Component-based Records: a Novel Method to Record Transaction Design Work

L. Ding, M. Giess, YM Goh, C.A. McMahon, U. Thangarajah

Innovative Design and Manufacturing Research Centre, Department of Mechanical Engineering, University of Bath

Abstract: The growing pressures from global competitive markets signal the inevitable challenge for companies to rapidly design and develop new successful products. To continually improve design quality and efficiency, companies must consider how to speed design processes, minimise human-errors, avoid unnecessary iterations and sustain knowledge embedded in the design process. All of these issues strongly concern one topic: how to make and exploit records of design activities. Using process modelling ideas, this paper introduces a new method called component-based records, in place of traditional design reports. The proposed method records transaction elements of the actual design processes undertaken in a design episode, which aims to continually improve design quality and efficiency, reduce designers' workload for routine tasks, and sustain competitiveness of companies.

Keywords: process modelling, transaction design, XML, Topic Maps

1. Introduction

To survive today's fierce competitive market; engineering companies must continually design and develop successful new products that have higher quality with lower cost and shorter product introduction lead times. Effective and efficient design processes are crucial in determining the capabilities, costs and other attributes of products. Such processes depend on

the knowledge and creativity of designers and the efficiency with which resources for designing are used. With the change towards whole product lifecycle support and the increase in the knowledge-intensivity and complexity of modern-day design tasks, recording of the information, knowledge and experiences accumulated in designs is becoming particularly important today, not only for design of new products but also for product lifecycle support. Thus, major challenges for companies include: how to implement an appropriate design process to improve the performance of its products; how to make effective records of the work that is carried out in design activities; how to standardise and automate repetitive work to minimise error and rework in the design process; and how to capture the knowledge embedded in the design process to ensure the sustained competitiveness of a company. To respond to these challenges, various models and techniques for description or planning of design processes (i.e. design process model) have been proposed. Broadly, a process model can be descriptive, prescriptive, or have aspects of both [1]. A descriptive process model attempts to capture tacit knowledge about how work is really done (e.g. IDEFØ [2]). A prescriptive process model tells people what work to do and perhaps also how to do it (e.g. Signposting [3]).

Process modelling has achieved considerable success in improving the management of design processes, such as in lead time reduction, task scheduling and project decomposition [4]. However, there are still a number of limitations need to be overcome [1, 5], many of which are compounded by limitations in the way that actual design processes are recorded, such as lack of completeness of actual process descriptions, weakly structured and raw records, and poor capture of rationale.

Notwithstanding the difficulties in representing process steps, there is considerable value in better representation of design processes. Firstly, individuals and organizations tend to follow similar approaches in their work and learn and adapt through successive execution of processes [6]. Lessons from previous designs also benefit individuals and organizations by avoiding similar failures. Secondly, novice designers especially will benefit from a more complete record of such occurrences. Design processes, including design activities, decisions-made, and corresponding rationale, are currently largely still recorded in text documents (e.g. design reports, meeting minutes) and in some cases may be retained in employees' memories. It is difficult for novice designers to assimilate and digest processes recorded in text documents, and the employees who carried out the work may not be available. Furthermore, an analysis of information requests from novice designers found that they were aware of their knowledge needs in only 35% of their queries [7]. A useful process model will help designers, especially novice designers, pick up the correct information resources and methods at an early stage and minimise mistakes, false assumptions or incomplete information. Thirdly, better capture of processes will assist especially embodiment design for mature products, e.g. in automotive and aerospace engineering, in which a great deal of work is transactional, involving repetitive information access and manipulation steps. Fourthly, recording design activities in a better structured form will strengthen data traceability and information retrieval. It especially benefits product lifecycle support, for example tracing design rationale from service feedback.

Using process modelling ideas, this paper introduces a new method to record transaction elements of the actual design processes undertaken in a design episode. The method, called component-based recording, is used in place of traditional design reports. The proposed

method aims to 1) combine documentation and computer interpretable data to record the actual design work that has been done - recording information flow and dependencies, relationships between activities, successful and unsuccessful practices, and so on so that designers and engineers at later stages of the product lifecycle can look back to learn the lessons and continually improve design process; 2) allow routine work to be standardised and where appropriate reused, thereby freeing designers to focus their creativity and innovation on value-adding activities; 3) simplify definition of process model to make the recording of work quicker and easier; 4) allow both bottom-up and top-down recording of the process undertaken by an engineering team as it is carried out, and then browsing and retrieving of the record of the model from different viewpoints according to various users and purposes.

The following parts of this paper are organised as follows. Section 2 gives the background of this research, including relevant literature from process modelling; and a brief investigation of design records and design work. Section 3 presents the method of documentation of design records using a component-based model, including the basic framework, the definition of an activity, XML schemas and a Topic Map approach for organising activity records. Section 4 describes the implementation of the proposed approach with a case study. Finally, Section 5 gives the conclusions and further research discussions.

2. Background

The following section presents a critical overview of process modelling, and the status of design work and design records.

2.1 An overview of approaches to process model

Compared to many other project-like activities, design processes may be characterised by involvement of large number of tasks, complicated interactions among tasks and people, and unavoidable inclusion of iterations and rework. These characteristics make design processes challenging to model and a number of process models and techniques have been proposed in recent decades for representation, scheduling and capture of design processes.

A process is often modelled as an activity net. The early activity net-based techniques for project planning, task scheduling and control, including the Critical Path Method (CPM) [8] and the Project Evaluation and Review Technique (PERT) [9], form the foundation for many project management models. Generalized precedence relations (GPRs) [8] were then proposed to extend CPM from "strict precedence" (i.e. activity finish-to-start relationship) to four possible relationships (i.e., start-to start, finish-to-finish, start-to-finish, and finish-to-start).

CPM/PERT is often used to describe sequential tasks, while the DSM method, developed by Steward [10], is a scheduling technique that has been extensively used to support concurrent processes. The DSM uses a square matrix to represent a process by showing information flow between activities [10-13]. Typically, a cell on the diagonal of the square matrix represents each activity; the left of the matrix gives activity names; and a mark in an off-diagonal cell indicates an activity interface [7]. The DSM provides a simple way to visualise the structure of an activity network and to compare alternative process architectures [14]. Research has been carried out based on the original DSM method to manage issues like iterative groups and task overlapping. For example, two sequencing models [15-16] aim to reduce the number of information feedback loops, information crossovers and the length of iterative cycles [4]; an extended framework [17] uses a graph theoretic approach for transformation and analysis of a

network of design activities; a sequential iteration model [18] suggests an initial ordering of the coupled design tasks to minimize their expected duration; an extended sequential iteration model [19] allows for random duration of tasks as well as allowing multiple tasks to be attempted simultaneously; the work on transformation matrix method (WTM) [20] models design iteration by replacing the off-diagonal DSM elements with the strength of dependence between tasks, given rise to transfer of work, or rework involved in the iterations; an analytical model has been proposed which combines the decisions of overlap and communication in the presence of uncertainty and dependence between tasks, with the goal of minimizing time-to-market [21]; and a second-generation simulation model [22] accounts for many important characteristics of engineering design process, such as information transfer patterns, uncertain task durations, resource conflicts, overlapping and sequential iterations, and task concurrency.

Besides the work on DSM, research work has been carried out to strengthen the guidance and scheduling of design process. The major efforts are: a Q-GERT model [23], which allows for queuing delays by considering probabilistic routing of tasks to servers, and probabilistic iteration; a triangularization algorithm [24] for organizing design activities such that the number of cycles is minimized; a product development strategy combining parallel and serial processing [25] aiming to determine how much parallelism is desirable, and whether minimizing development time justifies an increase in development cost; a model-based framework [26] based on the (evolution and sensitivity) properties of the information exchanged between overlapping consecutive stages of a development process; a multiple-phase project model [27-28], which explicitly models process, resources, scope and targets so as to improve project performance and understand the dynamic concurrence relationships that constrain the sequencing of tasks as well as the effects of and interactions with resources,

project scope and targets; a signposting model [3, 29], which associates confidence levels to the parameters in a task and uses these to prioritise or “signpost” the next appropriate task; a rich model of the product development process architecture [6], where each activity has an uncertain duration and cost, an improvement curve, and risks of rework based on changes in its inputs; and a generalized homogeneous and non-homogeneous state-space concept proposed to model concurrent, coupled and design tasks and to analyze and control the stability and convergence rate of the design tasks [30].

Many researchers have studied how to represent design process so as to aid understanding and capture of the design process and knowledge. The decomposition of large design projects into smaller elements is seen in the work of Alexander [31] and Kusiak and Park [32]. The Structured Analysis and Design Technique (SADT) is process model and representation method that has been widely used, particularly its well-known derivative, IDEFØ [2], one of the ICAM Definition Language family of modelling techniques. An IDEFØ model is composed of a set of hierarchically linked diagrams, which provide a static descriptive view of a process. Maimon and Braha [33] developed a method of modelling design processes based on the Analysis-Synthesis-Evaluation (ASE) paradigm. The proposed design process model is denoted as tuples containing artifact space, a set of explicit constraints, analyzer, synthesizer and evaluator. Zeng and Gu [34] proposed a design process model that embodies synthesis and evaluation process, design problem redefinition process, and design decomposition process. The design process can be viewed as sets of decisions [35], so that the process can be modelled by capturing decision points, outcomes, conflicting requirements etc. One of the earlier recording systems applied during decision-making processes is called Issues Based Information System (IBIS) [36]. IBIS uses directed graphs, where nodes representing issues

are linked to solution nodes, and the solution nodes can then be linked to nodes representing arguments for or against them. Tools building on an IBIS-type approach to design rationale capture include Compendium [37] and the Design Rationale Editor (DRed) [38]. Fathianathan and Panchal [39] presented an approach for modelling design process using design nodes, each of which embodies the design problem and design task constructs in an integrated manner. The design nodes are defined and instantiated during an ongoing design process, and thus facilitate dynamic decision making on how the design process should progress. Gorti et al [40] developed a knowledge representation model, which incorporates both an evolving artefact and its associated design process. In the proposed model, a set of objects are used to capture design process, which have five basic components (i.e. unique identifier, non-unique identifier, type-attribute-value triplets, a set of methods, a set of relationships, and a set of constraints) and five entities (i.e. goal, plan specification, decision, and context). The Design Roadmap [41] represents bipartite relationships between task and feature nodes allowing for process and information flows to be modelled. Barrientos et al [42] illustrated a design episode using flow diagrams to model design evolution, representing the state of a design over time and information flow.

In respect of making records of design activities, Qureshi [43] proposed an Integration Core Model attempting to provide a framework for archiving design process information at the actual design time. The work highlighted that the temporal aspect of the design process can be useful for revealing the design history indicating the evolutionary of the design. Models such as Petri net also inherently allow the temporal aspect to be modelled. The Petri net comprises two types of node: place and transition. The ‘transitions’ represent events and ‘places’ represent conditions associated with the incident events. Tokens, which are assigned to places,

represent units of information or resources that flow through the net. The execution of events through tokens firing will indicate information flow and the sequential relationship between the nodes. The Petri net is useful for modelling information processing systems, such as Gonikhin and Medland's [44] presentation of an integrated constraint modelling and Petri net technique which gave a design modelling approach which could be employed during both the early stages of a design and the down stream manufacturing operations. By fixing 'goals' in certain constraint transitions (to stop them being fired) the downstream activities can be pursued without destroying previously set design relationships. Other examples using Petri net include [45-47].

Recent industrial efforts in Business Process Reengineering (BPR), Work Flow Management (WFM) and manufacturing process planning/scheduling focus on process specifications to support process interoperability and information exchange in the product lifecycles. For instance, the Process Interchange Format (PIF) is developed to support the exchange of process descriptions among different process representations (including process and workflow modelling). The Business Process Modelling Language (BPML) is developed as a meta-language for the modelling of business processes for expressing business processes and supporting entities [48]. The Process Specification Language (PSL) initiated at NIST has strong links to PIF and the WFM but focuses on defining a neutral representation (a language) for sharing of process data among all aspects of a manufacturing process [49].

2.2 Limitations of current process modelling when applied to design activities

Section 2.1 has shown that process models have been successful in improving management of design processes, such as planning and scheduling design tasks, identifying precedence relationships between tasks, capturing and representation of high-level information and

enhancement of design processes. However, to enhance application of process models, a number of limitations need to be overcome, including:

1. Current process models represent processes at a coarse granularity level. Most modelling for task scheduling and description, such as for WFM, are based on the high-level activities associated with a business process. Furthermore, many methods developed to date are limited to the number of the activities. For example, DSMs up to 500 elements are often "rolled up" or "dithered" and represented smaller matrices [14]; and function boxes at each level of IDEFØ diagrams are limited from three to six. Due to lack of completeness of process descriptions, process models generally do not allow information dependencies to be described with sufficient operational detail and accuracy. Such approaches are useful for management purposes, e.g. scheduling work and archiving, but are often of limited value in guiding day-to-day work by engineers.

2. The time required to build models is considerable, even for relatively coarse granularity models [1]. It is especially time-consuming to identify the possible variations in a complex process, and thus hinders practical applications of process models.

3. Often there are considerable differences between actual and documented processes. In theory, design reports should tell the "real" work, including failures and successes. However, there has often been pressure for designers not to mention failures though companies have realised the advantages for future projects of recording difficulties as well as successes. In effect, many companies have forced employees to work in a constant state of cognitive dissonance, where they must pretend to follow a documented process, while doing "what really needs to be done" [1]. Many process models that have been proposed require strong assumptions and availability of data [4]. For example, the sequential

iteration model [18] assumes that only one task is performed at a time, there is no delay between the task executions, and each execution of a task is fixed and deterministic. Obviously, such assumptions are not true in a practical environment. In addition, process models are invariably subjective according to the interpretation of the builder and the 'real work' people [5].

4. Current efforts at modelling design processes are mainly suitable for improving existing processes, and provide little guidance to designers for defining an ongoing design process [39]. Efforts on issues that can improve performance of design processes significantly are not sufficient, such as how to reuse methods that have been proved in previous designs to minimise mistakes, and how to retrieve design knowledge to avoid missing pertinent information and solutions.

2.3 Design process and records

Frankenberger and Badke-Schaub [50] proposed that a design process can be viewed abstractly by identification of phases of routine work on the one hand and 'critical situations' on the other. The 'critical situation' elements of the design process (e.g. design plan or review meetings) are when decisions are made for the next steps of the process, such as a new direction on a conceptual or embodiment design level. Usually, it involves consideration of information which has been generated from different individual designers from different tasks at different times, and (usually synchronous) communication between various members of the design team. In contrast, routine work normally refers to activities that are able to be undertaken by an individual designer or single working team. According to the different objectives of these work activities, i.e. whether they aim to increase a level of understanding, or whether they involve manipulating information to achieve a tangible outcome, routine work

activities can be further divided into two groups: learning and transaction. The ‘critical situation’, learning and transaction activities have different characteristics and objectives; for example, ‘critical situations are often carried out by discussion, such as in meetings, telephone calls and even in chat-room exchanges, and many important decisions and the associated rationale are considered in the discussions; transaction activities are often limited to a single person with some regular operations (e.g. data processing, calculation, analysis and simulation) and usually involve repetitive information resources and manipulations; learning activities address the identification of knowledge that does not exist for the participants before the activity. The nature of a learning activity varies according to the participants. It is our view that it is difficult, based on current techniques, to describe all of these kinds of activity clearly and effectively using current process modelling paradigms, and that different activity models and records are needed for each type of activity.

The Work Package 1 (Advanced Product Information Representation & Management) of the Knowledge and Information Management Through Life (KIM) project, with which the authors are involved, aims to capture the whole design process more completely and document it with better structured form. This paper focuses on making records of transactional design work, while separate work in dealing with documentation of critical situations type work, such as DRed [38] and Multimedia Minutes [51], will be reported separately.

Various types of information are generated during the whole design process, from early information of conceptual design (e.g. design specifications, product functions and sketches) to the more elaborated information of embodiment design (e.g. product models and calculations), and the final detailed solution (e.g. dimensions and tolerances). With the widespread utilisation of computerised information, design information has been stored in a

large number of information formats, such as word processed documents, CAD models, databases, and even in audio/video files. Simoff and Maher [52] classifies computerized design information into four groups: structure-valued data (e.g. object-oriented data structures, relational tables, CAD models, and attribute-value pairs), weakly structured data (e.g. free-text documents, CAD drawings, and calculation documents), raw data (e.g. sketches, raster images of photographs, audio and video data), and links data (e.g. hyperlinks within and between structured-valued data and weakly structured data, and information about the sequence of visited links). Generally, structured-value data has been well recorded and some of them (e.g. conventional databases) already have well-developed retrieval mechanisms. Unfortunately, most design records are still captured in weakly structured form, raw data or a mixture of these, such as design reports and meeting minutes, and therefore lack good mechanisms for information retrieval. Moreover, documentation of the process followed in design is generally carried out in retrospect, in written reports and other documents. The record is as a consequence often a partial record, and details of the process are often mixed up with information about the rationale employed in design decisions and about the information used in the process. The creation of retrospective records is often carried out reluctantly by those involved, with little thought about who the recipient of the information might be. And there is particular reluctance, often encouraged by organisations, to record process steps that have led to error or failure, even though knowledge of such outcomes would be particularly valuable in design. To address these problems, this paper will present a new method to capture transactional design processes in a more structured and computer-interpretable form so as to further support retrieval and reuse of design records at later stages or in new designs.

3. Component-based design records

Generally, an IDEFØ model is composed of a hierarchical series of diagrams that gradually display increasing levels of detail describing functions and their interfaces within the context of a system [2]. The hierarchical structure of IDEFØ allows it to easily describe the activities of a system into a desired level of detail, but it limits its applications to top-down representation and interpretation. The number of function boxes in a diagram at each level of decomposition is limited to from three to six, which make it difficult to model some complicated situations. In addition, the IDEFØ model is developed for a particular viewpoint, i.e. data flow, which does not allow it to satisfy the requirements of all users at different stages throughout the product life cycle. However, on the other hand, a transaction activity may be expressed as an information manipulation, including information inputs and outputs, methods of manipulation, and resources that have been used. That is essentially the definition of an IDEFØ function node. Thus, the solution we propose is to make a record of design activities individually at the time they are carried out, and then to process, assemble and re-assemble the records of these activities based on different logical relationships at a later stage.

3.1 Framework of component-based process model

As shown in Figure 1, the architecture of the component-based process model is as follows:

1) A design process is documented by a series of typed activities and recorded by a collection of such activities. All processes, whether simple or complex, are implemented by a number of sequencing or parallel activities carried out by same or different members. To support this documentation, a client-server structure has been developed to enable multiple clients to capture activities from different geographic locations at the same time.

2) Each activity is recorded as an individual Extensible Markup Language (XML)-based document. Rather than reporting on design activities after they have taken place, designers simply record at the time of carrying out an activity what the inputs and the outputs are, a brief note on the methods applied and comments such as assumptions made, rationale and so on. There is no need at this stage to organise complex relationships among activities. XML has been chosen for a number of reasons: the representation is computer interpretable and also human readable, it is application-independent, it allows a mixture of data and text, and it is extensible and tailorable. XML will be discussed further in the later section.

3) The proposed model separates how data/information is stored and how it is used. Data/information consistency is always a big issue for all information management methodology. On one hand, to reduce the risk of inconsistencies occurring, all data/information should be stored only once. On the other hand, a single data/information element may be used in different applications several times. To solve these contradictory requirements, data storage must be separate from data applications. Cross-referencing is initially applied within a document to refer to information elsewhere. Here, explicit cross-referencing is adopted in the proposed model, where the data/information generated/used during a design process is only recorded by a clear cross-referencing rather than incorporating the data/information itself. Cross-referencing cannot only guarantee the consistency of the data, but also benefit information organisation or data mining at later stages.

4) The model is designed to allow information to be created in reusable chunks. Traditional design records are design reports, meeting minutes and so on, where the design process is described in text documents. As shown in Figure 2, design reports mix the process, rationale, decisions and the information resources. Furthermore, the rationale behind the design is often

poorly recorded in these traditional design records – meeting minutes, for example, often say little about the rationale behind decisions [53]. The records are thus very difficult to reuse or retrieve for future product upgrade or new design projects. In contrast, the component-based model is able to support design process reuse through the generation and application of a process template library. Mature processes or processes that have been shown to be correct can be standardised as process templates and stored into the process template library for reuse in future design activities. When a design activity applies a standard process (i.e. a process template stored in the library), the process model will record its explicitly referencing of the process template instead of complex description of the method. The combination of process templates and cross-referencing assist exploration of the activities applying a similar process, so as to benefit correlation and data mining.

3.2 Activity definition

Activities are the constituent elements of the proposed process model. Here, an activity refers to a package of work to be done to produce results. To encourage designers to capture complete activities in real-time, the description of an activity must be rich but simple.

In traditional design reports, the same activity could be recorded differently by different people. An example of a simple transaction activity is motor selection. Figure 3 (a)-(c) shows a record of the motor selection activity in three different reports, showing that more or less the same contents are displayed in rather different ways (all of which would be difficult for a computer to interpret). However, looking back on what has actually been done in the selection, the information in the activity that really needs to be recorded is clear:

- the inputs and where they come from (e.g. motor power is 15 KW, which is an output of the activity of “motor power calculation”; and approximate motor speed is 1500 r/min, which comes from the activity of “motor speed calculation”);
- the outputs and where they are stored (e.g. the motor model is chosen as 4PE160L and stored in a file named as Choose_motor_source.xml);
- the methods applied in the activity and where the corresponding process template (including descriptions and their executable applications) is stored. For example, the rules of motor selection are: the output power should enable the application (in this case a pump) to work correctly – i.e. motor power must be larger than the input motor power value - and the speed must be close to the required speed. The process template is stored in Choose_motor_method.xml.
- The information resources that support this activity and where they come from (i.e. the motor is selected from the catalogue of sg-motors.phf).

Furthermore, extra information related to management objectives needs to be stored, such as who carried out the work, when it was done and how long the activity took. Therefore, as illustrated in the XML schema in Figure 4, a transaction activity can be explicitly defined with six attributes and seven elements, each of which can be further described by one or more elements and attributes:

- Attributes: An activity can have six attributes, including: 1) activity “ID” is the ID of activity that is set for computer identification. Thus, it is unique and automatically generated by the system; 2) activity “title” is the name of activity. It is normally defined by designers, but could be selected from a library or list; 3) “Objective” refers to the

targets of the activity. The “objective” is important for the assembly of activity descriptions (e.g. hierarchical decomposition, and generation of standardised activity) at a later stage. 4) “type” is indicated as “abstract” or “detailed” according to whether the activity needs to be divided into further detailed activities or not; 5) “desc” and “commentary” are defined for designers to insert simple description or extra comments (e.g. to describe assumptions) by free-text.

- Status: Status defines a state of the activity at the current time, including “not started”, “processing” and “completed”. It helps project management, especially for scheduling.
- Period: Period indicates the time of an activity starts and finish; and consists of the two attributes “startTime” and “endTime”.
- Actors: Actors are the people who really carry out the activity. Each actor gives an explicit cross-reference linked to the record of the specific person , which is usually stored in human resource or department database. The database can further provide various attributes for the actor, such as “name”, “position”, “department”, “location” and “contact”.
- Input: Input refers to the data or objects that are transformed by the activity into output. It consists of one or more “input_element”s, each of which represents a particular data or object. The “input_element” is further defined by three attributes: 1) “source” provides a unique URI of a certain information source that stores the data or object; 2) “title” refers to a certain mark-up identifying the particular data or object in the information source. Through combination of “source” and “title”, a particular data or object is embedded in the process model by an explicit cross-reference; 3)“desc” provides a free-text description of the input element.

- Output: Output refers to the data or objects that are produced by the activity. Similar to input, output consists of one or more “output_element”s, each of which represents a particular data or object. Three attributes with the same definitions of the attributes in “input_element” are included in the “output_element”: “source”, “title” and “desc”. An explicit cross-reference can be defined by the “source” and the “title”.
- Method: Method expresses the means or operations that are used to transform inputs to outputs. One or more “method_element”s can be included. As discussed before, a process template is introduced to record the method. Using the method template, the “method_element” will only record as a unique URI where the particular method template is stored, not the detailed description of what the method is and how the method works. It not only standardises the description of a method and saves the designers’ time from repetitive and non-creative work, but also supports users in identifying the relationships among activities that adopt the same method so as to rapidly recognise best practices or to assist in identifying practices that regularly lead to failure. The detailed definition of the method template will be given in the next section.
- Resource: Resources indicate the materials (e.g. online sources, catalogues, and guidance’s) that are used to support the activity. Correspondingly, they can consist of one or more “resource_element”s, each of which represents a particular material based on three attributes: “title”, “source”, “desc” and “type”.

Based on the above definition, Figure 5 shows the XML document for the example activity – motor selection.

3.3 XML schema and XSL/XSLT transformation

XML is a descendant of SGML (the Standard Generalised Markup Language, ISO 8879) and became an ISO standard in 1998. Various XML-based techniques have been proposed for different applications covering resource description, ontologies, interoperability standards and so on, such as the Resource Description Framework (RDF) using XML as an interchange syntax [54], the Web Ontology Language (OWL) written in XML [55]; Simple Object Access Protocol (SOAP) based on XML for exchanging information over HTTP [55]; and Commerce XML (cXML) for consistent communication of business documents by defining a set of XML DTDs [57].

Using XML to record design activities offers many benefits. Firstly, XML is both computer interpretable and human readable. XML tag names are normally transferred from the meaning of the data and therefore they are readable. Each XML tag immediately precedes the associated data, so as to make the data structure easily understandable by both humans and computers. Further such a data structure also makes it easy to manipulate and exchange the data. Figure 5 shows the XML schema describing the structure of the XML document of a design activity. Comparing the XML schema with the definition of activity discussed above, XML is obviously human understandable, even by novices.

Secondly, XML allows users to define their own tags (i.e., the labels that are embedded within text to distinguish individual/groups elements for display or identification purposes) based on the specific needs of a document, and therefore it is extensible. Figure 6 gives another XML schema representing the structure of an XML document of a method template according to the requirement to describe a typical design method. The XML schema consists of ten elements, including: objective, scope, conditions, inputs, outputs, description, settingtime, modifiedtime,

expiretime, contact and links. The “objective”, “scope” and “conditions” assist designers to search and choose the most suitable method for their design task, where the “objective” gives what kind of function or target the method can support; the “scope” shows the scope of the method that can be applied; and the “conditions” represents what pre-conditions are needed for application of this method. The “inputs”, “outputs”, “description” and “links” support designers to instantiate the method template and automatically execute the instance of this method. The “description” gives the detailed explains about how the method is carried out and the rationale behind of it; and the “links” provides a URI link of an interface called “DLink” that is able to automatically execute the method, or a series of URIs called “MLink”, each of which representing a sub-method-template being comprised in the method template. For instance, a method template for choosing a motor consists of four sub-method templates: computing motor power, computing motor speed, selecting motor and validating selection. The “settingtime”, “modifiedtime”, “expiretime”, and “contact” are designed to guarantee the method template is valid and correct.

Thirdly, XML allows to mixing different types of information, including data and text, so making it suitable to capture the design activity. Furthermore, XML allows the explicit identification of units of data so that corresponding metadata, description, manipulation and exchanging can be associated with the particulate unit of data.

Finally, one of most important powers of XML lies in the separation of the information and its presentation. The information is stored in an XML file once, and then the content of the XML file can be transformed for different viewers or devices based on style sheet processing using an XSL (Extensible Stylesheet Language)/XSLT (XSL Transformation) processor [58]. Thus, an XML-based process model has the capability 1) to freely use existing information in the

activities in new combinations, contexts and processes, such as tailoring the information for different users, combining the information for various purposes and applications, 2) to return useful results by searching and re-organisation, like exploring information dependencies, overview of whole design process by particular structure (i.e. decomposition); 3) to establish design reports in different forms according to different options and security levels; and 4) to interact references or resources using XLink [59], XPointer [60] and XInclude [61]. It is the biggest benefit from the introduction of XML/XSLT to record design.

As an example, the XML document shown in Figure 5 gives the activity of motor selection. With its supporting information sources, e.g. calculate_motor_power_source.xml, computer_motor_speed_source.xml, choose_motor_method.xml and choose_motor_source.xml, it can be transformed to a detailed report, which is shown in Figure 7(a). In contrast, if designers do not want to broadcast the rationale behind this activity, a brief report can be generated by tailoring the sensitive information. As shown in Figure 7 (b), the information, such as the method applied and the links to where the inputs and the outputs are stored, are masked. Combining with XML documents recording the previous activities of calculation of motor power and calculation of motor speed, a broad report is generated as shown in Figure 7 (c).

3.4 Browsing a series of activities through Topic Maps

By allowing an engineer to browse through a series of interconnected activity records, with the activities associated by the information interdependencies between them, it becomes possible to traverse a series of activities according to a particular association of interest. These viewpoints may be dependent upon the role the engineer plays within the design episode, for example a certification engineer may wish to identify the methods deployed within the episode

whereas a purchasing engineer may wish to see which components have been selected. These potential viewpoints cannot readily be identified a priori and hence facility must be made to allow the engineer to dynamically update how the interdependencies are displayed to meet their particular viewpoint.

3.4.1 Topic Maps

Topic Maps are a means of expressing how different concepts are related and where these topics may be located within an underlying document corpus or information set. The notions of topic, occurrence and association are known as topic characteristics [62].

A topic is, in essence, a mechanism for defining a given subject, where that subject may be a tangible artefact or an abstract concept. Not all subjects may be directly computationally addressable [63], hence the topic is a reification of the subject (reification is the act of treating an abstract entity as tangible) which provides a proxy or surrogate through which to manipulate or otherwise interact with the subject. Different types of topic may be defined in order to differentiate between distinct classes of topic, both to provide greater clarity and to support certain treatments, such as querying, conducted through the Topic Map Query Language (TMQL) [64]. Types are also used for the associations, where pre-defined forms of relationship between specific types of topic are defined as association types, within which each topic plays a given pre-defined *role*. This provides a degree of rigour in construction and assists in further processing, although it is possible to include non-defined or typed associations and topics to deal with emergent concepts. The third characteristic, occurrence, is an instantiation of the given topic or subject within an information set, for example an entry on a web page, a document or a specific element of a document.

A number of languages for Topic Maps have been developed, the earliest major standard was ISO13250, which is expressed as an application of SGML and uses the HyTM syntax [65]. A separate consortium sought to facilitate the application of Topic Maps on the Internet, for which they identified XML as a key enabling technology, which led to the development of the XTM syntax as expressed as an XML grammar [66]. A convergence of the two standards has resulted in the development of ISO13250 as an XML syntax, XTM 2.0 [67].

3.4.2 Construction of a Unified Topic Map

As a Topic Map is an abstract layer constructed independently of the information set which it describes, it is possible to manipulate or modify a map to reflect the evolution of the information set (predominantly as new activities are completed and the corresponding records incorporated into the overall map). As Topic Maps were originally intended to support the combining of indexes from separate electronic documents [62] the ability to *merge* separate maps into a complete, consistent single map is intrinsic to Topic Maps and is referred to as a 'central operation' in the XTM 2.0 data model [68].

It is perhaps more useful (and true to the standard) to consider merging as being the act of combining topics. It is by identifying identical topics and blending these into a single topic, with the combined associations of the two individuals, that merging proceeds. Topics are considered identical when they refer to the same subject, which the Topic Map community recognised may be subjective in certain cases which gave rise to the development of Published Subject Identifiers (PSIs) [e.g. 69-70]. These PSIs are in essence resources of known provenance and assured longevity which are intended to unambiguously define a subject. In this research the role of PSIs may be taken by the information entities used within the design activity, as referenced by the process model. If these may be unambiguously identified and

referenced in a consistent manner, Topic Maps automatically support the combination of multiple occurrences of a given information resource within different activities.

3.4.3 Browsing a Topic Map

It is possible to traverse the associations between different topics in order to comprehend a particular trail of activity, perhaps by following the flow of information between activates or locating where a specific resource has been utilised. Such a perspective may be improved by allowing a topic to be viewed as part of the broader interconnected map, extending beyond its immediate neighbours. Although not intrinsically a means of visualisation, Topic Maps are well-suited to visually displaying a set of relationships across information resources such that engineers may comprehend the evolution of information and the nature of the activities which such information connects within the context of the broader design episode. It will be discussed further in the following section.

4. Implementation and Case study

The implementation of the proposed component-based process model has been developed using a Java program, the Document Object Model (DOM) and the SOAP. The XML documents recording the process are stored in a server. When a client logs in, the server firstly registers the available services for the particular client. Then, XML documents are loaded into the DOM, where elements, child elements and attributes are considered hierarchical child nodes of the root document element. Thus, as shown in Figure 9 (b), the activities in the process model are displayed in a tree-format through a user interface developed in Java. All operations and modifications of the process model made by clients are updated in real-time through DOM instead of the server. In addition, clients send their requests to the server and

receive server' commands using a SOAP technique. The XML documents stored in the server are updated when users formally save them. To satisfy different requirements and users, the XML documents in the server can be tailored, combined and transformed to different sets of information in different display formats through parse programming by JavaScript and XSL/XSLT, producing information such as information dependencies, decomposition overview, and various design reports. In addition, the XML documents may be converted to XTM further treatment as a topic map so as to further identify information dependencies and visualise activity relationships. With Java bindings specified in the published recommendations for DOM [71], details of each node in the DOM may be readily referenced within the JavaScript and used to construct XTM representations of each activity. The developed JavaScript may be accessed.

An undergraduate engineering design project – a layshaft sub-assembly design – has been chosen to demonstrate the capability of the proposed process model to record and manipulate embodiment design activities. The assignment is to design a motor driven layshaft to drive a pump. Figure 10 gives the relative positions of the pump, motor and layshaft as well as possible positions for the bearings. Table 1 provides the summary of the specification of the design project. The project covers most basic types of activities that may occur in embodiment and detail design, including determining a basic configuration for the primary design; selection of appropriate 'off-the-shelf' (i.e. catalogue) components to satisfy functional requirements (such as, electric motor, pulleys, keyways, bearings, etc); establishing dependent parameters (such as shaft forces, etc.); and completing the detail design of the shaft (e.g. choosing an appropriate material, determining suitable tolerances, etc) to satisfy strength and constraints.

The whole design took about 5 hours by a team that understood the required process well and was captured by a video. Based on the proposed process model described above, designers recorded each activity by simply entering information concerning inputs, outputs, methods, resources, various attributes, etc. through the interface shown in Figure 9(b). Nearly 50 activities have been captured and recorded in separate XML documents, as shown in Table 2. According to different purposes, the activities recorded can be assembled into different output records and formats by the XSL/XSLT transformation system. Figure 11 gives some examples of various reports generated from the 50 XML documents for different purposes: Figure 11 (a) gives an example for showing the activities are displayed as a decomposition diagram by matching the objective decomposition diagram providing by users. The example demonstrates recorded activities can be assembled based on different structures for better understanding process behaviours; Figure 11 (b) and (c) provide a table of the activities that have information dependencies and a table of the activities that share common information resources, respectively. The examples show that various relationships between activities can be recognised after generation by a well-formatted process model; Figure 11 (d) presents a text report of an activity and its corresponding support activities. This example shows that as the design process has been clearly recorded in the process model, various reports for different uses can be automatically generated. Thus, designers no longer need to spend long periods of time writing lengthy reports and design efficiency is improved as well.

Figure 12 shows a sample topic map as a visual map of the activities and their information dependencies. This is a small section of a larger map, and may be expanded to include a greater number of activities if required. This map is presented in an IDEFØ-like format, with inputs and outputs to the left and right of an activity respectively, and the resources underneath

the activities. This view may be rotated in any desired manner to provide a dynamic view of the underlying design episode. Each node in the map corresponds to a topic page, which may be accessed from the visual environment, and which provides information of all associations for that specific topic and also a URI to the underlying information resource which the topic represents.

5. Discussion and Conclusions

The growing pressures from global competitive markets signal the inevitable challenge for companies to rapidly design and develop new successful products. To continually improve design quality and efficiency, companies must consider how to speed design processes, minimise human-errors, avoid unnecessary iterations and sustain knowledge embedded in the design process. All of these issues strongly concern one topic: how to make and exploit records of design activities. Traditional design reports cannot satisfy such requirements because they mix design process, rationale and decision in text documents that are not readily computer-interpretable; lack explicit cross-referencing; poorly record rationale and discourage reuse. Although various process models have been proposed to date, their main purpose is for gaining insight into design dependencies, for decision-making and for design process management, not for making records of actual design activities. This paper presents a general component-based approach to making records of transactional design activities, based on design a process model idea that aims to continually improve design quality and efficiency, reduce designers' workload for routine tasks, and sustain competitiveness of companies. In summary, the proposed component-based records has the following advantages:

- It captures real design work: it captures the work that has actually been done, and therefore, the real information flow and dependencies, practices and indeed rework and

iterations are recorded clearly. It allows design information to be retraceable and helps companies in continually improving design.

- It shifts design records to design reuse: transaction activities are more about regular approaches and routine work. Thus, the process templates of standardised approaches in the proposed process model help designers to escape repetitive work so as to focus their creativity on value-adding activities. Furthermore, automation and standardisation of regular design approaches can further minimise design errors and speed the design process.

- It uses XML-based recording. XML is not only computer interpretable, but more important it can be assembled, tailored and transformed in different ways by XSL/XSLT, for example combining existing information to high-level reports and tailoring subsets of the existing information for different users.

- It supports to generate design reports automatically or semi-automatically, and therefore frees designers from report writing and improves design efficiency. The proposed approach records the explicit cross-references (i.e. unique URIs) of information generated/used within the design process. The information required for a design report can be automatically generated with the unique URIs using the programme of XML/XSLT (as mentioned at previous point). Furthermore, the unique URI allows to identification of information dependencies and activity relationship, and the capability of creating information in reusable chunks.

- Topic Maps provide a number of important benefits that make their application in this research compelling. Firstly, the Topic Map standard provides facilities to automatically merge different maps and condense identical nodes (those referring to the same

information resource) in two separate maps into a single node in the merged map. This is significant when considering that activity records may be created by disparate groups, which must then be amalgamated for purposes of browsing interlinked activities. Secondly, a Topic Map is intended to serve as an abstraction of the underpinning information resources, and hence the underlying information resources may be represented in whatever form necessary to facilitate their processing within an engineering environment.

The implementation of the component-based process model is based on a client-server structure and has been realised by Java programming language, DOM and SOAP. The XSL/XSLT transformation processor for generation of various viewpoints, reports and a browsing environment has been developed using Javascript and XSL/XSLT. And the system to convert the XML documents to XTM further treatment as a topic map has been given using JavaScript. The further work will focus on the following issues:

- Information capture tools: one of the motivations for the work in this paper is that in future dedicated information capture tools may be used to support the semi-automatic production of documentary records. Some efforts on automatic information capture are being carried out by both academics and the software industry. For example, the Simulation Process Studio (SPS) with UGS NX 3 Simulation [72] provides a palette of standard steps for users to drag and drop into the process with connecting lines defining the flow. The steps can then be saved as an XML file and made available to the standard Unigraphics NX application. Ciflex [73] and the more immersive work conducted within Virtual Reality environments [74] have met with some success in capturing detailed interactions and retrospectively inferring processes or information needs from this

captured detail. However, in this work the intent is to capture activities in a more 'light-weight' manner, capturing the specific information resources utilised and the manipulations applied within a computational environment.

- As the level of capture is at a highly detailed level, providing some broader depiction of the activities will be necessary to provide a coherent view of the captured design. A long-term intent of this work is to link the captured activities into a high-level process model which has been defined a priori, such that the high-level map not only guides the actual product development and design process but provides some 'sense-making' to the detailed records. Efforts are currently underway to conduct a product redesign using the IPPOP project's PPO core [75], which essentially links high-level activities to a central product representation, and allows key information generated in each activity (such as parameters) to be displayed in a collaborative environment. By capturing the detailed activity in the manner described in this work, it is possible to provide an XSLT report in this collaborative area.

- The capture of activity by considering information manipulation is most applicable to those forms of activity where an individual is working within an electronic environment and interacting with addressable resources. Further work has been conducted in looking at capturing different modes of design activity (e.g. learning activities and discursive activities) in representations. Currently, work has been carried out in looking at how synchronous and asynchronous modes of activity in the Computer-Supported Collaborative Working (CSCW) paradigm may be captured and co-ordinated, where the activity model is employed in the asynchronous modes of working [76-77].

Acknowledgements

The work reported in this paper has been supported by a number of grants for Engineering and Physical Sciences Research Council (EPSRC), involving a large number of industrial collaborators. In particular, current research is being undertaken as part of the EPSRC Innovative Manufacturing Research Centre at the University of Bath (reference GR/R67507/01). The authors gratefully express their thanks for the advice and support of all concerned.

References

[1] T.R. Browning, E. Fricke, and H. Negele, Key Concepts in Modeling Product Development Processes, Systems Engineering, 9(2), (2006), 104-128

[2] Integration definition for function modelling (IDEF0), Draft Federal Information Processing Standards Publication 183, 1993 December 21

[3] P.J. Clarkson and J.R. Hamilton, 'Signposting', A Parameter-driven Task-based Model of the Design Process, Research in Engineering Design, 12, (2000), 18-38,

[4] R.P. Smith and J.A. Morrow, Product development process modelling, Design Studies 20, (1999), 237-261

[5] J.S. Busby and G.M. Williams, The value and limitations of using process models to describe the manufacturing organization, international Journal of Product Research, 31(9), (1993), 2179-2194

[6] T.R. Browning and S.D. Eppinger, Modeling impacts of process architecture on cost and schedule risk on product development, IEEE Transactions on Engineering Management, 49 (4), (2002), 428-442

[7] S. Ahmed and K.M. Wallace, Understanding the knowledge needs of novice designers in the aerospace industry, Design Studies, 25, (2004), 155-173

[8] S. E. Elmaghraby, Activity Nets: A Guided Tour through Some Recent Developments, European Journal of Operational Research, 82, (1995), 383-408

[9] B.V. Dean, (Ed.), Project Management: Methods and Studies, North-holland, Amsterdam, (1985)

[10] D.V. Steward, The design structure system : a method for managing the design of complex systems, IEEE Transactions on Engineering Management, EM-28(3), (1981), 71-74

[11] J.L. Rogers, Knowledge-based tool for decomposing complex design problems, Journal of Computing in Civil Engneering, 4(4), (1990), 298-312

[12] S.D. Eppinger, Model-based approaches to managing concurrent engineering, Journal of Engineering Design, 2(4), (1991), 283-290

[13] S.D. Eppinger, D.E. Whitney, R.P Smith and D.A. Gebala, A model –based method for organizing tasks in product development, Research in Engineering Design, 6(1), (1994), 1-13

[14] T.R. Browning, Applying the design structure matrix to system decomposition and integration problems: a review and new directions, IEEE Transactions Lon Engineering Management, 48(3), (2001), 292-306

[15] C. McCulley, and C.L. Bloebaum, A genetic tool for optimal design sequencing in complex engineering systems, Structural Optimization, 12 (2-3), (1996), 186-201

[16] S.S. Altus, I.M. Kroo and P.J. Gage, A genetic algorithm for scheduling and decomposition of multidisciplinary design problems, Journal of Mechanical Design, 118(4), (1996), 486-489

[17] U. Belhe and A. Kusiak, Modeling relationships among design activities, Journal of Mechanical design, 118(4), (1996), 454-460

[18] R.P. Smith and S.D. Eppinger, A predictive model of sequential iteration in engineering design, Management Science, 43(8), (1997), 1104-1120

[19] S. D. Eppinger, M.V. Nukal and D.E. Whitney, Generalized models of design iteration using signal flow graphs, Research in Engineering Design, 9(2), (1997), 112-123

[20] R P Smith and S. D. Eppinger, 'Identifying controlling features of engineering design iteration' Management Science 43(3), (1997), 276–293

[21] C.H. Loch and C. Terwiesch, Communication and Uncertainty in concurrent Engineering, Management Science, 44(8), (1998), 1032-1048

[22] S.-H. Cho and S. D. Eppinger, (2001). Product Development Process Modeling Using Advanced Simulation. In: *Proceedings of 2001 ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference, DETC'01*, Pittsburgh, Pennsylvania DETC2001/DTM-21691.

[23] B.W. Taylor and L.J. Moore, R and D project planning with Q-GERT network modeling and simulation, Management Science, 26(1), (1980), 44-59

[24] A., Kusiak and J. Wang, Efficient organizing of design activities, International Journal of Production Research, 31(4), (1993), 753-769

[25] F. AltSahlla, E. Johnson and P. Will, Is concurrent engineering always a sensible proposition? IEEE Transactions on Engineering Management, 42(2), (1995), 166-170

[26] V. Krishnan, S.D. Eppinger, and D.E. Whitney, A model-based framework to overlap product development activities, Management Science, 43(4), (1997), 437-451

[27] D.N. Ford and J.D. Sterman, Dynamic modelling of product development processes, System dynamics Review, 14(1), (1998), 31-68

[28] D.N. Ford and J.D. Sterman, Overcoming the 90% syndrome: iteration management in concurrent development projects, concurrent engineering: research and applications, 11(3), (2003), 177-186

[29] D. C. Wynn, C. M. Eckert and P. J. Clarkson, Applied Signposting: A Modeling Framework to Support Design Process Improvement. In: *Proceedings of IDETC/CIE 2006*, Philadelphia, USA, DETC2006-99402

[30] S.G. Lee, K.L. Ong and L.P. Khoo, Control and Monitoring of Concurrent Design Tasks in a Dynamic Environment, Concurrent Engineering, 12(1), (2004) 59-66

[31] C. Alexander, Notes on the synthesis of form, Harvard University Press, Cambridge, 1964

[32] A. Kusiak and K. Park, Concurrent engineering: decomposition and scheduling of design activities, International Journal of Product Research, 28(10), (1990), 1883-1900

[33] O. Maimon and D. Braha, On the complexity of the design synthesis problem, IEEE Transactions on Systems, Man, and Cybernetics – Part A, Systems and Humans, 26(1), (1996), 142-151

[34] Y. Zeng, and P. Gu,, A science-based approach to product design theory, Part I: formulation and formalization of design process, Robotics and Computer Integrated Manufacturing 15, (1999) 331- 339

[35] F. Mistree, B. Bras, W.F. Smith and J.K. Allen, Modeling Design Process: A Conceptual, Decision-Based Perspective, International Journal of Engineering Design and Automation, 1(4), (1996),209-221

[36] W. Kunz and H.W.J. Rittel, Issues as elements of Information Systems. Center for Planning and Development Research, (1970), Berkeley, USA

[37] J. Conklin, A. Selvin, S.B. Shum, M. Sierhuis, Facilitated Hypertext for Collective Sensemaking: 15 years on from gIBIS. Proc. 12th ACM Conference on Hypertext and HypermediaArhus, Denmark, (2001)

[38] R. H. Bracewell and K.M. Wallace, A tool for capturing design rationale. In: *International Conference on Engineering Design (ICED'03)*, (2003), Stockholm, Sweden.

[39] M. Fathianathan and J.H. Panchal, Design Nodes: An Approach for Modeling Design Processes, Proceedings of DET 2007,4th International Conference on Digital Enterprise Technology, Bath, U.K. 19-21 September 2007, 66-75

[40] R.R. Gorti, A. Gupta, G.J. Kim, R.D. Sriram and A. Wong, An Object_oriented Representation for Product and Design Process, Computer Aided Design, 30(7), (1998), 489-50,

[41] H. Park and M. R. Cutkosky, Framework for Modeling Dependencies in Collaborative Engineering Processes. *Research in Engineering Design* 1, (1999), 84-102.

[42] F. A. Barrientos, I. A. Tumer and D. G. Ullman, Modeling Uncertainty Reduction in Concurrent Engineering Design Teams. In: *Proc. IDETC/CIE 2007,* (2007). Las Vegas, Nevada, DETC2007-35581.

[43] S. M. Qureshi, J J. Shah, S. D. Urban, E. Harter, C. Parazzoli and T. Bluhm, (1997). Integration Model to Support Archival Of Design History In Databases. In: Proceedings of DETC97, Sacramento, California, September 1997

[44] O. Gonikhin and A.J. Medland, Use of networks in describing the design to manufacturing process, Computer-Integrated Manufacturing Systems, 3(3), (1990), 171-177

[45] D.A. Tacconi and L.L. Frank, A new matrix model for discrete event systems: application to simulation, IEEE Control Systems Management, 17, (1997), 62-71

[46] S. Guan, H. Matsuda and M. Nakamura, Scheduling for Farm Work Planning based on Petri Net Model and Simulated Annealing, Agricultural Information Research, 16(3), (2007), 188-195

[47] J., Kim, A.A. Desrochers and A.C. Sanderson, Task planning and project management using Petri nets, 1995 IEEE International Symposium on Assembly and Task Planning, p.0265

[48] A. Arkin, Business Process Modelling Language, version 1.0, BPMI, November 2002

[49] 18629-1, I. 2004. Industrial automation systems and integration – Process specification language – Part 1: Overview and basic principles

[50] E. Frankenberger and P. Badke-Schaub, Modelling design processes in industry – empirical investigations of design work in practice, Automation in construction, 7(2), (1998), 139-155

[51] A.P. Conway, A.J. Wodehouse, W.J. Ion, N.P. Juster, A study of information & knowledge generated during engineering design meetings, 16th International Conference on Engineering Design, ICED'07, 26-31 August, 2007, Paris, France

[52] J. S. Simoff and M.L. Maher, Ontology-based multimedia data mining for design information retrieval, Computing in Civil Engineering (Proceedings of the International Computing Congress held in conjunction with the 1998 ASCE Annual Convention, boston, Massachusetts, October 18-21, 1998), 1998, 212-223

[53] G. Huet, C.A. McMahon, F. Sellini, S.J. Culley and C. Fortin, Knowledge loss in design reviews, Advances in Integrated Design and Manufacturing in Mechanical Engineering II, Edited by Tichkiewitch, S., Tollenaere, M. and Ray, P., Published by Springer, ISBN 978-4020-6760-0, 227-292

[54] Resource Description Framework (RDF), http://www.w3.org/RDF/

[55] OWL Web Ontology Language Overview, http://www.w3.org/TR/owl-features/

[56] Simple Object Access Protocol (SOAP), http://xml.coverpages.org/soap.html

[57] Commerce XML Resources, http://www.cxml.org/

[58] M. Kay, XSL Transformations (XSLT) Version 2.0, http://www.w3.org/TR/xslt20/ (accessed 24 January 2007)

[59] S. DeRose and E. Maler and D. Orchard, XML Linking Language (XLink) Version 1.0, http://www.w3.org/TR/xlink/ (accessed by 27 June 2001)

[60] S. DeRose, R.D. Jr., P. Grosso, E. Maler, J. Marsh and N. Walsh, XML Pointer Language (Xpointer), http://www.w3.org/TR/xptr/ (accessed 16 August 2002)

[61] J. Marsh, D. Orchard and D. Veillard, XML Inclusions (Xinclude) Version 1.0 (Second Edition), http://www.w3.org/TR/xinclude/, (accessed 15 November 2006)

[62] S. Pepper, The TAO of Topic Maps, (2002) [Online Resource] http://www.ontopia.net/topicmaps/materials/tao.html (accessed 1 August 2006)

[63] R. Barta, Is He The One? Subject Identification in Topic Maps, (2003), [Online Resource] http://topicmaps.it.bond.edu.au/docs/21/toc (accessed 1 April 2006)

[64] L. M. Garshol and R. Barta, Information Technology — Document Description and Processing Languages: Topic Maps Query Language. ISO/IEC JTC1/SC34, (2005)

[65] BS ISO/IEC 13250: 2000 Information Technology. SGML Applications, Topic Maps, (2000).

[66] Topicmaps.Org XML Topic Maps (XTM) 1.0, (2001) [Online Resource] http://www.topicmaps.org/xtm/ (accessed 1 April 2006)

[67] BS ISO/IEC 13250-3:2007 Information Technology - Topic Maps - Part 3: XML syntax, (2007)

[68] BS ISO/IEC 13250-2:2006 Information Technology - Topic Maps - Part 2: Data model, (2006)

[69] Cover, R. TopicMaps.Org Consortium Continues Development Efforts within OASIS [Online Resource] http://xml.coverpages.org/ni2001-10-02-a.html (2001)

[70] T. Bandholtz,, P. Durusau, E. Freese, L. M. Garshol, J. Mason, S. Pepper and S. Stringer-Hye, OASIS Topic Maps Published Subjects Technical Committee - Meeting Minutes : 2004-11-01 : Conference Call [Online Resource] http://www.oasis-open.org/committees/download.php/9956/min110104.html (accessed 1 April 2006)

[71] W3c (1998) Document Object Model (DOM) Level 1 Specification.

[72] Simulation Process Studio, http://newsletter.plmworld.org/archive/Vol3No3/Ogilvie.php, (accessed 26 April, 2008)

[73] D. Campbell, S. J. Culley, C. A. McMahon and F. Sellini, An Approach for the Capture of Context-dependent Document Relationships Extracted from Bayesian Analysis of Users' Interactions with Information. *Journal of Information Retrieval*, 6, (2006), 115-141

[74] J. M. Ritchie, R. C. W. Sung, G. Robinson, P. N. Day, R. G. Dewar, J. R. Corney and J. E. L. Simmons, *Part A: Automated Design Analysis, Assembly Planning and Motion Study Analysis using Immersive Virtual Reality*. 2nd Advanced Study Institute on Product Engineering Tools and Methods based on Virtual Reality. Chania, Crete. 30 May – 6 June 2007

[75] F. Noel, & Brissaud, D. (2003) Dynamic data sharing in a collaborative design environment. *International Journal of Computer Integrated Manufacturing*, 16, 546 – 556

[76] A. Conway, M. D. Giess, L. Ding, Y. M. Goh, A. Lynn, W. Ion and C. McMahon, *Holistic Engineering Design: A Combined Synchronous and Asynchronous Approach.*

ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference 2008. New York, USA. 3-6 August, 2008

[77] M. D. Giess, A. Conway, C. Mcmahon and W. Ion, *The Integration Of Synchronous And Asynchronous Design Activity Records.* Design2008. Dubrovnik, Croatia. May 19-22, 2008

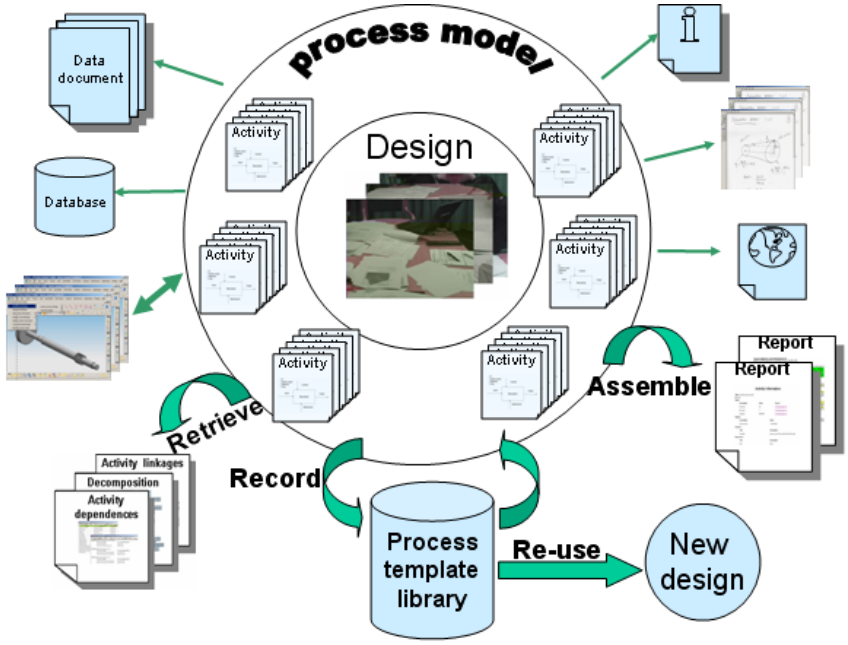


Figure 1 The proposed framework of component-based process model

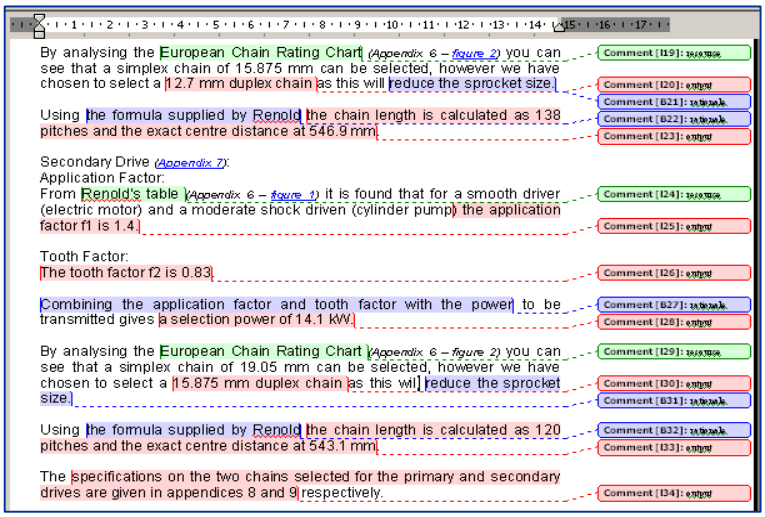


Figure 2: An example of design report

Using the pump input power and the various efficiencies; the rated motor power can be calculated as 11.67 kW. Using the calculated rated power a motor has been selected, the motor is a three phase 4 pole AC motor and the rated power is 15 kW. The motor is 4PE160L. The motor provides plenty enough power for this application; the load speed of this motor is 1450 rpm.

(a)

The motor selected has to have 2 main purposes. The first one is being sure that the output power will enable the pump to work correctly; the second is to have a speed closely proportional to the pump speed. For this last condition to be successful, the motor speed has been set to be of around 1500, enabling the system to have an overall 10:1 speed ratio. To be able to drive the pump, the motor should have a power output of at least 11.67 kW. After searching in catalogues and online data, a suitable motor was found: The following Data applies to this motor: 4PE160L.

(b)

A minimum power of 11.67kW is required. This means a 3-phase, 4-pole induction motor running at a synchronous speed of approximately 1500 rpm would be most appropriate. Consequently it will be possible to power the motor using a mains supply. Now looking in the catalogues, the most appropriate motor can be found:

(c)

Figure 3 Examples of design reports for activities of motor selection

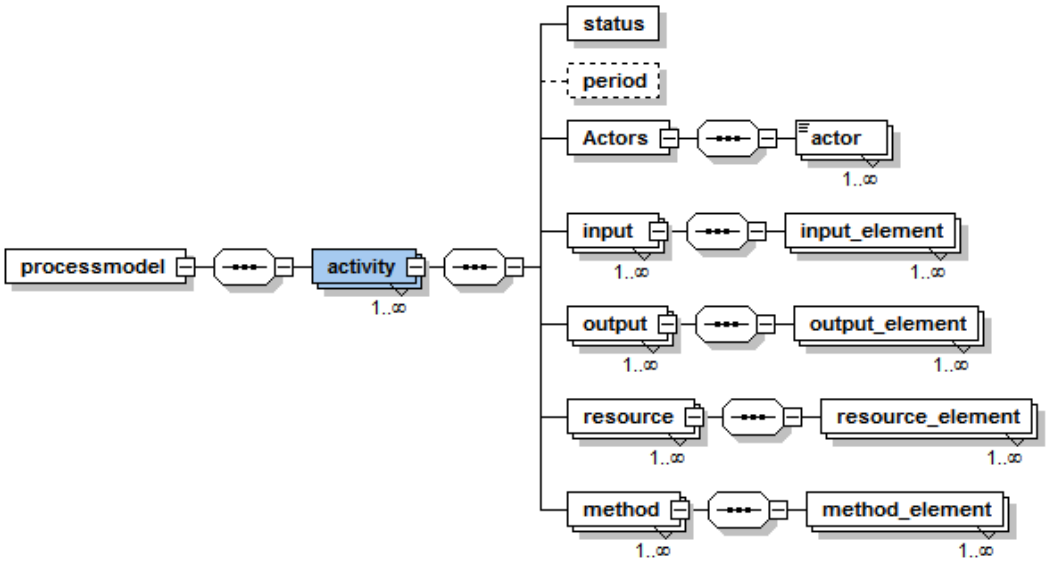


Figure 4: XML schema of activity definition

```
<?xml version="1.0" encoding="UTF-8"?>
<processmodel xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" id="889923" xsi:noNamespaceSchemaLocation="
pmLDSchema.xsd">
  <activity desc="motor_choose" id="301" title="Choose_motor" type="abstract" objective="Choose_motor">
    <status id="302" value="completed"/>
    <period endTime="2007-10-10T14:00:00" id="AC001PR" startTime="2007-10-10T12:00:00"/>
    <actors id="342">
      <actor id="00723" name="Mike Lee" source="people_design_group1.xml"/>
    </actors>
    <input id="297">
      <input_element  id="757" title="power_motor" type="real" desc="power_motor=15KW" source="
Calculate_motor_power_source.xml"/>
      <input_element id="758" title="speed_motor" type="real" desc="speed_motor=1500rev/min" source="
Computer_motor_speed_source.xml"/>
    </input>
    <output id="759">
      <output_element id="motor_model" title="motor_model" type="text" desc="motor_model=4PE160L"  source="
Choose_motor_source.xml" />
    </output>
    <resource id="299">
      <resource_element id="305" desc="catalog" title="motor_catalog" type="catalog" source="sg-motors.pdf"/>
    </resource>
    <method id="300">
      <method_element desc="motor_model must satisfied with motor_power and speed_motor" id="306" source="
Choose_motor_method.xml"/>
    </method>
  </activity>
</processmodel>
```

Figure 5 XML document for motor selection

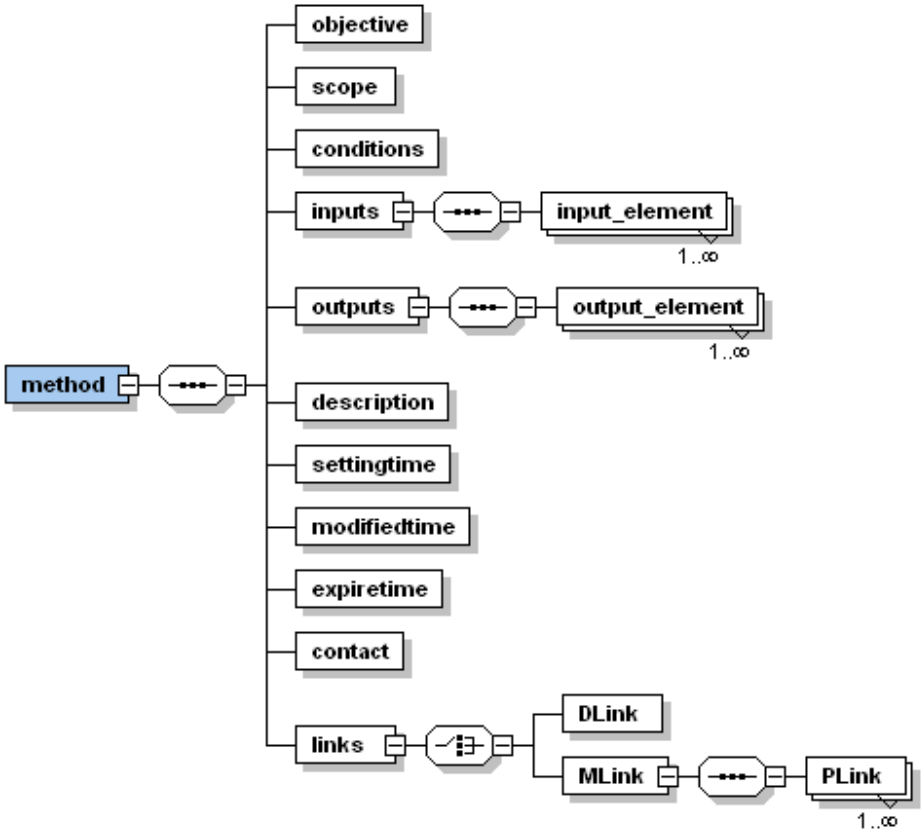
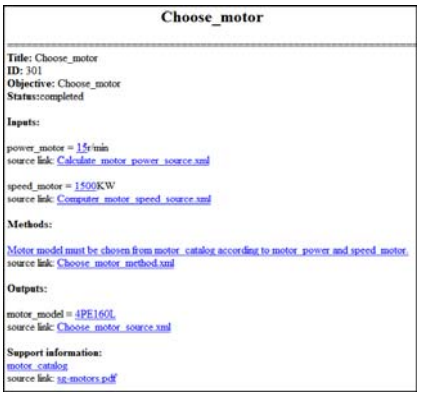


Figure 6: XML schema of process template definition



Choose_motor

Title: Choose_motor
ID: 301
Objective: Choose_motor
Status:completed

Inputs:

power_motor = 15r/min
source link: Calculate_motor_power_source.xml

speed_motor = 1500KW
source link: Computer_motor_speed_source.xml

Methods:

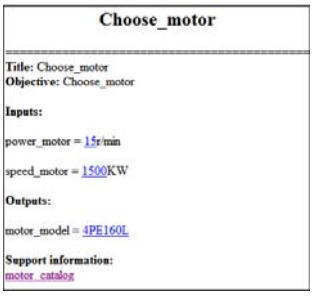
Motor model must be chosen from motor_catalog according to motor_power and speed_motor.
source link: Choose_motor_method.xml

Outputs:

motor_model = 4PE160L
source link: Choose_motor_source.xml

Support information:
motor_catalog
source link: sg-motors.pdf

(a)



Choose_motor

Title: Choose_motor
Objective: Choose_motor

Inputs:

power_motor = 15r/min

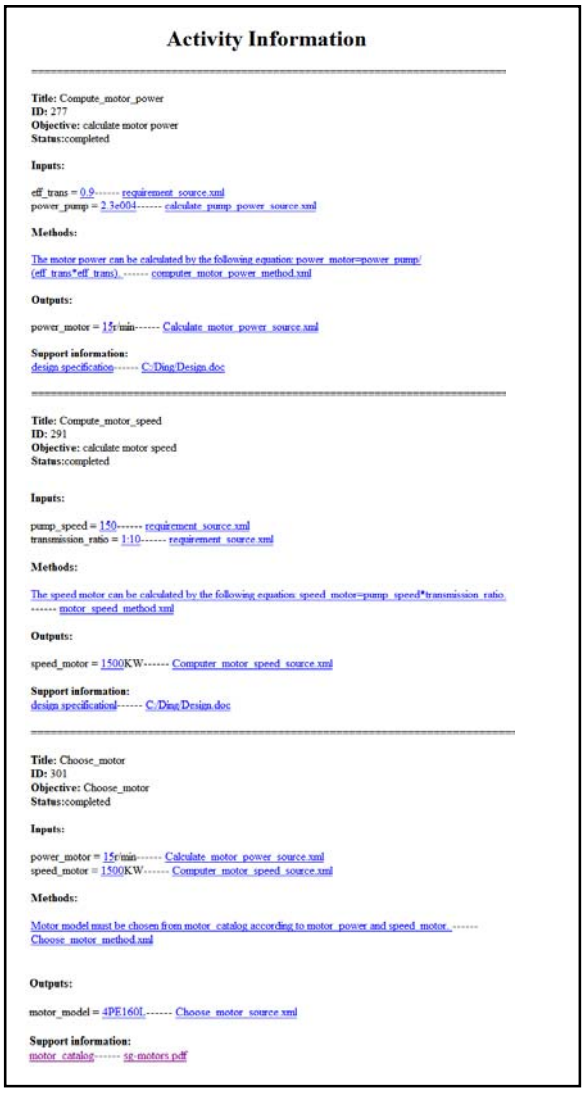
speed_motor = 1500KW

Outputs:

motor_model = 4PE160L

Support information:
motor_catalog

(b)



Activity Information

Title: Compute_motor_power
ID: 277
Objective: calculate motor power
Status:completed

Inputs:

eff_trans = 0.9------ requirement_source.xml
power_pump = 2.3e004------ calculate_pump_power_source.xml

Methods:

The motor power can be calculated by the following equation: power_motor=power_pump/(eff_trans*eff_trans)------ computer_motor_power_method.xml

Outputs:

power_motor = 15r/min------ Calculate_motor_power_source.xml

Support information:
design specification------ C:/Ding/Design.doc

Title: Compute_motor_speed
ID: 291
Objective: calculate motor speed
Status:completed

Inputs:

pump_speed = 150------ requirement_source.xml
transmission_ratio = 1:10------ requirement_source.xml

Methods:

The speed motor can be calculated by the following equation: speed_motor=pump_speed*transmission_ratio.
------ motor_speed_method.xml

Outputs:

speed_motor = 1500KW------ Computer_motor_speed_source.xml

Support information:
design specification------ C:/Ding/Design.doc

Title: Choose_motor
ID: 301
Objective: Choose_motor
Status:completed

Inputs:

power_motor = 15r/min------ Calculate_motor_power_source.xml
speed_motor = 1500KW------ Computer_motor_speed_source.xml

Methods:

Motor model must be chosen from motor_catalog according to motor_power and speed_motor.------
Choose_motor_method.xml

Outputs:

motor_model = 4PE160L------ Choose_motor_source.xml

Support information:
motor_catalog------ sg-motors.pdf

(c)

Figure 7: Examples of various design reports generated

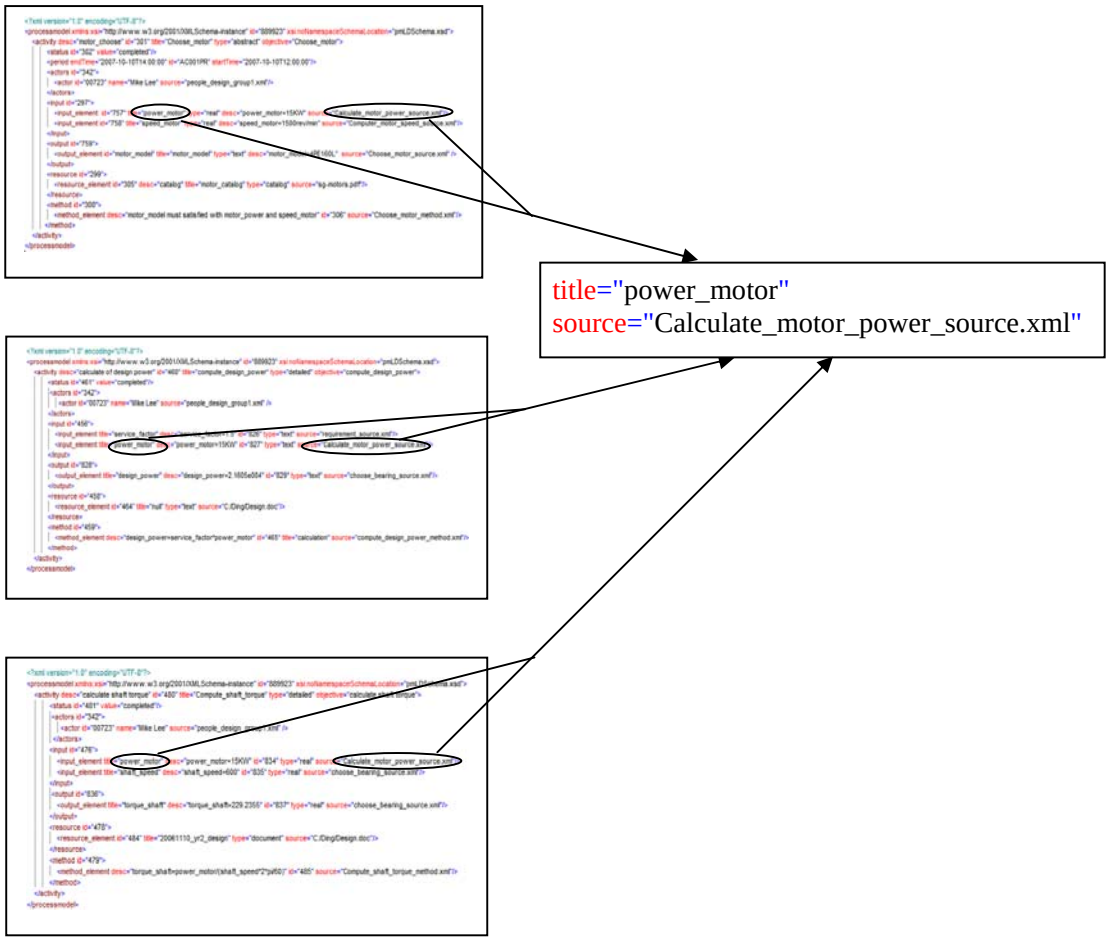
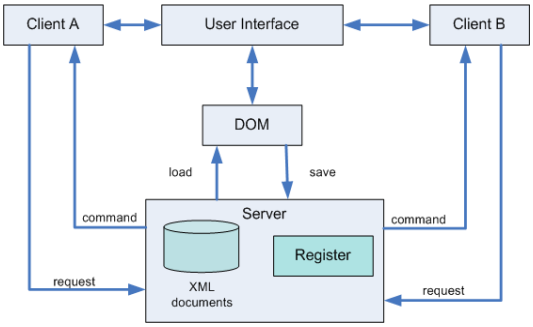
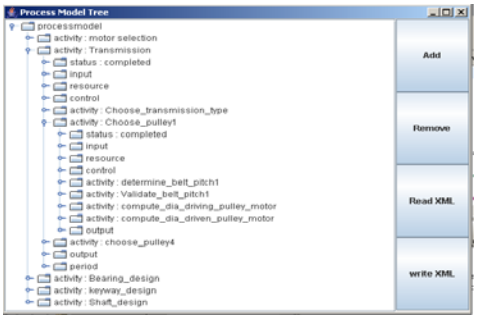


Figure 8 Example of Sharing-resource relationship



(a) Client-Server structure of the implementation



(b) Interface of the implementation

Figure 9: Implementation

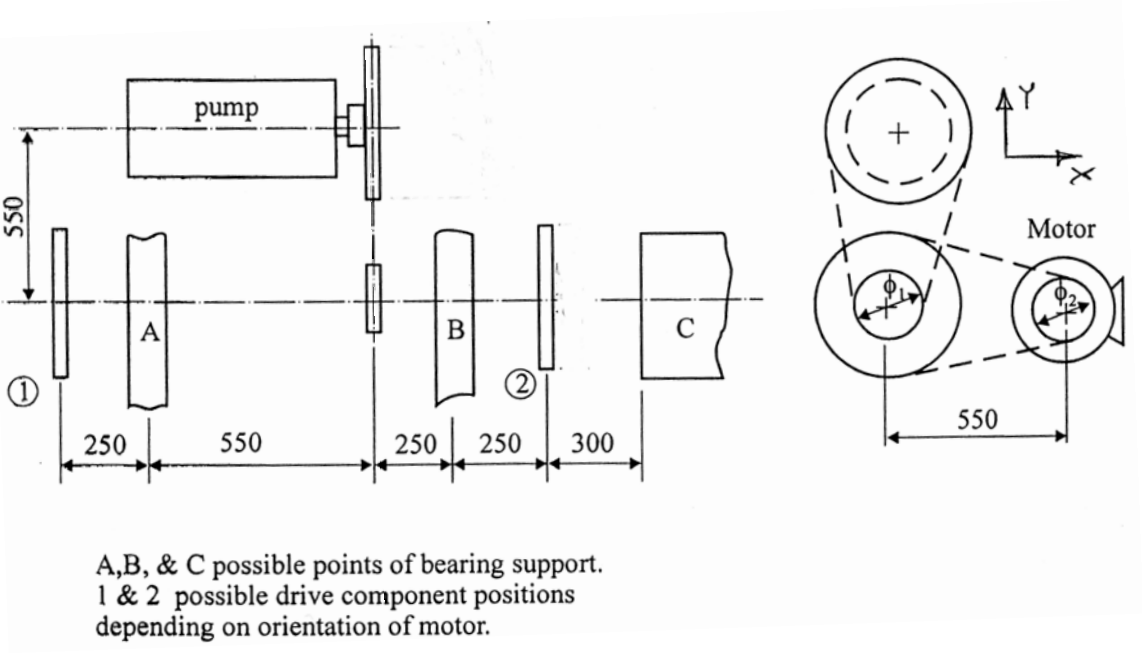
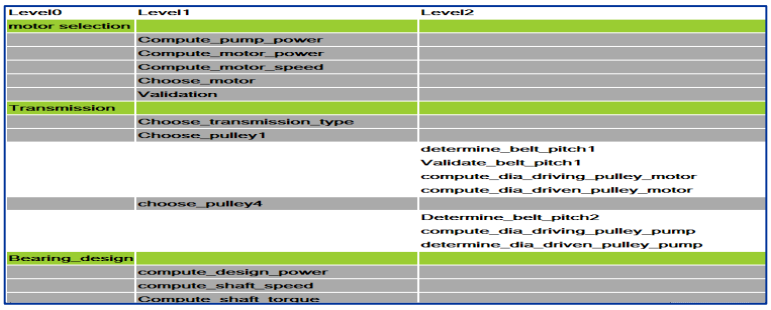
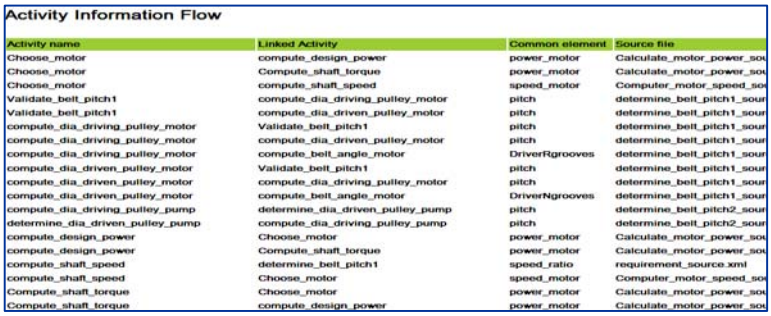


Figure 10: Layout of the case study



Level0	Level1	Level2
motor selection		
	Compute_pump_power	
	Compute_motor_power	
	Compute_motor_speed	
	Choose_motor	
	Validation	
Transmission		
	Choose_transmission_type	
	Choose_pulley1	
		determine_belt_pitch1 Validate_belt_pitch1 compute_dia_driving_pulley_motor compute_dia_driven_pulley_motor
	choose_pulley4	
		Determine_belt_pitch2 compute_dia_driving_pulley_pump determine_dia_driven_pulley_pump
Bearing_design		
	compute_design_power	
	compute_shaft_speed	
	Compute_shaft_torque	

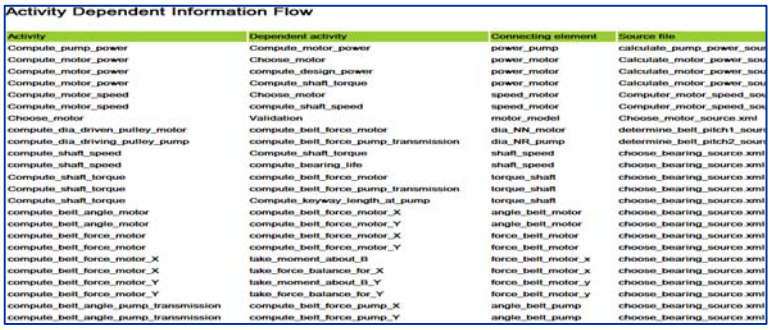
(a) Decomposition diagram of the activities



Activity Information Flow

Activity name	Linked Activity	Common element	Source file
Choose_motor	compute_design_power	power_motor	Calculate_motor_power_sou
Choose_motor	Compute_shaft_torque	power_motor	Calculate_motor_power_sou
Choose_motor	compute_shaft_speed	speed_motor	Computer_motor_speed_so
Validate_belt_pitch1	compute_dia_driving_pulley_motor	pitch	determine_belt_pitch1_sour
Validate_belt_pitch1	compute_dia_driven_pulley_motor	pitch	determine_belt_pitch1_sour
compute_dia_driving_pulley_motor	Validate_belt_pitch1	pitch	determine_belt_pitch1_sour
compute_dia_driving_pulley_motor	compute_dia_driven_pulley_motor	pitch	determine_belt_pitch1_sour
compute_dia_driving_pulley_motor	compute_belt_angle_motor	DriverRgrooves	determine_belt_pitch1_sour
compute_dia_driven_pulley_motor	Validate_belt_pitch1	pitch	determine_belt_pitch1_sour
compute_dia_driven_pulley_motor	compute_dia_driving_pulley_motor	pitch	determine_belt_pitch1_sour
compute_dia_driven_pulley_motor	compute_belt_angle_motor	DriverNgrooves	determine_belt_pitch1_sour
compute_dia_driving_pulley_pump	determine_dia_driven_pulley_pump	pitch	determine_belt_pitch2_sour
determine_dia_driven_pulley_pump	compute_dia_driving_pulley_pump	pitch	determine_belt_pitch2_sour
compute_design_power	Choose_motor	power_motor	Calculate_motor_power_sou
compute_design_power	Compute_shaft_torque	power_motor	Calculate_motor_power_sou
compute_shaft_speed	determine_belt_pitch1	speed_ratio	requirement_source.xml
compute_shaft_speed	Choose_motor	speed_motor	Computer_motor_speed_so
Compute_shaft_torque	Choose_motor	power_motor	Calculate_motor_power_sou
Compute_shaft_torque	compute_design_power	power_motor	Calculate_motor_power_sou

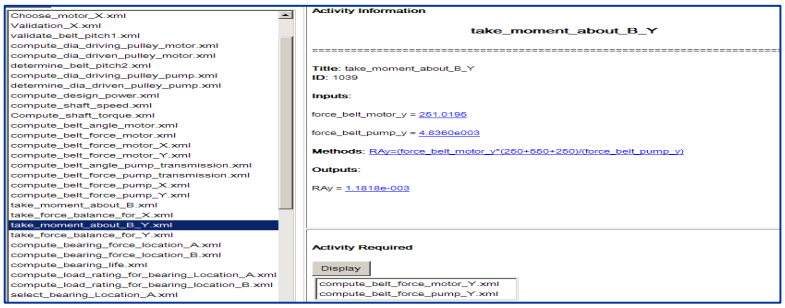
(b) Organisation of the activities with information dependencies



Activity Dependent Information Flow

Activity	Dependent activity	Connecting element	Source file
Compute_pump_power	Compute_motor_power	power_pump	calculate_pump_power_sou
Compute_motor_power	Choose_motor	power_motor	Calculate_motor_power_sou
Compute_motor_power	compute_design_power	power_motor	Calculate_motor_power_sou
Compute_motor_power	Compute_shaft_torque	power_motor	Calculate_motor_power_sou
Compute_motor_speed	Choose_motor	speed_motor	Computer_motor_speed_so
Compute_motor_speed	compute_shaft_speed	speed_motor	Computer_motor_speed_so
Choose_motor	Validation	motor_model	Choose_motor_source.xml
compute_dia_driven_pulley_motor	compute_belt_force_motor	dia_NN_motor	determine_belt_pitch1_sour
compute_dia_driving_pulley_pump	compute_belt_force_pump_transmission	dia_NR_pump	determine_belt_pitch2_sour
compute_shaft_speed	Compute_shaft_torque	shaft_speed	choose_bearing_source.xml
compute_shaft_speed	compute_bearing_life	shaft_speed	choose_bearing_source.xml
Compute_shaft_torque	compute_belt_force_motor	torque_shaft	choose_bearing_source.xml
Compute_shaft_torque	compute_belt_force_pump_transmission	torque_shaft	choose_bearing_source.xml
Compute_shaft_torque	Compute_keyway_length_at_pump	torque_shaft	choose_bearing_source.xml
compute_belt_angle_motor	compute_belt_force_motor_X	angle_belt_motor	choose_bearing_source.xml
compute_belt_angle_motor	compute_belt_force_motor_Y	angle_belt_motor	choose_bearing_source.xml
compute_belt_force_motor	compute_belt_force_motor_X	force_belt_motor	choose_bearing_source.xml
compute_belt_force_motor	compute_belt_force_motor_Y	force_belt_motor	choose_bearing_source.xml
compute_belt_force_motor_X	take_moment_about_B	force_belt_motor_x	choose_bearing_source.xml
compute_belt_force_motor_X	take_force_balance_for_X	force_belt_motor_x	choose_bearing_source.xml
compute_belt_force_motor_Y	take_moment_about_B_Y	force_belt_motor_y	choose_bearing_source.xml
compute_belt_force_motor_Y	take_force_balance_for_Y	force_belt_motor_y	choose_bearing_source.xml
compute_belt_angle_pump_transmission	compute_belt_force_pump_X	angle_belt_pump	choose_bearing_source.xml
compute_belt_angle_pump_transmission	compute_belt_force_pump_Y	angle_belt_pump	choose_bearing_source.xml

(c) Organisation of the activities sharing common information resources



(d) A text report of an activity with its support activities

Figure 11 Examples of various reports generated from the case study

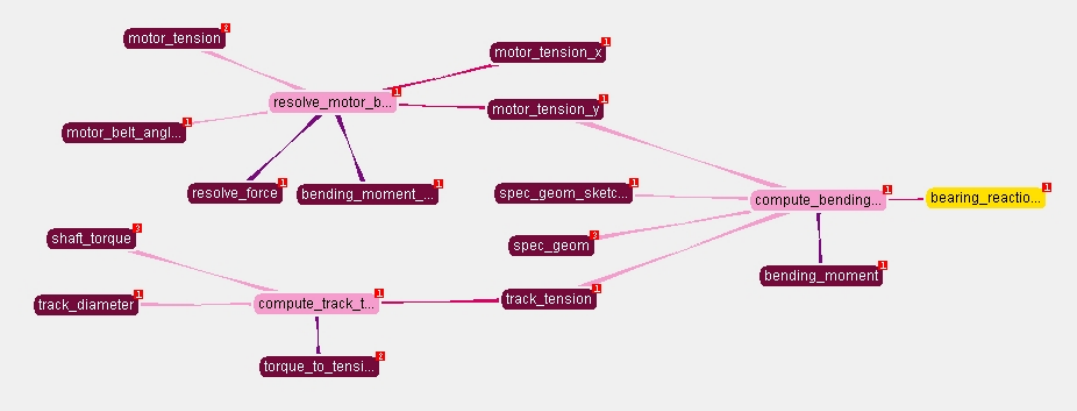


Figure 12 Sample Topic Map Visualisation

Table 1: the summary of the specification of the design project

Pump speed	150 rev/min
Delivers	2 litres/s
Delivery pressure	3.5 MPa
Pump efficiency	90%
Tolerance of the axial dimension	±20 mm
Tolerance of the centre distance between motor, pump and shaft	±50 mm
Project life	5 years at 250 days a year and 8 hours a day

Table 2: XML documents for the activities captured

Computer_pump_power_X.xml	take_force_balance_for_Y.xml
Computer_motor_power_X.xml	compute_bearing_force_location_A.xml
Compute_motor_speed_X.xml	compute_bearing_force_location_B.xml
Choose_motor_X.xml	compute_bearing_life.xml
Validation_X.xml	compute_load_rating_for_bearing_Location_A.xml
Validate_belt_pitch1.xml	compute_load_rating_for_bearing_Location_B.xml
compute_dia_driving_pulley_motor.xml	select_bearing_Location_A.xml
compute_dia_driven_pulley_motor.xml	selection_bearing_location_B.xml
determine_belt_pitch2.xml	compute_keyway_length_motor.xml
compute_dia_driving_pulley_pump.xml	compute_keyway_length_at_pump.xml
determine_dia_driven_pulley_pump.xml	design_shaft.xml
compute_design_power.xml	determine_belt_pitch1.xml
compute_shaft_speed.xml	choose_transmission_type.xml
Compute_shaft_torque.xml	Transmission.xml
compute_belt_angle_motor.xml	Choose_pulley1.xml
compute_belt_force_motor.xml	choose_pulley4.xml
compute_belt_force_motor_X.xml	compute_force_on_motor.xml
compute_belt_force_motor_Y.xml	compute_force_on_pump.xml
compute_belt_angle_pump_transmission.xml	compute_bearing_force_in_X.xml
compute_belt_force_pump_transmission.xml	compute_bearing_force_in_Y.xml
compute_belt_force_pump_X.xml	Bearing_selection.xml
compute_belt_force_pump_Y.xml	Bearing_design.xml
take_moment_about_B.xml	keyway_design.xml
take_force_balance_for_X.xml	Motor_selection_level.xml
take_moment_about_B_Y.xml	