21^{st} ICCRTS

Implementing UPDM to Develop Command and Control Systems

Topics

Topic 7: Methodological Development, Experimentation, Analysis, Assessment and Metrics

Author
Rudolph Oosthuizen
(CSIR, South Africa; roosthuizen@csir.co.za)

Cobus Venter
(CSIR, South Africa; jpventer@csir.co.za)

Point of Contact
Rudolph Oosthuizen
(CSIR, South Africa; roosthuizen@csir.co.za)

Name of Organization: Council for Scientific and Industrial Research
PO Box 395
Pretoria
0001
info@csir.co.za

Implementing UPDM to Develop Command and Control Systems

Abstract

Systems engineering is an established approach to develop systems, including complex sociotechnical systems such as Command and Control (C2) systems. These systems often occur through the introduction of a new technology into an existing system. In systems engineering, modelling is applied to capture and represent the mental models of the systems' stakeholders during the concept development stage. These models, consisting of various views on the system structure and behaviour, can be used to derive requirements for system development. The views of the models must represent the mental model of the originator as well as ensure that the interpreters develop the same understanding. An architectural frameworks, such as the Ministry of Defence Architecture Framework (MoDAF) and Department of Defence Architecture Framework (DoDAF), supports Model Based Systems Engineering (MBSE). The Unified Profile for DODAF and MoDAF (UPDM) supports development of the model to ensure transportability to other participants in the development process. This paper proposes a model development process within UPDM for a C2 system during concept development.

1 Introduction

Systems engineering aims to solve problems by bringing systems into being through systems thinking for understanding the part in the context of the whole (Hitchins 2008, Stensson 2010). The basic systems engineering process distils the stakeholder needs to develop concepts, define requirements, and design and development solutions through interdisciplinary activities to ensure stakeholders' needs are met in a cost-effective and timely manner (Haskins 2010, Oliver et al. 2009). However, today's problems tend to be complex and ill-defined with a wider impact than the system of interest. Developing useful solution systems are hampered by complexity, communication, and understanding (Holt & Perry 2008).

Capturing mental models of all the stakeholders in suitable models and communicating it amongst team members will improve successful problem analysis, solution concept development and solution implementation (De Weck et al. 2011, Sterman 1994). A model is an explicit and incomplete representation or abstraction of reality to aid its description and understanding (Ramos et al. 2012, Kant 1950). Therefore, all models have a level of inaccuracy and the success of a model is determined by its reliability, completeness, accuracy, power to convince, ease of use, compatibility, and extendibility (Hull et al. 2005, Buede 2000, Ramos et al. 2012, Polack et al. 2008). In systems engineering, models are constructed through schematics and network diagrams of the system.

Modelling consists of an iterative process to develop, use and update models to obtain insight into a complex system's behaviour (Maria 1997, Harrison et al. 2007). Different viewpoints of the system may be structured and organised into an architecture. Modelling can assist design by understanding the relationship between the system as a whole and its parts to derive emergent properties (Buede 2000, Polack et al. 2008, Ramos et al. 2012, Maria 1997, Haskins 2010, Ramos et al. 2011). Model-Based Systems Engineering (MBSE) employs modelling languages, such as Systems Modelling Language (SysML) or Unified Modelling Language (UML). The language utilises a graphical notation, enhanced with relevant parameters, attributes and qualifying information. These languages are used to model complex systems and their architectures through suitable diagrams that capture, analyse, and specify behaviour, structure, requirements, relationships, and capabilities of a system through various consistent viewpoints.

Models also tend to be more useful than text-based systems engineering documents to develop system concepts and requirements as they support experimentation with knowledge on the problem and develop an understanding of the implications of different solutions (Estefan 2007, Haskins 2010).

The UPDM guides the modelling of a system and its context. DoDAF and MoDAF support modelling of a complex enterprise architecture by organizing it into logical, standardized, complementary and consistent viewpoints. The architecture frameworks enable sharing, integrating and re-using of consistent information in the model and its views (Fowler 2004, Hause 2011, Hause 2014, Hayden & Jeffries 2012).

This paper reports on a process implemented for conceptual modelling of C2 systems by first discussing systems engineering and modelling literature before relating it to practical experience. The UPDM will also be discussed to highlight the use of MoDAF and supporting modelling languages. This will lead to the presentation and discussion of a modelling process for conceptual modelling of military command and control type systems. Many other architecture frameworks and modelling languages exist but are outside the scope of this paper.

2 Modelling in System Engineering

2.1 Systems Engineering

The basic objective of systems engineering is to solve problems by bringing systems into being (Stensson 2010). The systems approach, through systems thinking, aims to understand the part in the context of the whole, while interacting with and adapting to the environment. Hitchins (2008) also defines systems engineering as “the art and science of creating whole solutions to complex problems”.

Systems engineering consists of interdisciplinary activities required to support the design and development of a useful system that creatively exploit energy, materials and information within organized systems of humans, machines and the environment. The systems engineering process has to ensure that the stakeholders’ needs are met in a cost-effective and timely manner. The basic systems engineering process distils the needs of stakeholders along with the characteristics of the environment to develop concepts and define requirements (Haskins 2010, Oliver et al. 2009).

The input to the systems engineering process is a need from a customer or stakeholder which must be analysed to discover requirements for defining the purpose, objective and high-level functions of the solution system (Buede 2000, White 2009, Ramos et al. 2010). The operational context and expected scenarios are defined from the initial high-level requirements to derive the required roles, tasks and functions of the system. The problem is analysed through interacting with the stakeholders to ensure that the goals and objectives of the design are well understood (Stanton et al. 2012, Meadows 2008). As noted by Simon (1996), “… solving a problem simply means representing it so as to make the solution transparent …”.

The demands on systems engineering to produce systems capable of effective operation within complex environments are ever-increasing. Holt & Perry (2008) list the three evils of systems engineering as complexity, communication, and understanding. The level of system complexity depends on the number of system elements as well as their interaction. An improper understanding of the problem and user needs leads to inaccurate, contradictory, incomplete and ambiguous requirements. This results in the improper application of systems engineering. Communication problems between engineers, the development team and the stakeholders lead to different interpretations of the meaning of requirements and associated models. Modelling has the potential to combat all three these evils.

Various forms of models and constructs can be used to capture and represent the information and knowledge on the problem. However, problems faced today can be ill-defined, with a wider impact, which may result in serious socioeconomic complexities and other environmental effects. Making absolute sense of a complex system is almost impossible; however, a suitable mental model to absorb and interpret information is important. The mental models of all the stakeholders must be captured in models for successful problem analysis (De Weck et al. 2011, Sterman 1994).

In this paper, the focus is on the concept stage, which assesses opportunities, explores concepts, identifies stakeholder requirements, and proposes concept solutions. In the development stage describing the requirements is refined, and the solution is created, verified and validated.

2.2 Modelling

A model is defined as an explicit and incomplete representation or idealized abstraction of reality, or a selected part thereof, to aid its description and understanding (Ramos et al. 2012, Kant 1950). A model is generated for a purpose and describes the essential nature, characteristics or pattern of something (including a process or system) without being the thing itself. A mental model can be viewed as humans' interpretation of something via their senses (Oliver et al. 1997, Haskins 2010, Oliver et al. 2009, Stanton et al. 2012).

Modelling is an iterative process to develop, use and update models using a standard, rigorous, structured methodology and common notation. The act of modelling and the resulting models themselves support insight into a system's behaviour as a basis for making decisions (Maria 1997, Harrison et al. 2007).Performing analysis and modelling constitutes design in principle (Buede 2000, Haskins 2010). The success of a model is determined by its reliability, consistency, completeness, accuracy, power to convince, ease of use, compatibility, and extendibility (Hull et al. 2005, Buede 2000, Ramos et al. 2012, Polack et al. 2008).

For systems engineering the purpose of modelling is to understand and describe the problem and its perceived solution sufficiently to gain insight into complex systems and to support answering questions on the system. The model of the system describes how the system will change states due to external inputs. Complex systems must initially be abstracted and modelled at a high level in order to synthesise a suitable and acceptable functional architecture. Architecting requires high level technical knowledge and creativity to establish a framework of models for system development, and assists in trade-offs and design decisions (Maier & Rechtin 2000, Ramos et al 2010, Buede 2000, Hitchins 2008).

Conceptual models describe and represent selected aspects of the structure, behaviour, operation and characteristics associated with a system as well as selected interactions with its operational environment, enabling systems and interfacing with other systems. Models are constructed through schematics and network diagrams of the system to indicate how information, material and consumables flow through the system (Haskins 2010, Maria 1997). These models represent the system’s functional design, and are used to communicate ideas (shared vision) to other stakeholders.

The model should be a sufficient approximation of the system it represents, incorporating most of its salient features. The model must achieve a balance between realism and simplicity, to enable understanding and simulation (Maria 1997). These can be used to confirm anticipated system behaviours and to justify requirements. Models are utilized to experiment with knowledge on the problem and to develop an understanding of the implications of different solutions. Simulations may address system functions or the detailed structure through identified scenarios. These are required to support improved system development decisions through clarifying requirements (Buede 2000, Polack et al. 2008, Ramos et al. 2012).

Modelling for systems engineering has evolved from functional flow block diagrams, structure analysis and design techniques to an object oriented representation. Modelling of a complex socio-technical system is difficult as it has to present the structure and behaviour of human work in the system. Behaviour is caused by dynamic interaction between the humans, operators, system elements and the environment. However, modelling of system interfaces at multiple levels of abstraction may aid in understanding complex systems (Bahill & Szidarovszky 2009, Piaszczyk 2011).

The modeller should ensure that the system stakeholders are involved from the start, and should apply methods that support mutual understanding through engagement and debate (Yearworth & Cornell 2012). Subject matter experts provide operational experience and domain knowledge to identify requirements as well as to assist in design evaluations and decisions. Scenarios are used to assess the effects and goals of a cognitive in context. They can also be used during interviews to elicit knowledge from operational stakeholders (Elm 2008, Ockerman et al. 2005). Functional analysis can only commence once all of the system requirements (functional, performance, specifications and standards), architectural concepts, concept of operations and constraints have been fully identified. The typical outputs of functional analysis include the following (Haskins 2010):

1. Behaviour Diagrams. These describe systems behaviour using constructs of time sequences, concurrencies, conditions, synchronisation points, state information and performance.
2. Context Diagrams. This is the top-level diagram of a data flow that portrays all inputs and outputs of a system without decomposition.
3. Control and Data-Flow Diagrams. These are box diagrams, flowcharts, input-process-output charts and state transition diagrams that provide sequences in which operations may be performed by the system. They are linked to data flows between the functions.
4. Entity Relationship Diagrams. The logical relationships between functions or architectural elements are depicted in these diagrams.
5. Functional-flow Block Diagrams. The functional flow block diagrams provide insight into the flow between the system functions.

Models should be more useful than text-based systems engineering documents to develop system concepts and requirements as common notations depict many system characteristics and attributes in a consistent way (Buede 2000, Polack et al. 2008, Ramos et al. 2012). MBSE focuses on the application of information rich models to complement traditional systems engineering methodologies. MBSE employs a process to develop and increase the detail in models using a concurrent and incremental process to enable understanding between stakeholders (Estefan 2007, Haskins 2010).

MBSE utilizes modelling notations such as SysML and UML to develop consistent system views. These can be used to model complex systems through diagrams of system structure, parameters, attributes, requirements, behaviour and relationships. The structural diagrams represent the parts of a situation with their logical relationships. The behavioural diagrams represent the parts of a situation and their causal interactions. The requirement views specify desired structural and behavioural properties as derived from stakeholder needs. Parametric views provide the critical engineering parameters of the system for evaluating performance, reliability and physical characteristics (Friedenthal et al. 2012, Hause 2014).

The various output models support simulating and analysing the behaviour of the system to characterize the functional architecture. The different views of the system may be organized in an architecture, to structure the model. A systems’ architecture presents the fundamental organization of the system components as well as their relationships to each other and to the environment. These present the principles guiding system design and evolution. A typical architecture framework provides at least operational, logical and physical viewpoints (Kossiakoff & Sweet 2011, Ryan et al. 2014).

Different views on the system model are used to analyse the system at different levels. The key structural elements are identified to develop behavioural models such as Sequence Diagrams (interaction between elements with interdependencies, control and timing aspects), State Machine Diagrams (Trigger, Guard and Action nomenclature for element lifecycle), Activity Diagrams (flow of objects and control through decision points with swim-lanes), and Use Case Diagrams (interact with stakeholders to derive system requirements). Simulation models are developed once the requirements are understood and potential architectures identified to confirm and verify the understanding of the problem and requirements (Ryan et al. 2014, Hause 2010)

2.3 Quality in Modelling

By its very nature, modelling creates an incomplete and approximate replica of the problem or system. A complete replica would be reality itself. Modelling of systems is based on abstracting the characteristics of a domain of interest to the modeller and ignoring the others (Hause 2011). In computer science, abstraction is the mechanism and practice of factoring out details so that one can focus on a few key concepts at a time (Illingworth 1991). Depending on the level of abstraction, and the focus of concern, there will be different viewpoints with its own unique elements that can be applied to the system.

The main aspects that determine the quality in models are the ability to communicate concepts, demonstrating traceability of requirements, implementation of a standardized process and the quality assurance actives built into the process (Hause 2011). Validating the model requires collecting proper data to ensure that reality is represented sufficiently accurate. Decisions based on models are only as good as their validation (Lucas & McGunnigle 2003, Davis 2004).

Especially in the development of new systems and of system-of-system concepts, a model is not a representation of reality but rather an idea. Such a model can therefore not be validated against the reality to determine the correctness. In an environment where more than one possible outcome could be correct this could lead to many arguments regarding the correct model to represent the perceived future or future system. In such environment the concept of Requisite Models can be implemented (Phillips 1984). Making absolute sense of a complex system is practically impossible, but a suitable mental model to absorb and interpret information is important (Sterman 1994).

2.4 Psychology of Systems Modelling

A requisite model is defined as a model whose form and content are sufficient to solve a particular problem (Phillips 1984). A requisite model is therefore a model where the role players agree that the model adequately represent their own mental models. Especially for very complex environments the concept of requisite models can assist in deciding when a model is adequate. Requisite models simplify the representation of a future reality in three ways (Phillips 1984):

1. Elements of the reality that are not expected to contribute significantly to solving the problem are omitted from the model.
2. Complex relationships among elements of the reality are approximated in the model.
3. Distinctions in either form or content at the level of the reality may be blurred in the model.

As stated above, models are used to communicate needs and ideas about a problem to be solved and possible solutions between stakeholders. Shannon (2001) noted that the fundamental problem of communication is reproducing the message from a sender either exactly or approximately at the receiver. His basic model of communication included phases (elements) of encoding, transmission and decoding as well as the effect of noise on the transmission channel. Berlo (1960) also identified the context and factors affecting the coding and decoding of a message as, communication skills, attitudes, knowledge, culture and social system. Communication is considered effective if it receives the desired result, response or reaction.

The modeller, often a systems engineer, encodes his mental model (understanding) of the problem, context and solution system. The stakeholders must decode and interpret the model to contribute and validation. However, the system engineer and the stakeholder may not have the same skill, attitudes, knowledge, culture and social system. One controllable is the content and framework of the model. Even if the stakeholders may not be fluent in the underlying modelling notation (SysML or UML), a proper selection of viewpoints on the model may enhance correct interpretation. The level of abstraction needs to suit the context of the specific stakeholder.

3 System Architecture

3.1 Architecture Frameworks

An Architectural Framework is a guide of how to organize and present architectural models of the system. Because “architecting” of enterprises is such an all-encompassing discipline, it can result in very complex models. MBSE within an Architectural Framework utilize different views to denote different stakeholder interests, and to provide a means for evaluation and report generation as well as to simplify maintenance of the models (Hause 2010).

Some architectural Frameworks, such as DoDAF and MoDAF, were developed to support development and acquisition of military systems. These architectural frameworks organize the enterprise architecture of a complex organization, capability or system into logical, standardised, complementary, integrated and consistent views of a model to improve its planning, organization, procurement and management. The MoDAF is an enabler for managing complexity through a specification of how to represent an integrated model of an enterprise. MoDAF provides the ability to share, integrate, search and re-use architectural information across various layers. The MoDAF viewpoints provide consistent perspectives of the system architecture and they enable a user to articulate and analyse issues and requirements to specify, design and validate system solutions. The same information may be represented in more than one view, and there may be important relationships between the information in different views that should also be captured (Hayden & Jeffries 2012, Hause 2014).

3.2 Unified Profile for DoDAF and MODAF

UPDM guides the use of modelling notation for MoDAF and DoDAF. UPDM, SySML and UML originated in different “domains”, but under guidance of the Object Management Group (OMG). UPDM and SySML implementations are extensions from UML (Fowler 2004, Hause 2011).

SysML is suitable for systems engineering and include concepts such as interface and flow specifications, system concepts, parametric constructs, and integrated requirements. The goal is to provide a common modelling language for systems engineering to analyse, specify, design and verify complex systems. The system’s behaviour is modelled with use case, activity, sequence and state machine diagrams. The use-case diagram provides a high-level description of the functional usage of the system. The flow of data and control between activities are captured in an activity diagram. A sequence diagram represents the behaviour and interactions between collaborating parts of a system. Actions and state transitions of a system in response to events are modelled in a state machine diagram. Requirements diagrams ensure that hierarchies and the derivation, satisfaction, verification and refinement relationships are clear. System parameter constraints and other parameters such as performance, reliability and physical properties are captured in the parametric diagram (Hause 2014).

Block diagrams model the structure of the system. Block definition diagrams describe the system hierarchy and classification. Internal block diagrams describe the internal structure with parts, ports, and connectors, of a system. The blocks in the block diagrams can represent any level of the system hierarchy to describe a system as a collection of parts and connections between them that enable communication and other forms of interaction. Ports provide access to the internal structure of a block for use when the object is used within the context of a larger structure (Hause 2014). SysML enhance systems quality, improve the ability to exchange systems engineering information amongst tools and help bridge the semantic gap between systems, software and other engineering disciplines (Hause 2011).

UPDM aims to enhance the quality, productivity, and effectiveness of enterprise and system of systems architecture modelling through model reuse and maintainability. Standardisation improves tool interoperability and communications between stakeholders. It provides a standards-based “model of models” approach, also suitable for to non-technical users and managers to model complex systems of systems (Hause 2010, Hause 2014).

4 System Concept Development Process

4.1 Concept Development

A typical systems engineering process for developing a solution concept for the design of a system from user requirement is shown in Figure 1 (DOD 2001). The user requirements include the mission, measures of effectiveness, environmental conditions and other constraints. These requirements are analysed to determine the mission environment, functional requirements, design constraints etc. This is followed by an in-depth functional analysis to decompose the operational activities into lower level functions.

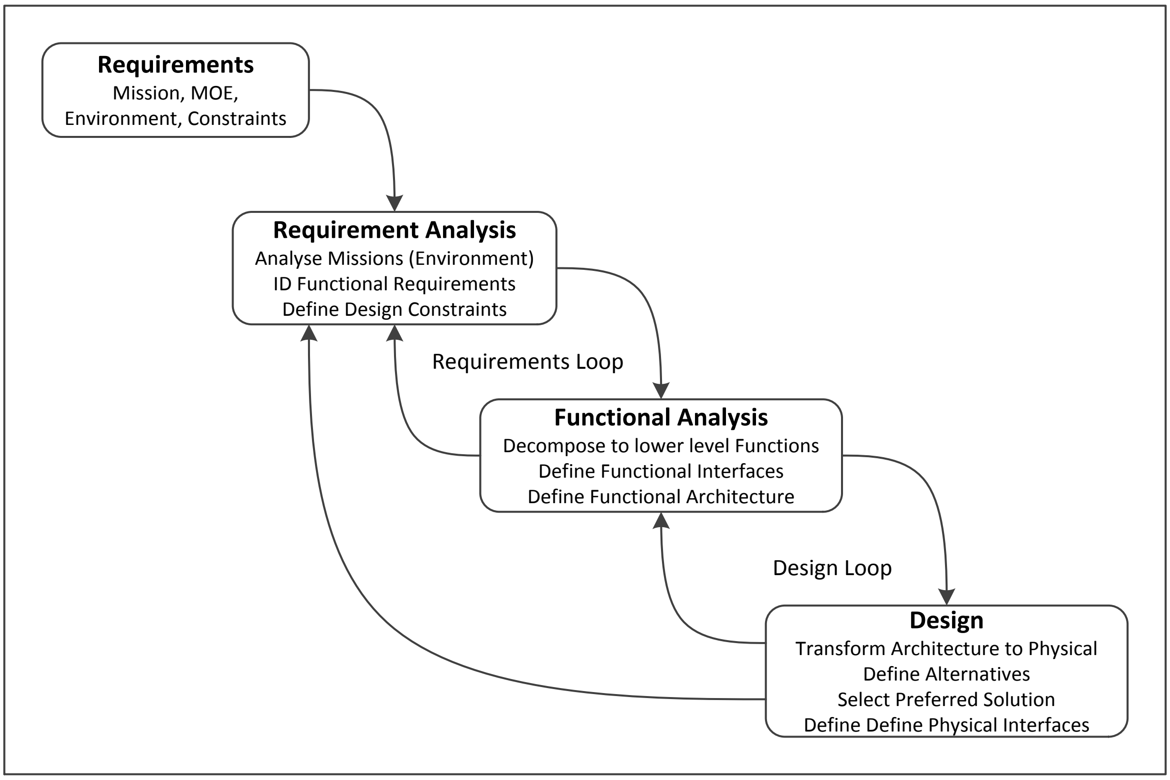

Figure 1: Systems Engineering Process

The aim is to define the interfaces between the functions and to develop a functional architecture. The functional analysis may identify gaps in the requirements which necessitate going back to requirements analysis. This is referred to in the process as the requirements loop. The next step is to transform the functional architecture to a physical architecture of the system by allocating functions to physical elements. This may result in alternative implementations of the system functions. Analysis of the alternative options enables selection of a preferred solution for the system realization. Again, this step may identify discrepancies and deficiencies in the functional architecture to feed back to the functional analysis, resulting in the design loop.

4.2 Modelling for Concept Development

The concept development process from Figure 1 requires a modelling approach that is able to address the complexities expected in a military (e.g. command and control) system. As discussed in the previous section, one such framework is MoDAF. The relationship between the MoDAF views and the systems engineering process from Figure 1 are provided in Figure 2.

MoDAF defines a standard set of “Views” or model categories. The viewpoints provide consistent perspectives of an architecture to enable a user to articulate and analyse system issues to specify, design and validate solutions across a wide range of activities in the system. The same information on the model may be represented in different Views to help identify and capture important relationships. The MoDAF reference model ensures consistency between Views.

The MoDAF views support the requirements analyses and functional analysis phases. An Operational Viewpoint (OV) defines the logical and operational aspects of the architecture. The OVs describe the key behavioural and information aspects of capabilities in context of an operation or scenario. The OVs support development of user requirements, capturing future concepts in a logical architecture and supporting the operational planning process. The analysis of the operational domain can be seen as part of understanding the problem definition for which a solution (system) is to be developed.

The OV-2 (Operational Node Connectivity Description) shows internal andr external Operational Nodes with needlines between them to indicate a need to exchange information and other resources. This is used to identify a logical network of information flows or information exchange requirements for a C2 system. An Operational Node presents an element of the operational system, in the form of operational roles (human) or functional grouping, that produces, consumes, or processes information.

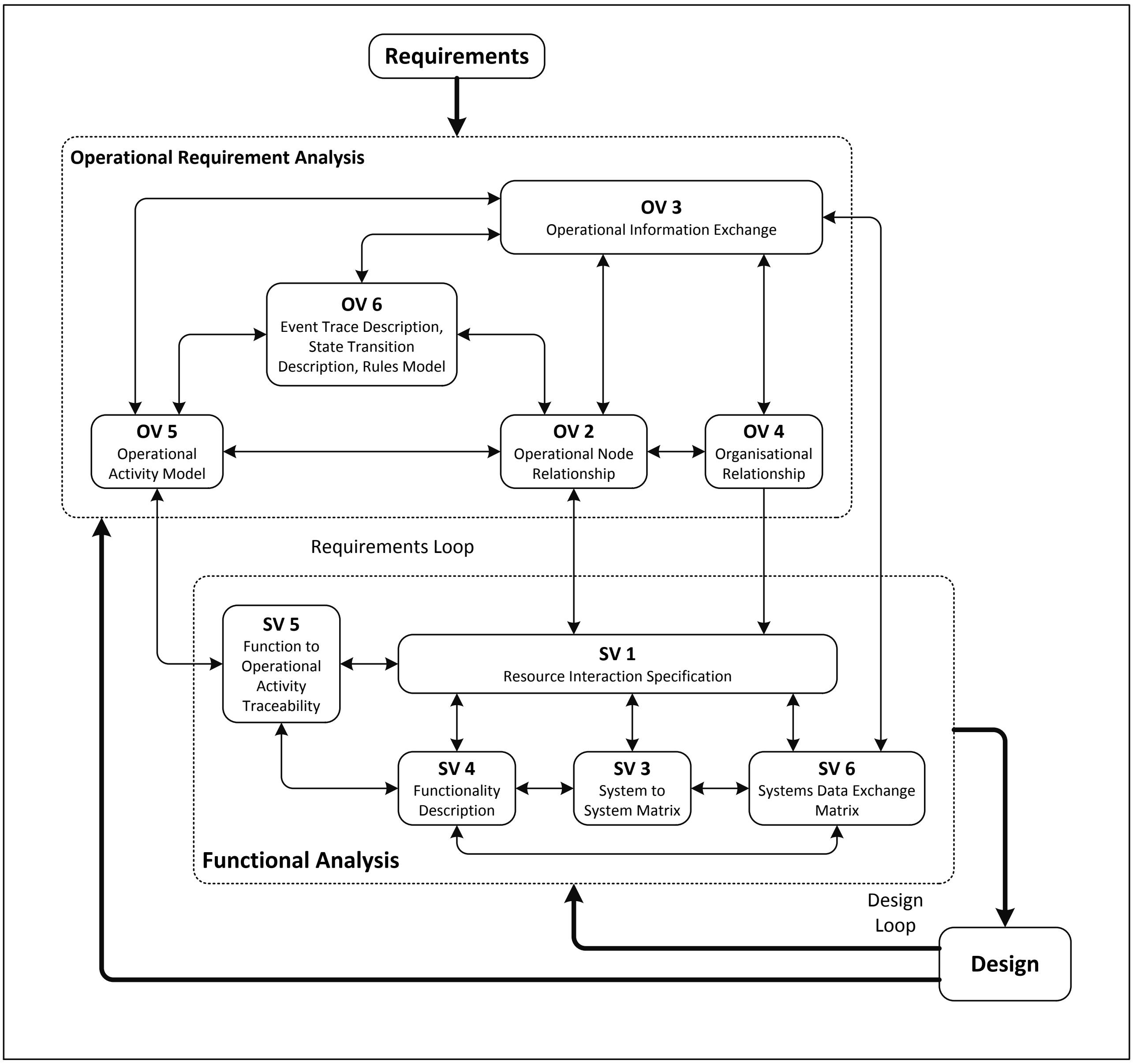

Figure 2: MoDAF in the Systems Engineering Process

The OV-5 (Operational Activity Model) describes the operations (activities, inputs and outputs) that are conducted in the course of achieving a mission as well as the operational flows between activities. The OV-5 helps to delineate lines of responsibility for activities. It may also uncover redundancy and identify operational activities and their interactions that need further analysis. The OV-2 indicates the nodes and the interactions required for the operational activities of OV-5.

The OV-3 (Operational Information Exchange Matrix) captures information elements and their most important attributes that are exchanged between nodes and activities. The OV-4 (Organisational Relationships Chart) provides and clarifies the command structure or relationships among key human roles players or organizations in the system. The required human roles and skills to perform the operational activities or business processes are defined and described in the operational architecture.

The OV-6 (Operational Activity Sequence & Timing Description) diagrams add additional views on the dynamic operational behaviour of the system. These diagrams address the timing and sequencing of events from the activities of OV-5. The two main types are dynamic behaviour and timing performance diagrams are Operational State Transition Description (OV-6b) and Operational Event-Trace Description (OV-6c). The OV-6b is a graphical state-chart diagram that describes the operational node or activity state changes. The OV-6c provides a time-ordered information exchanges between participating operational nodes during a scenario (Hayden & Jeffries, 2012).

A System Viewpoint (SV) describes resources that realize capability through resource functions and interactions between resources. The SVs can be used to specify solutions to requirements specified in the

OVs, or simply to provide more detail to the logical OV architecture. The SVs can also provide detailed system interface models (Hause 2014, Hayden & Jeffries 2012). The ‘Design Loop’ requires some form of physical design and functional allocation.

The SV-1 (Systems Interface Description) depicts systems and identifies the interfaces between them. SV-1 links together the OV and SV by depicting which systems are resident at which Nodes. Systems Nodes may in reality also be Operational Nodes as they typically encapsulate systems, organisations and personnel as the performers of an operational activity. The SV-1 records the system characteristics of the operational nodes as well as how the OV needlines are realized as system interfaces. An interface represents a simplified, abstract representation of one or more interaction paths between systems.

The SV-4 (Systems Functionality Description) provides system functional hierarchies and functional flows as well as the data flows between the functions. The system functions can also include Human Machine Interface functions to capture the human role in the system. The SV-4 has to ensure that the functional decomposition reaches an appropriate level of detail and the functional connectivity is complete.

The SV-4 system functions relate to the OV-5 operational activities, although there may not exist a one-to-one mapping. Therefore a SV-5 (Systems Function Traceability Matrix) is required. The SV-5 provides the mapping and relationships between operational activities and the system functions applicable to an architecture. The SV-3 (Resource Interaction Matrix) provides a summary of the resource interfaces defined in the SV-1.

The SV-6 (Systems Data Exchange Matrix) specifies the characteristics of the automated system data exchanged between systems. This excludes non-automated information exchanges and verbal orders that are captured in the OVs. The data exchanges address interactions between systems and system functions. The SV-6 covers how the system data exchange is implemented in terms of periodicity, timeliness, throughput, size, information assurance, and security characteristics of the exchange.

4.3 Concept Development Process

The high level process for developing a solution concept is shown in Figure 3. The three main products from the conceptual modelling process are the User Requirements, Operational Concept Description (OCD) and a Solution Description. The user requirements define the force elements, doctrine and C2 structure to implement a capability in the form of a higher order user system. This should relate to the MoDAF Capability views in the form of a Required Operational Capability (ROC).

The OCD for the system, as part of the operational requirement analysis, focuses on the OVs. The available force elements, C2 structure and doctrine is analysed to develop the OVs. The OCD use the operational nodes, activities and the associated flow of information to describe the operational use of the system. This is demonstrated with the use of different implementation scenarios to derive environmental interaction and constraints. The system behavioural views (OV-6b and OV-6c) helps the stakeholders to understand the system behaviour.

The Option Description focuses on the SVs as part of the functional analysis. The Option Description identifies logical system nodes (elements) that will perform the system functions to derive the solution architecture design. The system elements relates to the operational nodes and the functions to the operational activities, although a simple one-to-one mapping is neither essential nor typical. The information elements used by the operational nodes are analysed to derive the actual data distributed between system nodes. The Option Description utilizes system elements, system functions and associated flow of data to describe the solution option for the C2 system of interest.

An advantage for implementing the process described above in a UPDM capable modelling tool (software) through a MBSE process is the inherent capability to capture the traceability from the system elements and functions back to the user requirements captured during the operational analysis.

A difficulty with a MBSE process is to manage the complexity to model the interaction of the system with the users and other stakeholders. Developing simple models of complex systems is not easy. Simple diagrams may omit essential information about the system operations and functions. Maintaining the required balance is important.

The Option Description product is the source for defining the system functional requirements. The OCD is usually used alongside the system functional requirement to provide context on the utilisation of the proposed system. These serve as input to the detailed design of the solution system. Throughout this process continuous interaction with the stakeholders is required for validating models (views) to ensure that all the mental models are correctly captured and represented.

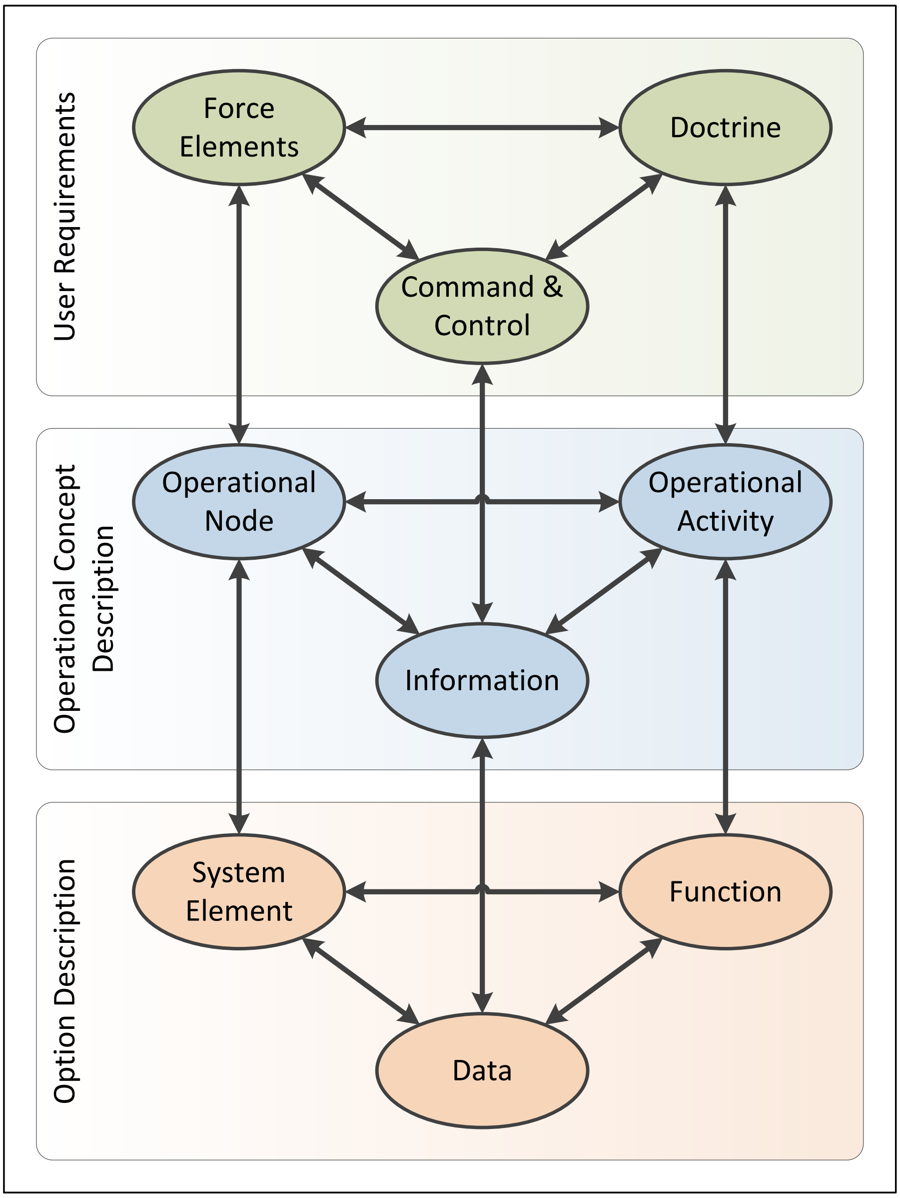

Figure 3: Relationship between Operational Concept Description and Option Description

5 Conclusion

As discussed in this paper, systems engineering is seen as an established approach to define the problem (discover requirements) and define solutions in order to develop systems. Modelling is applied to capture and represent the mental models of the systems' stakeholders. These models, consisting of various integrated and interrelated viewpoints on the system structure and behaviour, can be used to derive requirements for system concepts and actual system development.

The UPDM enables MBSE by ensuring commonality across different viewpoints, processes and tools as well as transportability to other participants in the development process. Use of stakeholders' mental models tied to systems engineering should provide a better approach to systems development.

This paper provides and discusses a modelling process for conceptual modelling of military command and control systems with UPDM. Despite some difficulties experienced with modelling of complex systems the process outputs should enable the systems engineering process to acquire an effective solution system.

6 References

Bahill, A.T. and Szidarovszky, F., 2009. Comparison of dynamic system modeling methods. Systems Engineering, 12(3), pp.183-200.

De Weck, O.L., Roos, D, and Magee, C.L., 2011. Engineering systems: Meeting human needs in a complex technological world. MIT Press.

Berlo, D. K. 1960. The process of communication. New York: Holt, Rinehart & Winston.

Buede, D.M. 2000. Engineering Design of Systems - Models and Methods. John Wiley & Sons.

Davis, P.K. 2004. Space Modeling and Simulation: Roles and Applications throughout the System Life Cycle. L. B. Rainey (Ed.). AIAA.

DOD, D., 2001. Systems Engineering Fundamentals. DoD, Department of Defense (USA).

Elm, W.C., Gualtieri, J.W., McKenna, B.P., Tittle, J.S., Peffer, J.E, Szymczak, S.S., Grossman, J.B. 2008. Integrating Cognitive Systems Engineering Throughout the Systems Engineering Process. Journal of Cognitive Engineering and Decision Making, vol. 2, no. 3, pp. 249–273.

Estefan, J. 2007. Survey of Model-Based Systems Engineering (MBSE) Methodologies. Jet Propulsion Laboratory, California Institute of Technology, Pasedena, CA.

Fowler, M., 2004. UML distilled: a brief guide to the standard object modeling language. Addison-Wesley Professional.

Friedenthal, S., Moore, A. and Steiner, R., 2012. Integrating SysML into a systems development environment a practical guide to SysML. Elsevier, pp.523-556.

Harrison, J.R., Lin, Z., Carroll, G.R., & Carley, K.M. 2007. Simulation Modeling in Organizational and Management Research. Academy of Management Review, vol. 32, no. 4, pp. 1229-1245.

Haskins C. (ed.), 2010. INCOSE Systems Engineering Handbook. Version 3.2, INCOSE-TP-2003-002-03.2.

Hause, M., 2010. The Unified Profile for DoDAF/MODAF (UPDM) enabling systems of systems on many levels. In Systems Conference, 2010 4th Annual IEEE (pp. 426-431). IEEE.

Hause, M., 2011. “Are we there yet?” Assessing Quality in Model Based Systems Engineering. In INCOSE International Symposium (Vol. 21, No. 1, pp. 510-522).

Hause, M. C., 2014. SOS for SoS: A new paradigm for system of systems modeling. In Aerospace Conference, 2014 IEEE (pp. 1-12). IEEE.

Hayden, J. L., and Jeffries, A. 2012. On Using SysML, DoDAF 2.0 and UPDM to Model the Architecture for the NOAA’s Joint Polar Satellite System (JPSS) Ground System (GS). Copyright 2012 by Jeffries Technology Solutions, Inc. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission

Hitchins, D. K. 2008. Systems Engineering: A 21st Century Systems Methodology. Wiley. com.

Holt, J. and Perry, S., 2008. SysML for systems engineering (Vol. 7). IET.

Hull, E., Jackson, K., & Dick, J. 2005. Requirements Engineering. Vol. 3. London: Springer.

Kant, I. 1950. Prolegomena to Any Future Metaphysics,(first published in 1783) Lewis White Beck, editor and translator.

Kossiakoff, A., Sweet, W.N., Seymour, S. and Biemer, S.M., 2011. Systems engineering principles and practice (Vol. 83). John Wiley & Sons.

Lucas, T.W., & McGunnigle, J.E. 2003. When is Model Complexity too Much? Illustrating the Benefits of Simple Models with Hughes' Salvo Equations. Naval Research Logistics, vol. 50, no. 3, pp. 197-217.

Maier, M.W., & Rechtin, E. 2000. The Art of Systems Architecting (Vol. 2), Boca Raton: CRC press.

Maria, A., 1997. Introduction to Modeling and Simulation. In Proceedings of the 29th conference on Winter simulation. IEEE Computer Society, 7-13.

Meadows, D., 2008. Thinking in Systems: A Primer. Chelsea Green Publishing.

Object Management Group (OMG), 2007a. Unified Modeling Language: Superstructure version 2.1.1 with change bars ptc/2007-02-03. [online] Available from: http://www.omg.org [Accessed September 2007].

Ockerman, J., McKneely, J.A.B., & Koterba, N. 2005. Hybrid Approach to Cognitive Engineering: Supporting Development of a Revolutionary Warfighter-Centered Command and Control System. In Proceedings of the 10th International Command and Control Research Technology Symposium.

Oliver D.W., Kelliher T.P., & Keegan Jr. J.G. 1997, Engineering Complex Systems with Models and Objects. McGraw-Hill.

Phillips, L.D., 1984. A theory of requisite decision models. Acta psychologica, 56(1), pp.29-48.

Piaszczyk, C., 2011. Model based systems engineering with Department of Defense Architectural Framework. Systems Engineering, 14(3), pp.305-326.

Polack, F.A., Hoverd, T., Sampson, A.T., Stepney, S., & Timmis, J. 2008. Complex Systems Models: Engineering Simulations. In Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems (pp. 482-489). MIT Press.

Ramos, A.L., Ferreira, J.V., & Barceló, J. 2012. Model-Based Systems Engineering: An Emerging Approach for Modern Systems. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, vol. 42, no. 1, pp. 101-111.

Ryan, A. 2007. Emergence is Coupled to Scope, not Level. Complexity, vol. 13, no. 2, pp. 67 – 77.

Ryan, J., Sarkani, S., & Mazzuchi, T. (2014). Leveraging variability modeling techniques for architecture trade studies and analysis. Systems Engineering, 17(1), 10-25.

Shannon, C.E. 2001. A mathematical theory of communication. ACM SIGMOBILE Mobile Computing and Communications Review 5, no. 1: 3-55.

Simon, H. A. 1996. The Sciences of the Artificial. (3rd ed.), MIT Press, Cambridge, MA.

Stanton, N.A., Baber, C., & Harris, D. 2012. Modelling Command and Control: Event Analysis of Systemic Teamwork. Ashgate Publishing.

Stensson, P. 2010. Thoughts about the Consequences of Inappropriate Application of Systems Engineering. In Proceedings of the 7th European Systems Engineering Conference, Stockholm, INCOSE.

Sterman, J.D, 1994. Learning in and about Complex Systems. System Dynamics Review, vol. 10, no. 2-3. pp. 291-330.

White Jr, K.P. 2009. Systems Design. In: A.P. Sage & W.B Rouse eds. Handbook of Systems Engineering and Management, Second Edition, John Wiley & Sons, 507 – 533.

Yearworth, M., & Cornell, S. 2012. Contested Modelling. In Proceedings of the 56th Annual Meeting of the ISSS-2012, San Jose, CA, USA.