The University of Tokyo - Graduate School of Engineering -
Department of Systems Innovation

Knowledge Management and System-Level Design Tools utilizing OPM and Modelica for a Student Solar-Boat Project

Master's thesis

Joshua Sutherland (37-146879)
Supervisor: Prof. Aoyama

March 2016

Table of Contents

List of Figures

List of Tables

List of Appended Publications

The following published papers were extended as part of this thesis:

- Sutherland, J., Kamiyama, H., Aoyama, K., & Oizumi, K. (2015). Systems Engineering and the V-Model: Lessons from an Autonomous Solar Powered Hydrofoil. Presented at the 12th International Marine Design Conference (IMDC), Tokyo Japan.
- Sutherland, J., Kamiyama, H., Oizumi, K., & Aoyama, K. (2014). Future Research Requirements for the Systems Engineering Discipline: Lessons Learned from the Autonomous Solar Boat Racing Project of Students of the University of Tokyo and the KTH Royal Institute of Technology. Presented at the 第 6 回 システム創成学学術講演会 新たな工学知の創成と融合, Department of Systems Innovation, School of Engineering, the Univ. of Tokyo.

The following papers which are based on this thesis were in review when this thesis was submitted:

- Sutherland, J., Oizumi, K., Aoyama, K., Takahashi, N., & Eguchi, T. (2016). System-Level Design Trade Studies by Multi Objective Decision Analysis (MODA) utilizing Modelica. In *1st Japanese Modelica Conference*. Tokyo Japan
- Sutherland, J., Oizumi, K., Aoyama, K., Eguchi, T., & Takahashi, N. (2016). System-Level Design Tools Utilizing OPM and Modelica. In *ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference IDETC2016*. Charlotte, North Carolina

Acknowledgements and thanks

Thanks are given to Prof. Aoyama for allowing me to join his laboratory in Japan and explore our common interest in Systems Engineering. His tuition, guidance and criticism have been extensive and highly valued. I have tried my upmost to make best use of this opportunity.

In addition Dr. Oizumi has provided much additional feedback, ideas, explanations and guidance. Thank you.

Dr. Eguchi's extensive industrial experience and knowledge was highly useful. Many thanks for this.

Thanks is given to Mr. Takahashi from Dassault Systèmes who introduced me to Dymola and Modelica. His encouragement, help and knowledge of how customers make use of simulation "in the wild" was gladly received.

To the various researchers and engineers around the world of took the time to respond to my rather naïve questions by email including: Dr. M Tiller of Xogeny, Prof. D Dori of MIT, Dr. A Qamar of Georgia Tech, Prof. M Otter of DLR, Dr. M Nicolai of Siemens and Mr. M Cilli of U.S. Army. Many thanks.

Thank you to all those who pay taxes in Japan for funding my scholarship distributed by the Japanese Ministry of Education, Culture, Sports, Science & Technology (MEXT). I have done my upmost to learn while simultaneously educating others; this money has not been squandered. Thank you to the Japanese Embassy in London for awarding me this scholarship.

And to everyone from the Manufacturing Systems Engineering Laboratory for making me welcome in Japan and providing plenty of comradery. It has been fun.

1 Introduction

1.1 Chapter purpose

This chapter aims to provide the reader with:

- Familiarization as to what the Solar-Boat project is
- Explanation of how problems with past Solar-Boat projects were identified and what those problems were
- For each of the identified problems a descriptions are provided of:
  - What makes these problems difficult to solve
  - Some high level proposed solutions

As a result of this the thesis title of "Knowledge Management and System-Level Design Tools utilizing OPM and Modelica for a Student Solar-Boat Project" should be broadly understood.

1.2 Student Solar-Boat project

1.2.1 Lake Biwa Crewless Solar Powered Boat Race

Every summer the Department of Systems Innovation at the University of Tokyo forms a student team to compete in the annual Lake Biwa Crewless Solar Powered Boat Race. At this event students from multiple universities race solar automatous solar powered boats which they have designed and manufactured at their respective universities.

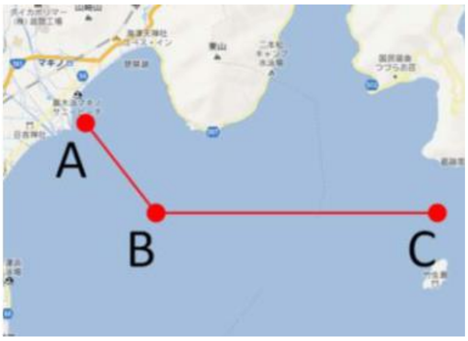


Figure 1 Route map of the crewless Solar-Boat race. (Frithiof et al., 2013)

A successful race entrant will complete the 20km course, Figure 1 (waypoint A to B to C to B to A) in the fastest time. Subject to a time penalty, students may attempt to repair their boat on route if it suffers a break down. All boats have the opportunity to compete twice over a weekend with scores being added together. As of 2015 the boats are subject to the technical constraints listed in Table 1.

Component	**Rule**
Solar panels	Maximum $2m^2$
Batteries for propulsion	Maximum 25Wh of lead based
Cargo	Must carry GPS logger (64g 68mm x 46mm x 18mm)

Table 1 2015 Lake Biwa Crewless Solar Powered Boat Race technical constraints

1.2.2 Identifying problems with 2014 Solar-Boat development

The author was able to participate in the 2014 Solar-Boat project to develop and then race a boat as part of the 2014 competition which lasted from April to September. The resulting output of the project was the design and manufacturing of a hydrofoil which was ranked 3rd in the race, Figure 2 provides a brief overview of this system.

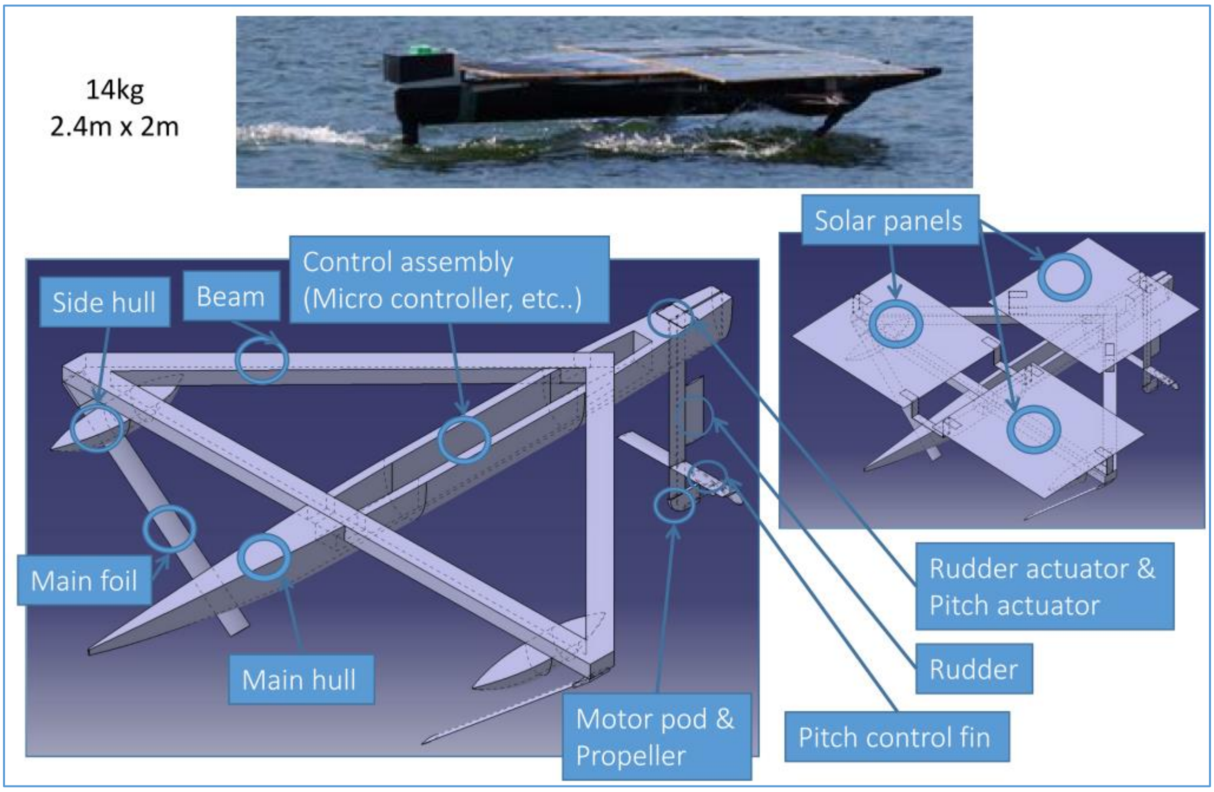


Figure 2 Overview of the 2014 designed, manufactured and raced boat. (Sutherland, Kamiyama, Aoyama, & Oizumi, 2015)

This experience was then written up as a paper (Sutherland et al., 2015) with the aim of identifying challenges with such a project and then attempt to provide description to potential solutions to these challenges.

As such the main activities completed by the team and the dates they were performed were compiled. These activities were then displayed on two visualization tools: The Knowledge Growth Curve as proposed in (Scheithauer, 2012) and four V-Model views proposed in (Scheithauer & Forsberg, 2013) (Figure 5).

1.2.2.A Understanding the 2014 Solar-Boat project using a Knowledge Growth Curve

The Knowledge Growth Curve attempts qualitatively to display the amount of knowledge the team has of the system they are attempting to develop. Figure 3 displays this for the

2014 Solar-Boat project. It shows when several key physical testing events occurred and how they impacted growth of team knowledge of the system. The project was characterized by a slow rate of initial knowledge acquisition taking up the bulk of the project time followed by a rapid acquisition of knowledge as the deadline neared. An "ideal" line has been drawn on Figure 3 indicating that obtaining knowledge about the system earlier in the project would be preferable.

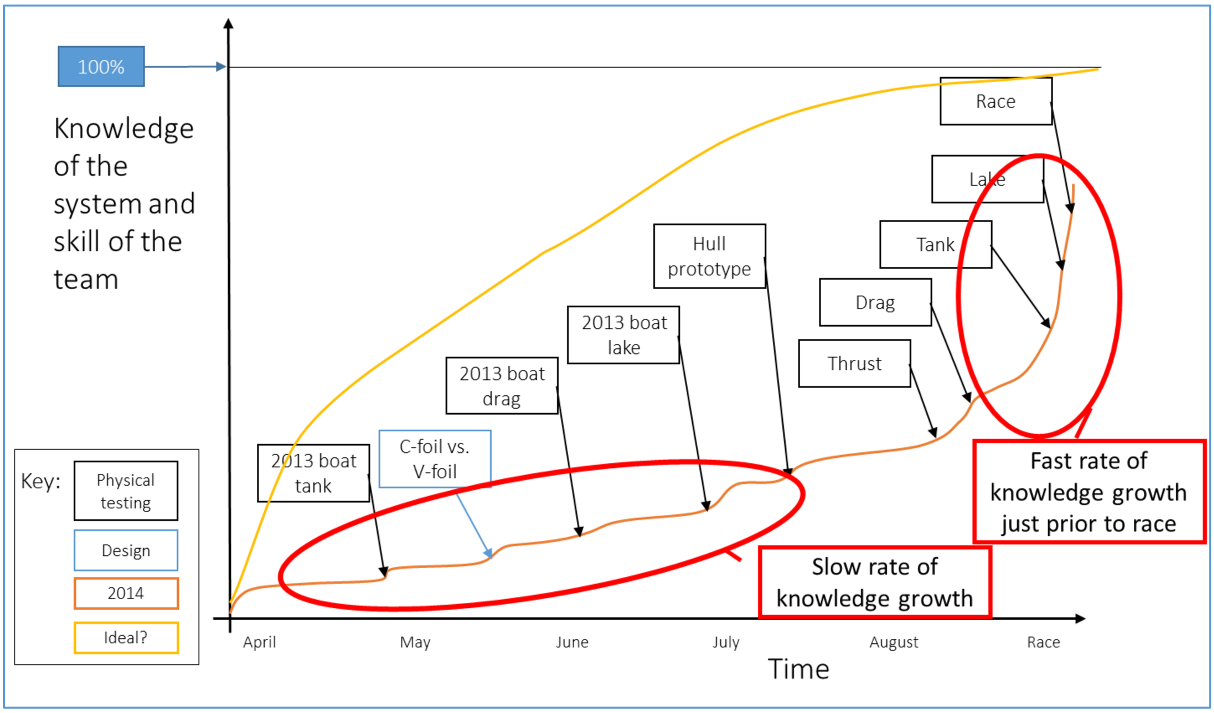


Figure 3 Knowledge Growth Curve of the Solar-Boat 2014 project. (Sutherland et al., 2015)

1.2.2.B Understanding the 2014 Solar-Boat project using V-Models

The Knowledge Growth Curve while a useful thought experiment is highly subjective and does not necessarily articulate the nature of individual problems on the project, such that they can be addressed. The V-Model however can be useful for displaying this information. The V-Model is well known and used within the Systems Engineering community. As per (INCOSE, 2011): "The Vee model is used to visualize the system engineering focus, particularly during the Concept and Development Stages. The Vee highlights the need to define verification plans during requirements development, the need for continuous validation with the stakeholders, and the importance of continuous risk and opportunity assessment".

As shown in Figure 4 the V-Model graphically displays the decomposition of a design into smaller sub system designs and component designs which are ultimately manufactured or procured and integrated into sub systems and assembled into a final finished product. While a single conceptual V (such as Figure 4) conveys significant level of information with regard to a Systems Engineering methodology of top down design followed by bottom up integration, if a project team is to use it to convey the information related to their particular project (in planning, ongoing development or retrospective review) the diagram can quickly become cluttered resulting in it being difficult to understand.

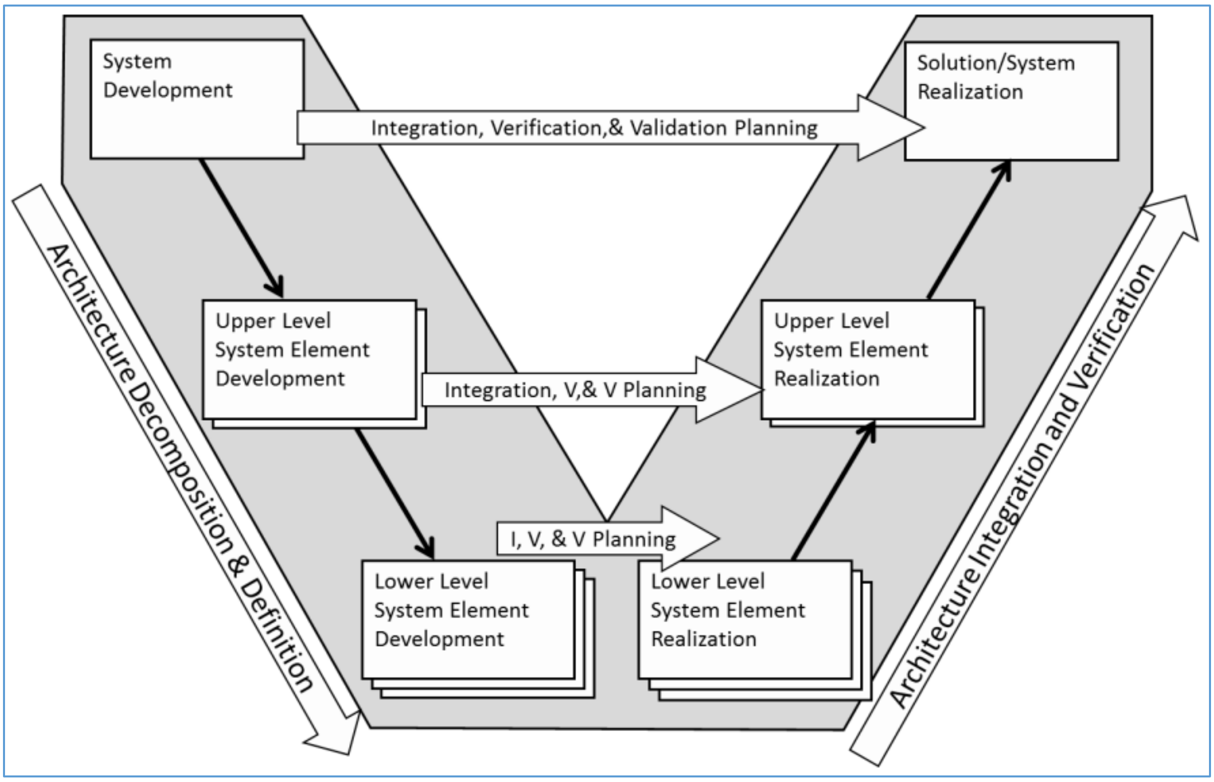


Figure 4 Simplified V-Model. (INCOSE, 2011)

In (Scheithauer & Forsberg, 2013) a set of consistent V-Model views are proposed (Figure 5) to allow project development teams to adequately describe their projects by way of the V-Model without adding unnecessary complexity to a single V. Their aim is to update the V-Model (from its roots in the 1980's independently being invented by NASA and a collaboration of Hal Mooz and Kevin Forsberg) such that there is no longer a requirement for excessive customization, and thus allow the V-Model to better cope with more modern approaches to systems development. The four views described by (Scheithauer & Forsberg, 2013) are described as: The Basic V (a standardized system of architectural decomposition), The Dynamic V (displays the iterations occurring over the product development), The Assurance V (Verification and Validation activities) and The Development V (hand-over of information within the system architecture), and. In (Sutherland et al., 2015) V-Model views proposed by (Scheithauer & Forsberg, 2013) were applied to 2014 Solar-Boat boat development retrospectively, to analyze the results and use this to generate an improved project development process.

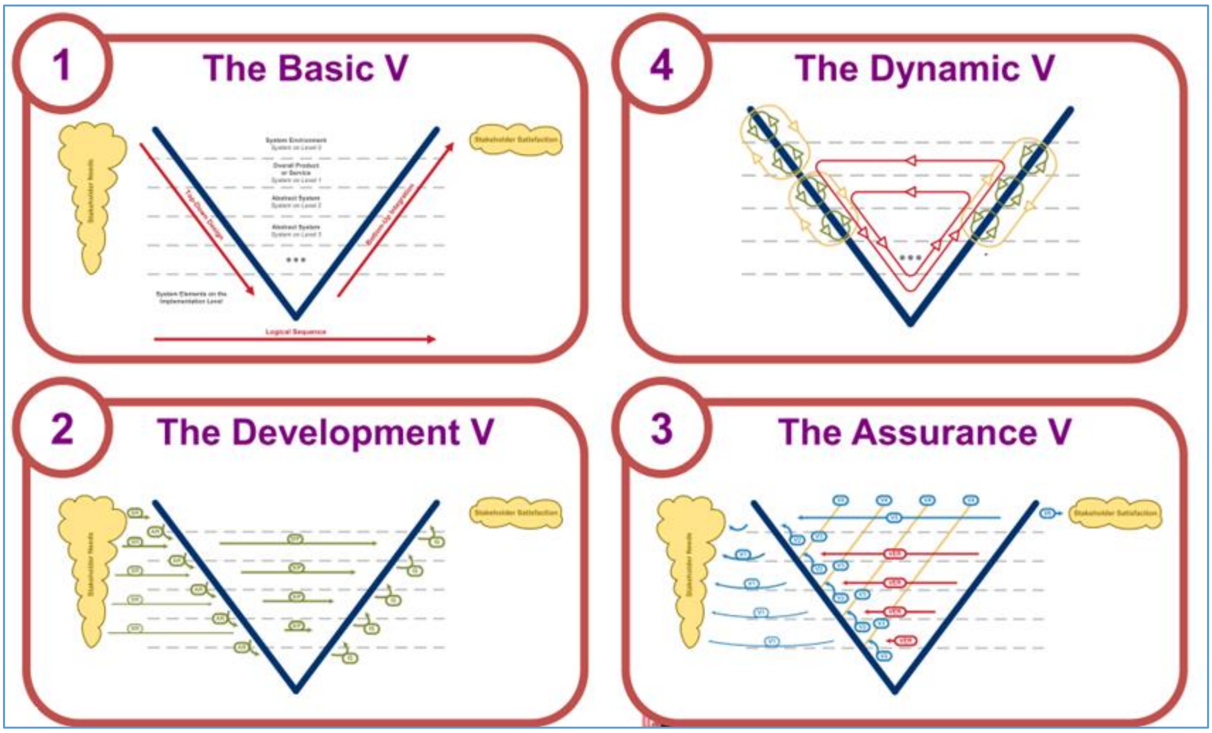


Figure 5 Four V-Model views. From: (Scheithauer & Forsberg, 2013)

1.2.2.B.a The Basic V

The Basic V (a standardized system of architectural decomposition) was used initially to review the project (Figure 6). The V is not symmetrical in time showing a long design phase followed by a short integration phase. These distinct phases correspond to the slow rate of knowledge acquisition and fast rate on Knowledge Growth Curve of Figure 3. Further interrogation of the diagram results in the identification of the problems shown in Table 2. The Basic Vs highlighting these problems are displayed in the Appendix.

Problem visualized	Basic V - Diagram
Long design process with unclear design target	Figure 6
Failure to deliver stakeholder satisfaction (lost race)	Figure 84
First testing of new system in real environment occurred too close to race	Figure 85
Major rework required of powertrain after integration testing	Figure 86
First testing of old system in real environment occurred too late	Figure 87

Table 2 Summary of Solar-Boat 2014 development problems displayed on Basic V diagrams

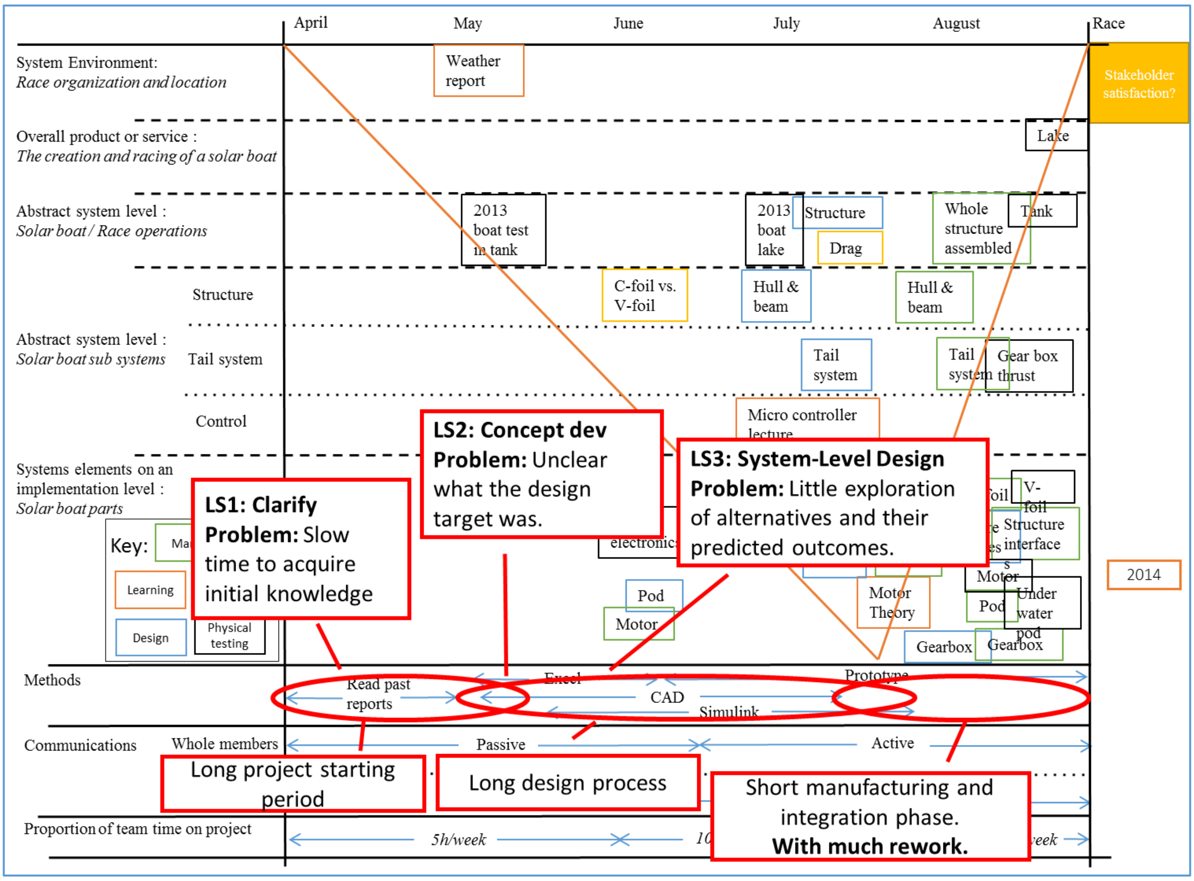


Figure 6 Basic V – Solar-Boat 2014. Indicating the long design process and short manufacturing phase of 2014 Solar-Boat development. (Sutherland et al., 2015)

1.2.2.B.b The Dynamic V

Plotting project activity using the Dynamic V (which displays the iterations occurring over the product development) reveals two interesting events in the project. Figure 7 shows where the use of modeling prevented a poor Systems-Level design choice (using C-Shaped hydrofoils) being manufactured. Figure 8 however shows an example where the power train system was not sufficiently modelled and understood during design resulting in significant rework being required. Figure 88 in Appendix shows how major rework of the yaw control system was needed. This was only detected once the entire system had been integrated. This is summarized in Table 3.

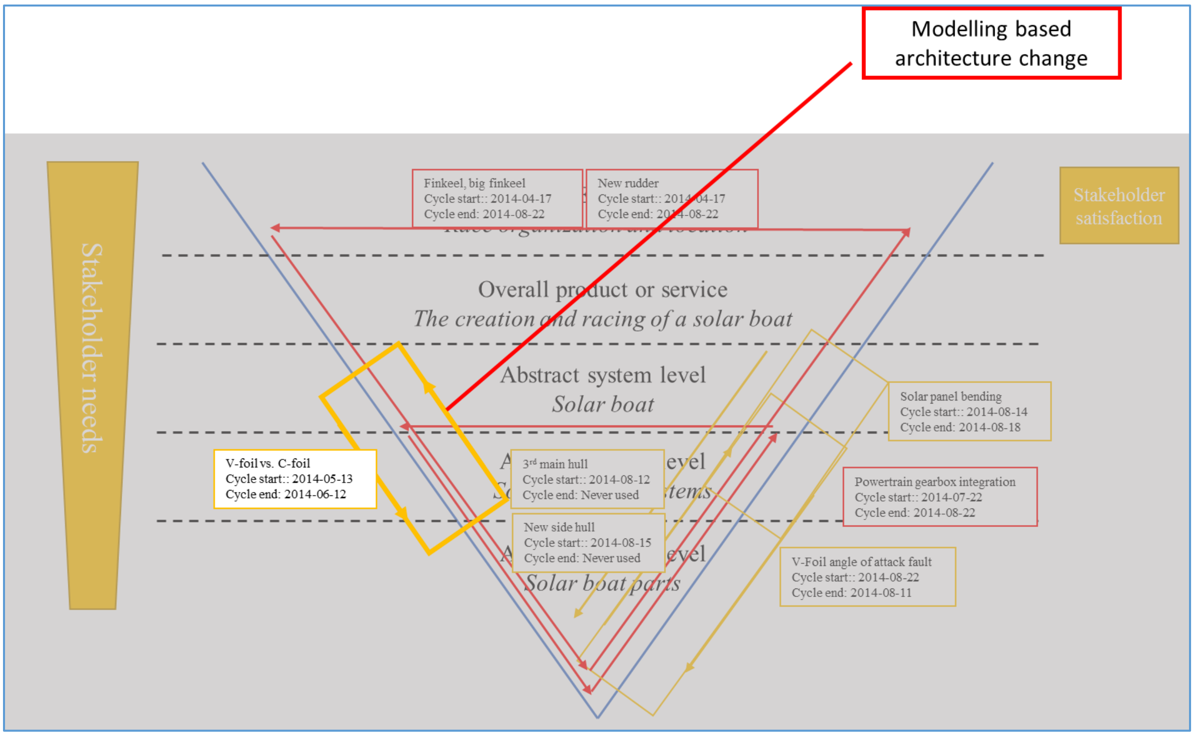


Figure 7 Dynamic V - Solar-Boat 2014. Indicating where modeling corrected a poor Systems-Level Design choice (using C shaped hydrofoils) prior to manufacturing. (Sutherland et al., 2015)

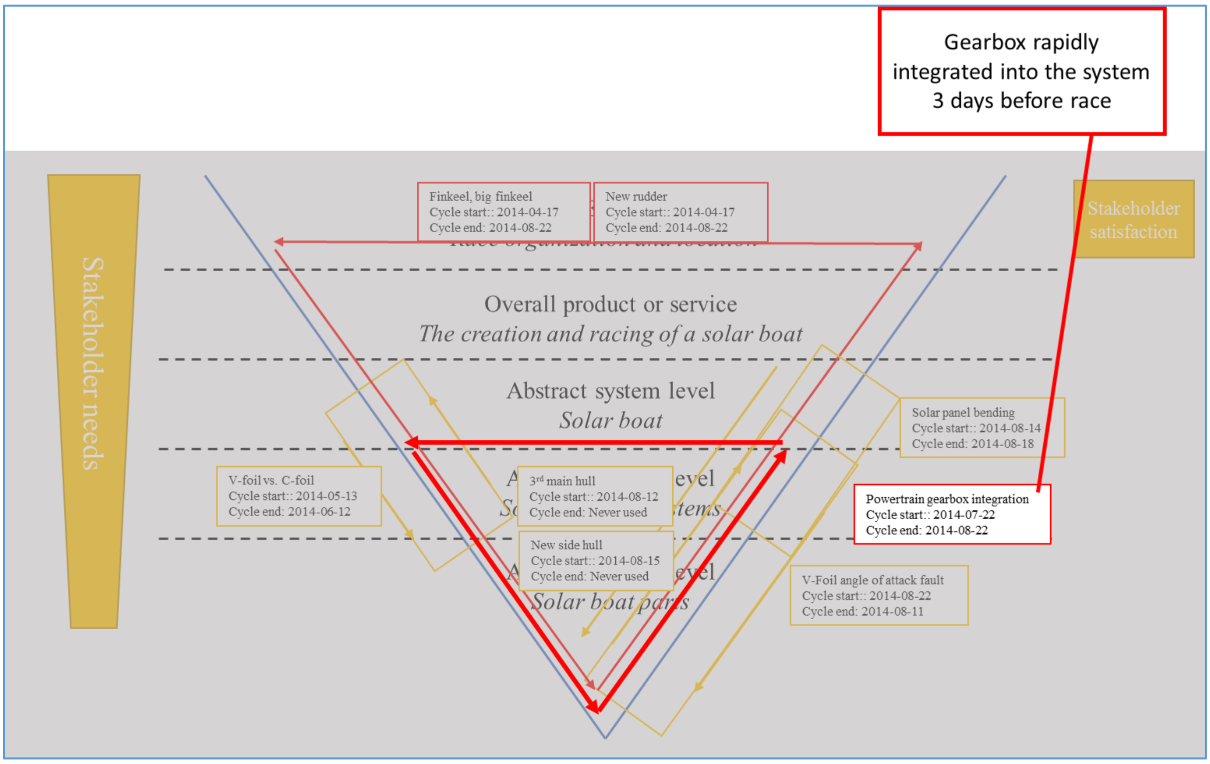


Figure 8 Dynamic V - Solar-Boat 2014. Indicating where a poor powertrain design choice was only spotted once they system was manufactured and integrated. (Sutherland et al., 2015)

Problem visualized	Basic V - Diagram
Problem detected on integration: Powertrain	Figure 8
Problem detected on integration: Yaw control	Figure 88

Table 3 Summary of Solar-Boat 2014 development problems displayed on Dynamic V diagrams

1.2.2.B.c The Assurance V

Displaying 2014 Solar-Boat project activity using the Assurance V (which displays the validation and verification activates). The problems identified and displayed on Assurance Vs in the Appendix are summarized in Table 4.

Problem visualized	Basic V - Diagram
High level simulation was done but not to the level to select parts	Figure 89
Lower level simulation was not verified	Figure 90
Experimental verification occurred too late to make design changes	Figure 91
Full system verification occurred very close to the race	Figure 92

Table 4 Summary of Solar-Boat 2014 development problems displayed on Assurance V diagrams

1.2.2.B.d The Development V

Displaying 2014 Solar-Boat project activity using the Development V (hand-over of information within the system architecture). The problems identified and displayed on Development Vs in the Appendix are summarized in Table 5.

Problem visualized	Basic V - Diagram
No validation of requirements	Figure 93
Unknown when design work products were complete and what form they took	Figure 94
Full systems integration occurred as soon as manufacturing stopped, little time for subsystem verification	Figure 95

Table 5 Summary of Solar-Boat 2014 development problems displayed on Assurance V diagrams

1.2.2.C 2014 Solar-Boat problems summarization and thesis goals

It is common in product development to define several lifecycle stages. Examples of such can readily be found in influential sources such as (Ulrich & Eppinger, 2011) and (INCOSE, 2015). Taking inspiration from these it is possible to define a set of Lifecycle Stages (LS) for the Solar-Boat project as listed in Table 6. Further assumed difficulties with each lifecycle stage and problems identified with 2014 project are listed. It is the aim of this thesis to attack the problems associated with the early stages of the lifecycle (LS1 Clarify, LS2 Concept development and LS3 System-Level Design) due to the likely positive impact these will have on downstream stages. For completeness an initial set of proposed solutions are provided; these will be explored in greater detail in this thesis.

Further it should be noted that the problems identified with the Solar-Boat project while at a small scale are representative of the problems of large-scale complexity and rapid development faced by industry.

Lifecycle Stage:	Ideal output:	Difficulties:	2014 identified problems:	Proposed solutions:	Difficulties with solutions:
LS1: Clarify	Well educated with past project knowledge.	What is important to learn? Overloaded with documents	Slow time to acquire initial knowledge (Figure 3)	Provide knowledge in models.	Language selection and integration.
LS2: Concept development	Detailed systems functionality and performance measures.	How to decompose high level project goals into low level system functionality?	Unclear what the design target was (Figure 6)	Trade-off analysis of multiple designs using models to simulate performance.	Consistent assessment of a reasonable number of alternatives. Numerical optimization vs. exploratory approaches.
LS3: System-Level Design	Systems Architecture and Formal Structure	How to map functional descriptions to formal design descriptions which deliver the functions?	Little exploration of alternatives and their predicted outcomes. Trial and error of building real systems (e.g. Figure 8).		
LS4: Detailed Design	Detailed specifications of components (e.g. 3D).	How to ensure consistency with System-Level design?	Little prediction of performance (e.g. Figure 8).		
LS5: Production, Test and Refinement	Race ready boat.	Debugging. Predicting effects of modifications.	Based on trial and error (e.g. Figure 8).		
LS6: Race	Winning the race.	Making repairs while still running the boat.	Lost race (Figure 84) due to faults which likely could have been predicted with modeling.		
LS7: Knowledge transfer	Knowledge is in an accessible form for next year.	Highlighting appropriate knowledge for the next year's team.			

Table 6 Solar-Boat 2014 lifecycle stages with identified problems and potential solutions of the 2014 project

2 Developing requirements for the proposed solution

2.1 Chapter purpose

Based on these identified problems of Section 6 an expansion of the high-level solutions and requirements for those solutions should be identified. These are described in the following two sections: Section 2.2 Knowledge Management – Provide knowledge in models and Section 2.3 System-Level Design – Complete trade-off analysis using models.

2.2 Knowledge Management – Provide knowledge in models

To address the issue Lifecycle Stage (LS) 1: Slow time to acquire initial knowledge (shown in Table 6), it is proposed that models can be used to provide knowledge to students starting up the project. Some requirements for this are shown in Figure 9. Models have been identified by the International Council of Systems Engineering (INCOSE) as an important enabler of better Systems Engineering and have listed in there 2025 vision statement with the setting of the Grand Challenge "Model-based systems engineering is a standard practice and is integrated with other modeling and simulation as well as digital enterprise functions" (INCOSE, 2014). Where Model-based systems engineering is defined as "The formalized application of modeling to support: System requirements, Analysis, Design and Verification & Validation" (INCOSE UK, 2015). As such the use of models is appropriate.

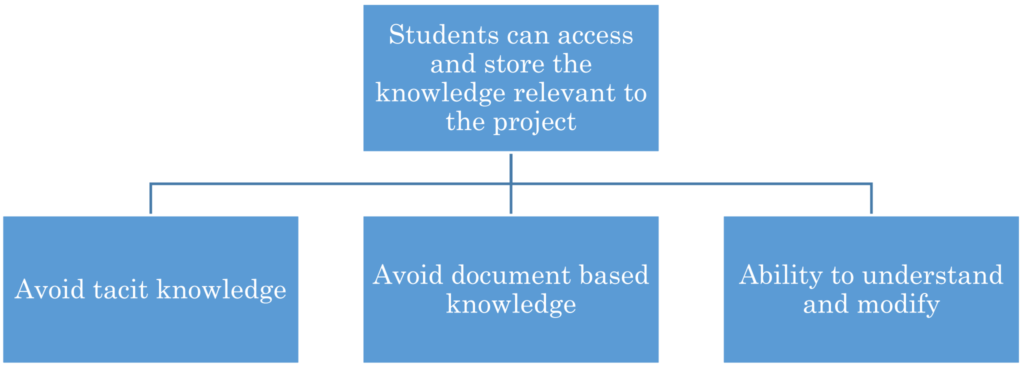


Figure 9 Basic requirements for model-based knowledge management

A survey of past Solar-Boat project knowledge locations was conducted with the results displayed in Table 7 for the first three lifecycle stages and Table 8 for the final four. It should be noted that knowledge for early and late lifecycle stages 1, 2, 5, 6 and 7 knowledge rests mainly in documents and the brains of past team members and students. Whereas in the middle of the project (lifecycle stages 3 and 4) knowledge rests in documents, models and physical components.

If as proposed a migration is made to model-based knowledge management, it is important to list difficulties of such an approach which include:

- Selecting appropriate modeling languages which are capable of holding the knowledge of interest but make this knowledge accessible:
  - In the engineering field there is a large proliferation of modeling languages competing to capture various knowledge types, how to select the right ones?
- Integration of multiple modeling languages together so different types of knowledge can be adequately captured:
  - If one language is not sufficient for modeling how to integrate multiple languages and minimize inconsistency?
- Keeping said models updated:
  - If a model no longer matches reality its value is lowered, its value might even fall to such levels where it is a hindrance and actively causes problems due to the inaccuracies it presents.

A summarization of the knowledge which should ideally be managed is shown in Table 9.

		Knowledge locations:						
		Documents (Word, Power Point…)	Brains		Models			Physically realized components
Life cycle stage:	Knowledge type:		Past team	Teachers	Excel	MATLAB	CAD	
LS1: Clarify	SolarBoat Project - Procedures and conventions	Y	Y	Y				
	Race - Race rules and constraints			Y				
	SolarBoat Project - Resources			Y				
LS2: Concept dev	SolarBoat Project - Project intension		Y					
	SolarBoat - Functional architecture (behavior)		Y					
	SolarBoat Project - Assessment scenarios		Y					
LS3: System-Level Design	SolarBoat - Systems Architecture	Y	Y		Y	Y	Y	
	SolarBoat - Subsystems	Y			Y			
	SolarBoat Project - Systems-Level Design Simulation models and results	Y			Y	Y		

Table 7 Knowledge locations of the first three lifecycle stages for past Solar-Boat projects

		Knowledge locations:						
		Documents (Word, Power Point...)	Brains		Models			Physically realized components
Life cycle stage:	Knowledge type:		Past team	Teachers	Excel	MATLAB	CAD	
LS4: Detail Design	SolarBoat - Detail Design						Y	Y
	SolarBoat Project - Detailed Design - Simulation models and results	Y				Y	Y	
LS5: Production, Test and Refinement	SolarBoat Project - Manufacturing procedures	Y						
	SolarBoat Project - Experiment and results	Y			Y			
LS6: Race	SolarBoat Project - Race Results	Y	Y	Y				
LS7: Knowledge transfer	SolarBoat Project – Salient points	Y						

Table 8 Knowledge locations of the last four lifecycle stages for past Solar-Boat projects

	Content:
Race:	Race rules
	Environmental inputs
SolarBoat project:	Project intension
	Resources
	Design processes
	Manufacturing processes
	Testing processes
SolarBoat:	Design (including alternative designs)
	Predicted performance
	Tested performance

Table 9 Knowledge to be managed

2.3 System-Level Design – Complete trade-off analysis using models

To address the issue of Lifecycle Stages (LS) 2 and 3, of being unclear as to what the design target was and little exploration of alternatives and their predicted outcomes (shown in Table 6), it is proposed that a logical trade study methodology be used to assess multiple designs using models to simulate performance. Figure 10 presents important requirements for a trade study methodology. It stresses the need for alternative designs to be assessed utilizing the same approach such that alternatives are all treated fairly.

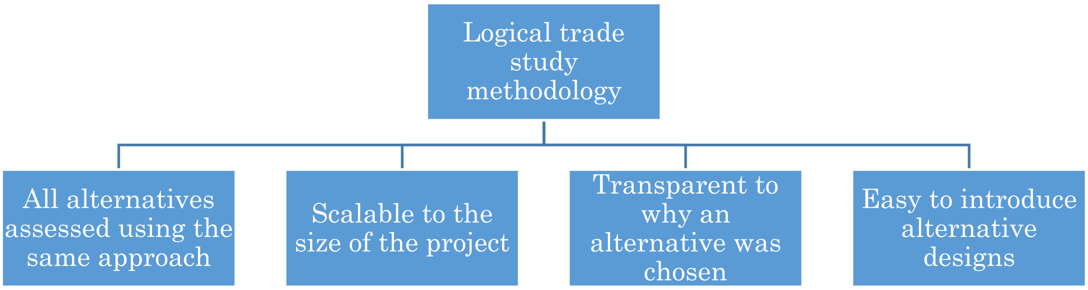


Figure 10 Requirements for a trade study methodology

Logically this leads to the creation of a basic flow for the assessment of designs as shown in Figure 11. Which raises the following difficulties:

- How to synthesize an assessment procedure? What assessment criteria is important to all designs?
- How to synthesize alternative designs?
- How to predict each designs performance?
- How to analyze the performance based on the assessment procedure?

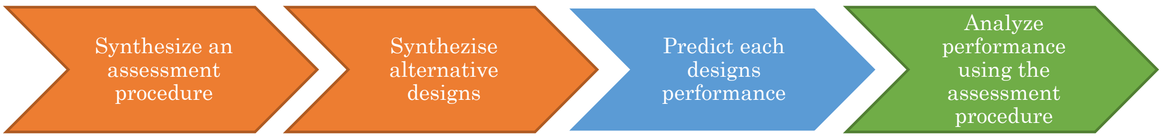


Figure 11 Key steps to completing a trade study. (Orange: Synthesize, Blue: Prediction and Green: Analysis)

Table 12 applies this to Lifecycle Stages 2 (Concept development) and 3 (System-Level Design) such an approach must be able to support the "identification of essential problems" (Concept development) and "definition of the product architecture and the decomposition of the product into subsystems and components" (System-Level Design). Expanding upon this it is implied then:

- Concept development requires a methodical decomposition of the problem the Solar-Boat is expected to solve
- System-Level Design requires the assessment competing systems composed of subsystems rather than parameter variation

These stages are chosen as at the early stage there is opportunity for a larger variation of designs, which once chosen will be costly to correct. Figure 12 displays some past examples of architecture variants and Table 10 provides an example concept selection table which might produce such architecture variants. In such a table a subsystem is defined as a column of the table, an alternative implementation can be selected as a row from the column.

Figure 12 Alternative Solar-Boat physical architectures

Solar energy to electrical energy	Electrical energy to mechanical rotation energy	Increase torque reduce speed of mechanical rotation energy	Mechanical energy to thrust	Provide buoyancy	Change pitch	Change roll	Change yaw	Reduce drag and move components further from water	Reduce drag at different speeds	Reduce wave damage
Solar panel (3)(4)(5)	Electric motor (3)(4)(5)	Gearbox (3)(4)(5)	Water propeller (3)(4)(5)	Mono hull	Actively controlled stabilization (3)(4)	Actively controlled stabilization (3)	Simple rudder (3)(4)	Surface piercing – VFoil (4)(5)	Planing hull (4)(5)	Wave-piercing bow (3)(4)(5)
		No gearbox	Air propeller	Catamaran (3)	No active control (5)	No active control (4)(5)	Thrust vectoring (5)	Surface piercing - CFoil	Displacement hull (3)	Standard bow
			Jet pump	Trimaran (4)(5)				Fully submerged - TFoil (3)		
								Fully submerged - LFoil		
								No hydrofoil		

Table 10 Example Solar-Boat concept selection table ((3) = 2013 Solar-Boat, (4) = 2014 Solar-Boat and (5) = 2015 Solar-Boat)

To provide a clear summary the problems associated with completing a trade-off analysis using models are listed as follows:

- Providing a framework which can assess all alternative designs
- Being able to compare a reasonable number of design alternatives
- Balancing the use of numerical optimization with exploratory approaches enabling human designers to obtain a better understanding of the trade space:
    - While single point global optimization can be useful, it has limitations in that it does not necessarily provide a lot of understanding as to why a particular system is better than another, is highly dependent on the value function used and if exhaustive searches are to be used is very computationally expensive O(n!). If a designer is interested in more of the -ilities: adaptability, maintainability, qualitative issues, and resilience these can be difficult to contain in such

optimization schemes. Understanding is very important even for non-novice engineers, particularly for capacity to adapt design as requirements change (or e.g. rules or course change in the race). As such exploratory design must not be ruled out.

2.4 Aim of this thesis

Given these identified problems with Solar-Boat development described in Section 1.2.2 and the requirements of the proposed solutions the aim this thesis becomes to propose tools and methodologies to help students:

- Manage project knowledge
- Explore concept designs

3 Literature Review and State of the Art

3.1 Chapter purpose

This chapter aims to provide the reader with:

- A description of current relevant research and state-of-the-art approaches and tools to the problems identified and described in Section 6 and requirements described in Sections 2
- Highlights of the gaps in the current literature which this thesis aims to resolve.

3.2 Trade studies

3.2.1 Existing literature

The International Council of Systems Engineering (INCOSE) provides by way of (INCOSE, 2015) and (SEBoK, 2015) a decision management process which is intended for trade studies. The recursive loop for each decision is shown in Figure 13.

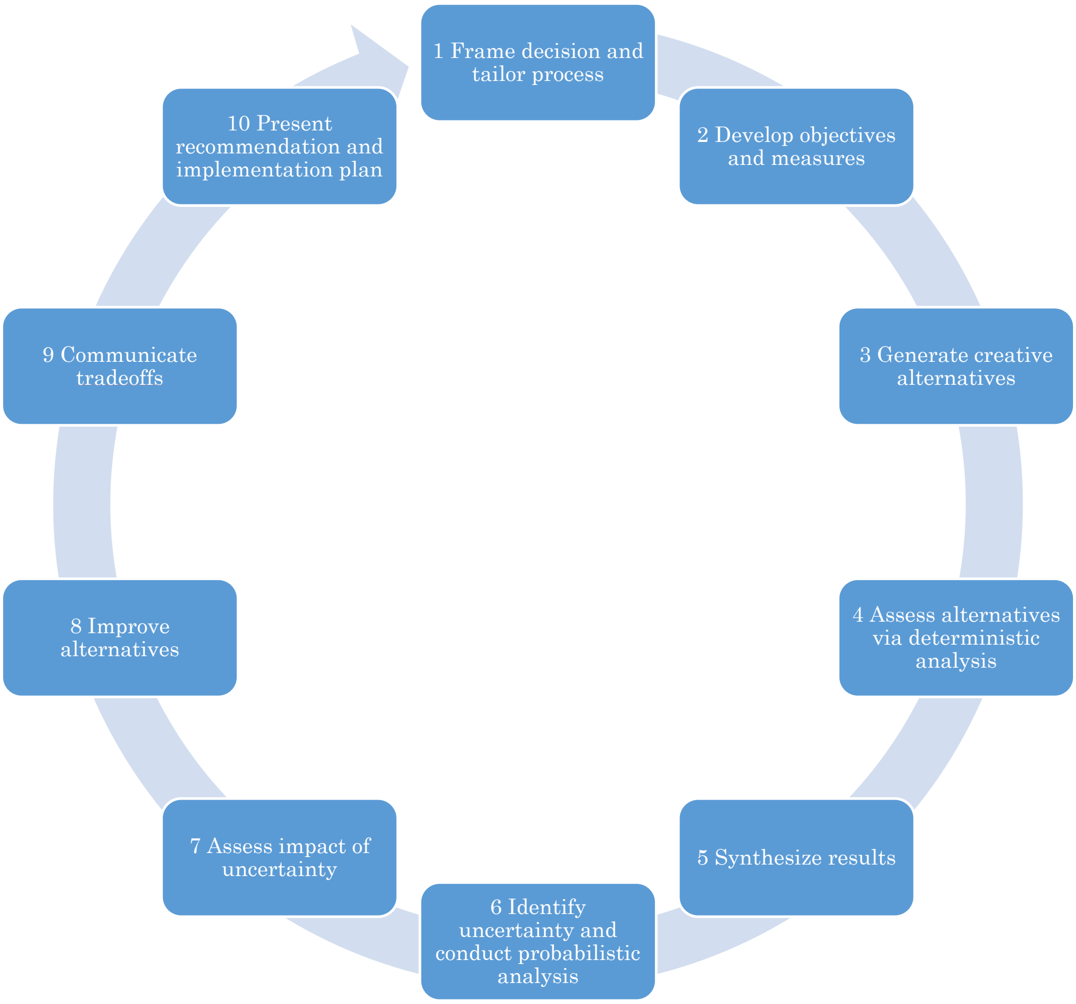


Figure 13 INCOSE Decision Management Process. Adapted from (INCOSE, 2015) and (SEBoK, 2015)

Two examples of the implementation of this process are provided in literature (Cilli & Parnell, 2014) and (Edwards et al., 2015). Where the former aims to create a baseline for future exploration of trade studies by way of integrating decision analysis best practices with systems engineering while demonstrating this with a fictional UAV case study. The latter aimed to explore the trade space for a new US Army ground fighting vehicle where a huge number of system configurations 10^{20+} where compared, attempting to balance Performance; Acquisition Cost; Time to Complete; Operation & Service Cost; Spiral Upgrades. Table 11 provides a comparison of the first five steps of the implementation of the INCOSE Decision Management Process by these two papers.

Decision stage	**(Cilli & Parnell, 2014)**	**(Edwards et al., 2015)**
1 Frame decision and tailor process	Multi Objective Decision Analysis (MODA) is appropriate when there is no single measure to use to assess performance.	Compare designs to the following objectives: Performance; Acquisition Cost; Time to Complete; Operation & Service Cost; Spiral Upgrades.
2 Develop objectives and measures	Decompose fundamental objectives to low level measures. Define performance measures to value mapping.	Reduce hundreds of requirements to a smaller set of Functional Objectives. Define Systems Architecture from Work Breakdown Structure. Map subsystems to Functional Objectives.
3 Generate creative alternatives	<u>Define a common Systems Architecture</u> with alternative subsystem implementations. Non exhaustive search.	<u>Define a common Systems Architecture</u> with alternative subsystem implementations. Use generic algorithm to vary subsystems in Product Structure locations and simulate performance.
4 Assess alternatives via deterministic analysis	Subject Matter Experts to provide scores for each design by however they wish (e.g. test data or simulation).	Simulation based on rules provided by Subject Matter Experts in mapping subsystems to Functional Objectives and other subsystems.
5 Synthesize results	Combine performance scores by way of weighted sum of value (MODA).	Identify the configurations which are on the Pareto Frontier for any of the five objectives.

Table 11 Comparison of the implementation of first five stages of the INCOSE Decision Management Process implemented by (Cilli & Parnell, 2014) and (Edwards et al., 2015)

The salient points underlined in Table 11 are discussed as follows:

- **2 Develop objectives and measures:** (Cilli & Parnell, 2014) provide a fundamental objectives hierarchy and encourage functional decomposition, but no clear method is provided; (Edwards et al., 2015) take an existing set of hundreds of requirements and reduce it to a smaller set of functional objectives.
- **3 Generate creative alternatives:** In both papers a common Systems Architecture is defined such that individual implementations of various subsystems can be varied in each position of the Systems Architecture. In (Cilli & Parnell, 2014) it is not clear how this Systems Architecture was defined and in (Edwards et al., 2015) the Systems Architecture is defined from the Work Breakdown Structure which itself is defined from one which is predefined for a ground vehicle in MIL-STD-881C (US DoD, 2011).
- **4 Assess alternatives via deterministic analysis:** Here the papers differ. In (Cilli & Parnell, 2014) scores are assigned each performance measure on each alternative design by a subject matter expert, but for (Edwards et al., 2015) simulation is performed. The simulation models were created however by Subject Matter Experts in mapping subsystems to Functional Objectives and other subsystems rather than building a model whereby such interactions are inherent to the construction of the model.

3.2.2 Application to Solar-Boat project

In reviewing the first five stages of the INCOSE Decision Management Process and how they could be applied to Solar-Boat project Table 12 is created. Here requirements for the Solar-Boat project are listed against each decision stage along with issues which need to be addressed. Discussion of the sleight points underlined in Table 12 is as follows:

- **2 Develop objectives and measures:** A functional architecture is required such that assessment criteria can be defined. However unlike (Edwards et al., 2015) no large requirements document is provided ahead of time.
- **3 Generate creative alternatives:** A Systems Architecture must be defined such that alternative implementations of the designs can be obtained. However unlike (Edwards et al., 2015) no standard Systems Architecture is predefined and available. While MIL-STD-881C (US DoD, 2011) provides multiple detailed work breakdown structures for standard military systems its description for the identification of major subsystems and functional requirements for entirely new systems is brief and does not indicate how each sub system interact with each other.
- **4 Assess alternatives via deterministic analysis:** A numerical model is required to assess performance, however unlike (Edwards et al., 2015) there are no subject matter experts to define how to connect such a simulation model together.

Decision stage	**Requirements for SolarBoat project**	**Issues needing to be addressed**
1 Frame decision and tailor process	Define the decision to make	What is a Concept Design and Systems-Level design
2 Develop objectives and measures	Identify Functional Architecture of SolarBoat	No requirements document to work from
	Identify how to assess SolarBoat	What can be simulated?
3 Generate creative alternatives	Develop a Systems Architecture	No MIL-STD-881C type document to provide a Systems Architecture
	Populate the Systems Architecture with subsystem alternatives	Too many alternatives?
4 Assess alternatives via deterministic analysis	Build numerical model	Unsure what to model
	Simulate to determine each alternatives performance	Each alternative needs to be assessed in the same way
5 Synthesize results	Compare the simulation results of the alternatives	Extract a range of simulation results and compare

Table 12 Reviewing the first five stages of the INCOSE Decision Management Process if they were applied to Solar-Boat project

3.3 Modeling languages

Given the clear difference in activities in steps 2, 3 of the INCOSE Decision Management Process which is largely about clarifying the problems space, determining a functional architecture and subsequently a systems architecture vs. step 4 where numerical simulation is required it is appropriate to split the work there and survey how these tasks could be undertaken. With the first to be undertake by a systems modeling language and the latter by numerical simulation. As shown in Table 13.

<table>
<tr><th>Decision stage</th><th>Requirements for SolarBoat project</th><th>Modeling approach</th></tr>
<tr><td>1 Frame decision and tailor process</td><td>Define the decision to make</td><td></td></tr>
<tr><td rowspan="2">2 Develop objectives and measures</td><td>Identify Functional Architecture of SolarBoat</td><td rowspan="3">Systems modeling</td></tr>
<tr><td>Identify how to assess SolarBoat</td></tr>
<tr><td rowspan="2">3 Generate creative alternatives</td><td>Develop various common models for SolarBoat alternatives</td></tr>
<tr><td>Populate the common alternatives with subsystem alternatives</td><td rowspan="3">Multi domain numerical simulation</td></tr>
<tr><td rowspan="2">4 Assess alternatives via deterministic analysis</td><td>Build numerical model</td></tr>
<tr><td>Simulate to determine each alternatives performance</td></tr>
<tr><td>5 Synthesize results</td><td>Compare the simulation results of the alternatives</td><td></td></tr>
</table>

Table 13 Modeling approaches for different decision stages

3.3.1 Systems modeling languages

A systems modeling language aims to document the function, structure and behavior of a system to improve analysis the enable more detailed design (Grobshtein, Perelman, Safra, & Dori, 2007), both Object-Process Methodology (OPM) and Systems Modeling Language (SysML) are examples of systems modeling languages (Grobshtein et al., 2007). Both are available as published standards: OPM (ISO, 2015) and SysML (OMG, 2015). While their aims are similar, they take quite different approaches to modeling systems, where SysML has 9 different diagram types (Figure 14) each to display a different aspect to the system (and subsequently manage complexity by limits the number of aspects seen at any time), OPM has a single diagram type known as an Object Process Diagram (OPD) (and associated automatically generated text known as Object Process Language (OPL)) and manages complexity by detail decomposition (Dov Dori, 2015). While SysML is popular within the

Systems Engineering community, and enables multiple Model Based Systems Engineering methodologies (Estefan, 2007) it has been found that while SysML is good at handling detail OPM is better for enabling a holistic understanding of the system and its environment without the need to understand multiple diagram types (Grobshtein et al., 2007). For this reason OPM was selected as the systems modeling language.

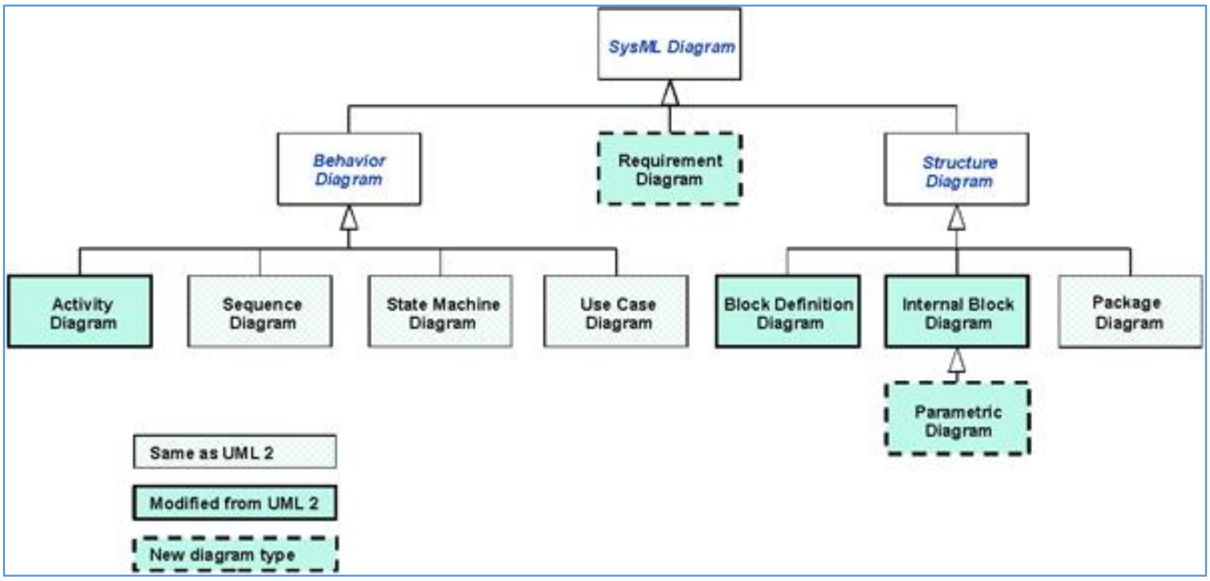


Figure 14 SysML diagram types and comparison to UML (OMG, 2016)

3.3.1.A Object Process Methodology (OPM)

As the reader might not be familiar with OPM a simple example is provided in Figure 15. Here the two key building blocks of OPM are presented: A stateful object (green box) and a process (blue oval). This combined with relations between them are considered sufficient to enable OPM to be a universal ontology as described in (Dov Dori, 2015).

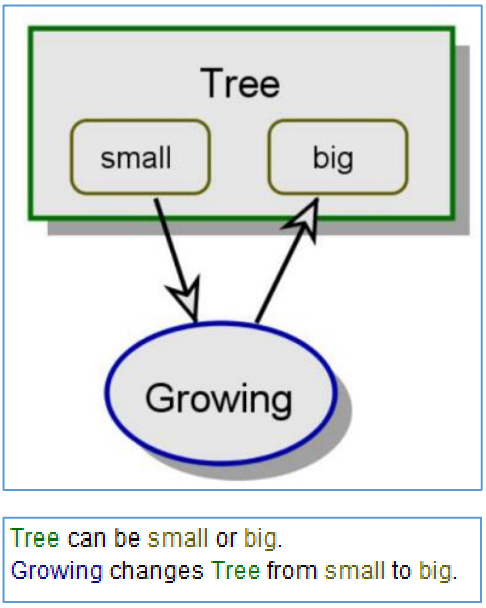


Figure 15 OPM's two key building blocks: Stateful object and a Process (Dov Dori, 2015)

Two dense diagrams displaying both OPD and OPL of much more of the language specification are provided in Figure 16 (structural relations, aiming to allow the modeling of the structure of the system of interest) and Figure 17 (dynamic relations, aiming to model the behavior of the system resulting in some change to the system over a time period). The following key points should be noted about OPM diagrams:

- Diagrams must be consistent with each other only in that they cannot display information which contradicts each other. As such it is possible to produce two diagrams which of the same part of the system even if one contains a large amount of detail and the other suppresses a large amount of detail.
- An unlimited number of diagrams can be created to provide different views of the system so long as they are consistent with each other.
- Hierarchy (of processes and objects) is the preferred method of handling complexity.

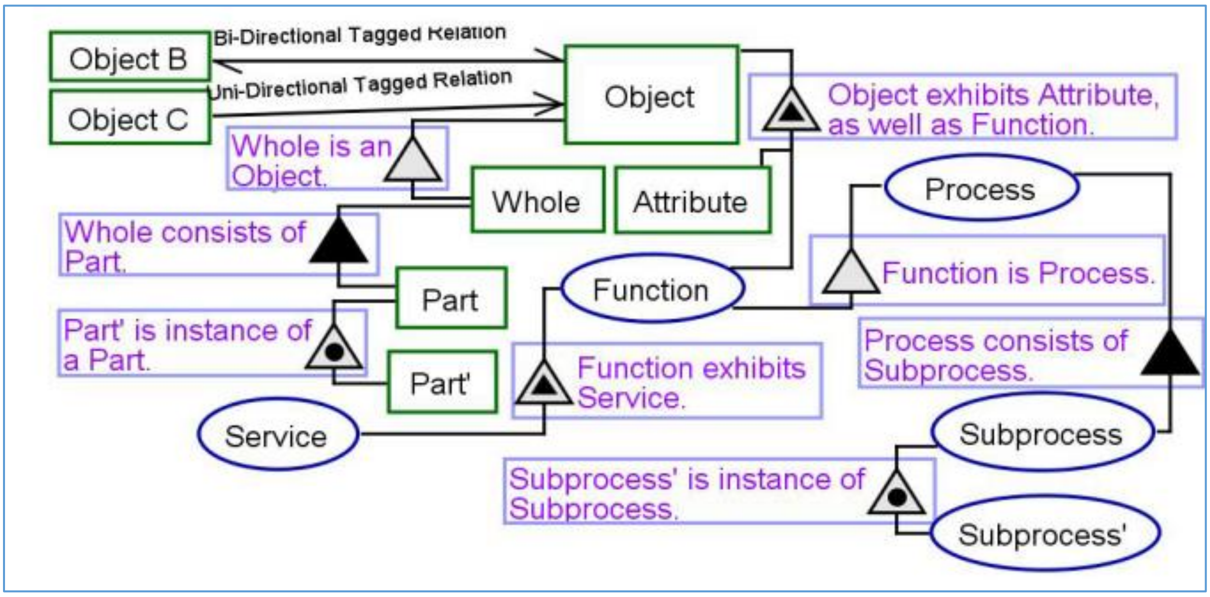


Figure 16 OPM Notation – Objects, Processes and Structural Relations (Mordecai & Dori, 2015)

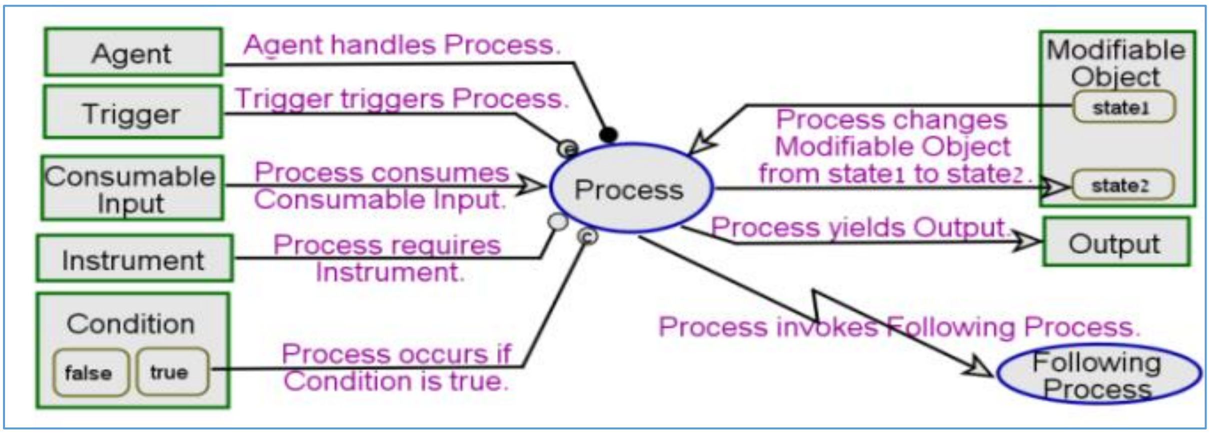


Figure 17 OPM Notation – Procedural Relations and State Dynamics (Mordecai & Dori, 2015)

Figure 18 is a simple example of modeling a Solar-Boat in OPM, where it is shown to consist of a Hull and Solar panels (structure), enables the process "Driving" (behavior). The "Driving" process affects the stateful attribute of the SolarBoat, Speed while consuming Solar Insolation. To manage the amount of detail displayed, hierarchy is employed, as such while a diagram such as Figure 18 does provide a compact description of a generic Solar-Boat, further detail can be revealed by the decomposition of the objects and processes in the model.

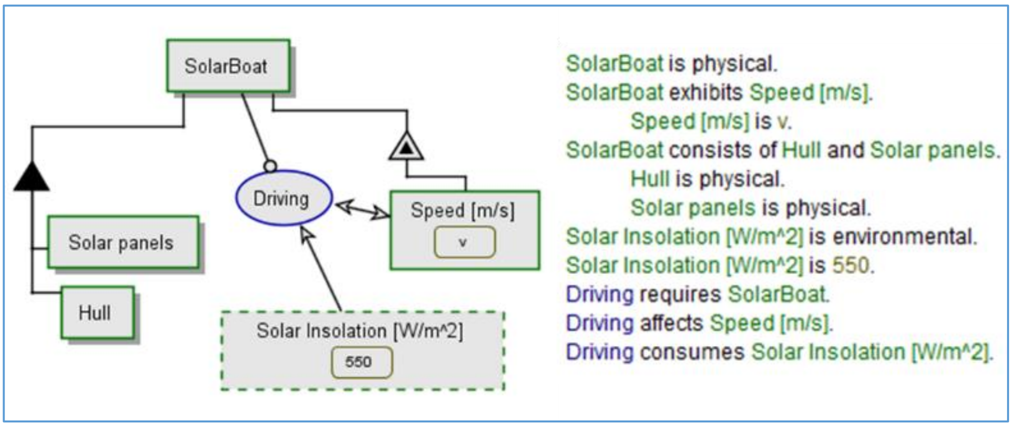


Figure 18 OPM – A simple Solar-Boat model

3.3.1.A Developing OPM models

Within the OPM ISO standard (ISO, 2015) there is only brief description of the methodological aspects of OPM model creation and the main focus is on the syntax of the diagrams and accompanying text based language. The following salient points are noted from this reference:

- The combination of OPM system structure and behavior aspects enables the system to perform a function which should deliver value to some stakeholder.
- The systems functional value should be identified as a single primary process first before structural aspects are introduced to enable the delivery of the functions.
- Functions should clearly be distinguished from behaviors and structures to deliver the behaviors and therefore functions. In that for a given system multiple alternative behaviors might be able to deliver the same functions. A concrete example is given whereby the function is to cross a river alternative structures and associated behaviors include a bridge and a ferry boat.

In (Crawley, Cameron, & Selva, 2015) broader techniques for the analysis, synthesis and assessment of System Architectures are provided making extensive use of OPM. Here the generalization of a function in OPM is made as a pair of a process and an object (stateful or otherwise) with the process creating, consuming or affecting the said object as shown in Figure 19. While the form (another object) which delivers the function is attached to the process as an instrument.

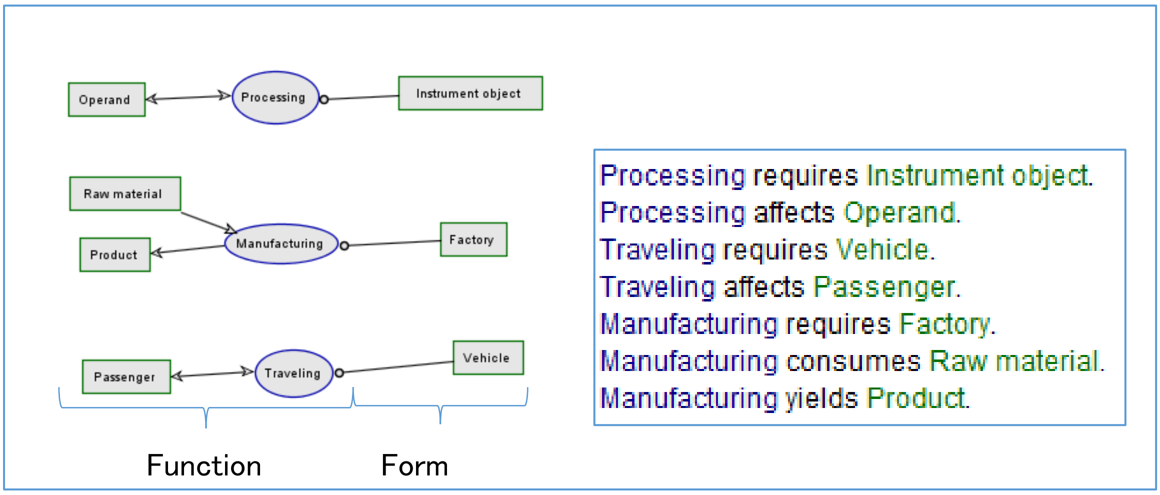


Figure 19 Generalized description of functions and form in OPM adapted and expanded from (Crawley et al., 2015)

To enable the exploration of alternative this is then expanded into a chain of increasing detail (Figure 20). Here intent (similar to functions in (ISO, 2015)) indicates a form independent function which the system must complete to deliver value to its stakeholders while function (similar to behaviors in (ISO, 2015)) indicate a specific way to implement the intent.

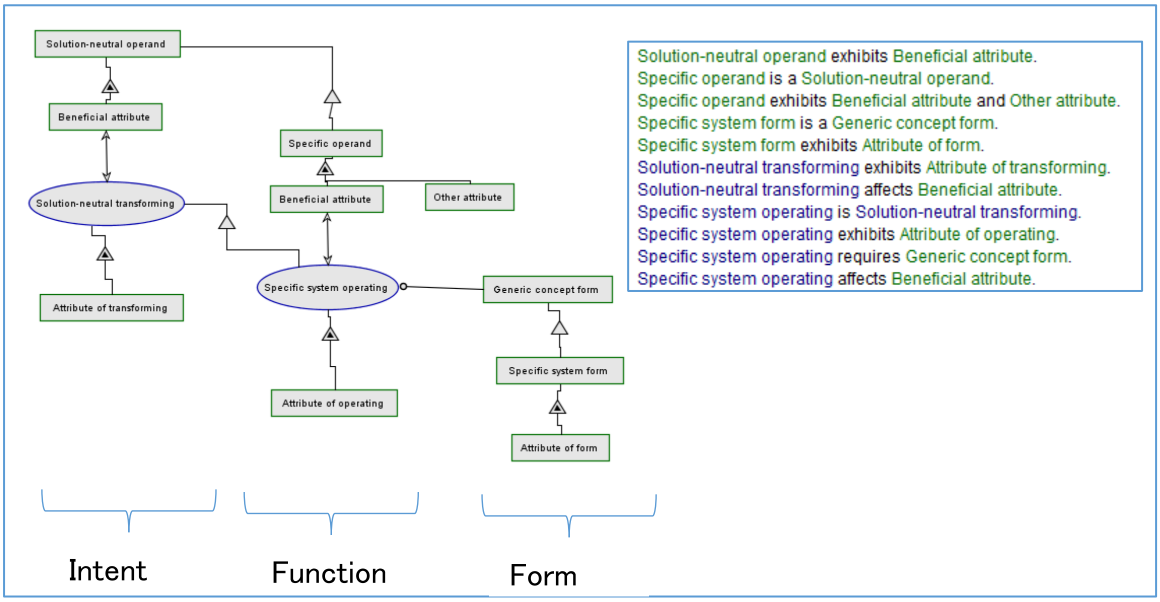


Figure 20 From an intent function description describing a specific function implementation and form which enables the specific function to be implemented. Adapted from (Crawley et al., 2015)

From this (Crawley et al., 2015) then define groupings of these paring to develop definitions of Functional architecture, System architecture and Formal structure as shown in Figure 21.

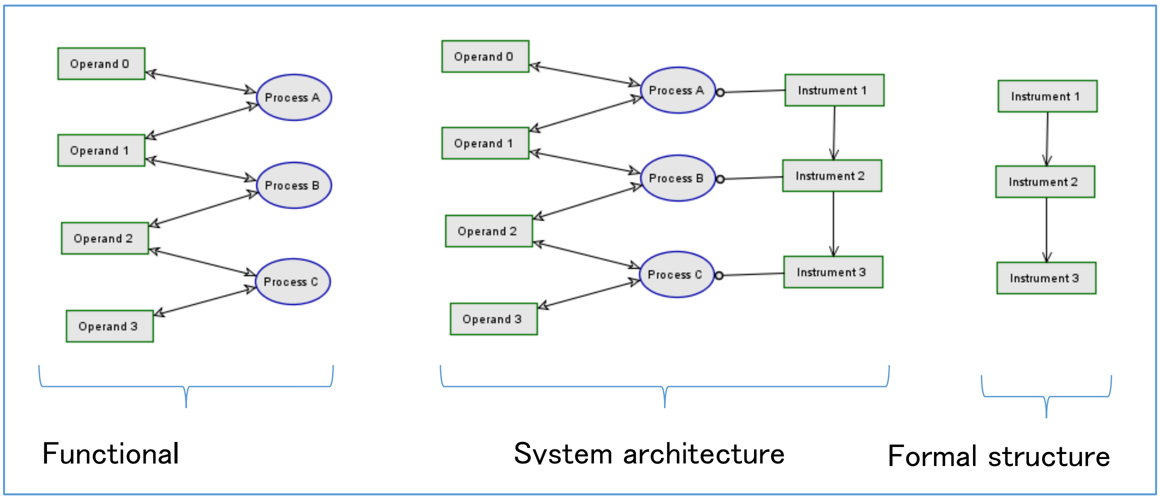


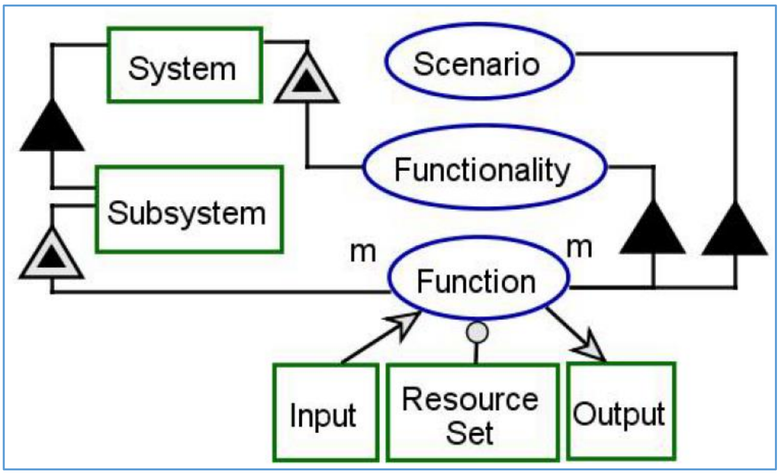
Figure 21 OPM based definitions of architecture types. Adapted from (Crawley et al., 2015)

While (Crawley et al., 2015) use different terminology to (ISO, 2015) they follow a similar methodological approach of initially identifying object modification and process pairs which results in value being generated for beneficiaries and only later assigning specific forms to deliver these processes.

In (Mordecai & Dori, 2015) further details of an approach to model systems of multiple competing variants in OPM is provided with a system design pattern (Figure 22). To paraphrase the description provided by (Mordecai & Dori, 2015):

"**System** has some high-level emergent **Functionality** … [which is] … abstract and emergent … its constituent **Functions** are actual operations of the **Subsystems** ... Unlike **Functionality**, which relates to the functional system aspect, **Scenario** relates to the behavioral aspect … a set of **Functions**, aimed at accomplishing some goal or objective … distinction between **Functionality** and **Scenario** provides for modeling two system aspects—the functional and the procedural—to coexist within the same model. This way, we can model emergence—emergent traits of a system as its functionality—a specific combination of ordered functions whose value to some beneficiary is greater than the sum of the values of the individual functions."

This research is of interest to this project as it explicitly attempts to model a system in OPM where it is recognized the system must perform under different scenarios which are made up of multiple system functions. While at the same time it is shown that whole system functionality is made up of individual functions which are delivered from distinct sub systems.



System consists of several subsystems;
System exhibits Functionality;
Scenario and Functionality consist of several Functions, exhibited by Subsystems.
Function receives Input, uses a Resource Set, and generates Output.

Figure 22 Basic system design pattern from (Mordecai & Dori, 2015)

3.3.2 Multi domain numerical modeling languages

As shown in Table 13 there is an identified need for a multi domain numerical modeling language so that alternative designs can have their performance assessed. It is important to use one which allows the modeling of a Solar-Boat which has electrical, mechanical and structural components all of which are associated with different physical phenomena. Further there is the need to assess each design under a range of criteria (which might well change). This makes Modelica (Modelica Association, 2012) a good candidate due to its acausal equation object-oriented nature. This differs from casual modeling using MATLAB/Simulink which most engineers are familiar with.

Further additional benefits of using Modelica include:

- Existing libraries of components
- Ability to define common System Architectures where components can be swapped in and out of so long as they have the same interface
- Models are stored in a text file making programmatic manipulation and version control simpler
- Models are built out of a hierarchy of other models

3.3.2.A Defining Modelica model architectures

Reviewing literature on the creation of Modelica model architectures again yields little past research. In (Chapon, Bouchez, & France, 2009) it discusses the requirements for a system to link UML architectural descriptions to Modelica models, but there is no description of how the Modelica models are created. In (Batteh & Tiller, 2009; Tiller, Bowles, & Dempsey, 2003) to automotive model architectures are presented, but how they are created is not discussed. Indicating the need to demonstrate how to create Modelica model architectures logically.

3.3.3 Integrating systems modeling languages with numerical modeling languages

A short literature review was conducted to find existing work attempting to link systems modeling languages to numerical modeling which is summarized in Table 14. While SysML has had a lot of attention including a detailed Modelica linking standard (OMG, 2010) less research has been applied to OPM. OPM has been used to create executable simulations before in the form of stateful transitions whether it be based on petri nets (Wang, Agarwal, & Dagli, 2015) or the OPM model itself (Yaroker, Perelman, & Dori, 2013). While past research with continuous numerical simulation (Bolshchikov, Renick, Mazor, Somekh, & Dori, 2011; Dov Dori, Renick, & Wengrowicz, 2015) made use of MATLAB/Simulink to run the simulations for the purpose of gaining greater understanding of existing descriptive models (they were not trying to actively design a new system and assess its performance with numerical modeling). MATLAB/Simulink were selected in these studies as the authors wished to focus on modelling the behavior rather than the objects which enable the behavior.

		Numerical modeling languages:	
		Modelica	**MATLAB/Simulink**
Systems modeling language:	**OPM**	No literature found	(Bolshchikov, Renick, Mazor, Somekh, & Dori, 2011)
	SysML	(OMG, 2010)	(Qamar, During, & Wikander, 2009)

Table 14 A review of literature linking systems modeling languages to numerical modeling

4 Proposed tools and methodologies for Knowledge Management and System-Level Design

4.1 Chapter purpose

This chapter aims to provide the reader with detailed descriptions of:

- Particular tools and methodologies for design exploration with the intension of selecting a Solar-Boat System-Level Design. Consisting of:
  1. Clarifying the Solar-Boat project
  2. Identifying SolarBoat required processes and subsystems
  3. Creating a comparison criteria for alternative boats
  4. Generating alternative architectures for assessment
  5. Comparing alternatives and selecting
- Particular tools and methodologies for knowledge management consisting of:
  - Storing of certain types of knowledge in a systems modeling language (OPM)
  - Storing of certain types of knowledge in a multi domain numerical modeling language (Modelica)
  - Running simulations and obtaining greater understanding of the trade space
- Difficulties associated with implementing and assumptions which have been made

4.2 Assumptions about the system of interest and the modeling languages

Before any modeling activity is undertaken it is important to review what assumptions the modeling is being done under such that every engineer working on the problem has a consistent understanding. These assumptions are detailed in Table 15 (indicating Mapping (M) between models) and Table 16 (indicating Hierarchy (H)).

Assumption	**Consequences**
M1: Subsystems operate independently to enable behavior other than on defined interfaces.	There is no interaction between components in difference subsystems other than on defined interfaces.
M2: All components are rigidly connected to the internals of a subsystem. All subsystems are rigidly connected to the internals of a system of interest.	The system of interest is a rigid body.
M3: Functions and behaviours of which the engineer wishes to model can readily be modelled in the languages chosen.	Don't attempt to model aspects which are not applicable for the modeling languages which have been selected.
M4: Masses exhibit weight.	Weight force must be computed.

Table 15 Modeling assumptions (mapping between models) of the system of interest

Assumption	Consequences
H1: Systems of interest are made of subsystems which are made of subsystem components.	Hierarchical decomposition is appropriate for modeling such systems of interest.
H2: Each component contributes mass and cost to each sub system which contributes to each system of interest.	Mass and cost of a system is the sum of the mass and cost of its subsystems which is the sum of the mass and cost of it components.

Table 16 Modeling assumptions (hierarchical decomposition) of the system of interest

4.3 Problems and difficulties to overcome

Some problems and difficulties associated with this integration are listed as follows:

- System-Level design requires design synthesis not simply analysis. As such all the challenges of design synthesis present themselves including: bounding the number of alternative designs to a reasonable level, selecting criteria by which to assess alternatives, identification of creative alternatives, and ensuring any model predictions are appropriately accurate.
- Given that initially a concept design (description of function) is to be developed followed by a system-level design (description of form) the languages capture both the functional and structural aspects of the design.
- The choice of a language inevitably results in a limitation in the type of models created, as languages place limitations and emphasis on the types of knowledge they can store and how it is presented to the user.
- As per Assumption H1 of Table 16 it is assumed that the system of interest is made of the hierarchical decomposition of subsystems which are made of subsystem components. This leads to the standard challenges for top-down and bottom-up approaches for system design. Top-down approaches can result in the missing of small components which in aggregate have a large contribution to the system's attributes (e.g. a large number of rivets contributing to the mass of an airplane). Bottom-approaches can be time-consuming and result in engineers not taking into consideration the whole of the system and its primary purposes for existence.

4.4 Broad stages and aims for Solar-Boat System-Level Design exploration and selection

The first three Lifecycle Stages (LS) of the Solar-Boat project were identified as problematic and some initial solutions proposed for them (Section 1.2, Table 6). To create a more concrete solution set a series of design questions are posed as shown in Figure 23 as to what should be answered at each stage. The design questions are expanded as a more specific set of activities as shown in Figure 24 enabling the ability to move between the early lifecycle stages. The main focus of the LS1 Clarify is the identification of what value is derived from the system, how to measure this value and what resources are available to deliver it. LS2 Concept development focuses on a decomposed functional description of what every system alternative must do and how to assess the alternatives. While LS3 System-Level Design aims to develop specific alternative designs predict their performance by numerical simulation and compare.

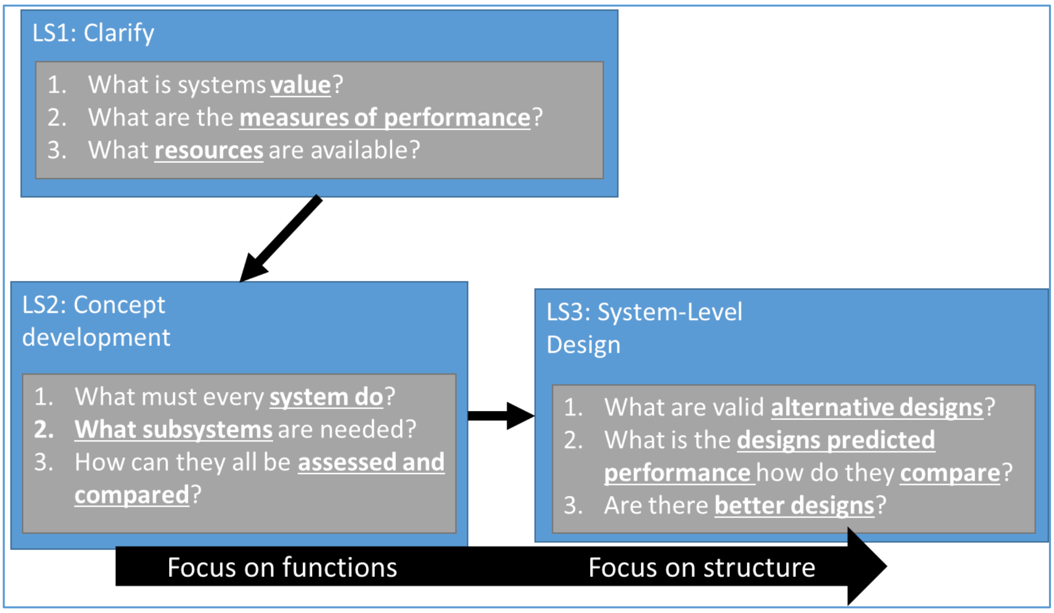


Figure 23 Design questions posed at different Lifecycle Stages (LS)

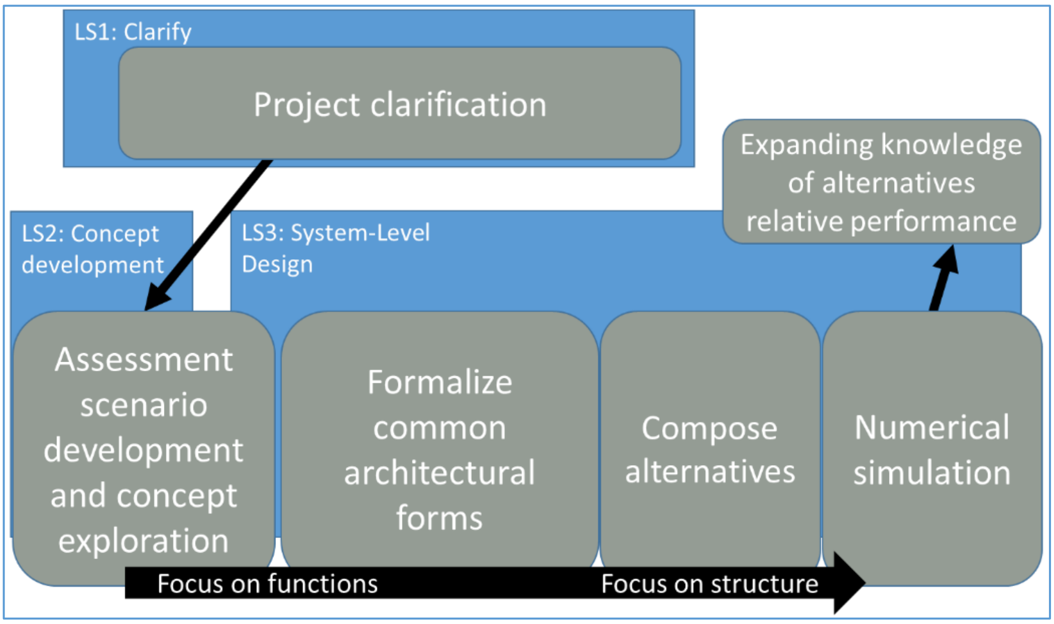


Figure 24 High level activities performed at each Lifecycle Stage (LS) in the proposed methodology

The broad set of activities displayed in Figure 24 are used to define a framework enabling the logical creation of conceptual models of system designs and how to assess them in OPM which are logically transferred to numerical models in Modelica by way of multiple model types (all of which have hierarchical decomposition) which are ultimately simulated. To achieve this, the proposed design methodology:

- Utilizes OPM to decompose the functionality required of the System of Interest and explore concepts.
- Defines a framework for mapping functional descriptions (in OPM) to a formalized common architectural form which can deliver the functionality. Alternative designs can then be developed on the common architecture (and ultimately be simulated in Modelica).
- Utilizes by way of the INCOSE Decision Management Process (INCOSE, 2015; SEBoK, 2015), Multi Objective Decision Analysis (MODA) (Cilli & Parnell, 2014) such that all the alternative designs can be quickly compared to one another.

4.5 Methodology and tools overview

4.5.1 Aim and difficulties

The aim of this section it to introduce and describe a set of model types which map from highly functional descriptions (and as such lacking information of the structure needed to deliver the functions) of the system of interest to ultimately a structural description which can be simulated in Modelica and is ready to move into detailed design.

This raises the following general difficulties:

- How to map functional descriptions to a structural description
- How to handle causal and acausal type numerical modeling
- How to avoid replication of common items

Difficulties with the implementation of hierarchy include:

- Selection of what information to display at each level
- How to break down functions and structure consistently with each other
- Important dominant parts might not be revealed until very deep levels of decomposition are completed

4.5.2 Overview

Figure 25 provides a break down the broad activities of Figure 24 into a specific set of inputs/outputs (in green) which are linked together by specific transitions (labels on the links). The letters indicate a step in the process. To aid comprehension subsequent description of the methodology will make reference to the Solar-Boat project and many terms will be introduced on diagrams now to only be explained in later sections.

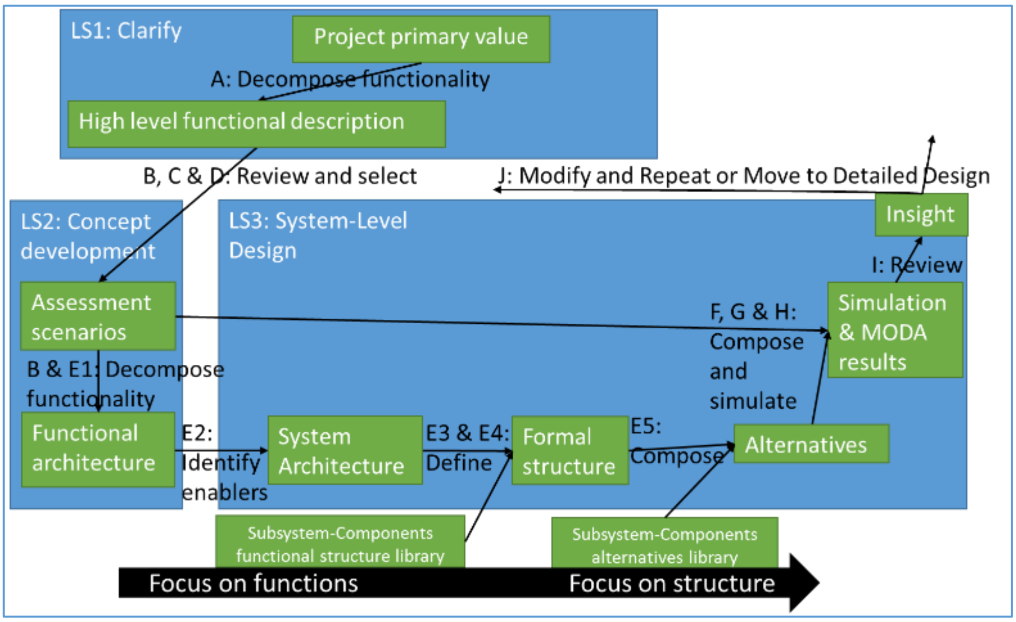


Figure 25 Proposed methodology inputs/outputs (in green) and activities (labels on the links, letters referenced later in text) at various lifecycle stages (in blue)

Figure 26 which depicts a representation of the various model types advocated in the proposed methodology at various levels of hierarchy makes reference to the Solar-Boat as an example (black triangle as per OPM indicating the higher level object or process consists of those at the lower level). A description of the hierarchy levels used in the Solar-Boat example is shown in Table 17. A depiction of the hierarchy in 3D is shown in Figure 27 while Figure 28 displays screenshots of example models at various levels of hierarchy.

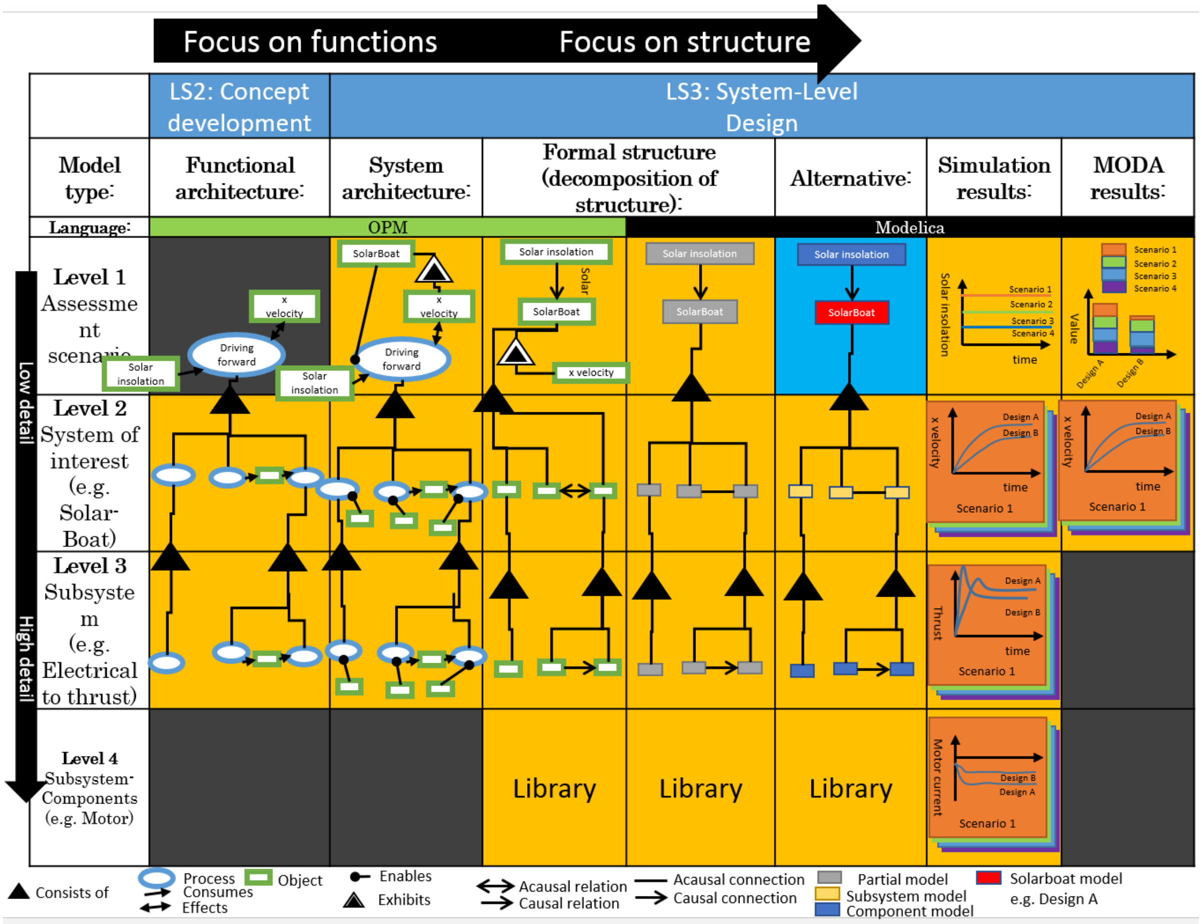


Figure 26 Representation of different model types at various hierarchy levels

Level:	Name	Definition:	Example processes:	Example object:
0	Functional Architecture – Primary Value	Concisely define the highest level value deriving functionality of the system.	Racing in Solar-Boat race event	SolarBoat Race
1	Assessment Scenario	A process which is considered appropriate to use to compare alternative designs with. When implemented as Modelica model which has been populated with a System Alternative this can be simulated as it has no connectors exposed	Driving forward, Floating	Assessment scenario result
2	System of Interest	The system which enables the process in the Assessment Scenario to be completed.	Converting Electrical to Thrust	Solar-Boat
3	Subsystems	The subsystems which make up the System of Interest.	Converting Electrical to Rotation	Electrical to Thrust subsystem
4	Subsystem-Components	The components which make up a subsystem.		DC Motor

Table 17 Definitions and examples of Solar-Boat modeling hierarchy

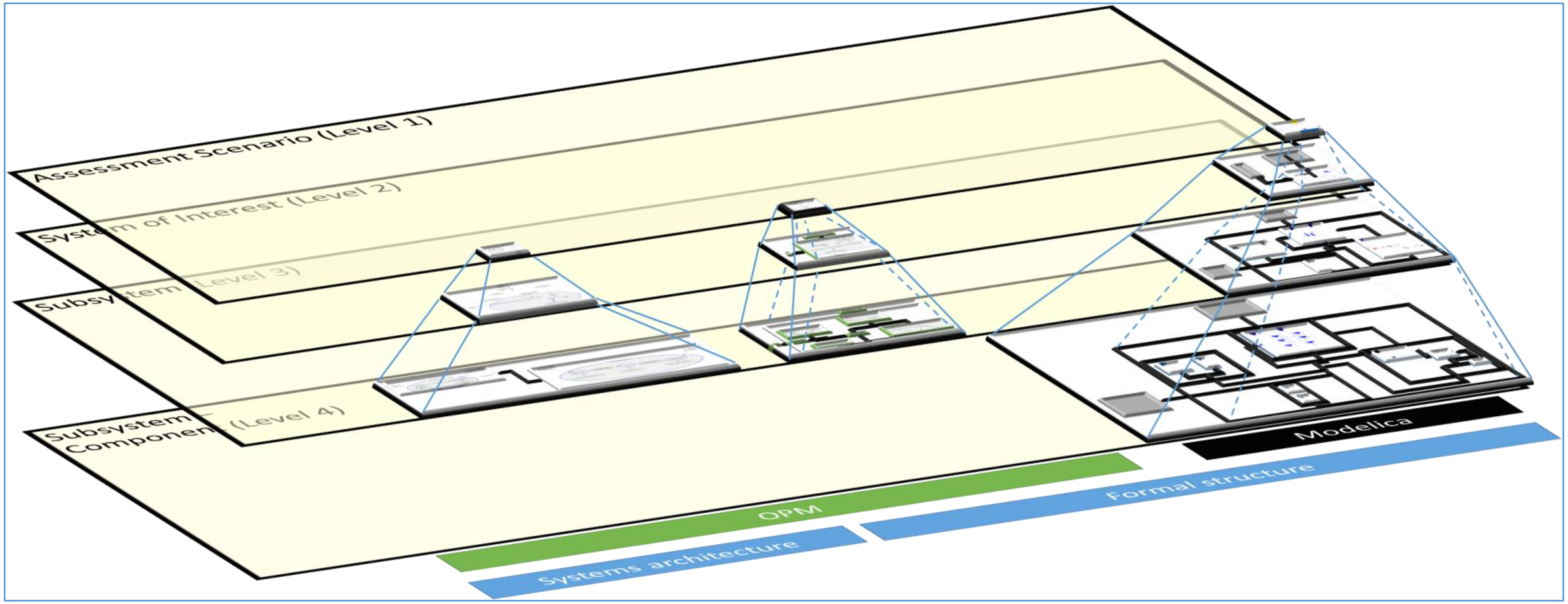


Figure 27 Visual description of the assessment hierarchy. Left: Example System Architecture decomposition in OPM. Middle: Example Formal Architecture decomposition in OPM. Right: Example Formal Architecture decomposition in Modelica.

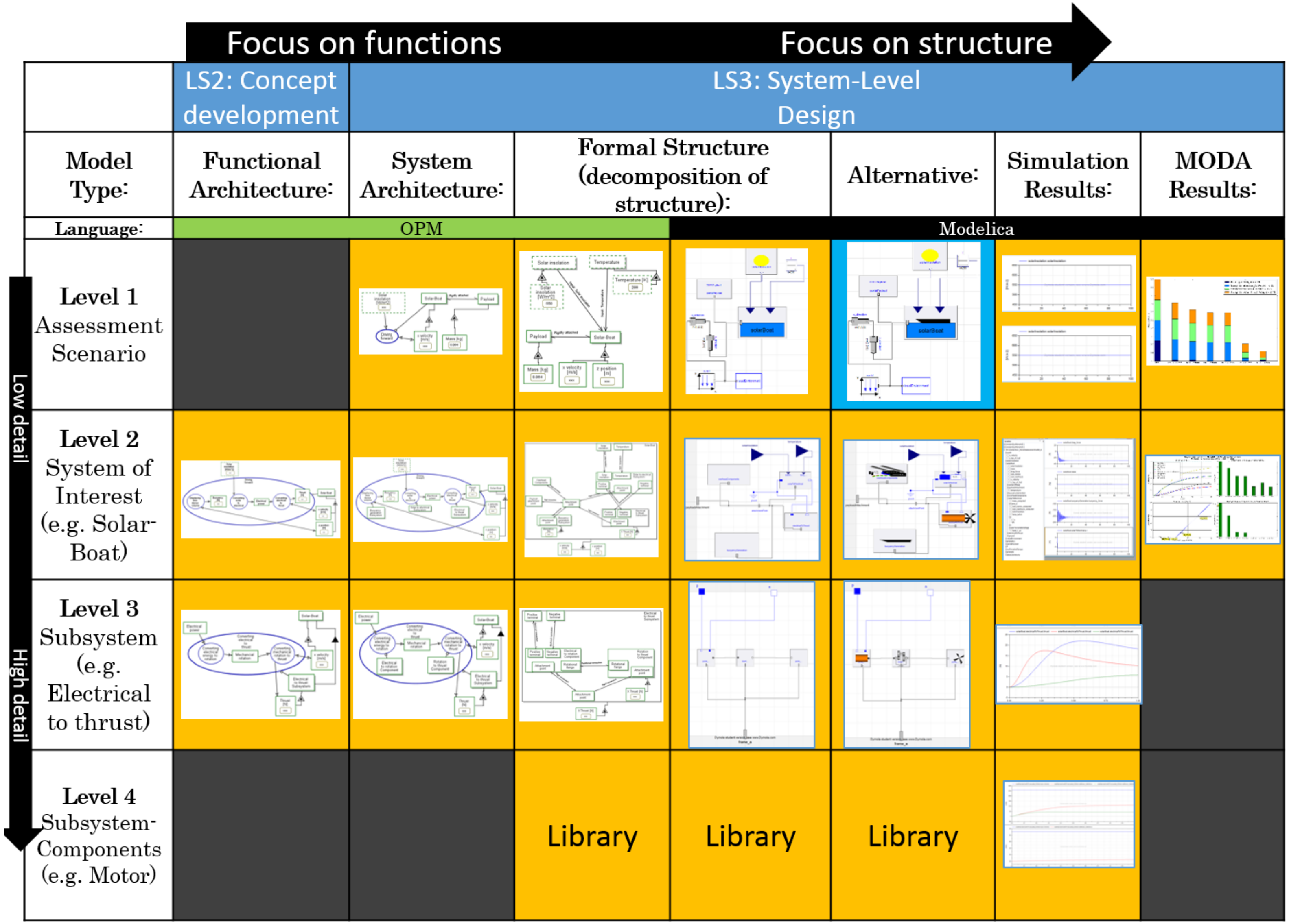


Figure 28 Examples of the different model types at the different levels of hierarchy

Names of the various transitions to complete decomposition to lower hierarchical levels, mapping between different model types and composition to higher hierarchical levels and are provided in Figure 29 with letters indicating a step in the process which consistent with Figure 25.

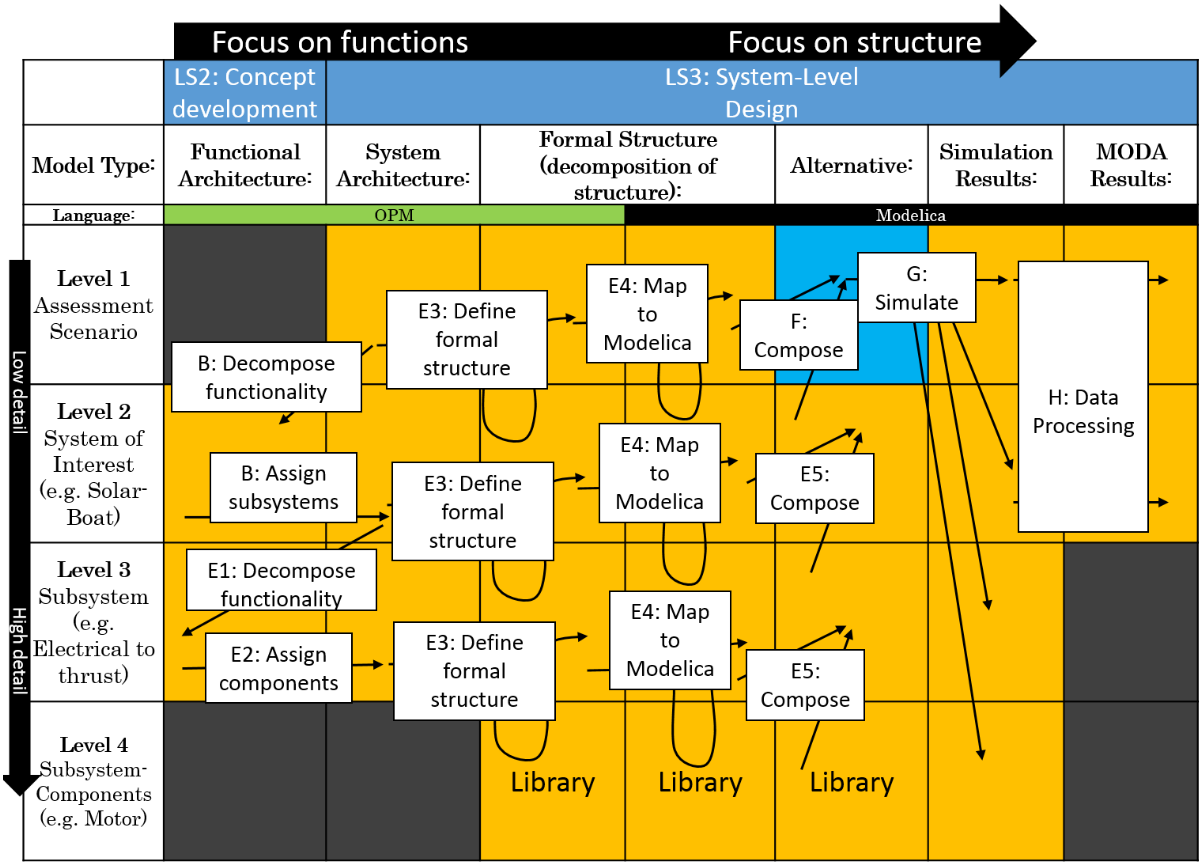


Figure 29 Transitions to move between the different model types and the hierarchy levels

Figure 30 lists the steps needed to realize the flow presented in Figure 25 (i.e. referencing the letters), further if the step is additionally represented on Figure 29 this is also noted for reference. Detail of each of steps including diagram examples for Solar-Boat are presented in subsequent sections.

Step A: Identifying and Decomposing Functional Architecture – Primary Value.
Step B: Identifying Subsystems Required for Modelling the System of Interest (Figure 29).
Step C: Reviewing and Selecting Assessment Scenarios.
Step D: Configuring Specific Assessment Scenarios:

- **Step D1:** Varying Assessment Scenario inputs.
- **Step D2:** Defining each Assessment Scenario's Value Function, Weight, Simulation Length and Data Extraction Method.

Step E: Synthesizing System of Interest Designs in OPM and Modelica:

- **Step E1:** System of Interest Process Decomposing (Figure 29).
- **Step E2:** Assigning Subsystem-Components to Enable the Processes (Figure 29).
- **Step E3:** Mapping to a Formal Structure in OPM (Figure 29).
- **Step E4:** Mapping to a Formal Structure in Modelica (Figure 29).
- **Step E5:** Composing Alternatives in Modelica (Figure 29).

Step F: Composing each System of Interest Alternative into each Assessment Scenario for Simulation (Figure 29).
Step G: Simulating every Assessment Scenario and System of Interest Alternative Combination (Figure 29).
Step H: Consolidating Simulation Results with MODA (Figure 29).
Step I: Reviewing Results.
Step J: Modify and Repeat or Move to Detailed Design.

Figure 30 Proposed flow to develop System-Level designs based on the letters indicated on Figure 25 and Figure 29

Initially a single OPM process and therefore the corresponding Functional Architecture – Common are used to define a Systems Architecture at corresponding levels of decomposition which is mapped for a Formal Structure (in OPM then Modelica). Variation of the Subsystem alternatives produces System of Interest alternatives all of which make use of the same Functional Architecture such that they can all be subject to the same assessment criteria. Four hierarchy levels are defined in this thesis, but of course an infinite number could be defined.

4.6 Types of models and hierarchy used

In the previous section various model types and hierarchy levels were introduced briefly. In this section the detail is presented, however description of the transition between the models (i.e. following the flow of Figure 30) is explained in subsequent sections.

4.6.1 Types of models

It is important to establish some definitions of the types of models to be created. These definitions aim to make use of generic words not specific to OPM or Modelica. However this is not always possible or desirable as suitable definitions for OPM and Modelica concepts have already been clearly defined in their respective language specifications ((ISO, 2015) and (Modelica Association, 2012)). In such cases introducing a new word created for this thesis would likely increase confusion so it has been avoided.

Initially the following definitions should be reviewed:

- **Behavior:** Dynamic aspect: How a system changes with time (Dov Dori, 2015)
- **Structure:** Static aspect: What the system is made of (Dov Dori, 2015)
- **Function:** Process that provides functional value to a beneficiary (ISO, 2015). I.e. behavior with value.

Based on these initial definitions and the work of (Crawley et al., 2015) presented in Section 3.3.1.A, Table 18 is shown which describes at a high level the models needing to be created and where they are proposed to be implemented in OPM and Modelica. Table 19 expands on this by describing how each of these models (shown from left to right) contains functional and structural information (at one layer in a hierarchy), while Figure 28 of the previous section displayed example OPM and Modelica diagrams of what each of these models is visualized as. Each of these model types are explained in detail in the next sub-sections.

Model type:	**Definition:**	**OPM**	**Modelica**
Functional Architecture	As per Figure 21 from (Crawley et al., 2015) this is a purely process and operand description of the item for modeling. There is no indication of what objects are need to implement the system (form), as such no Modelica model can exist.	Y	N
System Architecture	As per Figure 21 from (Crawley et al., 2015) this involves the assignment of objects to processes. By doing this functional requirements of each object are implied.	Y	N
Formal Structure	As per Figure 21 from (Crawley et al., 2015) this is the direct connection of objects. As such this is applicable to both OPM and Modelica modeling.	Y	Y
Connectors	Defines how the object is connected to other objects. This can be seen on the object to object relations on the System Architecture or Formal Structure.	Y	Y
Alternative	A fully specified definition of the item for modeling such that if included in the appropriate model it could be simulated.	N	Y
Simulation result	Time series object attribute values.	N	Y

Table 18 Definitions of the types of models needing to be created at various levels of the OPM and Modelica assessment hierarchy

Focus on functions → Focus on structure

Model type:		**Functional architecture:**	**Systems architecture (decomposition of process):**	**Formal structure (decomposition of structure):**		**Alternative:**	**Simulation result:**
Language:		OPM			Modelica		
One level of hierarchy:	**Function:**	Processes affecting object states.		Instrument objects: Attribute variables, process attachment, connector types.	Name of partial model. Connector types.	Equations, algorithms and associated object attributes.	Computed object attribute values.
	Structure:	No	Implied structure based on instrument objects assigned to processes.	Between objects: Causal unidirectional structural link and Acausal bidirectional structural link.	Connections between connectors.	Connections (causal and acausal) between Subsystems and Subsystem Components.	No

Table 19 Description of how Function and Structure are stored in different model types in different model types and modeling languages.

4.6.1.A Functional Architecture

Functional Architecture based on that proposed by (Crawley et al., 2015) aims to be purely process and operand description of the item for modeling. As such when represented in OPM, objects are only those which are effected by a process, not those which enable a process; there is no indication of what objects are need to implement the system (form), as such no Modelica model can exist. Expanding the example in Figure 26 Level 1 as Figure 31, it is possible to see that the processes "Driving forward" consumes "Solar insolation" and affects "x velocity" but there is no indication to the form which enables this function (given the design target of a Solar-Boat is assumed this particular model does not strictly exist hence the dark background on Figure 26).

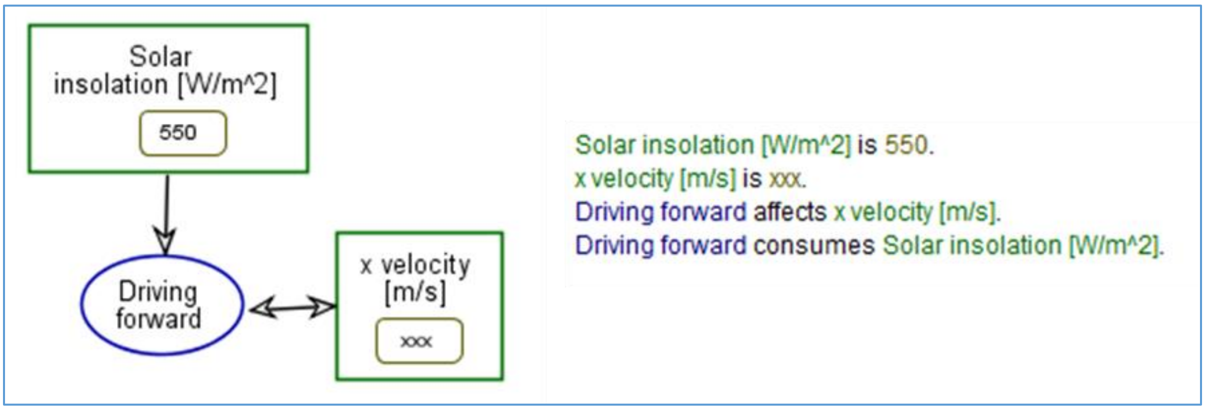


Figure 31 Example Functional Architecture (Level 1 Solar-Boat)

4.6.1.B System Architecture

System Architecture as per that proposed by (Crawley et al., 2015) adds form to the model by the assignment of objects to enable the processes (functional requirements of the assigned objects is the functional architecture). For this research the assumption is made that each process is enabled by an individual object forming a process and object pair in the System Architecture diagram. Further, for the proposed methodology, the name given to the enabling object is assumed to be used to categorize a library of components, with the name corresponding to the input, output and processing performed by the object. In Figure 26 Level 1 a representation is presented which is expanded as Figure 32. The "Driving forward" process is shown to be enabled by the "SolarBoat" object, which is shown to be the exhibiter of the "x velocity".

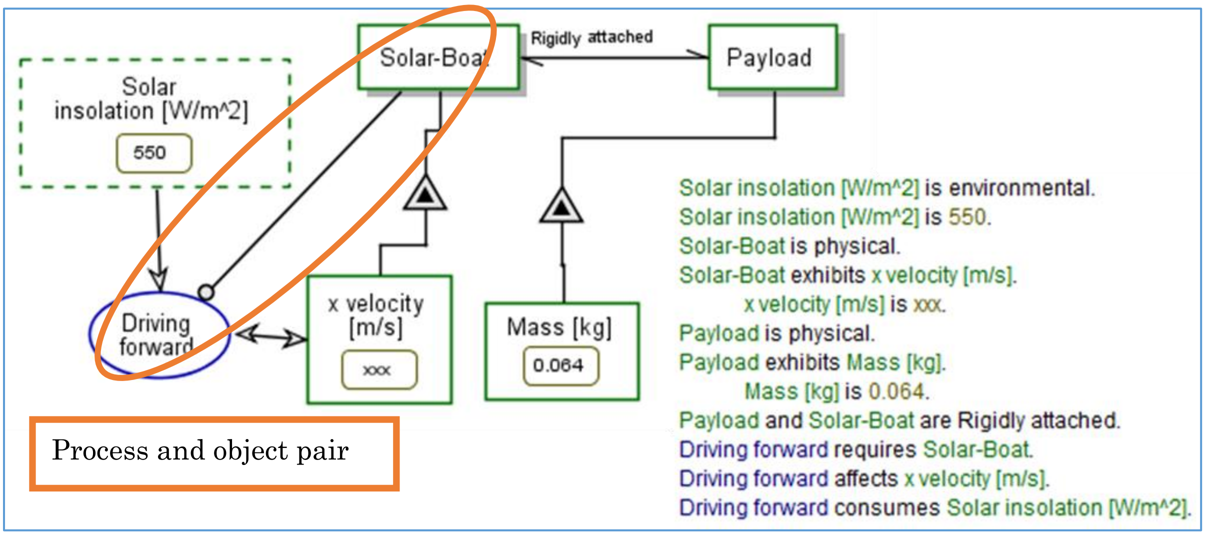


Figure 32 Example System Architecture (Level 1 Solar-Boat, process object pair highlighted)

4.6.1.C Formal Structure

Formal Structure is presented in this research in two separate modeling languages: OPM and Modelica. While the Formal Structure presented in (Crawley et al., 2015) is System Architecture without the representation of processes, in this research the representation is somewhat more specific to enable the creation of a Modelica model. In Modelica, Formal Structure is taken to mean a model which defines the interfaces exposed by model components, the connections between such interfaces and the public accessibility of the attributes of interest. By utilizing Modelica's polymorphism and object replaceability features such an interface based model can enable multiple alternative designs sharing common architectures. Such replacement is of Modelica replaceable partial components. To aid the creation of the Formal Structures in Modelica, initially a Formal Structure is created in OPM which is subsequently mapped to Modelica. No processes are displayed in the model as processes are realized by the behavior the components (which are realized by the equations of in Modelica models). However due to the focus on Modelica model creation in the Formal Structure OPM model, interfaces exposed by model components and the connections between such interfaces and the public accessibility of attributes of interest are explicitly modeled.

Figure 26 Level 1 presents a representation of both OPM and Modelica Formal Structures which are expanded as Figure 33 and Figure 34 (left side) respectively. Key points from Figure 33 to note: No longer is the “Driving forward” process displayed as this behavior is contained in the enabling object “SolarBoat” as such the “Solar insolation” is now directly connected to the “SolarBoat” enabling object (not the process). For the purposes of mapping to Modelica the relations between objects is as such: single headed relation between an object represents a causal connection and a double headed fishhook relation an acausal.

The Modelica model implements this same structure as the OPM model. Where the Solar-Boat block is a Modelica replaceable partial component. Attribute of interest “x velocity” and “z position” exist as variables of the SolarBoat model in Figure 34 (hence the model constrains all motion except that in x and z). Further the named relations between objects in Figure 33 are now specific connection types in Figure 34.

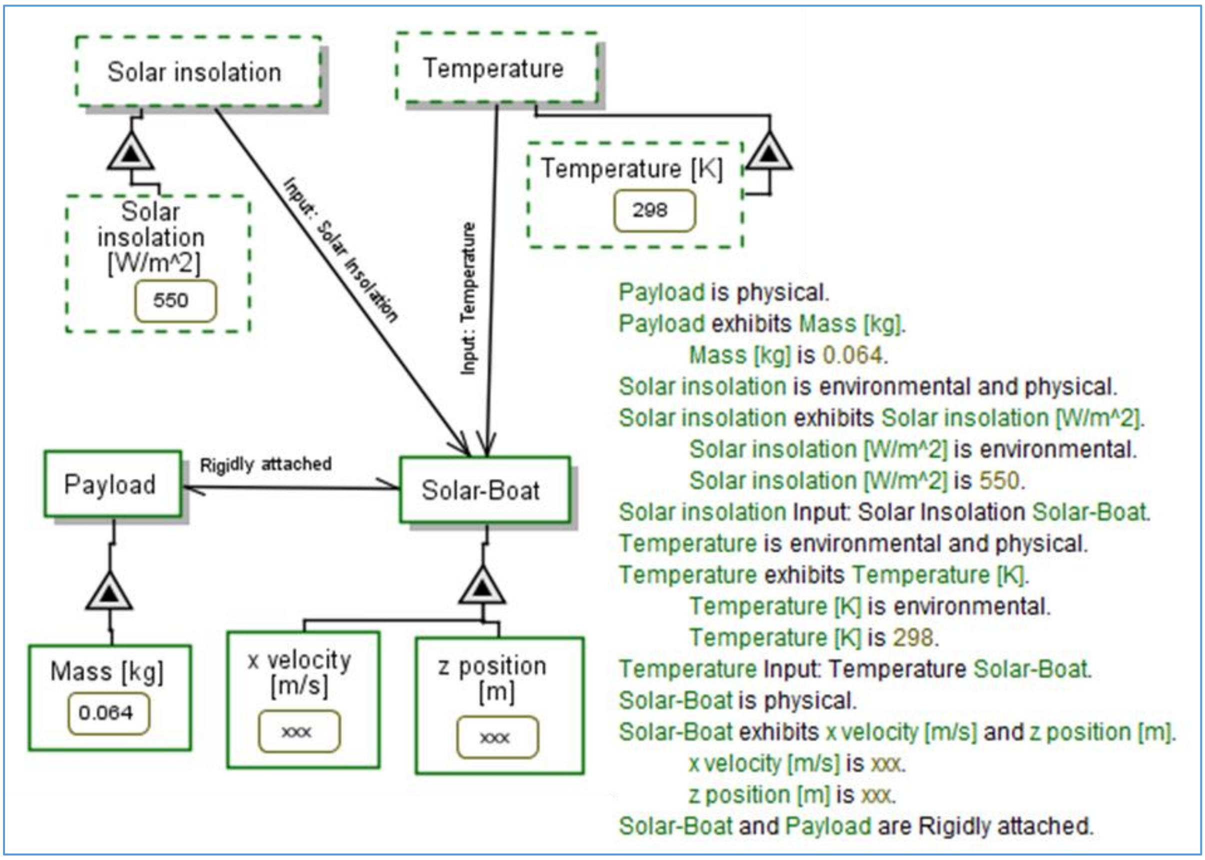


Figure 33 Example Formal Structure – OPM (Level 1 Solar-Boat)

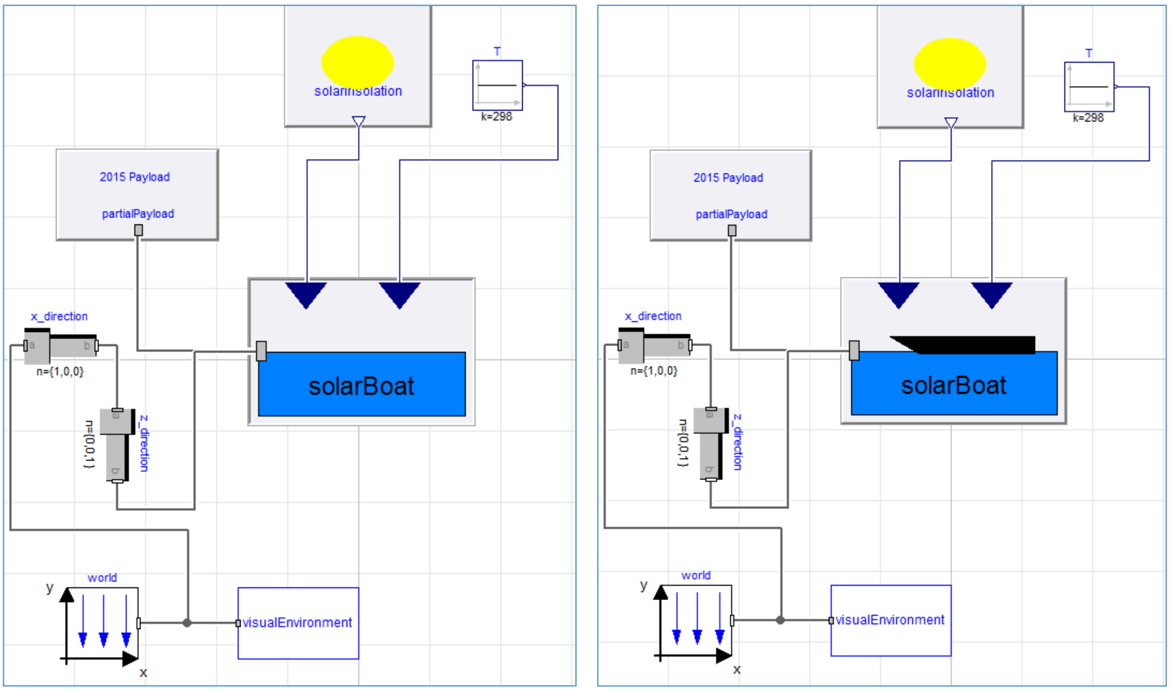


Figure 34 Level 1 Solar-Boat. Left: Example Formal Structure – Modelica. Right: Example Alternative

4.6.1.D Alternative

Alternative is for this research a fully implemented Modelica model (no use of partial). As such at Level 1 this represents a model which can be simulated (an Assessment Scenario and System of Interest Alternative Pair), while at Level 2 it represents an Alternative System of Interest design composed out of Subsystems (Level 3) which intern has been composed out of Subsystem-Components (Level 4) from the library. Coloring is used in Figure 26 to represent the Formal Structure Modelica replaceable partial components (in grey) being replaced with Modelica components (colored), this can be viewed in more detail in Figure 34 (right) where the Modelica replaceable partial component of a SolarBoat has been replaced with an alternative design.

An example of an alternative Subsystem-Components (Level 4) from the library is presented in Figure 35 modeling the electric motor "Turnigy L3040A-480G" which itself is made out of components from the Modelica Standard library. But of course new Subsystem-Components can be defined from equations or additional custom models.

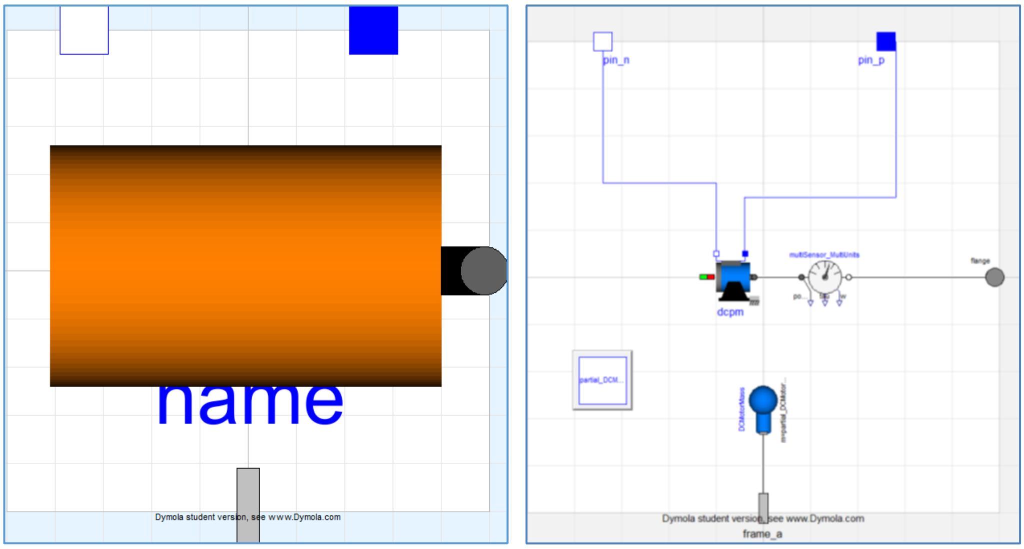


Figure 35 Example Subsystem Component – Alternative (DC motor) used to populate a Subsystem – Formal Structure. (Left: Modelica icon layer, Right: Modelica diagram layer).

4.6.1.E Simulation Results

Simulation Results are the output of simulating a Modelica model; time series of the various variables defined in the model. As per Figure 26 the time series are associated with every variable in every model at every level of the hierarchy. On Figure 26 the multiple data series represent comparing alternatives and the stacks represent the different Assessment Scenarios that have been run.

4.6.1.F Multi Objective Decision Analysis (MODA) Results

Multi Objective Decision Analysis (MODA) Results for the purposes of this research are the consolidation of the Simulation Results of each Assessment Scenario and System of Interest Alternative pair in a format to enable a quick overview of the value provided by the various alternatives, as opposed to exploring the large number of variables and alternatives and quickly becoming overwhelmed. Following the method advocated by (Cilli & Parnell, 2014). The comparison involves extraction from each Assessment Scenario and design alternative pair simulated, a measure of the System of Interest's performance given the symbol x_i (e.g. max x_velocity) from the raw simulation results. The extracted variable is then used to compute an unweighted value by the utilization of the value function (given the symbol $v_i(x_i)$; see Figure 36 for an example). The unweighted value is then multiplied by the weight (given the symbol w_i) assigned for that scenario (weighted value); as per Equation 1. Where the sum of all the weights (w_i) for all the Assessment Scenarios is 1 (Equation 2).

$\boldsymbol{v(x) = \sum_{i=1}^{n} w_i v_i(x_i)}$	Equation 1 from (Cilli & Parnell, 2014)
$\boldsymbol{\sum_{i=1}^{n} w_i = 1}$	Equation 2 from (Cilli & Parnell, 2014)

The "ideal system" having a total weighted value of 1 as its performance is assumed to always be at the stretch goal. As such the total value can be presented as a breakdown by Assessment Scenario. A representation of is presented in Level 1 of Figure 26 while larger more detailed examples are presented in Section 4.7.8.

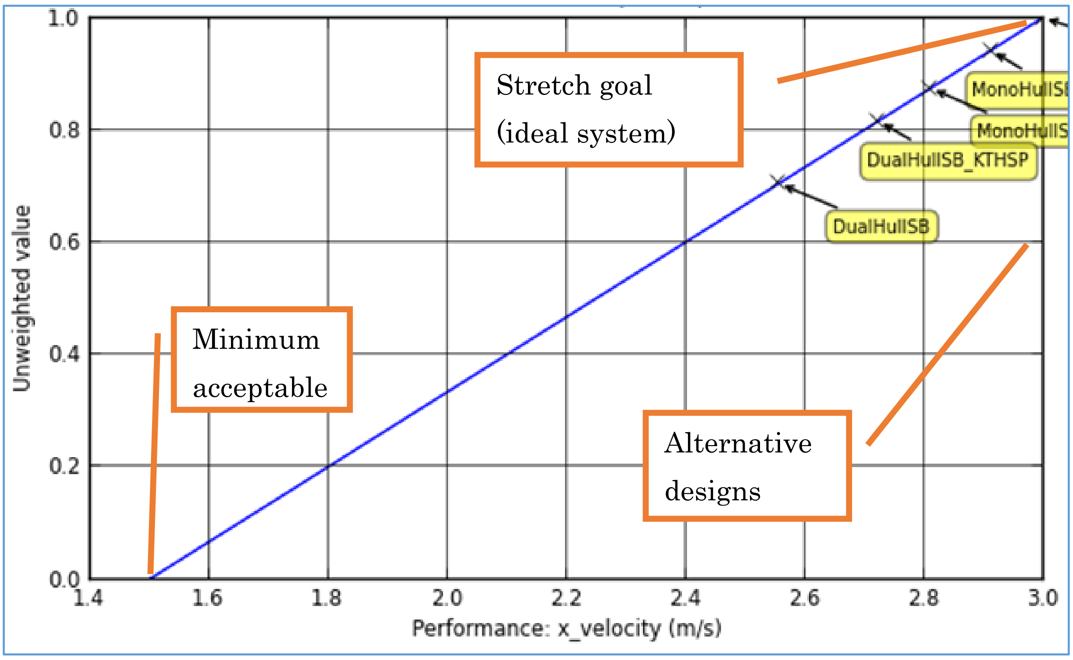


Figure 36 Example Linear Value Function.

4.6.2 Using hierarchy to handle complexity

Based on the assumption that the system of interest is fundamentally hierarchical it is then assumed that each model type described in the previous section can be decomposed. Based on this Table 19 is modified as Table 20 where by descriptions of how Function and Structure are stored between hierarchy levels for the different model types. The hierarchy used for this research when applied to the Solar-Boat was presented previously in Table 17.

Model type:		Functional architecture:	Systems architecture (decomposition of process):	Formal structure (decomposition of structure):		Alternative:	Simulation result:
Language:		OPM			Modelica		
Between levels:	Function:	Decomposition of processes into sub-processes.		Constituent objects functions imply functionality.	Name of constituent partial models.	Constituent Subsystem Components.	Different object attributes at each layer.
	Structure:	No	Implied structure based on Instrument object assigned to decomposed sub-processes.	Instrument objects constituent objects.	Partial models constituent partial models.	Subsystems constituent Subsystem Components.	Time series of computed object attribute values in structural hierarchy.

Table 20 Description of how Function and Structure are encapsulated in different model types between hierarchy levels.

4.6.3 Combining model types and different levels of hierarchy

Based on the descriptions of the model types the links between them and hierarchy decomposition it is possible to combine this information into Figure 28 (presented previously) where each model type for each hierarchy level has an example screenshot provided. Seen in this view the following salient points are noted about the presented description:

- There is a clear separation of the activities of functional description and later structural description
- By defining clear reasoning and usage for each model type it is hoped the user is somewhat guided to create manageable diagrams
- The user is guided to make a highly modular architecture

4.7 Detailed description of each step of the methodology

In this section the steps described in Figure 30 are expanded such that sufficient detail is provided. The Solar-Boat is used as an example through out to illustrate the process. To aid comprehension where appropriate a miniaturization of Figure 29 (transitions between different model types) is presented throughout the section with the particular transition of interest highlighted.

4.7.1 Step A: Identifying and Decomposing Functional Architecture – Primary Value

A Functional Architecture – Primary Value diagram (known as System Diagram in (Dov Dori, 2002; ISO, 2015)) is drawn to initially, concisely define the most high level value deriving functionality of the system. An example is shown in Figure 37. It has a single process "Racing in SolarBoat Race Event" which is enabled by the System of Interest (SolarBoat) and the attributes which are to be varied by the process (SolarBoats dynamic states and the Race ranking). Additionally weather is an input to the process. Additional points of note about the knowledge displayed in this diagram are described in Table 21. Given this model type was not described in the previous sections a more formal definition is provided:

Functional Architecture – Primary value: Corresponds to the "OPM Systems Diagram". This indicates the primary value delivering process of the system. This is common to all alternative designs. E.g. Racing in Solar-Boat race (Figure 37).

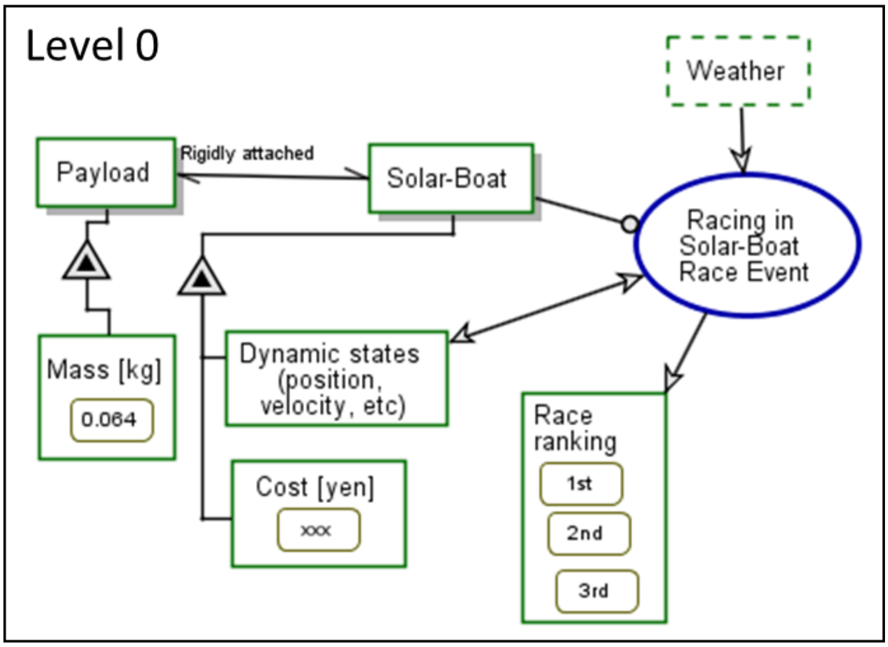


Figure 37 Level 0: Functional Architecture – Primary Value for Solar-Boat

Knowledge:	Discussion:
Primary process: “Racing in SolarBoat Race Event”	Describes concisely the process which delivers value.
Primary operand: “Race ranking”	The value of this determines the success of the project.
Inputs: “Weather”	It is recognized that weather is an important input to this system.
Operand: “Payload”	It is noted that the payload has mass and is rigidly connected to the Solar-Boat.
Operand: “Solar-Boat”	It is noted that the SolarBoat has cost and its state will change due to racing.

Table 21 Discussion of knowledge described in Functional Architecture – Primary Value for Solar-Boat (Figure 37)

However the primary process “Racing in Solar-Boat Race Event” lacks detail to create valid alternatives for assessment. As such the primary value process is decomposed into sub-processes which represent the functionality of all valid alternative System of Interest designs are expected to perform. This is represented in Figure 38 (known in this research as Functional Architecture – Common) where the process “Racing in SolarBoat Race Event” is decomposed into “Floating”, “Driving forward”, “Turning”, “Handling disturbance” and “Ending race”. Each of these indicate by an effects link which dynamic states of Solar-Boat is affected by the process (to avoid diagram clutter only those for “Floating” (z position [m]) and “Driving forward” (x velocity [ms^{-1}]) are displayed). Detail of which is described in Table 22.

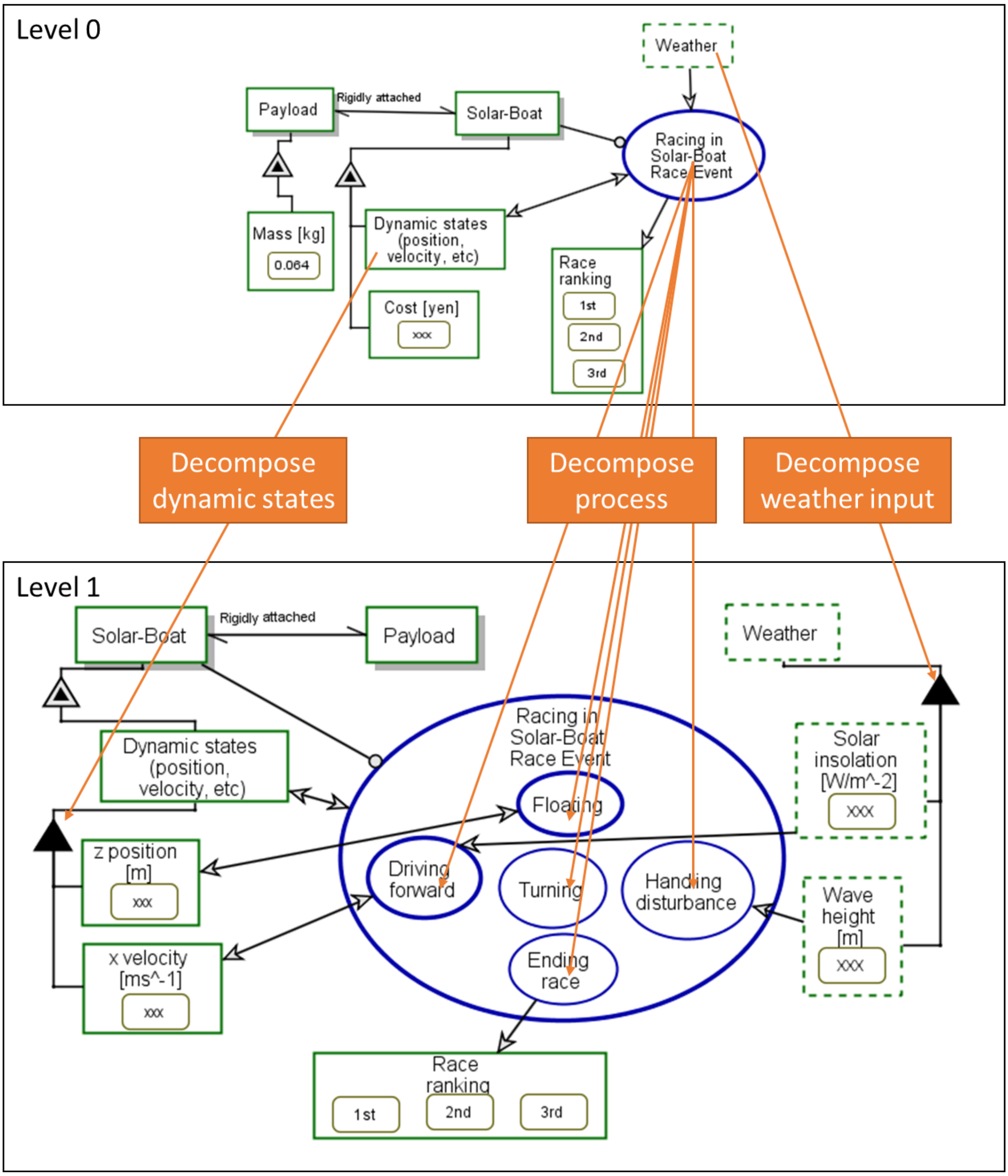


Figure 38 Level 0 to Level 1: Decomposing the projects Primary Value into sub-processes (Functional Architecture - Common)

Knowledge:	Discussion:
Sub process: “Floating”	At the start of the race the boat is floating in the water waiting to start.
Sub processes: “Driving forward”, “Turning” and “Handling disturbance”	During the race the boat attempts to navigate to its waypoints (performing “Driving forward” and “Turning”) while waves cause it to need to perform “Handling disturbance”.
Sub process: “Ending race”	By completing the “Ending race” process the race result is known.
Weather: “Solar insolation” and “Wave height”	Weather is broken down into two constituent components.

Table 22 Discussion of knowledge described in the Functional Architecture - Common for Solar-Boat (Figure 38)

Given this model type was not described in the previous sections a more formal definition is provided:

Functional Architecture – Common: Description of the system only using processes and objects which are affected by the processes. This is common to all alternative designs. E.g. Floating, Driving forward, Handling disturbance. (Figure 38 bottom).

4.7.2 Step B: Identifying Subsystems Required for Modelling the System of Interest

Dependent on the purpose of the modeling activity and trade study different processes in the decomposed project primary value (Figure 38) may or may not be appropriate for modeling (at potentially different levels of detail). The purpose of this trade study is assumed to be for the development of a simple Solar-Boat prototype with only the core functionality, as such it is deemed appropriate to only model the "Floating" and "Driving forward" processes. However these processes are not of sufficient detail for modeling so should be further decomposed. Step B as part of the larger methodology is shown in Figure 39.

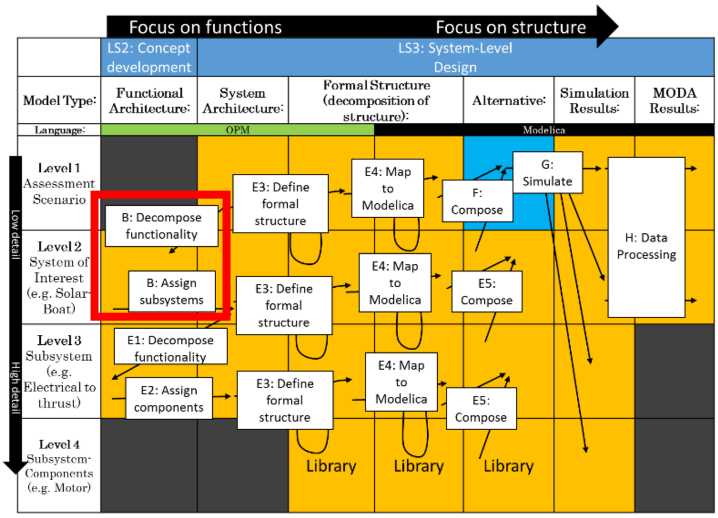


Figure 39 Step B: Identifying Subsystems Required for Modelling the System of Interest (miniaturization of Figure 29)

An example of such functional decomposition of "Driving forward" process into "Displacing less dense volume", "Converting solar to electrical" and "Converting electrical to thrust" is displayed in Figure 40 (discussed in Table 23).

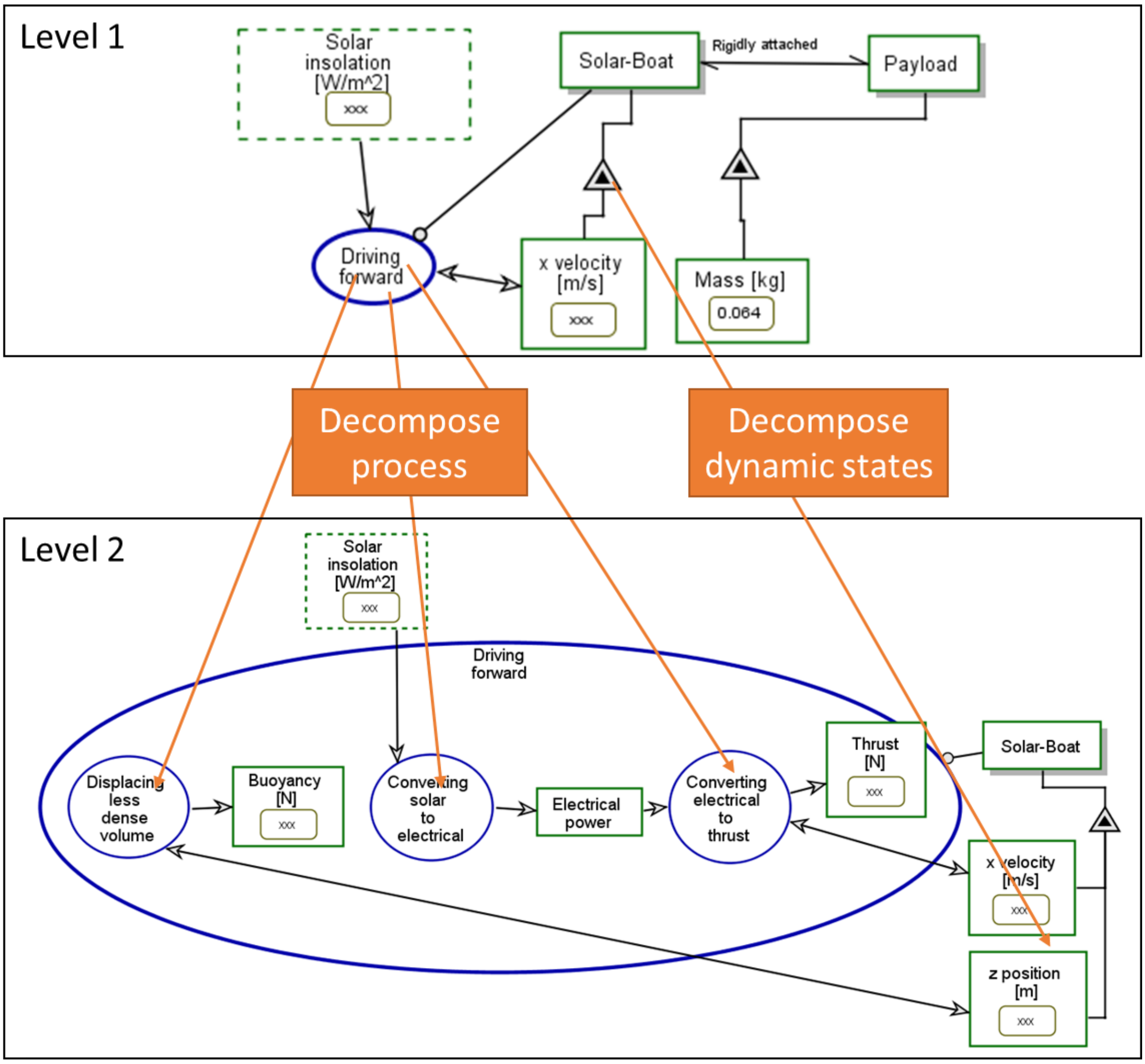


Figure 40 Level 1 to Level 2: Decomposing the System Architecture of "Driving Forward"

Knowledge:	Discussion:
Sub process: "Displacing less dense volume"	By including this process it is implied that the boat does not sink due to buoyancy.
Operand: "z position"	An important measure of performance for this process.
Sub processes: "Converting solar to electrical" and "Converting electrical to thrust"	Two concurrent processes are identified to create thrust.
Operand: "x velocity"	An important measure of performance for this process.

Table 23 Discussion detail described in decomposition of: "Driving Forward" (Figure 40)

But to define what Subsystems must be implemented, objects must be assigned to enable the processes and as such develop a System Architecture. An example is shown in Figure 41 where “Buoyancy generation”, “Solar to electrical” and “Electrical to thrust” Subsystems have been defined. Similar decomposition and subsystem assignment can occur for the other processes. Decomposition of “Floating” results in the “Displacing less dense volume” process and therefore need for “Buoyancy generation” Subsystem (but not the subsystems to power the craft). Figure 42 subsequently consolidates the identified Subsystems and attributes of interest of the System of Interest (Solar-Boat). If the System of Interest (Solar-Boat) is to be assessed without the subsystems associated with “Turning”, “Handling disturbance” and “Ending race” this should be sufficient to develop a Formal Structure and complete Modelica modeling. Alternative Functional Architectures are possible which aim to deliver the same higher level process of “Floating” or “Driving forward”. In this case only one alternative is reviewed, but one could imagine a process of using solar energy to convert water into steam to drive a turbine being a valid alternative.

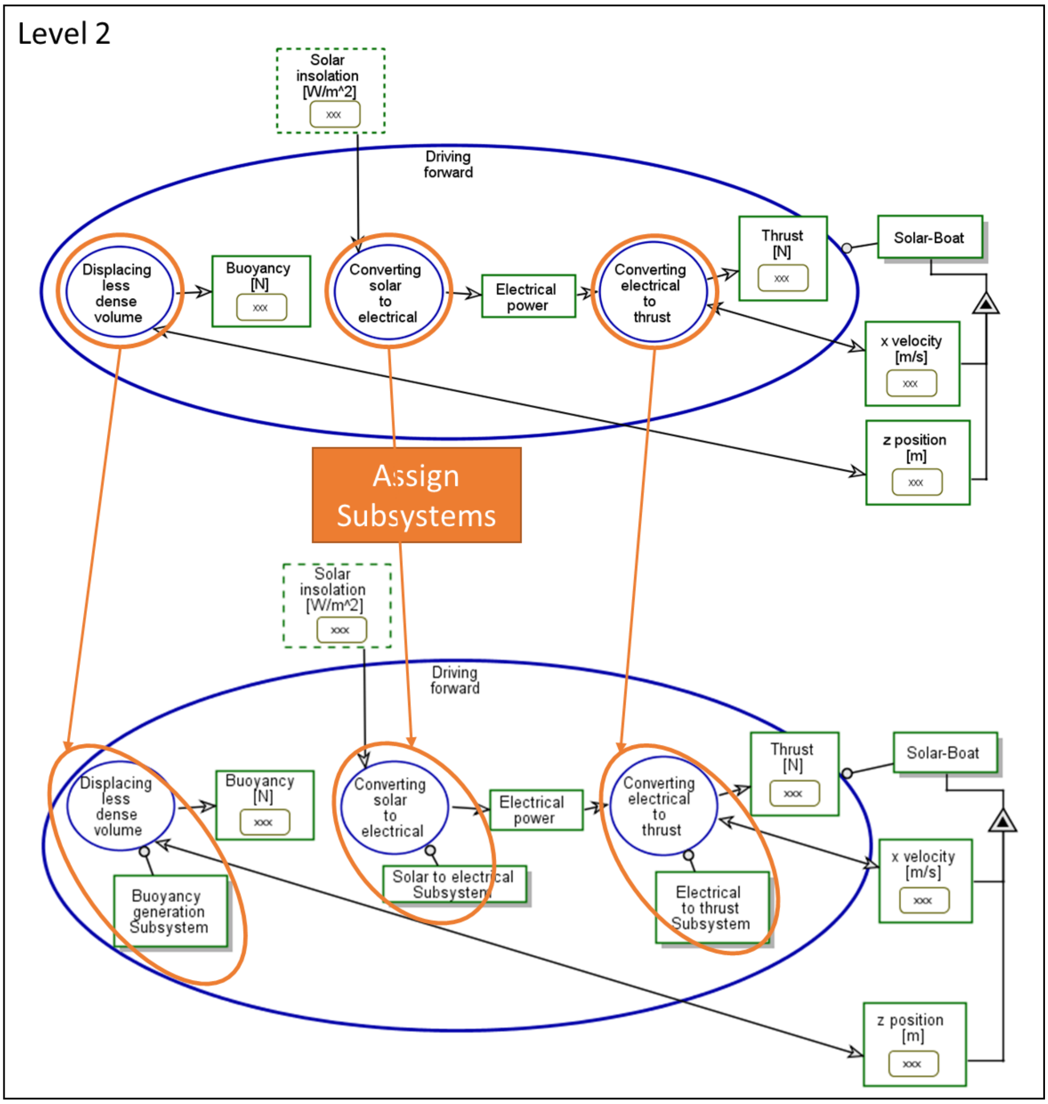


Figure 41 Level 2: System Architecture for the decomposed “Driving Forward” process

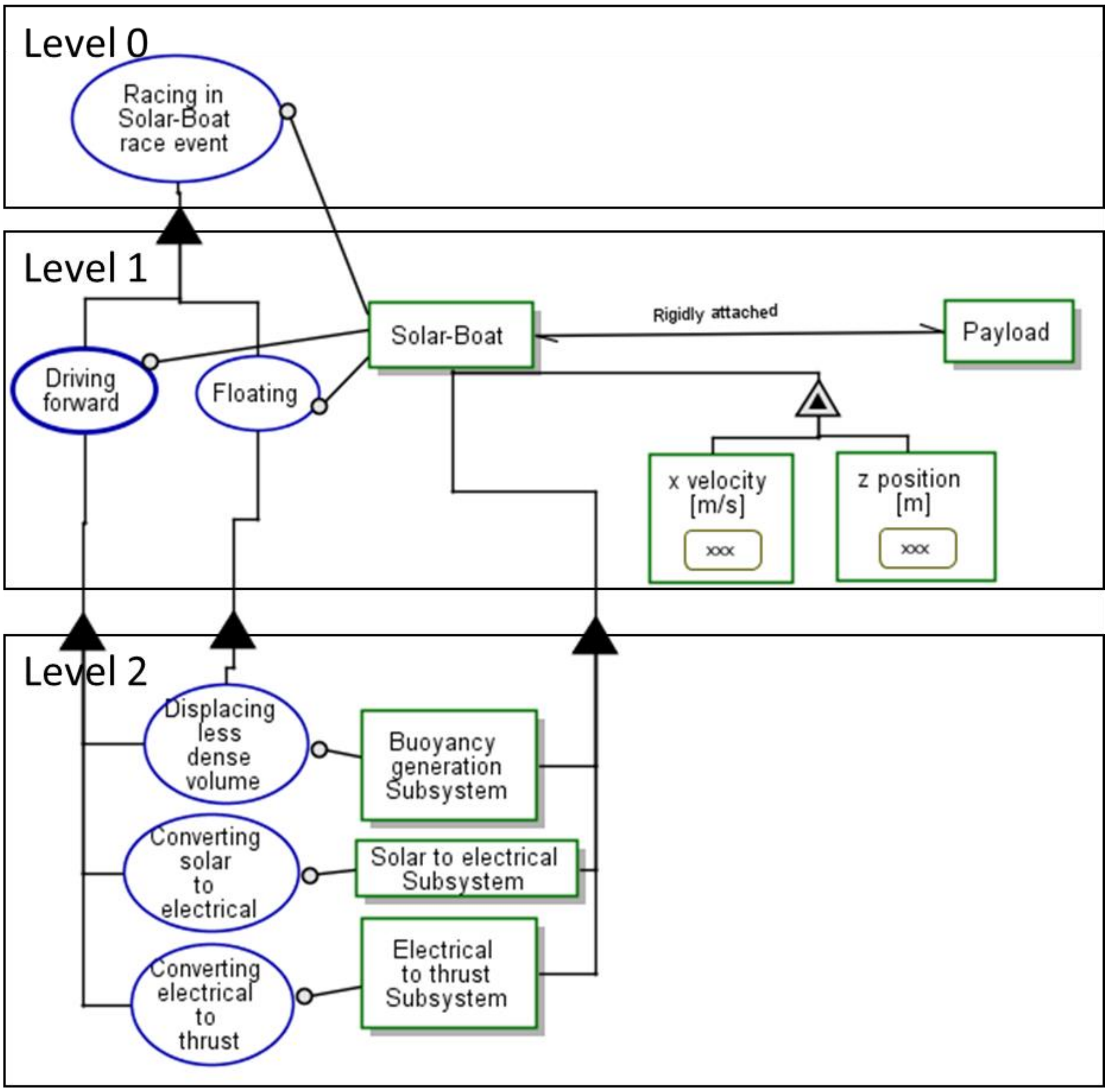


Figure 42 Level 1 and Level 2: Solar-Boat subsystems and attributes of interest

4.7.3 Step C: Reviewing and Selecting Assessment Scenarios

To develop a method to assess the System of Interest in completing the required functions, selection amongst of the processes which were decomposed in Step B should be made to develop Assessment Scenarios which are to be used to assess the System of Interest. In this case it is assumed "Floating" and "Driving forward" are selected. Figure 43 displays this selection, including the attributes of the System of Interest varied by the Assessment Scenario processes and their inputs.

Assessment Scenario – Attribute of Interest: The measures of interest for the particular assessment scenario. E.g. x direction velocity (ms^{-1}). Given the symbol x_i. In Figure 43 is identified as "z position" for "Floating" and "x velocity" for "Driving forward".

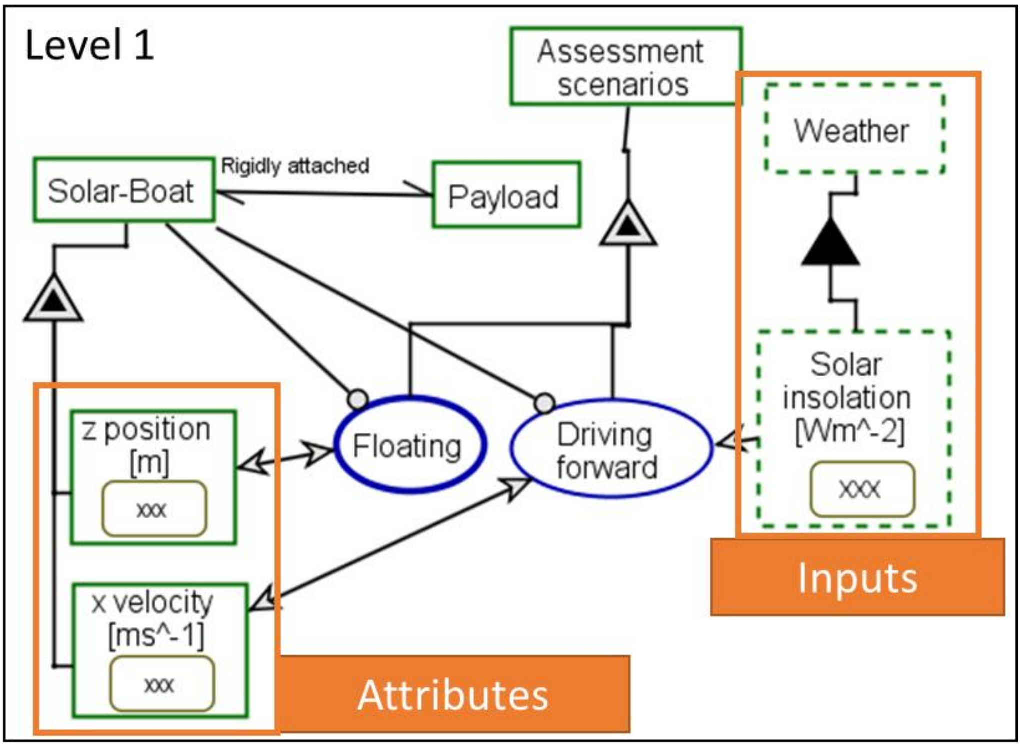


Figure 43 Level 1: Selected Assessment Scenarios

4.7.4 Step D: Configuring Specific Assessment Scenarios

4.7.4.A Step D1: Varying Assessment Scenario inputs

Involves defining explicitly the values in the inputs to the process (in the running example this would be the specific solar insolation conditions) for the purpose of later creating a specific model to simulate.

4.7.4.B Step D2: Defining each Assessment Scenario's Value Function, Weight, Simulation Length and Data Extraction Method

Is done for the later MODA computation combining the results of multiple Assessment Scenarios for each Alternative. With the value function (given the symbol $v_i(x_i)$) defines the mapping the attribute of interest to value space (i.e. Figure 36 shown previously). In this research they are all assumed to be linear functions as such are defined by the performance associated with ideal performance (stretch goal) with a value of one and minimum acceptable performance with a value of zero.

The weight (given the symbol w_i) is used to combine the results of multiple Assessment Scenarios for one design Alternative. By multiplying by each Assessment Scenarios unweighted value for each System of Interest alternative and summing the total weighted value for the Alternative can be computed.

Simulation Length indicates how long to simulate the Assessment Scenario. Data Extraction Method describes how to one value from the time series e.g. mean, maximum or minimum.

4.7.5 Step E: Synthesizing System of Interest Designs in OPM and Modelica

In Step E alternative designs to implement the functions are synthesized by making use of the hierarchy and a defined set of model types in OPM and Modelica.

4.7.5.A Step E1: System of Interest Process Decomposing

Involves the decomposition of functionality from the System Architecture of the System of Interest which was created in Step B. The purpose of Step E1 (Figure 44) is to decompose the functionality of Subsystems such that alternative implementations of the Subsystems can be identified. The three subsystems perform the following processes "Converting electrical to thrust", "Converting solar to electrical power" and "Displacing fluid with less dense volume" which should each be decomposed.

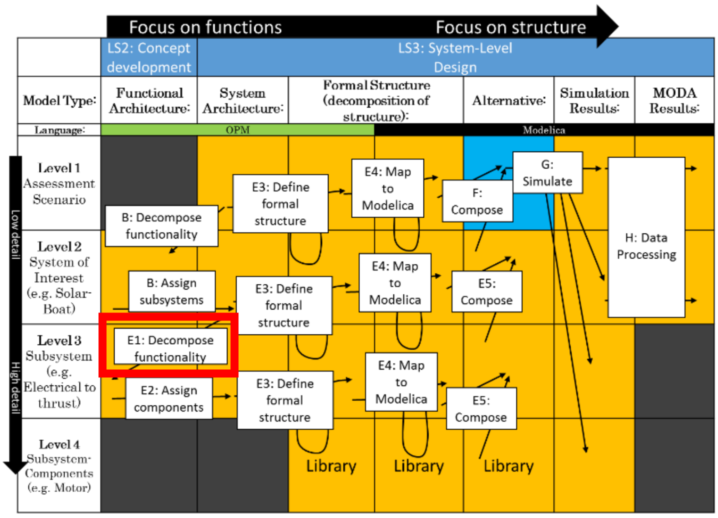


Figure 44 Step E1: System of Interest Process Decomposing (miniaturization of Figure 29)

An example of this is shown in Figure 45 where the "Converting electrical to thrust" process is decomposed into "Converting electrical energy to rotation" and "Converting mechanical rotation to thrust". Of course other decompositions could also be valid.

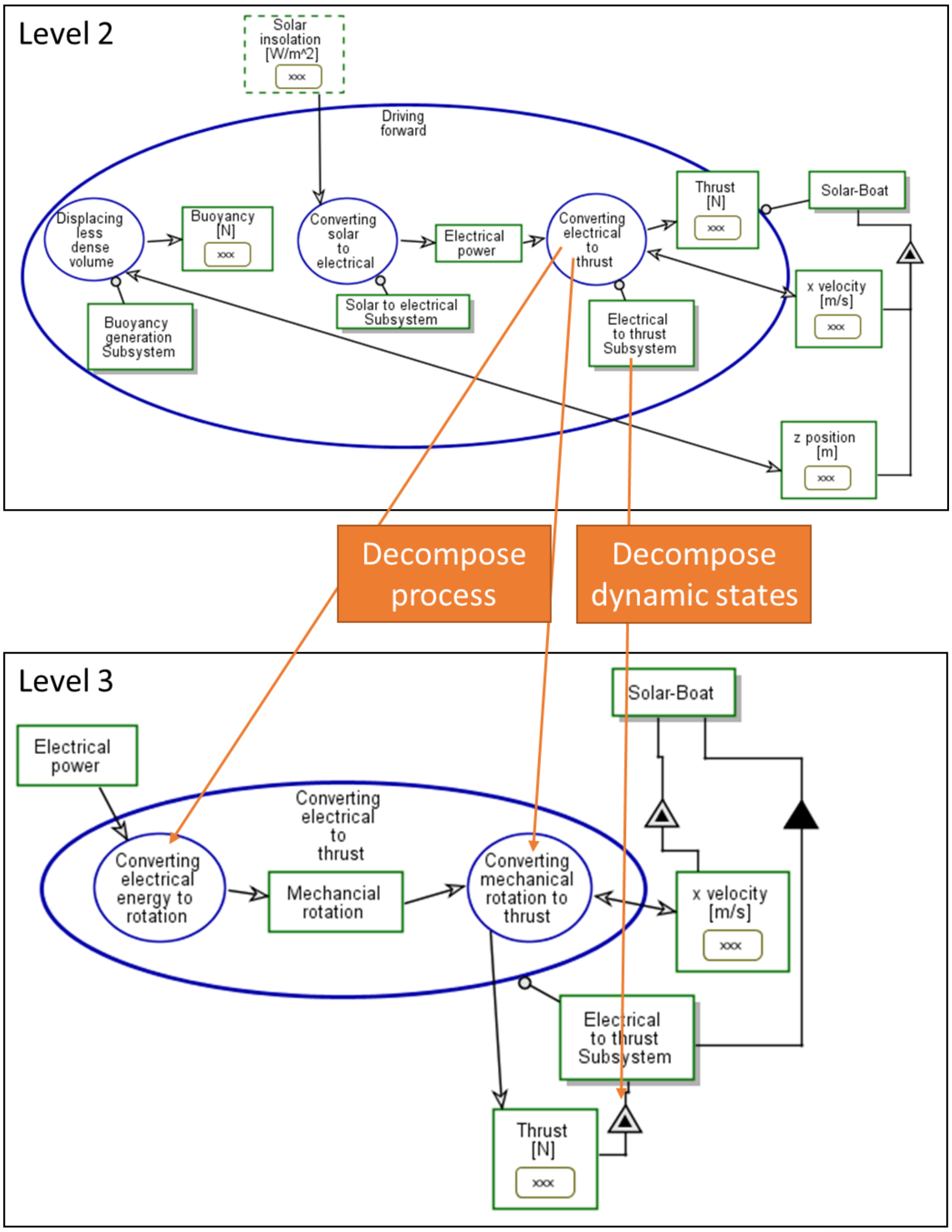


Figure 45 Level 2 to Level 3: Decomposing the System Architecture “Converting electrical to thrust” to a Functional Architecture

4.7.5.B Step E2: Assigning Subsystem-Components to Enable the Processes

Is completed to develop a System Architecture (as in forms which are capable of implementing the processes are introduced). This step is shown in Figure 46.

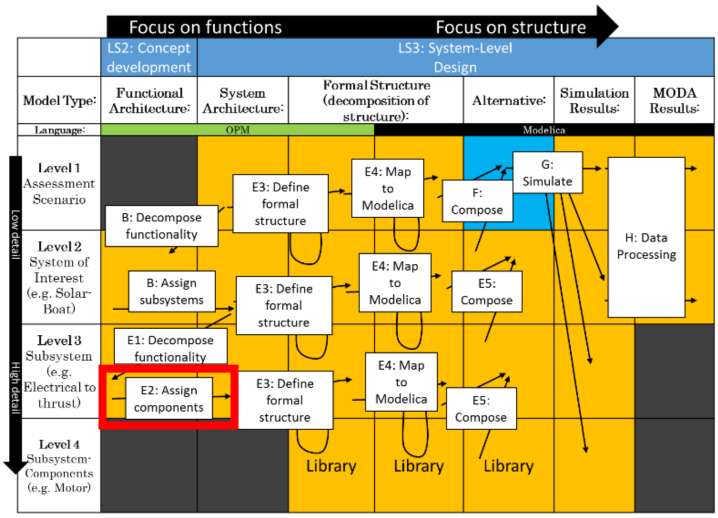


Figure 46 Step E2: Assigning Subsystem-Components to Enable the Processes (miniaturization of Figure 29)

An example of this is shown in Figure 47. A naming convention similar to that used for the naming of objects in Step B should be used (based on the input and output of the process). This step is based on the assumption that each process is enabled by an individual object forming a process and object pair in the System Architecture diagram.

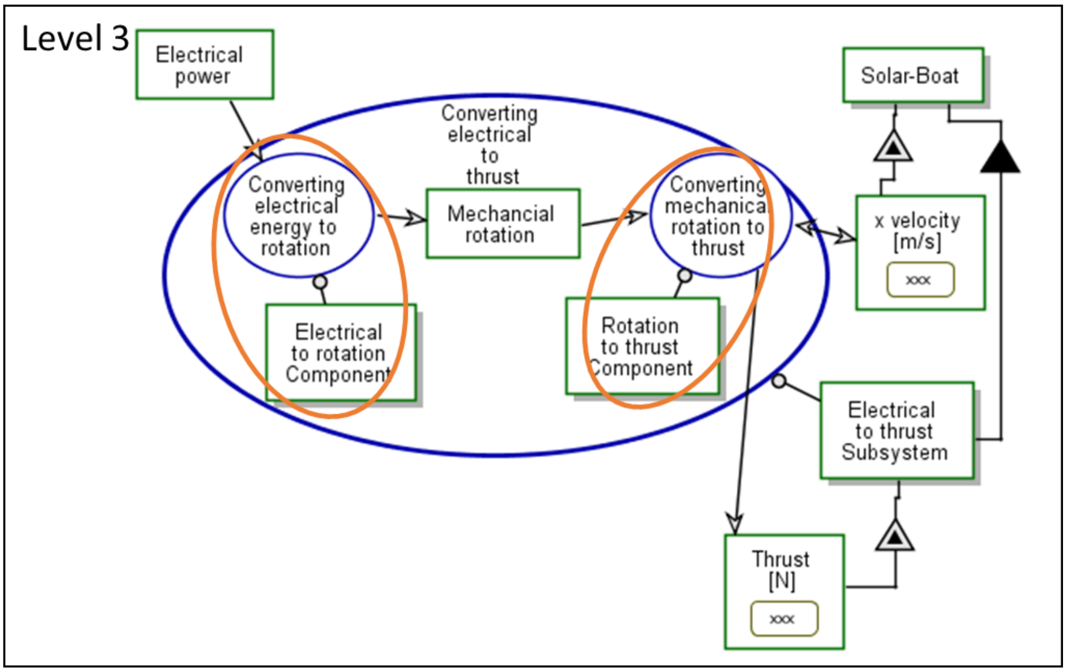


Figure 47 Level 3: Creating System Architecture from Functional Architecture of "Electrical to thrust" subsystem

4.7.5.C Step E3: Mapping to a Formal Structure in OPM

Is required given the System Architecture does not define specifically the connections between the objects such that a numerical model in Modelica can be developed.

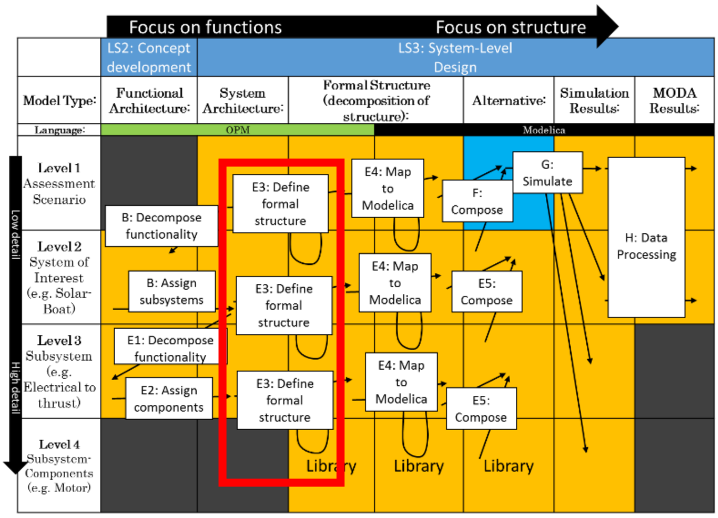


Figure 48 Step E3: Mapping to a Formal Structure in OPM (miniaturization of Figure 29)

To achieve this as per Figure 46, starting at the lowest level of hierarchy each OPM process and object pair of the System Architecture is compared to a library keyed on the process and object pairs input and outputs. This component from the library is then selected and incorporated into the higher level model. As such there is an assumption that there is a sufficient library of Subsystem-Components described as Formal Structure models. Starting at Level 3, initially the Subsystem enabling the super process is in-zoomed with the Subsystem-Components added to the in-zoomed object (shown in Figure 49). In the case of Solar-Boat it is assumed it is a rigid body with all Subsystem-Components rigidly connected to a single position in the Subsystem (and similarly on the higher levels), therefore a rigid "Attachment point" is added.

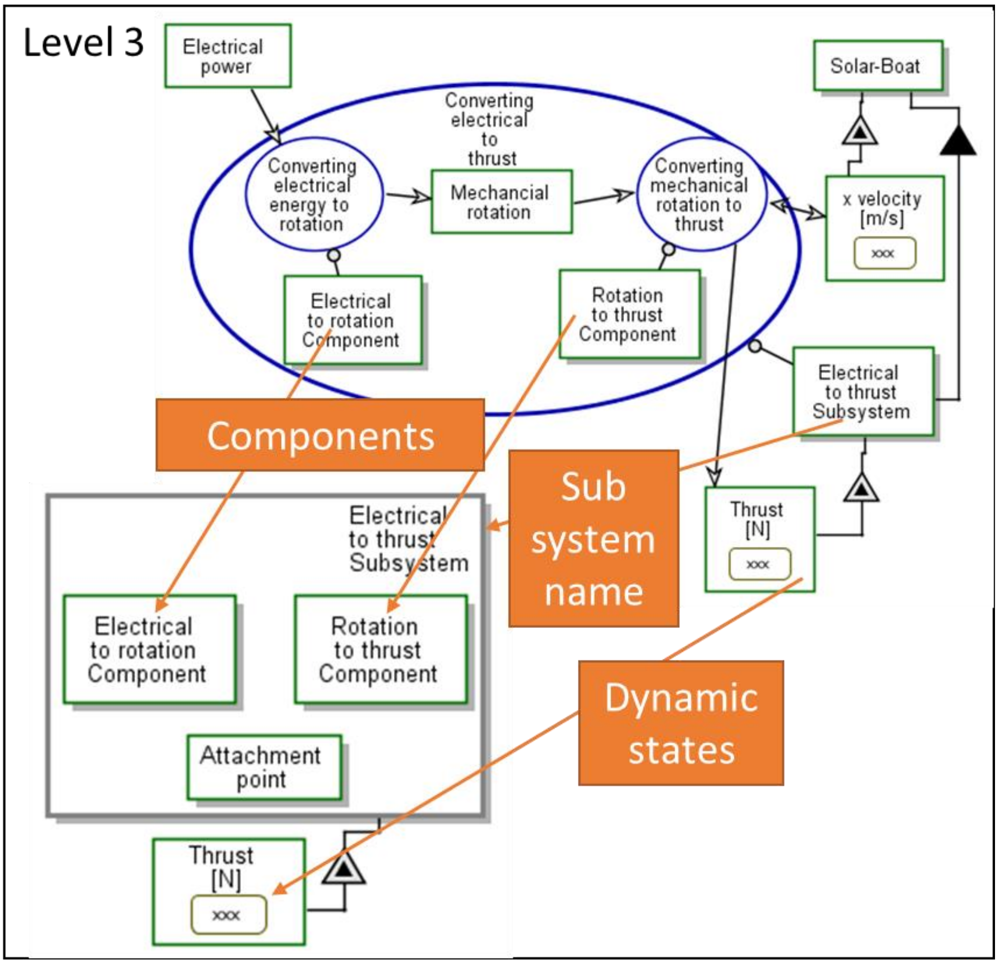


Figure 49 Level 3: Initial Formal Structure for "Electrical to thrust" subsystem

To define the internals of the Subsystem Formal Structure and its additional external interfaces the process and object pairs which define the processes the Subsystem enables must be compared to the library, as each pair is to be represented by a single object in the Formal Structure diagram. This is depicted in Figure 50 and Figure 51. In Figure 50 the process "Converting electrical energy to rotation" is reviewed and Formal Structures from the Level 4 Subsystem-Component library is selected which define the interfaces of the Subsystem-Component (two electrical pins and a rotational flange).

The "Converting mechanical rotation to thrust" process which is enabled by the "Rotation to thrust component" is different as thrust is a force which acts on the entire rigid body and rather than being connected to a connection. As such the existing attachment point and a rotational flange is sufficient (with a configuration to which direction the thrust is acting is) as shown in Figure 51.

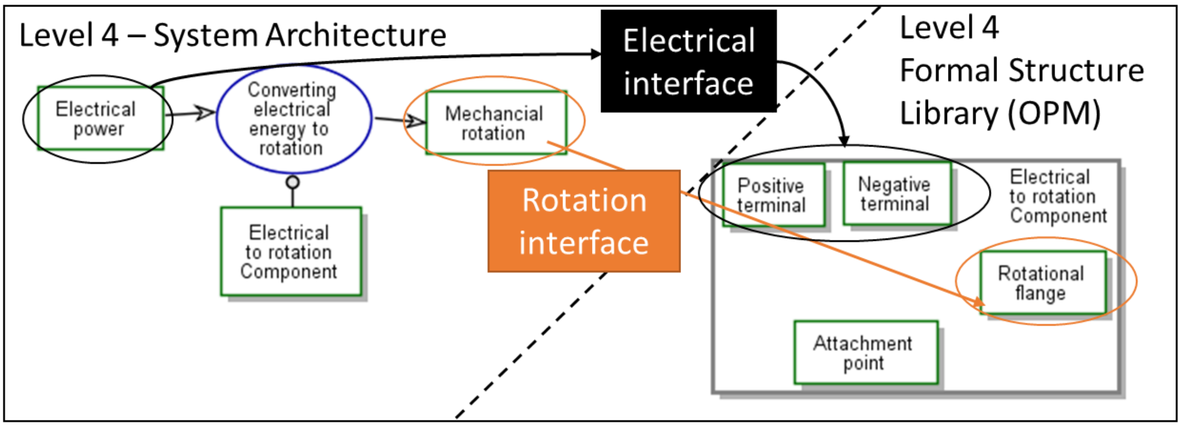


Figure 50 “Converting electrical energy to rotation” Level 3 System Architecture object process pairs being compared to the Level 4 library of subsystem-components

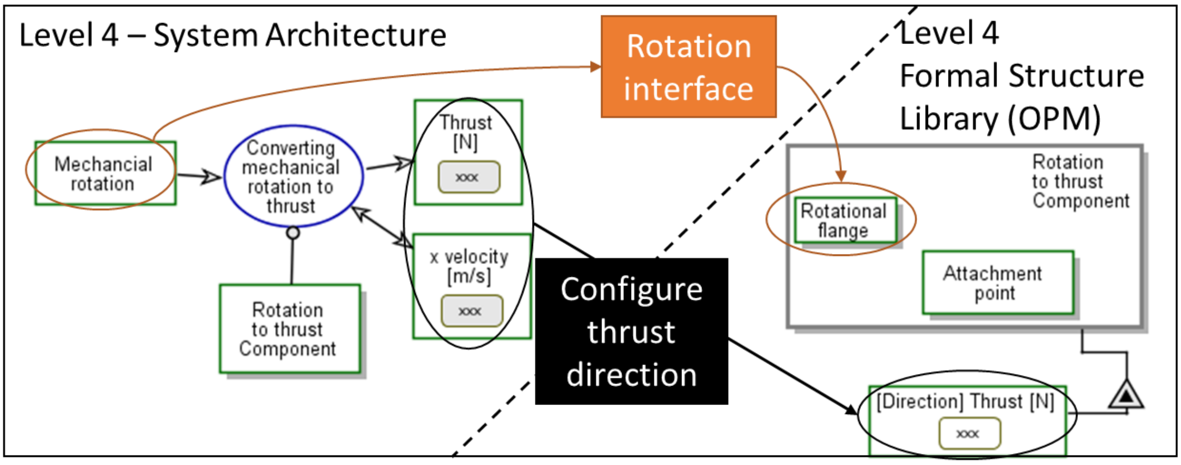


Figure 51 Level 3 System Architecture object process pairs being compared to the Level 4 library of subsystem-components.

The Formal Structure for Subsystem-Components can be used to compose a Formal Structure of the Subsystem, where relational links are defined to be causal (single headed arrow) or acausal (double headed fish hook arrow). With the Subsystem-Components being connected together based on the flow defined in the System Architecture (Figure 47) this is depicted in Figure 52.

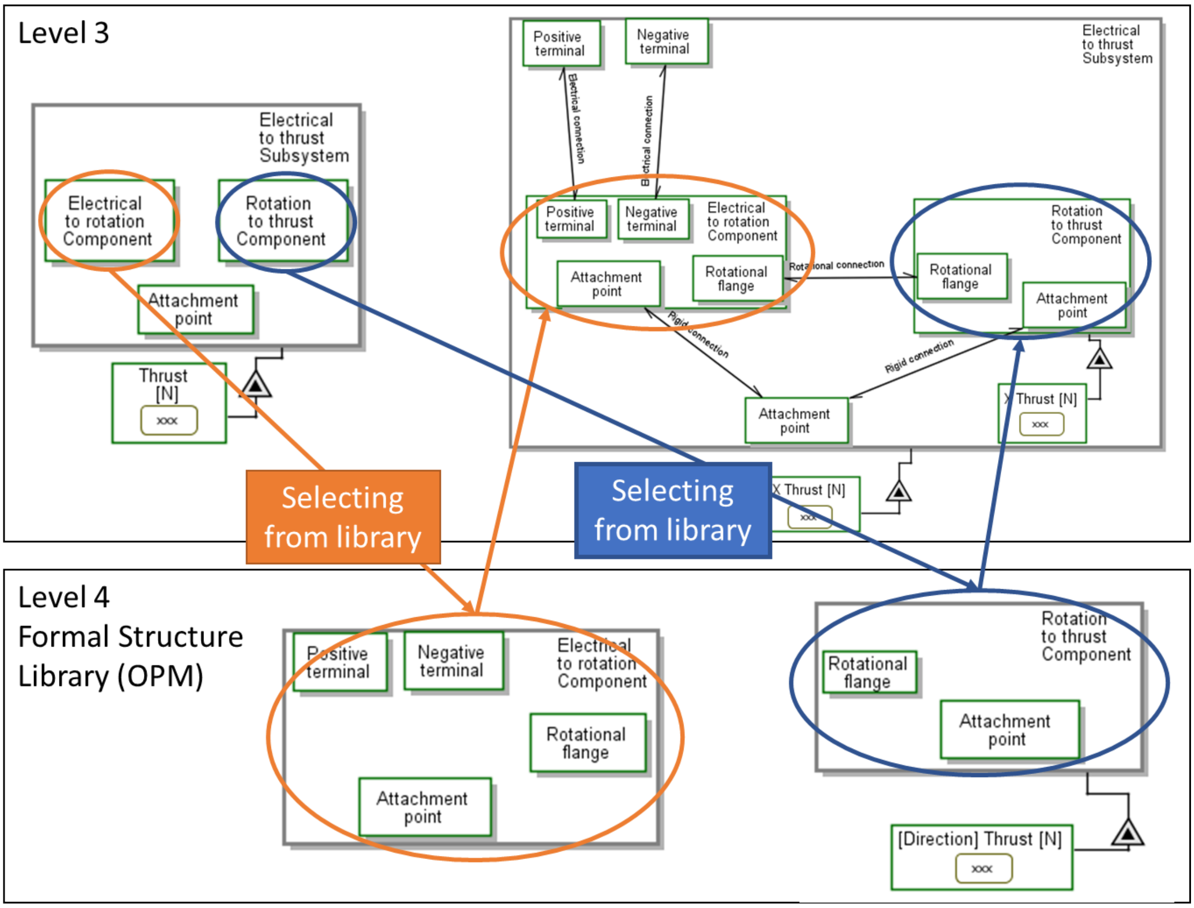


Figure 52 Level 3: Composing a Formal Structure of the "Electrical to thrust" subsystem out of Level 4 subsystem-components

As per Figure 53 the process of defining Formal Structure of one level of hierarchy from the interfaces defined by a lower layer enables Formal Structure for all levels to be created. As such Level 2 is depicted being created in Figure 54.

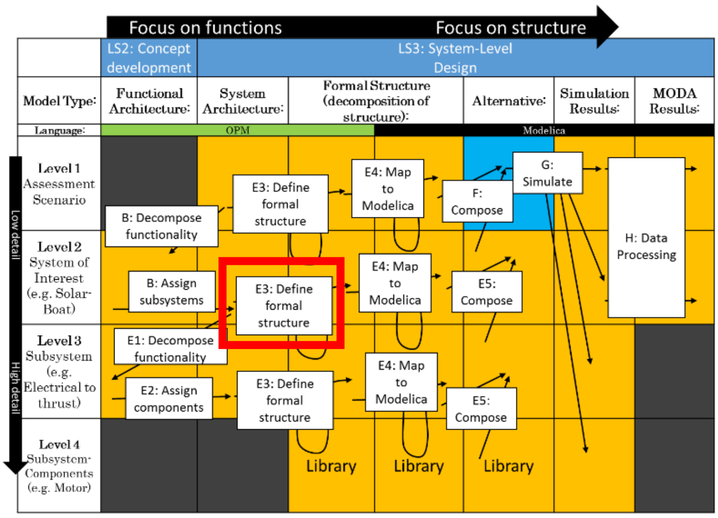


Figure 53 Step E3: Mapping to a Formal Structure in OPM Level 2 (miniaturization of Figure 29)

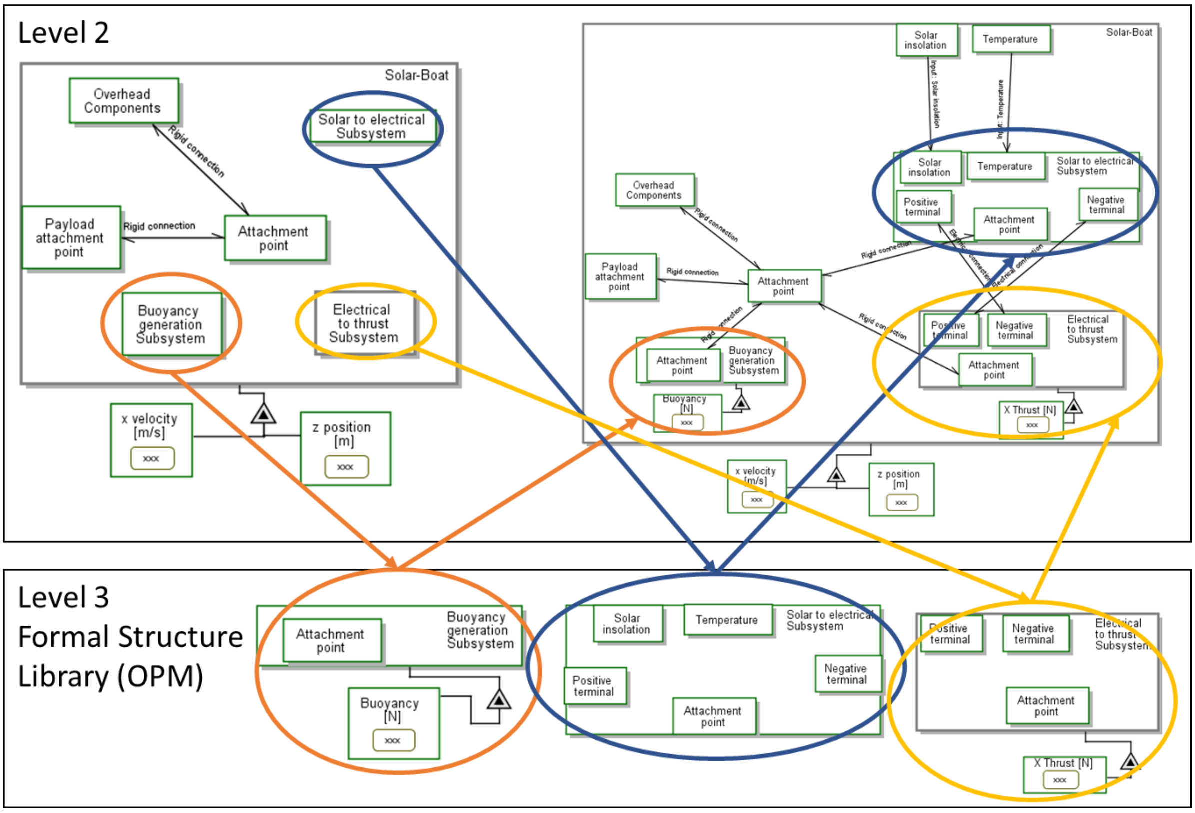


Figure 54 Level 2: Composing a Formal Structure (OPM) of the Solat-Boat out of Level 3 subsystems

4.7.5.D Step E4: Mapping to a Formal Structure in Modelica

This marks the initial start of numerical model creation. Further, as per Figure 55 the process of defining Formal Structure of one level of hierarchy from the interfaces defined by a lower layer enables Formal Structure for all levels to be created.

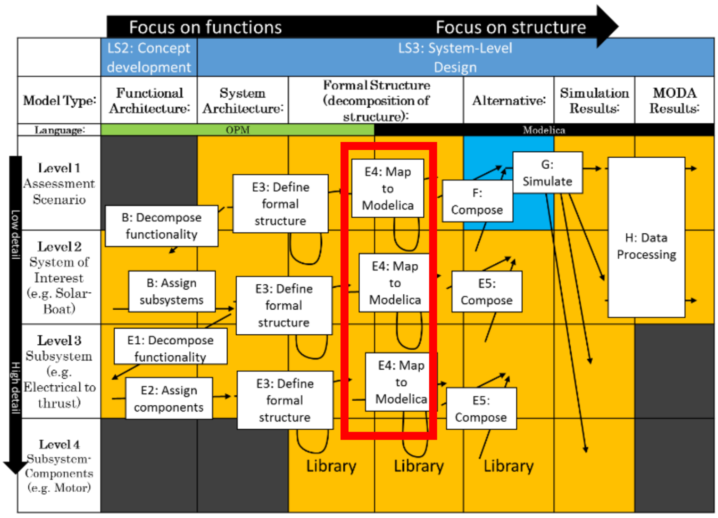


Figure 55 Step E4: Mapping to Formal Structure in Modelica (miniaturization of Figure 29)

The Formal Structure, defined in OPM is used to define Modelica models of Modelica replaceable partial components. By using Modelica replaceable partial components multiple alternative implementations can be created off of a single architecture. As such initially the interfaces for the Subsystem must be defined as depicted in Figure 56 this involves selecting the appropriate connector for the model in Modelica, as such defining in Modelica the interface for the Subsystem. In addition any attributes of interest (x Thrust in Figure 56) are set as publically assessable variables. As per Figure 57 to create a Modelica model for the architecture defined in the OPM Formal Structure the selection of the appropriate Modelica replaceable partial components from the Subsystem-Components library is required.

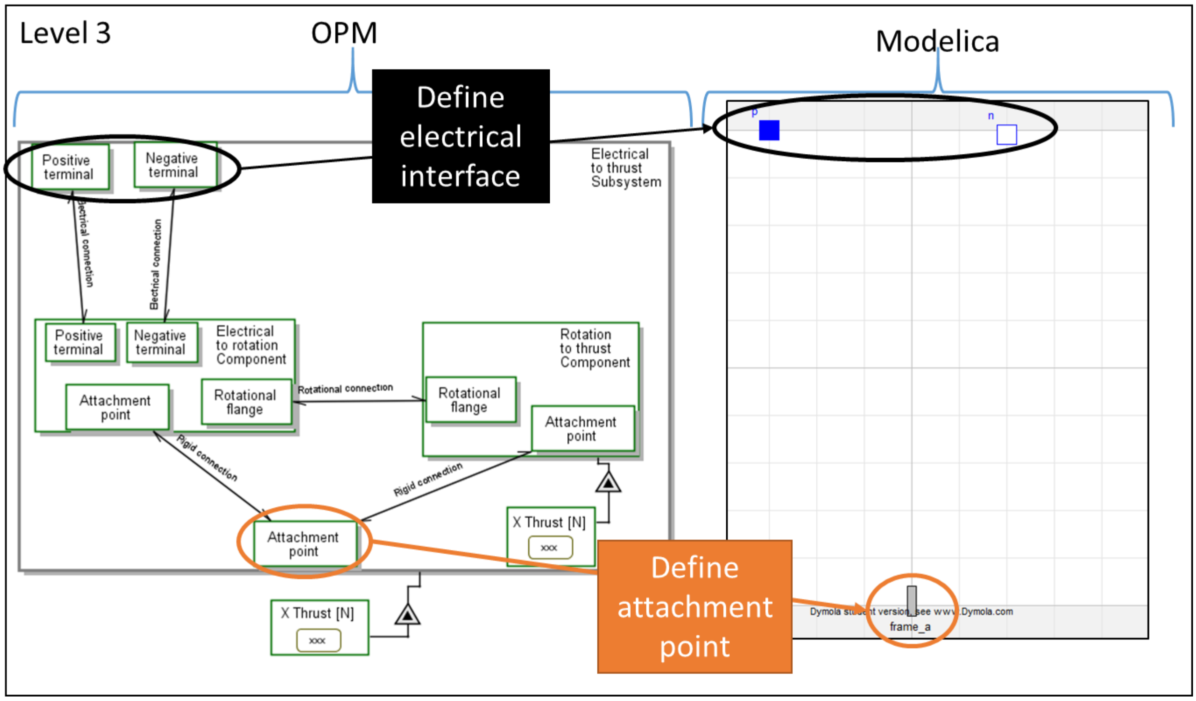


Figure 56 Level 3: Defining connectors for a subsystem in Modelica Formal Structure (Right) from OPM Formal Structure (Left)

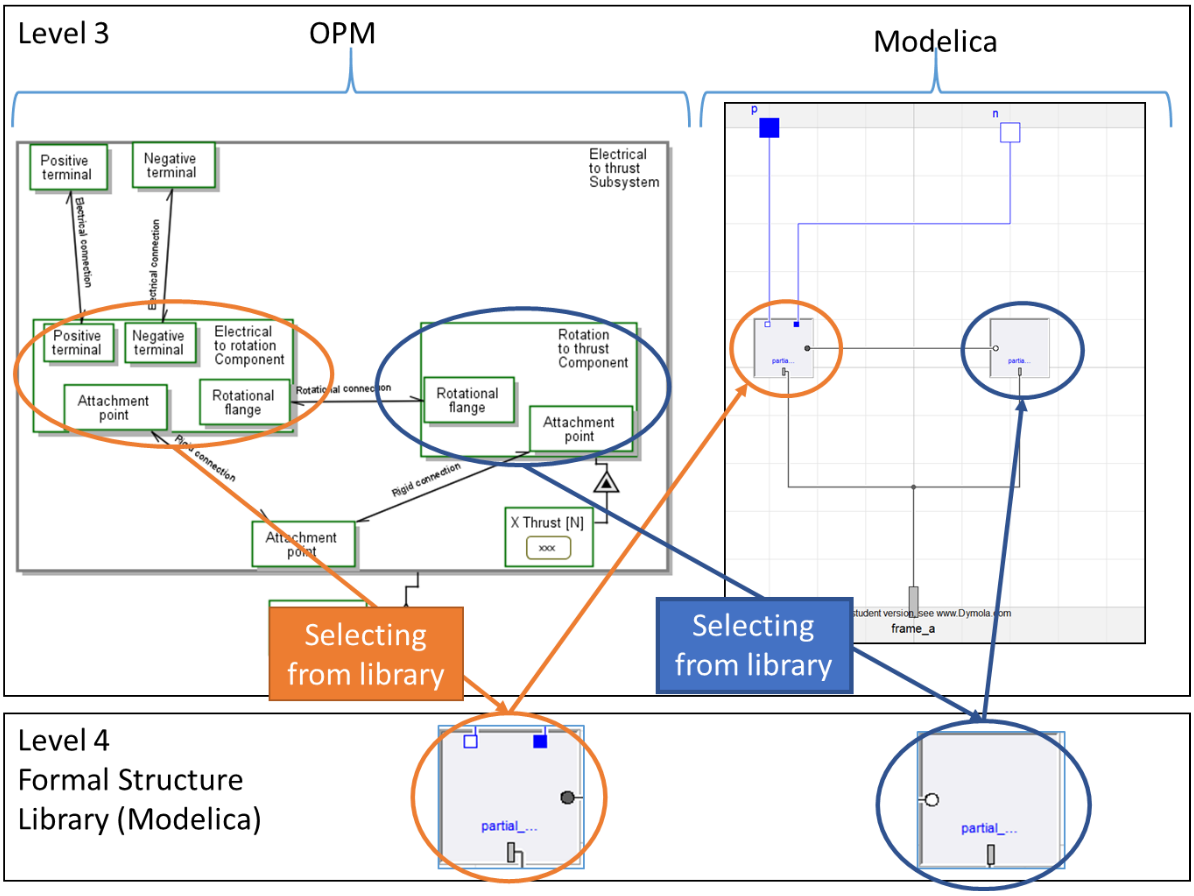


Figure 57 Level 3 Formal Structure: Selecting Modelica replaceable partial components from the subsystem-components library

Once the Level 3 Subsystems are defined, Level 2 can be created (Figure 58) as depicted being created in Figure 59.

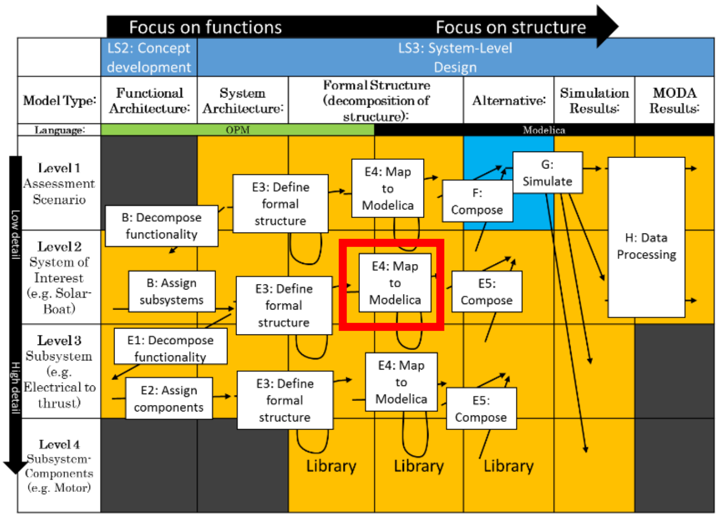


Figure 58 Step E4: Mapping to a Formal Structure in Modelica Level 2 (miniaturization of Figure 29)

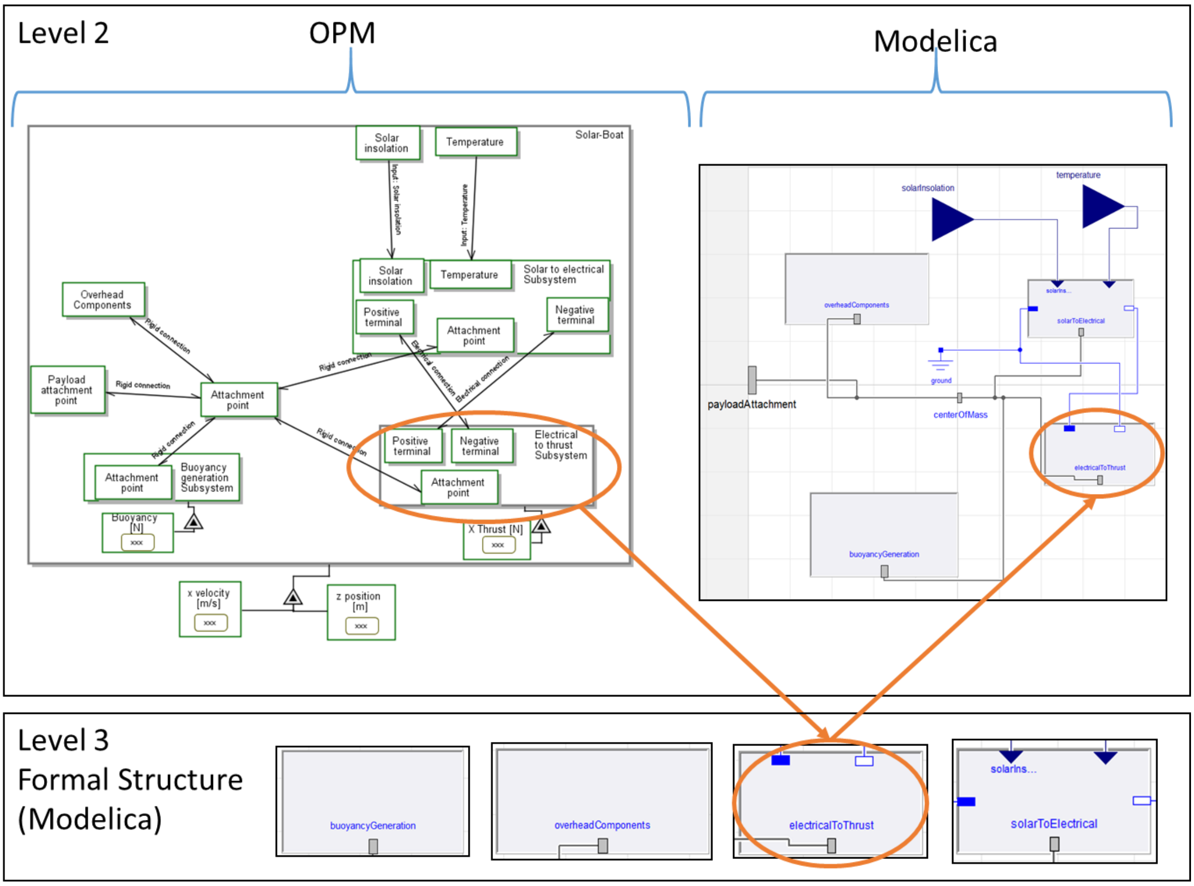


Figure 59 Level 2: Composing a Formal Structure (Modelica) of the Solat-Boat out of Level 3 subsystems

4.7.5.E Step E5: Composing Alternatives in Modelica

Is where alternative designs are created by the composition onto the Formal Structure from and adequate library of Subsystem-Components (Level 4) to form Subsystems (Level 3) and ultimately a System of Interest alternative (Level 2) as shown in Figure 60 and Figure 61.

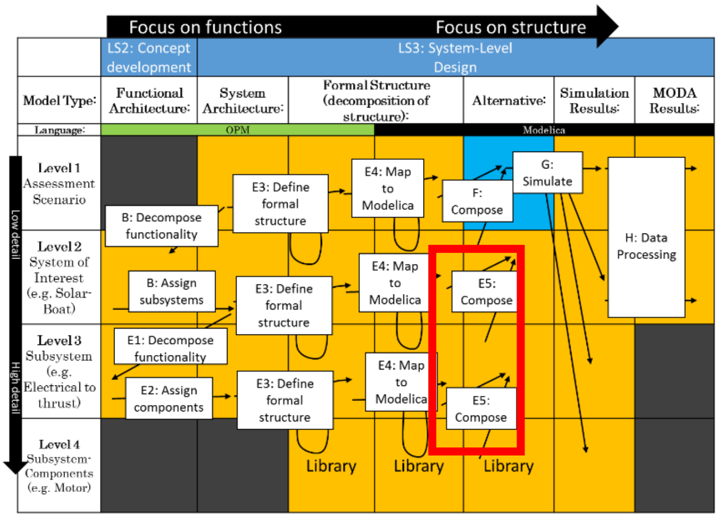


Figure 60 Step E4: Mapping to Formal Structure in Modelica (miniaturization of Figure 29)

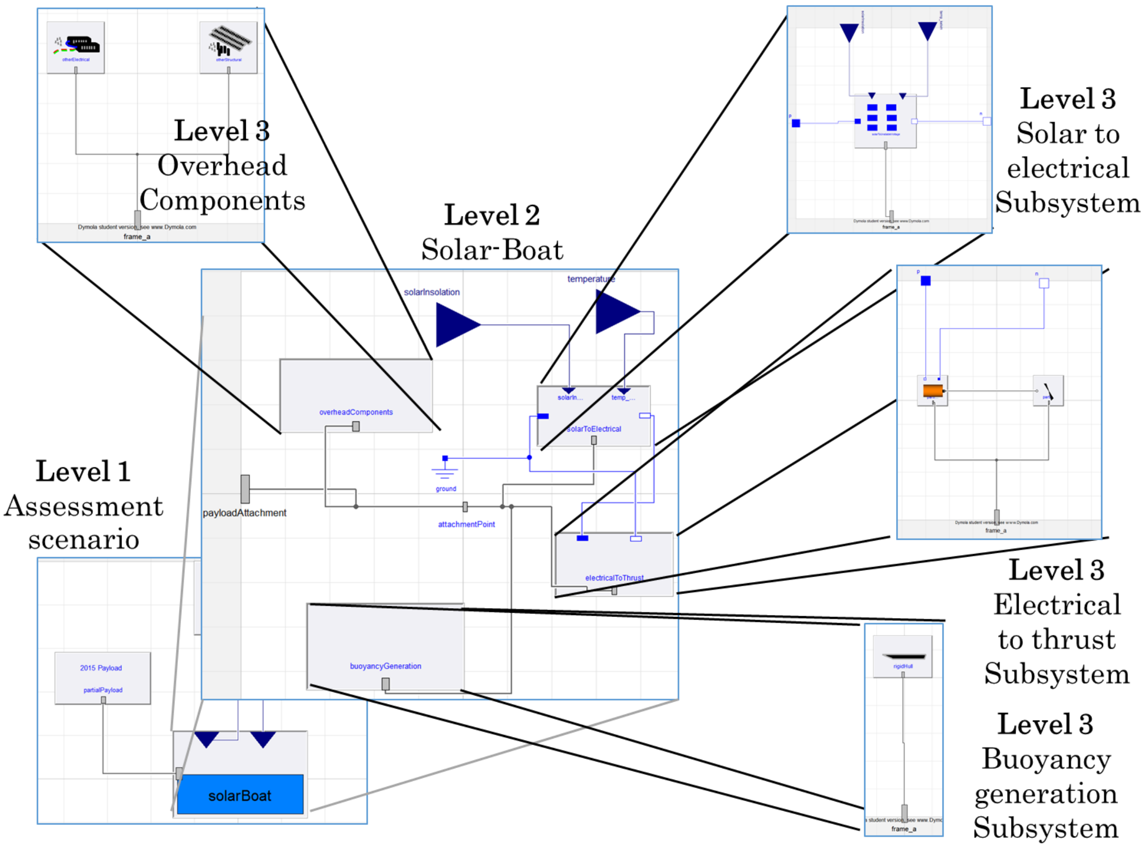


Figure 61 Composing an Alternative System of Interest (Level 2) which is then placed into an Assessment Scenario (Level 1)

4.7.6 Step F: Composing each System of Interest Alternative into each Assessment Scenario for Simulation

As shown in Figure 62 and Figure 61 composing the Assessment Scenario is similar to composing the System of Interest alternatives.

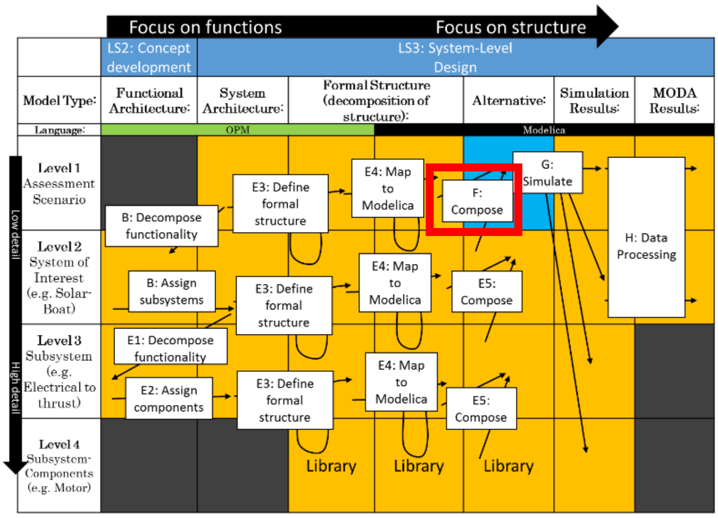


Figure 62 Step F: Composing each System of Interest Alternative into each Assessment Scenario for Simulation (miniaturization of Figure 29)

Table 24 shows how the different System of Interest – Alternatives can be placed in different Assessment Scenarios – Formal Structures.

System of Interest – Alternatives:	Assessment Scenarios:			
	Floating	Straight line average sun	Straight line best sun	Straight line worst sun
Solar-Boat Design A				
Solar-Boat Design B				
Solar-Boat Design C				

Table 24 System of Interest – Alternatives (Level 2) being placed in the Assessment Scenario – Formal Structure (Level 1)

4.7.7 Step G: Simulating every Assessment Scenario and System of Interest Alternative Combination

Given models have been created for each Assessment Scenario and System of Interest Alternative combination they can now be simulated for the simulation length specified in the Assessment Scenarios description. The step is depicted in Figure 63 where interaction is only with the Level 1 Modelica model. The result of which is time series data which explicitly includes the variable of interest specified for the Assessment Scenario.

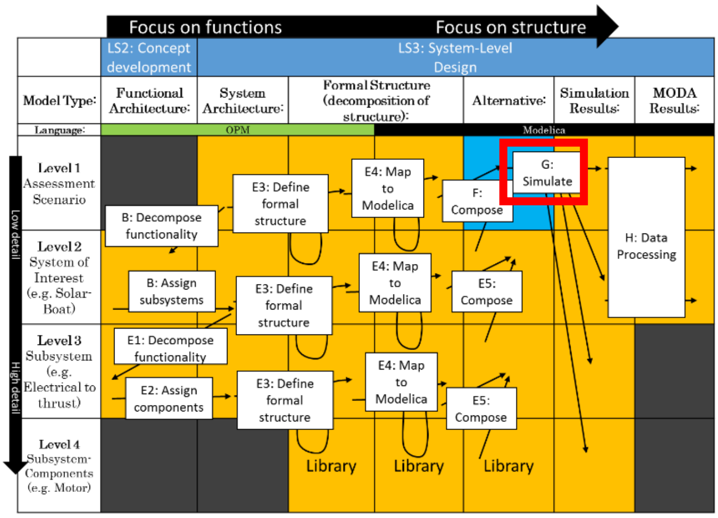


Figure 63 Step G: Simulating every Assessment Scenario and System of Interest Alternative Combination (miniaturization of Figure 29)

4.7.8 Step H: Consolidating Simulation Results with MODA

To provide a fast initial comparison of the performance of the various Alternative System of Interest designs Multi Objective Decision Analysis (MODA) is employed using the value function, weighting and data extraction rules defined in Step D to compute the total weighted value for each alternative. Further this avoids potential overwhelm associated with a large number of simulation results. Description of the MODA method was provided previously in Section 4.6.1.F while Figure 64 depicts the Step, including the extraction of data from Level 2. Example processed results for three Solar-Boat alternative designs (RS_Tokyo2014, RS_Tokyo2015 and RS_KTH2014) and an "Ideal" are assessed for four assessment scenarios (floating, and straight line driving with alternative inputs) is provided in Figure 65. Note that in the legend of Figure 65 the weight associated with each Assessment Scenario is indicated.

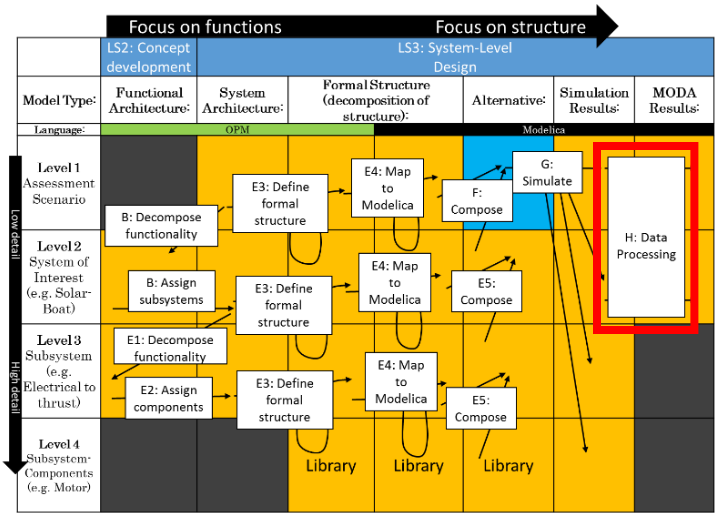


Figure 64 Step H: Processing Simulation Results (miniaturization of Figure 29)

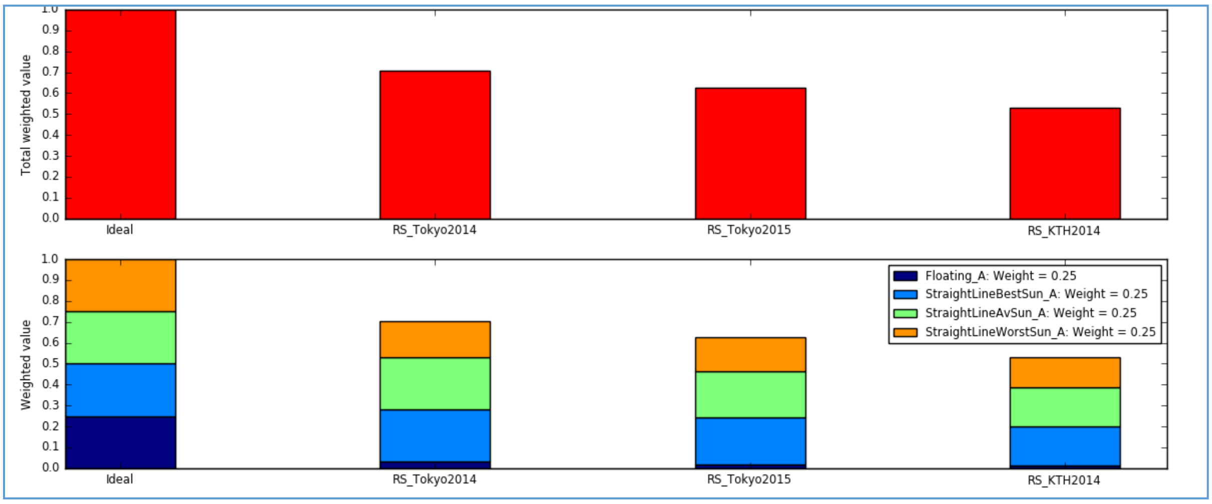


Figure 65 Example MODA results (three Solar-Boats and four assessment scenarios)

Initially assessment of the System of Interest Alternatives with the highest performance can be those with the highest score of the MODA results of Step H. However such results lack the detail into what caused performance to be better or worse for a particular alternative. As such summary for each Assessment Scenario is produced, an example is provided in Figure 66 (straight line average weather) where:

- Plot 1 displays the time series of the attribute of interest (x velocity) for each design alternative (with lines indicating the minimum acceptable performance and ideal performance)
- Plot 2 the extraction from Plot 1 (based on the Data Extraction Method for the Assessment Scenario, in this case maximum) which will be used later to compute value
- Plot 3 the Value Function and the position of each alternative design on it (mapping performance to value)
- Plot 4 the Value contributed by the Assessment Scenario (read from Plot 3)

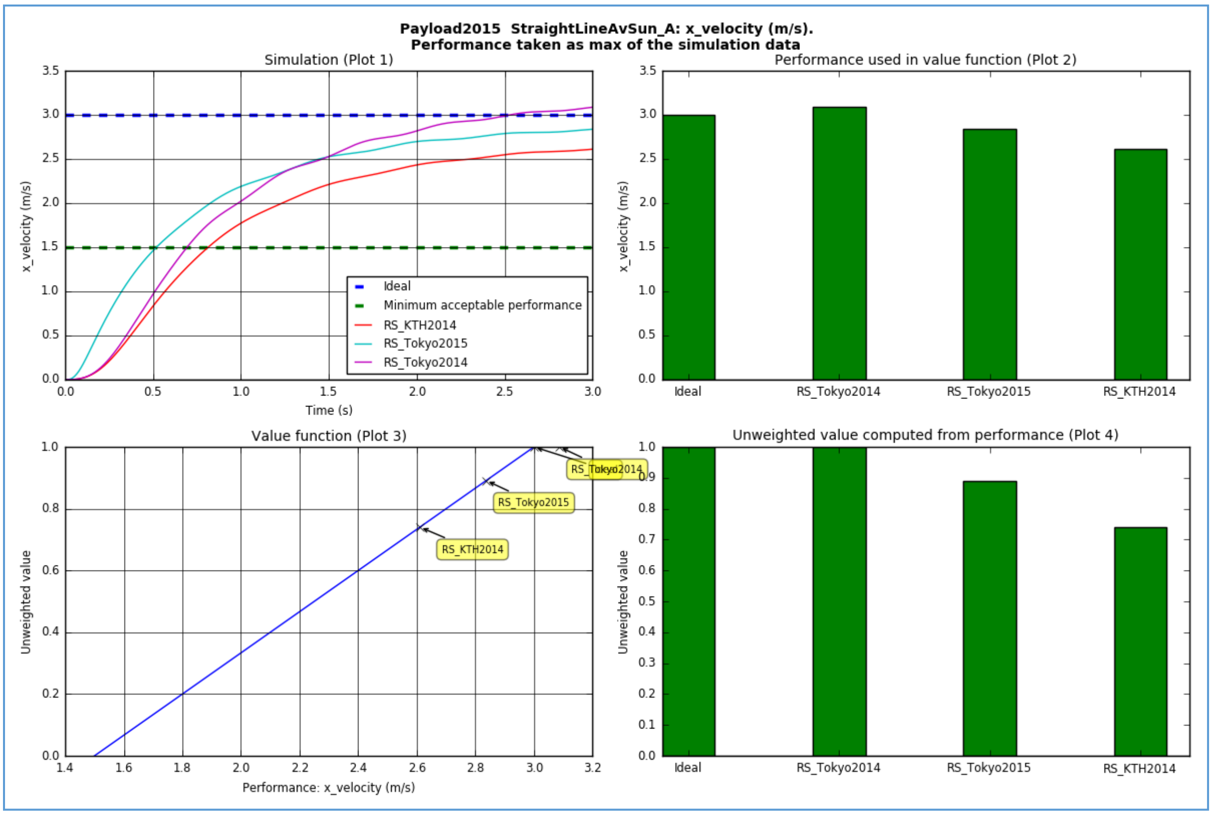


Figure 66 Example comparison of maximum speed during average solar power for three Solar-Boat designs

4.7.9 Step I: Reviewing Results

Initially the MODA processed results can be reviewed (e.g. Figure 65) then the data processed for each Assessment Scenario reviewed (e.g. Figure 66). In addition the extensive results of the Modelica simulations can be reviewed.

4.7.10 Step J: Modify and Repeat or Move to Detailed Design

Depending on the engineers preference, design study purpose and results of Step I, the project might immediately move into detailed design, or one, some or all of the steps of the methodology described might be completed again with the aim of understanding existing alternative designs better (i.e. Assessment Scenario variation) or development of additional alternative designs (e.g. component variation, architecture variation).

4.8 Software implementation used in this research

Implementation of this method in this research was completed as follows:

- Step A -> Step E3: OPCAT software (D. Dori, Linchevski, Manor, & Opm, 2010) was used to manually describe the System and Assessment Scenarios
- Step E4 -> Step E5: Were completed using commercial Modelica implementation Dymola; defining Modelica models of the System of Interest and the Formal Structure of Assessment Scenario
- Step F-> Step H: Were automated by using custom python scripts. Figure 67 provides an overview of the automation scheme. Description of which are provided in subsequent sections.

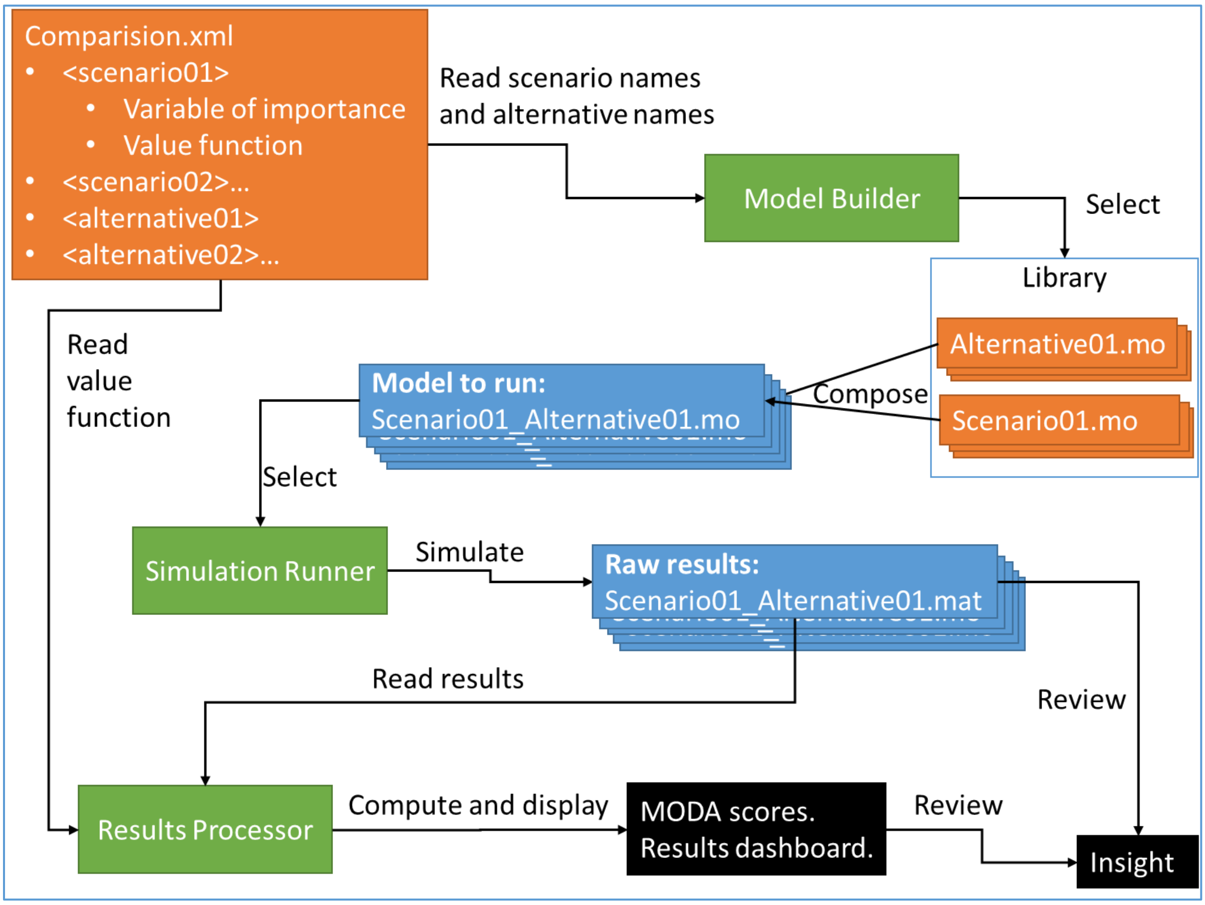


Figure 67 Automating Step F -> Step G: Green: System processing elements. Orange: Inputs. Blue: Intermediate results. Black: Final results.

Figure 67 has three important processing elements (in green) listed as: Model Builder (Step F), Simulation Runner (Step G) and Results Processor (Step H). The required initial inputs (in orange) take the form of Comparison.xml detailing what Assessment Scenarios and alternative System-Level Designs to consider and a library of Modelica models which are the Assessment Scenarios and alternative System-Level Designs referenced by the

Comparison.xml. With the Assessment Scenario describing how to assess a design alternative subject to a set of stated conditions.

The intermediate results are the combined models ready to be simulated and the .mat files the results of these simulations when themselves can be reviewed or are combined by way of MODA to create consolidated results.

4.8.1 Input – Comparison.xml

Comparison.xml is an input to the approach, it is an XML file listing and describing what Assessment Scenarios to complete (including time to simulate and how to processes its results) followed by a listing of alternative System-Level Designs to assess. A code snippet is provided in Figure 68 of an example file (truncated and modified for simplicity). Where three Assessment Scenarios and two alternative designs are listed (plus the ideal).

```
<system_name="solarBoat"/>
<scenario name="Floating">
    <variable="z_top_of_hull"/>
    <variable_units name="m"/>
    <value_func_direct name="neg"/>
    <min_accep_perform val="-0.1"/>
    <stretch_goal val="-0.4"/>
    <weight val="0.5"/>
    <sim_length val="70"/>
    <extract_data_type name="mean"/>
</scenario>
<scenario name="StraightLineAvSun">
    <variable="x_velocity"/>
    <variable_units name="m/s"/>
    <value_func_direct name="pos"/>
    <min_accep_perform val="1.5"/>
    <stretch_goal val="3"/>
    <weight val="0.5"/>
    <sim_length val="3"/>
    <extract_data_type name="max"/>
</scenario>
<scenario name="Cost">
    <variable="cost_money"/>
    <variable_units name="yen"/>
    <value_func_direct name="neg"/>
    <min_accep_perform val="300000"/>
    <stretch_goal val="0"/>
    <sim_length val="1"/>
     <extract_data_type name="max"/>
</scenario>
<design name="Ideal"/>
<design name="Boat_Alternative_01"/>
<design name="Boat_Alternative_02"/>
```

Figure 68 Example Comparison.xml (truncated and modified for simplicity)

4.8.2 Input – Library

The library of Modelica models described in previous sections was created by manually varying the Subsystem-Components and Subsystems on the Formal Structure to create Alternative Solar-Boats (System of Interest) for assessment. While the Scenarios were created integrating the Solar-Boats.

4.8.3 Processing – Model Builder

The Model Builder processing element generates a Modelica model for each combination of Assessment Scenario and System of Interest (Solar-Boat) alternative described in the Comparison.xml file. The Model Builder requires that the Assessment Scenarios and System of Interest (Solar-Boat) alternatives named in the Comparison.xml are available from the library. This is achieved programmatically by duplicating existing model for the Assessment Scenario and manipulating the .mo text file to change the Solar-Boat alternative to the one for assessment.

4.8.4 Processing – Simulation Runner

Simulation Runner subsequently simulates all the models created by Model Builder for the simulation length specified in the Comparison.xml file. This is achieved programmatically by utilizing Dymola's python interface. The subsequent results (in the .mat file) can then be further reviewed by the engineer if they wish.

4.8.5 Processing – Results Processor

The results processor extracts for each model simulated (Assessment Scenario and design alternative pair) the time series of the raw simulation results the variable of interest for each Assessment Scenario to measure the System of Interest's performance (e.g. max x_velocity). This extraction is enabled by Dymat python package (DyMat, 2015).

4.9 Past knowledge being captured in models

Table 25 (first three lifecycle stages) and Table 26 (final four lifecycle stages) indicate where knowledge potentially can now be stored (including in OPM and Modelica models). These should be contrasted with Table 7 and Table 8 of the previous projects. Comparing the first three lifecycle stages it is possible to see much of the tacit knowledge can now be stored in OPM models which makes it more transparent and accessible. Further at Lifecycle Stage 3 where much knowledge was stored as Excel spreadsheets Modelica models have now taken this role. Further Table 27 provides a summary of what knowledge can now be stored in OPM and Modelica and Figure 69 briefly shows how Project and Race information can be stored in OPM diagram. Further Figure 70 presents a visualization of how by having model based knowledge at the core of the project the various lifecycle stages can be enabled by consuming knowledge and subsequently generate new knowledge.

		Knowledge locations:					
		Documents (Word, Power Point...)	Models				Physically realized components
Life cycle stage:	Knowledge type:		OPM	Modelica - Partial	Modelica - Complete	CAD	
LS1: Clarify	SolarBoat Project - Procedures and conventions	Y					
	Race - Race rules and constraints		Y				
	SolarBoat Project - Resources		Y				
LS2: Concept dev	SolarBoat Project - Project intension		Y				
	SolarBoat - Functional architecture (behavior)		Y				
	SolarBoat Project - Assessment scenarios		Y	Y	Y		
LS3: System-Level Design	SolarBoat - Systems Architecture		Y	Y	Y		
	SolarBoat - Subsystems		Y	Y	Y		
	SolarBoat Project - Systems-Level Design Simulation models and results	Y			Y		

Table 25 Proposed knowledge locations of the first three lifecycle stages of the Solar-Boat project

Life cycle stage:	Knowledge type:	Knowledge locations: Documents (Word, Power Point...)	Models: OPM	Models: Modelica - Partial	Models: Modelica - Complete	Models: CAD	Physically realized components
LS4: Detail Design	SolarBoat - Detail Design					Y	Y
	SolarBoat Project - Detailed Design - Simulation models and results	Y			Y	Y	
LS5: Production, Test and Refinement	SolarBoat Project - Manufacturing procedures	Y					
	SolarBoat Project - Experiment and deployment results (physical testing)				Y		
LS6: Race	SolarBoat Project - Race Results	Y					
LS7: Knowledge transfer	SolarBoat Project - What to review first?	Y					

Table 26 Proposed knowledge locations of the last four lifecycle stages of the Solar-Boat project

		OPM		Modelica	
Object	Sub name	Processes	Objects	Partial models	Model instance
Race	Race rules and constraints		Y		
SolarBoat Project	Resources		Y		
	Project intension		Y		
	Assessment scenarios	Y	Y	Y	Y
	Systems-Level Design Simulation models and results				Y
	Detailed Design – Simulation models and results				Y
	Experiment and deployment results (physical testing)				Y
SolarBoat	Functional architecture (behavior)	Y	Y		
	Systems Architecture	Y	Y	Y	Y
	Subsystems	Y		Y	Y

Table 27 Summary of knowledge stored in OPM and Modelica

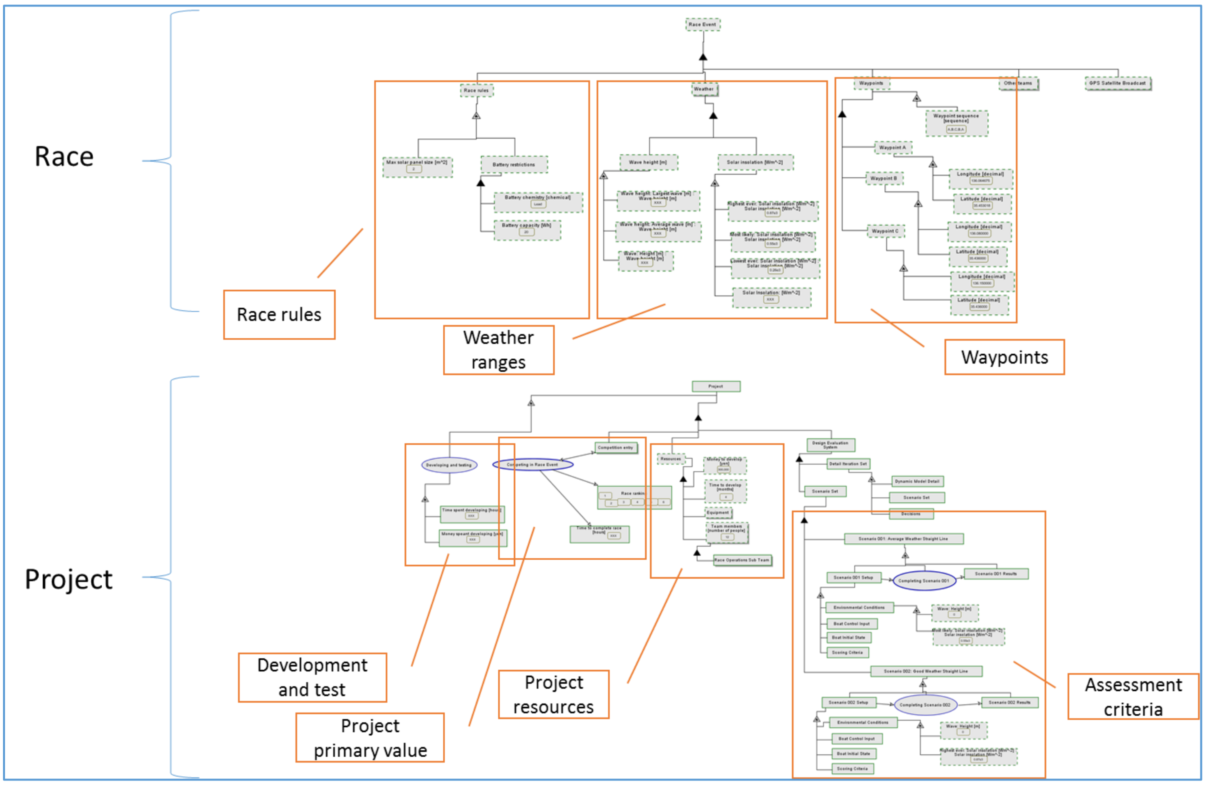


Figure 69 Capturing more project knowledge in OPM diagrams

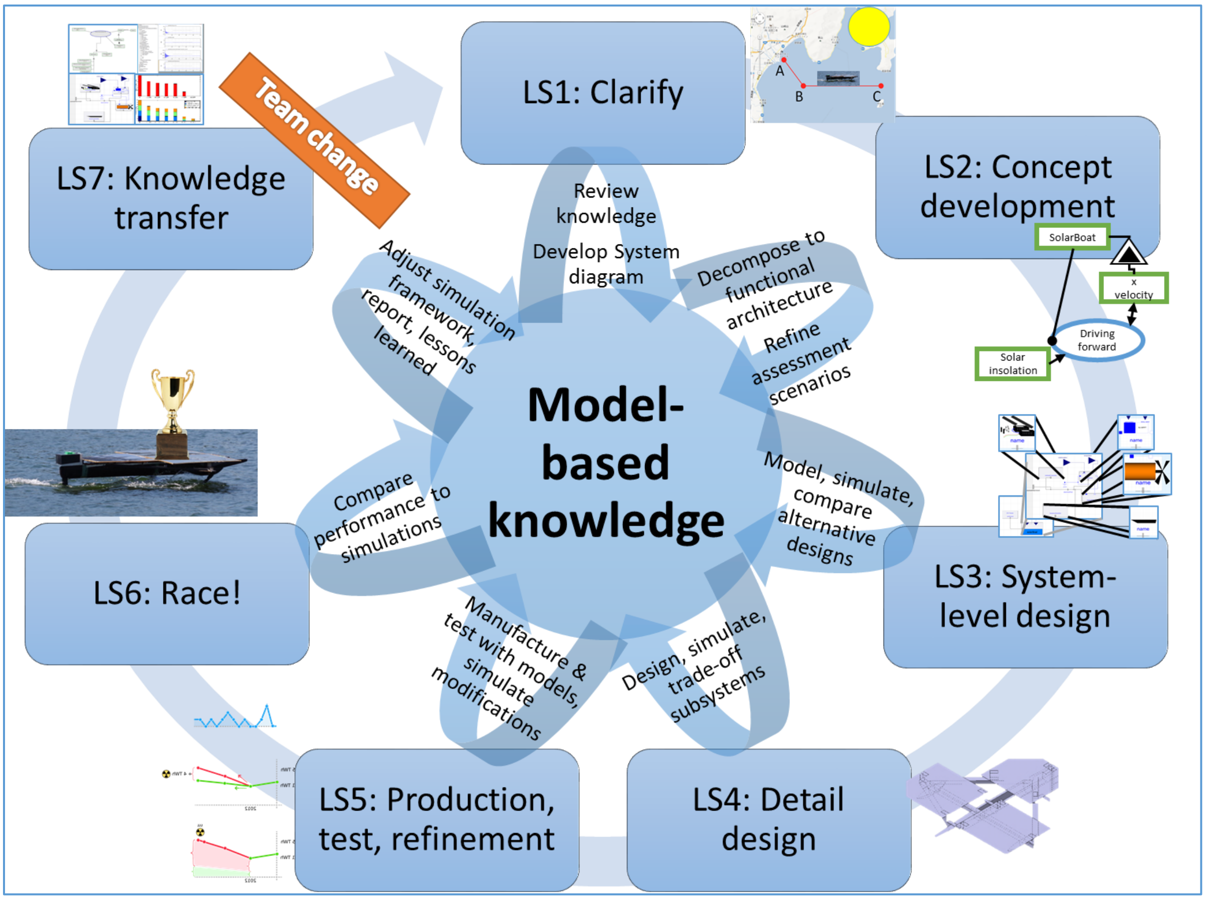


Figure 70 Visualization of how the lifecycle stages both consume and generate knowledge from the model based store which is positioned at the core of the project.

5 Usage examples:

5.1 Chapter purpose

This chapter aims to provide the reader with:

- Examples of the usage of the system proposed such that an understanding of how the proposed methodology can aid the selection of a System-Level Solar-Boat design and addresses the problems identified in the early sections

It is divided into two parts:

- Section 5.2 develops a Solar-Boat design subject to the standard race rules payload of 0.064kg, the design is developed over three design steps
- Section 5.3 assumes that the required payload to be carried is increased by 15kg. The most promising designs of Section 5.2 are reviewed subject to this payload. Subsequently the designs are updated.

5.2 Develop initial Solar-Boat design for prototype for standard rules

Following on from the example developed in Section 4.7, several designs are assessed subject to standard rules of a payload of 0.064kg. Based on the assessment the designs are updated.

5.2.1 Design Set 1 (Subsystem-Component variation: compare a heavy motor to low mass one with different propellers)

Steps A, B and C are assumed to be completed as per the description presented previously in this paper. As such: Step A, The Functional Architecture – Primary Value has been identified (Figure 37) and decomposed (Figure 38); Step B, resulted in the decision to model Solar-Boats capable of performing "Floating" and "Driving forward" processes. Decomposition of these into Functional Architecture and object assignment to develop System Architecture identifies the need for "Buoyancy generation", "Solar to electrical" and "Electrical to thrust" subsystems (Figure 42); Step C, Selection of "Floating" and "Driving forward" as Assessment Scenarios (Figure 43) is made, given all Solar-Boat alternative models are created to perform these processes.

Step D is performed defining the specific Assessment Scenarios shown in Table 28 which for simplicity are assumed they all have an equal weight (i.e. 0.25). In Table 28 one variant of the "Floating" and three "Driving forward" specific Assessment Scenarios by varying the incoming solar insolation. Minimum acceptable performance and stretch goals can be set based on past experience and required performance.

Assessment Scenario name	Measure of interest	Minimum acceptable performance	Stretch goal	Sim time (s)	Data extraction type
Floating	z position (m)	-0.1	-0.4	70	Mean
Best ever insolation (870 Wm^2) straight line driving	x velocity (m/s)	2	4	3	Max
Average insolation (550 Wm^2)	x velocity (m/s)	1.5	3	3	Max
Worst ever insolation (260 Wm^2)	x velocity (m/s)	0.5	2.5	3	Max

Table 28 Specific Assessment Scenarios for initial Solar-Boat design

Step E results in various Solar-Boats being synthesized by the utilization of OPM and Modelica as described in the following sub-steps:

- **Step E1** requires the decomposition of the System of Interest's System Architecture processes into Subsystem Functional Architecture to identify functionality. Previously "Converting electrical to thrust" was decomposed into two processes (Figure 45). The other System of Interest processes "Displacing less dense volume" and "Converting solar to electrical" are decomposed into a single processes each such that the hierarchy's consistency is maintained.
- **Step E2** results in a System Architecture by the assignment of Subsystem-Components to enable the processes identified in the Subsystems. For the "Electrical to thrust" Subsystem these are "Electrical to rotation" and "Rotation to thrust", while "Buoyancy generation" and "Solar to electrical" Subsystems have components "Buoyancy generation" and "Solar to electrical" respectively due to single processes decomposition.
- **Step E3** involves the creation of a Formal Structure in OPM. Which as such requires a review of the Subsystem-Component library to identify the interfaces used by the Subsystem-Components. As shown previously in Figure 52 for "Electrical to thrust" Subsystem in addition to the standard "Attachment point" two electrical terminals are required. "Solar to Electrical" Subsystem requires a solar insolation connection, two electrical terminals, "Attachment point" and a temperature connection. The

temperature connection is only introduced at this point as the modeler did not consider it when completing the Functional Architecture decomposition, but given the Level 4 model contains the temperature connection it is incorporated. While "Buoyancy generation" Subsystem only requires the standard "Attachment point". These Level 3 Formal Structures are subsequently combined to create Level 2 (Figure 54) and Level 1 Formal structures (Figure 33).

- **Step E4** as per the previous sections results in the creation of a Formal Structure in Modelica which various design alternatives can be composed starting with library Modelica replaceable partial components from Level 4.
- **Step E5** is where various alternative Solar-Boats are composed from various alternative Subsystems created by the variation of Subsystem-Components in the Modelica Formal Structure. For the study the "Buoyancy generation" and "Solar to electrical" subsystems are fixed (no hull or solar panel array variation respectively). The "Electrical to thrust" subsystem is varied however by varying the components used as "Electrical to rotation" and "Rotation to thrust", i.e. a different motor and propeller are used. Six different "Electrical to thrust" subsystems are proposed utilizing two different motors (Low Mass-LM and High Mass-HM) and three different two blade propellers (160mm, 200mm and 220mm diameters). Resulting in six different "Electrical to thrust" subsystems which with the single choices of "Buoyancy generation" and "Solar to electrical" subsystems results by way of composition on the Level 2 Modelica Formal Structure six Solar-Boat alternatives (which are named after the motor and propellers they use). All are assumed to have the same overhead components (masses representing what is not already modeled). They are all detailed in Table 29.

Alternative	**Buoyancy Generation**	**Solar To Elec**	**Elec To Thrust**	**Overhead components**
HM_160mm	Single hull	FT-136SE	H motor: No gearbox: 160mm prop	0.5 + 2.3 kg
HM_200mm	Single hull	FT-136SE	H motor: No gearbox: 200mm prop	0.5 + 2.3 kg
HM_220mm	Single hull	FT-136SE	H motor: No gearbox: 220mm prop	0.5 + 2.3 kg
LM_160mm	Single hull	FT-136SE	L motor: No gearbox: 160mm prop	0.5 + 2.3 kg
LM_200mm	Single hull	FT-136SE	L motor: No gearbox: 200mm prop	0.5 + 2.3 kg
LM_220mm	Single hull	FT-136SE	L motor: No gearbox: 220mm prop	0.5 + 2.3 kg

Table 29 Alternative Solar-Boat designs created by electrical to thrust Subsystem variation (H = High mass motor, L = Low mass motor)

Steps F and G results in the six Solar-Boat alternatives being composed into the four specific Assessment Scenarios Formal Structure (Step F) resulting in twenty four separate models which are simulated (Step G).

Steps H and I are completed consolidating the time series results from the twenty four simulation runs by way of MODA (Step H) results of which are displayed in Figure 71, where the total weighted value for each alternative (x-axis) is the total height of the column which is broken down into contributions by each Assessment Scenario. In Figure 71 alternative “HM_160mm” is shown to be performing significantly better than the other alternatives. As such, a review of the Simulation Results is conducted where the other alternatives are found to have lower speeds (Figure 72) due to lower thrust (Figure 73). Further investigation into Subsystem-Components reveals the angular velocity of the low mass motor is far from nominal speed of the motor when compared to the high mass motor (Figure 74). However the low mass designs do perform better in the floating scenario.

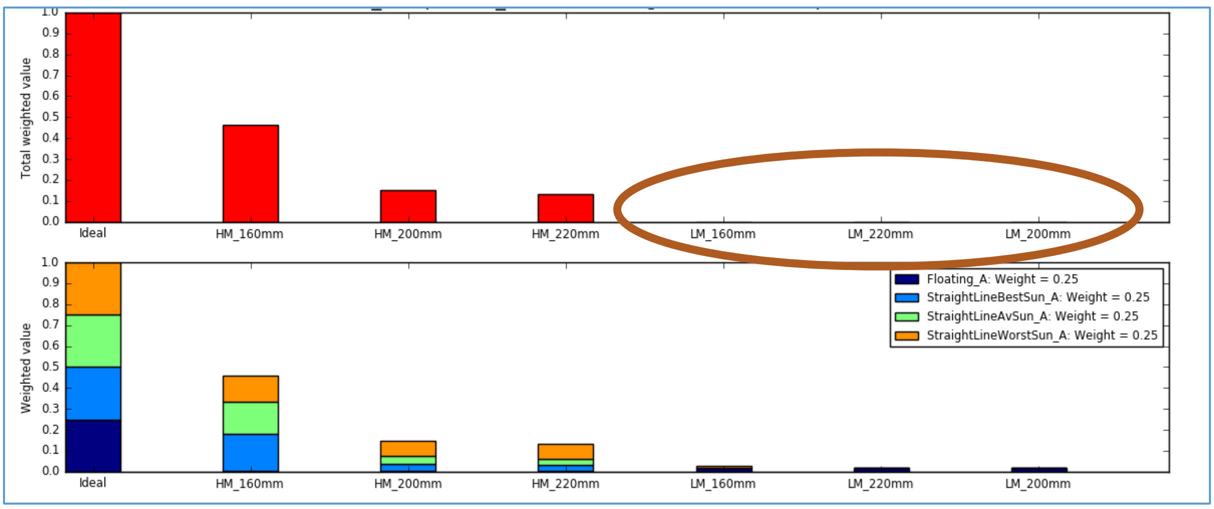


Figure 71 MODA processed results from simulation varying motors and propellers. Top: Sum of weighted value (zero indicates failing one or more Assessment Scenarios, ringed). Bottom: Value contribution of each Assessment Scenario

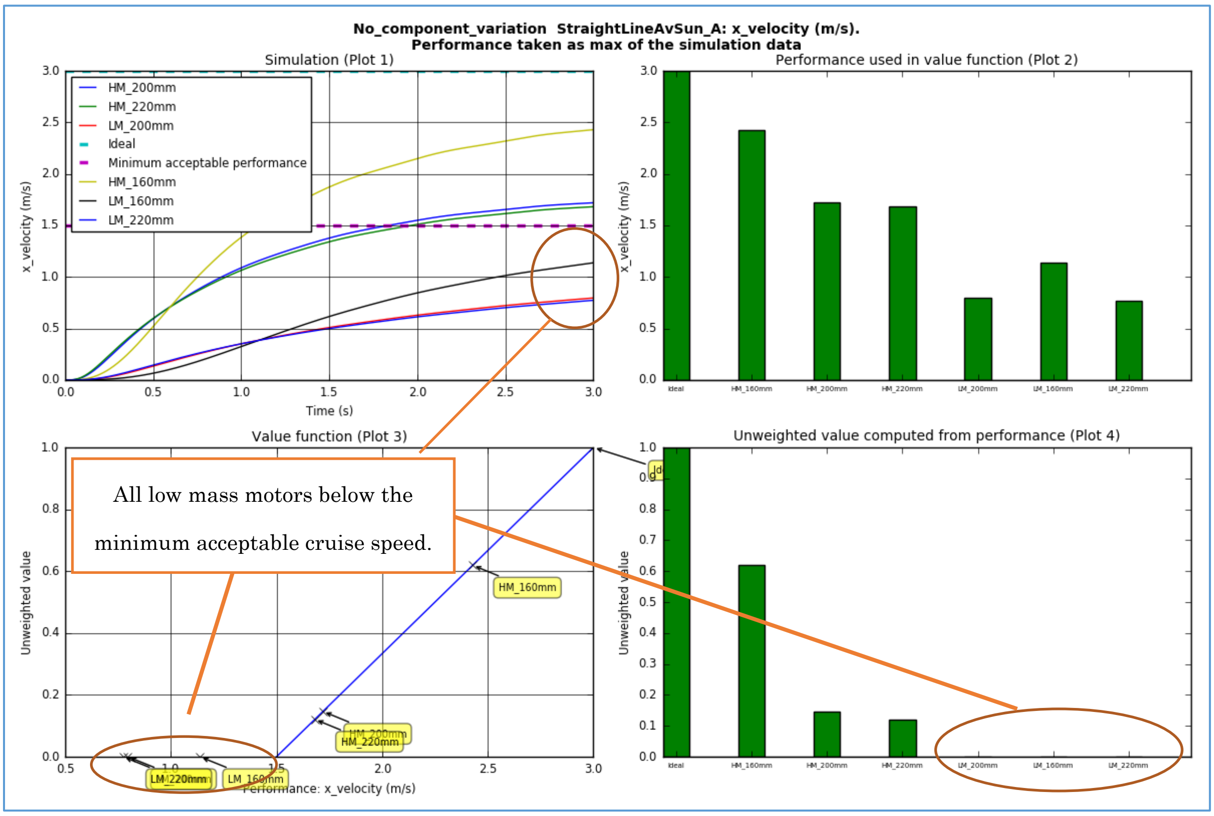


Figure 72 Results for the straight line average sun scenario

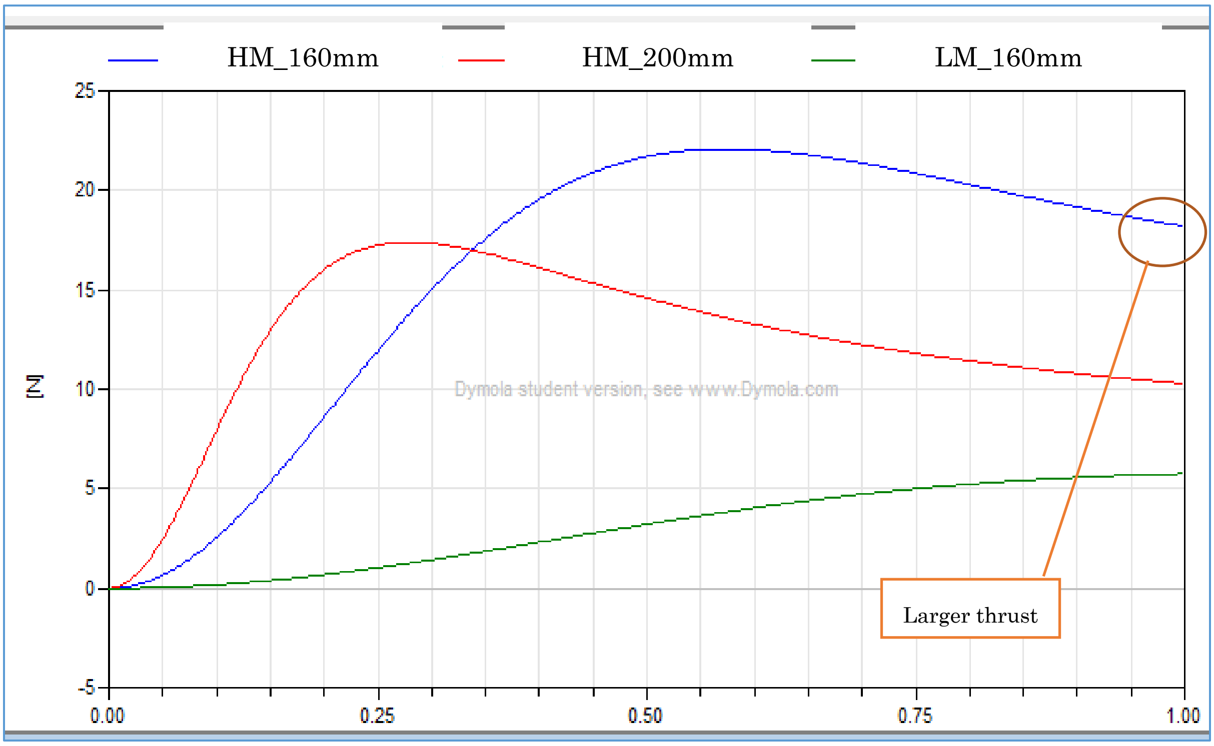


Figure 73 Time series (seconds), comparing thrust results for three different alternatives

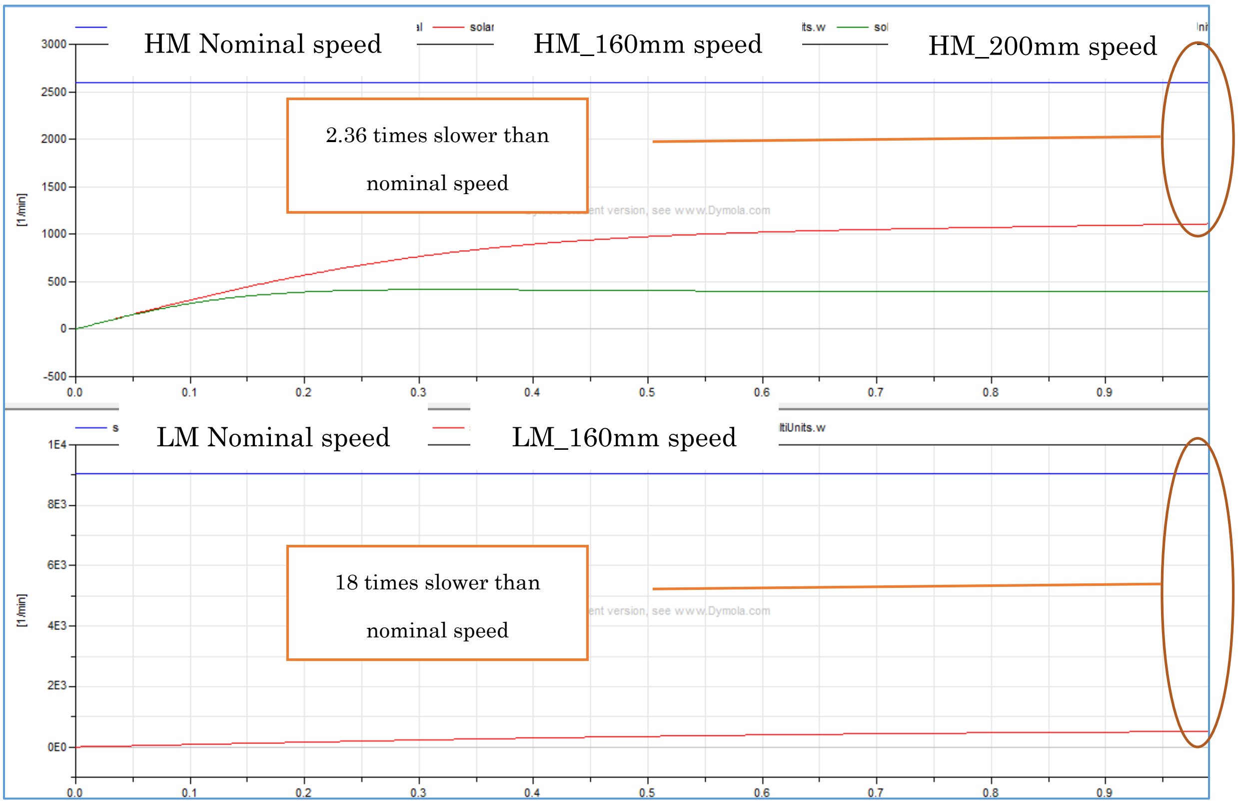


Figure 74 Time series (seconds), comparing motor spin speed for two heavy motor (top) and one low mass motor (bottom) alternatives to their nominal speeds.

Step J results in the decision to modify the designs and repeat the assessment. Given ideally a Solar-Boat alternative could be developed with the low mass motor (good floating performance) which operates closer to its nominal speed.

5.2.2 Design Set 2 (Subsystem Functional Architecture variation: add a motor speed changing device)

Based on the results of the previous section an attempt is made to develop a new Solar-Boat design with higher performance. Which is described as follows:

Steps A, B, C and D handle Subsystem identification and Assessment Scenario specification. These need no modification.

Step E requires modification to vary the alternative designs considered.

Step E1 involves a review of the Functional Architecture, given the target is to change the motor spin speed to produce more thrust. As such Figure 45 Functional Architecture of "Electrical to thrust" Subsystem is reviewed and new processes "Changing rotation speed" process inserted between the existing processes of "Converting electrical energy to rotation" and "Converting mechanical rotation to thrust".

Step E2 results in a new System Architecture created from the new Functional Architecture. Given a new process has been introduced a new object to enable it must also be introduced

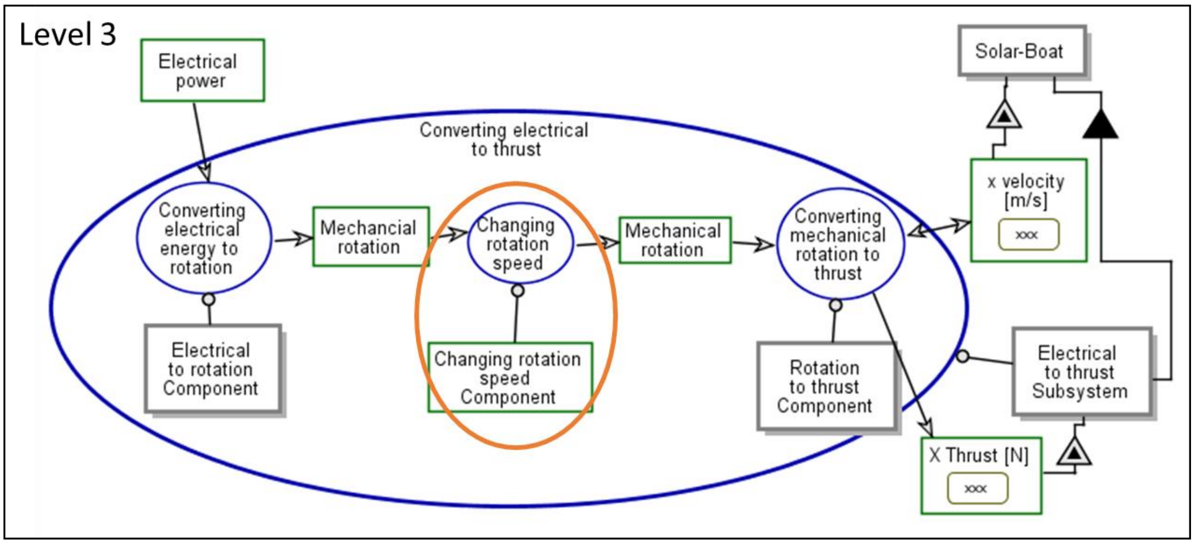


Figure 75 Level 3: Alternative System Architecture "Converting electrical to thrust"

Steps E3 and E4 involves the Formal Structure in OPM and Modelica being updated given the System Architecture has been updated. Which by following the processes described previously and depicted in Figure 49 and Figure 52 Formal Structure can be created with a position for the "Change rotation speed Component" introduced. It should be noted that the interface for the "Electrical to thrust" subsystem will not vary and as such Level 2 and Level 1 require no architecture changes. Subsequently the updated Formal Structure in OPM can be used to update the Formal Structure in Modelica which is shown on the left side of Figure 76. Resulting in a new "Electrical to thrust" subsystem Formal Structure.

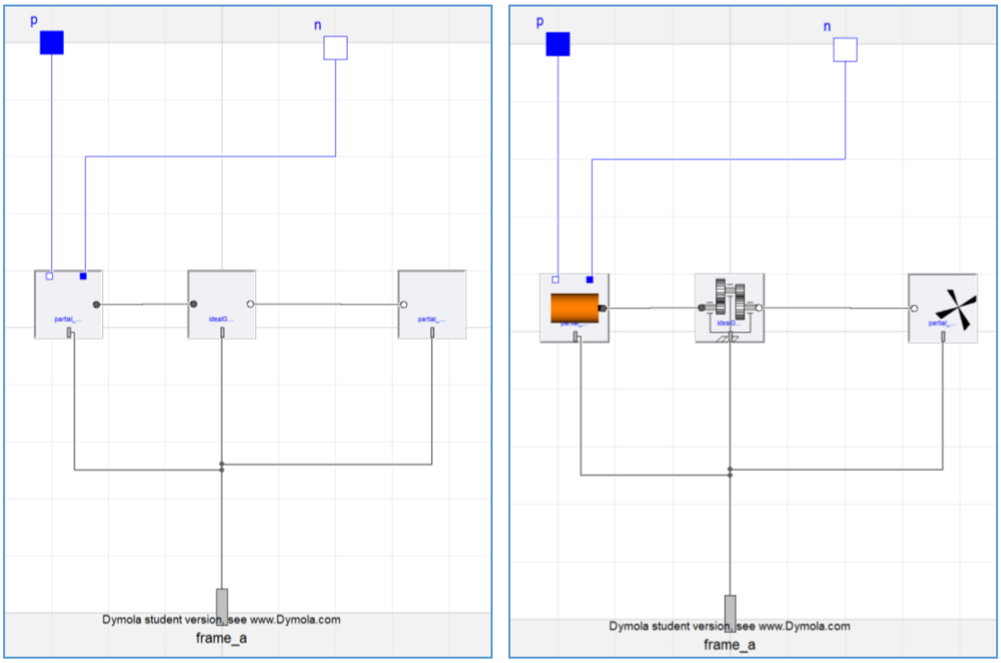


Figure 76 Level 3: "Electrical to thrust". Left: formal structure in Modelica. Right: alternative composed of subsystem-components

Step E5 (composition of alternatives) can now be performed utilizing the new "Change rotation speed Component" in the new Formal Structure. As in, new alternative "Electrical to thrust" Subsystems can be created by the composition of Subsystem-Components. Keeping the existing two motors and three propellers a 3:1 gearbox is introduced for the heavy motor and a 13:1 gearbox for the low mass motor. Resulting in six alternative "Electrical to thrust" Subsystems (one example utilizing a particular motor, gearbox and propeller is displayed on the right side of Figure 76). Which when composed into the Level 2 Formal Structure while keeping the existing hull and solar panel array results in six alternative Solar-Boats which are named corresponding to the motor, gearbox and propeller they utilize.

Steps F and G involves composing the six new alternative designs into the four specific Assessment Scenarios Formal Structure (Step F) resulting in twenty four separate models which are then simulated (Step G).

Steps H and I are completed resulting in Figure 77 where two alternatives incorporating the low mass motors (LM_13_200mm and LM_13_220mm) now deliver greater than that of any high mass motor configurations as a result of better performance in the "Driving forward" and "Floating" Assessment Scenarios.

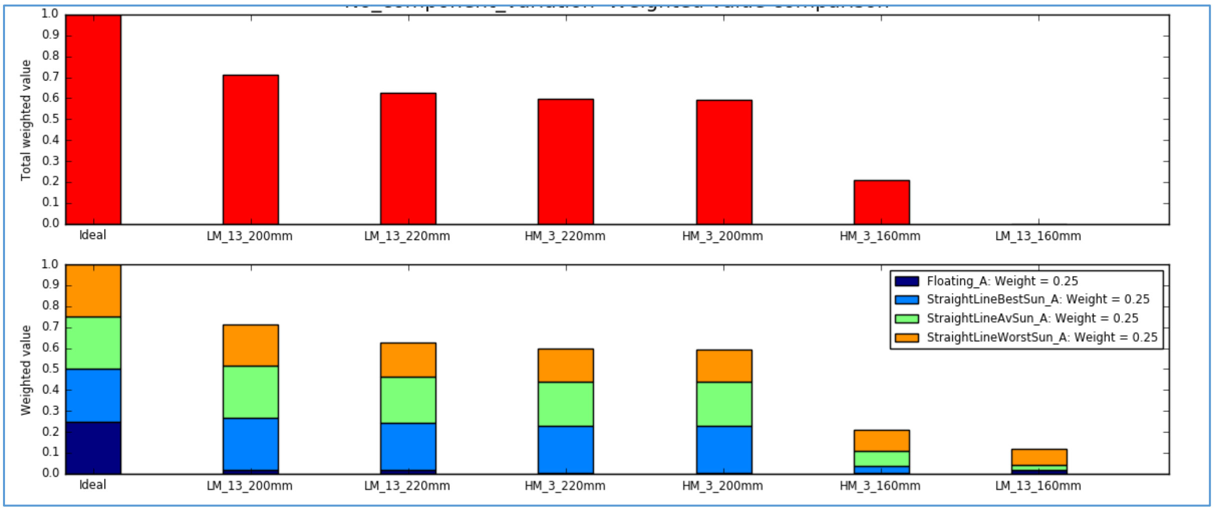


Figure 77 MODA processed results from simulation varying motors, gearboxes and propellers

Step J given the better performance it now be appropriate to move into detailed design of a prototype based on this System-Level Design.

5.2.3 Design Set 3 (Subsystem-Component variation with cost impact: compare new expensive solar panels which are more efficient but more heavy)

Further component exploration of interest could involve the performance evaluation associated with incorporating higher efficiency, high mass and high cost solar panels (solar to electrical Subsystem). Creating alternatives based on these and simulating results in Figure 78. The bottom chart clearly displays the large cost of the new solar panels (exceeding the project budget). While the weighted total value of alternatives incorporating the panels is not significantly different to those utilizing existing panels. Indicating they are not a wise purchase.

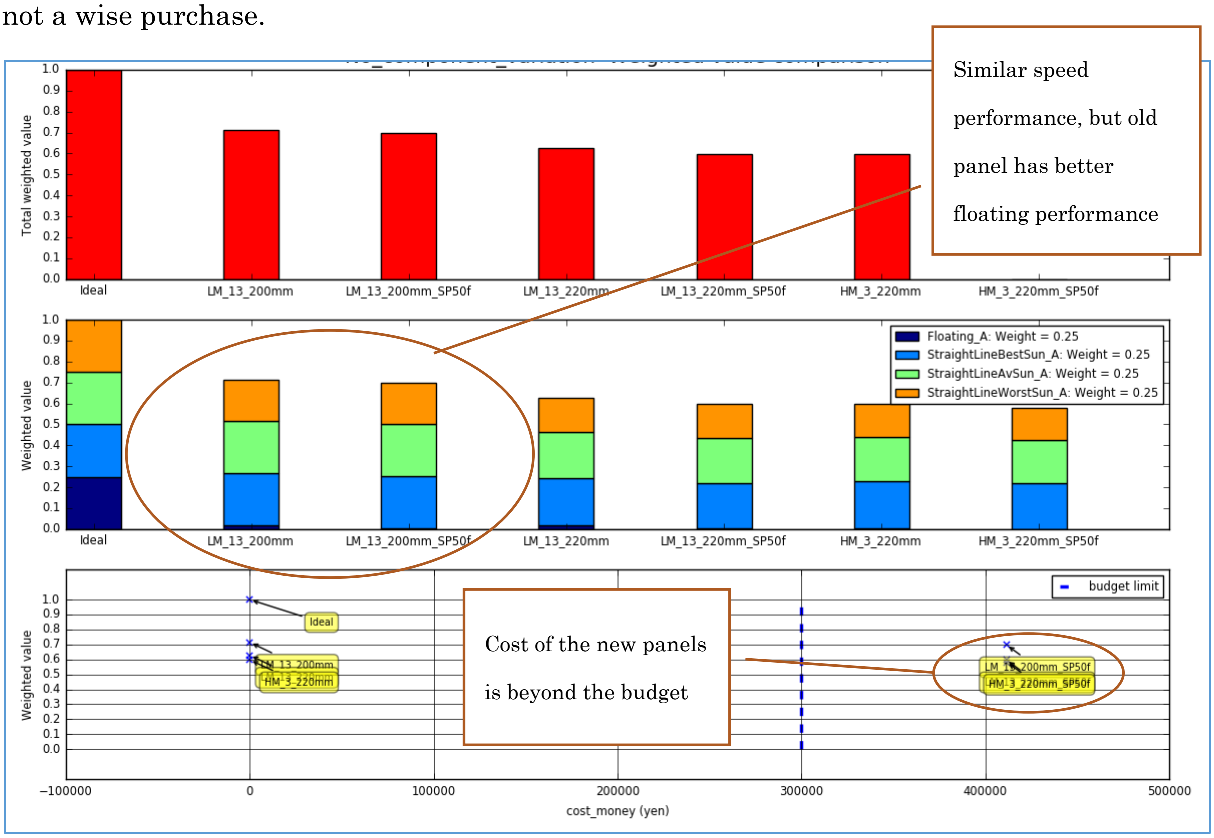


Figure 78 MODA processed results for Solar-Boat alternatives incorporating higher efficiency but higher mass and more expensive solar panels (known as SP50f)

5.3 Experience a rule change. Payload varied from 0.064kg to 15kg:

Another scenario is to imagine a rule change which requires the payload to be increased to 15kg. Initially the existing designs are compared with the new heavy payload. Changes are then made to improve performance.

5.3.1 Design Set 4 (Rule change: see impact of larger payload)

Subjecting the designs of Section 0 to the same Assessment Scenarios but now with a larger payload and processing the results enables Figure 79 to be created where clearly all the Alternative designs are failing most of the Assessment Scenarios. Reviewing the detailed results (Figure 80) indicates that the alternatives incorporating the high mass motor (HM) or high mass solar panel (SP50f) are sinking. While the others have a z position which is below the minimum acceptable (but not completely submerged).

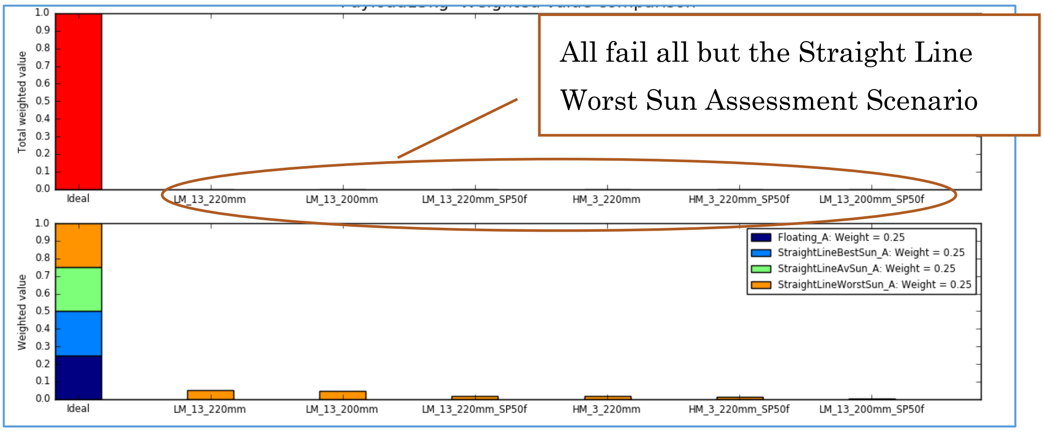


Figure 79 MODA processed results of Solar-Boat alternative designs of Section 0

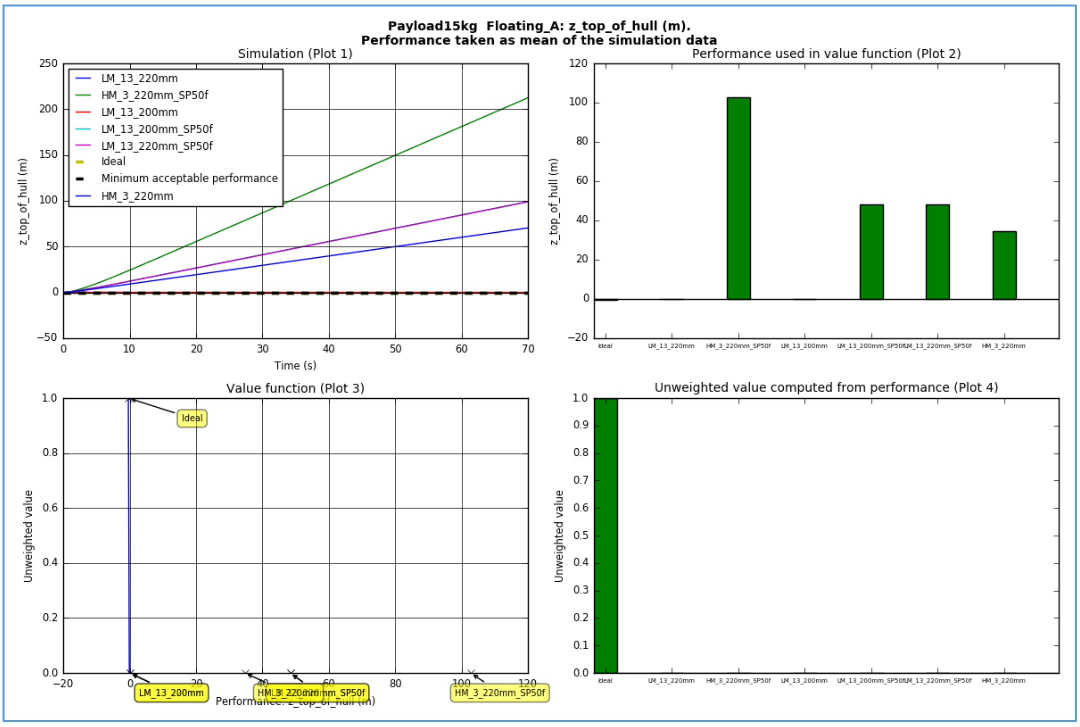


Figure 80 Results of Assessment Scenario Floating for alternative designs of Section 0

5.3.2 Design Set 5 (Subsystem Formal Structure variation: Address the payload issue with different buoyancy system & Change assessment approach)

Initially in an attempt to obtain passing performance is made by taking the highest performance Solar-Boat Alternatives from Section 0 but vary the Buoyancy Generation system. Introducing a double width (DW) variant of the existing single hull design and a dual hull (DH) variant where two of the existing single hulls are placed parallel to each other. This is described in Table 30.

Alternative	Buoyancy Generation	Solar To Elec	Elec To Thrust	Overhead components
HM_3_220mm_ SP50f_DW	Single hull (Double Width)	SP50f	H motor: 3:1 gearbox: 220mm prop	0.5 + 2.3 kg
HM_3_220mm_ SP50f_DH	Dual hull (Standard Width)	SP50f	H motor: 3:1 gearbox: 220mm prop	0.5 + 2.3 kg
LM_13_ 220mm_DW	Single hull (Double Width)	FT-136SE	L motor: 13:1 gearbox: 220mm prop	0.5 + 2.3 kg
LM_13_ 220mm_DH	Dual hull (Standard Width)	FT-136SE	L motor: 13:1 gearbox: 220mm prop	0.5 + 2.3 kg

Table 30 Alternative Solar-Boat designs created by varying the Buoyancy Generation subsystem

Subjecting the new alternative designs to the Assessment Scenarios which have been used previously (Table 28) and processing the results generates Figure 81 where clearly all the Alternative designs while now passing the straight line driving scenarios remain failing the floating scenario. As such the results of the floating scenario are reviewed (Figure 82). Here is it found that the performance of the alternative designs is similar and not far from minimum acceptable performance. Given manufacturing is easier when keeping close to an existing design it is considered to modify the Minimum acceptable performance of the floating assessment scenario from (-0.1m to -0.8m) as shown in Table 31.

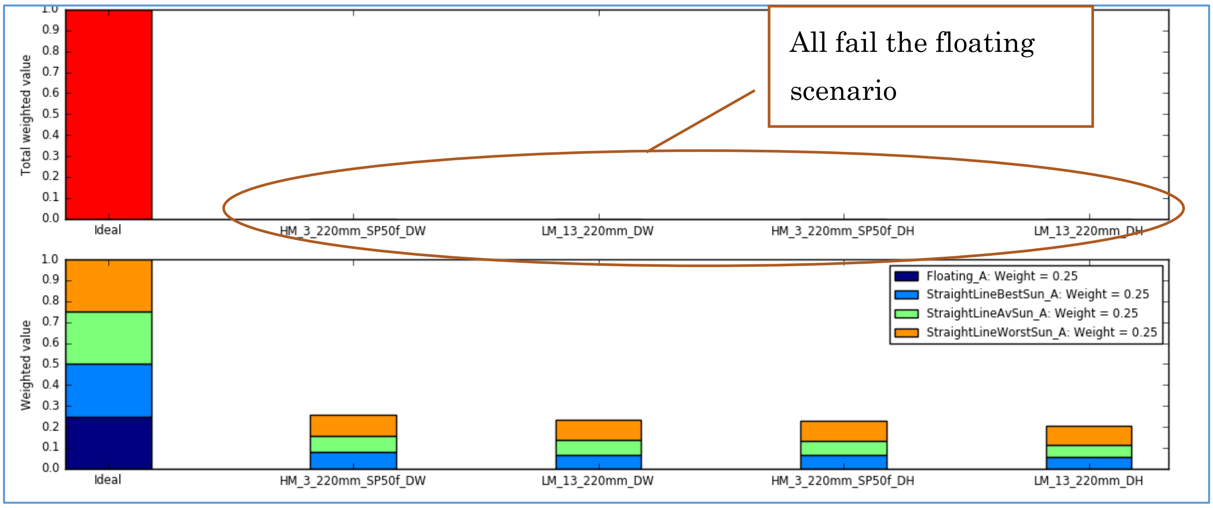


Figure 81 MODA processed results from simulation varying buoyancy generation

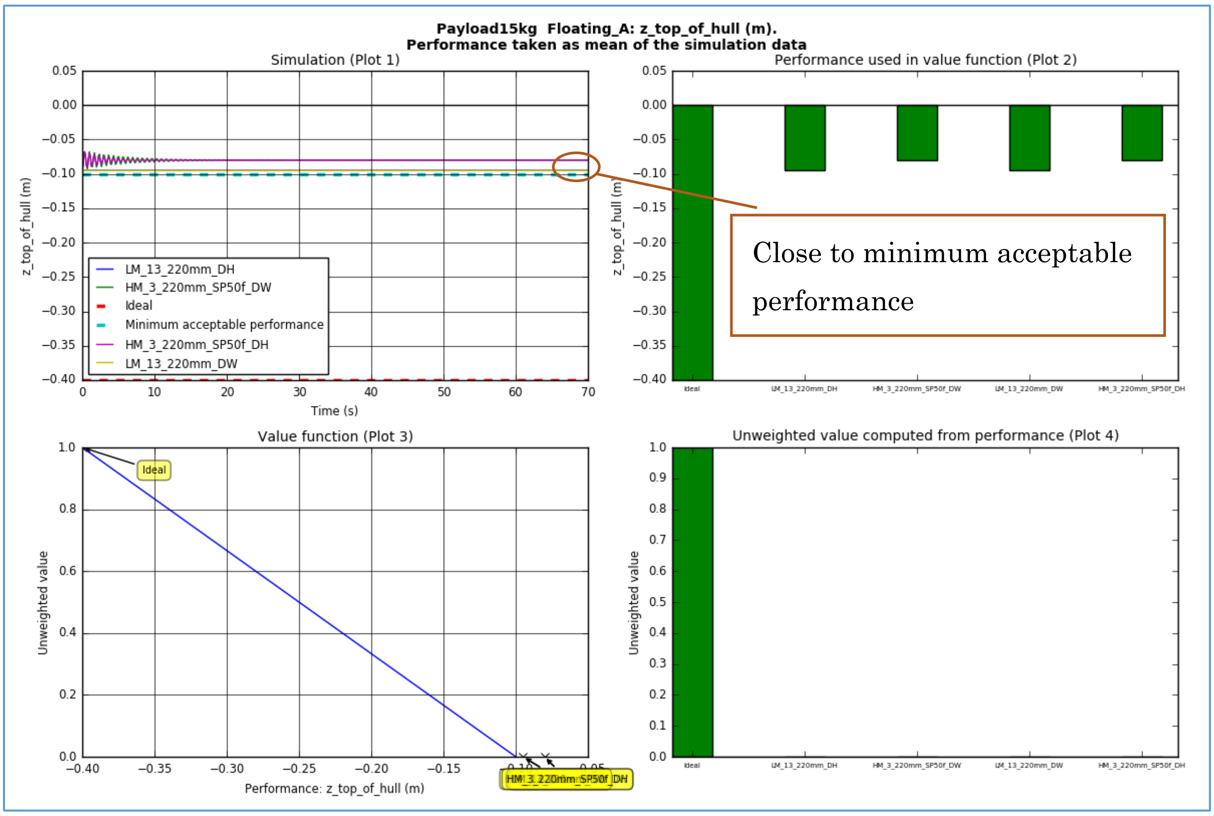


Figure 82 Results of Assessment Scenario Floating for alternative designs varying buoyancy generation

Assessment Scenario name	Measure of interest	Minimum acceptable performance	Stretch goal	Sim time (s)	Data extraction type
Floating	z position (m)	-0.08	-0.4	70	Mean

Table 31 Updated Floating Specific Assessment Scenario from Table 28

Running the new updated floating assessment scenario of Table 31 with existing ones of Table 28 results in Figure 83 where all the alternative designs are shown to pass all the assessment scenarios. The performance difference between all the designs is small, with the high mass motor (HM) and high mass panel (SP50f) achieving higher performance through speed not floating ability.

Given the high cost of the high mass panel (SP50f) explored in Section 0 for little performance gain it is appropriate to discard these designs. The little performance difference between the low mass motor designs (LM_13_220mm_DW and LM_13_220mm_DH) means that the decision as to which of select can be made on other factors, such as the ease of manufacturing two small hulls when compared to one large hull.

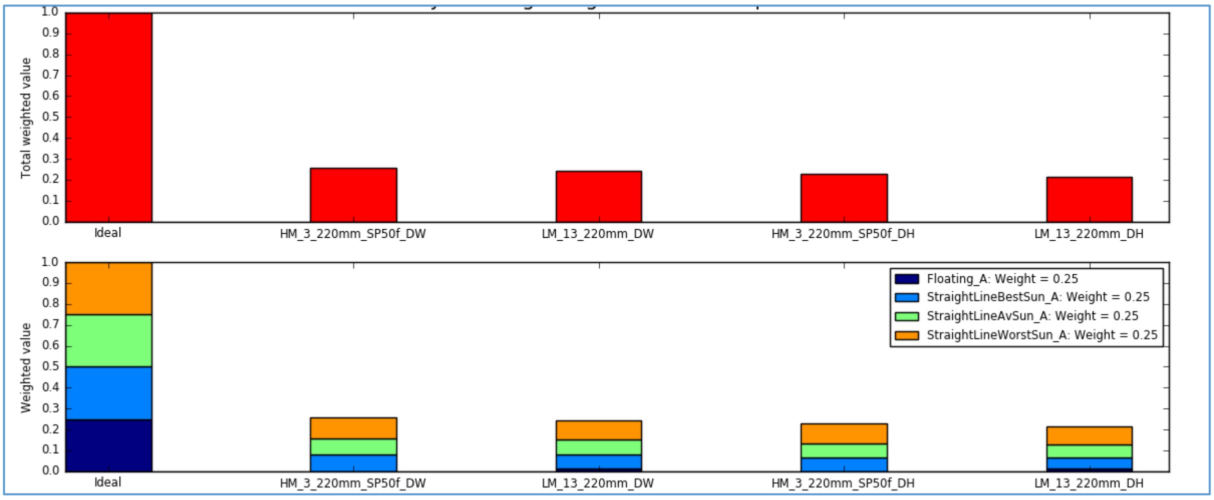


Figure 83 MODA processed results from simulation varying buoyancy generation after changing the Floating Specific Assessment Scenario

6 Discussion

6.1 Chapter purpose

This chapter aims to provide the reader with a discussion around:

- The key ideas this thesis has attempted to describe and demonstrate
- How the approach used is novel compared to current literature
- The benefits of the proposed system but also its shortcomings
- Proposed future research directions, based on logical next steps and the impact of future technologies

6.2 Providing knowledge in models

In Section 2.2 the following problems were identified with providing knowledge in models: "Selecting appropriate modeling languages", "Integration of multiple modeling languages together so different types of knowledge can be adequately captured" and "Keeping said models updated".

As part of this thesis the first two of these has been addressed in that OPM and Modelica have been found to be appropriate when used with the defined model types and mapping. However keeping the models updated has not been explicitly addressed.

By using a systems modeling language (OPM), attempts have been made to capture tacit knowledge previously stored in non-ideal locations (brains, documents) in a more accessible location. The simulation models held in an appropriate framework provide much an easier way for computational model reuse. Further the usage of these tools facilitates exploration of the knowledge space around the system of interest.

6.3 Complete trade-off analysis of multiple designs using models to simulate performance

In Section 2.3 the following problems were identified with completing tradeoff analysis using multiple models: "Providing a framework which can assess all alternative designs", "Being able to compare a reasonable number of design alternatives" and "Balancing the use of numerical optimization with exploratory approaches".

As part of this thesis all three have been addressed by way of demonstrating:

- Conducting System-Level design trade studies following the INCOSE Decision Management Process as elaborated by (Cilli & Parnell, 2014). However due to the lack of detailed requirements documents or an assumed System Architecture, a process is defined to create such a Systems Architecture which can ultimately be mapped to a Formal Structure for implementation in a numerical simulation tool. As such the following are defined:
  - The definition and usage of a series of model types implemented which map between each other (making use of OPM and Modelica)
  - The definition of hierarchy between each model type

  - A suitable hierarchy for a student Solar-Boat project

- Due to the need to assess multiple Systems of Interest – Alternatives subject to the same Assessment Scenarios a software program implemented in python was created to enable the fast comparison of these alternatives.
- Two case studies are presented attempting to describe design exploration around a powertrain and a whole Solar-Boat.

6.4 Points of Novelty

- Describing a method to identify a System Architecture and Formal Structure to which common Assessment Scenarios can be applied, and attempting to eliminate a point of assumed knowledge from implementations of the INCOSE Decision Management Process as elaborated by (Cilli & Parnell, 2014) and (Edwards et al., 2015).
- Specifically the linking of a conceptual modeling methodology, OPM, to a numerical simulation tool (in this case Modelica) for the purposes of design.

6.5 Shortcomings

- Mapping from System Architecture to the Formal Structure requires a library of components arranged under the descriptions System Architecture. Existing libraries are not described in such a way. Arrangement of such libraries needs exploration.
- Currently behavior being modelled in Modelica created under the current scheme assumes the processes all occur simultaneously at all times. However real systems demonstrate processes which are not occurring at all times (e.g. time triggering of a process or if/then rules), further OPM supports such system descriptions. As such the methodology should be expanded to incorporate this.
- The current approach of mapping Functional Architecture to Formal Structure is made under the assumption of very simple modularity (one process is enabled by one object), with no provision for situations where two subsystems enable a process or vice-versa without merging such subsystems and processes to simple process and object pairs.
- The target for this project has been a vehicle assumed to be a point mass subject to forces. No attempt has been made to generalize to other domains or more complex systems.
- It is assumed that the system is decomposed from the top down from a functional perspective. While this is logical it does have some drawbacks in that very rapidly the designer can be considering the operations of the system (and its interaction with the environment) which might not be the designer's intent.
- By encouraging the use of components to build up complete subsystems immediately (using Modelica acausal links) there is potential for the designer to skip what could be useful "back of the envelope" calculations which do not take into account acausal

connections. A concrete example being for the design of a powertrain where it can be useful to reason about assumed efficiencies and power flow between components and not the medium by which that power flows of components in a powertrain as power flows.

6.6 The future work

6.6.1 Logic

- Demonstration on a more complex system is required to see how well the logic scales
- Timing and control logic for behavior: in the current demonstration the behavior which is being modelled in Modelica is assumed to occur simultaneously. Real systems exhibit causality when a particular behavior is triggered (e.g. time based if->then).

6.6.2 Modelica libraries

- High quality, well-described and readily available libraries of components can drastically increase the speed at which a model can be created. This is perhaps especially true with the functional-description-first approach described in this thesis, where potentially the requirement to model a component not typically familiar with the designer might be identified. This contrasts with approaches where a numerical model is created from the start without analysis of the functions the system is to perform, as such the designer is likely limited by the numerical models they already have or can imagine.

6.6.3 Current implemented system

- Automation of the generation of Modelica models of Subsystems, and Systems of interest out of a pool of library components would enable a larger number of alternatives to be explored similar to that demonstrated by (Edwards et al., 2015).
- Parameter variation of components has not been applied. I.e. a single hull design was used of fixed size. Varying parameters will result in different predicted performance.

7 Conclusions

In this thesis it has been attempted to show how Knowledge Management and System-Level Design can be accomplished by creating a design methodology incorporating:

(a) A defined framework for the mapping of conceptual modeling in OPM to numerical modeling in Modelica
(b) Utilization of the INCOSE Decision Management Process to formalize decision making
(c) Ensuring previous design knowledge embedded in the models which was previously either implicit or document-based
(d) A practical demonstration of model based design and Model Based Systems Engineering for educational purposes

The defined framework to enables the logical creation of conceptual models in OPM which are logically transferred to numerical models in Modelica by way of six model types were described (and their hierarchical decomposition for complexity management):

1. Functional Architecture (OPM)
2. Systems Architecture (OPM)
3. Formal structure (OPM)
4. Formal structure (Modelica)
5. Alternative (Modelica)
6. Simulation result (Modelica))

These models (and their hierarchical decomposition) are then incorporated into the INCOSE Decision Management Process, where the decomposition described here is used to create a structure on which alternative designs can be assessed in the same way.

The careful selection of modeling languages, a dedicated conceptual language and methodology (OPM) and multi domain numerical modeling language (OPM) enables the storing of early Lifecycle stage knowledge.

This was practically demonstrated with a Solar-Boat project, where by the system of interest was functionally decomposed and mapped to alternative numerical simulation models which were assessed by way of Multi Objective Decision Analysis (MODA) for a range of scenarios. By defining an appropriate architecture and utilizing a common assessment framework it means significant design exploration can be enabled by simply switching components and subsystems. Which can be done by a human designer or automation software (as was created as part of this thesis).

The motivation for this came from an analysis of past Solar-Boat projects which is documented in Section 1.2.2 where in Table 6 multiple problems with various Lifecycle Stages were identified. By providing this thesis (and accompanying model libraries) future Solar-Boat project students can make use of significant captured design knowledge and expedite the rate of knowledge acquisition at early stages of the project and as such increase the speed at which a quality design can be created and deployed.

In Table 6 two tentative solutions were proposed to solve early stage problems, these being "Provide knowledge in models" and "Complete trade-off analysis of multiple designs using models to simulate performance". This thesis demonstrates how this can be done.

Appendices

Appendix: V-Models Describing 2014 Solar-Boat Development

The Basic V – Solar-Boat 2014

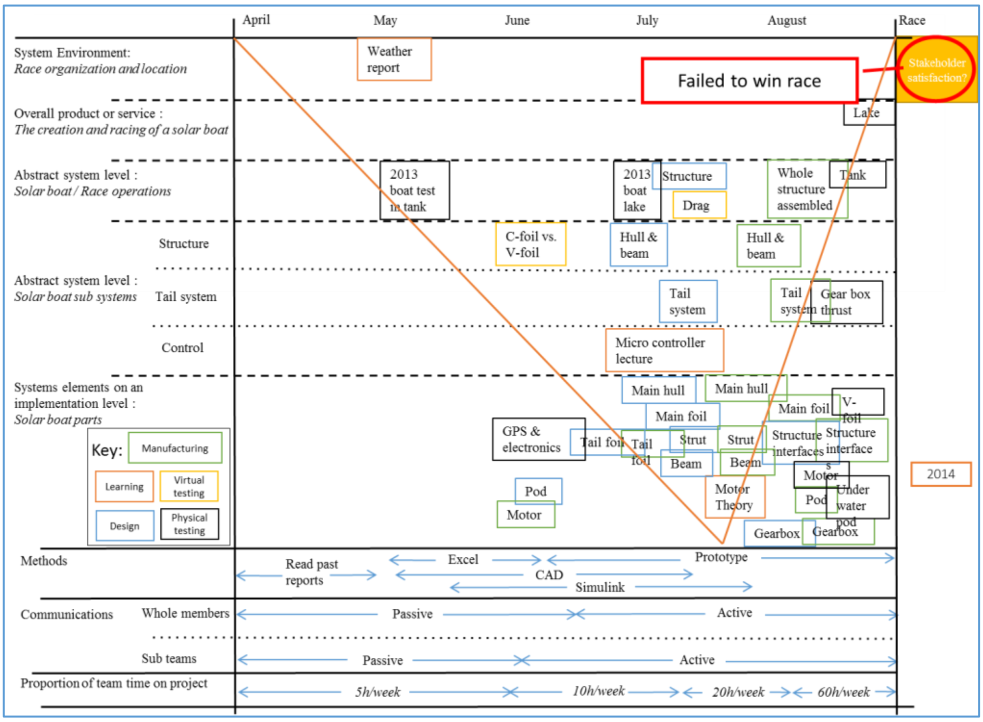


Figure 84 Basic V - Solar-Boat 2014. Indicating value to deliver stakeholder satisfaction. (Sutherland et al., 2015)

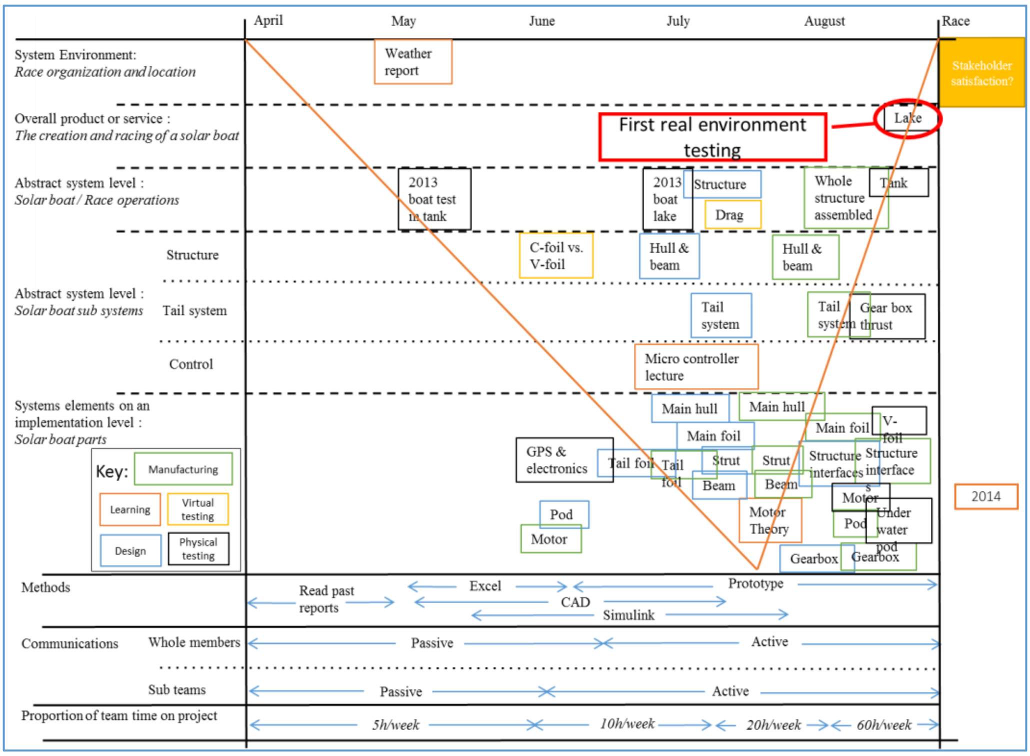


Figure 85 Basic V – Solar-Boat 2014. Indicating the lateness of first full integration testing of the system. (Sutherland et al., 2015)

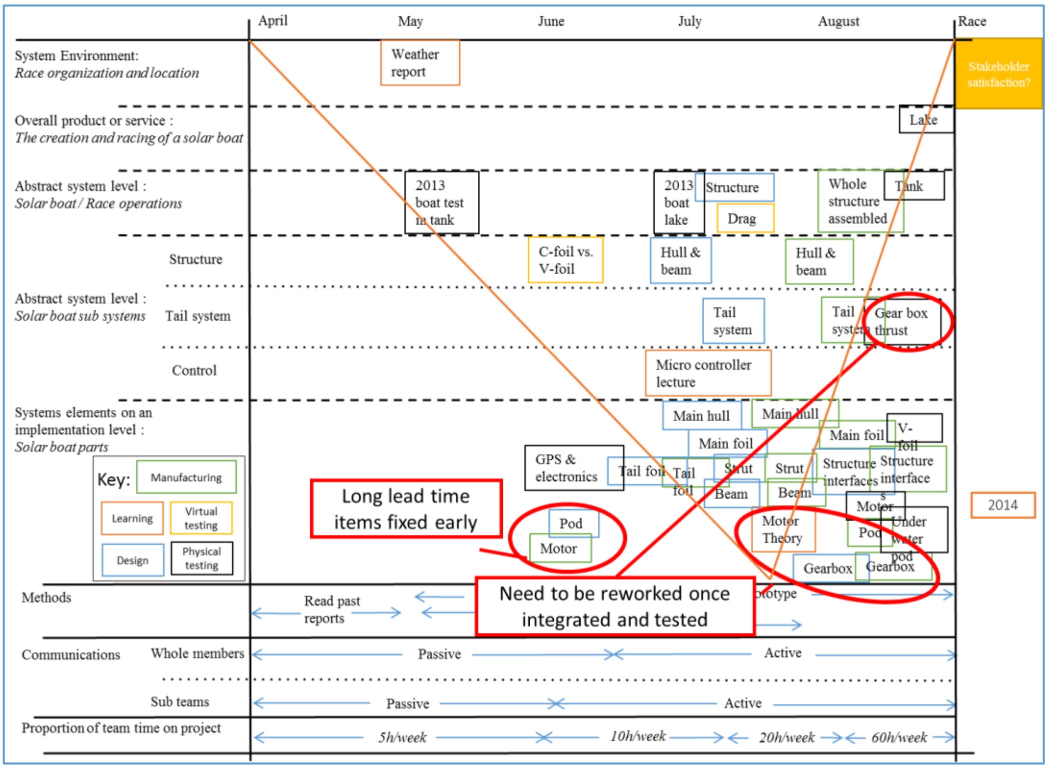


Figure 86 Basic V - Solar-Boat 2014. Indicating the need to rework major subsystems late in the project. (Sutherland et al., 2015)

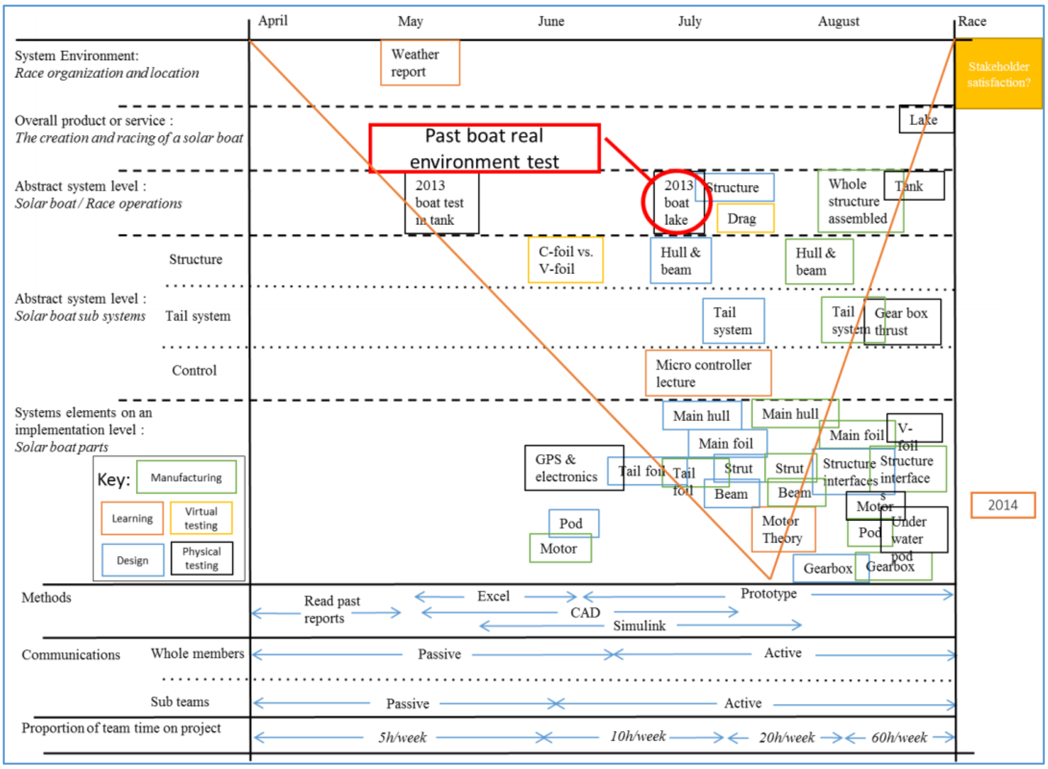


Figure 87 Basic V - Solar-Boat 2014. Indicating the first interaction with a past boat in its deployment environment occurred mid-way into the project (Sutherland et al., 2015)

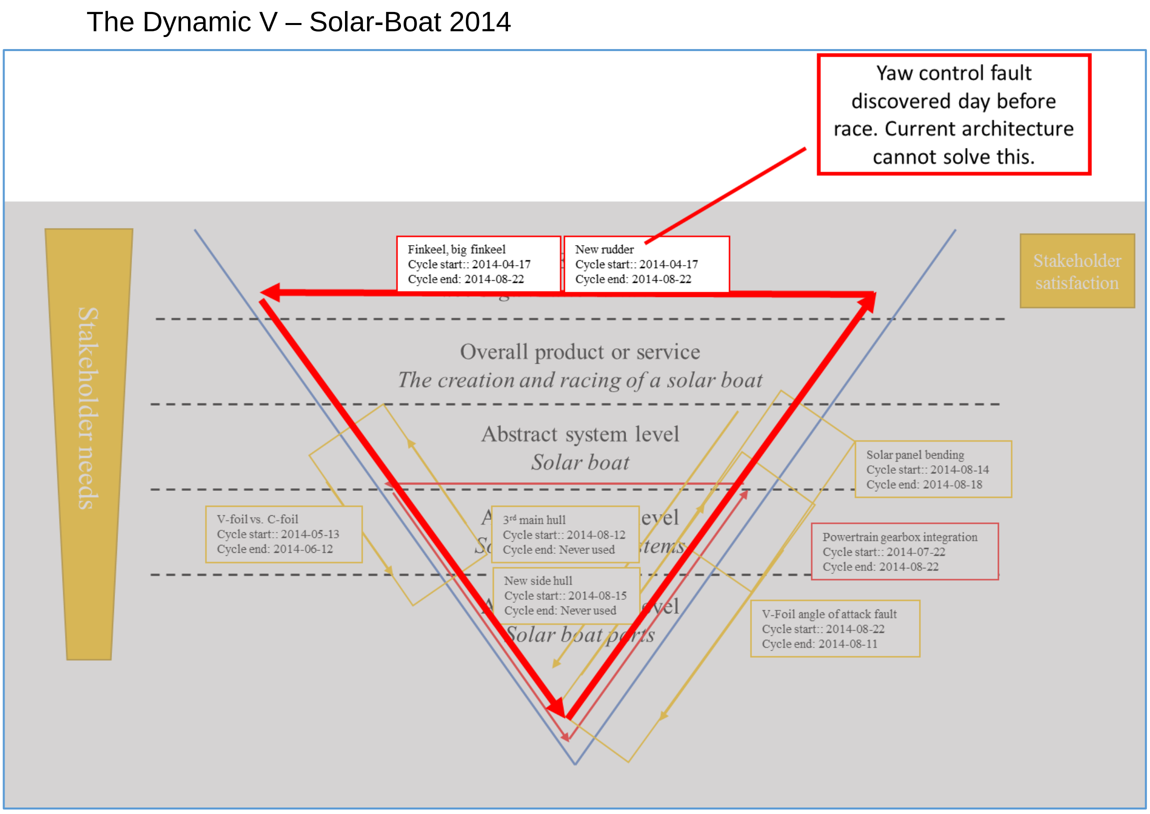


Figure 88 Dynamic V - Solar-Boat 2014. Indicating where major rework was required after the system was deployed in its environment. (Sutherland et al., 2015)

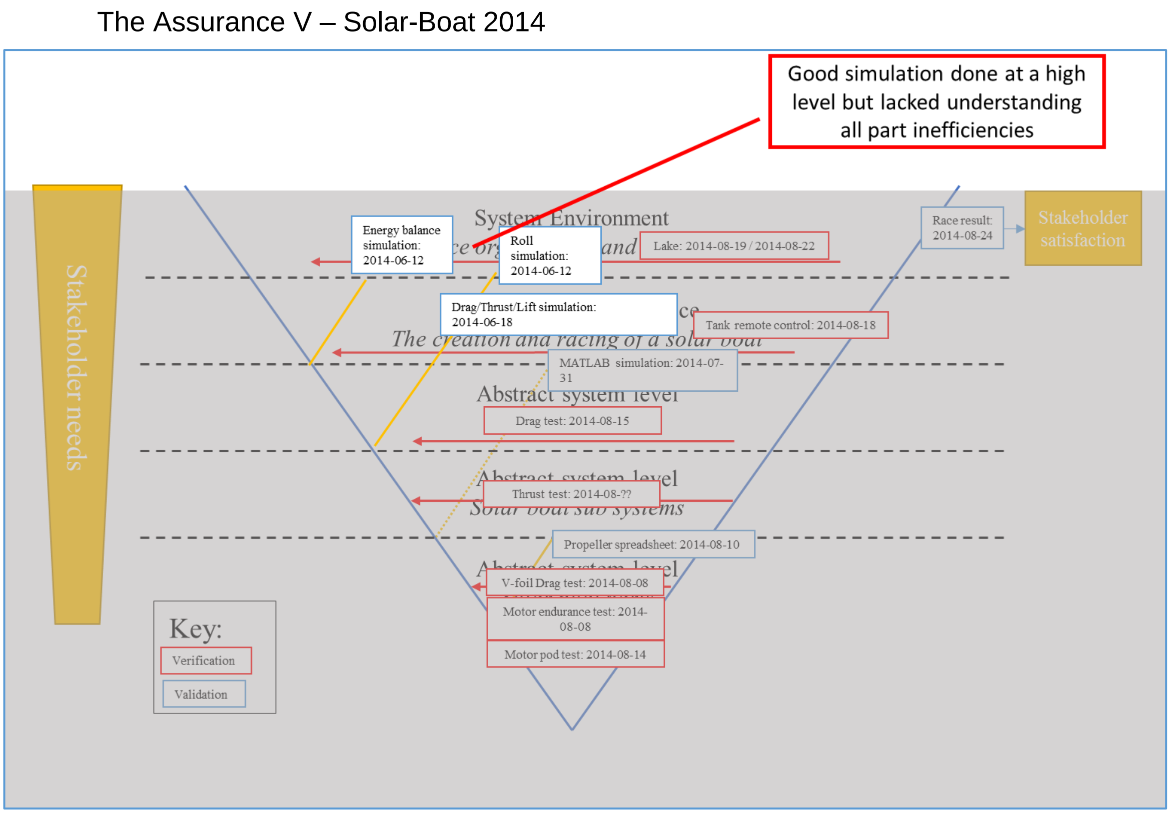


Figure 89 Assurance V - Solar-Boat 2014. Indicating high level simulation lacked detailed understanding to provide adequate part selection. (Sutherland et al., 2015)

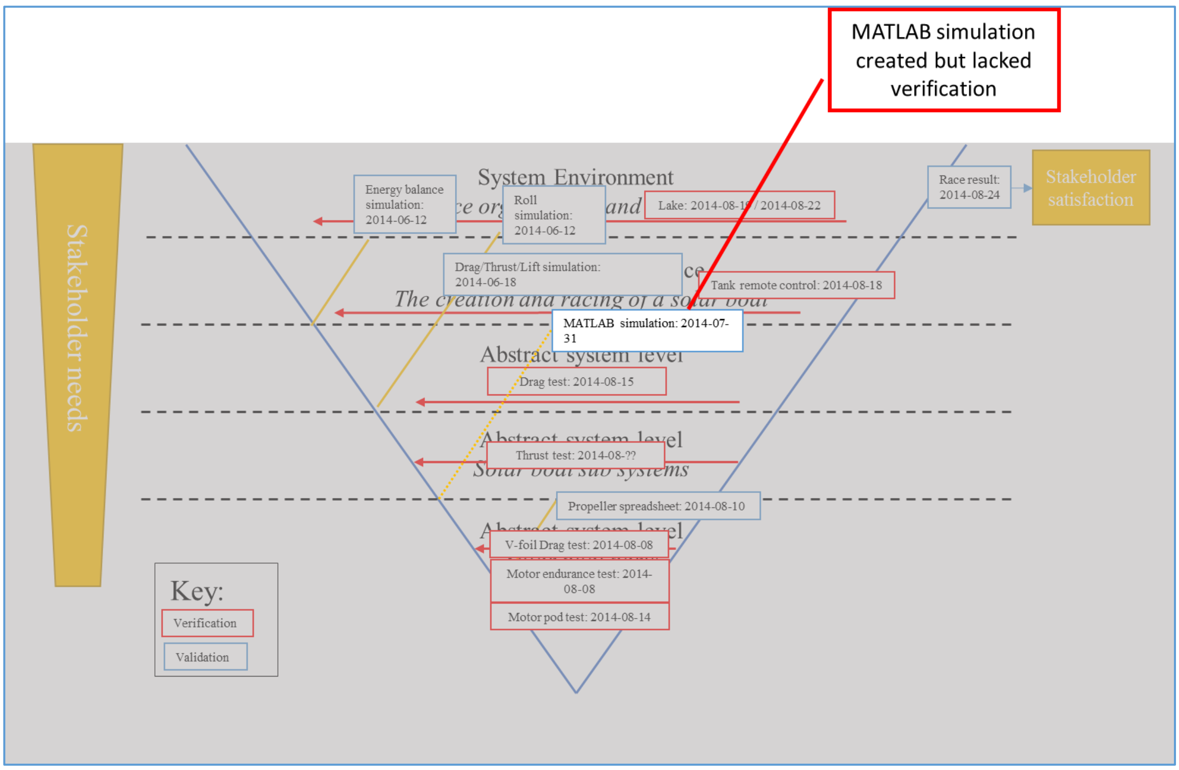


Figure 90 Assurance V - Solar-Boat 2014. Indicating lower level simulation lacked verification. (Sutherland et al., 2015)

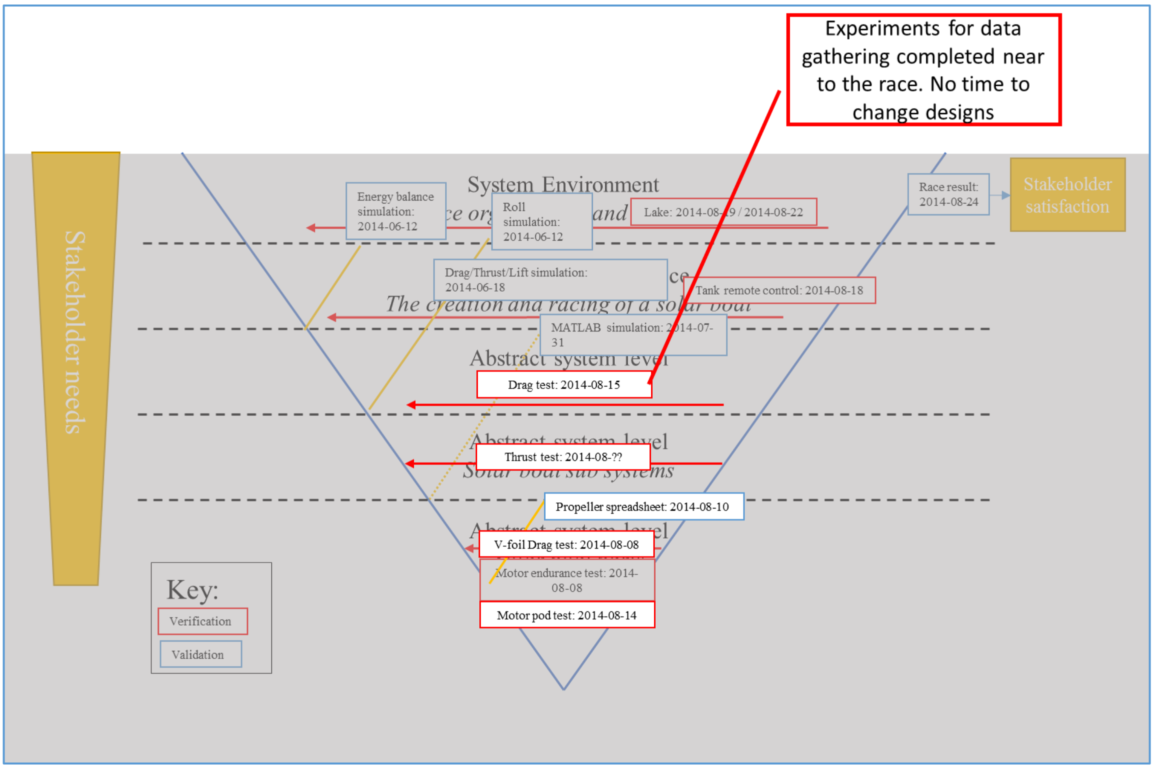


Figure 91 Assurance V - Solar-Boat 2014. Indicating experimental Verification occurred too late to make design changes. (Sutherland et al., 2015)

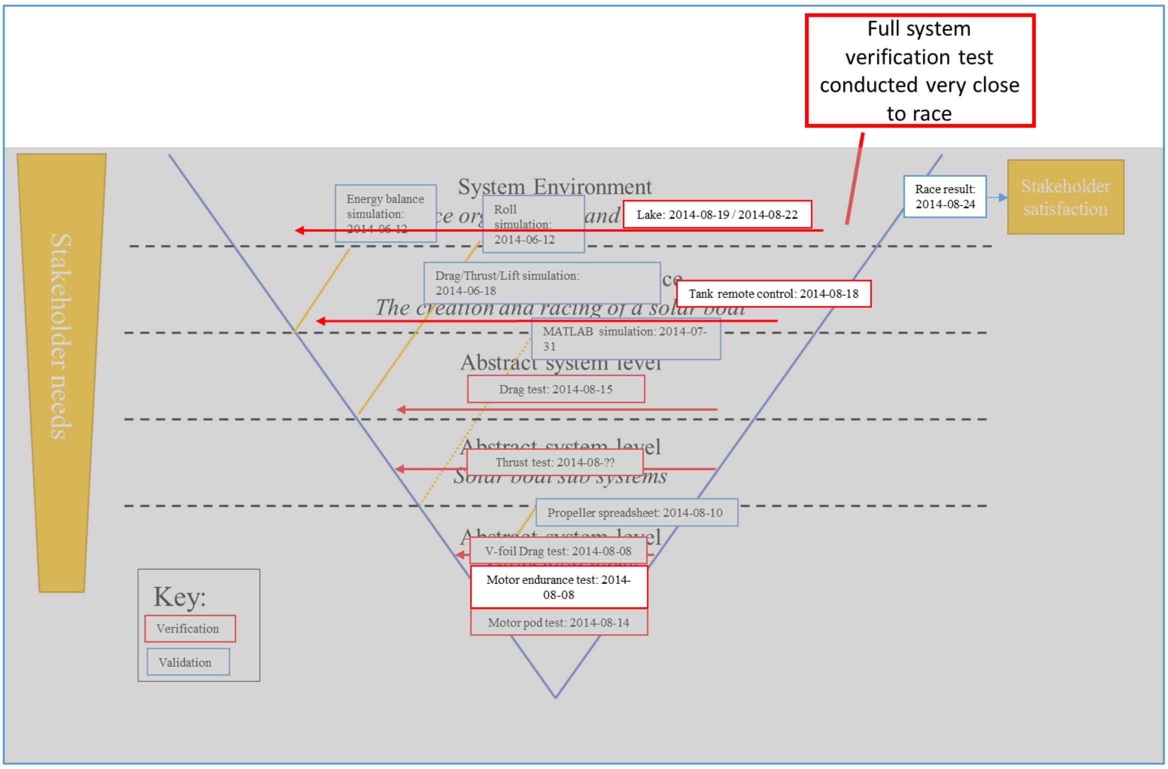


Figure 92 Assurance V - Solar-Boat 2014. Indicating full system verification was conducted very close to the race. (Sutherland et al., 2015)

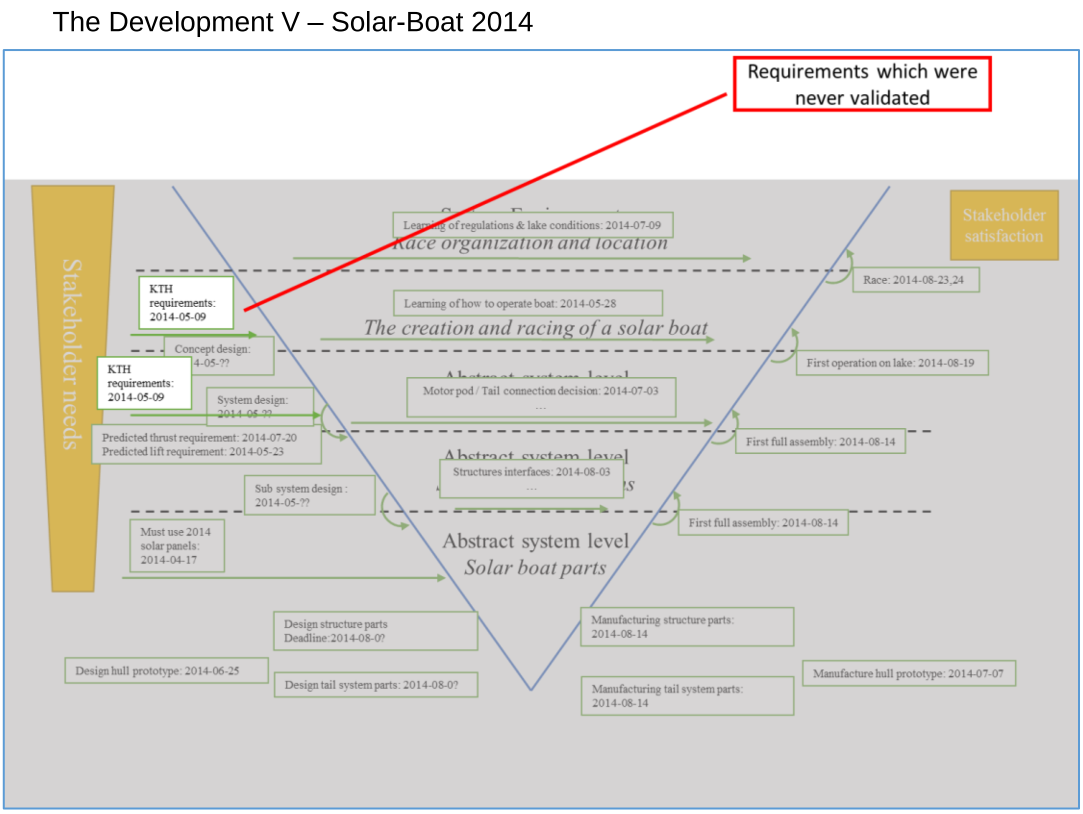


Figure 93 Development V - Solar-Boat 2014. Indicating requirements documents were never validated. (Sutherland et al., 2015)

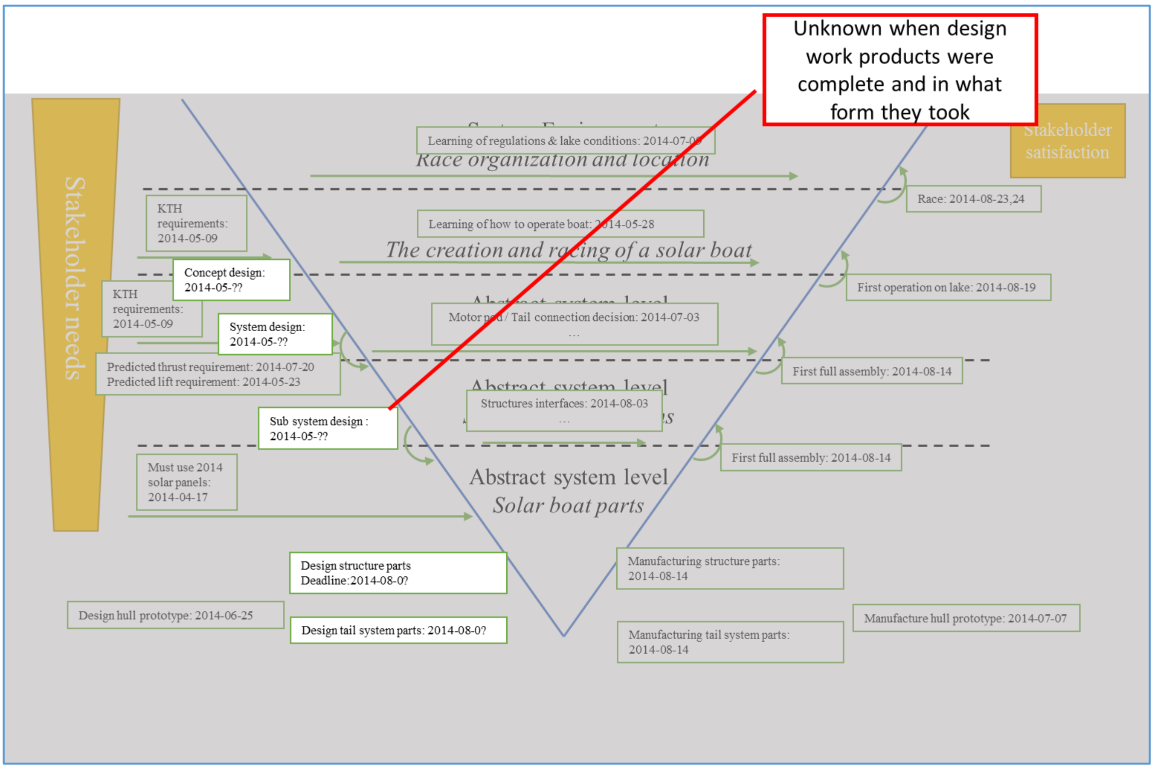


Figure 94 Development V - Solar-Boat 2014. Indicating being unclear when design work products were complete. (Sutherland et al., 2015)

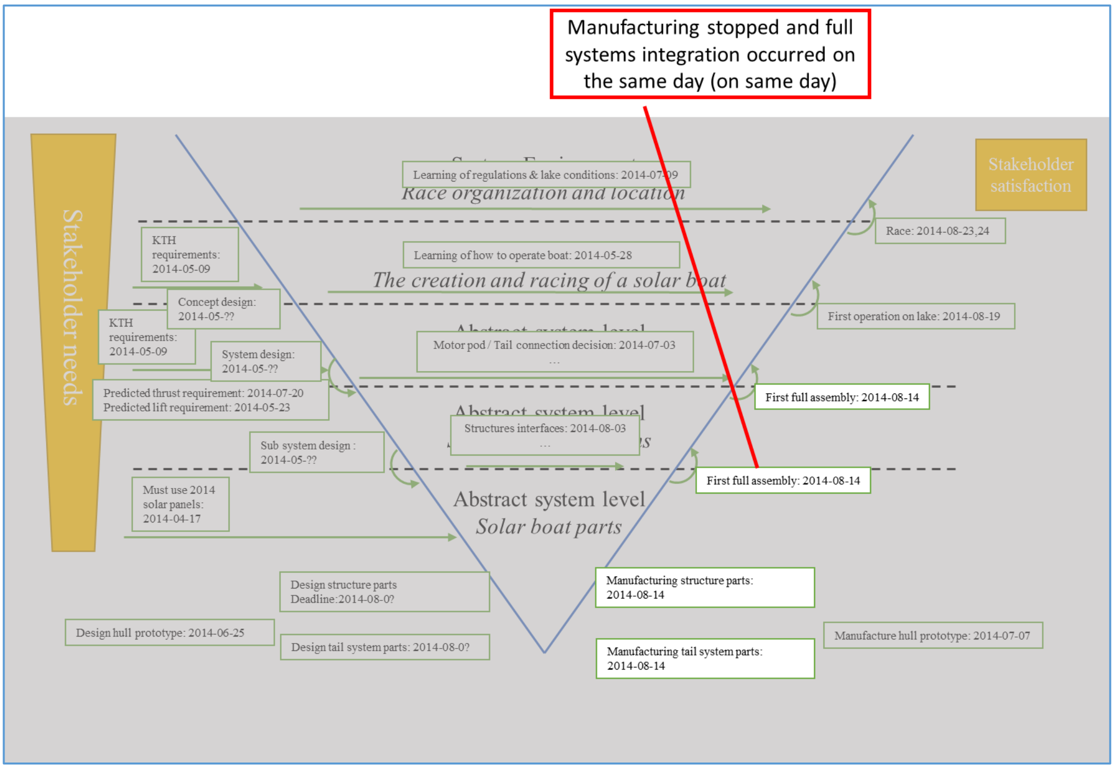


Figure 95 Development V - Solar-Boat 2014. Indicating manufacturing stopped and systems integration was completed on the same day. (Sutherland et al., 2015)

Bibliography

Batteh, J., & Tiller, M. (2009). Implementation of an extended vehicle model architecture in modelica for hybrid vehicle modeling: development and applications. In *Proceedings of the 7th International Modelica Conference* (pp. 823–832). Retrieved from http://www.ep.liu.se/ecp/043/097/ecp09430109.pdf

Bolshchikov, S., Renick, A., Mazor, S., Somekh, J., & Dori, D. (2011). OPM Model-Driven Animated Simulation with Computational Interface to Matlab. In *2011 20th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)* (pp. 193–198). http://doi.org/10.1109/WETICE.2011.32

Chapon, D., Bouchez, G., & France, A. (2009). On the link between architectural description models and modelica analyses models. In *Proceedings 7th Modelica Conference, Como, Italy* (pp. 784–789). Retrieved from http://www.modelica.org/events/modelica2009/Proceedings/memorystick/pages/papers/0079/0079.pdf

Cilli, M. V., & Parnell, G. S. (2014). Systems engineering tradeoff study process framework. In *24th INCOSE Int'l Symposium, Las Vegas, NV*. Retrieved from http://www.researchgate.net/profile/Gregory_Parnell/publication/268820833_4.3.1_Systems_Engineering_Tradeoff_Study_Process_Framework/links/54bd0f390cf218d4a16955ba.pdf

Crawley, E., Cameron, B., & Selva, D. (2015). *System Architecture: Strategy and Product Development for Complex Systems* (1 edition). Boston: Prentice Hall.

Dori, D. (2002). *Object-process methodology: a holistic systems paradigm*. Berlin ; New York: Springer.

Dori, D. (2015, December). *Object-Process Methodology – the new ISO 19450 Standard: Principles and MBSE Applications*. Presented at the INCOSE Webinar Series 85. Retrieved from https://connect.incose.org/Library/Webinars/Pages/INCOSE-Webinars.aspx

Dori, D., Linchevski, C., Manor, R., & Opm, O. M. (2010). OPCAT–An Object-Process CASE Tool for OPM-Based Conceptual Modelling. In *1st International Conference on Modelling and Management of Engineering Processes* (pp. 1–30). Retrieved from http://esml.iem.technion.ac.il/wp-content/uploads/2011/07/OPCAT-Description-and-Demo-to-MMEP-2010-Cambridge-UK.pdf

Dori, D., Renick, A., & Wengrowicz, N. (2015). When quantitative meets qualitative: enhancing OPM conceptual systems modeling with MATLAB computational capabilities. *Research in Engineering Design*, 1–24. http://doi.org/10.1007/s00163-015-0209-9

DyMat. (2015). A package for reading and processing the result files of Dymola and OpenModelica. Retrieved from https://pypi.python.org/pypi/DyMat

Edwards, S., Cilli, M. V., Peterson, T., Zabat, M., Lawton, C., & Shelton, L. (2015). Whole Systems Trade Analysis. In *25th INCOSE Int'l Symposium, Seattle*. Seattle, WA, USA.

Estefan, J. A. (2007). Survey of model-based systems engineering (MBSE) methodologies. *California Institute of Technology, Pasadena, California, USA May*, *25*. Retrieved from http://pdf.aminer.org/000/260/416/towards_a_unified_paradigm_for_verification_and_validation_of_systems.pdf

Frithiof, N., Lindh, E., Sundberg, K., Ekelöw, J., Lake, H., & Ericson, M. (2013). *P.1217 Final report v1.0 Project ÆGIR: A SOLAR POWERED AUTONOMOUS RACE CRAFT.*

Grobshtein, Y., Perelman, V., Safra, E., & Dori, D. (2007). Systems modeling languages: OPM versus SysML. In *Systems Engineering and Modeling, 2007. ICSEM'07. International Conference on* (pp. 102–109). IEEE.

INCOSE. (2011). *Systems Engineering Handbook: A guide for system life cycle processes and activities* (v3.2.2 ed.). International Council of Systems Engineering.

INCOSE. (2014). *A world in motion: Systems Engineering Vision 2025*. INCOSE. Retrieved from http://www.incose.org/newsevents/news/details.aspx?id=321

INCOSE. (2015). *Wiley: INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, 4th Edition - INCOSE*. Retrieved from http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118999401.html

INCOSE UK. (2015, February). Z9: What is Model-Based Systems Engineering? Retrieved April 10, 2015, from http://www.incoseonline.org.uk/Documents/zGuides/Z9_model_based_WEB.pdf

ISO. (2015). *ISO/PAS 19450:2015 Automation systems and integration -- Object-Process Methodology* (No. ISO/PAS 19450:2015). Retrieved from http://www.iso.org/iso/catalogue_detail.htm?csnumber=62274

Modelica Association. (2012). Modelica-A Unified Object-Oriented Language for Systems Modeling Language Specification Version 3.3. *Https://www. Modelica. org/documents/ModelicaSpec33. Pdf.*

Mordecai, Y., & Dori, D. (2015). Agile Modeling of an Evolving Ballistic Missile Defense System with Object-Process Methodology. In *IEEE International Systems Conference*. Vancouver BC, Canada. Retrieved from http://www.researchgate.net/profile/Yaniv_Mordecai/publication/272204847_Agile_Modeling_of_an_Evolving_Ballistic_Missile_Defense_System_with_Object-Process_Methodology/links/54df8c7f0cf2953c22b3ceec.pdf

OMG. (2010). *SysML-Modelica Transformation Specification v1.0* (No. syseng/2010-06-08). OMG. Retrieved from http://www.omg.org/spec/SysML-Modelica/1.0

OMG. (2015). *OMG Systems Modeling Language v1.4*. Retrieved from http://www.omg.org/spec/SysML/1.4/

OMG. (2016). OMG SysML. Retrieved January 10, 2016, from http://www.omgsysml.org/

Scheithauer, D. (2012). Managing Concurrency in Systems Engineering. *INCOSE International Symposium*, *22*(1), 2016–2030. http://doi.org/10.1002/j.2334-5837.2012.tb01453.x

Scheithauer, D., & Forsberg, K. (2013). V-Model Views. *INCOSE International Symposium*, *23*(1), 502–516. http://doi.org/10.1002/j.2334-5837.2013.tb03035.x

SEBoK. (2015). Guide to the Systems Engineering Body of Knowledge (SEBoK). Retrieved from http://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK)

Sutherland, J., Kamiyama, H., Aoyama, K., & Oizumi, K. (2015). Systems Engineering and the V-Model: Lessons from an Autonomous Solar Powered Hydrofoil. In *12th International Marine Design Conference (IMDC)*. Tokyo Japan. Retrieved from http://www.nakl.t.u-tokyo.ac.jp/imdc2015/

Tiller, M., Bowles, P., & Dempsey, M. (2003). Development of a vehicle model architecture in Modelica. In *Paper presented at the 3rd International Modelica Conference*. Retrieved from https://www.modelica.org/events/Conference2003/papers/h32_vehicle_Tiller.pdf

Ulrich, K., & Eppinger, S. (2011). *Product Design and Development* (5 edition). New York: McGraw-Hill/Irwin.

US DoD. (2011). *Work breakdown structures for defence material items* (No. MIL-STD-881C). Retrieved from http://quicksearch.dla.mil/qsDocDetails.aspx?ident_number=36026

Wang, R., Agarwal, S., & Dagli, C. H. (2015). OPM & color petri nets based executable system of systems architecting: A building block in FILA-SoS. In *Systems Conference (SysCon), 2015 9th Annual IEEE International* (pp. 554–561). http://doi.org/10.1109/SYSCON.2015.7116810

Yaroker, Y., Perelman, V., & Dori, D. (2013). An OPM conceptual model-based executable simulation environment: Implementation and evaluation. *Systems Engineering*, *16*(4), 381–390. http://doi.org/10.1002/sys.21235