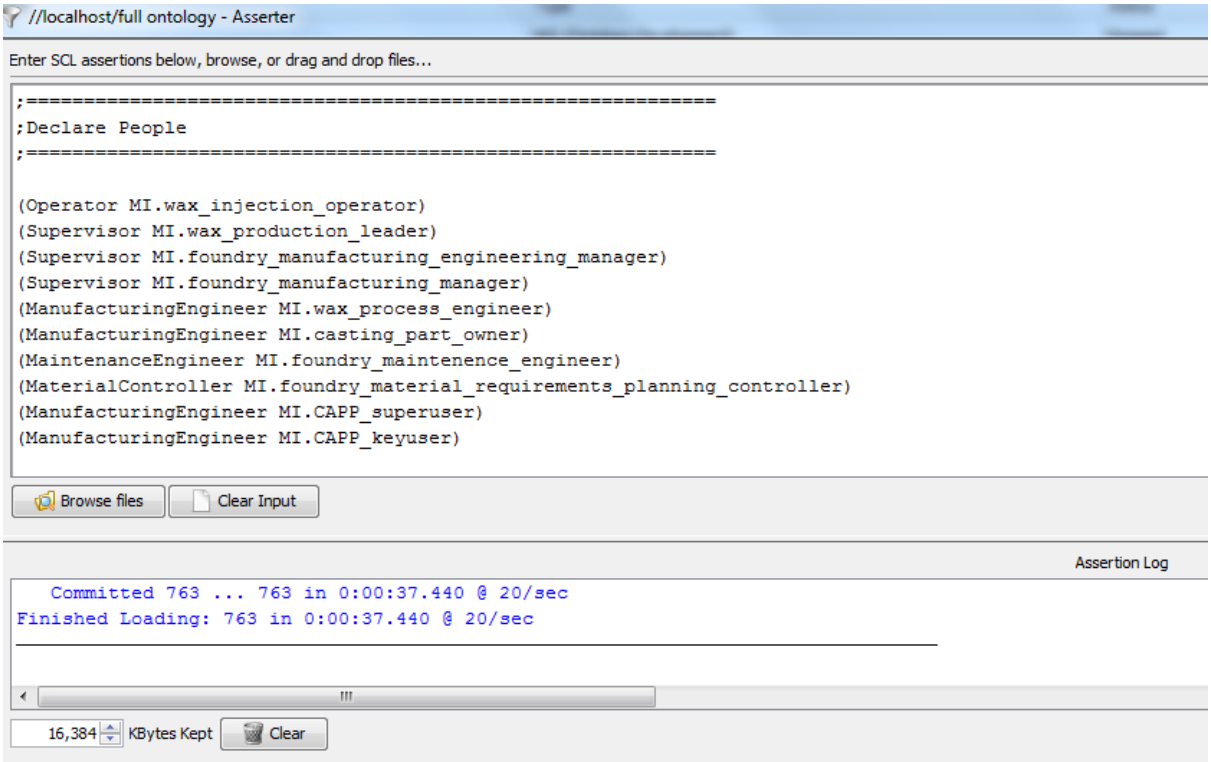
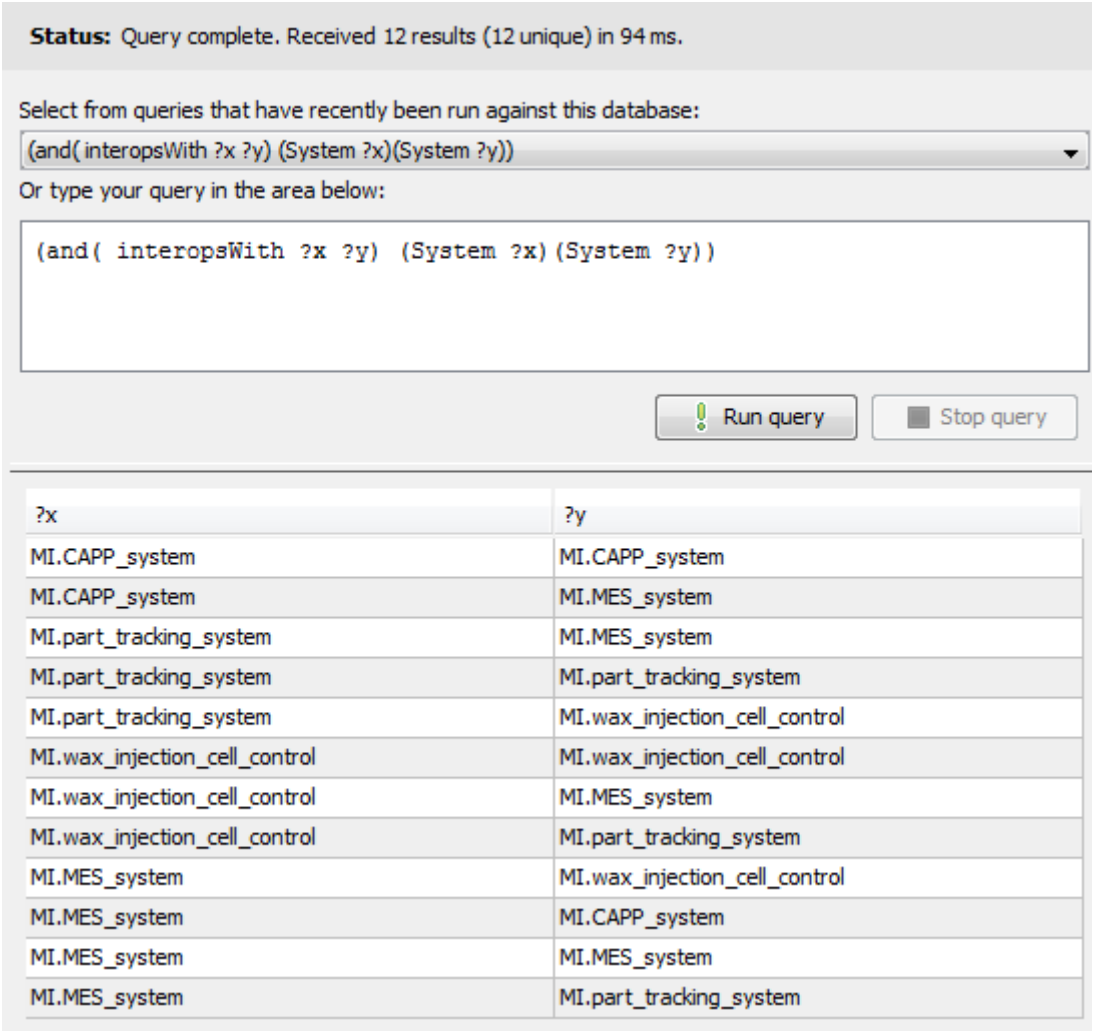


Figure 8-4 - IODE screenshot showing the 763 facts required to declare the 4 systems to the ontology have successfully loaded.

The final testing activity was to run the queries detailed in Section 7.4.6 against the fact base. Figure 8-5 shows the results of the key 'interoperates with' query which identifies what systems interoperate with each other. The solution correctly identified the interactions between the systems despite none of them being explicitly declared i.e. none of these systems were linked using an explicit '*interopsWith'* relationship, but these relationships were inferred through the complex ontology logic.



?x	?y
MI.CAPP_system	MI.CAPP_system
MI.CAPP_system	MI.MES_system
MI.part_tracking_system	MI.MES_system
MI.part_tracking_system	MI.part_tracking_system
MI.part_tracking_system	MI.wax_injection_cell_control
MI.wax_injection_cell_control	MI.wax_injection_cell_control
MI.wax_injection_cell_control	MI.MES_system
MI.wax_injection_cell_control	MI.part_tracking_system
MI.MES_system	MI.wax_injection_cell_control
MI.MES_system	MI.CAPP_system
MI.MES_system	MI.MES_system
MI.MES_system	MI.part_tracking_system

Figure 8-5 - IODE screenshot showing the results of the ‘interoperates with' query

8.6 Declaring a system from a different timeframe – Experiment 3 results

Finally, a legacy shop floor data collection system named ‘ActivPlant’ was declared to the system. This system had been designed and implemented twelve years before the other four systems and was therefore defined using different terms and was not designed to interoperate with the newer systems.

This experiment was designed to demonstrate the ability to determine the requirement for interoperability and the suitability for interoperability of systems from different timeframes and the potential to constrain system designs within the limits of interoperability. The experiment used a system from a previous timeframe as this allowed the results to be validated against the real world experiences as shown in Figure 8-6. The capability could then be inferred to future timescales, which cannot be proven in the same way.

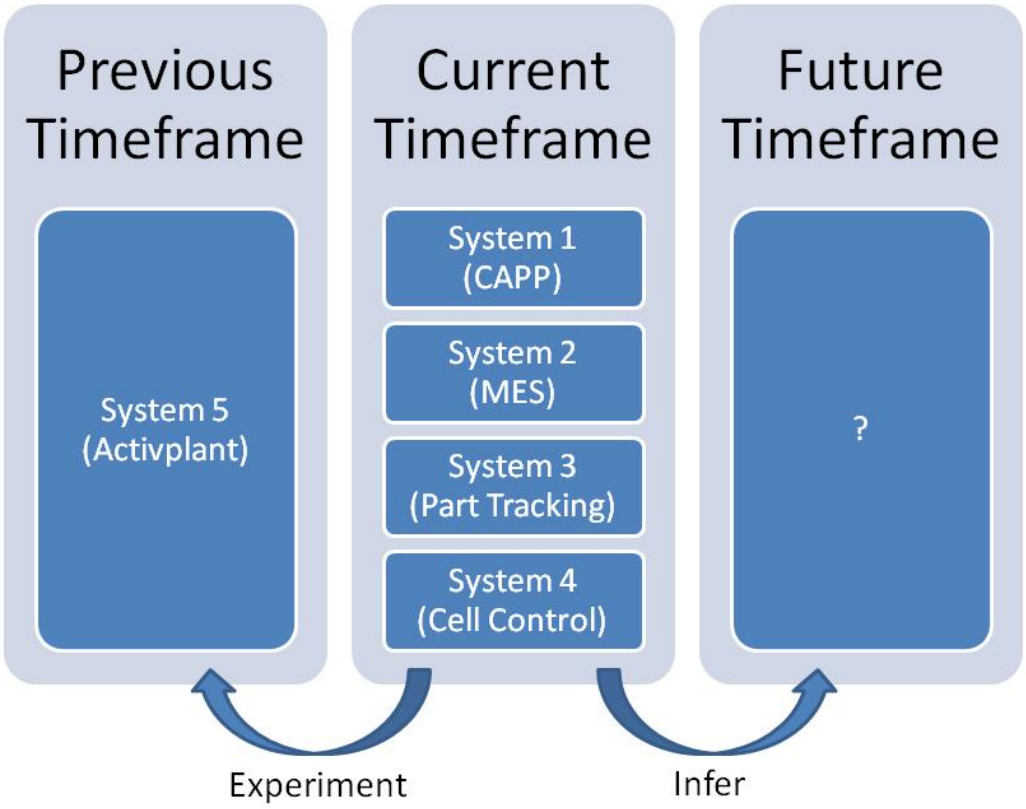


Figure 8-6 - Cross timeframe capability demonstrated using a previous timeframe to enable validation

This section details the results of loading these facts into the Ontology, the subsequent execution of the queries defined in Section 7.4.6. and the comparison of these results and the real world understanding of the interoperability issues between Activplant and the new systems.

Appendix D lists the facts collected from a BPMN and specification for the legacy Activplant system, which was installed in 1999. This was effectively the 5th system to be declared into the Level 2 ontology. The system required 120 facts to be declared which is an increase from systems 3 and 4. This reflected the fact that the system was not designed to interoperate with the other systems and therefore does not share or re-use many of their term instances. Unfortunately because some instances were shared the increasingly laborious and error prone manual review of all previously declared facts had to take place to ensure no duel declarations of the same instances.

8.6.1 Structural logic response

As shown in Figure 8-7 and Figure 8-8 the ontology solution reported that there were 83 instances of logical rules being violated by the legacy system facts, however it was identified that of these 83 viloations there were only 15 unique error messages (repeated a number or times) falling into the following 4 categories:

- Some dataset's retention category are unspecified

The system retention category was specified as 'non mandatory' meaning it is not suitable to be the sole repository for mandatory retention data. This system and any system interoperating with it require mandatory retention data sets to be output to other systems suitable for the mandatory retention of data. Confirmation that any overall system (i.e. the network of interoperating systems of which this is one) satisfies data retention requirements cannot be achieved while the data retention categories are ambiguous. The ontology logic fails-safe by assuming any data defaults to the highest (mandatory) retention category hence the system flagged that the retention categories are not all specified and also that not all input data sets are being output to other systems, which is a requirement for mandatory retention data being input to a non mandatory retention system (this is coded into the ontology logic).

- Metrics do not have validity timescales

The metrics and metric targets do not have validity timescales, meaning that the system itself and any interoperating system which are using the metrics to inform decisions (a key part of the MI process) will be unable to discern whether the metric or target are still valid. This gives rise to the significant risk of invalid decisions based on 'out of date' information (metrics were identified as a key decision input as part of the MI definition research). This timescale aspect is crucial when considering systems and process over multiple timeframes or a time continuum.

- Security Category of data unspecified

Similarly to the retention category issue, the lack of explicit security classification mean the nature and requirements for data exchange between systems is difficult to design suitably. The assumption of a high security requirement, while ‘fail-safe’ may result in interoperation with other systems being inhibited or prevented by unnecessary security rigor.

- Some data structures not specified

The lack of specification regarding data structures is a clear barrier to interoperability, as systems would be unable to reliably interpret and process exchanged or shared data. This can result in unpredictable or unreliable system function and output.

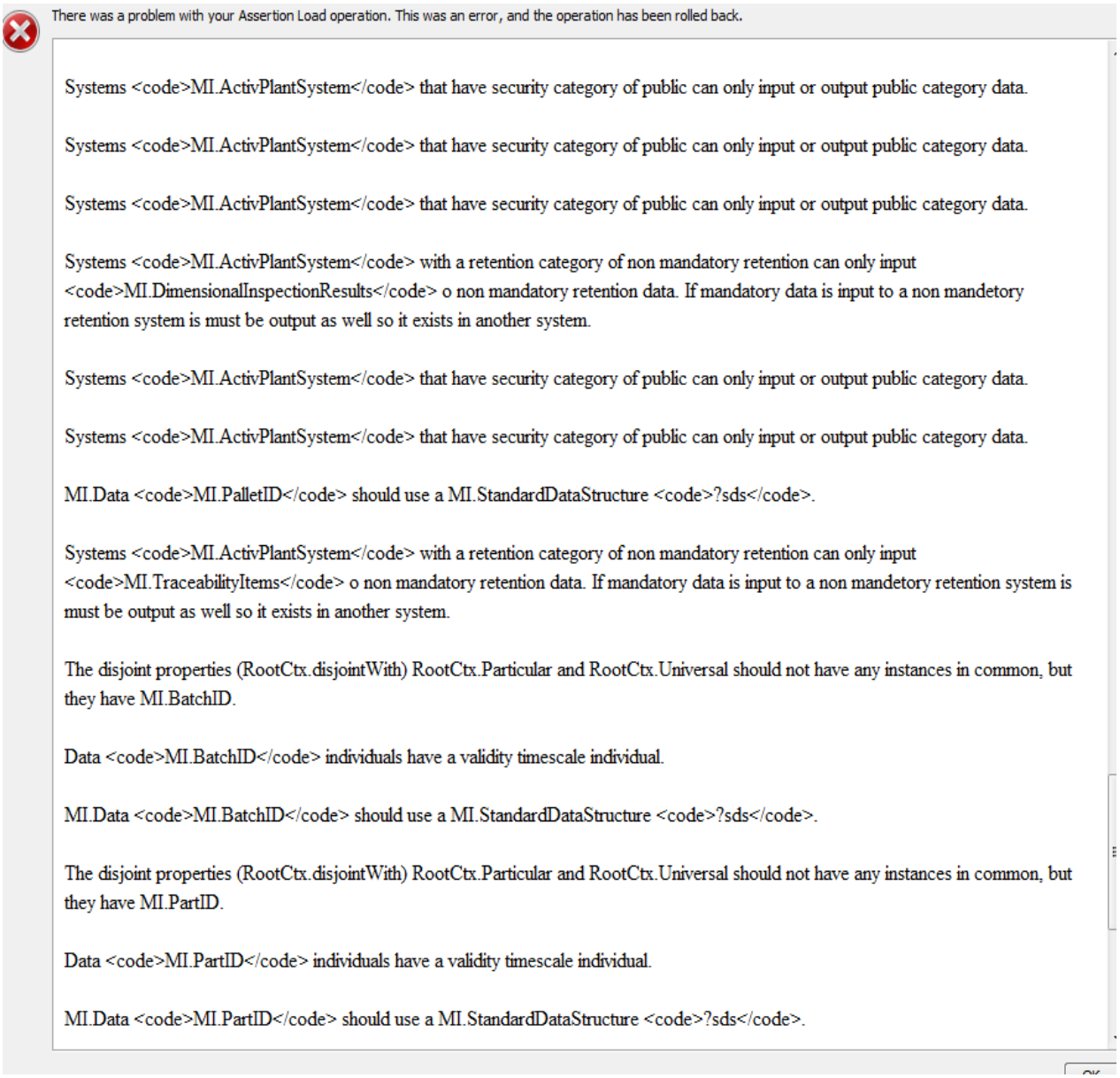


Figure 8-7 - IODE screenshot showing examples of the Activplant declaration errors

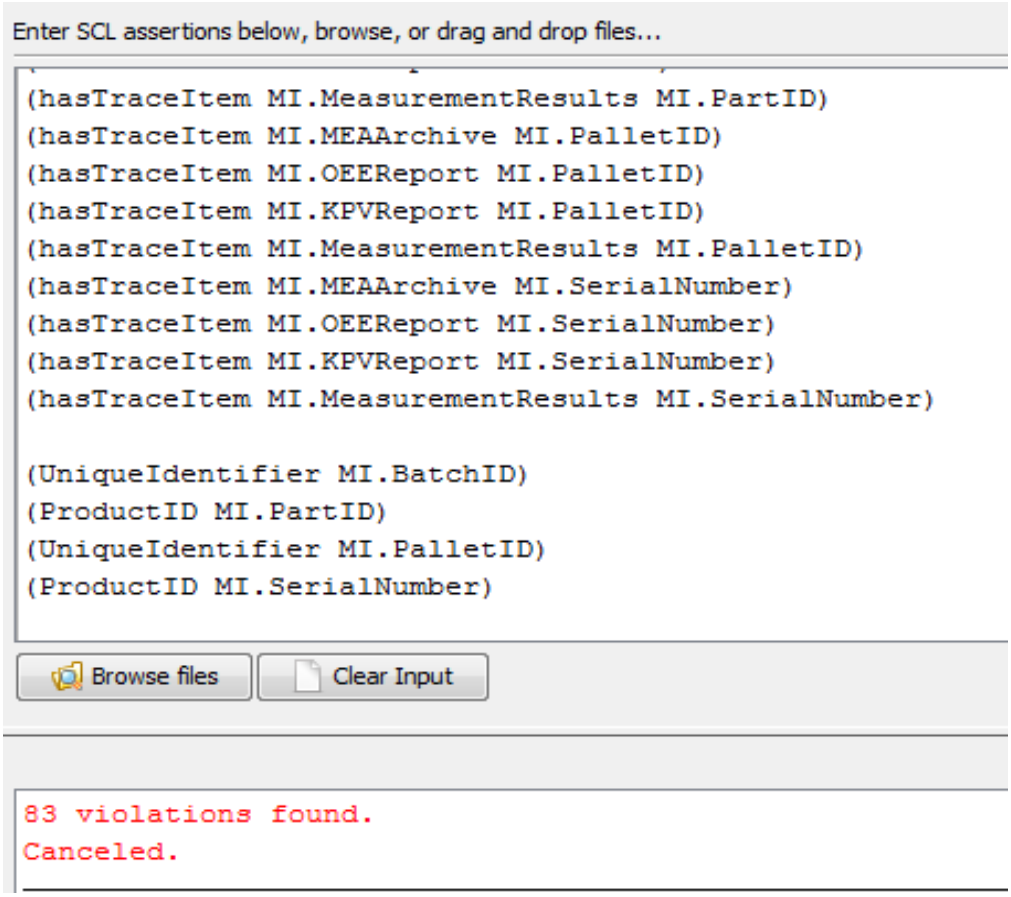


Figure 8-8 - IODE screenshot showing the Activplant load rejection due to 83 individual logic violations which were due to 15 unique errors as reported in the error messages.

The Activplant system was rejected from the ontology system meaning the ontology system therefore correctly declared the Activplant system as unsuitable for interoperation with the other systems in the knowledge base. While this limitation regarding interoperability was proven against the 4 systems previously declared to the knowledge base, it is reasonable to infer this inability to any systems than have not been specifically designed to overcome the Activplant systems interoperability shortcomings i.e. even systems not yet in the knowledge base.

8.6.2 Observed system issues

The actual issues experienced with the Activplant system were:

- Data Storage: The system is only certified for the storage of non mandatory retention data, which means any mandatory data input into the system has to be stored elsewhere. However at the point of implementation it was not clear which data sets this affected so no mechanisms for this archiving or export capability were put in place.

- Unexpected OEE results: the initial metric results from the system could not be validated. This was found to be due to the input signal data not being in the structure that had been assumed, leading to errors in the metric calculation.
- System slowing: this was found to be due to the system being unable to discern which datasets and metric were no longer valid. The system when processing reports or queries would therefore use all data in the database (including invalid legacy data). This was a similar issue for any other systems interoperating with the Activplant system: they had to take all possible data whether still relevant or not e.g. if a 'running' condition was set for a particular machine in the machine monitoring module, but not subsequently updated due to a system shutdown the machine would still be seen to be running by the system despite the machine signal no longer being valid (the machines never run for more than 1 hr continuously).
- Migration: when specifying a replacement system for the Activplant solution, it was found to be impractical to identify the system functions in detail due to the lack of clarity caused by the data structures not being explicit. Some of the data structures were not man-readable, meaning experimentation was required to derive the structure, and even then with some level of uncertainty. This has meant that the Activplant replacement system and all interfaces have had to be designed anew.

It can be seen that the ontology system rejection messages regarding Activplant are consistent with the nature of the interoperability (including migration which can be thought of as interoperability between versions of the same system) issues observed in reality.

8.6.3 Query response

A key issue with the solution model shown in Figure 7-1 was identified at this stage: if the structural logic rejects the system from the ontology for not meeting the logical requirements for interoperability, then the system does not exist in the ontology to allow the queries listed in Section 7.4.5 to be run.

To overcome this issue for this research, a separate version of the ontology was created where all logic rule were declared as 'soft' i.e. they would flag an error but not prevent the facts loading. The facts were then loaded and the queries run. The queries correctly returned the expected results, identifying inputs, outputs, resources and constraints and inferring through the structural logic that this system would be required to interoperate with the newer systems.

As should be expected, manual queries of other logic returned unreliable results as the Activplant declaration did not meet the requirements of the structural logic, and the IODE

tool is designed to reject unacceptable facts. On this basis, while the queries were proven to work in this case, forcing declarations into the ontology in this way should not be used as part of the process and the application of the solution concept should itself be considered a sequential process as shown in Figure 7-1. If a system fails to load into the knowledge base, then the process cannot proceed to the query stage until the issues raised by the logic are addressed. It may be beneficial for a mechanism for running these queries against systems prior to loading to prevent failure in system design and loading, however, this is out of scope of this research.

8.7 Experimental results summary

The results from the experiments showed that the ontology was able to identify systems that interoperate, even if that relationship is not explicitly declared (in Figure 8-5). If the systems are not suitable for interoperation it will reject the system from the knowledge based returning a related error. The solution demonstrated this capability within a single timeframe and between systems from different timeframes. The semantic and logical consistency provided by the heavyweight ontology with a core concept ontology level provided by the solution concept described in Section 4.2.2, meet the requirement described in Section 4.2.1 which validates the concept described in Figure 4-7.

The errors generated by the ontology relating to the Activplant system were consistent with the real world issues with the system. While the detailed failure modes may have been difficult to predict, if this approach had been used for the Activplant system the 4 root causes to a number of operation and interoperation issues would have been resolved at the definition stage.

The queries designed to answer the competency questions that were not resolved by the structural logic were shown to be effective in themselves, but the issue was identified that the queries cannot be run against a system that has been rejected from the knowledge base. This creates a dependency, that the fundamental design of a system must be compatible with interoperability before the further queries that would aid system design work, can be run.

9 Discussion, conclusions and further work

9.1 Introduction

This chapter reviews the experimental results against the hypothesis and the research questions. Conclusions are drawn about the research and solution concept before suggestions for further work are made.

9.2 Hypothesis review

The research hypothesis was stated in Section 1.2.4 as: “Multiple systems can be defined using a consistent foundational ontology comprising constraining logic to highlight whether they meet the requirements for interoperation. Interoperability can then be ensured for and with future systems and versions of systems through the inherent process of system definition enabling the rapid development of interoperable systems.”

This has been proven to be conceptually true. The factors limiting the pace of development are largely technological and so should be practically resolvable through further development.

A possible limitation discovered through this the research was that a system can only be analyzed by queries once it is in the knowledge base as shown in Figure 7-1, but it cannot enter the knowledge base reliably until it meets the requirement for interoperability. This limits the amount of information about a system that can be gained by the user to that provided by the structural logic feedback (when rejecting a new system declaration to the knowledge base).

9.3 Research questions review

The research questions, as stated in Section 4.3, have been answered in the following way:

1. What is Manufacturing Intelligence?

The research detailed in Chapter 5 resulted in a definition which was able to resolve the apparent inconstancies in the current descriptions of MI: “Manufacturing Intelligence enables good manufacturing decisions based on understanding of the current status and the ability to predict and control the outcome of any given decision.”

Its purpose being: “Communicating and improving manufacturing performance as quantified by appropriate metrics, with the appropriateness of metrics also being informed by MI understanding.

On achievement of MI: “The organisation has the right metrics and targets in place to achieve the organisations objectives. Everyone is aware of the current performance of the organisation and process against the target metrics, likely future trends in the metrics and how to either maintain or improve performance to achieve the short, medium and long term targets”.

2. What are the concepts in the MI Systems domain?

The UML diagrams in Section 6.2.1 and Figure 6-40 show the Level 1 and Core Concept ontologies, while Appendix C shows the full ECLIF Fact Listing that was used to populate the Level 2 Ontology, which in this case is actually the knowledge base. These ontologies together form the MI systems ontology, albeit the Core and Level 1 ontologies would be applicable in other domains as they are effectively Systems and Manufacturing systems ontologies in their own right.

3. Can a Domain foundational ontology or Core Concept be defined and formalised?

The domain foundational ontology has been defined in this research as the core concept ontology as defined in Section 6.5. The formalisation of this has been described in Chapter 7 resulting in the KFL and ECLIF code within the IODE ontology tool. The resulting solution was tested and proven to respond in line with the design intent.

4. Can the proposed concept be proven through the formalisation of the ontology?

The solution proposal described in Section 4.2.2 has been successfully proven in Chapter 8.5: the heavyweight ontology has provided a basis of logical and semantic

consistency across timeframes, in this case for the analysis and definition of MI systems interoperability capability.

9.4 Key observations

The following conclusions regarding the solution concept, ontology development and IODE tool were drawn from the research:

9.4.1 Solution concept

- The research has successfully answered the research questions and the solution concept can therefore be considered to have proven the research hypothesis.
- This solution was demonstrated using the MI domain but the same approach would be applicable to any other system domain due to the scope of the core concept ontology.
- In this work it is demonstrated that the ontology solution provides consistency of meaning across timeframes, as it would across physical or organisational domains (see Figure 4-7) therefore different timeframes can be considered in a similar way to conventional domains.
- It has been shown that the solution concept shown in Figure 4-8 allows new systems to be defined using new terms, whilst maintaining constancy with legacy terms, and allowing the legacy logic to be applied to the new terms appropriately due to the consistency provided by the core ontology.
- The queries that allow rapid understanding of the systems within the knowledge base cannot be used on system facts that failed the structural logic tests as this failure results in the system facts being rejected from the knowledge base. Initial feedback is therefore limited to the error messages from the attempt to load the system facts into the knowledge base. While this does not hazard the success of the solution, it does significantly increase the time to resolve MI system design for interoperability issues.

9.4.2 Ontology development

- Defining constraining logic at the right level of specificity or ontology level is crucial to the solution concept succeeding e.g. relationships may be used between multiple classes meaning the relationship signatures have to be defined at the highest common level and then logic used to constrain the relationship back down to an appropriate level.
- The heavyweight ontology development led to the systemic nature of the system entities being formalised within the ontology structure (i.e. a system could be identified as such by the attributes that make it a system rather than its name): the entities were logically constrained such that they must have inputs, outputs, resources and constraints defined to be valid.
- The ‘System’ core concept ontology would be applicable to any systems field.
- The Manufacturing Systems ontology (Level 1 ontology) which includes the core concept ontology is applicable to any Manufacturing Systems field.

- Domain ontologys (Level 1) require the most constraining logic and ontology definitions when compared to core concept or Level 2 ontologys. They also therefore generate significantly more testing errors and complexity.
- The level of term abstraction referenced in ontology logic should be used to define at what ontological level the logic resides to aid logic development and control: logic that only uses the types in the core ontology should be considered the primary core logic, the logic that references domain Level 1 types will be the next level and instance logic the most specialised logic. Logic that refers to both types and instances may have to be split to allow this tiering to take place
- The understanding of the ontology specialisation shown in Figure 4-5 evolved throughout the research: the term 'foundation ontology' was updated to 'core concept ontology' as it is a foundational ontology basis within this domain, and the instances actually formed part of the MI or Level 2 ontology (see Figure 9-1).
- The overall MI system ontology is invalid without all 3 levels of ontology, as is the constraining logic, hence the decision to include the instances as part of the ontology.

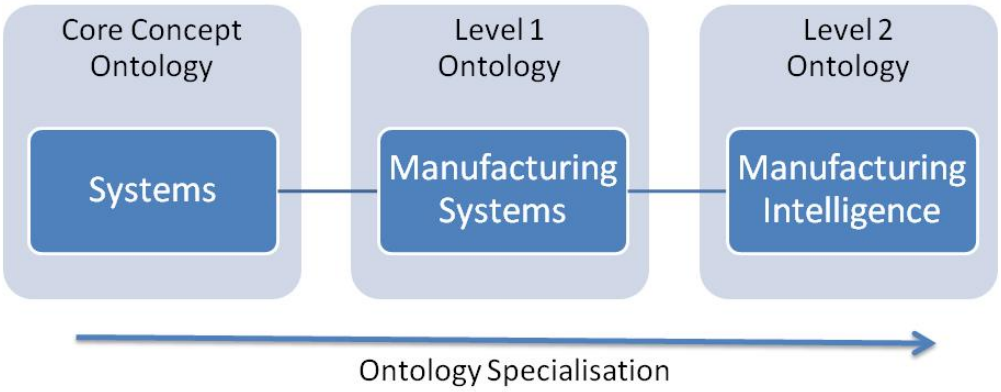


Figure 9-1 - The updated view of the ontology specialisation levels

- The use of the core concept ontology and Level 1 ontology ensures that any new concepts that emerge over time, can be described in a consistent way and adopt any constraining logic that has been defined for its parent concept allowing new concepts and instances to be added over time, whilst maintaining the ontological and logical consistency and constraints.
- While the development of new concepts was considered by this research, the ongoing development of the constraining logic was not. The later stages of the development of the logic, where corrections were made, proved that changing the logic can have very unpredictable results due to the change in inference conditions and constraints. Adding logic without understanding all of the existing logic would give rise to a significant risk of

invalidating the overall logic through the creation of self referencing logic or circular statements. This issue was considered out of scope for this research but would be a significant area of future work.

9.4.3 The IODE tool

- The ontology tool used in this research (IODE) proved very powerful and comprehensive. It can however appear to generate false errors on failures to load facts into the knowledge base, and due to its lack of graphical interface or visual programming aids or prompts it requires a significant coding capability, knowledge of ECLIF and KFL and significant manual debugging of code.
- When declaring large numbers of facts into a large knowledge base, it is very easy to declare a fact twice (semantically or syntactically inconsistency). This is a significant hazard to the function on the structural logic.

9.5 Novelty areas

The new knowledge generated through this research fall into the following key areas:

- A definition of Manufacturing Intelligence: a clear definition of Manufacturing Intelligence which resolves the identified inconsistencies of scope and meaning.
- The solution concept (as described in Figure 4-8) for evaluating system interoperability in dynamic environments for current and emerging systems, through the use of heavyweight ontologies to provide semantic consistency across timeframes.
- A 'Systems' core concept heavyweight ontology (Figure 9-2)
- A 'Manufacturing Systems' domain heavyweight ontology (Level 1)
- A 'Manufacturing Intelligence' domain heavyweight ontology (Level 2)

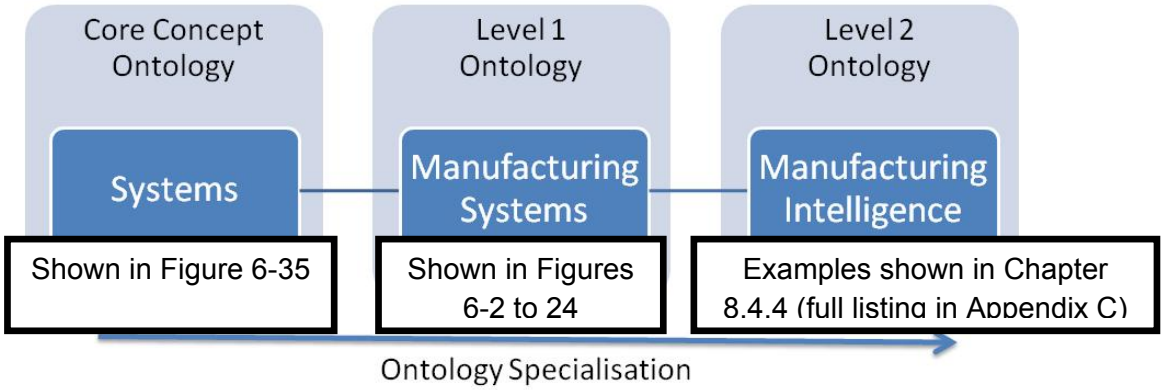


Figure 9-2 The ontology representation with references to the ontology term listings

- Common logic rules (as part of the heavyweight ontologies) focusing on the requirements for systems interoperability.
- A set of definitive question which can be used to evaluate a systems suitability and requirement for interoperability (the ontology competency questions in Figure 4-2)
- Detailed knowledge about the creation of multi-level heavyweight ontologies.

Figure 9-3 summarises the novelty detailed within this thesis: the novel concept of a core concept ontology providing consistency across timeframes which was developed into the solution for system interoperability through the creation of multi level ontologies, constraining logic and a domain definition.

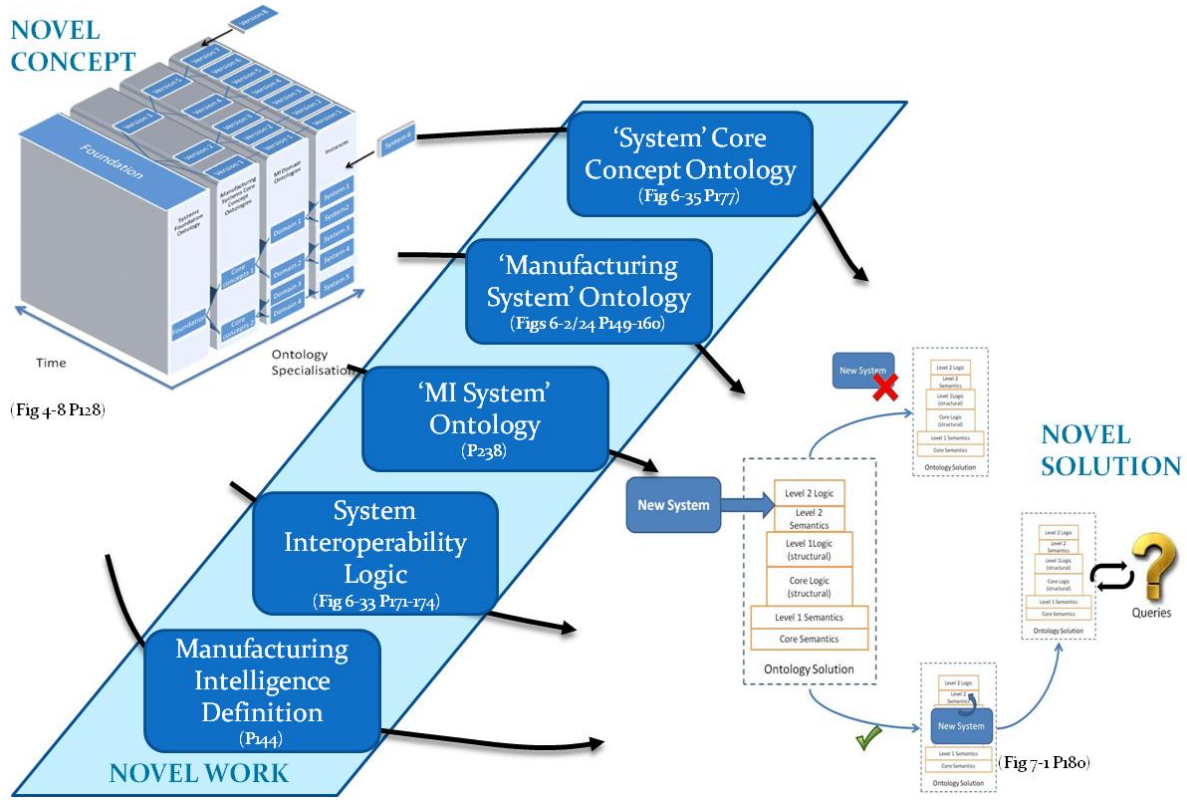


Figure 9-3 Summary of research novelty

9.6 Further work

This work defined and explored an approach to providing a rigorous mechanism for the consistent sharing of meaning through heavyweight ontologies, which uses a multi-level ontology with domain ontologies built upon a core concept ontology to provide cross domain consistency (in this work, the timeframe being considered is treated as a domain). The heavyweight nature of the ontologies allows constraining, automated logic to be constructed to infer additional facts based on those declared to it and to test for specific conditions; in this case interoperability. This research identified areas for future work as being:

- Can the approach to assessing interoperability explored in this research be applied to unrelated fields and other constraining requirements e.g. could the same approach be used to idenitify and ensure that all legal cases taken on by a legal body are successful? In this example, the heavyweight ontology would be required to create a common understanding of legal case precedence and legislation, with the multi-level ontology approach allowing domain ontologies in the different legal domains, while the constraining logic and queries would be used to set and assess the criteria for success?
- Can the solution concept explored in this work be combined with real time inputs and a decision making capability to provide an intelligent automation solution? If a solution, could:
    - Use a heavyweight multi level ontology approach to provide consistent meaning across domains which experience different levels and rates of change.
    - Combine the text based IODE declarations with real time condition signals, and replace the interoperability logic with other decision making logic.
- Then it may be possible to create a solution that takes inputs from multiple domains, some of which are ‘sensed’ in real time, interprets them consistently, and provides decision feedback based on structural logic. An example might be a production scheduling solution, which is able to consistently interpret text based input from many enterprise functions (e.g. finance, sales, maintenance), along with real time signals (e.g. stock level, waiting time, equipment availability) and structural logic to decide what products to launch, when, and where based on the current and likely upcoming conditions.
- A method of developing the structural logic in the solution concept through time is required that will allow resolution of any issues this raises with the ontology or knowledge base.

- Exploring the applicability of the ‘System’ core concept ontology to other system fields.
- Usability developments to the IODE tool set such as:
    - The ability to easily identify similar or identical facts in the knowledge based during declaration (i.e. before the facts are committed to the knowledge base) to avoid unintended duel declarations.
    - A mechanism for loading unacceptable facts temporarily to allow some queries to be run on them (this would require the tool to resolve the violated logic temporarily to ensure queries return valid results).
    - A native, configurable graphical user interface that’s allows queries etc to be assigned to buttons etc without the need for significant programming skills.

10 References

Agostinho, C., Almeida, B., Nuñez-Ariño, M.J. & Jardim-Gonçalves, R. 2009, "Interoperability and Standards: The Way for Innovative Design in Networked Working Environments", *Proceedings of the 19th CIRP Design Conference*, March 13, 2009, pp. 139.

Alsafi, Y. & Vyatkin, V. 2010, "Ontology-based reconfiguration agent for intelligent mechatronic systems in flexible manufacturing", *Robotics and Computer-Integrated Manufacturing,* vol. 26, no. 4, pp. 381-91.

AMR Research 2011, 7/06/2011-last update*, Enterprise Manufacturing Intelligence* [Homepage of Wikimedia Foundation Inc], [Online]. Available: http://en.wikipedia.org/wiki/Enterprise_manufacturing_intelligence [2011, 07/07].

Anon 2008, "Manufacturers are investing heavily in PLM", *T & P: Tooling & Production,* vol. 74, no. 11, pp. 39-39.

Bell, D. 2004, 15/09/2004-last update*, UML basics: The class diagram* [Homepage of IBM], [Online]. Available: http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/ [2011, 6/1/2011].

Blomqvist, E. & Öhgren, A. 2008, "Constructing an enterprise ontology for an automotive supplier", *Engineering Applications of Artificial Intelligence,* vol. 21, no. 3, pp. 386-397.

Borgo, S. & Leitão, P. 2007, *Foundations for a Core Ontology of Manufacturing*, Springer US.

Brown, M. 2007, "'Don't give me more data; give me more knowledge!'", *Hydrocarbon Processing,* vol. 87, no. 4, pp. 19.

Cassidy, P. 2008, "Toward an open-source foundation ontology representing the Longman's defining vocabulary: The COSMO Ontology OWL version", *Proc. Third International Ontology for the Intelligence Community Conference*, eds. K.B. Laskey & D. Wijesekera, CEUR Workshop Proceedings, Fairfax, VA.

Cesar, P. 2008, "A scada wireless network attack protection", *Intech'08 - Proceedings of the 9th International Conference on Intelligent Technologies,* vol. ISSU, pp. 139-142.

Chen, D., Doumeingts, G. & Vernadat, F. 2008, "Architectures for enterprise integration and interoperability: Past, present and future", *Computers in Industry,* vol. 59, no. 7, pp. 647-659.

Chen, L.-. & Chien, C.-. 2011, "Manufacturing intelligence for class prediction and rule generation to support human capital decisions for high-tech industries", *Flexible Services and Manufacturing Journal,* vol. 23, no. 3, pp. 263-289.

Choi, S.S. 2010, "XML-based neutral file and PLM integrator for PPR information exchange between heterogeneous PLM systems", *International Journal of Computer Integrated Manufacturing,* vol. 23, no. 3, pp. 216-228.

Chun-Che Huang & Shian-Hua Lin 2010, "Sharing knowledge in a supply chain using the semantic Web", *Expert Systems with Applications,* vol. 37, no. 4, pp. 3145-61.

Chungoora, N. & Young, R.I.M. 2010a, *A framework to support semantic interoperability in product design and manufacture*, Loughborough University.

Chungoora, N. & Young, R.I.M. 2010b, "The configuration of design and manufacture knowledge models from a heavyweight ontological foundation", *International Journal of Production Research,* .

Coates, G.M., Hopkinson, K.M., Graham, S.R. & Kurkowski, S.H. 2010, "A Trust System Architecture for SCADA Network Security", *Power Delivery, IEEE Transactions on,* vol. 25, no. 1, pp. 158-169.

Costa, C.A. & Young, R.I.M. 2001, "Product range models supporting design knowledge reuse", *Proceedings of the Institution of Mechanical Engineers -- Part B -- Engineering Manufacture,* vol. 215, no. 3, pp. 323-337.

Degler, D. & Battle, L. 2003, "Knowledge Management in Pursuit of Performance: The Challenge of Context" in *EPSS Revisited: A Lifecycle for Developing Performance-Centered Systems*, ed. G. Dickelman, International Society for Performance Improvement, Books34x7.

Deshayes, L., Foufou, S. & Gruninger, M. 2007, *An Ontology Architecture for Standards Integration and Conformance in Manufacturing*, Springer Netherlands.

Dorador, J.M. & Young, R.I.M. 2000, "Application of IDEF0, IDEF3 and UML methodologies in the creation of information models", *International Journal of Computer Integrated Manufacturing,* vol. 13, no. 5, pp. 430-445.

FIPS, P. 1993, "183 (1984).“Integration definition for Function Modeling (IDEF0)”. Federal Information Processing Standards, United States National Institute of Standards and Technology (NIST)", *Computer Systems Laboratory, Gaithersburg,* .

Frankovic, B. & Budinska, I. 2006, "The Role of Ontology in Building of Knowledge Systems for Industrial Applications", *4th Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence*Herľany, Slovakia, Jan 20-21, pp. 15-25.

Fulcher, J. 2005, "Just married: integration platforms plus automated mapping systems boost continuous-improvement efforts", *Manufacturing Business Technology.Vol.23,* vol. 23, no. 9, pp. 46-47.

Guerra-Zubiaga, D.A. & Young, R.I.M. 2008, "Design of a manufacturing knowledge model", *International Journal of Computer Integrated Manufacturing,* vol. 21, no. 5, pp. 526.

Gunendran, A.G. & Young, R.I.M. 2006, "An information and knowledge framework for multi-perspective design and manufacture", *International Journal of Computer Integrated Manufacturing,* vol. 19, no. 4, pp. 326-338.

Guo Wen-yue, Qu Hai-cheng & Hong, C. 2010, "Semantic web service discovery algorithm and its application on the intelligent automotive manufacturing system", .

Hak-Man Kim, Jong-Joo Lee & Dong-Joo Kang 2007, "A Platform for Smart Substations", *Future Generation Communication and Networking (FGCN 2007)*, pp. 579.

Hodges, M.S. 2007, *Computers: Systems, Terms and Acronyms, 17th Edition,* SemCo Enterprises.

Huang, K., Yang, W.L. & Wang, R.Y. 1999, *Quality information and knowledge,* Prentice Hall, Upper Saddle River, NJ.

IBM, 1993, *IBM Dictionary of Computing,* 10th edn, McGraw-Hill, Inc, New York, NY, USA.

IODE, H. 2010, "Ontology Library reference" in Highfleet's IODE/XKS Documentation, Baltimore.

ISA 2005, *ANSI/ISA-95.00.03-2005, Enterprise-Control Systems Integration Part 3: Activity Models of Manufacturing Operations Management*, American National Standards, North Carolina.

ISA 2000, *ANSI/ISA-95.00.01-2000, Enterprise-Control Systems Integration Part 1: Models and Terminology*, American National Standards, North Carolina.

ISO 2010, *ISO 15531-44 Industrial automation systems and integration -- Industrial manufacturing management data -- Part 44: Information modelling for shop floor data acquisition*, ISO, Geneva.

ISO 2007, *ISO/IEC 24707 Information technology — Common Logic (CL): a framework for a family of logicbased languages*, ISO, Switzerland.

ISO 2004, *ISO 18629-1: Industrial automation systems and integration -- Process specification language -- Part 1: Overview and basic principles*, ISO, Switzerland.

ISO 2002, *ISO/IEC 15288: Systems and software engineering - System life cycle processes*, ISO, Geneva.

ISO 1999, *ISO TC184/SC5/WG1 Industrial Automation Systems—Concepts and Rules for Enterprise Models, April 14, 1999*.

Izza, S. 2009, "Integration of industrial information systems: from syntactic to semantic integration approaches", *Enterprise Information Systems,* vol. 3, no. 1, pp. 1-57.

Jacobson, S.F., & Eriksen, L. 2011, *The Manufacturing Performance Dilemma, Part 1: Overcoming Visibility Hurdles With Enterprise Manufacturing Intelligence*, Gartner, Stamford, USA.

Jones, S. *, Improve Your Business Decisions*. Available: http://www.pcpro.co.uk/business-intelligence/research [2011, 5/8/2011].

Jr., J.A.J., & Michel, F.J. 2000, *Next Generation Manufacturing: Methods and Techniques,* John Wiley & Sons (US).

Kim, K., Manley, D.G. & Yang, H. 2006, "Ontology-based assembly design and information sharing for collaborative product development", *Computer-Aided Design,* vol. 38, no. 12, pp. 1233.

Kyoung-Yun Kim, Chin, S., Kwon, O. & Ellis, R.D. 2009, "Ontology-based modeling and integration of morphological characteristics of assembly joints for network-based collaborative assembly design", *(AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* vol. 23, no. 1, pp. 71-88.

Lagos, N. & Setchi, R.M. 2007, "A manufacturing ontology for e-learning", *Innovative Production Machines and Systems*, July 2-13, pp. 552-557.

Latif, M.N., Boyd, R.D. & Hannam, R.G. 1993, "Integrating CAD and manufacturing intelligence through features and objects", *International Journal of Computer Integrated Manufacturing,* vol. 6, no. 1, pp. 87-April.

Lecky, S. 2003, "Systems documentation explained - the Data Dictionary", *The Journal of Information Development,* vol. Ist Quarter.

Li, Z., Yang, M.C. & Ramani, K. 2009, "A methodology for engineering ontology acquisition and validation", *(AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* vol. 23, no. 1, pp. 33-47.

Lin, H.K. & Harding, J.A. 2007, "A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration", *Computers in Industry,* vol. 58, no. 5, pp. 428-437.

Liu, S. & Young, R.I.M. 2007, "An exploration of key information models and their relationships in global manufacturing decision support", *Proceedings of the Institution of Mechanical Engineers -- Part B -- Engineering Manufacture,* vol. 221, no. 4, pp. 711-724.

Maizlish, B. & Handler, R. 2005, *IT Portfolio Management Step-by-Step: Unlocking the Business Value of Technology,* John Wiley & Sons (US).

Masolo, c., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A. & Schneider, L. 2003, *The WonderWeb library of foundational ontologies preliminary report*, ISTC-CNR, Padova, Italy.

Matt, D.T. 2007, "Reducing the structural complexity of growing organizational systems by means of axiomatic designed networks of core competence cells", *Journal of Manufacturing Systems,* vol. 26, no. 3-4, pp. 178-187.

Meier, H., Roy, R. & Seliger, G. 2010, "Industrial Product-Service Systems--IPS2", *CIRP Annals - Manufacturing Technology,* vol. 59, no. 2, pp. 607.

MESA 2012, *, What is P2E? It's 'Plant to Enterprise'* [Homepage of MESA International], [Online]. Available: http://www.mesa.org/en/aboutus/whatisp2e.asp [2012, 03/16].

Mochal, T. 2005, *The Complete Book of Project-Related Terms and Definitions: Mysteries Explained,* TenStep, Inc.

Noy, N.F. & McGuinness, D.L. 2001, , *Ontology Development 101: A Guide to Creating Your First Ontology* [Homepage of Stanford Knowledge Systems Laboratory], [Online]. Available: http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html3/30/2011].

OMG 2012, , *Object Management Group Business Process Model and Notation* [Homepage of Object Management Group], [Online]. Available: http://www.bpmn.org/ [2011, 02/02].

OMG 2008, 17/01/2008-last update, *Business Process Modeling Notation, V1.1* [Homepage of Object Management Group], [Online]. Available: http://www.omg.org/spec/BPMN/1.1/PDF [2011, 24/5/2011].

Panetto, H. & Molina, A. 2008, "Enterprise integration and interoperability in manufacturing systems: Trends and issues", *Computers in Industry,* vol. 59, no. 7, pp. 641-646.

Qiao, L. & Liu, W. 2009, "Agile manufacturing data management", vol. 407-408, pp. 189-193.

Rafinejad, D. 2007, *Innovation, Product Development and Commercialization: Case Studies and Key Practices for Market Leadership,* J. Ross Publishing.

Ray, S. & Jones, A. 2006, *Manufacturing interoperability*, Springer Netherlands.

Scholten, B. 2009, *MES Guide for Executives: Why and How to Select, Implement, and Maintain a Manufacturing Execution System,* ISA, North Carolina, USA.

Schoop, R., Colombo, A.W., Suessmann, B. & Neubert, R. 2002, "Industrial experiences, trends and future requirements on agent-based intelligent automation", *IECON 02 [Industrial Electronics Society, IEEE 2002 28th Annual Conference of the]*, pp. 2978.

Shen, T.H., Carlson, C.S. & Peter Tarczy-Hornoch 2009, *SNPit: A federated data integration system for the purpose of functional SNP annotation*, Elsevier Scientific Publishers.

Siemens AG 2011, 12/04/2011-last update, *Enterprise Manufacturing Intelligence* [Homepage of Siemens AG], [Online]. Available: http://www.automation.siemens.com/mcms/mes/en/mes_suites/intelligencesuite/Documents/ED_IS_general.pdf [2011, 07/12].

Skarka, W. 2005, "Contemporary problems connected with including standard for the exchange of product model data (ISO 10303 - STEP) in designing ontology using UML and XML", *Computer Assisted Mechanics and Engineering Sciences,* vol. 12, no. 2, pp. 231-246.

Taghaboni-Dutta, F., Trappey, A.J.C. & Trappey, C.V. 2010, "An XML based supply chain integration hub for green product lifecycle management", *Expert Systems with Applications,* vol. 37, no. 11, pp. 7319-28.

Taylor, J. 2003, *Managing Information Technology Projects: Applying Project Management Strategies to Software, Hardware, and Integration Initiatives,* AMACOM.

Toung, E. 2006, "Plant floor, enterprise intersect at 'manufacturing intelligence'", *Plant Engineering,* vol. 60, no. 9, pp. 31-4.

Unver, H.O. 2012, "An ISA-95-based manufacturing intelligence system in support of lean initiatives", *International Journal of Advanced Manufacturing Technology,* , pp. 1-14.

Usman, Z. 2012, *A Manufacturing Core Concepts Ontology to Support Knowledge Sharing (Doctoral Thesis)*, Loughborough University.

Usman, Z., Young, R., Chungoora, N., Palmer, C., Case, K. & Harding, J. 2011, "A Manufacturing Core Concepts Ontology for Product Lifecycle Interoperability" in , eds. M. Sinderen & P. Johnson, Springer Berlin Heidelberg, , pp. 5-18.

Vincent Wang, X. & Xu, X.W. 2013, "An interoperable solution for Cloud manufacturing", *Robotics and Computer-Integrated Manufacturing,* .

Wang, P. 2010, "Chasing the Hottest It: Effects of Information Technology Fashion on Organizations", *MIS Quarterly,* vol. 34, no. 1, pp. 63-85.

Wang, X., Vitvar, T., Hauswirth, M. & Foxvog, D. 2007, "Building Application Ontologies from Descriptions of Semantic Web Services", *Web Intelligence, IEEE/WIC/ACM International Conference on*, pp. 337.

White, S.A. 2005, "Introduction to BPMN", [Online], . Available from: http://www.bpmn.org/Documents/Introduction_to_BPMN.pdf.

Wiles, T.E. 2002, "Chapter 5.10 - Computers" in *Standard Handbook of Plant Engineering*, ed. R.C. Rosaler, 3rd Edition edn, McGraw-Hill Engineering, Books24x7.

Willcocks, L.P. & Lester, s. 2003, "Chapter 20 - Information Technology and Organizational Performance—Beyond the IT Productivity Paradox" in *Strategic Information Management: Challenges and Strategies in Managing Information Systems,* eds. R.D. Galliers & D.E. Leidner, Third Edition edn, Elsevier Science and Technology Books, Inc, Books24x7.

Xiong, G., Nyberg, T.R., Wang, F. & IEEE/ITSS 2010, "Real-Time Manufacturing Integration and Intelligence Solution Applied in Global Process Industry", , pp. 5.

Young, R., Chungoora, N., Usman, Z., Anjum, N., Gunendran, G., Palmer, C., Harding, J., Case, K. & Cutting-Decelle, A.F. 2010, "An exploration of foundation ontologies and verification methods for manufacturing knowledge sharing", *Workshop on Interoperability for Enterprise Software and Applications (I-ESA Workshop, 2010)*Coventry, UK, April 13.

Young, R.I.M., Gunedran, G., Chungoora, N., Harding, J. & Case, K. 2009, "Enabling Interoperable Manufacturing Knowledge Sharing in PLM", *The 6th International Product Lifecycle Management Conference*, July 6-8.

Young, B., Cutting-Decelle, A., Guerra, D., Gunendran, G., Das, B. & Cochrane, S. 2005, *Sharing Manufacturing Information and Knowledge in Design Decision and Support*, Springer Netherlands.

Young, R.I.M., Gunendran, A.G., Cutting-Decelle, A. & Gruninger, M. 2007, "Manufacturing knowledge sharing in PLM: a progression towards the use of heavy weight ontologies", *International Journal of Production Research,* vol. 45, no. 7, pp. 1505-1519.

Yu-Liang Chi 2010, "Rule-based ontological knowledge base for monitoring partners across supply networks", *Expert Systems with Applications,* vol. 37, no. 2, pp. 1400-7.

Zahedi, F. & Sinha, A.P. 2010, "Ontology design for strategies to metrics mapping", , pp. 7.

Zaremba, M.B. 2003, "Integration and control of intelligence in distributed manufacturing", *Journal of Intelligent Manufacturing,* vol. 14, no. 1, pp. 25-42.

Zhang, J.K., Xu, W. & Ewins, D. 2007, "System Interoperability Study for Healthcare Information Systems with Web Sevices", *Journal of Computer Science,* vol. 3, no. (7), pp. 515-522.

11 Appendix A – Ontology development change log

Element	Change Description
Term Pool	MI Definition Work - Initial creation
Core Term Pool	Subject matter expert brain storm and initial core term list creation
Hierarchy	Grouping terms around core list and structuring into hierarchies
Core Term Pool	Update core term list based on hierarchy core terms
Term Pool	Identify terms from BPMN maps and ad to hierarchy model
Core Term Pool	Review core terms following BPMN work
Term Pool	Provide the core concepts to further subject matter experts for population - added to hierarchies
Hierarchy	Combine hierarchies together to reduce the number of core terms (high number of core terms is counter intuitive) also number of core terms with few defined specialisations
Hierarchy	Error corrections identified during term coding into IODE
Ontology	Identify and address duplication or overlap with MLO and IMKS work
Relationships	Initial relationship definition (between core terms)
Logic	Initial list of logic statements generated
Hierarchy	Following initial logic listing some issues in Hierarchy v6 and Core terms list v5 require updating: 'Response' structure aligned under core concept 'decision' as any response is a part of a decision process. The missing requirement for an approved and unapproved status for data and objects identified as a key requirement for usability. Input and Output data types identified as underutilised as part of the hierarchy, the importance of input and output aspects of many different terms has identified that these should be elivated to core concept/ terms. Team leader term removed as too specific and replaced with 'Leader': a generic elevation of an employee. Specification has been removed as a core concept and made a sub concept of Target as the logic development show that specifications are a thing to be met which is a core characteristic of targets.
Core Term Pool	The relationships initial defined for Specification and System (specifies) and Prediction and Response (anticipates) are effectively re-routed to the core concepts that they are moved under hence Target specifies System and Prediction anticipates Decision. Manufacturing Method has not driven any key logic at this stage so has been removed as a core concept. It can be seen at this stage that the generic model of a system (ie inputs, output, resource and constraints is in the core concept pool. Given the nature of this ontology this is intuitively logical, it could be proposed that the work to date may have been ‘too bottom up’ to identify this basic core model. Having identified this, the ‘system’ prefix which has specialised both of these terms has been removed leaving the core terms “Constraint” and “Resource”.
Relationships	It has become apparent that the Core concept UML model is defining the core relationships. Basic logic will be used to narrow the generalised relationships in line with the UML diagram: Relationships may be used between multiple classes meaning the Relationship Signatures have to be defined at the highest common level and then logic used to constrain the relationship back down to an appropriate level.
Relationships	An error has been idenitified due to the assumption the the MLO heirichy is Top - Type - Instance (decending), however instance and type are both instances of Top hence if something can be either a type or instance it must be declared as Top in the Relationship signature. The core relations have all been reviewed accordingly.

Relationships	The relationships that are required to create the logical rules (IC and Irs) beyond the basic core concept model represent a greater level of specialisation of the ontology, but still more general than domain instances or facts.
Logic	The use of Uniary relationships was considdered for concepts such as object status, where status could be applied as a 'tag' to an object such as 'deleted' or 'obselete', however it was felt that there would be need for logic to control the change of status which means using a Binary relationship 'hasStatus' and createing status objects.
Logic	The use of supTC for some of the data authLevel relationships was considdered to allow any data type or instance to be covered. This has been adopted for some concepts but not data at this point as they should be applied to instances as the controls over data require contextual information such as product type etc to set the controls rather than just the data type.
Logic	Instances and types have bcome intrmingled in the logic, as the understanding of the logic has evolved: the level of absstraction will be used to differentiate ie logic that only uses the types in the core model wil be consisered foundational the primary foundational logic, the logic that refernaces types will be the next level and instance logic the most specialised logic. Logic that refers to both types and instances may have to be split to allow this tiering to take place.
Logic	The hard Ics stating that a person must have operational and technichal authority levels have been downgraded to a Soft IC as Irs have been added the give a person a default based auth level if no level is explicitly set. The IC logic is not that a person should have a auth level declared (explicetly).
Logic	Individual Ics stating that a person and data must have technical and operational authority levels set have lead to a generalised core IC that they should have an authority level (core type) defined.
Ontology	The ontology development has been clarified in 3 tiers: core, level1 and level 2. The KFL files and logic have been restructured accordingly so that the levels can be loaded, tested and developed individualy and incrementaly. This structure makes coding and debugging easier.
Logic	Core and Level 1 logic loaded into IODE v3.4, and sucsesfuly debugged however using test cases identified that the logic was not firing correctly allowing invalid facts to be loaded into the knowledge base. It appears this is IODE issue with the use of supTC. Version 4.2.1 was tested and found to trigger correctly. V3.4 can therefore only be used for basic taxonomic development (this is preferable due to the the use of OMS's being removed in v4.1.2)
Logic	Code fixing:removed use of 'exists' in Irs, as this requires the IR to create instances which is not possible in IODE.
Logic	Next round of checking whether logic applies to types or instances (ongoing and itterative process).
Logic	37IC uses the relationship hasConstraint for Specifications which flagged that Specifications have been declared as a Target rather than a constrraint. The solution options are to create a hasSpecification relationship or realign specifications as a type of constraint. The function of specifications in this domain context are to constrain the system scope and development so to realigning would be appropriate which prevents creating an additional specialised relationship (it is becoming clear that the unnessisary prolliferation of specialised relationships will make the management and maintnence of the logic at that level impractical)

12 Appendix B – Full ontology logic listing

No.	Logic statement	Ontology Level
2	~~Every type of data has some associated type of technical authority level~~ - removed as auth level cannot be set at type level (requires other context info)	
3	~~Every type of data has some associated type of operational authority level~~ removed as auth level cannot be set at type level (requires other context info)	
4	~~Decisions about systems can only be made by people who are superusers or administrators~~ Removed, not relevant to competency qns	
5	~~Non std systems specifications require manager and superuser approval~~ Removed, not relevant to competency qns	
6	Every <sym>Person</sym> <code>?p</code> item must have a <sym>TechAuthorityLevel</sym> <code>?ta</code>.	Level 1
7	Every <sym>Person</sym> <code>?p</code> item must have a <sym>OpAuthorityLevel</sym> <code>?oa</code>.	Level 1
8	If some person <code>?p</code> individual does not have some associated technical authority level individual, then it should default to User.	Level 1
9	If some person <code>?p</code> individual does not have some associated Operational authority level individual, then it should default to Employee	Level 1
10	If a person individual has an op auth level which is either equivalent to or higher than that of a data individual, then the person has access to the data	Level 1
11	If a person individual has an tech auth level which is either equivalent to or higher than that of a data individual, then the person has access to the data.	Level 1
12	~~Having technical authority does not give a person operational authority~~ - removed as does not contribute to the ontologys logical purpose	
13a	<sym>Data</sym> <code>?d</code> should use a <sym>StandardDataStructure</sym> <code>?sds</code>.	Level 1
13b	<sym>Data</sym> <code>?d</code> types should use a <sym>StandardDataStructure</sym> type <code>?sds</code>.	Level 1
14a	Anything <code>?t</code> with an associated individual of <sym>ObsoleteStatus</sym> or <sym>DeletedStatus</sym> cannot be an input to a System individual.	Level 1
14b	Anything <code>?t</code> with an associated type of <sym>ObsoleteStatus</sym> or <sym>DeletedStatus</sym> cannot be an input to a System type.	Level 1
14c	Anything <code>?t</code> with <sym>ObsoleteStatus</sym> or <sym>DeletedStatus</sym> cannot be an input to a System individual.	Level 1
14d	Anything <code>?t</code> with <sym>ObsoleteStatus</sym> or <sym>DeletedStatus</sym> cannot be an input to a System type.	Level 1
15a	If a system <code>?s</code> output <code>?o</code> is associated with an ApprovedStatus individual, then the system input(s) <code>?i</code> must also have the same status	Level 1
15b	If a system <code>?s</code> output <code>?o</code> is associated with an ApprovedStatus type, then the system input(s) <code>?i</code> must also also have the same status type.	Level 1
15c	If a system output <code>?o</code> has ApprovedStatus, then the system input(s) <code>?i</code> must also have ApprovedStatus.	Level 1
16	Any <sym>Data</sym> type affected by a <sym>ChangeDecision</sym> type is of <sym>UnapprovedStatus</sym>.	Level 1
17a	If some data individual does not have some associated status type/individual, then it should default to UnapprovedStatus.	Level 1

17b	If some data type does not have some associated status type/individual, then it should default to UnapprovedStatus.	Level 1
18	If a system input is not approved the system output is not approved	
22	~~System or system type output data must have a traceability item or type of traceability item~~ - Removed as duplicates IC 72	
25	~~For a person to work with a system there must be a HMI~~ -generalised to interopsWith level	
26	~~For two or more systems to work together there must be a system system interface defined -~~ generalised to interopsWith level	
27	If two systems output the same metric their input should be the same.	Level 1
34a	A <sym>Metric</sym> <code>?me</code> should be defined for any <sym>Monitor</sym> <code>?mr</code> to monitor.	Level 1
34b	A <sym>Metric</sym> <code>?me</code> type should be defined for any type of <sym>Monitor</sym> <code>?mr</code> to monitor.	Level 1
35	A <sym>Metric</sym> <code>?me</code> must have a <sym>Target</sym> <code>?t</code> to target.	Level 1
36	A <sym>Monitor</sym> <code>?m</code> must have a defined <sym>Response</sym> <code>?r</code>.	Level 1
37	A system <code>?s</code> must be described by a functional and non functional spec	Level 1
38	~~If a system uses IT there must be an IT spec~~. - Removed as IT type not defined	
39a	Proactive feedback individuals <code>?pf</code> are informed by input data individuals <code>?d</code>	Level 1
40a	Reactive feedback individuals <code>?rf</code> are informed by output data individuals <code>?d</code>	Level 1
41	~~Responses require output data~~ – removed due to conflicts with proactive feedback use of input data	
42	~~Using non std data structures requires superuser authority~~ - Removed as conflicts with other logic	
43	~~Specifications must be approved by people with leaders and superusers authority or higher~~ deleted - not required for interoperability	
44a	A system individual <code>?s</code> must have some form of inputs, outputs, resource and constraints defined.	Level 1
44b	A system type must have some form of inputs, outputs, resource and constraints defined.	Level 1
45	~~System resource and constraints should be specified for a system -~~ removed, combined with the input and output logic	
46a	Target <code>?t</code> individuals have a validity timescale individual <code>?v</code>.	Level 1
46b	Target <code>?t</code> types have a validity timescale type <code>?t</code>	Level 1
47a	Data <code>?d</code> individuals have a validity timescale individual.	Level 1
47b	Data types <code>?d</code> have a validity timescale type<code>?v</code>.	Level 1
52	~~Data should not be stored in 2 archiving systems~~ - Removed, conflicts with logic 53	
54	Data individuals should have a data retention category type or instance.	Level 1
55	Data individuals should have an export control category type or individual.	Level 1
56	Data individuals should have a security control category type or individual.	Level 1
57	~~systems must have a security, export control and data retention category individual or type~~ - removed as Irs infer a default value	
58a	If some system individual does not have some associated security category type/individual, then it should default to Public.	Level 1

58b	If some system type does not have some associated security category type/individual, then it should default to Public.	Level 1
59a	If some system individual does not have some associated export control category type/individual, then it should default to ExportControlled.	Level 1
59b	If some system type does not have some associated export control category type/individual, then it should default to ExportControlled.	Level 1
60a	If some system individual does not have some associated retention category type/individual, then it should default to NonMandatoryRetention.	Level 1
60b	If some system type does not have some associated retention category type/individual, then it should default to NonMandatoryRetention.	Level 1
61	Systems <code>?s</code> that have security category of public can only input or output public category data.	Level 1
62	Systems <code>?s</code> that have security category of private can only input <code>?i</code> or output <code>?o</code> public or private category data.	Level 1
63	Systems <code>?s</code> that have security category of secret can only input <code>?i</code> or output <code>?o</code> public, private or secret category data.	Level 1
64	Systems <code>?s</code> that have security category of top secret can input <code>?i</code> or output <code>?o</code> public, private, secret or top secret category data.	Level 1
65	Systems <code>?s</code> with an export control category of non export control can only input <code>?i</code> or output <code>?o</code> non export controlled data.	Level 1
66a	Systems <code>?s</code> with a retention category of non mandatory retention can only input <code>?i</code> o non mandatory retention data. If mandatory data is input to a non mandatory retention system is must be output as well so it exists in another system.	Level 1
68	~~If a system outputs data to another system it is an input for that other system~~ - removed as non value added	
69	A system <code>?s</code> should have a functional specification <code>?fs</code> that describes metrics <code>?m</code> or metric targets <code>?mt</code>, if multiple functional specs exist not all may contain metrics.	Level 1
70	~~Systems that share the same input, outputs resources or constraints are parallel associated~~ - removed as non value added	
71	~~If a system outputs data that is not input into it, it is the data source for that data~~ - removed as can be discovered by query eg search for something that is and output from a system but not an input to the same system	
72a	If an entity is a data output <code>?x</code> individual then it must have a traceability item <code>?y</code> individual defined.	Level 1
72b	If an entity is a data output type then it must have a traceability item type defined.	Level 1
73	Traceability items <code>?x</code> <code>?y</code> must be consistent between interoperating systems <code>?a</code> <code>?b</code>.	Level 1
75	tech auth level increments from user - superuser - administrator	
76	op auth level increments from employee, leader, manager, executive	
77a	If some data individual does not have some associated retention category type/individual, then it should default to NonMandatoryRetention.	Level 1
77b	If some data type does not have some associated retention category type/individual, then it should default to NonMandatoryRetention	Level 1
78a	If some data individual does not have some associated export control category type/individual, then it should default to ExportControlled.	Level 1
78b	If some data type does not have some associated export control category	Level 1

	type/individual, then it should default to ExportControlled.	
79a	If some data individual does not have some associated security category type/individual, then it should default to Private	Level 1
79b	If some data type does not have some associated security category type/individual, then it should default to Private	Level 1
80	If a system input is not approved the system output is not approved.	Level 1
81	~~If 2 things interoperate they must share some type of interface~~ - removed as logic does not hold true	
82	~~Every data individual has some associated operational authority level individual~~. - removed as logic does not hold true	
83	~~Every data individual has some associated technical authority level individual~~.- removed as logic does not hold true	
84	~~Every type of data has some associated type of authority level~~ - removed as logic does not hold true	
85	An instance of <sym>Prediction</sym> anticipates the <sym>Output</sym> instance or <sym>Decision</sym> instance	Core
86	A <sym>Prediction</sym> type anticipates the <sym>Output</sym> type or <sym>Decision</sym> type	Core
87	<sym>Analysis</sym><code>?x</code> informs <sym>Decision</sym> <code>?y</code>	Core
88	<sym>Analysis</sym> <code>?x</code>or some type of Analysis should inform some <sym>Decision</sym> type <code>?y</code>.	Core
89a	If two system individuals <code>?x</code> <code>?y</code>interoperate with each other, then they require some associated interface individual.	Core
89b	If a system individual <code>?x</code> interoperate with a person individual <code>?y</code>, then they require some associated interface individual.	Core
89c	If two system types <code>?x</code> <code>?y</code> interoperate with each other, then they require some associated interface type	Core
89d	If a system type interoperates with a person type, then both system and person types require some associated interface type.	Core
90	If two arguments <code>?t1</code> <code>?t2</code> interoperate with each other they must have some associated interface <code>?i</code> in common.	Core
91	If the output of one system is the input of another system the systems interoperate	Level 2
92	If two things have the same interface they interoperate	Level 2
93	If something analyses something else they interoperate	Level 2
94	If something informs something else they interoperate	Level 2
95	If something may access something else they interoperate	Level 2

13 Appendix C – Full fact base listing

```
;=========================================================
;Declare People
;=========================================================

(Operator MI.wax_injection_operator)
(Supervisor MI.wax_production_leader)
(Supervisor MI.foundry_manufacturing_engineering_manager)
(Supervisor MI.foundry_manufacturing_manager)
(ManufacturingEngineer MI.wax_process_engineer)
(ManufacturingEngineer MI.casting_part_owner)
(MaintenanceEngineer MI.foundry_maintenence_engineer)
(MaterialController MI.foundry_material_requirements_planning_controller)
(ManufacturingEngineer MI.CAPP_superuser)
(ManufacturingEngineer MI.CAPP_keyuser)

(informs MI.wax_injection_operator MI.wax_production_leader)
(informs MI.CAPP_keyuser MI.CAPP_superuser)

(hasTechAuthorityLevel MI.wax_injection_operator MI.User)
(hasTechAuthorityLevel MI.wax_production_leader MI.User)
(hasTechAuthorityLevel MI.foundry_manufacturing_engineering_manager
MI.Administrator)
(hasTechAuthorityLevel MI.foundry_manufacturing_manager MI.User)
(hasTechAuthorityLevel MI.wax_process_engineer MI.Superuser)
(hasTechAuthorityLevel MI.casting_part_owner MI.Superuser)
(hasTechAuthorityLevel MI.foundry_maintenence_engineer MI.Superuser)
(hasTechAuthorityLevel MI.foundry_material_requirements_planning_controller
MI.User)
(hasTechAuthorityLevel MI.CAPP_superuser MI.Superuser)
(hasTechAuthorityLevel MI.CAPP_keyuser MI.User)

(hasOpAuthorityLevel MI.wax_injection_operator MI.Employee)
(hasOpAuthorityLevel MI.wax_production_leader MI.Leader)
(hasOpAuthorityLevel MI.foundry_manufacturing_engineering_manager MI.Manager)
(hasOpAuthorityLevel MI.foundry_manufacturing_manager MI.Manager)
(hasOpAuthorityLevel MI.wax_process_engineer MI.Employee)
(hasOpAuthorityLevel MI.casting_part_owner MI.Employee)
(hasOpAuthorityLevel MI.foundry_maintenence_engineer MI.Employee)
(hasOpAuthorityLevel MI.foundry_material_requirements_planning_controller
MI.Employee)
(hasOpAuthorityLevel MI.CAPP_superuser MI.Employee)
(hasOpAuthorityLevel MI.CAPP_keyuser MI.Employee)


(StandardDataStructure MI.XML)
(StandardDataStructure MI.profinet_signal)
```

(StandardDataStructure MI.SQL_table)
(StandardDataStructure MI.NX6_CAD_format)
(StandardDataStructure MI.std_document_template)
(ValidityTimescale MI.life_of_type)
(ValidityTimescale MI.life_of_order)
(ValidityTimescale MI.life_of_contract)

;===
;Declare Systems
;===

(System MI.CAPP_system)

(Input MI.core_specification)
(Input MI.method_of_manufacture)
(Input MI.manufacturing_bill_of_material)
(Input MI.bill_of_plant)
(Input MI.geometry_models)
(Input MI.bill_of_process)
(Input MI.product_history)

(Output MI.wax_process_instruction)
(Output MI.wax_process_parameter)
(Output MI.core_specification)
(Output MI.core_chaplet_inspection_spec)
(Output MI.manufacturing_technical_package)

(Resource MI.PLM_server)
(Resource MI.PLPS_process)
(Resource MI.it_infrastructure)
(Resource MI.ERP_system)
(Resource MI.ERP_bridge)
(Resource MI.PLM_database)
(Resource MI.production_schedule)
(Resource MI.casting_part_owner)
(Resource MI.wax_process_engineer)
(Resource MI.foundry_manufacturing_engineering_manager)
(Resource CAPP_superuser)
(Resource CAPP_keyuser)

(Constraint MI.PLPS_process)
(Constraint MI.CAPP_data_storage_capacity)
(Constraint MI.RR_quality_management_system)
(Constraint MI.CAPP_data_transfer_capacity)
(Constraint MI.CAPP_data_processing_capacity)
(Constraint MI.product_definition)
(Constraint MI.EASA_requirements)
(Constraint MI.technical_package_std)

(Constraint MI.business_IT_infrastructure_availability)

(MI.FunctionalSpecification MI.capp_functional_specification)
(MI.FunctionalSpecification MI.RR_quality_management_system)
(MI.NonFunctionalSpecification MI.capp_performance_requirements_spec)
(Metric MI.CAPP_system_response_time)
(MetricTarget MI.CAPP_system_response_time_target)
(hasTarget MI.CAPP_system_response_time MI.CAPP_system_response_time_target)
(specifies MI.capp_functional_specification MI.CAPP_system_response_time)
(specifies MI.capp_functional_specification MI.CAPP_system_response_time_target)

(hasConstraint MI.CAPP_system MI.capp_performance_requirements_spec)
(hasConstraint MI.CAPP_system MI.capp_functional_specification)
(hasConstraint MI.CAPP_system MI.RR_quality_management_system)
(hasConstraint MI.CAPP_system MI.PLPS_process)
(hasConstraint MI.CAPP_system MI.CAPP_data_storage_capacity)
(hasConstraint MI.CAPP_system MI.RR_quality_management_system)
(hasConstraint MI.CAPP_system MI.CAPP_data_transfer_capacity)
(hasConstraint MI.CAPP_system MI.CAPP_data_processing_capacity)
(hasConstraint MI.CAPP_system MI.product_definition)
(hasConstraint MI.CAPP_system MI.EASA_requirements)
(hasConstraint MI.CAPP_system MI.technical_package_std)
(hasConstraint MI.CAPP_system MI.business_IT_infrastructure_availability)

(hasResource MI.CAPP_system MI.PLM_server)
(hasResource MI.CAPP_system MI.ERP_system)
(hasResource MI.CAPP_system MI.it_infrastructure)
(hasResource MI.CAPP_system MI.ERP_bridge)
(hasResource MI.CAPP_system MI.PLPS_process)
(hasResource MI.CAPP_system MI.PLM_database)
(hasResource MI.CAPP_system MI.production_schedule)
(hasResource MI.CAPP_system CAPP_superuser)
(hasResource MI.CAPP_system CAPP_keyuser)
(hasResource MI.CAPP_system MI.casting_part_owner)
(hasResource MI.CAPP_system MI.wax_process_engineer)
(hasResource MI.CAPP_system MI.foundry_manufacturing_engineering_manager)

(hasInput MI.CAPP_system MI.core_specification)
(hasInput MI.CAPP_system MI.method_of_manufacture)
(hasInput MI.CAPP_system MI.manufacturing_bill_of_material)
(hasInput MI.CAPP_system MI.bill_of_plant)
(hasInput MI.CAPP_system MI.geometry_models)
(hasInput MI.CAPP_system MI.bill_of_process)
(hasInput MI.CAPP_system MI.product_history)

(hasOutput MI.CAPP_system MI.core_specification)
(hasOutput MI.CAPP_system MI.core_chaplet_inspection_spec)
(hasOutput MI.CAPP_system MI.wax_process_instruction)

(hasOutput MI.CAPP_system MI.wax_process_parameter)
(hasOutput MI.CAPP_system MI.manufacturing_technical_package)

(hasSecurityCategory MI.CAPP_system MI.Private)
(hasExportControlCategory MI.CAPP_system MI.ExportControlled)
(hasRetentionCategory MI.CAPP_system MI.MandatoryRetention)
(VisualSpecification MI.core_chaplet_inspection_spec)
(ProcessSpecification MI.wax_process_instruction)
(ProcessSpecification MI.wax_process_parameter)
(ProductSpecification MI.product_definition)
(MI.SystemResponseTimeMetric MI.CAPP_system_response_time)
(hasTimescale MI.CAPP_system_response_time_target MI.life_of_order)
(hasRetentionCategory MI.core_specification MI.MandatoryRetention)
(hasRetentionCategory MI.wax_process_parameter MI.MandatoryRetention)
(ProductSpecification MI.core_specification)
(MI.SysSysInterface MI.ERP_bridge)
(MI.hasInterface MI.CAPP_system MI.ERP_bridge)
(MI.hasInterface MI.CAPP_system MI.MES_CAPP_Bridge)

(MI.hasSecurityCategory MI.method_of_manufacture MI.Private)
(MI.hasSecurityCategory MI.manufacturing_bill_of_material MI.Private)
(MI.hasSecurityCategory MI.product_history MI.Private)
(MI.hasSecurityCategory MI.bill_of_process MI.Private)
(MI.hasSecurityCategory MI.geometry_models MI.Private)
(MI.hasSecurityCategory MI.manufacturing_technical_package MI.Private)
(MI.hasSecurityCategory MI.bill_of_plant MI.Private)
(MI.hasSecurityCategory MI.core_specification MI.Private)
(MI.hasSecurityCategory MI.wax_process_instruction MI.Private)
(MI.hasSecurityCategory MI.wax_process_parameter MI.Private)
(MI.hasSecurityCategory MI.core_chaplet_inspection_spec MI.Private)
(hasTimescale MI.CAPP_system_response_time_target MI.life_of_contract)

(ProductSpecification MI.core_specification)
(VisualData MI.core_specification)
(DimensionalData MI.core_specification)
(ProcessData MI.method_of_manufacture)
(ResourceData MI.manufacturing_bill_of_material)
(ResourceData MI.bill_of_plant)
(ProductData MI.geometry_models)
(ResourceData MI.geometry_models)
(ProcessData MI.bill_of_process)
(ProductionHistory MI.product_history)
(ProductData MI.product_history)
(ProcessData MI.wax_process_instruction)
(DatacardData MI.wax_process_parameter)

(ProductSpecification MI.core_chaplet_inspection_spec)
(VisualData MI.core_chaplet_inspection_spec)

(ProcessData MI.manufacturing_technical_package)
(ProductData MI.manufacturing_technical_package)
(ProcessSpecification MI.PLPS_process)
(OrderData MI.production_schedule)

(SystemStorageCapacityMetric MI.CAPP_data_storage_capacity)
(MetricTarget MI.CAPP_data_storage_capacity_target)
(hasTarget MI.CAPP_data_storage_capacity MI.CAPP_data_storage_capacity_target)
(specifies MI.capp_functional_specification MI.CAPP_data_storage_capacity_target)
(specifies MI.capp_functional_specification MI.CAPP_data_storage_capacity)

(SystemFlowCapacityMetric MI.CAPP_data_transfer_capacity)
(MetricTarget MI.CAPP_data_transfer_capacity_target)
(hasTarget MI.CAPP_data_transfer_capacity MI.CAPP_data_transfer_capacity_target)
(specifies MI.capp_functional_specification MI.CAPP_data_transfer_capacity_target)
(specifies MI.capp_functional_specification MI.CAPP_data_transfer_capacity)

(SystemFlowCapacityMetric MI.CAPP_data_processing_capacity)
(MetricTarget MI.CAPP_data_processing_capacity_target)
(hasTarget MI.CAPP_data_processing_capacity
MI.CAPP_data_processing_capacity_target)
(specifies MI.capp_functional_specification
MI.CAPP_data_processing_capacity_target)
(specifies MI.capp_functional_specification MI.CAPP_data_processing_capacity)

(ProductSpecification MI.product_definition)
(ProcessSpecification MI.EASA_requirements)
(ProcessSpecification MI.technical_package_std)
(ITSpecification MI.business_IT_infrastructure_availability)

(TraceabilityData MI.coreID)
(TraceabilityData MI.partID)
(TraceabilityData MI.operator_id)
(TraceabilityData MI.wax_robot_start_timestamp)
(TraceabilityData MI.wax_process_start_timestamp)
(TraceabilityData MI.wax_machine_id)
(TraceabilityData MI.tool_id)
(TraceabilityData MI.wax_process_end_timestamp)
(TraceabilityData MI.unique_identifier)

(TraceabilityItem MI.coreID)
(TraceabilityItem MI.partID)
(TraceabilityItem MI.operator_id)
(TraceabilityItem MI.wax_robot_start_timestamp)
(TraceabilityItem MI.wax_process_start_timestamp)
(TraceabilityItem MI.wax_machine_id)
(TraceabilityItem MI.tool_id)
(TraceabilityItem MI.wax_process_end_timestamp)

(TraceabilityItem MI.unique_identifier)

(hasTraceItem MI.CAPP_data_storage_capacity_target MI.unique_identifier)
(hasTraceItem MI.CAPP_data_transfer_capacity_target MI.unique_identifier)
(hasTraceItem MI.CAPP_data_processing_capacity_target MI.unique_identifier)
(hasTraceItem MI.core_specification MI.unique_identifier)
(hasTraceItem MI.method_of_manufacture MI.unique_identifier)
(hasTraceItem MI.manufacturing_bill_of_material MI.unique_identifier)
(hasTraceItem MI.bill_of_plant MI.unique_identifier)
(hasTraceItem MI.geometry_models MI.unique_identifier)
(hasTraceItem MI.bill_of_process MI.unique_identifier)
(hasTraceItem MI.production_schedule MI.unique_identifier)
(hasTraceItem MI.manufacturing_technical_package MI.unique_identifier)
(hasTraceItem MI.manufacturing_technical_package MI.partID)
(hasTraceItem MI.core_chaplet_inspection_spec MI.unique_identifier)
(hasTraceItem MI.core_chaplet_inspection_spec MI.partID)
(hasTraceItem MI.core_chaplet_inspection_spec MI.coreID)
(hasTraceItem MI.wax_process_parameter MI.unique_identifier)
(hasTraceItem MI.wax_process_parameter MI.partID)
(hasTraceItem MI.wax_process_instruction MI.partID)
(hasTraceItem MI.wax_process_instruction MI.unique_identifier)
(hasTraceItem MI.product_history MI.unique_identifier)
(hasTraceItem MI.product_history MI.tool_id)
(hasTraceItem MI.product_history MI.wax_machine_id)
(hasTraceItem MI.product_history MI.wax_process_start_timestamp)
(hasTraceItem MI.product_history MI.wax_process_end_timestamp)
(hasTraceItem MI.product_history MI.wax_robot_start_timestamp)
(hasTraceItem MI.product_history MI.operator_id)
(hasTraceItem MI.product_history MI.partID)
(hasTraceItem MI.product_history MI.coreID)
(hasTraceItem MI.coreID MI.unique_identifier)
(hasTraceItem MI.partID MI.unique_identifier)
(hasTraceItem MI.operator_id MI.unique_identifier)
(hasTraceItem MI.wax_robot_start_timestamp MI.unique_identifier)
(hasTraceItem MI.wax_process_start_timestamp MI.unique_identifier)
(hasTraceItem MI.wax_machine_id MI.unique_identifier)
(hasTraceItem MI.tool_id MI.unique_identifier)
(hasTraceItem MI.wax_process_end_timestamp MI.unique_identifier)
(hasTraceItem MI.unique_identifier MI.unique_identifier)

(hasTimescale MI.coreID MI.life_of_type)
(hasTimescale MI.partID MI.life_of_type)
(hasTimescale MI.operator_id MI.life_of_type)
(hasTimescale MI.wax_robot_start_timestamp MI.life_of_type)
(hasTimescale MI.wax_process_end_timestamp MI.life_of_type)
(hasTimescale MI.wax_process_start_timestamp MI.life_of_type)
(hasTimescale MI.wax_machine_id MI.life_of_type)
(hasTimescale MI.tool_id MI.life_of_type)

(hasTimescale MI.unique_identifier MI.life_of_type)
(hasTimescale MI.CAPP_data_storage_capacity_target MI.life_of_contract)
(hasTimescale MI.CAPP_data_transfer_capacity_target MI.life_of_contract)
(hasTimescale MI.CAPP_data_processing_capacity_target MI.life_of_contract)
(hasTimescale MI.core_specification MI.life_of_type)
(hasTimescale MI.method_of_manufacture MI.life_of_order)
(hasTimescale MI.manufacturing_bill_of_material MI.life_of_order)
(hasTimescale MI.bill_of_plant MI.life_of_order)
(hasTimescale MI.geometry_models MI.life_of_type)
(hasTimescale MI.bill_of_process MI.life_of_order)
(hasTimescale MI.production_schedule MI.life_of_order)
(hasTimescale MI.manufacturing_technical_package MI.life_of_type)
(hasTimescale MI.core_chaplet_inspection_spec MI.life_of_type)
(hasTimescale MI.wax_process_parameter MI.life_of_order)
(hasTimescale MI.wax_process_instruction MI.life_of_order)
(hasTimescale MI.product_history MI.life_of_type)

(hasRetentionCategory MI.core_specification MI.MandatoryRetention)
(hasRetentionCategory MI.method_of_manufacture MI.MandatoryRetention)
(hasRetentionCategory MI.manufacturing_bill_of_material MI.MandatoryRetention)
(hasRetentionCategory MI.bill_of_plant MI.MandatoryRetention)
(hasRetentionCategory MI.geometry_models MI.MandatoryRetention)
(hasRetentionCategory MI.bill_of_process MI.MandatoryRetention)
(hasRetentionCategory MI.production_schedule MI.NonMandatoryRetention)
(hasRetentionCategory MI.manufacturing_technical_package MI.MandatoryRetention)
(hasRetentionCategory MI.core_chaplet_inspection_spec MI.MandatoryRetention)
(hasRetentionCategory MI.wax_process_parameter MI.NonMandatoryRetention)
(hasRetentionCategory MI.wax_process_instruction MI.MandatoryRetention)
(hasRetentionCategory MI.product_history MI.MandatoryRetention)
(hasRetentionCategory MI.coreID MI.MandatoryRetention)
(hasRetentionCategory MI.partID MI.MandatoryRetention)
(hasRetentionCategory MI.operator_id MI.MandatoryRetention)
(hasRetentionCategory MI.wax_robot_start_timestamp MI.NonMandatoryRetention)
(hasRetentionCategory MI.wax_process_end_timestamp MI.MandatoryRetention)
(hasRetentionCategory MI.wax_process_start_timestamp MI.MandatoryRetention)
(hasRetentionCategory MI.wax_machine_id MI.MandatoryRetention)
(hasRetentionCategory MI.tool_id MI.MandatoryRetention)
(hasRetentionCategory MI.unique_identifier MI.MandatoryRetention)

(hasDataStructure MI.core_specification MI.std_document_template)
(hasDataStructure MI.method_of_manufacture MI.SQL_table)
(hasDataStructure MI.method_of_manufacture MI.XML)
(hasDataStructure MI.manufacturing_bill_of_material MI.SQL_table)
(hasDataStructure MI.manufacturing_bill_of_material MI.XML)
(hasDataStructure MI.bill_of_plant MI.SQL_table)
(hasDataStructure MI.bill_of_plant MI.XML)
(hasDataStructure MI.geometry_models MI.NX6_CAD_format)
(hasDataStructure MI.bill_of_process MI.SQL_table)

(hasDataStructure MI.bill_of_process MI.XML)
(hasDataStructure MI.production_schedule MI.SQL_table)
(hasDataStructure MI.manufacturing_technical_package MI.std_document_template)
(hasDataStructure MI.core_chaplet_inspection_spec MI.std_document_template)
(hasDataStructure MI.wax_process_parameter MI.SQL_table)
(hasDataStructure MI.wax_process_parameter MI.XML)
(hasDataStructure MI.wax_process_instruction MI.std_document_template)
(hasDataStructure MI.product_history MI.SQL_table)
(hasDataStructure MI.product_history MI.XML)
(hasDataStructure MI.coreID MI.XML)
(hasDataStructure MI.partID MI.XML)
(hasDataStructure MI.operator_id MI.XML)
(hasDataStructure MI.wax_robot_start_timestamp MI.XML)
(hasDataStructure MI.wax_process_end_timestamp MI.XML)
(hasDataStructure MI.wax_process_start_timestamp MI.XML)
(hasDataStructure MI.wax_machine_id MI.XML)
(hasDataStructure MI.tool_id MI.XML)
(hasDataStructure MI.unique_identifier MI.XML)
(hasDataStructure MI.coreID MI.SQL_table)
(hasDataStructure MI.partID MI.SQL_table)
(hasDataStructure MI.operator_id MI.SQL_table)
(hasDataStructure MI.wax_robot_start_timestamp MI.SQL_table)
(hasDataStructure MI.wax_process_end_timestamp MI.SQL_table)
(hasDataStructure MI.wax_process_start_timestamp MI.SQL_table)
(hasDataStructure MI.wax_machine_id MI.SQL_table)
(hasDataStructure MI.tool_id MI.SQL_table)
(hasDataStructure MI.unique_identifier MI.SQL_table)

;===

(System MI.MES_system)

(Input MI.core_specification)
(Input MI.wax_process_instruction)
(Input MI.wax_process_parameter)
(Input MI.core_chaplet_inspection_spec)
(Input MI.tool_id)
(Input MI.wax_product_history)
(Input MI.wax_process_history)
(Input MI.wax_operator_feedback)

(Output MI.core_specification)
(Output MI.core_chaplet_inspection_spec)
(Output MI.wax_process_instruction)
(Output MI.wax_process_parameter)
(Output MI.product_history)
(Output MI.wax_MES_report)
(Output MI.wax_operator_feedback)

(Output MI.wax_product_history)
(Output MI.tool_id)
(Output MI.wax_process_history)

(Resource MI.ERP_bridge)
(Resource MI.manufacturing_technical_package)
(Resource MI.casting_part_owner)
(Resource MI.wax_process_engineer)
(Resource MI.wax_injection_operator)
(Resource MI.production_plan)
(Resource MI.method_of_manufacture)
(Resource MI.MES_report_template)
(Resource MI.MES_HMI)
(Resource MI.production_schedule)
(Resource MI.IT_infrastructure)
(Resource MI.foundry_material_requirements_planning_controller)

(Constraint MI.MES_data_storage_capacity)
(Constraint MI.RR_quality_management_system)
(Constraint MI.MES_data_transfer_capacity)
(Constraint MI.MES_data_processing_capacity)
(Constraint MI.EASA_requirements)
(Constraint MI.business_IT_infrastructure_availability)

(MI.FunctionalSpecification MI.MES_system_functional_spec)
(MI.NonFunctionalSpecification MI.MES_system_nonfunctional_spec)

(Metric MI.MES_availability_metric)
(Metric MI.MES_response_time_metric)
(MetricTarget MI.MES_availability_metric_target)
(MetricTarget MI.MES_response_time_metric_target)
(hasTarget MI.MES_availability_metric MI.MES_availability_metric_target)
(hasTarget MI.MES_response_time_metric MI.MES_response_time_metric_target)
(specifies MI.MES_system_functional_spec MI.MES_availability_metric)
(specifies MI.MES_system_functional_spec MI.MES_response_time_metric)
(specifies MI.MES_system_functional_spec MI.MES_availability_metric_target)
(specifies MI.MES_system_functional_spec MI.MES_response_time_metric_target)

(hasConstraint MI.MES_system MI.MES_data_storage_capacity)
(hasConstraint MI.MES_system MI.RR_quality_management_system)
(hasConstraint MI.MES_system MI.MES_data_transfer_capacity)
(hasConstraint MI.MES_system MI.MES_data_processing_capacity)
(hasConstraint MI.MES_system MI.EASA_requirements)
(hasConstraint MI.MES_system MI.business_IT_infrastructure_availability)
(hasConstraint MI.MES_system MI.MES_system_functional_spec)
(hasConstraint MI.MES_system MI.MES_system_nonfunctional_spec)

(hasResource MI.MES_system MI.ERP_bridge)
(hasResource MI.MES_system MI.manufacturing_technical_package)
(hasResource MI.CAPP_system MI.casting_part_owner)
(hasResource MI.CAPP_system MI.wax_process_engineer)
(hasResource MI.CAPP_system MI.wax_injection_operator)
(hasResource MI.MES_system MI.production_plan)
(hasResource MI.MES_system MI.method_of_manufacture)
(hasResource MI.MES_system MI.MES_report_template)
(hasResource MI.MES_system MI.MES_HMI)
(hasResource MI.MES_system MI.production_schedule)
(hasResource MI.MES_system MI.IT_infrastructure)
(hasResource MI.MES_system
MI.foundry_material_requirements_planning_controller)

(hasInput MI.MES_system MI.core_specification)
(hasInput MI.MES_system MI.wax_process_instruction)
(hasInput MI.MES_system MI.wax_process_parameter)
(hasInput MI.MES_system MI.core_chaplet_inspection_spec)
(hasInput MI.MES_system MI.tool_id)
(hasInput MI.MES_system MI.wax_product_history)
(hasInput MI.MES_system MI.wax_process_history)
(hasInput MI.MES_system MI.wax_operator_feedback)

(hasOutput MI.MES_system MI.core_specification)
(hasOutput MI.MES_system MI.core_chaplet_inspection_spec)
(hasOutput MI.MES_system MI.wax_process_instruction)
(hasOutput MI.MES_system MI.wax_process_parameter)
(hasOutput MI.MES_system MI.product_history)
(hasOutput MI.MES_system MI.wax_MES_report)
(hasOutput MI.MES_system MI.wax_operator_feedback)
(hasOutput MI.MES_system MI.wax_product_history)
(hasOutput MI.MES_system MI.tool_id)
(hasOutput MI.MES_system MI.wax_process_history)

(Report MI.wax_MES_report)

(hasSecurityCategory MI.MES_system MI.Private)
(hasExportControlCategory MI.MES_system MI.ExportControlled)
(hasRetentionCategory MI.MES_system MI.NonMandatoryRetention)

(SystemPerformanceMetric MI.MES_availability_metric)
(hasTimescale MI.MES_availability_metric_target MI.life_of_order)
(hasTimescale MES_response_time_metric_target MI.life_of_order)
(ProductSpecification MI.core_specification)
(MI.SysSysInterface MI.MES_CAPP_Bridge)
(MI.SysSysInterface MI.MES_Middleware_Bridge)
(MI.hasInterface MI.MES_system MI.MES_HMI)
(MI.hasInterface MI.MES_system MI.MES_CAPP_Bridge)

(MI.hasInterface MI.MES_system MI.MES_Middleware_Bridge)
(HMI MI.MES_HMI)
(hasInterface MI.MES_system MI.ERP_bridge)
(hasTimescale MI.production_plan MI.life_of_order)
(hasTimescale MI.MES_response_time_metric_target MI.life_of_contract)

(Visualisation MI.MES_HMI)

(visualises MI.MES_HMI MI.core_specification)
(visualises MI.MES_HMI MI.wax_process_instruction)
(visualises MI.MES_HMI MI.wax_product_history)
(visualises MI.MES_HMI MI.core_chaplet_inspection_spec)
(visualises MI.MES_HMI MI.wax_operator_feedback)
(visualises MI.MES_HMI MI.wax_MES_report)
(visualises MI.MES_HMI MI.manufacturing_technical_package)

(Plan MI.production_plan)
(Plan MI.production_schedule)

(MI.hasSecurityCategory MI.tool_id MI.Private)
(MI.hasSecurityCategory MI.wax_product_history MI.Private)
(MI.hasSecurityCategory MI.wax_process_history MI.Private)
(MI.hasSecurityCategory MI.wax_operator_feedback MI.Private)
(MI.hasSecurityCategory MI.wax_MES_report MI.Private)

;==

(System MI.part_tracking_system)

(Input MI.stationID)
(Input MI.partID_reader_signal)

(Output MI.coreID)
(Output MI.wax_product_history)
(Output MI.wax_process_history)
(Output MI.wax_WIP_data)

(Resource MI.coreID)
(Resource MI.partID)
(Resource MI.wax_cell_layout)
(Resource MI.WIP_database)

(Constraint MI.partID_legibility)
(Constraint MI.part_tracking_system_range)
(Constraint MI.part_position_refresh_rate)

(ResourceData MI.stationID)
(ProductData MI.partID_reader_signal)
(ProcessData MI.partID_reader_signal)
(WIPInventoryData MI.wax_WIP_data)

(NonFunctionalSpecification MI.part_tracking_system_non_func_spec)
(FunctionalSpecification MI.part_tracking_system_func_spec)

(Metric MI.part_position_refresh_rate)
(MetricTarget MI.part_position_refresh_rate_target)
(hasTarget MI.part_position_refresh_rate MI.part_position_refresh_rate_target)
(specifies MI.part_tracking_system_func_spec MI.part_position_refresh_rate)
(specifies MI.part_tracking_system_func_spec MI.part_position_refresh_rate_target)

(hasTimescale MI.stationID MI.life_of_order)
(hasTimescale MI.partID_reader_signal MI.life_of_order)
(hasTimescale MI.wax_WIP_data MI.life_of_order)
(hasTimescale MI.wax_cell_layout MI.life_of_order)
(hasTimescale MI.WIP_database MI.life_of_contract)
(hasTimescale MI.part_position_refresh_rate MI.life_of_contract)
(hasTimescale MI.part_position_refresh_rate_target MI.life_of_contract)
(hasTimescale MI.wax_TAKT MI.life_of_contract)
(hasTimescale MI.wax_throughput MI.life_of_contract)
(hasTimescale MI.wax_WIP_level MI.life_of_contract)
(hasTimescale MI.wax_WIP_level_target MI.life_of_contract)
(hasTimescale MI.wax_throughput_target MI.life_of_contract)
(hasTimescale MI.wax_TAKT_target MI.life_of_contract)

(MI.hasSecurityCategory MI.stationID MI.Private)
(MI.hasSecurityCategory MI.partID_reader_signal MI.Private)
(MI.hasSecurityCategory MI.wax_WIP_data MI.Private)
(MI.hasSecurityCategory MI.wax_cell_layout MI.Private)
(MI.hasSecurityCategory MI.WIP_database MI.Private)
(MI.hasSecurityCategory MI.part_position_refresh_rate MI.Private)

(Metric MI.wax_WIP_level)
(Metric MI.wax_throughput)
(Metric MI.wax_TAKT)
(MetricTarget MI.wax_WIP_level_target)
(MetricTarget MI.wax_throughput_target)
(MetricTarget MI.wax_TAKT_target)
(hasTarget MI.wax_WIP_level MI.wax_WIP_level_target)
(hasTarget MI.wax_throughput MI.wax_throughput_target)
(hasTarget MI.wax_TAKT MI.wax_TAKT_target)
(specifies MI.part_tracking_system_func_spec MI.wax_WIP_level_target)
(specifies MI.part_tracking_system_func_spec MI.wax_throughput_target)
(specifies MI.part_tracking_system_func_spec MI.wax_TAKT_target)

(hasInput MI.part_tracking_system MI.stationID)
(hasInput MI.part_tracking_system MI.partID_reader_signal)

(hasOutput MI.part_tracking_system MI.coreID)
(hasOutput MI.part_tracking_system MI.wax_product_history)
(hasOutput MI.part_tracking_system MI.wax_process_history)

(hasResource MI.part_tracking_system MI.coreID)
(hasResource MI.part_tracking_system MI.partID)
(hasResource MI.part_tracking_system MI.wax_cell_layout)
(hasResource MI.part_tracking_system MI.WIP_database)

(hasConstraint MI.part_tracking_system MI.partID_legibility)
(hasConstraint MI.part_tracking_system MI.part_tracking_system_range)
(hasConstraint MI.part_tracking_system MI.part_position_refresh_rate)
(hasConstraint MI.part_tracking_system MI.part_tracking_system_func_spec)
(hasConstraint MI.part_tracking_system MI.part_tracking_system_non_func_spec)

(hasSecurityCategory MI.part_tracking_system MI.Private)
(hasExportControlCategory MI.part_tracking_system MI.MI.NonExportControlled)
(hasRetentionCategory MI.part_tracking_system MI.NonMandetoryRetention)

(MI.hasInterface MI.part_tracking_system MI.MES_Middleware_Bridge)

(hasDataStructure MI.partID_reader_signal MI.profinet_signal)
(hasDataStructure MI.stationID MI.SQL_table)
(hasDataStructure MI.wax_WIP_data MI.SQL_table)

(hasTraceItem MI.stationID MI.unique_identifier)
(hasTraceItem MI.partID_reader_signal MI.unique_identifier)
(hasTraceItem MI.partID_reader_signal MI.wax_robot_start_timestamp)
(hasTraceItem MI.partID_reader_signal MI.wax_process_start_timestamp)
(hasTraceItem MI.wax_WIP_data MI.partID)
(hasTraceItem MI.wax_WIP_data MI.coreID)
(hasTraceItem MI.wax_WIP_data MI.wax_machine_id)
(hasTraceItem MI.wax_WIP_data MI.wax_robot_start_timestamp)
(hasTraceItem MI.wax_WIP_data MI.wax_process_start_timestamp)
(hasTraceItem MI.wax_cell_layout MI.unique_identifier)
(hasTraceItem MI.WIP_database MI.unique_identifier)
(hasTraceItem MI.part_position_refresh_rate MI.unique_identifier)

;===

(ControlSystem MI.wax_injection_cell_control)
(SCADASystem MI.wax_injection_cell_control)
(ControlSystem MI.wax_injection_cell_control)
(AutomationSystem MI.wax_injection_cell_control)

(informs MI.wax_injection_cell_control MI.wax_process_engineer)
(informs MI.wax_injection_cell_control MI.foundry_maintenence_engineer)
(informs MI.wax_injection_cell_control
MI.foundry_material_requirements_planning_controller)

(Input MI.wax_inject_robot_command_confirmation)
(Input MI.core_request)
(Input MI.wax_injection_cycle_time)
(Input MI.wax_process_failure_warning)
(Input MI.core_chaplet_inspection_result)
(Input MI.wax_kpv)
(Input MI.operator_id)
(Input MI.wax_robot_end_timestamp)
(Input MI.wax_robot_start_timestamp)
(Input MI.wax_process_end_timestamp)
(Input MI.wax_process_start_timestamp)
(Input MI.wax_machine_id)
(Input MI.wax_product_history)
(Input MI.wax_process_history)

(Output MI.automation_instructions)
(Output MI.core_chaplet_inspection_spec)
(Output MI.wax_product_history)
(Output MI.wax_process_history)
(Output MI.wax_process_parameter)
(Output MI.wax_process_downtime_reasons)
(Output MI.wax_product_history)
(Output MI.wax_process_history)
(Output MI.wax_process_end_timestamp)
(Output MI.wax_process_start_timestamp)
(Output MI.operator_id)
(Output MI.wax_inject_robot_command_confirmation)
(Output MI.core_request)
(Output MI.wax_machine_id)

(Resource MI.wax_injection_control_program)
(Resource MI.wax_process_parameter)
(Resource MI.wax_cell_training_record)
(Resource MI.method_of_manufacture)
(Resource MI.wax_cell_HMI)
(Resource MI.wax_process_instruction)
(Resource MI.wax_injection_operator)

(Constraint MI.method_of_manufacture)
(Constraint MI.wax_cell_training_record)

(MI.FunctionalSpecification MI.wax_injection_cell_functional_specification)

(MI.NonFunctionalSpecification MI.wax_injection_cell_nonfunctional_specification)
(ThroughputMetric MI.wax_injection_cell_throughput)
(AvailabilityMetric MI.wax_injection_cell_availability)
(RFTMetric MI.wax_injection_cell_RFT)
(SystemResponseTimeMetric MI.wax_injection_cell_control_response_time)
(MetricTarget MI.wax_injection_cell_throughput_target)
(MetricTarget MI.wax_injection_cell_availability_target)
(MetricTarget MI.wax_injection_cell_RFT_target)
(MetricTarget MI.wax_injection_cell_control_response_time_target)
(hasTarget MI.wax_injection_cell_throughput
MI.wax_injection_cell_throughput_target)
(hasTarget MI.wax_injection_cell_availability MI.wax_injection_cell_availability_target)
(hasTarget MI.wax_injection_cell_RFT MI.wax_injection_cell_RFT_target)
(hasTarget MI.wax_injection_cell_control_response_time
MI.wax_injection_cell_control_response_time_target)

(hasConstraint MI.wax_injection_cell_control
MI.wax_injection_cell_functional_specification)
(hasConstraint MI.wax_injection_cell_control
MI.wax_injection_cell_nonfunctional_specification)

(hasInput MI.wax_injection_cell_control MI.wax_inject_robot_command_confirmation)
(hasInput MI.wax_injection_cell_control MI.core_request)
(hasInput MI.wax_injection_cell_control MI.wax_injection_cycle_time)
(hasInput MI.wax_injection_cell_control MI.wax_process_failure_warning)
(hasInput MI.wax_injection_cell_control MI.core_chaplet_inspection_result)
(hasInput MI.wax_injection_cell_control MI.wax_kpv)
(hasInput MI.wax_injection_cell_control MI.operator_id)
(hasInput MI.wax_injection_cell_control MI.wax_robot_end_timestamp)
(hasInput MI.wax_injection_cell_control MI.wax_robot_start_timestamp)
(hasInput MI.wax_injection_cell_control MI.wax_process_end_timestamp)
(hasInput MI.wax_injection_cell_control MI.wax_process_start_timestamp)
(hasInput MI.wax_injection_cell_control MI.wax_machine_id)
(hasInput MI.wax_injection_cell_control MI.wax_product_history)
(hasInput MI.wax_injection_cell_control MI.wax_process_history)

(hasOutput MI.wax_injection_cell_control MI.automation_instructions)
(hasOutput MI.wax_injection_cell_control MI.core_chaplet_inspection_spec)
(hasOutput MI.wax_injection_cell_control MI.wax_product_history)
(hasOutput MI.wax_injection_cell_control MI.wax_process_history)
(hasOutput MI.wax_injection_cell_control MI.wax_process_parameter)
(hasOutput MI.wax_injection_cell_control MI.wax_process_downtime_reasons)
(hasOutput MI.wax_injection_cell_control MI.wax_product_history)
(hasOutput MI.wax_injection_cell_control MI.wax_process_history)
(hasOutput MI.wax_injection_cell_control MI.wax_process_end_timestamp)
(hasOutput MI.wax_injection_cell_control MI.wax_process_start_timestamp)
(hasOutput MI.wax_injection_cell_control MI.operator_id)

(hasOutput MI.wax_injection_cell_control
MI.wax_inject_robot_command_confirmation)
(hasOutput MI.wax_injection_cell_control MI.core_request)
(hasOutput MI.wax_injection_cell_control MI.wax_machine_id)

(hasResource MI.wax_injection_cell_control MI.wax_injection_control_program)
(hasResource MI.wax_injection_cell_control MI.wax_process_parameter)
(hasResource MI.wax_injection_cell_control MI.wax_cell_training_record)
(hasResource MI.wax_injection_cell_control MI.method_of_manufacture)
(hasResource MI.wax_injection_cell_control MI.wax_cell_HMI)
(hasResource MI.wax_injection_cell_control MI.wax_process_instruction)
(hasResource MI.wax_injection_cell_control MI.wax_injection_operator)

(hasConstraint MI.wax_injection_cell_control MI.method_of_manufacture)
(hasConstraint MI.wax_injection_cell_control MI.wax_cell_training_record)

(hasSecurityCategory MI.wax_injection_cell_control MI.Private)
(hasExportControlCategory MI.wax_injection_cell_control MI.ExportControlled)
(hasRetentionCategory MI.wax_injection_cell_control MI.NonMandatoryRetention)

(hasTimescale MI.wax_injection_cell_throughput_target MI.life_of_contract)
(hasTimescale MI.wax_injection_cell_availability_target MI.life_of_contract)
(hasTimescale MI.wax_injection_cell_RFT_target MI.life_of_order)
(hasTimescale MI.wax_injection_cell_control_response_time_target
MI.life_of_contract)
(hasTimescale MI.wax_injection_cycle_time MI.life_of_contract)
(hasTimescale MI.wax_process_failure_warning MI.life_of_contract)
(hasDataStructure MI.wax_process_failure_warning MI.profinet_signal)
(hasTimescale MI.core_chaplet_inspection_result MI.life_of_type)
(hasDataStructure MI.core_chaplet_inspection_result MI.SQL_table)

(hasTimescale MI.wax_kpv MI.life_of_contract)
(hasDataStructure MI.wax_kpv MI.SQL_table)
(hasTimescale MI.wax_inject_robot_command_confirmation MI.life_of_contract)
(hasDataStructure MI.wax_inject_robot_command_confirmation MI.SQL_table)
(hasTimescale MI.wax_injection_cycle_time MI.life_of_contract)
(hasDataStructure MI.wax_injection_cycle_time MI.SQL_table)
(hasDataStructure MI.wax_inject_robot_command_confirmation MI.SQL_table)
(hasTimescale MI.wax_robot_end_timestamp MI.life_of_contract)
(hasDataStructure MI.wax_robot_end_timestamp MI.SQL_table)
(hasTimescale MI.wax_robot_start_timestamp MI.life_of_contract)
(hasDataStructure MI.wax_robot_start_timestamp MI.SQL_table)
(hasTimescale MI.wax_product_history MI.life_of_type)
(hasDataStructure MI.wax_product_history MI.SQL_table)
(hasTimescale MI.wax_process_history MI.life_of_type)
(hasDataStructure MI.wax_process_history MI.SQL_table)

(hasTimescale MI.wax_injection_control_program MI.life_of_type)
(hasDataStructure MI.wax_injection_control_program MI.XML)
(hasTimescale MI.wax_cell_training_record MI.life_of_type)
(hasDataStructure MI.wax_cell_training_record MI.SQL_table)
(hasTimescale MI.target_wax_injection_cycle_time MI.life_of_order)

(hasDataStructure MI.wax_injection_cycle_time MI.XML)
(MI.SysSysInterface MI.wax_injection_cell_control_MES_Bridge)
(MI.hasInterface MI.wax_injection_cell_control
MI.wax_injection_cell_control_MES_Bridge)
(HMI MI.wax_injection_cell_control_HMI)
(hasInterface MI.wax_injection_cell_control MI.wax_cell_HMI)
(hasInterface MI.wax_injection_operator MI.wax_cell_HMI)
(hasInterface MI.wax_process_engineer MI.wax_cell_HMI)
(hasInterface MI.foundry_material_requirements_planning_controller
MI.wax_cell_HMI)
(hasInterface MI.foundry_maintenence_engineer MI.wax_cell_HMI)
(hasInterface MI.MES_system MI.wax_cell_HMI)
(MI.SysSysInterface MI.wax_injection_cell_control_part_tracking_system_Bridge)
(hasInterface MI.wax_injection_cell_control
MI.wax_injection_cell_control_part_tracking_system_Bridge)
(hasInterface MI.part_tracking_system
MI.wax_injection_cell_control_part_tracking_system_Bridge)

(hasTraceItem MI.wax_process_history MI.operator_id)
(hasTraceItem MI.wax_process_history MI.wax_robot_start_timestamp)
(hasTraceItem MI.wax_process_history MI.wax_process_start_timestamp)
(hasTraceItem MI.wax_process_history MI.wax_machine_id)
(hasTraceItem MI.wax_product_history MI.coreID)
(hasTraceItem MI.wax_product_history MI.partID)
(PersonID MI.operator_id)
(WorkCentreID MI.wax_machine_id)
(StartTimestamp MI.wax_robot_start_timestamp)
(StartTimestamp MI.wax_process_start_timestamp)
(EndTimestamp MI.wax_robot_end_timestamp)
(EndTimestamp MI.wax_process_end_timestamp)
(Report MI.core_chaplet_inspection_result)
(Analysis MI.wax_robot_status_assesment)
(informs MI.wax_robot_status_assesment
MI.wax_inject_robot_command_confirmation_response)
(ReactiveFeedback MI.wax_inject_robot_command_confirmation_response)
(informs MI.wax_inject_robot_command_confirmation
MI.wax_inject_robot_command_confirmation_response)
(UtilisationData MI.wax_injection_cycle_time)
(ThroughputData MI.wax_injection_cycle_time)
(AvailabilityData MI.wax_process_failure_warning)
(InspectionRecord MI.core_chaplet_inspection_result)

(VisualData MI.core_chaplet_inspection_result)
(KPVData MI.wax_kpv)
(ResourceData MI.operator_id)
(TraceabilityData MI.operator_id)
(ProcessData MI.wax_inject_robot_command_confirmation)
(hasTraceItem MI.wax_inject_robot_command_confirmation MI.wax_machine_id)
(hasTraceItem MI.wax_inject_robot_command_confirmation MI.unique_identifier)
(hasTraceItem MI.wax_inject_robot_command_confirmation
MI.wax_process_end_timestamp)
(hasTraceItem MI.wax_inject_robot_command_confirmation
MI.wax_process_start_timestamp)

(TimeData MI.wax_injection_cycle_time)
(TimeData MI.wax_robot_end_timestamp)
(TimeData MI.wax_robot_start_timestamp)
(TimeData MI.wax_process_end_timestamp)
(TimeData MI.wax_process_start_timestamp)
(TraceabilityData MI.wax_process_end_timestamp)
(TraceabilityData MI.wax_process_start_timestamp)
(TraceabilityData MI.wax_machine_id)
(ProductionHistory MI.wax_product_history)
(ProductionHistory MI.wax_process_history)
(ProductionHistory MI.wax_process_failure_warning)
(ProcessParameterData MI.wax_process_history)
(OperationMetric MI.wax_injection_cycle_time)
(MetricTarget MI.target_wax_injection_cycle_time)
(hasTraceItem MI.core_chaplet_inspection_spec MI.coreID)
(hasTraceItem MI.wax_process_end_timestamp MI.partID)
(hasTraceItem MI.wax_process_start_timestamp MI.partID)
(hasTraceItem MI.wax_process_start_timestamp MI.operator_id)
(hasTraceItem MI.wax_process_end_timestamp MI.operator_id)
(hasTarget MI.wax_injection_cycle_time MI.target_wax_injection_cycle_time)

(ProductData MI.core_chaplet_inspection_spec)
(VisualSpecification MI.core_chaplet_inspection_spec)
(ProductSpecification MI.core_chaplet_inspection_spec)
(informs MI.method_of_manufacture MI.automation_instructions)
(specifies MI.method_of_manufacture MI.wax_process_parameter)
(informs MI.core_chaplet_inspection_spec MI.core_chaplet_inspection_result)

(ProgramData MI.wax_injection_control_program)
(ResourceData MI.wax_cell_training_record)
(informs MI.wax_cell_training_record MI.wax_injection_cell_control)
(HMI MI.wax_cell_HMI)
(visualises MI.wax_cell_HMI MI.wax_injection_cell_control)

(interopsWith MI.wax_injection_operator MI.wax_cell_HMI)
(specifies MI.wax_cell_training_record MI.TechAuthorityLevel)

(MI.Decision MI.wax_StartDecision)
(MI.Decision MI.wax_StopDecision)
(MI.Decision MI.wax_ChangeDecision)
(MI.Decision MI.wax_ContinueDecision)
(MI.Analysis MI.compare_actual_vs_spec_analysis)
(informs MI.compare_actual_vs_spec_analysis MI.wax_StartDecision)
(informs MI.compare_actual_vs_spec_analysis MI.wax_StopDecision)
(informs MI.compare_actual_vs_spec_analysis MI.wax_ChangeDecision)
(informs MI.compare_actual_vs_spec_analysis MI.wax_ContinueDecision)
(HMI MI.wax_cell_HMI)
(visualises MI.wax_cell_HMI wax_process_failure_warning)
(informs MI.wax_cell_HMI MI.wax_injection_operator)
(informs MI.wax_injection_operator MI.wax_cell_HMI)
(visualises MI.wax_cell_HMI MI.core_chaplet_inspection_result)
(visualises MI.wax_cell_HMI MI.WIPInventoryData)
(visualises MI.wax_cell_HMI MI.wax_product_history)
(visualises MI.wax_cell_HMI MI.wax_process_history)
(visualises MI.wax_cell_HMI MI.wax_process_parameter)
(visualises MI.wax_cell_HMI MI.wax_process_downtime_reasons)
(visualises MI.wax_cell_HMI MI.wax_injection_control_program)

(NonFunctionalSpecification MI.wax_injection_cell_nonfunctional_specification)
(FunctionalSpecification MI.wax_injection_cell_functional_specification)
(specifies MI.wax_injection_cell_functional_specification
MI.wax_injection_cycle_time)
(specifies MI.wax_injection_cell_functional_specification
MI.target_wax_injection_cycle_time)

(MI.Visualisation MI.wax_cell_HMI)

(MI.hasSecurityCategory MI.wax_inject_robot_command_confirmation MI.Private)
(MI.hasSecurityCategory MI.wax_process_downtime_reasons MI.Private)
(MI.hasSecurityCategory MI.core_request MI.Private)
(MI.hasSecurityCategory MI.wax_process_failure_warning MI.Private)
(MI.hasSecurityCategory MI.wax_injection_cycle_time MI.Private)
(MI.hasSecurityCategory MI.automation_instructions MI.Private)

14 Appendix D – Fact listing for a system from a difference time domain.

;;;===
;;;Activplant legacy system
;;;===

(System MI.ActivPlantSystem)
(VisualisationSystem MI.ActivPlantSystem)
(DataCollectionSystem MI.ActivPlantSystem)
(ReportingSystem MI.ActivPlantSystem)

(Metric MI.PeriodOfInactivity)
(MetricTarget MI.PeriodOfInactivityLimit)
(hasTarget MI.PeriodOfInactivity MI.PeriodOfInactivityLimit)
(Metric MI.Availability)
(MetricTarget MI.AvailabilityTarget)
(hasTarget MI.Availability MI.AvailabilityTarget)
(Metric MI.ThroughputTime)
(MetricTarget MI.DesignThroughputTime)
(hasTarget MI.ThroughputTime MI.DesignThroughputTime)
(specifies MI.BusinessRequirementDocument MI.PeriodOfInactivity)
(specifies MI.BusinessRequirementDocument MI.PeriodOfInactivityLimit)
(specifies MI.BusinessRequirementDocument MI.ThroughputTime)
(specifies MI.BusinessRequirementDocument MI.DesignThroughputTime)
(specifies MI.BusinessRequirementDocument MI.Availability)
(specifies MI.BusinessRequirementDocument MI.AvailabilityTarget)

(Output MI.MachineInhibitSwitchSignal)
(Output MI.MEAArchive)
(Output MI.OEEReport)
(Output MI.KPVReport)
(Output MI.MeasurementResults)
(Output MI.OperatorChangeLog)

(hasOutput MI.ActivPlantSystem MI.MachineInhibitSwitchSignal)
(hasOutput MI.ActivPlantSystem MI.MEAArchive)
(hasOutput MI.ActivPlantSystem MI.OEEReport)
(hasOutput MI.ActivPlantSystem MI.KPVReport)
(hasOutput MI.ActivPlantSystem MI.MeasurementResults)
(hasOutput MI.ActivPlantSystem MI.OperatorChangeLog)

(Input MI.PlannedDowntime)
(Input MI.DownTimeLosses)
(Input MI.MEAFile)
(Input MI.TraceabilityItems)

(Input MI.VisualInspectionResults)
(Input MI.DimensionalInspectionResults)

(hasInput MI.ActivPlantSystem MI.PlannedDowntime)
(hasInput MI.ActivPlantSystem MI.DownTimeLosses)
(hasInput MI.ActivPlantSystem MI.MEAFile)
(hasInput MI.ActivPlantSystem MI.TraceabilityItems)
(hasInput MI.ActivPlantSystem MI.VisualInspectionResults)
(hasInput MI.ActivPlantSystem MI.DimensionalInspectionResults)

(Resource MI.ActivplantDatabase)
(Resource MI.MagerleOperator)
(Resource MI.NGVEngineer)
(Resource MI.PalletTracker)
(Resource MI.OPCLayer)
(Resource MI.840DController)

(hasResource MI.ActivPlantSystem MI.ActivplantDatabase)
(hasResource MI.ActivPlantSystem MI.MagerleOperator)
(hasResource MI.ActivPlantSystem MI.NGVEngineer)
(hasResource MI.ActivPlantSystem MI.PalletTracker)
(hasResource MI.ActivPlantSystem MI.OPCLayer)
(hasResource MI.ActivPlantSystem MI.840DController)

(Constraint MI.BusinessRequirementDocument)
(hasConstraint MI.ActivPlantSystem MI.BusinessRequirementDocument)

(Operator MI.MagerleOperator)
(ManufacturingEngineer MI.NGVEngineer)

(informs MI.ActivPlantSystem MI.NGVEngineer)
(informs MI.ActivPlantSystem MI.MagerleOperator)

(hasTechAuthorityLevel NGVEngineer MI.User)
(hasTechAuthorityLevel MagerleOperator MI.User)
(hasOpAuthorityLevel MagerleOperator MI.Employee)
(hasOpAuthorityLevel NGVEngineer MI.Employee)
(StandardDataStructure FlatFileStructure)

(Visualisation MI.PlasmaScreen)
(visualises MI.PlasmaScreen MI.ActivPlantSystem)
(Report MI.OEEReport)

(NonFunctionalSpecification MI.BusinessRequirementDocument)
(FunctionalSpecification MI.BusinessRequirementDocument)
(hasSpecification MI.ActivPlantSystem MI.BusinessRequirementDocument)
(hasRetentionCategory MI.ActivPlantSystem MI.NonMandatoryRetention)

(DataStructure MI.FlatFileStructure)
(hasDataStructure MI.MEAArchive MI.FlatFileStructure)
(Report MI.RAGChart)
(Report MI.RFTReport)

(TraceabilityData MI.BatchID)
(TraceabilityData MI.PartID)
(TraceabilityData MI.PalletID)
(TraceabilityData MI.SerialNumber)

(TraceabilityItem MI.BatchID)
(TraceabilityItem MI.PartID)
(TraceabilityItem MI.PalletID)
(TraceabilityItem MI.SerialNumber)

(hasTraceItem MI.MEAArchive MI.BatchID)
(hasTraceItem MI.OEEReport MI.BatchID)
(hasTraceItem MI.KPVReport MI.BatchID)
(hasTraceItem MI.MeasurementResults MI.BatchID)
(hasTraceItem MI.MEAArchive MI.PartID)
(hasTraceItem MI.OEEReport MI.PartID)
(hasTraceItem MI.KPVReport MI.PartID)
(hasTraceItem MI.MeasurementResults MI.PartID)
(hasTraceItem MI.MEAArchive MI.PalletID)
(hasTraceItem MI.OEEReport MI.PalletID)
(hasTraceItem MI.KPVReport MI.PalletID)
(hasTraceItem MI.MeasurementResults MI.PalletID)
(hasTraceItem MI.MEAArchive MI.SerialNumber)
(hasTraceItem MI.OEEReport MI.SerialNumber)
(hasTraceItem MI.KPVReport MI.SerialNumber)
(hasTraceItem MI.MeasurementResults MI.SerialNumber)

(UniqueIdentifier MI.BatchID)
(ProductID MI.PartID)
(UniqueIdentifier MI.PalletID)
(ProductID MI.SerialNumber)

15 Appendix E – Full Listing of Type and Logic Declaration

```
CORE ONTOLOGY
Classes.KFL
:Ctx MI
:Inst UserContext
:supCtx MLO
:name "Manuf Intel Context"


:Use MI

;;;================================================
;;; Core Classes - Objects
;;;================================================
:Prop Input
:Inst Type
:sup Object
:name "Input"
:rem "An Input is taken into or used by a process or system. All process or systems require some form of Input . Input is often the result of an Output from another System or Process however an Output can also be the Input of of the same system or process: this is refered to closed loop Feedback. A single system may have multiple input types and sources. Wherever possible the source of input should be the only place that input is available from or the source designated the authorative source."

:Prop Output
:Inst Type
:sup Object
:name "Output Data"
:rem "An Output results from  the process of system activity. Even If an Input passes through the process unchanged it should be considered output from that process if collected after processing. Outputs are often collected to populate a Report."

:Prop TraceabilityItem
:Inst Type
:sup Object
:name "Traceability Item"
:rem "Traceability items are attached to data to allow it to be sorted, stored, retrieved and to enable meaningful analysis across datasets."

:Prop Data
:Inst Type
:sup Object
:name "Data"
:rem "Data is the lowest level of abstraction of information or knowledge and are numbers, characters or images that require a context to have meaning, at which point it becomes information."

:Prop System
:Inst Type
:sup Object
:name "System"
:rem "A systems takes inputs and processes them into a required output using defined processes and system resources within system constraints."

:Prop Constraint
:Inst Type
:sup Object
:name "Constraint"
:rem "Constraints are the limits, rules and structures within which an entitiy must exist or function."

:Prop Resource
:Inst Type
:sup Object
:name "Resource"
:rem "Resources are the things used or consumed or required by an entity or process."

;;; :Prop ManufacturingMethod
;;; :Inst Type
;;; :sup Object
;;; :name "Manufacturing Method"
;;; :rem "The Manufacturing Method comprises the Bill of Material, Bill of Plant and Bill of Process, which are combined with their constituate information to fully describe how to manufacturing a specific product in a specific facility. Manufacturing Methods can be generalised into generic template methods which are not directly applicable to any specific product or facility but that can be easily specialised to suit. note: this is distinct from the IMKS definition of Manufacturing Method."

:Prop Person
:Inst Type
:sup Object
:name "Person"
:rem "A Person is the role player or agent that carries out manual content of an activity or process."
```

```
:Prop Metric
:Inst Type
:sup Object
:name "Metric"
:rem "A Metric is a measure that quantifies an attribute. Metrics are usually chosen to drive specific direction or activities within an enterprise using metric targets and are linked to a required outcome. Metrics are usually required to be highly visual to be effective."

:Prop Target
:Inst Type
:sup Object
:name "Target"
:rem "Targets represent the aim of the enterprise or activity as described by the available metrics, therefore if the targets are defined effectively and are achieved, the enterprise will have achieved its aim."

:Prop Analysis
:Inst Type
:sup Object
:name "Analysis"
:rem "Analysis is the structured review of data, information and knowledge to elicit further data, information and knowledge from it."

:Prop Interface
:Inst Type
:sup Object
:name "Interface"
:rem "An interface is where one entity or system meets and interacts with another."

:Prop DataStructure
:Inst Type
:sup Object
:name "Data Structure"
:rem "To enable to automatic and repeatable interpretation and processing of data it must have an known structure which relates the data elements to each other and potentially other data-sets."

:Prop Visualisation
:Inst Type
:sup Object
:name "Visualisation"
:rem "Visualisation is the act or way of making something visually interpretable. This is considdered an object due to the inherent need for visibiliy to have some eliment of physical embodyment."

;;;================================================
;;; Core Classes - Events
;;;================================================

:Prop Prediction
:Inst Type
:sup Event
:name "Prediction"
:rem "A prediction is a statement of what will occur in the future such as a future position or the outcome of an event, predictions are based on data, information and knowledge who's availability is inversely proportional to the level of uncertainty in the prediction."

:Prop SustainmentProcess
:Inst Type
:sup Event
:name "Sustainment Process"
:rem " A Sustainment process is one that counteracts change and variation (both common and special cause) to ensure the continuity of the current or intended state or process. It is important to note that sustainment processes maintain the original intent of the current state, so in changing circumstances a sustainment process will introduce change to maintain the intended state i.e. Sustainment does to prevent all change in the process."

:Prop Decision
:Inst Type
:sup Event
:name "Decision"
:rem "A decision is the point of choosing from multiple options based on a desired outcome and the available information. The current decision may be significantly removed from the desired outcome i.e. many further decisions may be required, and may have direct, indirect, intended and unintended implications."

:Prop Collaboration
:Inst Type
:sup Event
:name "Collaboration"
:rem "Collaboration is where 2 or more entities work together towards a shared aim or outcome."


;;;================================================
;;; Core Classes - Quantity
;;;================================================

:Prop Timescale
:Inst Type
:sup Duration
:name "Timescale"
:rem "A Timescale is the period of time relating to or bounding a process or activity."

:Prop Status
:Inst Type
:sup Quantity
```

```
:name "Status"
:rem "Any object should have a status wihich describes the review or approval status of an
item, this indicates what Authority Level or Person has access to the item as well as how it
may be used."

:Prop AuthorityLevel
:Inst Type
:sup Quantity
:name "Authority Level"
:rem "Authority Levels are used to control the ability to carry out activities. Authority levels are
usually an incremental scale. If an individual or system does not have the required authority
level specified for an action, a logical check will not permit them to carry out that action."

Relations.KFL
;;; Start Date: 20th October 2011
;;; Author: Neil Hastilow
;;; Other Contributors: Tish Chungoora

:Use MI

;;;================================================
;;; Relations - From UML Core Concept Model
;;;================================================

:Rel anticipates
:Inst BinaryRel
:Inst IrreflexiveBR
:Inst TransitiveBR
:Sig Top Top
:name "anticipates"

:Rel informs
:Inst BinaryRel
:Inst TransitiveBR
:Sig Top Top
:name "informs"

:Rel hasStructure
:Inst BinaryRel
:Sig Data DataStructure
:name "has Structure"

:Rel hasTraceItem
:Inst BinaryRel
:Sig Top Top
:name "has Traceability Item "

:Rel hasAuthorityLevel
:Inst BinaryRel
:Sig Top Top
:name "has Authority Level"

:Rel hasStatus
:Inst BinaryRel
:Sig Top Status
:name "has Status"

:Rel analyses
:Inst BinaryRel
:Sig Analysis Data
:name "analyses"

:Rel hasTarget
:Inst BinaryRel
:Sig Metric Target
:name "has Target"

:Rel hasOutput
:Inst BinaryRel
:Sig System Output
:name "has Output"

:Rel specifies
:Inst BinaryRel
:Sig Top Top
:name "specifies"

:Rel hasConstraint
:Inst BinaryRel
:Sig Top Constraint
:name "has Constraint"

:Rel hasResource
:Inst BinaryRel
:Sig Top Resource
:name "has Resource"

:Rel visualises
:Inst BinaryRel
:Sig Visualisation Top
:name "visualises"

:Rel sustains
:Inst BinaryRel
:Sig SustainmentProcess Type
:name "sustains"

:Rel interopsWith
```

```
:Inst BinaryRel
:Inst SymmetricBR
:Sig Top Top
:name "interoperates with"

:Rel hasInterface
:Inst BinaryRel
:Sig Top Top
:name "has interface"

:Rel enables
:Inst BinaryRel
:Sig Top Top
:name "enables"

:Rel hasInput
:Inst BinaryRel
:Sig Top Input
:name "has Input"


Logic.KFL
;;; Start Date: 13th November 2011
;;; Author: Neil Hastilow
;;; Other Contributors: Tish Chungoora

:Use MI

;;;================================================
;;; Logic - UML Core Concept Model
;;;================================================



;;;================================================
;;; ICs - Core Concept model (constraining general relationships)
;;;================================================

;;;85 IC
(=> (anticipates ?x ?y)
        (or (and (Prediction ?x)
                        (Output ?y))
          (and (Prediction ?x)
                        (Decision ?y))))
:IC hard "An instance of <sym>Prediction</sym> anticipates the <sym>Output</sym>
instance or <sym>Decision</sym> instance."

;;;86 IC
(=> (anticipates ?x ?y)
        (or (and (supTC ?x Prediction)
                        (supTC ?y Output))
          (and (supTC ?x Prediction)
                        (supTC ?y Decision))))
:IC hard "A <sym>Prediction</sym> type anticipates the <sym>Output</sym> type or
<sym>Decision</sym> type."

;;;87 IC
(=> (Decision ?y)
        (exists (?x)
                (and (Analysis ?x)
                        (informs ?x ?y))))
:IC hard "<sym>Analysis</sym><code>?x</code> informs <sym>Decision</sym>
<code>?y</code>."

;;;88 IC
(=> (supTC ?y Decision)
        (exists (?x)
                (and (or (supTC ?x Analysis)
                                (= ?x Analysis))
                        (informs ?x ?y))))
:IC hard "<sym>Analysis</sym> <code>?x</code>or some type of Analysis should inform
some <sym>Decision</sym> type <code>?y</code>."


;;;89a IC
(=> (and (System ?x)
                (System ?y)
                (interopsWith ?x ?y))
        (exists (?i1 ?i2)
                (and (Interface ?i1)
                        (Interface ?i2)
                        (hasInterface ?x ?i1)
                        (hasInterface ?y ?i2))))
:IC hard "If two system individuals <code>?x</code> <code>?y</code>interoperate with each
other, then they require some associated interface individual."

;;;89b IC
(=> (and (System ?x)
                (Person ?y)
                (interopsWith ?x ?y))
        (exists (?i1 ?i2)
                (and (Interface ?i1)
                        (Interface ?i2)
                        (hasInterface ?x ?i1)
                        (hasInterface ?y ?i2))))
:IC hard "If a system individual <code>?x</code> interoperate with a person individual
<code>?y</code>, then they require some associated interface individual."
```

```
;;;89c IC
(=> (and (supTC ?x System)
                 (supTC ?y System)
                 (interopsWith ?x ?y))
        (exists (?i1 ?i2)
                 (and (supTC ?i1 Interface)
                          (supTC ?i2 Interface)
                          (hasInterface ?x ?i1)
                          (hasInterface ?y ?i2))))
:IC hard "If two system types <code>?x</code> <code>?y</code> interoperate with each
other, then they require some associated interface type."

;;;89d IC
(=> (and (supTC ?x System)
                 (supTC ?y Person)
                 (interopsWith ?x ?y))
        (exists (?i1 ?i2)
                 (and (supTC ?i1 Interface)
                          (supTC ?i2 Interface)
                          (hasInterface ?x ?i1)
                          (hasInterface ?y ?i2))))
:IC hard "If a system type interoperates with a person type, then both system and person
types require some associated interface type."

;;;90 IC
(=> (and (interopsWith ?t1 ?t2)
                 (hasInterface ?t1 ?i1)
                 (hasInterface ?t2 ?i2))
        (exists (?i)
                 (and (Interface ?i)
                                  (hasInterface ?t1 ?i)
                                  (hasInterface ?t2 ?i))))
:IC hard "If two arguments <code>?t1</code> <code>?t2</code> interoperate with each
other they must have some associated interface <code>?i</code> in common."


;;;================================================
;;; IRs - Core Concept model
;;;================================================


LEVEL 1 MODULES
Decision.KFL
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1
;;; Version 2 - reponse core concept demoted to sub concept of decision, change idenitified
by the logical descriptions being compatible.

:Use MI

;;;================================================
;;; Decision Classes - Level 1
;;;================================================

:Prop Response
:Inst Type
:sup Decision
:name "Response"
:rem "A response is a reaction to an occurrence or situation. The existence of a response is
dependent on the initial occurrence and the nature of the response will depend on the nature
of the occurrence"

:Prop StopDecision
:Inst Type
:sup Decision
:name "Stop Decision"
:rem " A Stop decision is one that results in the ceasation of a process or activity ."

:Prop StartDecision
:Inst Type
:sup Decision
:name "Start Decision"
:rem " A Start decision is one that results in the initiation of a process or activity."

:Prop ContinueDecision
:Inst Type
:sup Decision
:name "Continue Decision"
:rem "A Continue Decision is one that results in the unmodified continuation of a process or
activity ."

:Prop ChangeDecision
:Inst Type
:sup Decision
:name "Change Decision"
:rem "A Change Decision is one that results in the modified continuation of a process or
activity ."

                ;;;================================================
                ;;; Response Classes
                ;;;================================================

                :Prop Notification
                :Inst Type
                :sup Response
                :name "Notification"
                :rem " ."
```

```
:Prop ANDON
:Inst Type
:sup Response
:name "ANDON"
:rem " ."

:Prop Alarm
:Inst Type
:sup Response
:name "Alarm"
:rem " ."

:Prop Feedback
:Inst Type
:sup Response
:name "Feedback"
:rem " ."

;;;================================================
;;; Feedback Classes
;;;================================================

                    :Prop ProactiveFeedback
                    :Inst Type
                    :sup Feedback
                    :name "Proactive Feedback"
                    :rem " ."

                    :Prop ReactiveFeedback
                    :Inst Type
                    :sup Feedback
                    :name "ReactiveFeedback"
                    :rem " ."

                    :Prop ProcessException
                    :Inst Type
                    :sup Alarm
                    :name "ProcessException"
                    :rem " ."

                              :Prop Adaptation
                              :Inst Type
                              :sup ProactiveFeedback
                              :name "Adaptation"
                              :rem " ."

                              :Prop Correction
                              :Inst Type
                              :sup ReactiveFeedback
                              :name "Correction"
                              :rem " ."

                              :Prop Reschedule
                              :Inst Type
                              :sup ReactiveFeedback
                              :name "Reschedule"
                              :rem " ."

(informs Analysis Response)
(informs Analysis StopDecision)
(informs StatisticalAnalysis StopDecision)
(informs Analysis ContinueDecision)
(informs Analysis StartDecision)
(informs StatisticalAnalysis ContinueDecision)
(informs Analysis ChangeDecision)
(informs StatisticalAnalysis ChangeDecision)
(informs BooleanAnalysis Notification)
(informs BooleanAnalysis ANDON)
(informs BooleanAnalysis Alarm)
(informs Analysis Feedback)
(informs Analysis ProactiveFeedback)
(informs Analysis ReactiveFeedback)
(informs Analysis ProcessException)
(informs Analysis Adaptation)
(informs Analysis Correction)
(informs Analysis Reschedule)

Interface.KFL
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;================================================
;;; Interface  Classes - Level 1
;;;================================================

:Prop HMI
:Inst Type
:sup Interface
:name "Human Machine Interface"
:rem "A Human Machine Interface is the mechanism that allows a Person to Interface with a
Machine or System"

:Prop SysSysInterface
:Inst Type
```

```
:sup Interface
:name "System-System Interface"
:rem "System-System Interfaces are interfaces between systems with not direct human
interaction "

Logic.KFL
;;; Start Date: 21st October 2011
;;; Author: N Hastilow
;;; Other Contributors: Tish Chungoora

:Use MI


;;;============================================================
;;;Actions outstanding
;;;Review all logic and segregate logic only referring to CORE CONCEPT types
;;;review the use of type and instances for each rule and duel declare if reqd
;;;note - all logic is created at this level by default, but is promoted to core level after review
;;;============================================================

(ObsoleteStatus Obsolete)
(ArchievedStatus Archieved)
(DeletedStatus Deleted)
(ReturnStatus Return)
(ApprovedStatus Approved)
(UnapprovedStatus Unapproved)
(ConditionallyApprovedStatus ConditionallyApproved)


(UserTechAuthorityLevel User)
(SuperuserTechAuthorityLevel Superuser)
(AdministratorTechAuthorityLevel Administrator)

(EmployeeOpAuthorityLevel Employee)
(LeaderOpAuthorityLevel Leader)
(ManagerOpAuthorityLevel Manager)
(ExecutiveOpAuthorityLevel Executive)


(higherThan LeaderOpAuthorityLevel EmployeeOpAuthorityLevel)
(higherThan ManagerOpAuthorityLevel LeaderOpAuthorityLevel)
(higherThan ExecutiveOpAuthorityLevel ManagerOpAuthorityLevel)

(higherThan SuperuserTechAuthorityLevel UserTechAuthorityLevel)
(higherThan AdministratorTechAuthorityLevel SuperuserTechAuthorityLevel)

(hasTimescale MetricTarget UnknownValidityTimescale)
(hasTimescale Plan UnknownValidityTimescale)
(hasTimescale Vision UnknownValidityTimescale)
(hasTimescale Roadmap UnknownValidityTimescale)
(hasTimescale ProcessPlan UnknownValidityTimescale)
(hasTimescale ProductionPlan UnknownValidityTimescale)
(hasTimescale ReactionPlan UnknownValidityTimescale)
(hasTimescale InspectionPlan UnknownValidityTimescale)


;;;================================================
;;; General integrity constraints
;;;================================================


;;;62 IC.
(=> (and (System ?s)
                    (Input ?i)
                    (Output ?o)
                    (hasSecurityCategory ?s MI.Private)
                    (hasInput ?s ?i)
                    (hasOutput ?s ?o))
          (and (or (hasSecurityCategory ?i MI.Public)
                              (hasSecurityCategory ?i MI.Private))
                    (or (hasSecurityCategory ?o MI.Public)
                              (hasSecurityCategory ?o MI.Private))))
:IC hard "Systems  <code>?s</code> that have security category of private can only input
<code>?i</code> or output <code>?o</code> public or private category data."

;;;63 IC.
(=> (and (System ?s)
                    (Input ?i)
                    (Output ?o)
                    (hasSecurityCategory ?s MI.Secret)
                    (hasInput ?s ?i)
                    (hasOutput ?s ?o))
          (and (or (hasSecurityCategory ?i MI.Public)
                              (hasSecurityCategory ?i MI.Private)
                              (hasSecurityCategory ?i MI.Secret))
                    (or (hasSecurityCategory ?o MI.Public)
                              (hasSecurityCategory ?o MI.Private)
                              (hasSecurityCategory ?o MI.Secret))))
:IC hard "Systems  <code>?s</code> that have security category of secret can only input
<code>?i</code> or output <code>?o</code> public, private or secret category data."

;;;64 IC.
(=> (and (System ?s)
                    (Input ?i)
                    (Output ?o)
                    (hasSecurityCategory ?s MI.TopSecret)
                    (hasInput ?s ?i)
                    (hasOutput ?s ?o))
```

```
        (and (or (hasSecurityCategory ?i MI.Public)
                 (hasSecurityCategory ?i MI.Private)
                 (hasSecurityCategory ?i MI.Secret)
                 (hasSecurityCategory ?i MI.TopSecret))
             (or (hasSecurityCategory ?o MI.Public)
                 (hasSecurityCategory ?o MI.Private)
                 (hasSecurityCategory ?o MI.Secret)
                 (hasSecurityCategory ?o MI.TopSecret))))
:IC hard "Systems  <code>?s</code> that have security category of top secret can input
<code>?i</code> or output <code>?o</code> public, private, secret or  top secret category
data."

;;;66 ICa.
(=> (and (System ?s)
         (Input ?i)
         (hasRetentionCategory ?s MI.NonMandatoryRetention)
         (hasInput ?s ?i))
    (or (hasRetentionCategory ?i MI.NonMandatoryRetention)
        (hasOutput ?s ?i)))
:IC hard "Systems <code>?s</code> with a retention category of non mandatory retention
can only input <code>?i</code> o non mandatory retention data. If mandatory data is input to
a non mandetory retention system is must be output as well so it exists in another system."

;;;65 IC.
(=> (and (System ?s)
         (Input ?i)
         (Output ?o)
         (hasExportControlCategory ?s MI.NonExportControlled)
         (hasInput ?s ?i)
         (hasOutput ?s ?o))
    (and (hasExportControlCategory ?i MI.NonExportControlled)
         (hasExportControlCategory ?o MI.NonExportControlled)))
:IC hard "Systems  <code>?s</code> with an export control category of non export control
can only input  <code>?i</code> or output  <code>?o</code> non export controlled data."


;;;6 IC
(=> (Person ?p)
    (exists (?ta)
        (and (TechAuthorityLevel ?ta)
             (hasTechAuthorityLevel ?p ?ta))))
:IC hard "Every <sym>Person</sym> <code>?p</code> item must have a
<sym>TechAuthorityLevel</sym> <code>?ta</code>."

;;;7 IC
(=> (Person ?p)
    (exists (?oa)
        (and (OpAuthorityLevel ?oa)
             (hasOpAuthorityLevel ?p ?oa))))
:IC hard "Every <sym>Person</sym> <code>?p</code> item must have a
<sym>OpAuthorityLevel</sym> <code>?oa</code>."

;;;13a IC
(=> (Data ?d)
    (exists (?sds)
        (and (StandardDataStructure ?sds)
             (hasDataStructure ?d ?sds))))
:IC soft "<sym>Data</sym> <code>?d</code> should use a
<sym>StandardDataStructure</sym> <code>?sds</code>."

;;;13b IC - removed as not valid at type level
;(=> (supTC ?d Data)
;    (exists (?sds)
;        (and (supTC ?sds StandardDataStructure)
;             (hasDataStructure ?d ?sds))))
;:IC soft "<sym>Data</sym> <code>?d</code> types should use a
<sym>StandardDataStructure</sym> type <code>?sds</code>."

;;;34a IC
(=> (Monitor ?mr)
    (exists (?me)
        (and (Metric ?me)
             (hasMonitor ?mr ?me))))
:IC soft "A <sym>Metric</sym> <code>?me</code> should be defined for any
<sym>Monitor</sym> <code>?mr</code> to monitor."

;;;34b IC
(=> (supTC ?mr Monitor)
    (exists (?me)
        (and (supTC ?me Metric)
             (hasMonitor ?mr ?me))))
:IC soft "A <sym>Metric</sym> <code>?me</code> type should be defined for any type of
<sym>Monitor</sym> <code>?mr</code> to monitor."

;;;35 IC
(=> (Metric ?me)
    (exists (?t)
        (and (Target ?t)
             (hasTarget ?me ?t))))
:IC hard "A <sym>Metric</sym> <code>?me</code> must have a <sym>Target</sym>
<code>?t</code> to target."

;;;36 IC
(=> (Monitor ?m)
    (exists (?r)
        (and (Response ?r)
             (hasResponse ?m ?r))))
```

```
:IC hard "A <sym>Monitor</sym> <code>?m</code> must have a defined
<sym>Response</sym> <code>?r</code>."

;;;15a IC
(=> (and (System ?s)
        (Input ?i)
        (Output ?o)
        (hasInput ?s ?i)
        (hasOutput ?s ?o)
                        (ApprovedStatus ?a)
        (hasStatus ?o ?a))
   (hasStatus ?i ?a))
:IC hard "If a system <code>?s</code> output <code>?o</code> is associated with an
ApprovedStatus individual, then the system input(s) <code>?i</code> must also have the
same status."

;;;15b IC
(=> (and (System ?s)
        (Input ?i)
        (Output ?o)
        (hasInput ?s ?i)
        (hasOutput ?s ?o)
                        (supTC ?a ApprovedStatus)
        (hasStatus ?o ?a))
   (hasStatus ?i ?a))
:IC hard "If a system <code>?s</code> output <code>?o</code> is associated with an
ApprovedStatus type, then the system input(s) <code>?i</code> must also also have the
same status type."

;;;15c IC
(=> (and (System ?s)
        (Input ?i)
        (Output ?o)
        (hasInput ?s ?i)
        (hasOutput ?s ?o)
        (hasStatus ?o ApprovedStatus))
   (hasStatus ?i ApprovedStatus))
:IC hard "If a system output <code>?o</code> has ApprovedStatus, then the system input(s)
<code>?i</code> must also have ApprovedStatus."

;;;14a IC
(=> (and (Top ?t)
                        (ObsoleteStatus ?o)
                        (DeletedStatus ?d)
                        (or (hasStatus ?t ?o)
                                (hasStatus ?t ?d)))
            (not (exists (?s)
                                (and (System ?s)
                                    (hasInput ?s ?t)))))
:IC hard "Anything <code>?t</code> with an associated individual of
<sym>ObsoleteStatus</sym> or <sym>DeletedStatus</sym> cannot be an input to a System
individual."

;;;14b IC
(=> (and (Top ?t)
                        (supTC ?o ObsoleteStatus)
                        (supTC ?d DeletedStatus)
                        (or (hasStatus ?t ?o)
                                (hasStatus ?t ?d)))
            (not (exists (?s)
                                (and (supTC ?s System)
                                    (hasInput ?s ?t)))))
:IC hard "Anything <code>?t</code> with an associated type of <sym>ObsoleteStatus</sym>
or <sym>DeletedStatus</sym> cannot be an input to a System type."

;;;14c IC
(=> (and (Top ?t)
                        (or (hasStatus ?t ObsoleteStatus)
                                (hasStatus ?t DeletedStatus)))
            (not (exists (?s)
                                (and (System ?s)
                                    (hasInput ?s ?t)))))
:IC hard "Anything <code>?t</code> with <sym>ObsoleteStatus</sym> or
<sym>DeletedStatus</sym> cannot be an input to a System individual."

;;;14c IC
(=> (and (Top ?t)
                        (or (hasStatus ?t ObsoleteStatus)
                                (hasStatus ?t DeletedStatus)))
            (not (exists (?s)
                                (and (supTC ?s System)
                                    (hasInput ?s ?t)))))
:IC hard "Anything <code>?t</code> with <sym>ObsoleteStatus</sym> or
<sym>DeletedStatus</sym> cannot be an input to a System type."


;;;39a IC
(=>(ProactiveFeedback ?pf)
                        (exists (?d)
                        (and(Data ?d)
                                        (Input ?d)
                                        (informs ?d ?pf))))
:IC hard "Proactive feedback individuals <code>?pf</code> are informed by input data
individuals <code>?d</code>."


;;;40a IC
(=>(ReactiveFeedback ?rf)
```

```
                (exists (?d)
                (and(Data ?d)
                                (Output ?d)
                                (informs ?d ?rf))))
:IC hard "Reactive feedback individuals <code>?rf</code> are informed by ouput data
individuals <code>?d</code>."



;;;44a IC
(=> (System ?s)
        (exists (?i ?o ?r ?c)
                (and (hasInput ?s ?i)
                        (hasOutput ?s ?o)
                        (hasResource ?s ?r)
                        (hasConstraint ?s ?c))))
:IC hard "A system individual <code>?s</code> must have some form of inputs, outputs,
resource and constraints defined."

;;;44b IC - removed as not logical
;(=> (supTC ?s System)
;       (exists (?i ?o ?r ?c)
;               (and (hasInput ?s ?i)
;                       (hasOutput ?s ?o)
;                       (hasResource ?s ?r)
;                       (hasConstraint ?s ?c))))
;:IC hard "A system type must have some form of inputs, outputs, resource and constraints
defined."

;;;46a IC
(=> (Target ?t)
        (exists (?v)
                (and (ValidityTimescale ?v)
                        (hasTimescale ?t ?v))))
:IC soft "Target <code>?t</code> individuals have a validity timescale individual
<code>?v</code>."

;;;46b IC
(=> (supTC ?t Target)
        (exists (?v)
                (and (ValidityTimescale ?v)
                        (hasTimescale ?t ?v))))
:IC soft "Target <code>?t</code> types have a validity timescale type  <code>?t</code>."

;;;47a IC
(=> (Data ?d)
        (exists (?v)
                (and (ValidityTimescale ?v)
                        (hasTimescale ?d ?v))))
:IC hard "Data <code>?d</code> individuals have a validity timescale individual."

;;;47b IC - removed as not true at type level
;(=> (supTC ?d Data)
;       (exists (?v)
;               (and (supTC ?v ValidityTimescale)
;                       (hasTimescale  ?d ?v))))
;:IC hard "Data types <code>?d</code> have a validity timescale type<code>?v</code>."

;;;37 IC
(=> (System ?s)
        (exists (?fs ?nfs)
                (and (FunctionalSpecification ?fs)
                        (NonFunctionalSpecification ?nfs)
                        (hasConstraint ?s ?fs)
                        (hasConstraint ?s ?nfs))))
:IC hard "A system <code>?s</code> must be described by a functional and non functional
spec."

;;;73 IC
(=> (and (Input ?p)
                        (Input ?q)
                (Data ?p)
                (Data ?q)
                (System ?a)
                (System ?b)
                (TraceabilityItem ?x)
                (hasInput ?a ?p)
                (hasInput ?b ?q)
                (hasTraceItem ?p ?x)
                (interopsWith ?a ?b)
                (SameAs ?p ?q))
        (exists (?y)
                        (and (MI.TraceabilityItem ?y)
                                        (hasTraceItem ?q ?y)
                                        (SameAs ?x ?y))))
:IC hard "Traceability items <code>?x</code> <code>?y</code> must be consistent between
interoperating systems <code>?a</code> <code>?b</code>."

;;;72a IC
(=> (and (MI.Data ?x)
                (MI.Output ?z)
                (= ?x ?z))
        (exists (?y)
                (and (MI.TraceabilityItem ?y)
                    (MI.hasTraceItem ?z ?y))))
:IC hard "If an entity is a data output <code>?x</code> individual then it must have a
traceability item <code>?y</code> individual defined."
```

```
;;;72b IC
(=> (and (supTC ?x Data)
                (supTC ?x Output))
        (exists (?y)
                (and (supTC ?y TraceabilityItem)
                    (hasTraceItem ?x ?y))))
:IC hard "If an entity is a data output type then it must have a traceability item type defined."

;;;69 IC
(=> (and (System ?s)
                (FunctionalSpecification ?fs)
                (hasConstraint ?s ?fs))
        (exists (?m ?mt)
                (and (Metric ?m)
                        (MetricTarget ?mt)
                        (specifies ?fs ?m)
                        (specifies ?fs ?mt))))
:IC soft "A system <code>?s</code> should have a functional specification
<code>?fs</code> that describes metrics <code>?m</code> or metric targets
<code>?mt</code>, if multiple functional specs exist not all may contain metrics."

;;;54 IC
(=> (Data ?d)
        (exists (?rc)
                (and (or (RetentionCategory ?rc)
                                (supTC ?rc RetentionCategory))
                        (hasRetentionCategory ?d ?rc))))
:IC soft "Data individuals should have a data retention category type or instance."

;;;55 IC
(=> (Data ?d)
        (exists (?ec)
                (and (or (ExportControlCategory ?ec)
                                (supTC ?ec ExportControlCategory))
                        (hasExportControlCategory ?d ?ec))))
:IC soft "Data individuals should have an export control category type or individual."

;;;56 IC
(=> (Data ?d)
        (exists (?sc)
                (and (or (SecurityCategory ?sc)
                                (supTC ?sc SecurityCategory))
                        (hasSecurityCategory ?d ?sc))))
:IC soft "Data individuals should have a security control category type or individual."


;;;61 IC.
(=> (and (System ?s)
                (Input ?i)
                (Output ?o)
        (hasSecurityCategory ?s MI.Public)
                (hasInput ?s ?i)
                (hasOutput ?s ?o))
            (and (hasSecurityCategory ?i MI.Public)
                (hasSecurityCategory ?o MI.Public)))
:IC hard "Systems <code>?s</code> that have security category of public can only input or
output public category data."

;;;================================================
;;; IRs - 1st level of specialisation from core model
;;;================================================

;;;80 IR
(=> (and (System ?s)
        (Input ?i)
        (Output ?o)
        (hasInput ?s ?i)
        (hasOutput ?s ?o)
        (not (hasStatus ?i ApprovedStatus)))
   (hasStatus ?o MI.UnapprovedStatus))
:rem "If a system input is not approved the system output is not approved."

;;;27 IR
(=> (and (System ?x)
                (System ?y)
                (Metric ?m)
                (hasInput ?x ?d1)
                (hasInput ?x ?d2)
                (hasOutput ?x ?m)
                (hasOutput ?y ?m))
            (SameAs ?d1 ?d2))
:rem "If two systems output the same metric their input should be the same."

;;;;16 IR
(=> (and (supTC ?cd ChangeDecision)
        (supTC ?d Data)
        (modifys ?cd ?d))
   (hasStatus ?d MI.UnapprovedStatus))
:rem "Any <sym>Data</sym> type affected by a <sym>ChangeDecision</sym> type is of
<sym>UnapprovedStatus</sym>."

;;;10 IR
(=> (and (Data ?d)
        (Person ?p)
        (hasOpAuthorityLevel ?d ?x)
        (hasOpAuthorityLevel ?p ?y)
        (or (SameAs ?x ?y)
          (higherThan ?x ?y)))
```

```
   (mayAccess ?p ?d))
:rem "If a person individual has an op auth level which is either equivalent to or higher than
that of a data individual, then the person has access to the data."

;;;11 IR
(=> (and (Data ?d)
        (Person ?p)
        (hasTechAuthorityLevel ?d ?x)
        (hasTechAuthorityLevel ?p ?y)
        (or (SameAs ?x ?y)
          (higherThan ?x ?y)))
   (mayAccess ?p ?d))
:rem "If a person individual has an tech auth level which is either equivalent to or higher than
that of a data individual, then the person has access to the data."

;;;8 IR
(=> (and (Person ?p)
                    (not (exists (?ta)
                                        (hasTechAuthorityLevel ?p ?ta))))
            (hasTechAuthorityLevel ?p MI.User))
:rem "If some person <code>?p</code> individual does not have some associated technical
authority level individual, then it should default to User."

;;;9 IR
(=> (and (Person ?p)
                    (not (exists (?oa)
                                        (hasOpAuthorityLevel ?p ?oa))))
            (hasOpAuthorityLevel ?p MI.Employee))
:rem "If some person <code>?p</code> individual does not have some associated
Operational authority level individual, then it should default to Employee."

;;;17a IR
(=> (and (Data ?d)
                    (not (exists (?s)
                                        (hasStatus ?d ?s))))
            (hasStatus ?d MI.UnapprovedStatus))
:rem "If some data individual does not have some associated status type/individual, then it
should default to UnapprovedStatus."

;;;17b IR
(=> (and (supTC ?d Data)
                    (not (exists (?s)
                                        (hasStatus ?d ?s))))
            (hasStatus ?d MI.UnapprovedStatus))
:rem "If some data type does not have some associated status type/individual, then it should
default to UnapprovedStatus."

;;;79a IR
(=> (and (Data ?d)
                    (not (exists (?sc)
                                        (hasSecurityCategory ?d ?sc))))
            (hasSecurityCategory ?d MI.Private))
:rem "If some data individual does not have some associated security category
type/individual, then it should default to Private."

;;;79b IR
(=> (and (supTC ?d Data)
                    (not (exists (?sc)
                                        (hasSecurityCategory ?d ?sc))))
            (hasSecurityCategory ?d MI.Private))
:rem "If some data type does not have some associated security category type/individual,
then it should default to Private."

;;;78a IR
(=> (and (Data ?d)
                    (not (exists (?ec)
                                        (hasExportControlCategory ?d ?ec))))
            (hasExportControlCategory ?d MI.ExportControlled))
:rem "If some data individual does not have some associated export control category
type/individual, then it should default to ExportControlled."

;;;78b IR
(=> (and (supTC ?d Data)
                    (not (exists (?ec)
                                        (hasExportControlCategory ?d ?ec))))
            (hasExportControlCategory ?d MI.ExportControlled))
:rem "If some data type does not have some associated export control category
type/individual, then it should default to ExportControlled."

;;;77a IR
(=> (and (Data ?d)
                    (not (exists (?rc)
                                        (hasRetentionCategory ?d ?rc))))
            (hasRetentionCategory ?d MI.NonMandatoryRetention))
:rem "If some data individual does not have some associated retention category
type/individual, then it should default to NonMandatoryRetention."

;;;77b IR
(=> (and (supTC ?d Data)
                    (not (exists (?rc)
                                        (hasRetentionCategory ?d ?rc))))
            (hasRetentionCategory ?d MI.NonMandatoryRetention))
:rem "If some data type does not have some associated retention category type/individual,
then it should default to NonMandatoryRetention."

;;;58a IR
(=> (and (System ?s)
                    (not (exists (?sc)
```

```
                                        (hasSecurityCategory ?s ?sc))))
        (hasSecurityCategory ?s MI.Public))
:rem "If some system individual does not have some associated security category
type/individual, then it should default to Public."

;;;58b IR
(=> (and (supTC ?s System)
                (not (exists (?sc)
                                        (hasSecurityCategory ?s ?sc))))
        (hasSecurityCategory ?s MI.Public))
:rem "If some system type does not have some associated security category type/individual,
then it should default to Public."

;;;59a IR
(=> (and (System ?s)
                (not (exists (?ec)
                    (hasExportControlCategory ?s ?ec))))
        (hasExportControlCategory ?s MI.ExportControlled))
:rem "If some system individual does not have some associated export control category
type/individual, then it should default to ExportControlled."

;;;59b IR
(=> (and (supTC ?s System)
                (not (exists (?ec)
                    (hasExportControlCategory ?s ?ec))))
        (hasExportControlCategory ?s MI.ExportControlled))
:rem "If some system type does not have some associated export control category
type/individual, then it should default to ExportControlled."

;;;60a IR
(=> (and (System ?s)
                (not (exists (?rc)
                                        (hasRetentionCategory ?s ?rc))))
        (hasRetentionCategory ?s MI.NonMandatoryRetention))
:rem "If some system individual does not have some associated retention category
type/individual, then it should default to NonMandatoryRetention."

;;;60b IR
(=> (and (supTC ?s System)
                (not (exists (?rc)
                                        (hasRetentionCategory ?s ?rc))))
        (hasRetentionCategory ?s MI.NonMandatoryRetention))
:rem "If some system type does not have some associated retention category type/individual,
then it should default to NonMandatoryRetention."

Metric.KFL
;;; Start Date: 24th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;=================================================
;;; Metric Classes - Level 1
;;;=================================================

:Prop FIrstOrderMetric
:Inst Type
:sup Metric
:name "FIrst Order Metric"
:rem " ."

:Prop SecondOrderMetric
:Inst Type
:sup Metric
:name "Second Order Metric"
:rem " ."

:Prop ThirdOrderMetric
:Inst Type
:sup Metric
:name "Third Order Metric"
:rem " ."

:Prop QualityMetric
:Inst Type
:sup Metric
:name "Quality Metric"
:rem " ."

:Prop PerformanceMetric
:Inst Type
:sup Metric
:name "Performance Metric"
:rem " ."

:Prop ProjectMetric
:Inst Type
:sup Metric
:name "Project Metric"
:rem " ."

:Prop ProductMetric
:Inst Type
:sup Metric
:name "Product Metric"
:rem " ."
```

```
:Prop ReliabilityMetric
:Inst Type
:sup Metric
:name "Reliability Metric"
:rem " ."

:Prop ProcessMetric
:Inst Type
:sup Metric
:name "Process Metric"
:rem " ."

:Prop FinancialMetric
:Inst Type
:sup Metric
:name "Financial Metric"
:rem " ."

:Prop SupplierMetric
:Inst Type
:sup Metric
:name "Supplier Metric"
:rem " ."

:Prop CustomerMetric
:Inst Type
:sup Metric
:name "Customer Metric"
:rem " ."

;;;================================================
;;; Metric Classes -  Quality
;;;================================================

                    :Prop RejectRateMetric
                    :Inst Type
                    :sup QualityMetric
                    :name "Reject Rate Metric"
                    :rem " ."

                    :Prop YieldMetric
                    :Inst Type
                    :sup QualityMetric
                    :name "Yield Metric"
                    :rem " ."

                    :Prop CustomerServiceMetric
                    :Inst Type
                    :sup QualityMetric
                    :name "Customer Service Metric"
                    :rem " ."

                    :Prop ComplianceMetric
                    :Inst Type
                    :sup QualityMetric
                    :name "Compliance Metric"
                    :rem " ."

                    :Prop ConsistencyMetric
                    :Inst Type
                    :sup QualityMetric
                    :name "Consistency Metric"
                    :rem " ."

                    :Prop RFTMetric
                    :Inst Type
                    :sup QualityMetric
                    :name "Right 1st Time Metric"
                    :rem " ."

;;;================================================
;;; Metric Classes -  Performance
;;;================================================

                    :Prop AvailabilityMetric
                    :Inst Type
                    :sup PerformanceMetric
                    :name "Availability Metric"
                    :rem " ."

                    :Prop ThroughputMetric
                    :Inst Type
                    :sup PerformanceMetric
                    :name "Throughput Metric"
                    :rem " ."

                    :Prop UptimeMetric
                    :Inst Type
                    :sup PerformanceMetric
                    :name "Uptime Metric"
                    :rem " ."

                    :Prop UtilisationMetric
                    :Inst Type
                    :sup PerformanceMetric
                    :name "Utilisation Metric"
                    :rem " ."
```

```
:Prop DeliveryMetric
:Inst Type
:sup PerformanceMetric
:name "Delivery Metric"
:rem " ."

:Prop InventoryMetric
:Inst Type
:sup PerformanceMetric
:name "Inventory Metric"
:rem " ."

:Prop SystemPerformanceMetric
:Inst Type
:sup PerformanceMetric
:name "System Performance Metric"
:rem " ."

:Prop SystemCapacityMetric
        :Inst Type
        :sup SystemPerformanceMetric
        :name "System Capacity Metric"
        :rem " ."

        :Prop SystemLatencyMetric
        :Inst Type
        :sup SystemPerformanceMetric
        :name "System Latency Metric"
        :rem " ."

        :Prop SystemResponseTimeMetric
        :Inst Type
        :sup SystemPerformanceMetric
        :name "System Response Time Metric"
        :rem " ."

        :Prop InfrastructurePerformanceMetric
        :Inst Type
        :sup SystemPerformanceMetric
        :name "Infrastructure Performance Metric"
        :rem " ."

        :Prop HardwarePerformanceMetric
        :Inst Type
        :sup SystemPerformanceMetric
        :name "HardwarePerformanceMetric"
        :rem " ."

                :Prop SoftwarePerformanceMetric
                :Inst Type
                :sup SystemPerformanceMetric
                :name "Software Performance Metric"
                :rem " ."

                        :Prop SystemStorageCapacityMetric
                        :Inst Type
                        :sup SystemCapacityMetric
                        :name "System Storage Capacity Metric"
                        :rem " ."

                        :Prop SystemFlowCapacityMetric
                        :Inst Type
                        :sup SystemCapacityMetric
                        :name "System Flow Capacity Metric"
                        :rem " ."

;;;================================================
;;; Metric Classes -  Misc
;;;================================================

                :Prop RiskMetric
                :Inst Type
                :sup ProjectMetric
                :name "Risk Metric"
                :rem " ."

                :Prop MilestoneMetric
                :Inst Type
                :sup ProjectMetric
                :name "Milestone Metric"
                :rem " ."

                :Prop FeatureMetric
                :Inst Type
                :sup ProductMetric
                :name "Feature Metric"
                :rem " ."

                :Prop CapabilityMetric
                :Inst Type
                :sup ProcessMetric
                :name "Capability Metric"
                :rem " ."

                :Prop OperationMetric
                :Inst Type
                :sup ProcessMetric
```

```
:name "Operation Metric"
:rem " ."

        :Prop ProcessCapabilityMetric
        :Inst Type
        :sup CapabilityMetric
        :name "Process Capability Metric"
        :rem " ."

        :Prop ActivityMetric
        :Inst Type
        :sup OperationMetric
        :name "Activity Metric"
        :rem " ."
```

Person.KFL

```
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;================================================
;;; Person Classes - Level 1
;;;================================================

:Prop Operator
:Inst Type
:sup Person
:name "Operator"
:rem "An Operator is a Person responsible for operating or carrying out the Process."

:Prop Designer
:Inst Type
:sup Person
:name "Designer"
:rem "A Designer is the Person responsible for the design or specification for an item."

:Prop ManufacturingEngineer
:Inst Type
:sup Person
:name "Manufacturing Engineer"
:rem "A Manufacturing Engineer is responsible for creating a manufacturing specification,
process and technical package that can be carried out by the Operator to realise the
Designers design or specification for an item ."

:Prop MaintenanceEngineer
:Inst Type
:sup Person
:name "Maintenance Engineer"
:rem "A Maintenance Engineer is responsible for designing and carrying out Maintenance
Processes ."

:Prop Supervisor
:Inst Type
:sup Person
:name "Supervisor"
:rem "Supervisors have elevated Authority Levels and are responsible for Marshalling and
supervising resources and processes to ensure compliance to processes and plans. These
activities also tend to include Metric Design and Metric Target setting and Monitoring."

:Prop MaterialController
:Inst Type
:sup Person
:name "Material Controller"
:rem "Material Controllers are responsible for the launch, flow and management of Material
into and through a Manufacturing Process and its delivery to the next stage of the Supply
Chain in line with the Production Plan. Material Controllers are responsible for the material
management i.e. its whereabouts, and stage through the manufacturing process including
ensuring the consumed and combined materials are available at the appropriate process
steps but are not responsible for the Manufacturing Process itself."

:Prop FinancePerson
:Inst Type
:sup Person
:name "Finance Person"
:rem "A Finance person is one that works to ensure the operational and business finance
Targets are met."

:Prop ITPerson
:Inst Type
:sup Person
:name "IT Person"
:rem "An Information Technology Person is one who is skilled in the development,
deployment or support of IT Systems or elements of IT Systems ."

:Prop LogisticsPerson
:Inst Type
:sup Person
:name "Logistics Person"
:rem "Logistics Personnel are responsible for movement of products throughout an Enterprise
or Supply Chain (as opposed to material controllers who control the )."
```

Prediction.KFL

```
;;; Start Date: 26th September 2011
;;; Author: N Hastilow
;;; Version 1
```

;;; Version 1.1 Remove Sup Analysis from RiskAnalysis as it is a predictive analysis

:Use MI

;;;===
;;; Prediction Classes - Level 1
;;;===

:Prop Estimate
:Inst Type
:sup Prediction
:name "Estimate"
:rem "An Estimate is an approximate prediction of a future value which is based on limited information and judgment based on previous experience or associated trends."

:Prop FMEA
:Inst Type
:sup Prediction
:name "Failure Mode and Effect Analysis"
:rem "Failure Mode and Effect Analysis is the prediction of potential future failure mechanisms and the potential effects of those mechanisms."

:Prop RiskAnalysis
:Inst Type
:sup Prediction
:name "RiskAnalysis"
:rem " A risk analysis idenitifys risks using and If.....Then..... format, giving a ranking to the probability and impact of each risk allong with mitigations and treatments to reduce or resolve it."

:Prop Simulation
:Inst Type
:sup Prediction
:name "Simulation"
:rem "A Simulation is a representation of a system, enterprise, process etc which behaves in a similar way to the actual enterprise etc. Simulations are used to investigate and rapidly predict the response or outcomes of given scenarios with reduced risk, cost. "

:Prop PFMEA
:Inst Type
:sup FMEA
:name "Process Failure Mode and Effect Analysis"
:rem "Process Failure Mode and Effect Analysis is the prediction of potential future failure mechanisms and the potential effects of those mechanisms within a process ."

:Prop RoughOrderOfMagnitude
:Inst Type
:sup Estimate
:name "Rough Order Of Magnitude"
:rem "A Rough Order Of Magnitude Estimate is a very rough approximate prediction of a future value which is based on very limited information and judgment based on previous experience or associated trends. This represents a high risk Estimate due to limited information or guidance."

:Prop Quote
:Inst Type
:sup Estimate
:name "Quote"
:rem "A quote is an estimate based on detailed analysis of the the requirements and solution work content, quote usualy have a limited period of valdidity, but can also be invalidated by any specification Change."

Qualities.KFL
; Copyright 2008 Ontology Works, Inc. All rights reserved.

:Name "Qualities"
:Description "A theory of qualities"
:Use MLO

:Rel hasCtx
:Inst BinaryRel
:Sig Top Top

; author: kohl
; date: 1-11-2008
; depends on: ULO

; This theory defines a class of Qualities and well as a meta-class of QualityKinds. For example, Blue is a Quality and Color is a QualityKind.
; This theory also defined relations between Particulars (see ULO Documentation), their Qualities, and applicable QualityKinds.

:Prop Quality
:Inst Type
:Inst IntensionalRel
:sup AbstractEntity
:name "Quality"
:lex "?1 is a quality"
:rem "A quality is an <sym>MLO.AbstractEntity</sym> that can be attributed to a <sym>Particular</sym> via the <sym>hasQuality</sym> relation. Qualities cannot exist apart from the particulars they are attributed to (eg there could be no 'colored blue' quality if there were no blue-colored things). A quality is held in common by all those things the quality is attributed to. (eg the same quality instance is held by all blue-colored things)."

:Prop QualityKind
:Inst MetaProperty
:sup Type

```
        :name "Quality Type"
        :lex  "?1 is a kind of quality"
        :rem  "All and only subclasses of <sym>Quality</sym> are instances
(<sym>RootCtx.inst</sym>) of QualityKind."
        :metaPropFor Quality

        (=> (supTC ?x Quality)
                (inst ?x QualityKind))
        :IC hard "All subclasses of Quality must be instances of QualityKind, but ?x is not."

        :Rel  hasQuality
        :Inst AsymmetricBR
        :Sig  Particular Quality
        :name "has quality"
        :lex  "?1 is attributed quality ?2"
        :rem  "The <sym>RootCtx.Relation</sym> that holds between a
<sym>Particular</sym> and a <sym>Quality</sym> attributed to it."

        :Rel  hasQualityKind
        :Inst AsymmetricBR
        :Sig  Particular Top
        :name "has quality kind"
        :lex  "?1 is attributed some ?2"
        :rem  "The <sym>RootCtx.Relation</sym> that holds between a
<sym>Particular</sym> and a <sym>QualityKind</sym> because that particular is attributed
some <sym>Quality</sym> that is an instance (<sym>RootCtx.inst</sym>) of the
QualityKind."
        :exampleRem "If Helen's table is colored blue, then it is true that the table has been
attributed some color:<br>
        given: (hasColor helensTable Blue)<br>
        therefore: (hasQualityKind helensTable Color)<br>
        <br>
        This relation can be used when it is known that some quality has been attributed to
a particular, but only the kind of quality is known.  For example, it may be known that an
<sym>Object</sym> has a price, but it is unknown what that price is.  Then, the following
could be asserted:<br>
        (hasQualityKind object-56 Price)<br>
        This would be new information if, indeed, not everything has a price."

Relations.KFL
;;; Start Date: 13th November 2011
;;; Author: Neil Hastilow
;;; Other Contributors: Tish Chungoora

:Use MI

;;;================================================
;;; Relations - 1st level of specialisation from core model
;;;================================================

:Rel higherThan
:Inst BinaryRel
:Inst IrreflexiveBR
:Inst TransitiveBR
:Sig Top Top

:Rel hasOpAuthorityLevel
:Inst BinaryRel
:Sig Top OpAuthorityLevel
:name "has Operational Authority Level"

:Rel hasTechAuthorityLevel
:Inst BinaryRel
:Sig Top TechAuthorityLevel
:name "has Technical Authority Level"

:Rel hasDataStructure
:Inst BinaryRel
:Sig Top Top
:name "has Data Structure"

:Rel hasMonitor
:Inst BinaryRel
:Sig Top Top
:name "has Monitor"

:Rel hasResponse
:Inst BinaryRel
:Sig Top Response
:name "has Response"

:Rel hasTimescale
:Inst BinaryRel
:Sig Top Top
:name "has Timescale"

:Rel hasSpecification
:Inst BinaryRel
:Sig Top Specification
:name "has Specification"

:Rel hasSecurityCategory
:Inst BinaryRel
:Sig Top Top
:name "has Security Category"

:Rel hasRetentionCategory
:Inst BinaryRel
```

```
:Sig Top Top
:name "has Retention Category"

:Rel hasExportControlCategory
:Inst BinaryRel
:Sig Top Top
:name "has Export Control Category"

:Rel modifys
:Inst BinaryRel
:Sig Top Top
:name "modifys"

:Rel mayAccess
:Inst BinaryRel
:Sig Top Top
:name "may Access"

:Rel SameAs
:Inst BinaryRel
:Inst EquivalenceBR
:Sig Top Top
:name "Same As"

Resource.KFL
;;; Start Date: 25th October 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;================================================
;;; Resource  Classes - Level 1
;;;================================================

:Prop SystemResource
:Inst Type
:sup Resource
:name "System Resource"
:rem "System Resources are the things used and required by a system to perform the
process of turning inputs into outputs."

Status.KFL
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;================================================
;;; Status Classes - Level 1
;;;================================================

:Prop ObsoleteStatus
:Inst Type
:sup Status
:name "Obsolete Status"
:rem "Obsolete Status applies to items which are no longer valid and should not be used but
need to be kept for reference as part of a change history. Obsolete and Archived status are
commonly used for items once they have been Approved or Conditionally Approved, as
deleting approved items is rarely appropriate as they may have been used requiring a level of
traceability"

:Prop ArchievedStatus
:Inst Type
:sup Status
:name "Archieved Status"
:rem "Archieved Status applies to items which whilst still valid may not be immediately
relevant or in regular use. Archieved items are retained in large but slow archives which often
take considerable time or effort to retrieve the items, this is appropriate as the limited
relevance of the items mean they are infrequently retrieved. Obsolete and Archieved status
are commonly used for items once they have been Approved or Conditionally Approved, as
deleting approved items is rarely appropriate as they may have been used requiring a level of
traceability"

:Prop DeletedStatus
:Inst Type
:sup Status
:name "Deleted Status"
:rem " Deleted Status applied to items which have been removed or destroyed. This status
may be applied to the item records to indicate the item once existed but can no longer be
accessed without accessing backup records. Obsolete and Archived status are commonly
used for items once they have been Approved or Conditionally Approved, as deleting
approved items is rarely appropriate as they may have been used requiring a level of
traceability"

:Prop ReturnStatus
:Inst Type
:sup Status
:name "Return Status"
:rem " Return Status is applied to items which are were Unproven or have been Conditionally
Approved for development, and then modified as part of that development before being
returned for review prior to assessment for full approval."

:Prop ApprovedStatus
:Inst Type
:sup Status
:name "Approved Status"
```

```
:rem "Approved Status is applied to items which have been fully and unconditionally
approved for use."

:Prop UnapprovedStatus
:Inst Type
:sup Status
:name "Unapproved Status"
:rem "Unapproved status is the default status for any item prior to full or conditional approval.
This status does not necessarily mean the item needs modification, merely that it has not
been confirmed as suitable for use. Unproven items can either progress to being Fully or
Conditional Approved if suitable or may be deleted if unsuitable. Unapproved items would not
normally be Archived or given Obsolete Status as they should never be used precluding any
mandatory requirement for traceability."

:Prop ConditionallyApprovedStatus
:Inst Type
:sup Status
:name "Conditionally Approved Status"
:rem "Conditionally Approved Status is applied to items that are not suitable for full approval,
but are appropriate for use with limitations."

Sustainment Process.KFL
;;; Start Date: 26th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;================================================
;;; Sustainment Process Classes - Level 1
;;;================================================

:Prop ProcessMaintenance
:Inst Type
:sup SustainmentProcess
:name "Process Maintenance"
:rem " Process Maintenance counteracts change and variation (both common and special
cause) to ensure the continuity of the current process intent.  It is important to note that
Process Maintenance maintains the original intent of the process, so in changing
circumstances a Process Maintenance will introduce change to maintain the process intent"

:Prop Monitor
:Inst Type
:sup SustainmentProcess
:name "Monitor"
:rem " A Monitor can be used to detect the current state, changes and can be used with
analysis to identify trends, trigger reactions and generate predictions ."

:Prop Training
:Inst Type
:sup SustainmentProcess
:name "Training"
:rem "Training is used to maintain or standardise the capability and knowledge of a Person in
changing circumstances. It is possible that the only change is Time, in which case the
Training would be referred to a 're-training'' as it is the repetition of identical training to refresh
the knowledge of a person."

:Prop EquipmentMaintenance
:Inst Type
:sup SustainmentProcess
:name "Equipment Maintenance"
:rem "Equipment Maintenance counteracts change and variation (both common and special
cause) to ensure the continuity of the current Equipment fit form and function.  It is important
to note that Equipment Maintenance maintains the original intent of the Equipment, so in
changing circumstances a Equipment Maintenance will introduce change to maintain the
Equipment intent ."

;;;================================================
;;; Equipment Maintenance Level Classes
;;;================================================

:Prop ToolMaintenance
:Inst Type
:sup EquipmentMaintenance
:name "Tool Maintenance"
:rem "Tool Maintenance counteracts change and variation (both common and special cause)
to ensure the continuity of the current Tool fit form and function.  It is important to note that
Tool Maintenance maintains the original intent of the Tool, so in changing circumstances a
Tool Maintenance will introduce change to maintain the Tool intent (eg regrinding a cutting
tool)."

:Prop ITMaintenance
:Inst Type
:sup EquipmentMaintenance
:name "IT Maintenance"
:rem "IT Maintenance counteracts change and variation (both common and special cause) to
ensure the continuity of the current IT specification, capability and function.  It is important to
note that IT Maintenance maintains the original intent of the IT, so in changing circumstances
a IT Maintenance will introduce change to maintain the IT specification intent ."

:Prop MachineMaintenance
:Inst Type
:sup EquipmentMaintenance
:name "Machine Maintenance"
:rem "Machine Maintenance counteracts change and variation (both common and special
cause) to ensure the continuity of the current Machine fit form and function.  It is important to
note that Machine Maintenance maintains the original intent of the Machine, so in changing
```

```
circumstances a Machine Maintenance will introduce change to maintain the Machine intent
."

:Prop InfrastructureMaintenance
:Inst Type
:sup EquipmentMaintenance
:name "Infrastructure Maintenance"
:rem "Infrastructure Maintenance counteracts change and variation (both common and
special cause) to ensure the continuity of the current Infrastructure specification, capability
and function.  It is important to note that Infrastructure Maintenance maintains the original
intent of the Infrastructure, so in changing circumstances a Infrastructure Maintenance will
introduce change to maintain the Infrastructure specification intent ."

System.KFL
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;================================================
;;; System  Classes - Level 1
;;;================================================

:Prop StandardSystem
:Inst Type
:sup System
:name "Standard System"
:rem " "

:Prop NonStandardSystem
:Inst Type
:sup System
:name "Non Standard System"
:rem " "

                    :Prop TacticalSystem
                    :Inst Type
                    :sup NonStandardSystem
                    :name "Tactical System"
                    :rem " "

;;;================================================
;;; Standard System Level Classes
;;;================================================

                    :Prop VisualisationSystem
                    :Inst Type
                    :sup StandardSystem
                    :name "Visualisation System"
                    :rem " "

                    :Prop DataCollectionSystem
                    :Inst Type
                    :sup System
                    :name "Data Collection System"
                    :rem " "

                    :Prop DNCSystem
                    :Inst Type
                    :sup StandardSystem
                    :name "DNC System"
                    :rem " "

                    :Prop ProgrammingSystem
                    :Inst Type
                    :sup StandardSystem
                    :name "Programming System"
                    :rem " "

                    :Prop ControlSystem
                    :Inst Type
                    :sup StandardSystem
                    :name "Control System"
                    :rem " "

                    :Prop ReportingSystem
                    :Inst Type
                    :sup StandardSystem
                    :name "Reporting System"
                    :rem " "

                    :Prop CollaborationSystem
                    :Inst Type
                    :sup StandardSystem
                    :name "Collaboration System"
                    :rem " "

                    :Prop StorageSystem
                    :Inst Type
                    :sup StandardSystem
                    :name "Storage System"
                    :rem " "

                    :Prop AnalysisSystem
                    :Inst Type
                    :sup StandardSystem
                    :name "Analysis System"
```

```
:rem " "

:Prop AutomationSystem
:Inst Type
:sup StandardSystem
:name "Automation System"
:rem " "

:Prop Infrastructure
:Inst Type
:sup StandardSystem
:name "Infrastructure"
:rem " "

        :Prop ArchivingSystem
        :Inst Type
        :sup StorageSystem
        :name "Archiving System"
        :rem " "

        :Prop DocumentationSystem
        :Inst Type
        :sup StorageSystem
        :name "Documentation System"
        :rem " "

        :Prop DatabaseSystem
        :Inst Type
        :sup StorageSystem
        :name "Database System"
        :rem " "

        :Prop AdaptiveManufactureSystem
        :Inst Type
        :sup AutomationSystem
        :name "Adaptive Manufacture System"
        :rem " "

        :Prop SCADASystem
        :Inst Type
        :sup VisualisationSystem
        :name "SCADA System"
        :rem " "
```

Target.KFL

```
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1
;;; Version 2 Specification and sub concepts added - demoted from core concept status

:Use MI

;;;=================================================
;;; Target Classes - Level 1
;;;=================================================

:Prop MetricTarget
:Inst Type
:sup Target
:name "Metric Target"
:rem " A Metric Target represents the ideal value or state for a specific Metric. Metric Targets
are set in line with the overall enterprise objectives."

:Prop Plan
:Inst Type
:sup Target
:name "Plan"
:rem "A Plan represents the target sequence of activities and Timescale to achieve a specific
outcome."

:Prop Vision
:Inst Type
:sup Target
:name "Vision"
:rem "A Vision describes the desired final state. Visions should be high level and avoid
specific detail, instead creating a overall picture of a totally successful set of outcomes."

;;;=================================================
;;; Plan Classes
;;;=================================================

        :Prop Roadmap
        :Inst Type
        :sup Plan
        :name "Roadmap"
        :rem "A Roadmap is a chronological representation of events, milestones
or achievements where the connecting chronological Roadmap represents a consistent
theme or strategy ."

        :Prop ProcessPlan
        :Inst Type
        :sup Plan
        :name "Process Plan"
        :rem "A Process Plan represents the target sequence of activities and
Timescale to achieve a specific Process outcome ."

        :Prop ProductionPlan
        :Inst Type
```

:sup Plan
:name "Production Plan"
:rem "A Production Plan represents the target production activities (such as resourcing plan, material launch and processing) and Timescale to achieve a specific delivery requirement ."

:Prop ReactionPlan
:Inst Type
:sup Plan
:name "Reaction Plan"
:rem "A Reaction Plan details the prescribed Reaction activities or steps to a specific Reaction trigger such as an alarm, or other Event."

:Prop InspectionPlan
:Inst Type
:sup Plan
:name "Inspection Plan"
:rem "An Inspection Plan details the sequence of inspection activities required to fully validate that the target of the inspection meets the Specification."

Timescale.KFL
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;==
;;; Timescale Classes - Level 1
;;;==

:Prop RealTimeTimescale
:Inst Type
:sup Timescale
:name "Real Time Timescale"
:rem " The Real Time Timescale represents a zero or near zero delay between any given instant and an associated event."

:Prop ExecutionTimescale
:Inst Type
:sup Timescale
:name "Execution Timescale"
:rem "Execution Timescales are measured in hours and days and represent the appropriate timescales for operational execution activities."

:Prop ValidityTimescale
:Inst Type
:sup Timescale
:name "Validity Timescale"
:rem " The Validity Timescale represents the timescale over which an entity, logical statement or decision is valid. After this timescale the entity etc becomes invalid therefore should no longer be used or considered reliable."

:Prop ControlTimescale
:Inst Type
:sup Timescale
:name "Control Timescale"
:rem "Control Timescales are measured in Milliseconds to minutes and represent the appropriate timescales for operational control activities ."

:Prop EnterpriseTimescale
:Inst Type
:sup Timescale
:name "Enterprise Timescale"
:rem "Enterprise Timescales are measured in weeks and months and represent the appropriate timescales for enterprise execution activities ."

:Prop RetentionTimescale
:Inst Type
:sup Timescale
:name "Retention Timescale"
:rem "The Retention Timescale represents the timescale for which an item must be retained or retrievable. The implications of failing to retain for this length of time depend on the Data Retention Category."

(ValidityTimescale KnownValidityTimescale)
(ValidityTimescale UnknownValidityTimescale)

Traceability Item.KFL
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;==
;;; Timescale Classes - Level 1
;;;==

:Prop RealTimeTimescale
:Inst Type
:sup Timescale
:name "Real Time Timescale"
:rem " The Real Time Timescale represents a zero or near zero delay between any given instant and an associated event."

```
:Prop ExecutionTimescale
:Inst Type
:sup Timescale
:name "Execution Timescale"
:rem "Execution Timescales are measured in hours and days and represent the appropriate timescales for operational execution activities."

:Prop ValidityTimescale
:Inst Type
:sup Timescale
:name "Validity Timescale"
:rem " The Validity Timescale represents the timescale over which an entity, logical statement or decision is valid. After this timescale the entity etc becomes invalid therefore should no longer be used or considered reliable."

:Prop ControlTimescale
:Inst Type
:sup Timescale
:name "Control Timescale"
:rem "Control Timescales are measured in Milliseconds to minutes and represent the appropriate timescales for operational control activities ."

:Prop EnterpriseTimescale
:Inst Type
:sup Timescale
:name "Enterprise Timescale"
:rem "Enterprise Timescales are measured in weeks and months and represent the appropriate timescales for enterprise execution activities ."

:Prop RetentionTimescale
:Inst Type
:sup Timescale
:name "Retention Timescale"
:rem "The Retention Timescale represents the timescale for which an item must be retained or retrievable. The implications of failing to retain for this length of time depend on the Data Retention Category."

(ValidityTimescale KnownValidityTimescale)
(ValidityTimescale UnknownValidityTimescale)

Visualisation.KFL
;;; Start Date: 26th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;=================================================

;;; Visualisation Classes - Level 1
;;;=================================================

:Prop Report
:Inst Type
:sup Visualisation
:name "Report"
:rem "A Report visualises and comunicates Metrics, Data, Analysis etc. Reports are designed for a specific purpose, and will combine any information or data to that end."

Analysis.KFL
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;=================================================
;;; Analysis  Classes - Level 1
;;;=================================================

:Prop StatisticalAnalysis
:Inst Type
:sup Analysis
:name "Statisitical Process Control"
:rem " Statistical Analysis is the characterisation of a population by a mathmatical model against which the the conformance of the population can be judged. This can be extended to Statistical Process Control which uses this model to idenify anamolous occurances and the propability of any particular outcome"

:Prop BooleanAnalysis
:Inst Type
:sup Analysis
:name "Boolean Analysis"
:rem " Boolean Analysis refers to the analysis of a particular event or outcome in boolean or logical terms such as true or false. This would cover most rule based signals or automated systems and requires predefined logical criteria against which to analyse a result and give an outcome."

:Prop Diagnosis
:Inst Type
:sup Analysis
:name "Diagnosis"
:rem "Diagnosis is the retrospective Analysis that enables the understanding of an Event or occurrence including causal factors "

                    :Prop RootCauseAnalysis
                    :Inst Type
                    :sup Diagnosis
```

```
                    :name "Root Cause Analysis"
                    :rem "Root Cause Analysis is used to find the prime cause of an Event of
occurrence "

Authority.KFL
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;================================================
;;; Authority Level Classes - Level 1
;;;================================================

:Prop TechAuthorityLevel
:Inst Type
:sup AuthorityLevel
:name "Technical Authority Level"
:rem "Technical authority level represents the level of authority of a person or system over
technical aspects of the product, process, organisation eg those that can affect the fit, form or
function of the deliverable. Decisions that have higher risk implications generally require
higher levels of Technical Authority level. Operational and Technical Authority levels need to
be balanced appropriately as they Operational and Technical objectives often conflict."

:Prop OpAuthorityLevel
:Inst Type
:sup AuthorityLevel
:name "Operational Authority Level"
:rem "Operational authority level represents the level of authority of a person or system over
the operational aspects of the enterprise eg those that can affect the cost, delivery, resource
allocation etc. Operational and Technical Authority levels need to be balanced appropriately
as they Operational and Technical objectives often conflict."

;;;================================================
;;; Technical Authority Level Classes
;;;================================================

                    :Prop AdministratorTechAuthorityLevel
                    :Inst Type
                    :sup TechAuthorityLevel
                    :name "Administrator"
                    :rem "Administrator is the highest Technical Authority Level.
Administrators have full authority to make changes and decisions with minimal constraint.
The application of Authority Level is tightly controlled due to the associated risks "

                    :Prop UserTechAuthorityLevel
                    :Inst Type
                    :sup TechAuthorityLevel
                    :name "User"
                    :rem " User is the lowest Technical Authority Level, Users have limited
ability to make minor decisions and changes  and only within specified limits. User is the
default level of access for any individual or system with technical authority."

                    :Prop SuperuserTechAuthorityLevel
                    :Inst Type
                    :sup TechAuthorityLevel
                    :name "Superuser"
                    :rem " Superuser is a medium level of Technical Authority  between User
and Administrator. Superusers have the ability to modify configuration and make decisions or
changes within limits that are defined by Administrators"

;;;================================================
;;; Operational Authority Level Classes
;;;================================================

                    :Prop ExecutiveOpAuthorityLevel
                    :Inst Type
                    :sup OpAuthorityLevel
                    :name "Executive"
                    :rem " Executive is the highest Operational Authority Level. Executives
have full authority to make decisions over an enterprise strategy and direction as well as all
resources within the enterprise. "

                    :Prop ManagerOpAuthorityLevel
                    :Inst Type
                    :sup OpAuthorityLevel
                    :name "Manager"
                    :rem "Manager is an Operational Authority Level below Executive and
above Team Leader. Managers have significant authority to make decisions within
subdivisions of the enterprise"

                    :Prop LeaderOpAuthorityLevel
                    :Inst Type
                    :sup OpAuthorityLevel
                    :name "Leader"
                    :rem "Leader is an Operational Authority Level below Manager and
above Employee. Leaders have limited levels of authority to make decisions within aspects of
subdivisions of an enterprise. Leader is only one level of elivation above Employee, any level
above this will be a Manager of above."

                    :Prop EmployeeOpAuthorityLevel
                    :Inst Type
                    :sup OpAuthorityLevel
                    :name "Employee"
                    :rem "  Employee is the lowest Operational Authority Level. Employees
have little or no authority within the enterprise, and this tends to be limited within defined
options rather than free choices "
```

Collaboration.KFL
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;==
;;; Collaboration Classes - Level 1
;;;==

:Prop InternalCollaboration
:Inst Type
:sup Collaboration
:name "Internal Collaboration"
:rem " Internal Collaboration is the collaboration between entities within an enterprise."

:Prop ExternalCollaboration
:Inst Type
:sup Collaboration
:name "External Collaboration"
:rem " External Collaboration is the collaboration between entities in different enterprises ."

:Prop InternalSystemCollaboration
:Inst Type
:sup InternalCollaboration
:name "Internal System Collaboration"
:rem " Internal Collaboration is the collaboration between Systems within an enterprise."

:Prop ExternalSystemCollaboration
:Inst Type
:sup ExternalCollaboration
:name "External System Collaboration"
:rem "External Collaboration is the collaboration between Systems in different enterprises."

Constraint.KFL
;;; Start Date: 25th October 2011
;;; Author: N Hastilow
;;; Version 1
;;; Version 2 Specification added from Target

:Use MI

;;;==
;;; Constraint Classes - Level 1
;;;==

:Prop Specification
:Inst Type
:sup Constraint
:name "Specification"
:rem "A Specification describes a requirement in appropriate detail. Any solution that is constructed and tested against a suitable specification and passes should meet the customer needs and expectations."

:Prop SystemConstraint
:Inst Type
:sup Constraint
:name "System Constraint"
:rem "System Constraints are the limits, rules and structures within which any process must function."

:Prop SecurityCategory
:Inst Type
:sup Constraint
:name "Security Category"
:rem " "

:Prop RetentionCategory
:Inst Type
:sup Constraint
:name "Data Retention Category"
:rem " "

:Prop ChangeControl
:Inst Type
:sup Constraint
:name "Change Control"
:rem " "

:Prop ExportControlCategory
:Inst Type
:sup Constraint
:name "Export Control"
:rem " "

:Prop MandatoryRetention
:Inst Type
:sup RetentionCategory
:name "Mandatory Retention"
:rem " "

:Prop NonMandatoryRetention
:Inst Type
:sup RetentionCategory

```
:name "Non Mandatory Retention"
:rem " "

:Prop ExportControlled
:Inst Type
:sup ExportControlCategory
:name "Export Controlled"
:rem " "

:Prop NonExportControlled
:Inst Type
:sup ExportControlCategory
:name "Non Export Controlled"
:rem " "

:Prop Public
:Inst Type
:sup SecurityCategory
:name "Public"
:rem " "

:Prop Private
:Inst Type
:sup SecurityCategory
:name "Private"
:rem " "

:Prop Secret
:Inst Type
:sup SecurityCategory
:name "Secret"
:rem " "

:Prop TopSecret
:Inst Type
:sup SecurityCategory
:name "TopSecret"
:rem " "

;;;================================================
;;; Specification Classes
;;;================================================

:Prop ProductSpecification
:Inst Type
:sup Specification
:name "Product Specification"
:rem " ."

:Prop ITSpecification
:Inst Type
:sup Specification
:name "IT Specification"
:rem " ."

:Prop MaterialSpecification
:Inst Type
:sup Specification
:name "Material Specification"
:rem " ."

:Prop FunctionalSpecification
:Inst Type
:sup Specification
:name "Functional Specification"
:rem " ."

:Prop VisualSpecification
:Inst Type
:sup Specification
:name "Visual Specification"
:rem " ."

:Prop ProcessSpecification
:Inst Type
:sup Specification
:name "Process Specification"
:rem " ."

:Prop NonFunctionalSpecification
:Inst Type
:sup Specification
:name "Non Functional Specification"
:rem " ."

:Prop StatementOfRequirements
:Inst Type
:sup Specification
:name "Statement Of Requirements"
:rem " ."
```

Data.KFL

```
;;; Start Date: 24th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI
```

```
;;;================================================
;;; Data Classes - Level 1
;;;================================================

:Prop MaintenanceData
:Inst Type
:sup Data
:name "Maintenance Data"
:rem "Maintnence Data is the data used within a process to maintain or sustain the process.
This data is generaly used in a Maintnence Process or a Report"

:Prop InspectionData
:Inst Type
:sup Data
:name "Inspection Data"
:rem "Inspection Data is the Data used or generated by an inspection process or system."

:Prop ProcessData
:Inst Type
:sup Data
:name "Process Data"
:rem "Process Data is Data used or generated by a process."

:Prop MetaData
:Inst Type
:sup Data
:name "Meta Data"
:rem "Meta Data is Data about the containter of data or data content ."

:Prop OrderData
:Inst Type
:sup Data
:name "Order Data"
:rem "Order Data is Data relating to a Customer order, this can be directly relating to the
specifics of the order or data accumulated throughout the execution or fulfilment of the order."

:Prop TimeData
:Inst Type
:sup Data
:name "Time Data"
:rem "Time Data is Data that places or quantifies and item with relation to time. This includes
absolute or relative chronological ordering or Durration information."

:Prop ResourceData
:Inst Type
:sup Data
:name "Resource Data"
:rem "Resource Data is Data relating to Resources ."

:Prop TraceabilityData
:Inst Type
:sup Data
:name "Traceability Data"
:rem "Tracability Data is Data which can be used to sort, retreive or otherwise stratify other
data. Entitys can accumulate Traceability Data as they are processed as this ensures
tracability throughout the item lifecycle."

:Prop ProductData
:Inst Type
:sup Data
:name "Product Data"
:rem "Product Data is and Data that relates to a specific Product or Product type ."

;;;================================================
;;; Data Classes - Maintenance
;;;================================================

                    :Prop MaintenanceSchedule
                    :Inst Type
                    :sup MaintenanceData
                    :name "Maintenance Schedule"
                    :rem "A Maintenence Schedule is a Plan of Maintenence activities."

                    :Prop MaintenanceLog
                    :Inst Type
                    :sup MaintenanceData
                    :name "Maintenance Log"
                    :rem "A Maintenance Log is a historical record of executed Maintance
Process and relating Maintnence Data."

                                :Prop BreakdownLog
                                :Inst Type
                                :sup MaintenanceLog
                                :name "Breakdown Log"
                                :rem "A Breakdown Log is a historical log of equipment failure."

                                :Prop CalibrationSchedule
                                :Inst Type
                                :sup MaintenanceSchedule
                                :name "Calibration Schedule"
                                :rem "A Calibration Schedule is a Plan of calibration activities
for precision equipment or processes ."

                                :Prop ServiceSchedule
                                :Inst Type
                                :sup MaintenanceSchedule
                                :name "Service Schedule"
```

```
                    :rem "A Service Schedule is a Plan of equipment servicing
activities. These activities should be timed to ensure calibration can be maintained and
breakdowns avoided."

;;;================================================
;;; Data Classes - Inspection
;;;================================================

            :Prop InspectionRecord
            :Inst Type
            :sup InspectionData
            :name "Inspection Record"
            :rem " ."

            :Prop DimensionalData
            :Inst Type
            :sup InspectionData
            :name "Dimensional Data"
            :rem " ."

            :Prop VisualData
            :Inst Type
            :sup InspectionData
            :name "Visual Data"
            :rem " ."

            :Prop MaterialData
            :Inst Type
            :sup InspectionData
            :name "Material Data"
            :rem " ."

                    :Prop DescretePointData
                    :Inst Type
                    :sup DimensionalData
                    :name "Descrete Point Data"
                    :rem " ."

                    :Prop CloudPointData
                    :Inst Type
                    :sup DimensionalData
                    :name "Cloud Point Data"
                    :rem " ."

                    :Prop XRayData
                    :Inst Type
                    :sup VisualData
                    :name "X Ray Data"
                    :rem " ."

            :Prop CoatingData
            :Inst Type
            :sup MaterialData
            :name "Coating Data"
            :rem " ."

            :Prop AlloyData
            :Inst Type
            :sup MaterialData
            :name "Alloy Data"
            :rem " ."

                    :Prop MeasurementMachineData
                    :Inst Type
                    :sup DescretePointData
                    :name "Measurement Machine Data"
                    :rem " ."

                    :Prop GaugeData
                    :Inst Type
                    :sup DescretePointData
                    :name "Gauge Data"
                    :rem " ."

                    :Prop StructuredLightData
                    :Inst Type
                    :sup DimensionalData
                    :name "StructuredLightData"
                    :rem " ."

                    :Prop CTData
                    :Inst Type
                    :sup DimensionalData
                    :name "Computer Tomography Data"
                    :rem " ."

;;;================================================
;;; Data Classes - Process
;;;================================================

            :Prop ProcessParameterData
            :Inst Type
            :sup ProcessData
            :name "Process Parameter Data"
            :rem " ."

            :Prop KPVData
            :Inst Type
```

```
:sup ProcessData
:name "KPV Data"
:rem " ."

:Prop PerformanceData
:Inst Type
:sup ProcessData
:name "Performance Data"
:rem " ."

        :Prop DatacardData
        :Inst Type
        :sup ProcessParameterData
        :name "Datacard Data"
        :rem " ."

        :Prop ProgramData
        :Inst Type
        :sup ProcessParameterData
        :name "Program Data"
        :rem " ."

        :Prop RecipeData
        :Inst Type
        :sup ProcessParameterData
        :name "Recipe Data"
        :rem " ."

        :Prop AvailabilityData
        :Inst Type
        :sup PerformanceData
        :name "Availability Data"
        :rem " ."

        :Prop ThroughputData
        :Inst Type
        :sup PerformanceData
        :name "Throughput Data"
        :rem " ."

        :Prop UptimeData
        :Inst Type
        :sup PerformanceData
        :name "Uptime Data"
        :rem " ."

        :Prop UtilisationData
        :Inst Type
        :sup PerformanceData
        :name "Utilisation Data"
        :rem " ."

:Prop DeliveryData
:Inst Type
:sup PerformanceData
:name "Delivery Data"
:rem " ."

:Prop InventoryData
:Inst Type
:sup PerformanceData
:name "Inventory Data"
:rem " ."

:Prop SystemPerformanceData
:Inst Type
:sup PerformanceData
:name "System Performance Data"
:rem " ."

        :Prop WIPInventoryData
        :Inst Type
        :sup InventoryData
        :name "WIP Inventory Data"
        :rem " ."

        :Prop TotalInventoryData
        :Inst Type
        :sup InventoryData
        :name "Total Inventory Data"
        :rem " ."

        :Prop OnTimeDeliveryData
        :Inst Type
        :sup DeliveryData
        :name "On Time Delivery Data Data"
        :rem " ."

        :Prop SystemCapacityData
        :Inst Type
        :sup SystemPerformanceData
        :name "System Capacity Data"
        :rem " ."

        :Prop SystemLatencyData
        :Inst Type
        :sup SystemPerformanceData
        :name "System Latency Data"
```

```
:rem " ."

:Prop SystemResponseTimeData
:Inst Type
:sup SystemPerformanceData
:name "System Response Time Data"
:rem " ."

:Prop InfrastructurePerformanceData
:Inst Type
:sup SystemPerformanceData
:name "Infrastructure Performance Data"
:rem " ."

:Prop HardwarePerformanceData
:Inst Type
:sup SystemPerformanceData
:name "HardwarePerformanceData"
:rem " ."

:Prop SoftwarePerformanceData
:Inst Type
:sup SystemPerformanceData
:name "Software Performance Data"
:rem " ."

        :Prop SystemStorageCapacityData
        :Inst Type
        :sup SystemCapacityData
        :name "System Storage Capacity Data"
        :rem " ."

        :Prop SystemFlowCapacityData
        :Inst Type
        :sup SystemCapacityData
        :name "System Flow Capacity Data"
        :rem " ."

;;;================================================
;;; Data Classes -  Misc
;;;================================================

    :Prop ProductionOrder
    :Inst Type
    :sup OrderData
    :name "Production Order"
    :rem " ."

    :Prop ProductionHistory
    :Inst Type
    :sup ProductData
    :name "Production History"
    :rem " ."
```

Data Structure.KFL

```
;;; Start Date: 25th September 2011
;;; Author: N Hastilow
;;; Version 1

:Use MI

;;;================================================
;;; Data Structure Classes - Level 1
;;;================================================

:Prop StandardDataStructure
:Inst Type
:sup DataStructure
:name "Standard Data Structure"
:rem " A Standard Data Structure is one which has been fully and rigorously documented and
is designed to be used by multiple instances."

:Prop DataDictionary
:Inst Type
:sup DataStructure
:name "Data Dictionary"
:rem "A Data Dictionary lists all of the identified data items and provides a natural language
description of the data item, its purpose and relationship to other data items or Processes ."

:Prop NonStdDataStructure
:Inst Type
:sup DataStructure
:name "Non Std Data Structure"
:rem "A Non Standard Data Structure is one which has not been fully and rigorously
documented and has not been designed to be used by multiple instances. Non Standard
Data Structures may also be referred to as Customised, Bespoke, Ad-hoc or unstructured ."
```

Level 2 Modules
Logic2

```
;;; Start Date: 20th Jan 2012
;;; Author: N Hastilow
;;; Other Contributors: Tish Chungoora

:Use MI


;;;=========================================================
;;;Competency Question 1a - Can it identify which systems wish to / need to interoperate?
```

```
;;;=============================================================

(=> (and (MI.System ?s1)
                                        (MI.System ?s2)
                                        (MI.Output ?o)
                                        (MI.Input ?i)
                                        (MI.hasOutput ?s1 ?o)
                                        (MI.hasInput ?s2 ?i)
                                        (= ?o ?i))
                    (MI.interopsWith ?s1 ?s2))
:rem "If the output of one system is the input of another system the systems interoperate."

(=> (and( MI.hasInterface ?a ?b)
                                    (MI.hasInterface ?c ?d)
                                    (= ?b ?d))
          (MI.interopsWith ?a ?c))
:rem "If two things have the same interface they interoperate."

(=> ( MI.analyses ?x ?y)
                    (MI.interopsWith ?x ?y))
:rem "If something analyses something else they interoperate."

(=> ( MI.informs ?x ?y)
                    (MI.interopsWith ?x ?y))
:rem "If something informs something else they interoperate."

(=> ( MI.mayAccess ?x ?y)
                    (MI.interopsWith ?x ?y))


:rem "If something may access something else they interoperate."
```