Journal of Defence Modeling and Simulation

Relational Oriented Systems Engineering Framework for Flight Training

Charles Dickerson, Trevor Holden
Loughborough University, Leicestershire, LE11 3TU, UK

Abstract

The integration of systems of systems (SoS) associated with a flight training mission directly reflects the problem of developing a system engineering process for the design of Live, Virtual and Constructive experiments. Due to the complexity and disparity of technology in a flight training system of systems (FTSoS), modeling and analysis of architecture is becoming increasingly important. Relational Oriented Systems Engineering (ROSE) methodology is used to develop a framework for simulation and analysis of a navigational system of systems for a typical aircraft. The framework can be used for both prescription of navigation systems entering and exiting the SoS and for analysis of pilot behavior as navigation quality of service (QoS) changes. ROSE offers a novel approach to developing a model based systems engineering (MBSE) process for simulation and analysis of a complex SoS problem.

I. Introduction

Traditional approaches to the management of information in system of systems (SoS) are based on document-centric workflows, sequential work processes that embody information concerning the requirements, constraints, architecture, designs, decisions and other information about the SoS. A flight training system of systems (FTSoS) is a mix of Live, Virtual and Constructive (LVC) systems including documents, desktop PC-based training, ground simulators, training aircraft, information intensive subsystems, and a number of other subsystems that training

aircraft carry for flight realism. Current FTSoS entail ad-hoc approaches to circumvent issues surrounding integration and interoperability of new and disparate technology that are not repeatable and often paper documents are used across domains with no direct convergence between them. The solution to the integration and interoperability issues for FTSoS historically rely on large investments of subject matter expert (SME) man-hours.

Traditionally, changes between documents or to related elements outside the documents have led to various documents becoming outdated, inconsistent and in conflict with each other. Given the size and complexity of today's and future SoS, development can take considerable time and effort with the unfortunate consequence of the end result being incomplete and insufficient. However, machine readable models permit information to be more readily available, changes can be easily accommodated and traceability can be automated to propagate to all related elements / domains throughout the system representation. In addition, models can provide universal communication to all stakeholders and integrate with multiple modeling domains across life cycle from SoS to components. The models can represent a variety of views to gain a greater understanding of requirements and design choices to be made. Thus, one solution to management difficulties of the FTSoS is to use a model-based systems engineering (MBSE) approach. With MBSE, machine readable models form the core of all the systems engineering activities and permit seamless development by allowing collaboration across disciplines and supply chains, in addition to facilitating exchange of electronic data [1].

A Flight Navigational training

A flight plan is submitted before departure and entered into a flight management system (FMS) or it can be selected from a library via the aircraft communications addressing and reporting system (ACARS) [2] data-link. During a flight the pilot uses the FMS to modify the flight plan;

the FMS also sends the flight information to the navigation display via the electronic flight instrument system (EFIS). Once in flight, the FMS integrates the position estimates from a variety of disparate navigation sensors to determine actual aircraft position. During the flight the FMS constantly samples the data from the sensors to amend the aircraft position and accuracy; for both military tactical air navigation (TACAN) [3] and the standard VHF Omni-directional Range/Distance Measuring Equipment (VOR/DME); the information is presented to the pilot in the same way.

Legacy aircraft, however, generally used in FTSoS may not have the facility to integrate readings from various navigation sensors; thus the pilot must make a decision on the actual aircraft position based on disparate sensor readings and different TACAN ground units. As the aircraft navigates through the flight plan, navigation sensors entering and exiting the SoS will affect the aircraft position calculated by the FMS aboard the aircraft.

During the flight, the navigation Quality of Service (QoS) and thus the accuracy of readings presented to the pilot will fluctuate. The display of information from the disparate sensors that make-up the navigation aids within an aircraft affect pilot's behavior, decision making and judgment processes. An important aspect of flight training is the behavior such as, actions or reactions of the pilot to these variations in navigation readings. The current FTSoS rely on the best judgment of trainers; thus monitoring a number of student pilots to determine the 'state' of progress is not performed in a formal manner.

B Integrated architecture for Design and Analysis

Providing true virtual design and analysis with supporting architectures has been a keen area of research for a number of years. A number of software architectures have been investigated to support integrating systems for distributed systems including High Level Architectures (HLA), Distributive Interactive Simulation (DIS) Test and Training Enabling Architecture (TENA).

HLA and its associated standard for modeling and simulation, IEEE 1516, is a general purpose architecture for distributed computer simulation systems [4] to enable reuse and interoperation of simulations. HLA provides a structure that supports reuse of capabilities available in different simulations for the development of complex simulation applications, including training, analysis and engineering, by providing a common framework within which specific system architectures can be defined. However, applications must use the same run time infrastructure in order to interoperate and therefore different implementations of the middleware would lead to various interoperability issues and reliability limitations [5].

DIS [6] is an IEEE 1278 standard to define an infrastructure for aligning simulations of various types at multiple locations to create realistic, complex, virtual worlds for simulation of highly interactive activities for performing real-time platform-level war-gaming across multiple host computers. The implication of this decentralized environment is that each node retains responsibility for maintaining its own reference model and this increases the possibility of different representations of the same environment that may lead to negative training results.

TENA [7] is designed to promote integrated testing and simulation-based acquisition through the use of large-scale, distributed, real-time synthetic environment, which integrates testing, training, simulation, and high-performance computing technologies, distributed across many facilities, using a common architecture. TENA is based on lessons learned from large-scale distributed real-time systems, and is continuously being revised on real-world user feedback and there are currently no plans for standardization as HLA and DIS have with the IEEE.

TENA, HLA and DIS are not inherently interoperable with each other, therefore, additional interfaces and gateways are required to bridge any gap, which introduces increased complexity, cost and reduces the reuse capability of the supporting models.

Open architecture can assist in resolving the interoperability problems of achieving seamless communication and a shared common understanding of the context (including the environment and time). Open architecture can also permit transparency of aircraft types and systems within the FTSoS. Realizing loose coupling between middleware implementations of simulation services and coordination services through the use of open standards and architectures can enable the FTSoS to achieve the benefits of all three distributed architectures, by enabling model-based integration of mathematical models and simulations and then handling the semantic and syntactic translation on the simulation data itself. Due to the multi-disciplinary nature of the FTSoS, integration flexibility is essential to allow the need to update and evolve the distributed system.

C Model-based approaches and Relation Orientation.

Model-based approaches to software and systems engineering have been evolving over the past half century. The beginnings of MBSE can be traced to Wymore [8] who sought to put systems engineering on a sound mathematical basis and to use transformations between functional models and state diagrams to assure that intended system behaviors were achieved. Model-Driven Architecture (MDA), which focuses on machine readable models and MBSE subsequently began a convergence that resulted in the specification of the Systems Modeling Language (SysML) and the launch of the MBSE Initiative in 2007; which is currently researching modeling and simulation interoperability and how models interact with each other throughout the system lifecycle. The SysML facilitates the ability to show semantically rich relationships between model elements and thus incur the ability to trace requirements through the produced artifacts [9]. Concurrently, the author [10] began a research initiative using the Tarski mathematical theory of models [11] to more fully exploit the relational nature of models, graphical modeling languages, and their transformations [12]. Relational frames have since been developed to define the structure and specifications of the interfaces used in modular open

architecture design. This is a generalization of object-oriented frames described in [13], which are primarily structural slots for allocation of attributes or methods to a class.

D Meeting the challenge for Managing Complex Systems and SoS

Resolving the complexity challenges of an SoS involves solutions to issues of system interoperability and tight coupling to more easily accommodate modification and upgrades of systems entering the SoS and to permit backward compatibility of legacy systems. In MBSE, the models capture the structure and behavior of the system (within architecture) at various levels of detail, promote unambiguous communication and allow sophisticated trade-study analyses that would be more difficult to perform with a document centric system. A model-based relational oriented approach on systems engineering allows the facility for systems to access each other's models and provides a reusable framework for system design and analysis. Relational Oriented Systems Engineering (ROSE) introduced in [14] strengthens this flexibility by accessing the relationships between elements in models and between the model artifacts to provide frameworks within the architecture. ROSE extends object orientation (OO) by evolving MDA to include physics-based mathematical models for design, analysis and technology trade-offs into a rigorous repeatable method. The model driven approach of relational orientation can offer greater concurrency of verification with design as the same models used for design can also be reused for verification. Thus, relational orientation supports access to models of a system in much the same way as UML Classes access each other's software objects. In Section III, a ROSE framework and architecture are used to provide concordance between technical and human aspects of the FTSoS to provide the ability to understand how disparate stimuli affect pilot behavior to meet the challenge of the navigation problem. The ability to analyze the effects in a virtual framework can give the trainer a more detailed understanding of student progress through the pipeline with the facility to assess the effects of technologies entering and exiting the FTSoS.

II. Flight Training as a Complex SoS

Each system within an FTSoS must have sufficient levels of interoperability to support a coherent scheme of training for the pilot under realistic operational conditions. System of systems engineering (SoSE) can permit decision makers to understand the implications of choices and interactions between systems together with allowing a greater degree of training course flexibility while providing demonstrable grading, for the student pilot, to ensure required knowledge, skills and experiences are developed during each phase of aviation training. As a result, the assemblage of flight training systems embodies the core challenge of SoSE and structured repeatable solutions to this problem have significant technical and commercial interest. To assist in capturing and preserving relationships between systems in the FTSoS a proposed metamodel has been created, which is shown in Fig.1.

The FTSoS can be seen as an integration of a variety of disparate systems into a seamless whole. The aircraft represents a real-time avionics system; the ground-based systems communicate with the avionics system and both are integrated into the enterprise systems on the training base.

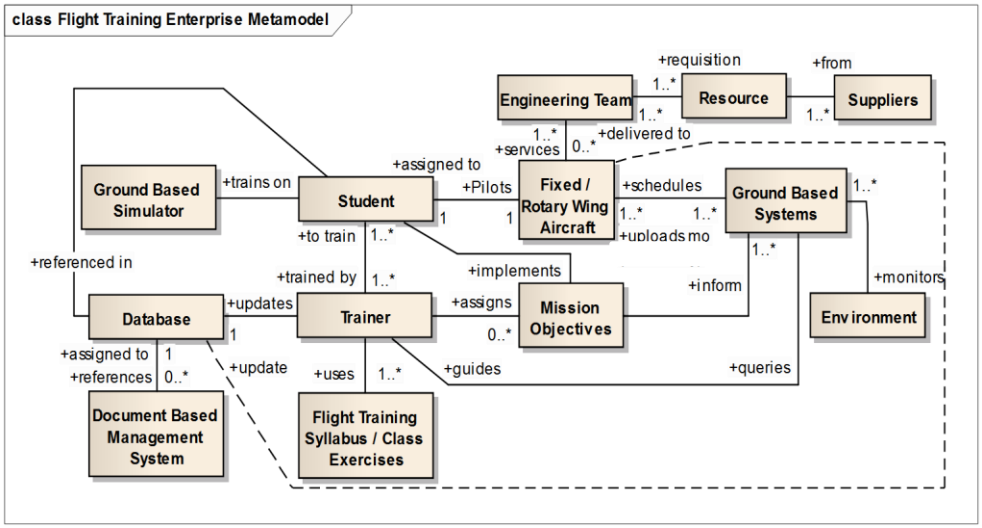


Figure 1. Proposed flight training metamodel for preserving relationships between systems of the FTSoS.

The aircraft is seen as one system that is integrated into an enterprise FTSoS. Each element of the metamodel can be elaborated further to gain additional details to establish domain

metamodels to describe the functions and features of that particular element. In traditional FTSoS, the aircraft operates as an independent element; however in modern training systems and combat systems, the aircraft must integrate closely with automated air traffic controls and associated ground systems. Data management and understanding relationships and technology differences between elements and sub-elements within the metamodel are key concerns for the ability to manage information flow. Traditionally, systems within the FTSoS are integrated in an ad-hoc fashion and acquisition of the aircraft and ground based systems were provided through one contractor as discussed in [15]. The consequence of this closed system architecture is high cost and limits on the ability to take advantage of rapid changes of technology and in software development due to tight coupling, especially in legacy systems.

A Open Architecture for the FTSoS

To alleviate the problem of obsolescence, interoperability and management difficulties associated with technology insertion, designing the FTSoS with an open modular architecture can more easily permit upgrades, additions to and swapping of components to provide a more flexible architecture using standardized key interfaces and more importantly access to commercial-off-the-shelf (COTS) technologies. This can allow the customer to acquire systems from various suppliers/vendors and assemble them together to satisfy the SoS requirements; thus assisting aircraft system architecture integration into an FTSoS architecture and permitting transparency to aircraft types and systems. Open standards in systems design can allow modification of systems as new technology is developed with minimal impact on existing systems to improve off/on-board integration of data and functionality permitting customization.

The proposed FTSoS open architecture is based around the key elements of metamodel in Fig.1. The FTSoS enterprise must manage the transition of student pilot from basic aircraft through to an advanced training jet. Any new aircraft technology and ground-based systems (GBS) entering

the FTSoS, traditionally require ad-hoc adaptations to circumvent integration and interoperability issues. As illustrated in the metamodel of Fig.1, the aircraft is one domain within the FTSoS; this domain describes the features and operations (F&O) of disparate aircraft in their own sub domains, each sub domain can share common F&Os through common standardized open interfaces to produce a more flexible architecture.

B Navigational Viewpoint on the Aircraft and its associated SoS.

For this paper, the navigation system of the aircraft domain is discussed to provide a simple example of the ROSE approach for SoSE. The metamodel of Fig. 2 describes key elements of a typical aircraft navigational system considered as a simple example for the ROSE approach of the importance of understanding the relationships and behavior of the relationships between entities; the performance of which is confined to a standard behavior and their accuracy is governed by the FAA 9840.1 standard.

The fly-by-wire (FBW) system is concerned with controlling the attitude of the aircraft. Inputs from the pilot's controls determine the aircraft dynamics, and how the aircraft will respond at various speeds and attitudes. The autopilot flight director system (AFDS) controls the speed, height, and heading at which the aircraft flies in addition to other navigation functions. The flight management system (FMS) performs navigation or mission functions, ensures the AFDS and FBW systems position the aircraft at the precise point in the sky to coincide with the multiple waypoints that characterize the aircraft route from departure to destination airfields.

The multifunction control and display unit (MCDU) operates as a pilot interface to initiate and monitor the progress of the flight. The electronic flight instrument (EFIS) and the heads-up display (HUD) incorporate the navigational display for the pilot and co-pilot. Sensors incorporate the inertial navigational system/inertial reference system (INS/IRS), distance measuring equipment, VOR, TACAN, global navigation satellite systems, air data systems, fuel

sensors, actuators and engine feedback sensors. Each of these systems form elements of the high-level metamodel for the flight navigation system illustrated in Fig.2.

Each element within the metamodel can be associated with blocks/classes (B/C) or packages in SysML to form the building blocks to describe the structure and architecture of the system. The association lines connecting the elements describe relationships between the elements. Abstraction is at a level of detail that structural relationships between the components of the system can be identified.

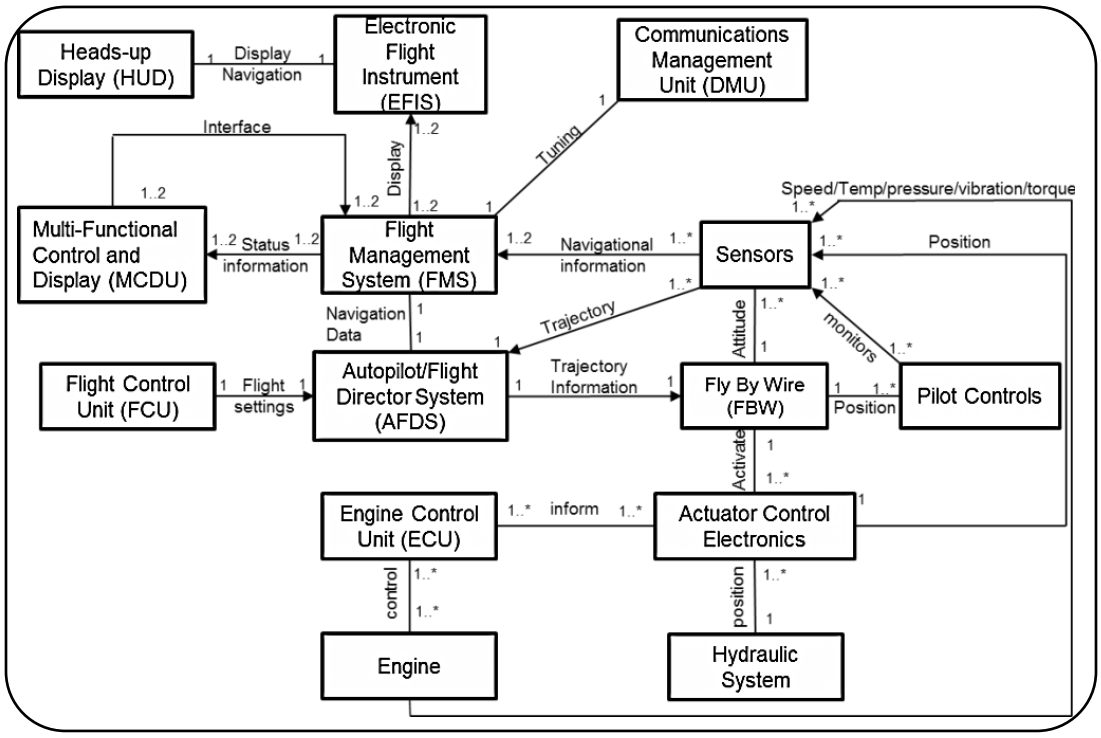


Fig.2 Metamodel of navigation control for typical aircraft

III. Relational Oriented Systems Engineering (ROSE)

ROSE [14] is a general systems methodology that employs a principle of *model specification and relational transformation* for system specification, analysis, and design. The methodology generalizes the hierarchical principle of *definition and decomposition* employed in legacy systems engineering. ROSE also extends the general system models and relational homomorphism used by Klir [16] and Lin [17].

A Model Specification and Relational Orientation

The ROSE methodology is used to specify a framework to capture the relationships between entities in UML/SysML models. The blocks/classes form the frames of the structures within ROSE. The types of relationships that elements of a model can relate to or depend upon each other can range from logical to metric including sensitivities derived from simulation and analytics. From the relational viewpoint, the specification of a model associated with a system is the specification of:

- Entities associated with the system
- Sentences (declarations) about the entities
- Model elements to interpret the sentences
- A semantic structure on the model elements
- Interpretation of the sentences into the semantic structure

Entities are abstractions that admit logical or physical existence. The entities of the system can include attributes, blocks/classes, and components of the system. There can also be entities associated with the system which are not part of it, e.g. the environment. The sentences are the basis for system specification. The model is valid when the interpretation of each sentence is true within the semantic structure; this concept will be referred to as a relational structure. The validation process is facilitated by two types of semantic structures: relational structures (i.e. a set of mathematical relations) and graphical models (e.g. class diagrams).

In general, a relational transformation is specified as an association between the elements or parameters of two models of a system that induces a further mapping between the relationships in the model. The elements of the metamodel are the relational frames of the structures. Relational frames are used to specify semantic structures for organizing knowledge about the system. Given a collection *M* of model elements and a semantic structure R for organizing the

relations on the elements, a relational frame $\underline{M}$ is defined to be the ordered pair (M, R). If M is a collection of mathematical objects, such as numbers, variables, or sets, and R = $\{R_\alpha\}$ is a relational structure on M, then the relational frame $\underline{M}$ = (M, R) becomes Lin's system model [17].

Modeling elements can have four types of *relational association*: (i) relation by belonging to a defined subset of elements (collection of the model elements), (ii) n-ary mathematical relation, (iii) hierarchical association (decomposition of individual model elements), and (iv) association with elements of another model by transformation. The first three types correspond to the internal structure of the model. The associated relational frames will be referred to as frames.

The fourth type of relation is external to the model, although hierarchical decomposition into sub-models can also admit transformations. The frame will be referred to as a transformational frame and denoted as $\underline{Q}$ = (M, N; Q) where M and N are the elements of two models and Q is the association between the structures of the frames. The transformation between the two frames is the association line(s) between the elements in the metamodel of Fig.2.

ROSE is concerned with a requirements frame $\underline{M}$ = (M, R) and a system design frame $\underline{N}$ = (N, S) which are associated by a transformational frame $\underline{Q}$ = (M, N; Q). The collective three frames are referred to as a framework. In relational orientation, systems are modeled using multiple frameworks that represent the various knowledge domains and components of the system. The frameworks are integrated into a framework structure by sharing common frames or by transformations between frames. The specification of frames for the models and transformational frames between the models is complete when they form a framework structure that is adequate for system specification, analysis and design. This resultant framework structure provides a metamodel of the system, i.e. an abstraction used for specifying the models of the system.

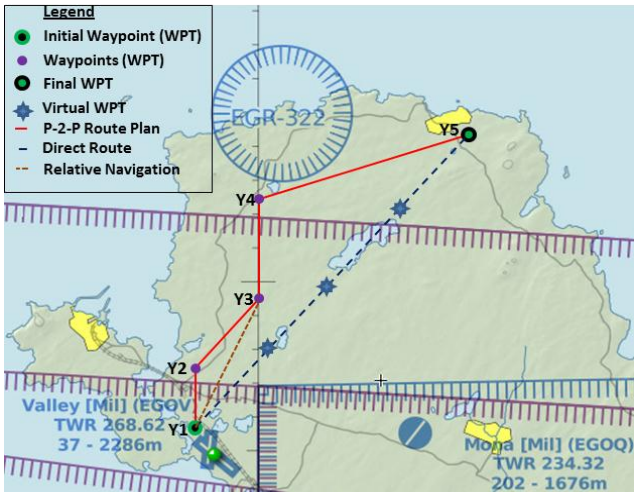
The frame M is a model of the sentences 'W' which specify the system. In the case of the Navigation problem sentence 'W' is from a navigational viewpoint of the flight mission that includes: "initial and final waypoints with timings; possible intermediate waypoints with timings, with any details describing availability of Navigation SoS, and excluded areas of flight", the pilot must then generate a route map from the sentences. It is intended to model the sentences without changing their meaning. The association Q interprets the sentences into the structure of the frame N. The transformational frame Q is central to analysis. The model (*M*, R; W) is the requirements or environmental model and the model (N, S; W) is the system design model where W in the design model is the interpretation of the requirements into the design specification.

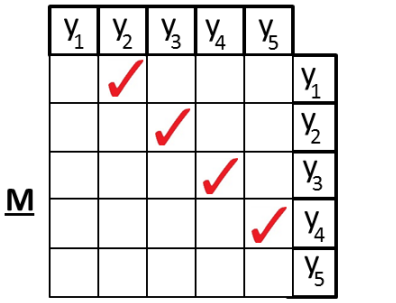
B Specification of Models for the Navigational SoS

The route map for the navigation system described in Section IIB includes geodetic and timing information as the basis for a requirements frame M = (*M*, R); representative of the route map within the 'FMS' element in Fig.2 or manual route plan via a map. Every navigation point and waypoint is an entity represented by a block/class. Waypoints relate to each other by precedence order. The Navigation SoS includes attributes such as: coordinates, metric accuracy, sensing range, and availability as the basis for a design frame N = (*N*, S); this frame is represented by the 'sensors' element of the metamodel in Fig.2. Each navigation system e.g. compass, VOR, RNAV, and maps with geodetic information are also entities represented by B/C. The frames are matrix representations of block definition (BD)/class diagram. Model elements for M and N are derived from the system attributes specified in the BD/class diagram. Fig. 3 illustrates a typical route map in which the pilot navigates from the initial waypoint to the final waypoint via intermediate waypoints, which may depend on the mission requirements ('W').

There are also various navigation strategies, for example: 1) point-to-point (P-2-P) – where waypoints are the navigation points; 2) relative navigation – consideration is given to how many

waypoints are within detection range, and subsets of points are grouped synthetically; 3) Direct – where a shortest route is flown along waypoints that are mostly virtual.





Requirements Model for P-2-P route plan

Figure 3 Navigation route plan describing requirements frame '<u>M</u>'.

Each waypoint along the route plan is given an ID y_i which corresponds to parameters describing instances of waypoints within the frames. The 'tick' marks in the slots of the '<u>M</u>' frame of Fig.3. designates the P-2-P route map where waypoint y_1 precedes y_2, y_2 precedes y_3 etc.; each 'tick' describe a relationship relating to geodetic information, time and distance between the waypoints. Relationships between waypoints determine when to switch primary source of navigation aid. Every waypoint is allocated to one or more navigation system in '<u>N</u>'.

When <u>N</u> and <u>M</u> are associated by a transformational frame Q, the framework structure illustrated in Fig.4 is created. In the other navigational strategies the association Q can be a multi-valued association between *M* and *N*. The notation $y_i Q x_k$ will be used to mean that y_i in *M* has been associated with x_k in *N* by Q that denotes that the ordered pair (y_i, x_k) belongs to Q. When Q is a function as in the P-2-P strategy, i.e. single valued, and the structures on *M* and *N* have the same –arity, the relational transformation is a relational homomorphism between the structures.

The calculation of the transformation of binary relationships is as follows:

$$(y_i, y_j) \in \mathrm{R} \text{ with } (y_i, x_k), (y_j, x_l) \in \mathrm{Q} \text{ implies } (x_k, x_l) \in \mathrm{RQ} \quad (1)$$

If RQ is a subset of one of the relations on *N*, e.g. RQ is a subset of S, then Q is said to induce a *relational transformation* M → N. In the special case when Q is determined by a relational homomorphism, q: $M \rightarrow N$, then equation (1) means if $(y_i, y_j) \in R$ then $(q(y_i), q(y_j)) \in S$ because RQ is a subset of S. A relational transformation Q is specified only at the parametric level and the induced transformation of relationships in M and N is calculated from just the association of parameters specified by Q and the relations on *M* or *N*. For the P-2-P navigation example, the associations made by Q induce a relational transformation N → M illustrated by the framework in Fig. 4. ROSE is then a realization of structured analysis in a classic Black Box/White Box paradigm that separates the external view of a system from the internal view.

The navigation SoS 'N', in Fig.4, reflects the presentation of navigational data to the pilot (N can be decomposed further to show ground based sensors and avionic sensors that are sub-elements).

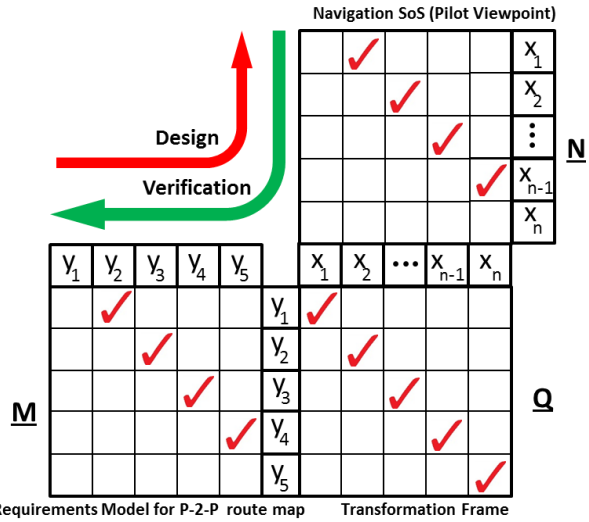


Figure 4. ROSE Framework for P-2-P Navigation Problem

For example, a pilot takes off from an un-instrumented airfield at (y_1) using a compass, he/she has to estimate the right time to transition from the compass to VOR, this is determined by the relationship between the blocks/classes as well as the relationship between the waypoints. For P-2-P the relational transformation would for example start with compass (x_1) and end with VOR (x_2); this is a consequence of doing the binary relationship between y_1 and y_2, which maps over to

x_1 and x_2 (the allocation of x_1 to y_1 and x_2 to y_2 from the route map in Fig. 3 using Q). Another relationship is the time evolution from y_1 to y_2 – implications on transitioning from x_1 to x_2 based on the attributes of x_1 and x_2, i.e. when the VOR becomes more accurate than the compass. The allocation of navigation SoS can cause the creation of new virtual waypoints; when deviations to the route map are required while en-route, a new set of waypoints are required, and these will form new parameters for the M frame.

The association of navigation SoS to waypoints depends on the navigation strategy; the Q frame, is for derived navigation QoS and is used for design decisions to obtain the desired QoS. Navigation QoS is derived from attributes of waypoints in combination with attributes of the navigation SoS en-route in flight through the waypoints. As the pilot flies the aircraft there will be a dynamic evolution of QoS. In general the Q frame is used to capture the total sensitivities of the y_i to the x_l, i.e. a change in value for one of the variables affects a change in value of the other. When the y_i are given as functions of the x_l, i.e. $y_i = f_i(x_{l,\dots,}x_n)$: the total derivative dy_i/dx_l, given by the frame Q, i.e. the total sensitivity of y_i to x_l:

$$\frac{dy_i}{dx_l} = \sum_{k=1}^{n} \frac{\partial f_i}{\partial x_k} \frac{\partial x_k}{\partial x_l}. \quad (2)$$

If the metric sensitivities in the environmental model are understood and if Q specifies the x_k as functions of the y_i, i.e. $x_k = g_k\ (y_{l,\ \dots,}\ y_m)$, then the sensitivities of the design model to the environmental model are given by the chain rule:

$$\frac{dx_k}{dy_i} = \sum_{j=1}^{m} \frac{\partial g_k}{\partial y_j} \frac{\partial y_j}{\partial y_i}. \quad (3)$$

The sensitivities captured in the matrix Q, are suitable for analyzing the implications of relationships of the requirements on the design parameters. The Q frame used in this way can provide the facility for an autonomous decision process with the FMS to acknowledge to the AFDS which navigation sensor to use during flight. Thus, the QoS training problem becomes the management of transition of navigation aids of the SoS.

C Specification of Relational Transformation (Pilot Viewpoint)

For a student pilot the selection of the correct navigation system of the aircraft to use as the primary source of navigation is the training task. Within the training curriculum, procedures and rules (task sequences) for decisions made by a pilot during flight can be mapped to the waypoints and the transitions to the waypoints. The rules are based on pilot observables such as: quality of needle reading and stability of digital readouts, which relate to QoS. Generally, the pilot actions specified by the training curriculum can be expressed in structured English as an imperative sentence or command. A state machine is created to represent the control flow of what pilot actions and decision processes should be during flight. This behavioral model should include system states (navigation aid being used) and its transitions between states (QoS guard settings). Extended state variables and guards are used to represent both qualitative aspects (the state) and the quantitative aspects such as navigation QoS. This decision tree [18] can be used to present to the student the rules of the optimal strategy of when to switch navigation aid.

In ROSE the pilot behavior frame P is a duplicate of the Q frame, but concentrates on timing aspects of pilot decisions with respect to selecting the primary source of navigation aid while executing the route plan. The rules are mappings from the states of the objects, gathered from state machines attached to the blocks/classes of the sensors and FMS elements of Fig.2, of M and N into P and based on SME knowledge. Analysis of student behavior with respect to the correct time and distance between waypoints can be exploited to study pilot behavior as illustrated in Fig.5.

With error being a function of distance (expected error over time) for example, 1° error for every 10 miles, the optimal point to switch over to another navigation aid is determined when errors or the QoS are equal for two different navigation aids. Thus, decision constraints are part of the P frame to support the trainer assessment of student progress in the training pipeline.

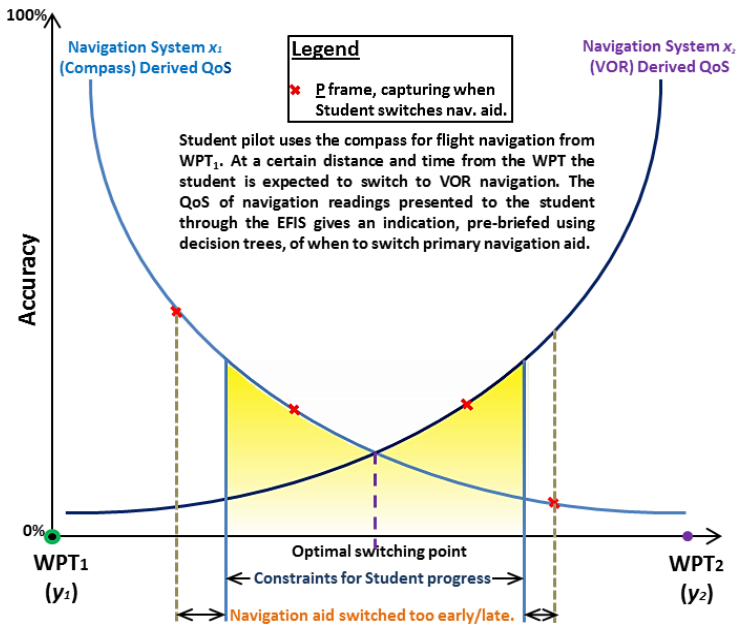


Figure 5 Q frame with P frame overlay slot - representing rules used to monitor student progress.

D Design Specification and Verification

During the time evolution of the flight the attributes of x_1 relative to x_2 at different states of the aircraft in relation to y_1 and y_2 will change and this change can be captured within the Q frame to provide a dynamic evolution of QoS as the pilot flies the aircraft through the waypoints to the landing site. In addition, a waypoint in M can be allocated to multiple navigation systems (represented by the 'tick' marks in Q), which will transform to a cluster of 'ticks' in N. The pilot directly interacts with the pilot controls, the MCDU, and the FMS system shown as elements in the metamodel of Fig.2; the behavior of the pilot captured by the P frame can be compared to the derived QoS of the Q frame and the associations with the M and N models to identify the deviation/actions taken by the pilot at a specific point in time and distance away from two waypoints, as indicated by the red crosses in Fig.5. Additional frameworks can be created for each element within the metamodel to monitor pilot and the FMS behavior from a navigational viewpoint by capturing the relationships and behavior between them. Analysis can be performed using the P frame to verify whether the pilot followed the correct rules and if the pilot operated the navigation system correctly. Thus, ROSE can be utilized to validate the training requirements.

Conclusion

The ROSE methodology has been used to develop framework LVC simulation and analysis for a navigational SoS for flight training. The metamodel and the framework address the traceability of relationships between elements of the FTSoS, which include requirements traceability, by factoring the design level architecture into more manageable domains and thus providing a structure to navigate through the requisite architectural artifacts. The complexity of the FTSoS has been reduced through the use of the metamodel and the factorization of the SoS architecture. The factorization also lends itself to a more rapid and efficient verification and validation of design and operational decisions.

The ROSE framework allows integration of models to permit more formal measurements of performance of the FTSoS and the behavior of the pilot, such as switching the navigation aid at the right time. The ROSE approach is used not only to specify attributes of the system but also the relationships that create constraints between the blocks/classes and their objects in the models, e.g. waypoints, and is used to specify frames as abstractions of relational (semantic) structures. The frameworks can be reused therefore for training, analysis, and pilot behavior in relation to planned.

The mathematical foundation for ROSE supports the rigorous development of structures for the design of systems and the assemblage of systems of systems. Frames and transformations prescribed by ROSE should be reusable to generate or modify architectural artifacts as necessary for further assessment studies; and if implemented in a modern modeling language should lend themselves to machine automation. Attention can then be focused on the management of the databases accessed to create the artifacts rather than management of the artifacts. The abstract approach employed should be applicable to general SoS problems once a domain model/metamodel is defined. Thus, the approach and results of this paper offer an early demonstration of a significant new methodology for SoS design and analysis.

References

[1] INCOSE,"INCOSE Systems Engineering Handbook V.3.2", ASIN: B003YSS8AY, 2010.

[2] A. Roy, "Secure Aircraft communications Addressing and Reporting System (ACARS)", Digital Avionics Systems, DASC, 20th Conf., Columbia, USA, Vol. 2, p-1-7, 14-18 October, 2001

[3] G. Jianming et. al., "Behavioural Modeling and EMC analysis for TACAN System", Microwave, Antenna, propagation, and EMC Technologies for Wireless Communications (MAPE), IEEE 4th International Symposium, Beijing, China, p. 576-579, 1-3 November, 2011

[4] D. Bodoh and F. Wieland, "Performance Experiments with the High Level Architecture and the Total Airport and Airspace Model (TAAM)", Parallel and Distributed Simulation Proceedings, Centre for Advanced Aviation System Development, VA, USA, , p. 31-39, 10-13 June, 2003

[5] C. Damon , "Bridging the HLA: Problems and Solutions", 6th IEEE International Workshop of Distributed Simulation and Real-Time Applications, Texas, USA, p. 33-42, 11-13 October, 2002.

[6] IEEE,"IEEE Standard for Distributed Interactive Simulation –Application Protocols (IEEE Std 1278.1a-1998)", ISBN: 0-7381-0174-5, 1998 (Current Version 6 August 2002).

[7] J.R.Noseworthy, "The Test and Training Enabling Architecture (TENA) Supporting the Decentralized Development of Distributed Applications and LVC Simulations",14th IEEE/ACM International Symposium of Distributed Simulation and Real-Time Applications, Virginia, USA, p 22-29, 17-20 October, 2010

[8] A Wymore, "Model-Based Systems Engineering," CRC Press ISBN: 978-0-8493-8012-9, 1993

[9] S. Friedenthal, R. Griego, and M. Sampson, "INCOSE Model Based Systems Engineering (MBSE) Initiative," in *INCOSE 2007 Symposium*, Keynote, San Diego, 24-29 June, 2007.

[10] C. Dickerson, "Towards a logical and scientific foundation for system concepts, principles, and terminology", Proc. IEEE SoSE '08, Monterey, California, USA, p177-182, 2-4 June, 2008.

[11] A. Tarski, "Contributions to the Theory of Models I,II,III", Nederl. Aka. Wetensch, Proc. Ser. A, Vol 57, pp. 275-581, 582-588, Vol 58, pp. 56-64, 1955.

[12] C. Dickerson, "Mathematical Foundation of Relational Oriented Systems Engineering,"Proc. IEEE SoSE '11, Albuquerque, New Mexico, USA, p. 197-202, 27-30 June, 2011.

[13] D. Kirk, M. Roper, & M. Wood, "Identifying and Addressing Problems in Object-Oriented Framework Reuse", Empirical Software Engineering, Vol.12, No.3, June 2007.

[14] C. Dickerson and D. Mavris, "Relational Oriented Systems Engineering (ROSE): Preliminary report", Proc. IEEE SoSE '11, Albuquerque, New Mexico, USA, p. 149-154, 27-30 June, 2011.

[15] United States Air Force (USAF), "A Training System for the 21st Century: JPATS and the T-6A", 2007.

[16] G.Klir, "Facets of Systems Science"; Plenum Press, new York, ISBN: 978-0-3064-3959-9, 1991.

[17] Y.Lin, "General Systems Theory: A Mathematical Approach"; Kluwer Academic/Plenum Publishers, New York, ISBN: 978-0-3064-5944-3, 1999.

[18] C. Dickerson and D. Mavris, "Architecture and Principles of Systems Engineering", CRC Press, ISBN: 978-1-4200-7253-2, 2009

Charles Dickerson became a Member of the IEEE in 2006. He received the Ph.D. in Mathematics from Purdue University, West Lafayette, IN, in 1980.

He is the Royal Academy of Engineering Chair of Systems Engineering at Loughborough University in the U.K. Before joining Loughborough University, he was a Technical Fellow for Systems Engineering at BAE Systems. He has also served as Aegis Systems Engineer for the U.S. Navy Ballistic Missile Defense Program and later as the Director of Architecture for the Chief Engineer of the U.S. Navy. His aerospace experience includes air vehicle survivability and design at the Lockheed Skunk Works and at Northrop Advanced Systems. He has also authored a graduate level textbook with the Georgia Institute of Technology on systems engineering and architecture, and authored a book on military system of systems architecture that received an international award.

Prof Dickerson is Chair of the INCOSE Architecture Working Group which collaborates with the IEEE System of Systems annual conferences in special sessions, and is Chair of the OMG Mathematical Formalisms Group.

Trevor Holden became a Member of IEEE in 2011. Trevor received the MSc. by Research on the development of an intelligent robotic manipulator from UCLAN University, Preston, Lancashire, UK in 2011.

He is currently pursuing a research council sponsored PhD project on open architecture for aviation SoS at Loughborough University in the UK. His research interests include systems engineering, electric propulsion techniques, and communication systems. He has a variety of experience in electrical / mechanical and software engineering from his 15years in industry.

Mr. Holden has published numerous papers on power-line communications, sensor fusion techniques and smart vehicle applications.