PAPER 12

A Practical Approach For Modelling Submarine Subsystem Architecture In SysML

Paul Pearce
ASC Pty. Ltd.
Osborne, South Australia
paul.pearce@asc.com.au

Sanford Friedenthal
SAF Consulting
Fairfax, Virginia USA
safriedenthal@gmail.com

***Abstract*—This paper outlines a practical design approach for submarine subsystems using a model-based systems engineering (MBSE) methodology. At the core of this approach is a system model that provides a precise, consistent, traceable and integrated description of the submarine subsystem architecture. The system model is used to support the flow-down of requirements from the submarine missions to system, to subsystem and to component specifications. The system model is defined in the OMG Systems Modeling Language (OMG SysML™) and implemented in an established commercial SysML graphical modelling tool. The MBSE approach is based on a combination of industry best practice and the practical experiences of deploying MBSE within a submarine design team. This approach is being used by the design team at both whole-of-submarine and subsystem levels of design. This paper will elaborate on the application of MBSE to submarine subsystem architecture and focus on modelling artefacts that provide direct benefit during the early stages of achieving a balanced submarine design. This paper will discuss a number of important topics associated with building a system model that is intended to evolve from an initial concept design over the lifecycle of the submarine. Of particular interest is the reference architecture 'scaffolding' used to support capabilities such as variant modelling, and the integration of Failure Modes & Effects Analysis (FMEA) within the system model.**

Keywords-Submarine Design; System Architecture; Model-Based Systems Engineering; System Modelling; SysML

I. INTRODUCTION

A. Complex System Design

A conventional military submarine[1] contains more than forty subsystems covering a wide range of functions; from combat and weapons handling to services that provide cooling and hydraulics. These subsystems are highly integrated and collocated within the confines of a single pressure hull. Undesirable behaviour and properties often emerge once the submarine is operational. Of course, it is unwise to wait until a submarine is built to discover such deficiencies, as modifications to completed submarines are extremely costly and can deprive the nation of an important asset.

Computer modelling is increasingly used during the earliest design phases (when the cost of change is relatively low) to define a robust system architecture and predict the emergent behaviour and properties of a design. Design integrity, consistency and traceability can be enhanced further by integrating different computer models of the same system. The practice of using integrated computer models to govern the specification, design, evaluation and support of a system throughout its lifecycle is called Model-Based Systems Engineering (MBSE).

B. Model-Based Systems Engineering

The last decade has seen a rise in the application of MBSE across several industries, most notably in defence, but also in aerospace and rail. As a result, many different MBSE methodologies have been published by a range of practitioners, utilising a variety of different processes, models and tools [1]. A holistic multi-disciplinary 'system model' is central to most MBSE methodologies, which can be expressed in SysML [2].

The aim of this paper is to outline an MBSE approach for specifying and designing submarine subsystems in SysML during the early design phases. This approach has been developed by ASC Pty. Ltd. to support the architecting of submarine subsystems for Australia's Future Submarine as outlined in the 2013 Defence White Paper [3].

This paper is organised into three sections; requisite background concepts, the method itself, and finally a number of further considerations.

C. For The Practitioner

This paper is written for the practicing engineer who is looking for examples of how to apply MBSE and SysML to the design of complex subsystems. Indeed, the approach outlined in this paper is designed to be used by domain engineers with minimal exposure to systems engineering theory or practice (let alone SysML). The importance of making the system model accessible to individuals who are not systems engineers is a recurring theme in this paper.

[1] A conventional submarine is powered by batteries that are routinely recharged by diesel generators. For this paper, a modern conventional submarine is defined as built from the late-1980s, characterised by greater use of integrated computer technology, compared with earlier generations of submarines.

II. BACKGROUND

This section introduces a number of important concepts to prepare the reader for subsequent sections of this paper where the MBSE methodology is discussed. These concepts underpin a design process framework that is applied at each level of design, recursively down the left-hand side of the systems engineering 'V' lifecycle, from the whole-of-submarine to its subsystems and their components. This framework is aligned with technical systems engineering processes defined in ISO-15288 [4] and modelling activities defined by the Object-Oriented Systems Engineering Method (OOSEM) [5]. The creation and development of this framework to support early stage submarine design is explained in [6]. Essentially a matrix was defined with ISO-15288 processes and OOSEM activities on opposing axes. The resulting points of intersection found between processes and activities in this matrix were then grouped into the four process areas illustrated in Fig. 1. In this figure, the rectangles with rounded corners represent processes and the rectangles with square corners represent products.

The Requirements Development and Architectural Design process groups involve the development of the system specification and architecture respectively and are where MBSE has been most widely applied in industry. The Technical Evaluation group of processes define engineering analyses and trade-studies that are used to evaluate system requirements and design artefacts. Finally, at the whole-of-submarine level of design, the Synthesis group of processes define traditional ship design practices, including initial submarine sizing and estimation, spatial design and integration (using CAD), hydrostatic design, hydrodynamic design and the design of the submarine hull structures. At the subsystem level of design, Synthesis processes comprise initial estimation of equipment properties, preliminary system schematics and sizing calculations. Design synthesis at the subsystem level is discussed later in this section.

Each concept described in this section is first introduced using terms familiar to most systems engineers, and then a corresponding implementation of that concept will be defined in SysML. The concepts that are discussed include:

- Black-Box Specification
- Requirements Traceability
- Abstraction
- Inheritance
- Architecture
- Variant Modelling
- Model Organization
- Design Synthesis

A. *The Black-Box Specification*

During the development of system requirements, it is useful to view the system-of-interest as a 'black-box'; in terms of its interfaces with the external environment and other entities, and how it is expected to function and perform within that context. A black-box definition does not assume or expose the internal structure, behaviour or properties of the system. This approach delineates the definition of the 'specification' (what the system is expected to do) from any particular 'solution' (how the system could be implemented).

A black-box specification of a system should as a minimum define the following features; functions to be performed by the system; key properties or measures of performance for that system, and; external interfaces. These features result from, and are further refined by, requirements elicitation and analysis tasks performed for the system-of-interest, usually over several design iterations.

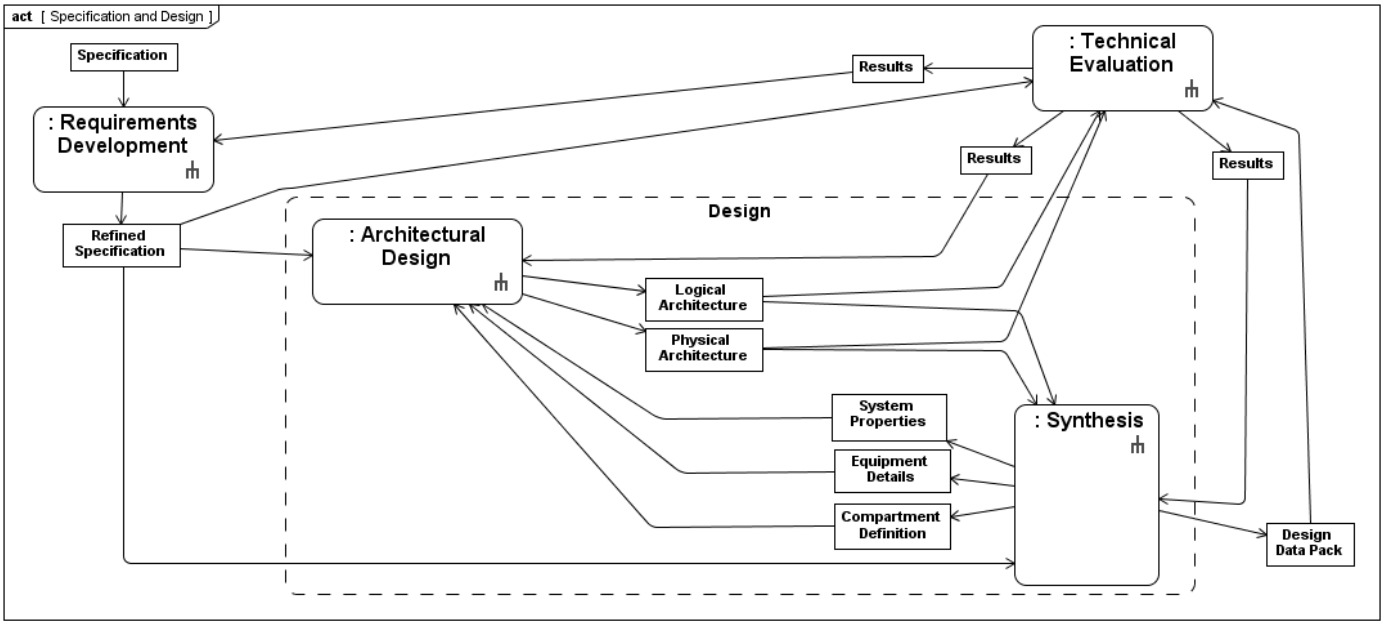

Figure 1. Specification & Design Process

A black-box specification can be defined in SysML using a 'block' element. A block defines features that map directly to system functions (as operations), properties (as system characteristics such as weight with values) and interfaces (as ports).

B. Requirements Traceability

A system specification can be viewed as an organised collection of textual requirements for that system. Beyond a certain threshold, specifications with a large number of requirements benefit from being managed in an electronic repository. This is certainly the case for submarine subsystem specifications, each of which can contain more than 1000 requirements. In such a tool, each requirement, in addition to its unique identifier and text, can be annotated with attributes such as performance bounds, priority, verification method and compliance status.

Textual requirement statements can be defined in SysML using 'requirement' elements. These elements contain fields for the requirement text and its unique identifier. Duplicating requirements between the requirements management tool and the system modelling tool can be avoided since most established SysML modelling tools support the synchronisation of requirement elements with their counterparts in an external requirements management tool repository, including traceability information.

By representing requirements in the system model, it is possible to link these elements with their corresponding feature in a black-box specification. In this paper, a «refine» relationship is used between requirements and black-box features to illustrate that the textual statement of the former 'refines' the latter. An example of a SysML black-box specification for a Diesel Generator System is illustrated in Fig. 2. In this diagram, and subsequent diagrams, the term 'system' is used interchangeably with 'subsystem'; as a diesel generator system is a submarine subsystem.

The chart in Table 1 summarises the mapping between types of requirements and black-box features. A populated black-box specification, with its features connected to corresponding requirements, represents the starting point for the development of one or more alternative architectures for a system-of-interest.

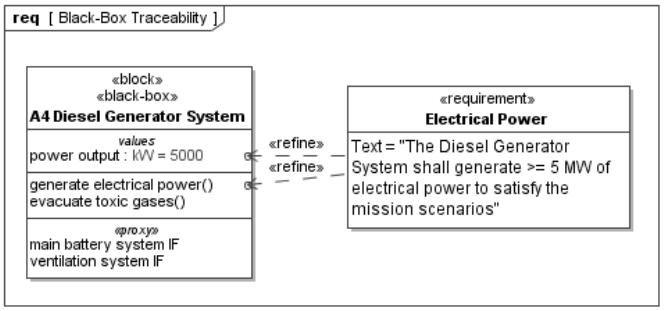

Figure 2. Example Black-Box Specification in SysML

TABLE I. MAPPING REQUIREMENTS TO BLACK-BOX FEATURES

Requirement Type	Black-Box Feature	
	General Term	***SysML Term***
Functional	Function	Operation
Performance	Property	Value
Interface	Interface	Port

C. Abstraction

Abstraction is an important technique for dealing with the complexity of a submarine subsystem design. For a particular level of abstraction, only relevant information and properties of the system are exposed and irrelevant lower level details are hidden. The level of abstraction is often correlated with the phase of the design. For example, a black-box specification is an extremely abstract representation of a system that hides nearly all details of the design. Conversely, a system design with a high level of physical detail and precision (as found on production drawings) is a very low level of abstraction.

D. Inheritance

The modelling approach described in this paper leverages several object-oriented design techniques embodied in SysML including inheritance and instantiation. The reader is referred to [6] for a discussion on the application of object-oriented concepts in submarine design. A brief introduction to the concept of inheritance will be provided here.

Recall that a black-box specification defines a set of features; functions, properties and interfaces. Typically, this specification is realised by one logical architecture, and one or more physical architectures (types of architecture are described in the next section). It is desired that each of these architectures share the same set of black-box features; indeed these architectures are 'inheriting' these features from a common black-box specification. This idea relates to the concept of abstraction described earlier; since each alternative architecture can redefine and extend the inherited features with additional functions, properties or interfaces to characterise that particular architecture (for example, Fig. 7 shows an extended black-box specification with an additional function and performance value).

In SysML, blocks can inherit features from other blocks if they are related using a 'generalisation' relationship, as illustrated in Figure 4 (as a line with an unfilled arrow). The advantage of this technique is that black-box features are only defined once in the system model (in the black-box specification) but can be reused or redefined as many times as required.

E. Architecture

Three distinct levels of system architecture are defined; functional, logical and physical, listed in order of decreasing level of abstraction and illustrated in Fig. 3.

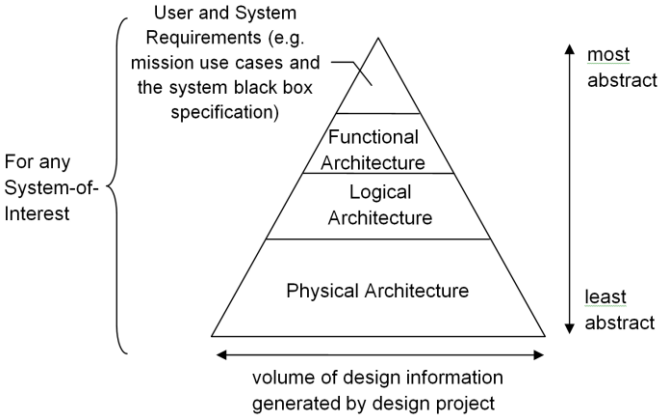

Figure 3. Levels of System Architecture and Abstraction

Functional architecture defines a solution-independent representation of the design; composed of pure functions[2]. In this paper, functional architecture is defined as the system functions contained in black-box specifications (as operations) across the submarine system hierarchy.

Logical architecture represents an intermediate abstraction between functional and physical architecture. Components of a logical architecture represent abstractions of physical solutions. For example, consider two diesel generator units delivered by two different suppliers. These units have many different physical characteristics (e.g. one is turbo-charged, one is not), but they share many common properties, such as power output and weight and they perform a common set of functions (e.g. generate electrical power). A component of a logical system thus defines functions, properties and interfaces that are common to a range of physical design alternatives. Most importantly, logical architecture remains largely independent of technology or suppliers and provides a reasonably stable baseline from which physical architecture can be derived and evolve. Logical systems and components are implemented in SysML as blocks with a «logical» stereotype applied[3].

Physical architecture is composed of tangible items of equipment that have been selected for a specific submarine subsystem design. If the information is available, physical components can be characterised by their supplier datasheets (e.g. a diesel generator from supplier X). Physical architecture also takes into consideration constraints such as redundancy (e.g. the number of diesel generators) as well as the arrangement and spatial layout of a particular submarine concept. Physical systems and components are implemented in SysML as blocks with a «physical» stereotype applied.

[2] function names use verb-noun combinations (e.g. "pump water")

[3] SysML stereotypes are used in this case to 'tag' a model element and extend it with a customised set of properties beyond the standard SysML definition. A comprehensive introduction to extending SysML using stereotypes is outlined in chapter 15 of [7]

Full design traceability is defined throughout each level of the architecture. Functions are assigned to logical and physical systems as block operations, inherited from the black-box specification. Secondly, the SysML «allocate» relationship is used to relate the components of logical architecture with their physical counterparts.

The example in Fig. 4 shows a function defined in the system black-box specification as an operation (recognised by the name followed by a set of closed parentheses), inherited by both Logical and Physical system representations using a generalisation relationship (a line with white closed arrow-head). The logical Diesel Generator System and a few of its components are allocated to physical counterparts using the allocation relationship (a dashed line with an open arrow at one end and an «allocate» label).

F. Variant Modelling

The preceding sections introduced architecture as levels of abstraction and differentiated between logical architecture (an abstraction of one or more physical solutions) from physical architecture (a specific design implementation).

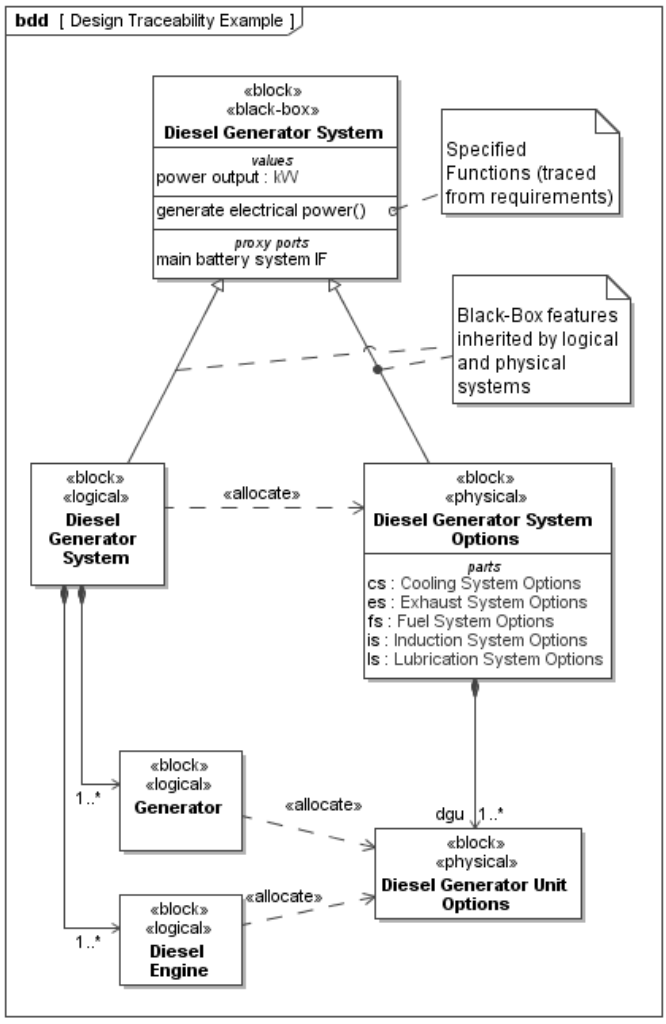

Figure 4. Design Traceability Example

A physical architecture evolves over time and can experience change from the top-down, driven by the introduction of new capabilities, or from the bottom-up, driven by new technologies, hardware/software upgrade cycles, or the need to replace obsolete equipment. Consequently, an approach to modelling variation in the physical architecture is required to minimise the total effort to capture, evaluate and maintain a number of variant architectures.

This paper outlines an approach to variant modelling proposed in [8], utilising the facilities of generalisation and redefinition in SysML to maximise the reuse of model elements. Model variation is provided by defining a physical 'options' tree containing the set of system elements that span variant designs, and one or more physical variant designs.

The physical options tree contains all allowable physical elements at each level of the submarine system hierarchy. In SysML, the top of the options tree is represented as a block. Fig. 6 illustrates the Diesel Generator System Options block and its composition as a set of physical components. Any particular physical design variant is defined by selecting a subset of these components. The pattern described above can be applied recursively to each level of the submarine product structure.

Any number of variations for a system-of-interest can be derived from the corresponding options tree. The architecture of a subsystem variant is defined by;

- a variant-specific black-box specification (corresponding to the specification being developed for the variant system), and;
- the variant system itself.

In SysML, the variant system is represented by a single block. An example is provided in Fig. 7, which defines an "A4 Diesel Generator System"; a diesel-generator system design for the submarine design called "A4". In this figure, it can be seen that the variant system inherits all of the features of its black-box specification as well as the selection of parts available from the options tree. It is possible to replace inherited block features with a different feature of the same name. In SysML this is called redefinition, and the new feature completely overrides the original feature. The benefit of redefinition is to change or extend a general feature in terms of its multiplicity, value or type (in SysML the definition of an element is commonly called its 'type'). For example, in Fig. 7, the *Diesel Generator Unit* part with a multiplicity of 'at least one' in the Diesel Generator System was changed to four distinct *Supplier X DG Unit* parts for the A4 Diesel Generator System variant.

G. Model Organisation

The system model developed in this paper is organised into three main packages; Reference Architecture, Variant Architecture and a Model Library as illustrated in Fig. 5. The Reference Architecture contains all system requirement sets that have been imported from a requirements management tool, all system functional and logical architecture and the physical options tree. The Variant Architecture contains all of the physical variant designs derived from the physical options tree. Many common definitions, types and elements are also defined in the Model Library.

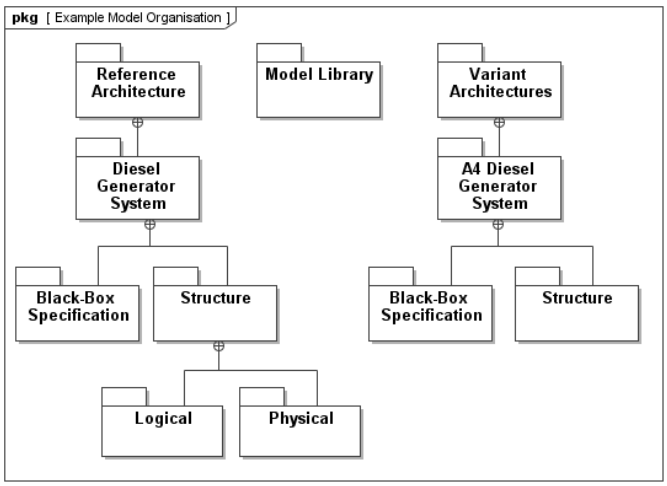

Figure 5. Example Model Organisation

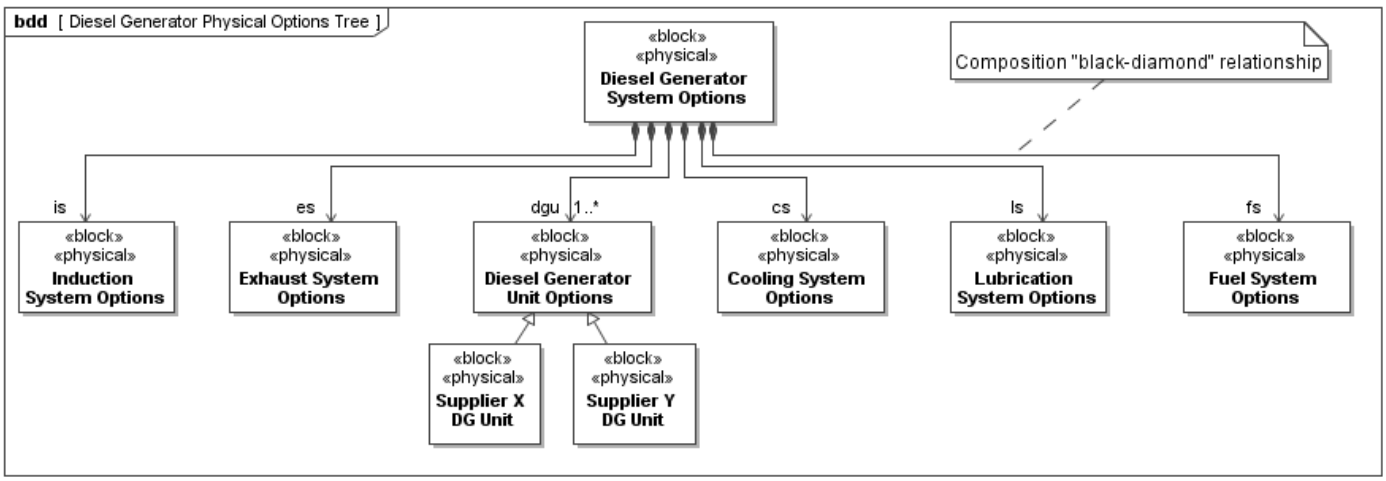

Figure 6. Example Physical Options Tree

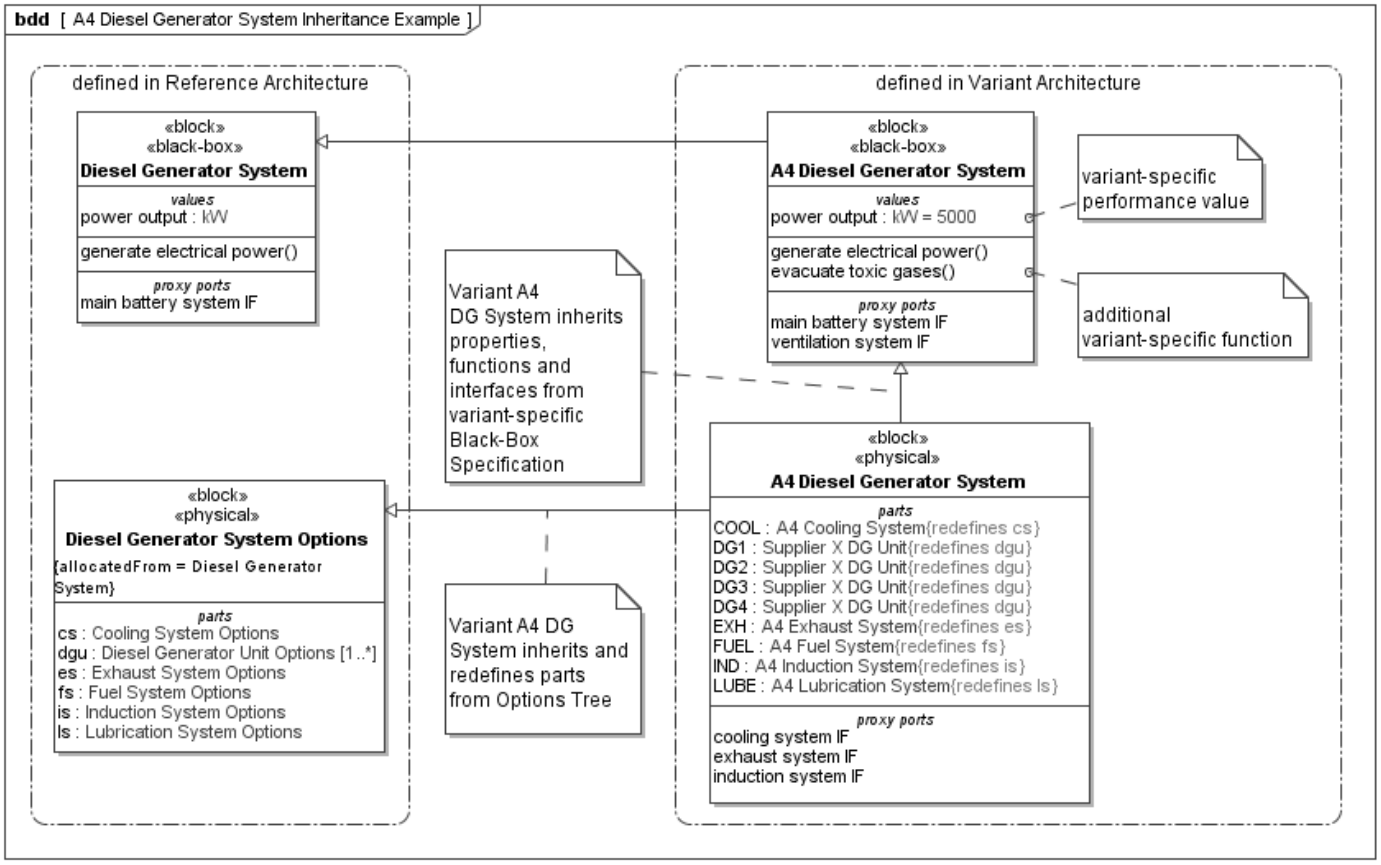

Figure 7. Example: Building a Physical Variant Design using Inheritance

H. Design Synthesis

Many technical design artefacts are developed by an engineer to synthesise a submarine subsystem design, including;

- schematics;
- parts lists;
- equipment datasheets and CAD models;
- engineering analyses and calculations;
- risk assessments, and;
- design reports.

Schematics are the keystone documents for a subsystem. These 'single-line diagrams' define the subsystem in terms of equipment, compartment location and interfaces between equipment and other subsystems. Subsystem schematics are typically developed in compliance with established standards such as Australian Standard (AS) 1100 [9].

A parts list is also associated with a subsystem schematic, and provides information relating to weight, hotel load, mechanical services and cooling for each listed part, in a tabular format.

Basic equipment sizes are sourced from supplier datasheets, and this information is combined with whole-of-submarine requirements for maintenance access, signatures and shock clearances to define envelopes for 3D parts that can be integrated within the submarine CAD model.

Engineering analyses and calculations are often used to select and estimate equipment and tank properties from subsystem requirements. The results from this work are reflected in parts lists and in the spatial information provided to the submarine CAD model. System- and Technical Readiness Levels (SRLs and TRLs) are also estimated for the system and for its key components respectively.

A subsystem design report describes the function of major elements of the subsystem and how they work together. This report summarises the work undertaken, including references to the artefacts that have been created, and recommends the next steps required for further development of the subsystem. Most importantly, this document provides the rationale for key design decisions that were made throughout the design.

The following sections also explain how the traditional artefacts described above can be integrated with a subsystem architecture defined in SysML, as part of the MBSE method.

III. MBSE Method Applied to Subsystem Architecture Design

This section outlines an approach to building subsystem architecture in SysML, using the concepts described in the previous section. The basic process is illustrated in Fig. 8, and spans the Specification & Design process described in Fig. 1. When applied to subsystem design, the first three steps are part of Requirements Development, the middle and the final steps are part of Synthesis, and remaining steps are part of Architectural Design. These processes are also recursive and can be applied to any level of the submarine design.

This approach assumes that preliminary requirements have been developed for the subsystem and are being managed in a requirements repository integrated with the SysML model. Furthermore, it is assumed that a logical architecture has been previously defined for this subsystem and that the physical options tree is already populated with the parts needed to build a new variant design. From this basis, the following subsections proceed to construct a new physical variant design.

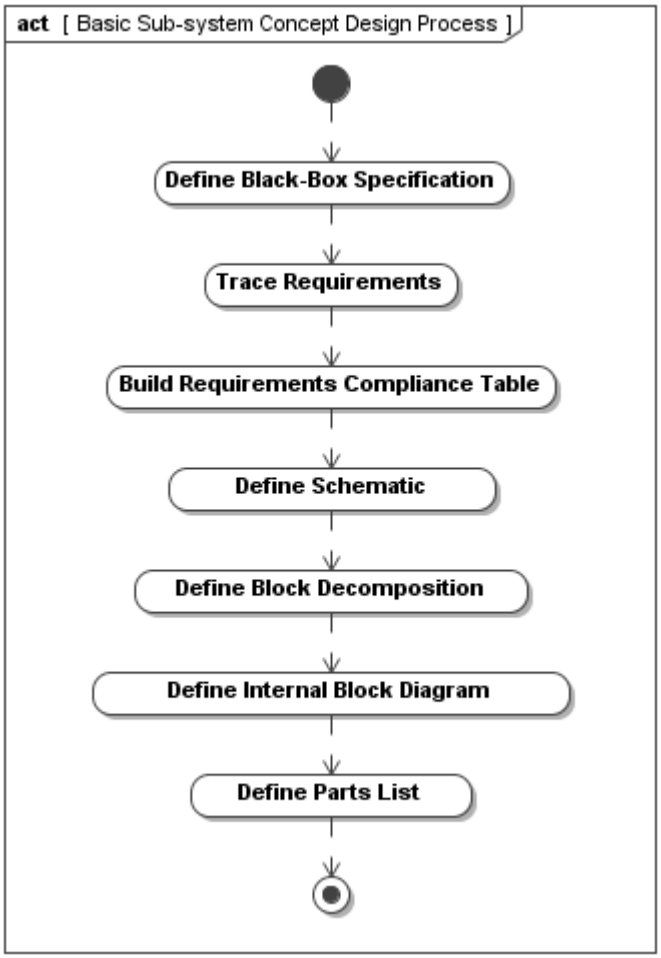

Figure 8. Basic Subsystem Concept Design Process

A. Define the Black-Box Specification

First a black-box specification block is defined for the subsystem. Key function, performance and interface requirements are identified from the requirements set and represented as operations, properties with values and ports on this block. Some features may be inherited and redefined from the generic black-box specification for the system (contained in the Reference Architecture). In Fig. 9, the upper block and the lower block represent the generic and A4-specific diesel generator system black-box specifications respectively.

B. Trace Requirements

Each feature of the black-box specification can then be traced to its corresponding requirement statement, which is also modelled in SysML as a requirement element. A «refine» relationship is used, connecting the requirement element to its black-box feature, as was illustrated in Fig. 2.

C. Build Requirements Compliance Table

As the black-box is populated and traced to requirements, it can be helpful to view this information in a table. The Requirements Compliance Table in Fig. 10 can be generated very quickly in the system model and is intended to be presented by the subsystem engineer at a requirements review for their system. As the design progresses, it can be asserted (using a SysML «satisfy» relationship) that elements of the design satisfy a particular requirement, and the name of each element that satisfies the requirement will appear in the corresponding "Satisfied By" column.

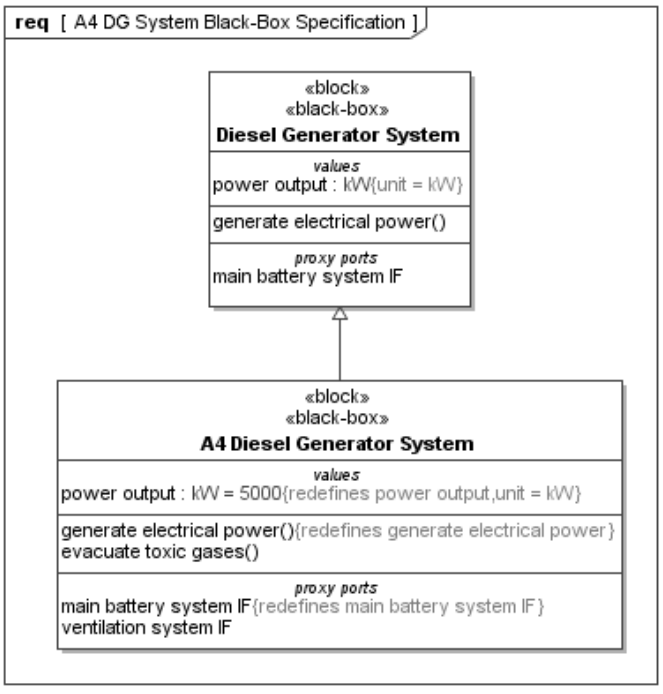

Figure 9. Defining a Variant Black-Box Specification

#	Name	Text	Satisfied By	Rationale	Black-Box Features
1	Toxic Gas Evacuation	The Air Dependant Propulsion System shall ventilate the submarine to = [TBD limits] while snorting			evacuate toxic gases()
2	Electrical Power	The Diesel Generator System shall generate >= 5 MW of electrical power to satisfy the mission scenarios		Refer to Analysis in	power output : kW generate electrical power()
3	Main Battery Interface	The Diesel Generator System shall supply electrical power to the Main Battery System			main battery system IF

Figure 10. Requirements Compliance Table

D. Define Schematic

At this step, the designer focuses internal to their black box and synthesises a candidate subsystem architecture design, as discussed earlier the previous section. This work draws heavily on earlier designs, equipment trade-offs and sizing calculations. A key result is a schematic (a 'single-line diagram'). The subsystem designers have traditionally sketched this diagram in Microsoft® Visio using standard symbol libraries. An example is illustrated in Fig. 11. This schematic is used to communicate the designer's intent during the formation of the subsystem concept design and often lacks precision and consistency. The following sections explain how this design intent is then formalised using SysML.

E. Define Block Decomposition

With a preliminary system schematic in-hand, the designer can identify the components of the system. Within the system model, the subsystem variant block can then be decomposed into its constituent parts, redefining existing parts from the physical options tree or creating new parts as required. The resulting system hierarchy can be represented on a SysML Block Definition Diagram (BDD), as illustrated in Fig. 12.

F. Define Internal Block Diagram

The system schematic can then be represented as a SysML Internal Block Diagram (IBD). The diagram frame represents the subsystem boundary. The ports on the frame represent the subsystem external interfaces, and are inherited from the black-box specification. Within the frame, the diagram is populated with the subsystem parts, and these parts can be connected together and to the ports on the diagram frame. Parts can be connected informally without ports, but the ports represent a more formal access point on the boundary of a block or part. As the design matures, ports can be defined in SysML with increasing detail corresponding to a more detailed interface definition. An example IBD is illustrated in Fig. 13.

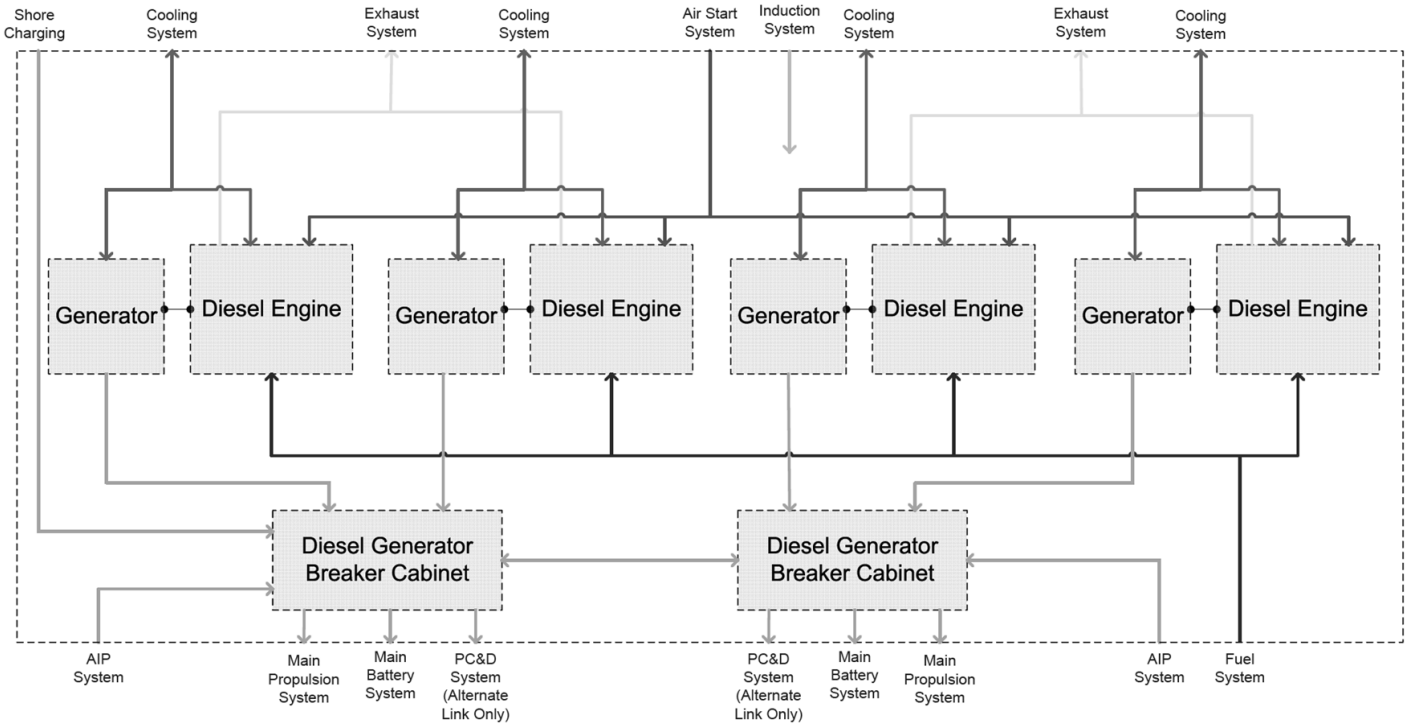

Figure 11. Example Preliminary Subsystem Visio Schematic

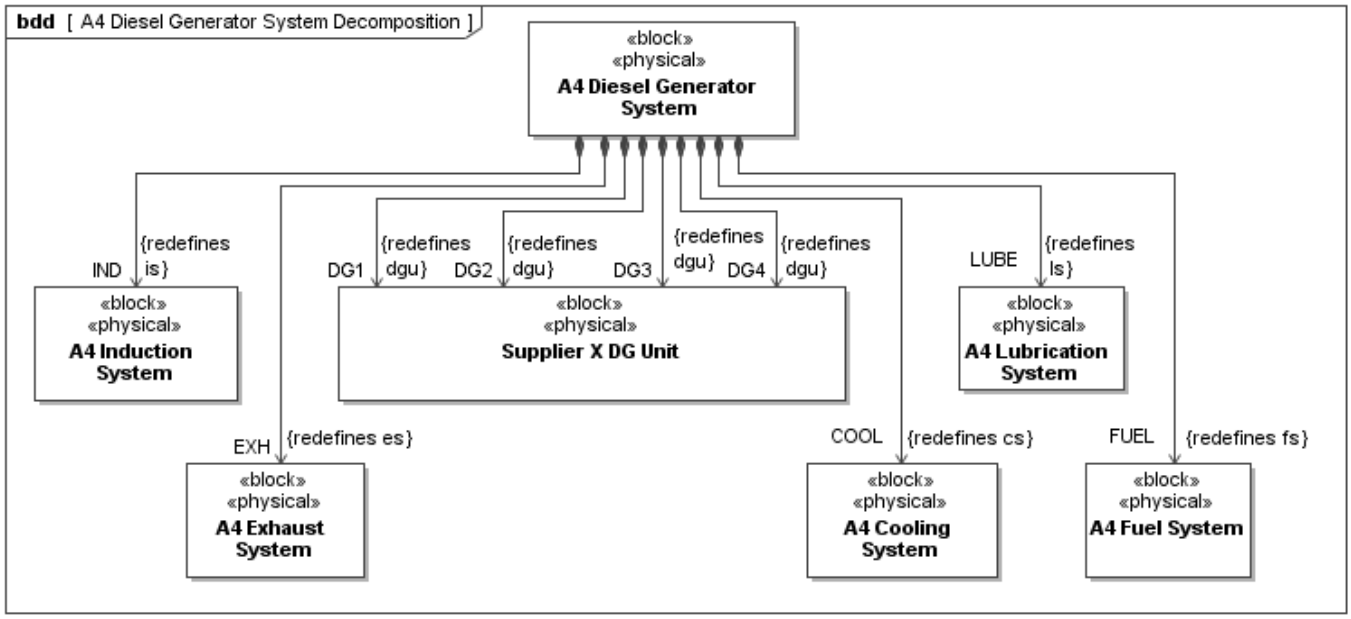

Figure 12. Subsystem Block Decomposition Diagram

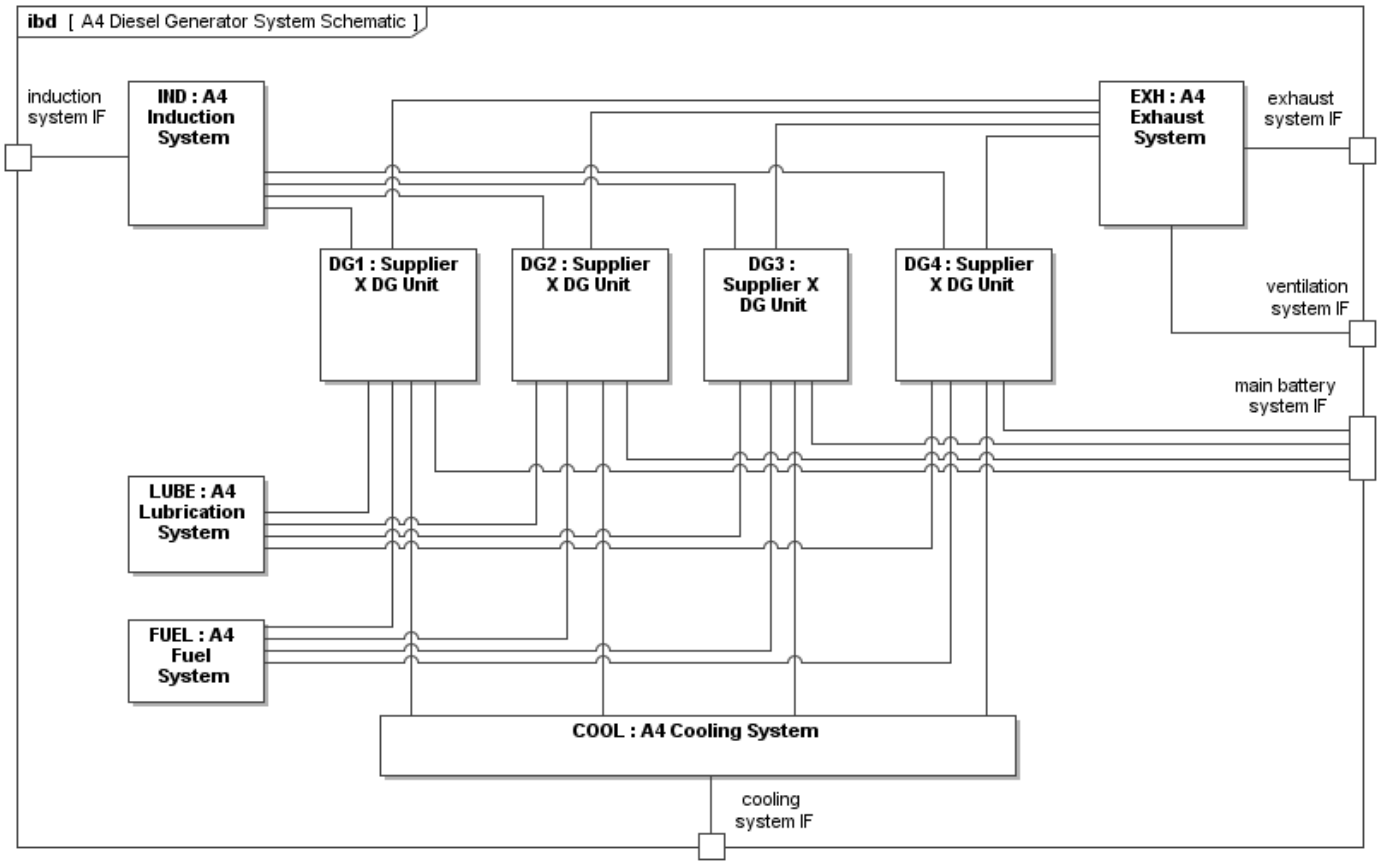

Figure 13. Subsystem Internal Block Diagram

G. Define Parts List

Finally, a parts list can be generated for all parts owned by the variant system block, as illustrated in Table 2. This table can be extended with columns to capture additional information associated with each part, such as its allocated compartment, build section or supply item serial number.

TABLE II. SUBSYSTEM PARTS LIST

#	Name	Type	Allocated Compartment
1	COOL	A4 Cooling System	
2	DG1	Supplier X DG Unit	Main Generator Room
3	DG2	Supplier X DG Unit	Main Generator Room
4	DG3	Supplier X DG Unit	Main Generator Room
5	DG4	Supplier X DG Unit	Main Generator Room
6	EXH	A4 Exhaust System	
7	FUEL	A4 Fuel System	
8	IND	A4 Induction System	
9	LUBE	A4 Lubrication System	

The method described in this section allows subsystem designers to capture the core architecture of their design, in a short number of steps, in an environment that supports full design traceability, subsystem integration and design variants. The next section discusses many aspects of this approach including areas where further work has been identified.

IV. FURTHER DISCUSSION

Traditional submarine subsystem design artefacts are document-centric. In recent decades, electronic publishing has replaced hardcopy reports and drawings, however even electronic documentation, such as Microsoft® Word and Visio or Adobe Acrobat PDF files are essentially static products that may be imprecise and difficult to maintain. In most cases such artefacts are developed by different individuals with different backgrounds, with different tools, often from different locations. Even the most pedantic desktop review will fail to detect every inconsistency and mistake within or across a set of Word reports, Visio schematics, or PDF drawings. Once these documents are published, the design is still subject to ongoing change, and the need to maintain the quality of a large document set over a long period of time presents a significant challenge. Applied carefully, systems modelling can help subsystem designers manage their design artefacts and associated data, by providing a common and consistent model within which multiple subsystems can be developed and integrated.

Using a system model to develop and manage subsystem architecture is a relatively new practice (certainly for the submarine design community) and it is essential that the design method fully engages individuals unfamiliar with MBSE. With this organisational goal in mind, this section discusses a number of topics surrounding the method described in section two, including tailoring the scope of this work, the de-emphasis of behavioural modelling and why Microsoft® Visio sketches may be used as precursors to formal SysML IBDs. There are also many opportunities to augment this approach, and this paper will discuss utilising symbol libraries in IBDs, the integration of Failure Modes and Effects Analysis (FMEA) data and the integration of calculation results from external tools.

A. Process Brevity

One strategy for engaging domain (non-systems) engineers is to adopt a policy of brevity. In other words, any system modelling approach must produce the most useful diagrams and tables in as few steps as possible. Furthermore, early stage submarine design is characterised by a small number of key properties, attributes of major equipment and layout sketches. Combining the need for brevity with a numeric design definition resulted in a subsystem concept design method that focussed on the reduced set of diagrams and tables described in the previous section.

B. Behavioural Modelling

In total, SysML defines nine diagram types, spanning four categories; requirements, behaviour, structure and parametrics [2]. The method described in this paper is limited to requirements and structural diagram types; i.e. the Requirements Diagram; Block Definition Diagram and Internal Block Diagram. SysML parametrics will be introduced later in this paper. The reader may observe that this method lacked behavioural modelling. This omission diverges from a traditional top-down systems engineering approach, so the following paragraphs will provide some justification for this approach.

Behavioural diagrams in SysML include Activity Diagrams, Sequence Diagrams and State-Machine Diagrams. An Activity Diagram defines system behaviour in terms of flows of data or material between system functions and can often contain partitions showing how flows and functions are allocated to system components. Activity Diagrams are closely related to Functional-Flow Block Diagrams (FFBDs) [10] (pp. 713-715) or IDEF0 [11] and are often used to illustrate operational scenarios for a system-of-interest. Sequence Diagrams define interactions between systems as a temporal sequence of messages exchanged between system elements. State-Machine Diagrams describe the response of a system to external and internal events, specifically in terms of states and transitions between these states. State Machine Diagrams also define functions that are performed in each state in response to explicit events.

The SysML behavioural diagrams are very expressive, but also can be complicated and time consuming to develop. In this light, the desire for brevity in the design process alone constitutes an argument for de-emphasising behavioural modelling.

More importantly, submarine design is an evolutionary process, where each design gradually improves upon previous designs. Indeed the list of subsystems and equipment types used in modern military conventional submarines is well established. A Ships Work Breakdown Structure (SWBS) is a standard list often used by surface ship and submarine designers to organise subsystems and equipment [12]. Furthermore, decades of development and experience have proven the architecture of many submarine subsystems. For example, seawater cooling and trim systems conform to a general pattern that can be recognised across different classes

of modern military submarines. A submarine seawater cooling system is generally composed of inlet hull valves, pipework, a pump assembly, a heat exchanger (to transfer heat away from internal water cooling systems) and outlet hull valves. A submarine trim system nearly always features tanks at opposite ends of the submarine and a pump assembly to transfer water between the tanks.

The functions of submarine subsystems and equipment are often poorly documented in traditional ship design practice, but are nevertheless well understood by experienced submarine designers. The emergent behaviour of integrated subsystems is less well understood. However since submarine concept design is primarily focused on a spatial definition of a concept submarine related to size, weight, and power, capturing system behaviour in terms of equipment functions may not contribute substantially to these early trades. The elaboration of system behaviour can follow after the concept design phase when behaviour analysis is needed to help more detailed functional requirements of the hardware and software. Consequently, identifying the subsystem functions in a black-box is sufficient to constitute the basis for the initial functional architecture for a submarine design (this was referred to earlier in this paper). Capturing the system functions as part of the black box specification can be used to flow-down the system functional requirements to subsystem functions, and each of the subsystem functions can be further elaborated by behavioural diagrams when required in subsequent design phases.

There are, of course many aspects of design where behavioural modelling must be undertaken as early as possible. For example, introducing unproven or completely new submarine systems or equipment requires an understanding of how these systems or equipment will function once integrated with other subsystems. Systems with a high degree of human interaction or those that contain a high percentage of software typically exhibit very complex operational behaviours and so necessitate behavioural modelling. In the phases that follow concept design, it is anticipated that behavioural modelling will be increasingly used to specify the subsystems, equipment, and the human-system interactions.

C. Sketch first, then Model

It is possible to devise a standard top-down approach to modelling systems in SysML. Such an approach might start with a set of requirements and identify functions as SysML Activities. These Activities could then be allocated to systems, subsystems, and components (modelled as SysML Blocks). The system decomposition would be defined in a BDD, and a schematic for that system would be defined in an IBD. Such an approach is described in [7] (pp. 33-41) as 'SysML-lite'. A top-down approach would then suggest that the IBD be used to develop schematics that conform to industry drawing standards However, the method described in this paper takes a different approach; as the development of a Visio system schematic precedes the development of the system IBD. There are two reasons for this decision; to better engage the domain engineer, and the lack of standard symbol libraries in commercial SysML modelling tools.

It is important that domain engineers be allowed to sketch their design schematics in their tool of choice (typically Microsoft Visio, but also hand-drawn diagrams are still very common). This ensures that engineers can focus on getting their design correct at the start without being encumbered or distracted by the complexities of using a SysML tool. Once the domain engineer is satisfied with their sketches, then system modelling is undertaken to formalise their design. Almost universally, it has been found that the formal system modelling process has helped domain engineers improve their designs from their original sketches. Firstly, the system model provided a much higher level of consistency in terms of names and types. Secondly, the identification of new interfaces was widely reported by domain engineers and demonstrated the ability of the system model to facilitate subsystem integration even from the earliest design phases.

Furthermore, engineers are required to communicate their designs with schematics that contain standard symbols for mechanical and electrical parts. These symbols are widely recognised by industry and there is a strong case to utilise these symbols in SysML IBDs. A comparison between the symbol for a SysML part and its counterpart in ISO 14617 [13] is provided in Fig. 14. The latter symbol clearly presents more information to the reader than the former.

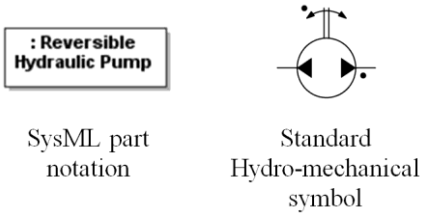

Figure 14. Comparing SysML part notation with an industry- standard hydromechanical symbol

Unfortunately current SysML modelling tools lack this capability 'out-of-the-box' and the undesirable consequence is that domain engineers then duplicate their schematics; on paper; in Visio; in SysML, and then in CAD (e.g. for production drawings). Most SysML tools do provide the ability to apply icons to model elements, and there are some efforts by individual organisations to customise their SysML tool-suite in this way to provide this capability to their own engineering teams [14] (p. 15). However the SysML community is still waiting for tool vendors to provide a commercial solution.

D. Failure Modes & Effects Analysis

A black-box specification provides a list of system functions that are a useful starting point for early stage Failure Modes & Effects Analysis (FMEA). For each function, one or more failure modes can be identified and each failure mode is characterised by a range of attributes including a unique identifier, an end effect, a cause, estimated severity, likelihood and a hazard risk index. The assessment of failure modes during the concept design phase will be approximate, but nevertheless allow subsystem design engineers to identify and address aspects of their design that may be problematic. Subsystem design engineers are normally assisted by reliability engineers who are familiar with typical submarine subsystem

failure modes, causes and effects. As a subsystem design is developed further, estimates of the severity and likelihood of failure modes are revised accordingly.

A meta-model has been developed for defining failure modes in SysML and assigning these failure modes to black-box functions. This model extends the standard SysML language with additional concepts to describe failure modes. This is achieved with SysML stereotypes. The technical theory behind extending SysML is quite involved and outside the scope of this paper, suffice to say that stereotypes are defined with a set of custom properties. A stereotype can then be applied to a model element, which results in the model element being extended with additional properties. A meta-model then defines how different stereotypes and model elements can relate to each other. A simple FMEA meta-model is illustrated in Fig. 15.

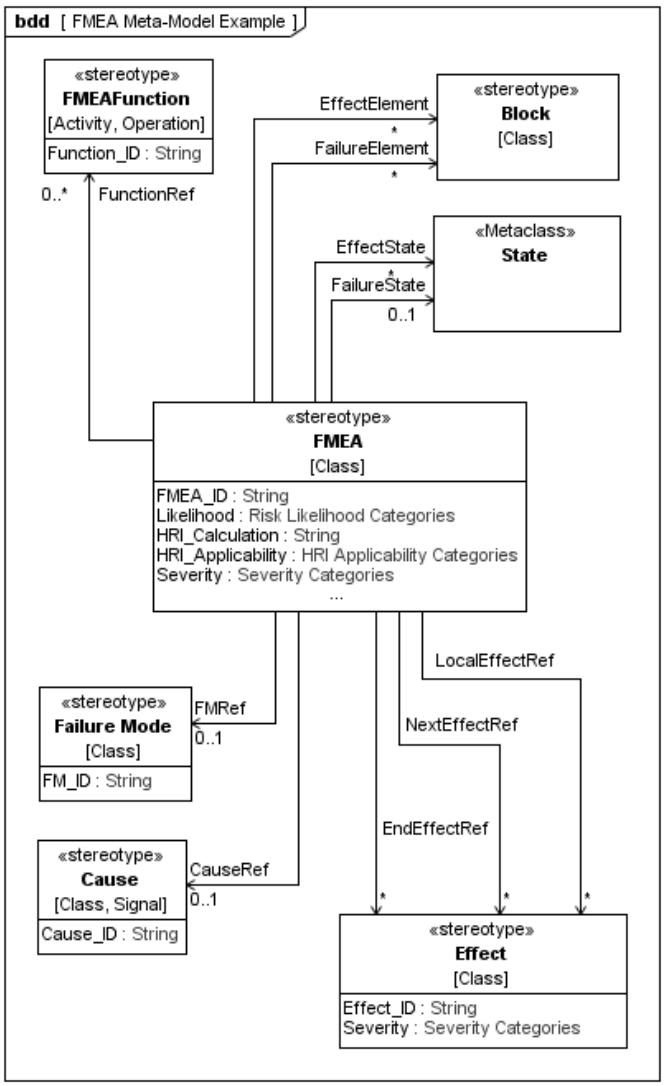

Figure 15. FMEA Meta-Model

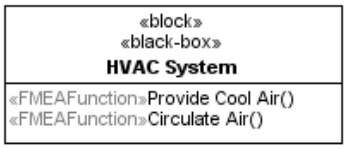

Figure 16. FMEA applied to an HVAC System Black-Box

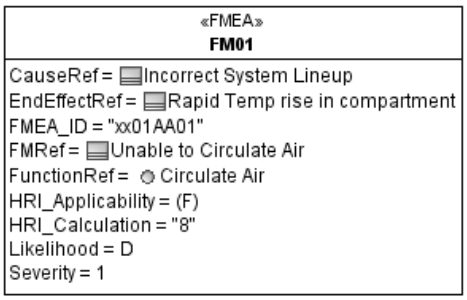

Figure 17. Example FMEA for an HVAC System

As an example, consider a Heating Ventilation & Cooling (HVAC) system black-box specification in Fig. 16. Two functions of that black-box are subject to FMEA, as signified by the 'FMEAFunction' stereotype. According to the meta-model in Fig. 15, each 'FMEAFunction' can be associated with one or more FMEA model elements (each representing an analysis), which are in turn associated with model elements representing an identified Failure Mode, Cause and zero or more Effects. A predefined set of Failure Modes and Causes are contained in the Model Library to limit the creation of arbitrary or poorly defined failure modes or causes.

An FMEA can also be associated with model elements that represent parts and states of a system. An example FMEA for the HVAC system is provided in Fig. 17.

The results of multiple FMEA items can be readily summarised for a system in a table, as illustrated in Fig. 18. If the FMEA data shown in this table were defined and managed in a Microsoft® Excel Spreadsheet, it could be synchronised with the system model using SysML Parametrics.

E. SysML Parametrics

Engineering analyses and calculations are a key aspect of design synthesis, as outlined at the end of section II. In most cases, the result of this work defines or constrains the properties of a system or its components. Consider a software tool for calculating the initial properties of a submarine battery system. This tool can be represented in SysML as a Constraint Block that defines a set of parameters and equations to represent the sizing algorithm. In SysML, a Parametric Diagram defines how the parameters of this analysis are bound to the properties of the related systems.

In Fig. 19 a (very simplified) battery sizing tool is represented by a Constraint Block that takes the desired properties of the Submarine as inputs and estimates the properties of the battery subsystem.

#	FMEA_ID	FunctionRef	FMRef	EndEffectRef	CauseRef	Severity	Likelihood
1	xx01AA01	Circulate Air()	Unable to Circulate Air	Rapid Temp rise in compartment	Incorrect System Lineup	1	D
2	xx01AA02	Circulate Air()	Unable to Circulate Air	Rapid Temp rise in compartment	Equipment Failure	1	C
3	xx01AA03	Circulate Air()	Unable to Circulate Air	Rapid Temp rise in compartment	Bulkhead shutdown	1	D
4	xx01AB01	Circulate Air()	Reduced Air Flow	Gradual Temp rise in compartment	Incorrect System Lineup	3	D
5	xx01AB02	Circulate Air()	Reduced Air Flow	Gradual Temp rise in compartment	Equipment Failure	3	C
6	xx02AA01	Provide Cool Air()	Temperature outside threshold boundaries	Compartment Temp too hot/cold	Temperature control system defective	2	D
7	xx02AA02	Provide Cool Air()	Temperature outside threshold boundaries	Compartment Temp too hot/cold	Equipment Failure	2	C

Figure 18. Example FMEA Summary for an HVAC System

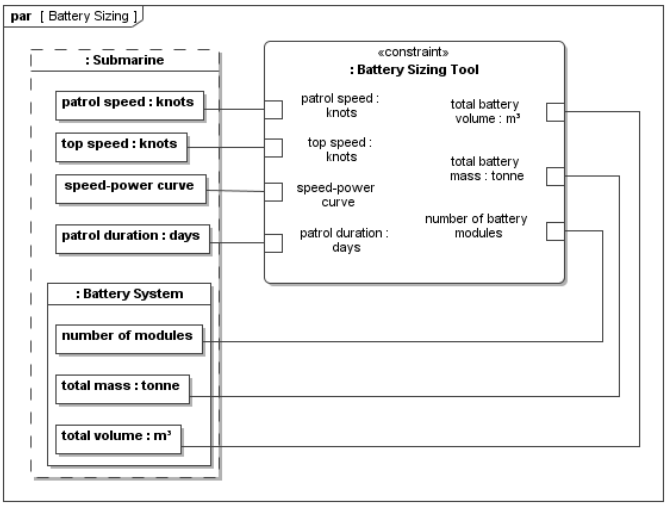

Figure 19. SysML Parametric Example: Battery Sizing Tool

Most SysML modelling tools provide a plugin that can remotely execute models in external mathematic solvers whose parameters match the parameters of the SysML Constraint Blocks. In this way, it is possible to integrate engineering analyses and calculations with the system model. This capability provides an even greater level of design traceability by enabling a direct link between the subsystem architecture and engineering analyses.

V. SUMMARY

This paper commenced by introducing the reader to a number of key background concepts and topics in section II. Firstly, a recursive specification and design process framework was introduced within the context of the systems engineering 'V' diagram. This section then defined the 'black-box specification' construct, different types of system architecture and the concepts of abstraction and inheritance (object-oriented techniques borrowed from the software community). Variant modelling was also introduced in this section, along with supporting architectural constructs; the physical options tree and physical variant designs. This section concluded with an overview of artefacts traditionally developed as part of the synthesis of a subsystem concept design.

Section III addressed the key aim of this paper, and outlined a short series of steps for defining a subsystem design in SysML. This was demonstrated by gradually building a simplified subsystem model from a black-box specification through to a schematic defined in a SysML Internal Block Diagram.

The discussion in section IV returned to the theme of engaging the domain engineer, highlighting the importance of defining a simple linear process, justifying why behaviour modelling can be de-emphasised during the earliest stages of submarine concept design, and why informal sketches are useful precursors to formal system modelling. The act of formally defining a subsystem in SysML helped engineers improve the quality of their work, when compared to their original sketches. It was also observed that current SysML modelling tools do not provide industry-standard symbol libraries, and that this omission could discourage the adoption of SysML modelling in organisations. Finally, two extensions to the systems modelling approach were presented to the reader; FMEA and SysML parametrics.

VI. CONCLUSION

This paper has outlined an approach to modelling submarine subsystem architecture in SysML. Alongside the primary goal of defining an MBSE method, it must also be practical, and thus a recurring theme of this paper is the need to make the method accessible to domain engineers who may not possess a background in systems engineering.

The advantages of designing submarine subsystems in this way reflects the objectives of MBSE, namely; improved design consistency, precision, traceability, subsystem integration, and design evolution. The approach outlined in this paper has been practiced as part of an integrated submarine design activity and the benefits above have been observed, particularly with regard to subsystem integration.

A significant contributing factor towards the success of this approach was making system modelling accessible to domain engineers. In the shipbuilding industry, where traditional ship design practices persist, this factor cannot be understated.

REFERENCES

[1] J. A. Estefan, "Survey of Model-Based Systems Engineering (MBSE) Methodologies," INCOSE MBSE Focus Group, Jet Propulsion Laboratory, California Institute of Technology, 2008.

[2] OMG, "OMG Systems Modeling Language," [Online]. Available: www.omgsysml.org . [Accessed 19 April 2013].

[3] Australian Government Department of Defence, "Defence White Paper 2013," 3 May 2013. [Online]. Available: http://www.defence.gov.au/whitepaper2013/docs/WP_2013_web.pdf. [Accessed 17 May 2013].

[4] ISO, "ISO/IEC 15288:2008(E) Systems and Software Engineering - System Life Cycle Processes," 2008.

[5] INCOSE, INCOSE Systems Engineering Handbook: A Guide for Systems Life Cycle Processes and Activities, v3.2 ed., C. Haskins, Ed., 2010.

[6] P. Pearce and M. Hause, "ISO-15288, OOSEM and Model-Based Submarine Design," in Proceedings from the SETE/APCOSE Conference 2012, Brisbane, QLD, 2012.

[7] S. Friedenthal, A. Moore and R. Steiner, A Practical Guide to SysML - The Systems Modeling Language, 2nd ed., Morgan Kaufmann/OMG Press, 2012.

[8] S. Friedenthal, "Variant Modeling Approach," 2012.

[9] Standards Australia, "AS 1100 Technical Drawing," 1992.

[10] B. S. Blanchard and W. J. Fabrycky, Systems Engineering and Analysis, 5th ed., Pearson Education Australia, 2011.

[11] Knowledge Based Systems Inc., "Integrated DEFinition Methods - IDEF0 Functional Modeling Method," [Online]. Available: www.idef.com/idef0.htm . [Accessed February 2012].

[12] NAVSEA, Expanded Ship Work Breakdown Structure for all Ships and Ship/Combat Systems, vol. I and II, 1985.

[13] ISO, "ISO 14617 Graphical Symbols for Diagrams," 2005.

[14] B. Cole, "Libraries and Domain-Specific Modeling," [Online]. Available: www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:2013_iw_mbse_libraries_sat_sun_breakout_takeaway_pts.ppt. [Accessed 9 May 2013].