

**A Unified Product and Project Lifecycle Model
for Systems Engineering**

Research Thesis

In Partial Fulfilment of the Requirements for the
Degree of Doctor of Philosophy

Amira Sharon

Submitted to the Senate of
the Technion - Israel Institute of Technology

Adar, 5770

HAIFA

March 2010

The research thesis was done at the Faculty of Industrial Engineering and Management under the supervision of Professor Dov Dori and Professor Moshe Shpitalni.

The generous financial help of the Technion is gratefully acknowledged.

Table of Contents

Abstract	1
<i>List of Symbols and Abbreviations</i>	2
1 Introduction and Literature Review	3
1.1 Project and Product Characteristics	5
1.2 Systems Engineering Management (SEM) and Project Management (PM)	7
1.3 Systems Engineering and Project Management Standards	10
1.3.1 The MIL 499B draft Standard	10
1.3.2 The IEEE 1220 standard	10
1.3.3 The ANSI/EIA-632 standard	10
1.3.4 The ISO/IEC 15288 standard	11
1.3.5 The INCOSE SE Handbook	11
1.3.6 The CMMI Model	11
1.3.7 The PRINCE2 Approach	12
1.3.8 The PMBOK Guide and the DAG	13
1.3.9 The DoD Guide to IPPD	15
1.4 The Need for Integrating the Project with the Product	16
1.4.1 The Links between Project and the Product within SEM	16
1.4.2 The impact of systems engineering on project parameters	17
1.4.3 The Gap between SEM and PM	20
2 Motivation Questions, and Framework	20
2.1 Research Motivation	20
2.2 The Research Question	21
2.3 Research Framework	21
2.3.1 Project-Product Lifecycle and Conceptual Design	23
2.3.2 Choosing the Underlying Language and Methodology	29
2.4 Content of the Dissertation	31
3 The Combined Project-Product Model	32
3.1 Extended OPM notation for process sequence and duration	36
3.2 The UAV Project Domain	37
3.2.1 Assigning duration to project processes	37
3.2.2 Specifying resources	38
3.2.3 Naming conventions of project objects, processes, and process durations	39
3.2.4 Hierarchical Consistent and Coherent Decomposition	39
3.3 The UAV Product Domain	43
3.4 Simulation-Based Verification of the PPLM Model	47
3.5 Summary	53
4 Application of Project Management Methods within Systems Engineering Management	54
4.1 The Compared Project Management Methods	54
4.1.1 The Critical Path Method and Program Evaluation and Reviewing Technique	54
4.1.2 The Gantt chart	55
4.1.3 The System Dynamics Method	56
4.1.4 The Design Structure Matrix	57
4.1.5 The Earned Value Management	57

4.2	Research Population and Setting	58	
4.3	Research Methodology	60	
4.3.1	The 14 project management factors and their latent dimensions	60	
4.3.2	The survey questions and their analysis method	61	
4.4	Results and Analysis	62	
4.4.1	Methods Comparison by Factors	62	
4.4.2	Methods' perceived suitability with respect to factors	63	
4.4.3	The project dimension	64	
4.4.4	The product dimension	65	
4.4.5	The project-product dimension	65	
4.4.6	The combined project-product dimension	66	
4.4.7	Methods comparison by dimension	66	
4.5	Discussion	68	
4.6	Summary	70	
5	Extraction of Coherent Project Views from the PPPLM Model-Based Plan.....	71	
5.1	Researched PM Methods and Views	71	
5.2	Activities Network Plan and Object & Activities Network Plan.....	73	
5.3	CPM, Object-enhanced CPM, and Critical Objects	75	
5.4	Gantt chart and Model-Based Gantt chart	80	
5.5	Milestones Breakdown Structure	80	
5.6	Process WBS and Model-Based WBS	82	
5.7	Process-Based DSM and Object-Process Model-Based DSM	85	
5.7.1	Constructing the Process-based DSM	85	
5.7.2	Constructing the Model-based Object-Process DSM	89	
5.7.3	Improving the OPM-based Project Model by Inspecting the DSM clustering	92	
5.8	Combined views	104	
5.8.1	Combined WBS-CPM	104	
5.8.2	Critical DSM: Combining DSM and CPM	105	
5.9	Summary	106	
6	Perceived Characteristics of SE Tools and Methods.....	109	
6.1	Introduction	109	
6.2	Research Population and Setting	109	
6.3	Research Methodology	110	
6.4	Results and Analysis	110	
6.4.1	The utility in the organizational hierarchy domain	110	
6.4.2	The utility span over the system complexity domain	112	
6.4.3	The utility span over the professional hierarchy domain	113	
6.4.4	Additional Findings	114	
6.5	Discussion and summary	116	
7	The Combined Project-Product Methodology	117	
7.1	The Generic Project Construct	117	
7.1	Planning to System Builds	117	
8	Perceived Differences between the Gantt and the PPLM OPM-based model.....	124	
8.1	First Stage and Second Stage	125	
8.1.1	Research Population and Setting	125	
8.1.2	Research Methodology	126	

8.1.3	Results and Analysis	130
8.2	Third Stage	142
8.2.1	Research Population and Setting	143
8.2.2	Research Methodology	143
8.2.3	Results and Analysis	145
9	Summary and suggested future work	152
9.1	Summary	152
9.1.1	The first research phase	152
9.1.2	The second research phase	153
9.1.3	The third research phase	154
9.1.4	The fourth research phase	154
9.2	Contributions of this work and suggested future work	155
	References.....	159
	Appendix A – MIT ESD Course Syllabus	163
	Appendix B – MIT ESD course schedule	171
	Appendix C – Stage (1) participants UA V assignment	172
	Appendix D – Stage (2) participants UA V assignment	176
	Appendix E – Stage (3) participants CT scanner assignment	179
	Appendix F – The provided Gantt chart of the CT scanner plan.....	191
	Appendix G – The provided CT scanner PPLM Model	192
	Appendix H – OPM Syntax and Semantics	200

List of Figures

Figure 1-1 The top-level OPM model of the PPLM research process	4
Figure 1-2 The NASA Project Management and Systems Engineering Competency Framework	8
Figure 1-3 ISO/IEC 15288 System Lifecycle Processes	11
Figure 1-4 CMMI Process Areas	12
Figure 1-5 Integration process for an imaginary avionics system	17
Figure 1-6 Examples of DoD programs with reduced buying power	19
Figure 2-1 US DoD Program Lifecycle	24
Figure 2-2 INCOSE Comparison of System Life Cycle phases	25
Figure 2-4 The combined timelines of Project i, which delivers Product i within the Enterprise	26
Figure 2-3 ISO/IEC 15288 Comparison of Breadth and Depth of Key Standard	26
Figure 2-5 Building the project plan iteratively while creating the product model.....	27
Figure 3-1 The top-level view of the UAV PPLM model	33
Figure 3-2 The automatically-generated OPL paragraph of the OPD in Figure 3-1 ...	33
Figure 3-3 OPD of the second level System Diagram (SD1) of UAV PPLM model..	35
Figure 3-4 The automatically-generated OPL paragraph of the OPD in Figure 3-3 ...	35
Figure 3-5 UAV PPLM model with assigned resources.....	38
Figure 3-6 OPD of the UAV Prototype Developing & Integrating In-zoomed.....	40
Figure 3-7 The automatically-generated OPL paragraph of the OPD in Figure 3-6 ...	41
Figure 3-8 The automatically-generated OPL paragraph of the OPD in Figure 3-6 (continued).....	42
Figure 3-9 OPD of UAV Specifying in-zoomed	42
Figure 3-10 The automatically-generated OPL paragraph of the OPD in Figure 3-9 .	43
Figure 3-11 OPD of the Surveillance In-zoomed	44
Figure 3-12 The automatically-generated OPL paragraph of the OPD in Figure 3-10	44
Figure 3-13 OPD of the Surveillance In-zoomed with UAV states.....	45
Figure 3-14 The automatically-generated OPL paragraph of the OPD in Figure 3-13	45
Figure 3-15 OPD of the Preparing In-zoomed.....	46
Figure 3-16 OPD of the Taxi & Takeoff In-zoomed	46
Figure 3-17 The automatically-generated OPL paragraph of the OPD in Figure 3-16	46
Figure 3-18 OPD of UAV Prototype unfolded	47
Figure 3-19 UAV Prototype Developing in-zoomed first simulation view.....	48
Figure 3-20 Definition process in-zoomed first simulation view	49
Figure 3-21 Definition process in-zoomed second simulation view	49
Figure 3-22 Definition process in-zoomed third simulation view	50
Figure 3-23 Vehicle Development process in-zoomed first simulation view.....	50
Figure 3-24 Vehicle Development process in-zoomed second simulation view	51
Figure 3-25 Payload and Engine Development process in-zoomed first simulation view	51
Figure 3-26 Integration and Testing process in-zoomed first simulation view	52
Figure 3-27 Integration and Testing process in-zoomed second simulation view.....	52
Figure 3-28 UAV Prototype Developing in-zoomed last simulation view.....	53
Figure 4-1 Example CPM chart of an Unmanned Aerial Vehicle (UAV) project.....	55
Figure 4-2 Gantt chart of the UAV project.....	56
Figure 4-3 System Dynamics chart of the UAV project	56
Figure 4-4 Example DSM.....	57
Figure 4-5 EVM chart of the UAV project.....	58

Figure 4-6 A rough specification and sketch of the UAV vehicle concept	59
Figure 4-7 Project management Methods Comparison by Sum of Factors Rankings .63	
Figure 4-8 The suitability of each one of the seven PM methods to handle four of the 14 factors.....	64
Figure 4-9 Project management methods comparison by dimensions.....	68
Figure 5-1 The Automatically Extracted Activities Network Plan.....	73
Figure 5-2 The Objects & Activities Network Plan (OANP) PPLM view of the network in Figure 5-1.....	74
Figure 5-3 The automatically extracted Critical Path	76
Figure 5-4 The Object-enhanced Critical Path Method (OCPM) PPLM view.....	77
Figure 5-5 The Critical Path Depicted in the OPM Model – OPD of UAV Prototype Developing & Integrating	78
Figure 5-6 The Critical Path Depicted in the OPM Model – OPD of UAV Specifying	78
Figure 5-7 The Critical Path Depicted in the OPM Model – OPM of Payload & Engine Developing.....	79
Figure 5-8 The Critical Path Depicted in the OPM Model – OPD of Vehicle Developing.....	79
Figure 5-9 The Critical Path Depicted in the OPM Model – OPD of Integrating & Testing.....	80
Figure 5-10 The Gantt PPLM view	81
Figure 5-11 The milestone construct	82
Figure 5-12 First four MBS levels of the UAV project	82
Figure 5-13 Automatically extracted process WBS for UAV Specifying	83
Figure 5-14 Automatically extracted process WBS for Integrating & Testing	83
Figure 5-15 An automatically extracted partial OWBS for UAV Specifying	84
Figure 5-16 An automatically extracted OWBS for Integrating & Testing.....	85
Figure 5-17 The Project Plan Activities DSM.....	89
Figure 5-18 OPDSM Sliding View for the First Seven Processes.....	90
Figure 5-19 The Project Plan OPDSM	91
Figure 5-20 OPD of SD1 - UAV Prototype Developing in-zoom.....	93
Figure 5-21 The automatically-generated OPL paragraph of the OPD in Figure 5 2093	
Figure 5-22 OPD of SD1.1 - Definition in-zoomed	95
Figure 5-23 The automatically-generated OPL paragraph of the OPD in Figure 5-2295	
Figure 5-24 OPD of SD1.2 - Integration and Testing in-zoomed.....	96
Figure 5-25 The automatically-generated OPL paragraph of the OPD in Figure 5-2496	
Figure 5-26 OPD of SD1.3 - Payload and Engine Development in-zoomed	97
Figure 5-27 The automatically-generated OPL paragraph of the OPD in Figure 5-2697	
Figure 5-28 OPD of SD1.4 - Vehicle Development in-zoomed.....	98
Figure 5-29 The automatically-generated OPL paragraph of the OPD in Figure 5-2898	
Figure 5-30 The Project Plan Activities DSM with iterations	102
Figure 5-31 Manually-clustered DSM matrix for the UAV project	103
Figure 5-32 Automatically Extracted Combined WBS-CPM for the UAV Specifying node.....	105
Figure 5-33 Automatically extracted Combined WBS-CPM for the Integration & Testing node.....	106
Figure 5-34 The Project Plan OPDSM with designated critical things (processes and objects).....	107
Figure 6-1 The utility span over the organizational hierarchy domain.....	111
Figure 6-2 The utility span over the system complexity continuity	112

Figure 6-3 The utility span over the professional hierarchy domain	114
Figure 7-1 The Generic Project Construct (GPC) of the Model-Based Project Plan	118
Figure 7-2 The combined product-project model System Build (SB) hierarchy	120
Figure 7-3 The generic lifecycle of a System Build	120
Figure 7-4 Five planned SBs spread over the project lifecycle	121
Figure 7-5 PPLM model of a SUD Developing process	122
Figure 7-6 PPLM plan of Developing Subsystem A	123
Figure 8-1 Object's states and equivalent objects.....	129
Figure 8-2 Time to produce Gantt chart and PPLM models.....	130
Figure 8-3 Number of processes and objects in the PPLM model – Stage 1.....	132
Figure 8-4 Averages of objects quantity distribution by types – Stage 1	132
Figure 8-5 Number of processes and objects in the PPLM model – Stage 2.....	139
Figure 8-6 Medians of objects quantity distribution by types – Stage 2	139
Figure 8-7 Total processes number comparison - Stage 2 against Stage 1.....	140
Figure 8-8 Total objects number comparison - Stage 2 against Stage1	141
Figure 8-9 Percentage of objects types comparison - Stage 2 against Stage 1	142
Figure 8-10 The distribution of answers to all 33 questions.....	147
Figure 8-11 PPLM vs. Gantt over the 33 questions span	148

List of Tables

Table 1-1 OPM Building Blocks	4
Table 1-2 DAG systems engineering standardized process terminology	9
Table 1-3 DAG Roles of Systems Engineering in System Life Cycle Processes.....	9
Table 1-4 INCOSE SE Handbook V3 Systems Engineering Processes	12
Table 1-5 PRINCE2 Processes and Activities	13
Table 1-6 Mapping PMBOK Processes onto DAG Processes.....	14
Table 1-7 Mapping PMBOK Processes onto DAG Processes (continued)	15
Table 2-1 Summary of Research Objectives and Framework	22
Table 2-2 IEEE 1220 Standard System Life Cycle	24
Table 2-3 Organization of the dissertation.....	31
Table 3-1 Classification of objects in the OPM project plan.....	34
Table 3-2 New OPM constructs defining time notion	36
Table 3-3 New OPM constructs defining time notion (continued).....	37
Table 4-1 The seven investigated project management methods	54
Table 4-2 The seven investigated project management methods	59
Table 4-3 The 14 Systems Engineering Management Factors	61
Table 4-4 All Factors Set Reliability	63
Table 4-5 The Project Dimension (factors 1, 2, 4, and 11).....	65
Table 4-6 The Product Dimension (factors 6, 7, 8, and 9).....	65
Table 4-7 The Project-Product Dimension (factors 3, 5, 10, 12, 13, and 14).....	65
Table 4-8 The Combined Project-Product Dimension (factors 1, 2, 4, and 11 combined with factors 6, 7, 8, and 9).....	66
Table 4-9 Sums calculations for comparison of methods by dimensions.....	67
Table 4-10 Summary of methods comparison findings	69
Table 5-1 Different types of Dependency Structure Matrix (DSM).....	72
Table 5-2 List of 23 Activities of the UAV project	73
Table 5-3 List of 23 Activities of the UAV project (continued).....	74
Table 5-4 OPM to DSM Extraction Data	87
Table 5-5 OPM to DSM Extraction Data (continued)	88
Table 5-6 Updated OPM to DSM Extraction Data.....	100
Table 5-7 Comparisons of DSM Clustering with OPM Clustering.....	104
Table 7-1 The SB-related definitions for the combined Product-Project Methodology	119
Table 8-1 The three research stages	124
Table 8-2 The three research stages (continued)	125
Table 8-3 Research population and settings for stage (1) and stage (2)	126
Table 8-4 Stage 1 participants group partition.....	127
Table 8-5 Comparison between Stage 1 and Stage 2 research populations.....	127
Table 8-6 Comparison between Stage 1 and Stage 2 research populations (continued)	128
Table 8-7 Expected results.....	129
Table 8-8 The structure of the questionnaire used in Stage 3	144
Table 8-9 The two assignment parts in stage 3	144
Table 8-10 Designed non-equivalence in the check questions	145
Table 8-11 Paired samples statistics for PPLM and Gantt chart	146
Table 8-12 Paired Samples Test for PPLM and Gantt chart.....	146
Table 8-13 Stage 3 detailed results for each question.....	148
Table 9-1 Issues of potential contribution of this research	155

Abstract

Project and product are two complementary facets of the lifecycle of any complex man-made system. The *project* focuses on the early phases of the system to be delivered, deployed, and supported, while the *product* focuses on the system itself – its function, structure, and behavior. Conceptual modelling and design is a major area common to both the project and the product, since evidently, *the product* is the deliverable of the *project*. Traditionally, however, the project and the product entities have been addressed as separate domains, each with its dedicated approaches, methods, and tools. This separation has hindered the integration of the project with the product it delivers, missing potential tangible benefits for all the stakeholders involved.

Systems Engineering Management (SEM) is an emerging practice that is being developed hand in hand with the maturation of systems engineering. Standards for SEM account for the intimate relationships between SEM and Project Management (PM) and highlight the criticality of these relationships in improving systems project management. While PM methods have traditionally focused on scheduling, budgeting, and scope management, SEM emphasizes the management of the project-product ensemble and issues related to the technologies of the system under development. The actual practice of systems engineering management involves continuous iterative zigzagging between the two domains – the systems engineering domain and the project management domain. This zigzagging is a cognitive process of understanding the intricate relationships between the *product domain* and the *project domain*, and planning the SEM efforts accordingly.

What the product-project ensemble has been lacking is a common underlying ontology, a conceptual model, and a supporting software environment. Attaining these missing elements enables the simultaneous expression of the function, structure and behavior of the project and the product. This thesis presents a model-based approach to managing the lifecycle of the product to be developed hand-in-hand with the lifecycle of the project, within the scope of which the product is developed. The cornerstone of this Project-Product Lifecycle Management (PPLM) approach is an underlying holistic conceptual model, supported by software capabilities for an integrated project and product lifecycle environment. The concurrent project-product model, built on common ontological foundations, enables better management, making it possible to directly link entities in one subsystem to those in the other. The expected value of the holistic, integrated conceptual model is the provision of both superior product lifecycle engineering and project management capabilities, yielding significant cut in time to market, reduced risk, and higher product quality.

List of Symbols and Abbreviations

APPEL	Academy of Program / Project & Engineering Leadership
ANSI	American National Standards Institute
CMMI	Capability Maturity Model Integrated
CSCI	Computer Software Configurable Items
DAG	Defence Acquisition Guidebook
DID	Data Item Description
DoD	Department of Defense
DoDAF	Department of Defense Architecture Framework
EIA	Electronic Industries Alliance
GPC	Generic Project Construct
HWCI	Hardware Configurable Items
IEC	International Electronics Commission
INCOSE	International Council on Systems Engineering
IMS	Integrated Master Schedule
IPPD	Integrated Product and Process Development
ISO	International Organization for Standardization
MBS	Milestones Breakdown Structure
MDA	Model-Driven Architecture
MBPP	Model-Based Project Product
MoDAF	Ministry of Defense Architecture Framework
NASA	National Aeronautics and Space Administration
OPCAT	Object-Process CASE Tool
OPD	Object-Process Diagram
OPL	Object-Process Language
OPM	Object-Process Methodology
PC	Program Control
PERT	Program Evaluation and Review Technique
PMBOK	Project Management Body of Knowledge
PM	Project Management
PMI	Project Management Institute
PLM	Product Lifecycle Management
PPLM	Project-Product Lifecycle Management
PRINCE	PRojects IN Controlled Environments
RFP	Request for Proposal
SD	System Diagram
SE	Systems Engineering
SEM	Systems Engineering Management
SEMP	Systems Engineering Management Plan
SEMS	Systems Engineering Master Schedule
SEP	Systems Engineering Plan
SOW	Statement of Work
UML	Unified Modelling Language
WBS	Work Breakdown Structure
TOGAF	Open Group Architecture Framework

1 Introduction and Literature Review

Complex systems in general and software-intensive ones in particular comprise involved relationships of numerous physical and informatical objects interacting via intricate processes. Systems Engineering Management (SEM), which aims to develop and sustain such large-scale systems, encompasses three tightly intertwined systems: the *product*, the *project* and the *enterprise*. The product (e.g., an airplane or a medical device) is the object resulting from the project, which can be thought of as the *process of achieving the product* and is expected to ultimately yield the product with its support environment. Finally, the *enterprise* is the organization in charge of the project of delivering the product.

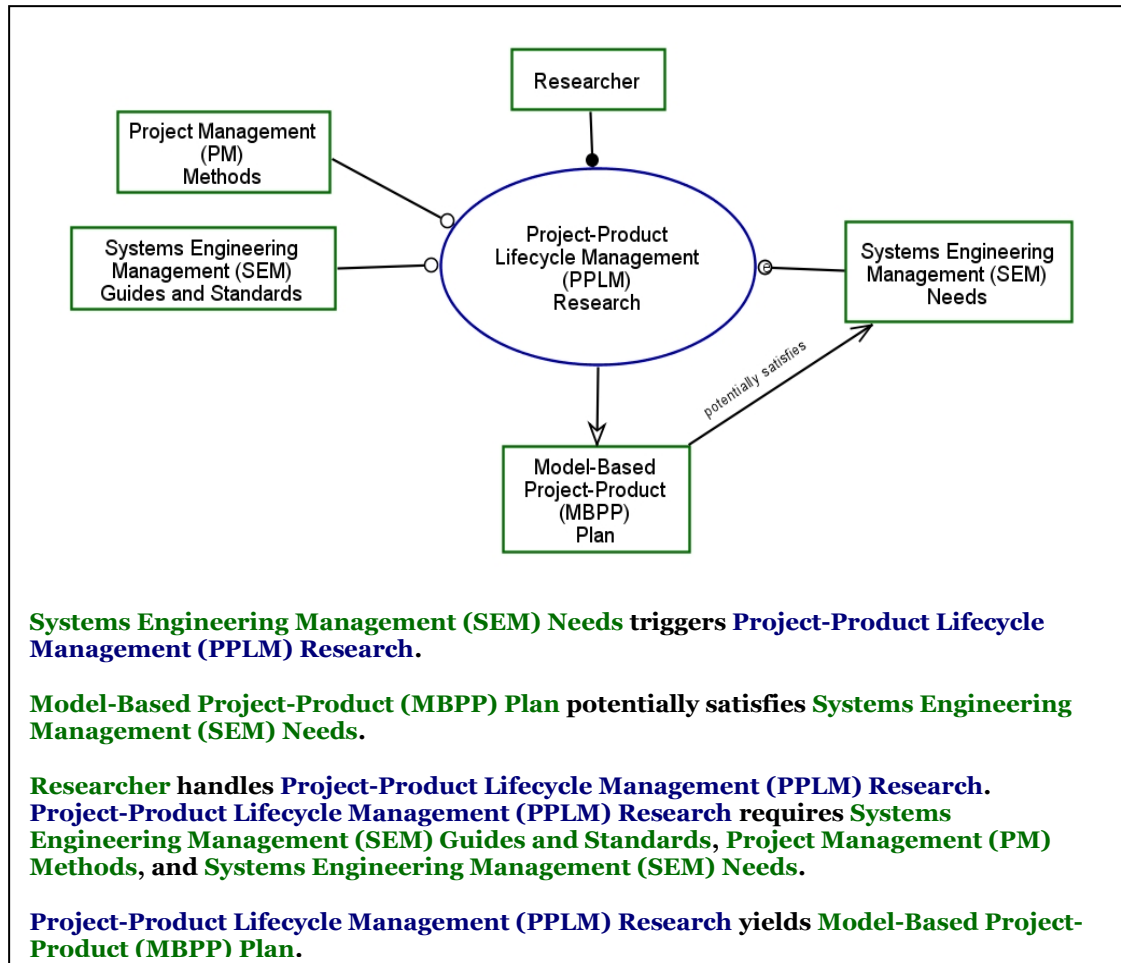
Project and product are two complementary facets of the lifecycle of any complex man-made system. The *project* focuses on the early phases of the system to be delivered, deployed, and supported, while the *product* focuses on the system itself – its function, structure, and behavior. While we refer to product as the expected outcome of the project, it is often the case that the outcome is not a new product in the classical sense but rather a one-time large-scale system, such as an intelligent cross-country highway or a vehicle rescue system. Hence, we interpret product widely as a "whole system product," which includes not just the delivered product but also the support infrastructure and the services provided to keep the product operational and beneficial.

Conceptual design is a major area common to both the project and the product. Starting a conceptual modelling process early on during the product lifecycle greatly benefits the final product outcome. The conceptual model describes the product's hierarchy of functions and its architecture—the combination of structure and behavior that enables the product to function optimally. The conceptual model also defines the product boundaries and its relation with the environment, providing a sound basis for system-level comprehension by different stakeholders, including the customer, systems engineering managers, systems engineers, project managers, system architects, and the multidisciplinary teams carrying out the project.

The conceptual model included in this thesis utilizes Object-Process Methodology (OPM), which is a comprehensive approach to systems modelling, engineering, and lifecycle support [1]. OPM integrates the object-oriented and process-oriented paradigms into a single frame of reference, in which the function, structure, and behavior of a system are reflected in a unified, holistic model. The elements of the OPM ontology are entities and links. The entities are objects and processes, collectively called things, and states. Table 1-1 presents OPM entities and two representative links. Appendix H provides the OPM Syntax and Semantics.

Figure 1-1 depicts the top-level OPM model of the PPLM research process and environment.¹ According to this basic model, the PPLM research process requires information from two sources: Project Management (PM) Methods, and Systems Engineering Management (SEM) Guides and Standards. The research process ultimately produces the Model-Based Project-Product (MBPP) Plan. This plan potentially satisfies the Systems Engineering Management (SEM) Needs, which are the trigger to the research in the first place.

¹ The text below the diagram is an integral part of the OPM model. This bi-modal representation (i.e., a diagram and its semantically-equivalent text) demonstrates one of the features of OPM that improves model comprehension.



1 Figure -1 The top-level OPM model of the PPLM research process

Table 1-1 OPM Building Blocks

Element	Description	Symbol
Object	A thing that exist	
Process	A thing that transform objects	
State	At any point in time an object can be in exactly one state or in transition between two states.	
Procedural Link	Connects an object and a process to describe the behavior of a system.	
Structural Link	Expresses static relation between a pair of objects or a pair of processes.	

² A single representative link out of nine links included in the OPM lexicon.

³ A single representative link out of four links included in the OPM lexicon.

1.1 Project and Product Characteristics

The most common definitions for a "Project", also referred to as "Program", use terms such as "beginning and end", "activities" or "phases", which include "planning, execution and completion". These terms are easily and obviously reflected by lifecycle representations, making the term "Project Lifecycle" intuitive. Systems engineering activities span the entire program lifecycle from systems requirements elicitation, definition, analysis, and conceptual design at the outset of a program through production, operational support, planning for replacement, and eventual retirement and disposal at the end of a program. The most obvious deliverable of a typical engineering project is a unique product. Hence, the project's end result is different than the results of other functions of the organization. A project is a non-recurring, large-scale effort to create a system that attains the goal of solving a problem or satisfying a need [1].

The building blocks of a project are work plan activities, which are aggregated into work packages. For each activity, resources are allocated, while constraints and risks need to be considered. At this stage, the product is populated with requirements that were elicited from the customer, components and interfaces that comprise the product architecture, functions that comprise the product behavior, and other artifacts. The research is aimed at developing a generic model that relates the project and product building blocks to each other in the combined PPLM model.

In this work we use the terms system and product interchangeably. In systems engineering, the product is usually referred to as *system*, encompassing its end-to-end performance over its entire lifecycle span. The systems engineering domain has offered many definitions of a system [2], [3], [4], [5], [6]. Common in these definitions is the notion of a system as a collection of elements that function together to achieve one or more stated purposes. The final version of the *system* is the ultimate deliverable of the project – the functioning final form of the value-providing product and its support environment that is delivered to the customer. Within project management, the term *product* is more customary.

The function of an artificial system is derived from the intent, goal, or purpose, for which the system was constructed. Function describes *what* the system shall do, does or can do. It is the behavior of the system, the activities, operations and transformations that cause, create or contribute to its value-providing performance, i.e., meeting the system's goals, or the actions for which a thing exists or is employed. The *goal* is at the basis of the function.

The system's *architecture* is the particular combination of structure and behavior that enables materialization of its function. Very similar or even identical functions can be achieved by systems that employ different architectures (structure-behavior combinations). The system's function can be architected, understood, and explained from two major complementary aspects: structure (or form) and behavior (or dynamics).

Systems architecting has matured through thousands of years of human experience and practice in different engineering fields, notably civil architecture. Current complex computer-based systems, involving numerous software and hardware artifacts, have raised the need to handle their complexity, which gradually evolved into systems architecture and the more comprehensive systems engineering domain.

Systems architecture holistically represents the combination of human, software and hardware components, their functions, relations and interconnections, and the interaction of the overall system with its environment. Through the architecture

documentation, systems architects and engineers specify requirements and constraints derived from the different stakeholders—developers, test engineers, users, clients, bystanders, and regulatory agencies—and defines how the system satisfies its functional and non-functional requirements.

The architecture, in turn, forms the basis for all the further system engineering stages downstream. A robust architecture is one that exhibits acceptable fault-tolerance, modifiability, reliability, maintainability, availability, and other desirable "non-functional" traits. The system architect is influenced by the system environment needs, direct and derived requirements, her or his professional experience, the developing organization, and other stakeholders, such as the system users and clients. Due to the importance of systems architecture in modern systems, various frameworks have evolved to support system architecting. Some of the most widespread frameworks are briefly surveyed below.

The Zachman Framework [7] is a framework for enterprise architecture that is based on a two-dimensional matrix defining six major concerns—What, How, Where, Who, When, and Why—as columns, and six model types—Visionary, Owner, Designer, Builder, Implementer and Worker—related to the different stakeholders. The framework is criticized as being bureaucratic and monolithic.

The Department of Defense Architecture Framework (DoDAF) [8] is a computer-based systems that builds on on the Zachman Framework. It has been developed for the US Government Department of Defense (DoD) and systems it acquires are required to be developed according to DoDAF. DoDAF has 26 views that are organized into four basic sets: overarching All View (AV), Operational View (OV), Systems View (SV), and the Technical Standards View (TV).

The Ministry of Defense Architecture Framework (MoDAF) [9] is based on the DoDAF framework and elaborated by the United Kingdom Ministry of Defense. MoDAF extends the DoDAF views set by strategic and acquisition views.

The Open Group Architecture Framework (TOGAF) [10] is a framework enterprise architecture which includes business processes, applications, data architecture, and technology architecture. TOGAF defines a methodology for the representation of the system building blocks and how they fit together. It also includes a list of recommended standards and tools.

As noted, system architecture is the underlying combination of structure (expressed in terms of objects and their attributes) and behavior (expressed in terms of processes and how they transform objects) that enables a system to attain its intended function. It is therefore a mandatory precursor to successful design, implementation, and evolution of artificial complex systems in general and products in particular. Proper system architecting relies on timely availability of reliable information for analysis and synthesis, which current industrial practices hardly provide. Common, shared project-product ontology will provide the much needed basis for a systematic description of the product and its realizing project, enabling the documentation of design rationale and tradeoffs that must eventually be made between required or desired product functionality on one hand and meeting project milestones, which determine time-to-market, on the other hand.

Dubbed "Model-Driven Architecture" (MDA), the realization that modern software systems should be based on a conceptual model from early lifecycle phases has become commonplace in recent years. General systems are next in line for being "model-driven". Having recognized the commonality among a large variety of systems, product lifecycle engineering combines a host of scientific, technological, ecological, and human aspects in a highly multidisciplinary fashion. Recently, the

notion of Product Lifecycle Management (PLM) has started to make headways not just in academic echelons, but also in leading enterprises, which have identified the need for a comprehensive approach to ensure that systems and products be managed by best industry practices backed by sound methodology.

1.2 Systems Engineering Management (SEM) and Project Management (PM)

Systems Engineering Management (SEM) and Project Management (PM) are two tightly intertwined domains. This observation is expressed in at least two prominent Systems Engineering (SE) handbooks. The first is the Systems Engineering Handbook of the International Council on Systems Engineering [2], which provides framework and guidelines for SE and addresses the strong relationships between SE and PM. The second is the NASA Systems Engineering Handbook [3], over one third of which is devoted to "management issues in systems engineering". Indeed, indicated in this Handbook is the fact that it covers topics that are also considered to be in the domains of Project Management/Program Control (PM/PC), "reflecting the unavoidable connectedness of these three domains".

Along these lines, the INCOSE SE Handbook includes under Systems Engineering process elements of planning, scheduling, reviewing, and auditing. It calls for including within the systems engineering deliverables the Systems Engineering Management Plan (SEMP) and the Systems Engineering Master Schedule (SEMS). In doing so, the SE Handbook further underscores the tight links and dependencies between the project management and the systems engineering domains.

The literature has been struggling with delineating the border between the two domains. For example, the INCOSE SE Handbook [2] indicates that "although there are some important aspects of project management in the Systems Engineering process, it is still much more of an engineering discipline than a management discipline. It is a very quantitative discipline, involving trade-off, optimization, selection, and integration of the products of many engineering disciplines." [2]. This definition is rather fuzzy, as it implies that PM is less "quantitative" than SEM, but PM does involve the quantities of time and budget, while SEM has managerial aspects that are not too far from those of PM. Perhaps a better distinction would be that PM focuses on the triangle of scheduling, budgeting, and scope management, while SEM emphasizes the management of the project-product ensemble. However, this thesis is aimed at bringing PM and SEM closer and show how the two seemingly disparate domains are in fact two facets of the same project-product super system [11], or ensemble. This is not a system of systems, since it does not conform to two of Maier's criteria [12] for identifying system of systems. The project system and the product system on their own exhibit neither operational independence of elements, nor managerial independence of elements. Although the project-product ensemble exhibits evolutionary development and emergent behaviors, the project does not exist independently of the product it is supposed to deliver. OPM defines a process as a transformation (generation, consumption of state change) of an object. In the context of this thesis, the project is the process that generates the product—the object.

The Academy of Program/Project & Engineering Leadership (APPEL) supports NASA's goal to bridge programs/projects and the academic community [13]. The cornerstone for applied research is the Centre for Program/Project Management Research (CPMR) which was formed as a partnership between NASA/APPEL and the Universities Space Research Association (USRA) to "engage universities in world-

class research that addresses internationally significant problems in the discipline of Project Management." The APPEL core curriculum provides fundamental knowledge for NASA's technical workforce. It contains a two-week course titled Project Management and Systems Engineering (PM&SE) that is designed for NASA project practitioners and systems engineers prior to or in the first year of entry into a project, a systems engineering or a functional supervisory position. A second four-day course, titled Advanced Project Management and Advanced Systems Engineering, is also designed for technical professionals who have had prior experience in projects or systems at a supervisory level. As depicted in Figure Figure 1-2, the NASA Project Management and Systems Engineering Competency Framework consists of five project management competency areas, three systems engineering competency areas, and five competency areas common to both the project management and systems engineering communities.



Figure 1-2 The NASA Project Management and Systems Engineering Competency Framework

Each of the three sections of the framework includes a summary overview of definitions of each of the competency areas and a list of the respective underlying competencies. Each competency area describes, in broad terms, what is expected of NASA Project Management and Systems Engineering personnel in terms of particular components or functions of the job. Each major Competency Area has underlying competencies that provide examples of the knowledge, skills, and behaviors that Project Managers and Systems Engineers are expected to possess and/or perform at different levels of job responsibilities. These skills, behaviors, and knowledge demonstrate performance to be achieved for successful mastery of each competency and are expressed in terms of levels.

The Defence Acquisition Guidebook (DAG) [14] is an online resource provided by the DoD, containing “fundamental principles and procedures that the Department follows” in achieving objectives that are described in DoD Directive 5000.01 and DoD Instruction 5000.02. It is “designed to complement those policy documents by providing the acquisition workforce with discretionary best practice that should be tailored to the needs of each program”. It contains eleven chapters, each dealing with a separate subject of program management, from which Chapter 4, devoted to systems engineering in acquisition programs, presents generic systems engineering processes, grouped in two categories: Technical Management Processes and Technical Processes, as shown in Table 1-2. According to this DAG, the program manager uses the technical management processes to manage the technical development of the

system. The roles of the program manager and the program systems engineer relate in the context of the system lifecycle processes, as shown Table 1-3, deriving close work and intricate relationships. Eight technical management processes run throughout the system lifecycle. A chapter on systems engineering describes systems engineering techniques and tools for management, oversight, and analysis and provides some general knowledge management resources, including Systems Engineering Plan (SEP), Integrated Master Plan (IMP), Integrated Master Schedule (IMS), Earned Value Management (EVM) and Work Breakdown Structure (WBS).

Table 1-2 DAG systems engineering standardized process terminology

Technical Management Processes	Technical Processes
Decision Analysis	Stakeholders Requirements Definition
Technical Planning	Requirements Analysis
Technical Assessment	Architectural Design
Requirements Management	Implementation
Risk Management	Integration
Configuration Management	Verification
Technical Data Management	Validation
Interface Management	Transition

Table 1-3 DAG Roles of Systems Engineering in System Life Cycle Processes

Life-cycle Processes	Program Manager	Chief / Systems Engineer
Stakeholder Management	Primary	Support
Technical Planning	Support	Primary
Decision Analysis	Primary	Support
Technical Assessment (Includes Program Status: Technical Progress, Schedule & Cost Management)	Shared	Shared
Configuration Management	Primary	Support
Data Management	Primary	Support
Requirements Management	Support	Primary
Contract Management	Primary	Support
Requirements Analysis	Support	Primary
Architecture Design	Support	Primary
Implementation	Support	Primary
Risk Management	Primary	Support
Interface Management	Support	Primary
Integration	Support	Primary
Verification	Support	Primary
Validation	Shared	Shared

Reading through the DAG's systems engineering chapter, one recognizes the efforts made to delineate the border between SEM and PM. Much of the need to draw the line between the two is rooted in historical reasons of the engineering and management domains growing as disparate disciplines in both academia and industry. The prevailing view was that engineers are professionals who got their education in engineering schools and master the scientific and technological aspects of the system or product to be delivered, while managers are a different kind of professionals, taught primarily in business schools to manage people, enterprises, and projects, but are much less verse in the science and technology aspects of the task at hand.

Lack of communication between project and SE managers hinders the progress of the project-product ensemble. To remedy this, a common language is needed to improve this communication. Ideally, a common ontology and a balanced mix of engineering and managerial skills is required to successfully run a real-life large-scale project, especially when the end result of the project is a complex functioning system or product. Due to the multidisciplinary and highly complex nature of current and currently developed systems, engineering efforts apply science and technology, as well as technical planning, management, and leadership activities [15]. Systems engineering managers must therefore rely on a combination of technical skills and management principles that address both complex technical and managerial issues. Following this train of thoughts, we adopt the following definition of SEM.

Systems Engineering Management is the integration of systems engineering and the parts of project management related to systems engineering.

1.3 Systems Engineering and Project Management Standards

Over the years, standards for systems engineering and project management with different foci and emphasis have evolved. In this section we briefly survey leading pertinent standards and their evolution.

1.3.1 The MIL 499B draft Standard

The MIL 499B draft Standard "Systems Engineering" [4] lists 22 key program tasks that are conducted during a typical program lifecycle. These tasks provide a broad perspective on the Systems Engineering process from "Need" to "Disposal", specifically addressing managerial issues, such as cost effectiveness, risk management, and configuration management. It also provides guidelines for systems engineering master schedule (SEMS) and systems engineering management plan (SEMP).

1.3.2 The IEEE 1220 standard

The IEEE 1220 standard "Application and Management of the Systems Engineering Process" [5] specifies the requirements for the systems engineering process (SEP) and its application throughout the product lifecycle. It details policies and procedures for systems engineering the enterprise shall develop and maintain for governing the conduct of the SEP. These policies and procedures specify requirements for planning, implementation, and control of product and process development and human/systems integration. It addresses the preparation and approval of a SEMP, a SEMS, a systems engineering detailed schedule, and ways to monitor and report technical progress of the project. In doing so, it points to the needed integration between the product/system and project aspects.

1.3.3 The ANSI/EIA-632 standard

The ANSI/EIA-632 standard "Processes for Engineering a System" [6] is an integrated set of fundamental processes for engineering a system. It lists 13 processes categorized under five topics, one being technical management, and 33 requirements on these processes. Technical management includes a planning process, an assessment process, and a control process. Systems engineering is considered as the coordination of processes, methods, and tools. Project-level implementation is tailored from the enterprise policies and procedures established by the standard, including project plans

and schedules, Statement of Work (SOW), and Work Breakdown Structure (WBS). Clearly, these are project management facets of the technical management.

1.3.4 The ISO/IEC 15288 standard

ISO/IEC 15288 standard "Systems and software engineering—System life cycle processes" [16] details enterprise processes that manage the organization's capability to acquire and supply products or services through the initiation, support, and control of projects (see Figure 1-3). The standard's lifecycle model is expressed in terms of 25 processes, seven of which are "project processes." In addition to the planning, assessment, and control technical management processes of ANSI/EIA-632, it lists four management processes: decision management, risk management, configuration management, and quality management.

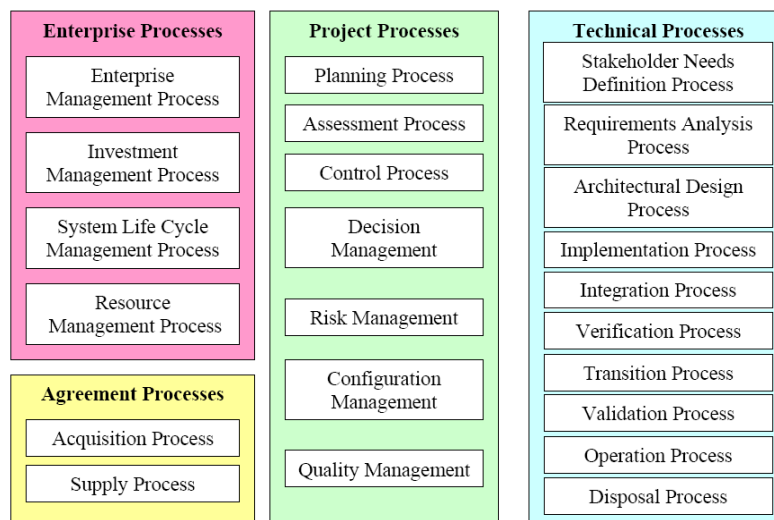


Figure 1-3 ISO/IEC 15288 System Lifecycle Processes

1.3.5 The INCOSE SE Handbook

The INCOSE SE Handbook V3 [17] is the new updated version, which is consistent with ISO/IEC 15288. As listed in Table 1-4, it categorizes the systems engineering processes into Technical Processes (Chapter 4), Project Processes (Chapter 5), and Enterprise and Agreement Processes (Chapter 6). The INCOSE Technical Processes resemble the DAG Technical Processes, while the INCOSE Project Processes are quiet similar to the DAG Technical Management Processes (see Table 1-2).

1.3.6 The CMMI Model

Defined as a "process improvement approach that provides organizations with the essential elements of effective processes", the Capability Maturity Model Integration (CMMI) standard [18] provides a deeper level of detail than its predecessors over a larger breadth of scope (see Figure 1-4). In addition to addressing eight "process areas," categorized under "project management," CMMI explicitly requires that certain relationships between the product domain and the project domain be maintained. In particular, CMMI Specific Practice SP 1.4 entails maintaining bidirectional traceability among the system or product requirements, the project plans, and the work products. As these three elements keep changing throughout the

product-project lifecycle, they require an updating mechanism. Traceability is particularly needed in conducting impact assessment of requirements changes on the project plans, activities, and work products. CMMI Specific Practice SP 1.5 calls for identifying inconsistencies between the project plans, work products, and the requirements via "documentation of inconsistencies, including sources, conditions, and rationale," and taking corrective actions as needed. However, no specific method for maintaining bidirectional traceability among the requirements and the project plans and work products is stipulated.

Table 1-4 INCOSE SE Handbook V3 Systems Engineering Processes

Technical Processes	Project Processes	Enterprise and Agreement Processes
<ul style="list-style-type: none"> • Stakeholder Requirements Definition • Requirements Analysis • Architectural Design • Implementation • Integration • Verification • Transition • Validation • Operation • Maintenance • Disposal 	<ul style="list-style-type: none"> • Project Planning • Project Assessment • Project Control • Decision Making • Risk and Opportunity Management • Configuration Management • Information Management 	<ul style="list-style-type: none"> • Enterprise Environment Management • Investment Management • System Life Cycle Process Management • Resource Management • Quality Management • Acquisition • Supply

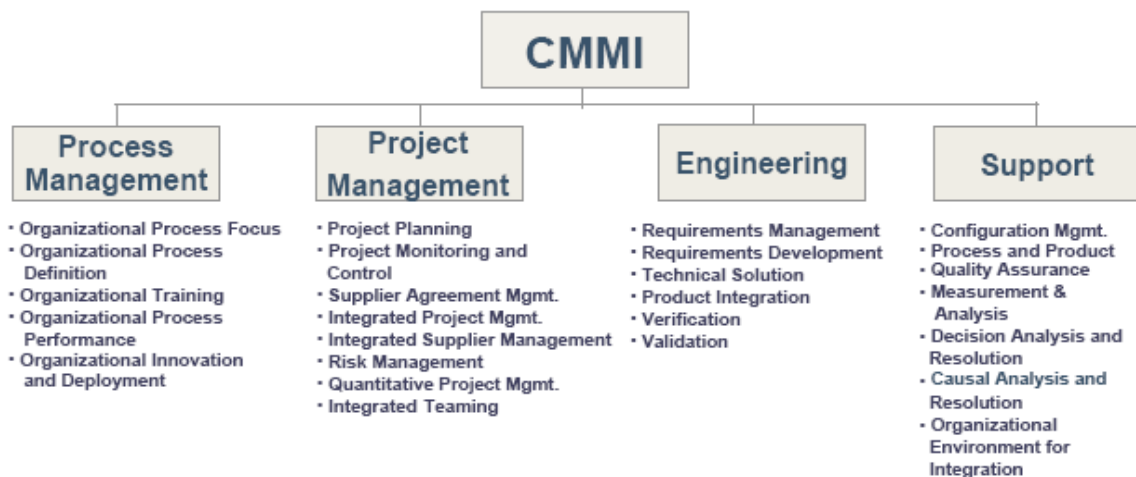


Figure 1-4 CMMI Process Areas

1.3.7 The PRINCE2 Approach

PRINCE2 (PProjects IN Controlled Environments) [19] is a project management approach that is used mainly in the UK, Western Europe and Australia. PRINCE2 is a process-driven project management methodology that defines classifies activities into higher level processes, as listed in Table 1-5. PRINCE2 promotes a product-based planning approach to project planning. It identifies all of the project deliverables that

make up or contribute to delivering the objectives of the project, and identifies the associated work required to deliver them [20].

Table 1-5 PRINCE2 Processes and Activities

Process	Key Activities Included
Starting up a project	<ul style="list-style-type: none"> • Appointing an executive and a <u>project manager</u> • Designing and appointing a project management team • Preparing a project brief • Defining the project approach • Planning the next stage (initiation).
Initiating a project	<ul style="list-style-type: none"> • Planning quality • Planning a project • Refining the business case and risks • Setting up project controls • Setting up project files • Assembling a Project Initiation Document.
Directing a project	<ul style="list-style-type: none"> • Authorizing initiation • Authorizing a project • Authorizing a stage or exception plan • Giving ad-hoc direction • Confirming project closure.
Controlling a stage	<ul style="list-style-type: none"> • Authorizing work package • Assessing progress • Capturing and examining project issues • Reviewing stage status • Reporting highlights • Taking corrective action • Escalating project issues • Receiving a completed work package
Managing stage boundaries	<ul style="list-style-type: none"> • Planning a stage • Updating a project plan • Updating a project business case • Updating the risk log • Reporting stage end • Producing an exception plan.
Closing a project	<ul style="list-style-type: none"> • Decommissioning a project • Identifying follow-on actions • Project evaluation review

1.3.8 The PMBOK Guide and the DAG

The Guide to the Project Management Body of Knowledge (*PMBOK Guide*) [21], published by the Project Management Institute (PMI), has become a standard and turned to be internationally recognized as the leading project management guide. It offers a process-based approach, which can be applied to a wide range of projects. The guide lists 44 processes that overlap and interact throughout a project or its various phases. The processes are categorized into five basic process groups and nine knowledge areas. The five basic process groups are Initiating, Planning, Executing, Controlling and Monitoring, and Closing. The nine knowledge areas are Project Integration Management, Project Scope Management, Project Time Management,

Project Cost Management, Project Quality Management, Project Human Resource Management, Project Communications Management, Project Risk Management, and Project Procurement Management.

Table 1-6 maps PM processes to SE processes as defined by the *PMBOK Guide* and the DAG, respectively. The Technical Management Processes are one of the two categories of processes defined in the DAG under generic systems engineering processes (see Table 1-2). The processes mapping in Table 1-6, shows that overall there is overlap, and only few areas are unique to PM.

Table 1-6 Mapping PMBOK Processes onto DAG Processes

PM	SE
PMBOK Processes [14]	DAG Technical Management Processes [4]
KA 4. Project Integration Management	
4.1 Develop Project Charter	Technical Planning
4.2 Develop Preliminary Project Scope Statement	Technical Planning
4.3 Develop Project Management Plan	Technical Planning
4.4 Direct and Manage Project execution	Decision analysis
4.5 Monitor and Control Project Work	Technical Assessment
4.6 Integrated Change Control	Configuration Management Technical Data Management
4.7 Close Project	
KA 5. Project Scope Management	
5.1 Scope Planning	Technical Planning
5.2 Scope Definition	Technical Planning
5.3 Create WBS	Technical Planning
5.4 Scope Verification	Technical Assessment
5.5 Scope Control	Decision Analysis Technical Assessment
KA 6. Project Time Management	
6.1 Activity Definition	Technical Planning
6.2 Activity Sequencing	Technical Planning
6.3 Activity Resource Estimating	Technical Planning
6.4 Activity Duration Estimating	Technical Planning
6.5 Schedule Control	Technical Planning
6.6 Schedule Control	Technical Assessment
KA 7. Project Cost Management	
7.1 Cost Estimating	Technical Planning
7.2 Cost Budgeting	Technical Planning
7.3 Cost Control	Technical Planning
KA 8. Project Quality Management	
8.1 Quality Planning	Technical Planning
8.2 Perform Quality Assurance	* Quality Consideration associated with the product, under <i>Systems Engineering Design Considerations</i>
8.3 Perform Quality Control	* Quality Consideration associated with the product, under <i>Systems Engineering Design Considerations</i>
KA 9. Project Human Resource Management	
9.1 Human Resource Planning	Technical Planning
9.2 Acquire Project Team	
9.3 Develop Project Team	
9.4 Manage Project Team	

Table 1-7 Mapping PMBOK Processes onto DAG Processes (continued)

PM	SE
PMBOK Processes [14]	DAG Technical Management Processes [4]
KA 10. Project Communications Management	
10.1 Communications Planning	Technical Data Management
10.2 Information Distribution	Technical Data Management
10.3 Performance Reporting	Technical Data Management
10.4 Manage Stakeholders	
KA 11. Project Risk Management	
11.1 Risk Management Planning	Technical Planning Risk Management
11.2 Risk Identification	Risk Management
11.3 Qualitative Risk Analysis	Risk Management
11.4 Quantitative Risk Analysis	Risk Management
11.5 Risk Response Planning	Technical Planning Risk Management
11.6 Risk Monitoring and Control	Risk Management
KA 12. Project Procurement Management	
12.1 Plan Purchases and Acquisitions	Technical Planning
12.2 Plan Contracting	Technical Planning
12.3 Request Seller Responses	
12.4 Select Sellers	
12.5 Contract Administration	
12.6 Contract Closure	

1.3.9 The DoD Guide to IPPD

The DoD Guide to Integrated Product and Process Development (IPPD) [22] was developed to support acquisition processes, as identified in DoD Instruction 5000.2. It states that “Processes should be developed concurrently with the products they support”. It stresses the importance of keeping the product and process design and performance in balance in order to achieve lifecycle cost and effectiveness objectives. It suggests a disciplined approach that includes five general activities, which should be re-evaluated as the product matures through the project progress: understanding the requirements, outlining the approach, planning the effort, allocating resources, and executing and tracking the plan.

The DoD IPPD guide describes a generic IPPD iterative process that is referred to as “a systems engineering approach” which “differs from the long held view that systems engineering is essentially a partitioning, trade-off, control process that brings the ‘-ilities’ and test functions together... This approach also transforms the stated needs into a balanced set of product and process descriptions”. The guide lists tools, teams, and processes that should be used in a structured manner, but it does not provide a shared ontology or methodology for bridging the gap between the product domain and the process domain to enable a real tangible integration of both domains. The lack of such common ground is manifested when “decisions made using this [IPPD] approach should be re-evaluated as a system matures and circumstances (budgetary, threat, technology) change... [and] descriptions are incrementally matured during each acquisition phase”.

1.4 The Need for Integrating the Project with the Product

1.4.1 The Links between Project and the Product within SEM

The implementation of SEM requires collaboration of multidisciplinary teams, coordination of processes, methods and tools, allocation of resources, and utilization of adequate facilities within enterprises [23], [24]. While the standards surveyed above discuss these issues, SE managers are left on their own when it comes to tailoring the standards, models, and best practices to their specific project and product needs and circumstances. A framework and a process for integrating perspectives of complex system development with its enterprise and project processes are clearly missing [24]. Furthermore, empirical investigations have shown that the relationships and interactions between the architecture of systems, their development projects, and the organizational teams involved, should be aligned in order for a company to become successful [25].

To demonstrate the tight links between the product and the project, consider the integration activities of a complex system in the aerospace area. What needs to be developed, tested, and delivered is determined by the **product** requirements, its architecture, and design. When each component should and can be developed and tested is stated in the **project** plan. Usually, the integration starts with initial integrations which focus on verifying the performance of individual software and hardware components, followed by integration of large subassemblies, proceeding to sub-systems integrations, and continues with the longitudinal process of the entire operational system integration. Figure 1-5 Integration process for an imaginary avionics system presents a realistic, although simplified, integration process for an imaginary avionics system. It contains three sites: (1) software lab site - in which Computer Software Configurable Items (CSCIs) are integrated and tested, (2) system lab site - in which integration and testing of combined configurations of CSCIs with Hardware Configurable Items (HWCI) is conducted, and (3) aerial platform site - in which the entire operational system is integrated and tested, either on ground or in testing and verification flights.

The small circles denote events while the arrows indicate the flow of the plan. The circled numbers denote elapsed time in months from project start. Following the events according to the arrows, together with the given time, reveals the integration plan logic. There are five CSCIs and three HWCI to be integrated in the system. Five software versions are planned to be integrated and tested at the software lab site, defined by software version index i - SW Vi. Each software version is a configuration composed of defined CSCIs according to the integration plan logic. The first integration event planned at the software lab site is the integration of CSCI 1 with CSCI 2 in a SW V0 configuration. Then, this configuration is combined with HWCI 3 into the first planned system version - Sys V0 - to be tested at the first integration event planned at the system lab site, at 12 month time after project start.

CSCI 3 is planned to be tested together with CSCI 1 and CSIS 2 - after the last two have been tested together - at the second event in the software lab site, defined as SW V1 integration version. This tripled CSCIs configuration is combined with HWCI 3 to be tested in the second integration event planned at the system lab site, at 18 month time after project start, 6 months after the predecessor integration event has been conducted in the system lab site. An integration plan contains both the What and the When, in a combined logic.

CSC5 5 is tested alone in two integration events, prior to its integration with CSCI 4 I in the third planned integration event at the system lab site, together with CSCI 1,

CSCI2, CSCI 3, HWCI 2, and HWCI 3, which have been tested earlier in the system lab site at Sys V1. The basic rationale is that Software Versions (SW Vi) are tested prior to System Versions (Sys Vi) which are combined CSCIs-HWCIs configurations.

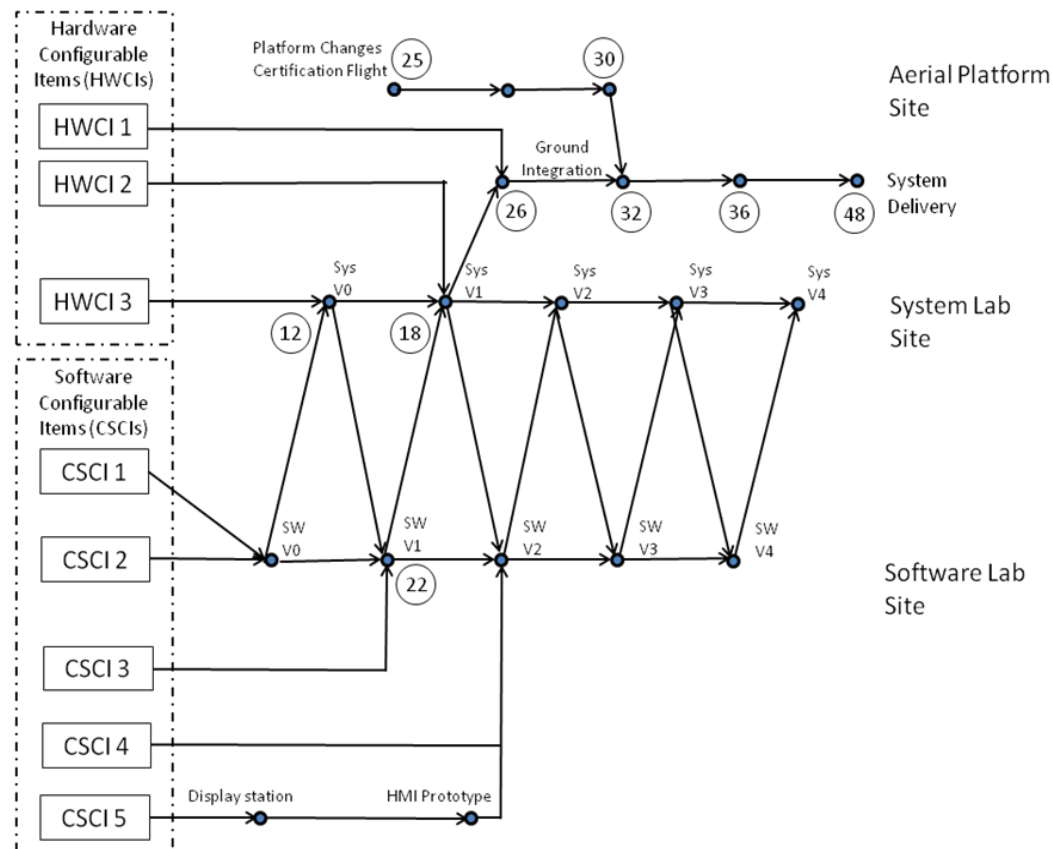


Figure 1-5 Integration process for an imaginary avionics system

The plan is dynamically re-estimated, re-evaluated, and re-planned, depending on parameters such as the project progress compared with the plan, resource availability, risks, and technological breakthroughs. Both the *What* and the *When* are the systems engineering manager's responsibility. Systems engineering managers must therefore combine technical skills and management principles.

Commonly accepted approaches to PM distinguish between the project processes and the product delivered by the project [23]. At the same time, some initiatives advocate developing project processes to concurrently support the produced products. However, actual techniques for joining a process and the product it delivers are not described by existing approaches. Even when product processes are mentioned, these are separated from the processes of the project management, so for the time being, SEM makes use of traditional PM tools.

1.4.2 The impact of systems engineering on project parameters

The impact of systems engineering on program cost was recognized over a decade ago, when it has been established that approximately 80% to 90% of the development cost of a large system is predetermined by the time only 5% to 10% of the

development effort has been completed. By the time system-level design is complete, 85% of the costs have been committed, and the cost to extract and rectify defects goes up exponentially [26]. This is one example of relationships between the project and the product, suggesting that careful management of these relationships is perhaps the most critical success factor for SEM to enable an enterprise to deliver and support a system. Failure to closely manage the intricate web of resource constraints emanating from both domains for meeting a development and test objectives is a recipe for inadequate performance and project time and cost overruns.

Large-scale projects almost invariably fail to execute exactly as planned, primarily due to uncertainties. According to the assessments of selected major weapon programs report to congressional committees [27], the DoD's weapon programs portfolio often experiences a reduced return on Investment. The study was undertaken to improve internal organizational acquisition processes and focused on large-scale federal programs that included tanks, aircraft, satellites, missiles, and information systems. Many of the programs in the 2006 assessment cost more and took longer to develop than estimated. The total RDT&E costs for 26 common set weapon programs increased by nearly \$44.6 billion, or 37 percent, over the original business case (the first full estimate). The same programs have also experienced an increase in the time needed to develop capabilities with a weighted-average schedule increase of nearly 17 percent. Some examples for differences between the initial estimation and the updated estimation are presented in Figure 1-6. Eight common causes of cost and schedule growth for large-scale systems have been identified: (1) overzealous advocacy, (2) immature technology, (3) lack of corporate roadmaps, (4) requirements instability, (5) ineffective acquisition strategy and contractual practices, (6) unrealistic program baselines, (7) inadequate systems engineering, and (8) inexperienced workforce and high turnover.

The cause of unrealistic program baselines is the failure to adequately conduct early studies, trades, and analysis that leads to an inaccurate cost, schedule, and performance program baseline. It is the inability to generate a credible program baseline. The study comments that "unrealistic cost and schedule expectations during proposal result in catastrophic consequences". Unrealistic program baselines inevitably lead to cost and schedule delays.


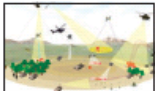




Program		Initial estimate	Initial quantity	Latest estimate	Latest quantity	Percent of unit cost increase
Joint Strike Fighter		\$189.8 billion	2,866 aircraft	\$206.3 billion	2,458 aircraft	26.7
Future Combat Systems		\$82.6 billion	15 systems	\$127.5 billion	15 systems	54.4
F-22A Raptor		\$81.1 billion	648 aircraft	\$65.4 billion	181 aircraft	188.7
Evolved Expendable Launch Vehicle		\$15.4 billion	181 vehicles	\$28.0 billion	138 vehicles	137.8
Space Based Infrared System High		\$4.1 billion	5 satellites	\$10.2 billion	3 satellites	315.4
Expeditionary Fighting Vehicle		\$8.1 billion	1,025 vehicles	\$11.1 billion	1,025 vehicles	35.9

Figure 1-6 Examples of DoD programs with reduced buying power

One of the recommendations in the study is to establish the program baseline prior to releasing the RFP and include, at a minimum, an Integrated Master Schedule (IMS) [28], which specifies networked, detailed tasks necessary to ensure successful program execution. The IMS is to be used to integrate the program schedule activities with all related components while including critical items from suppliers, teammates, or other detailed schedules that depict critical elements or information dependencies for the entire contractual effort in a single integrated network. The IMS Data Item Description (DID) contains the format and content preparation instructions for the IMS, but it offers no model or methodology. The time-based IMS flows directly from the Integrated Master Plan (IPM) which is the event-based top level plan of a program.

Another recommendation regarding the unrealistic program baselines cause is to implement rigorous trade studies of cost and schedule versus system impacts prior to Acceptance Tests Plan (ATP). In order to accomplish this capability, it is necessary to identify and define relationships between relevant project entities and corresponding product entities. However, each such project-product pair has traditionally been treated using different approaches, services, and tools, making it difficult to integrate the project and the product for gaining the obvious potential benefits for all the stakeholders involved.

The product design is done using a suite of CAD/CAM/CAE/PDM-type systems. Software design is usually done using some CASE tools. The project design and management is done via specialized project management software tools. Finally, the enterprise as a whole employs an ERP system. At later product lifecycle stages, when the product configuration is frozen and before it enters its manufacturing phase, ERP-type services for managing the project and product logistics, such as Master Production Scheduling, Bill of Materials, and MRP II, are used.

1.4.3 The Gap between SEM and PM

Systems engineering planning and control are the essentials of systems engineering management (SEM). While the project management (PM) is concerned with the entire work to be performed in the project, SEM is concerned with the required SE efforts to be performed. In order for a project to become successful, project managers and systems engineers must bridge the gap between management and engineering. This gap has been discussed by Adgar Schein [29] who identified managers and engineers as forming two distinct, separate organizational sub-cultures, each acting according to own specific goals, which may not always be one in alignment with the other. Nevertheless, both managers and engineers are equally required in order for the project-product ensemble to succeed. This further underlines the need for a common language and model as a basis for synergy between managers and engineers.

2 Motivation Questions, and Framework

The observations presented in Chapter 1 call for a fresh look at the methods and tools, used by systems engineering managers in an attempt to bring the product issues and the project issues together under the same conceptual model of a combined project-product system ensemble. This is the premise of the Project-Product Lifecycle Management (PPLM) research presented in this thesis [30], [31], aimed at developing a conceptual model and approach for fusing the *product* to be developed with the *project*, using software tool environment. The approach facilitates associating concepts from both domains, such that the resulting comprehensive model and supporting software will enhance systems engineering management within large-scale projects aimed at developing complex systems. It can serve as an actual bridge between the project management and systems engineering, within enterprises, potentially yielding significant cuts in time-to-market, project risks, and product malfunctions.

2.1 Research Motivation

IJEME [32] referred to the change of scope and complexity of engineering responsibilities during the past 20 years, seeking better management of technical functions within companies, and provided a phrase that presents the originating motivation for this research in the larger context: “... *in the last decade, many companies have reduced the numbers and levels of management positions ...as a consequence, engineers perform many management activities in modern companies. However, engineering management, which aims at bridging the gap between engineering and management, differs from ordinary management, since it requires the ability to apply engineering principles and the skills for organising and directing people and projects.*”

The fundamental premise underlying this thesis is that the ability to simultaneously express the required information from both the managerial project domain and the engineering product domain within a single integrated model-based framework is achievable. The vision that motivated this research is the development and assessment of a complete model-based methodology for joint project-product lifecycle management, yielding higher-quality management of both the system engineering of the product and the project or program delivering it. The integrated framework has the potential of supporting the needs of both practicing systems engineers and projects managers by leading to a more reliable planning process of the technical parts of the project plans,

which in turn will be less prone to the need for repeated changes and corrective actions.

The research goal is to develop a conceptual model-based plan for managing the lifecycle of the product to be developed hand-in-hand with the lifecycle of the project within the scope of which the product is developed. The underlying holistic conceptual model will potentially serve as a solid basis for supporting SEM, providing an integrated project and product lifecycle support environment. This integrated model will simultaneously express the function, structure and behavior of the two domains—the project and the product—via the same ontological foundations, making it possible to directly and precisely link entities in one domain to those in the other, thereby attaining a holistic view of the highly complex and intricate project-product ensemble.

2.2 The Research Question

Table 2-1 presents the research questions (third column) as gradually developed as the research evolved. The first step of the research was to try to establish the existence of project dimension and product dimension, as perceived by SEM practitioners while conducting SEM. Despite the differences between PM and SEM, the approaches and methods for project planning used by systems engineers are basically the same as those used by project managers. Therefore the first question was combined with the second question regarding the extent to which systems engineers perceive seven most common PM methods as being supportive of SEM. The research conducted to answer these two questions is described in Chapter 4.


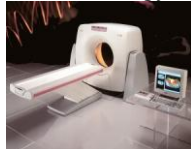
The next research question was how systems engineers who are SEM practitioners perceive systems engineering methods and tools for carrying out SEM applications. The research conducted to answer this research question is described in Chapter 6.

Following the first three research questions, the most common schedule-based planning tool, the Gantt chart, was compared with the PPLM model-based planning approach, as described in Chapter 8. The fourth research question was "How are the differences between a Gantt model plan and a PPLM model plan reflected by systems engineers?" This part of the research was conducted in three stages: (1) Gantt-PPLM comparison among the first participants group, based on a simplified Unmanned Vehicle (UAV) model, whose members combined the project and the product without specific guidelines, (2) same as (1), but among a second group and an improved test following lessons learned from (1), and (3) an elaborated Gantt-PPLM comparison among the second group, based on a simplified CT scanner model, whose members combined the project and the product according to the PPLM methodology.

2.3 Research Framework

The Product Lifecycle is distinguished from the Project Lifecycle. "Project" is perceived as a term in the managerial domain, usually encompassing schedule, milestones, money and other resources, risks, and deliverables. Conversely, "Product" is perceived as a term in the engineering domain and its development is regarded as belonging to the field of systems engineering. As such, it is concerned primarily with capture and analysis of requirements, design of the high-level system architecture, allocation of requirements into system's electronic, mechanical, software, and other components, and eventually with the integration, evaluation and validation of the system.

Table 2-1 Summary of Research Objectives and Framework

#	Topic	Research Question	Research Method	Notes	Thesis Chapter
1 + 2	Application of Project Management Methods within Systems Engineering Management	(1) While conducting the systems engineering management, to what extent do practitioners perceive a notion of a project-domain, a product-domain, and a combined project-product domain? (2) How do systems engineers perceive the extent to which the common PM methods support SEM?	Structured questionnaires	Focus on seven PM methods as viewed by 24 participants	Chapter 4
3	Perceived characteristics of SE tools and methods	(3) How do systems engineers perceive systems engineering methods and tools?	Qualitative research – interviews, inspections and analysis of artifacts	focus is put on ten systems engineers in a large enterprise	Chapter 6
4	Perceived PPLM model versus Gantt model	(4) How are the differences between a Gantt model plan and a PPLM model plan reflected by systems engineers?	<p><u>Stage (1):</u> Structured questionnaires: – UAV model – Project and product combined in a <u>free manner</u></p>  <p><u>Stage (2):</u> Improved structured questionnaires</p> <p><u>Stage (3):</u> Structured questionnaires – CT scanner model – Project and product combined according to PPLM methodology</p> 	<p>focus is put on Gantt as the most common planning model</p> <p>24 participants</p> <p>↓ Lessons learned</p> <p>32 participants</p> <p>32 participants</p>	Chapter 8

2.3.1 Project-Product Lifecycle and Conceptual Design

A system lifecycle is generally defined as the course of developmental changes through which a system passes from its conception to the termination of its use and subsequent salvage. The system development life cycle (SDLC) is a conceptual model used in project management that describes the stages involved in a system development project, from an initial feasibility study through maintenance of the completed application.

Various SDLC methodologies have been developed to guide the processes involved, including the waterfall model, which was the original SDLC method, joint application development (JAD), the fountain model, the spiral model, rapid application development (RAD), agile development and its Extreme Programming, XP, specialization, build-and-fix, and synchronize-and-stabilize. Frequently, several models are combined into some sort of hybrid methodology.

SDLC is defined by the U.S. Department of Justice (DoJ) as a software development process, although it is also a distinct process independent of software or other Information Technology considerations. SDLC is used by engineers to develop and support a system, including its requirements, validation, training, and user ownership through investigation, analysis, design, implementation, maintenance, and termination.

Documentation is crucial regardless of the type of model chosen or devised for any application, and is usually done in parallel with the development process. Some methods work better for specific types of projects, but in the final analysis, the most important factor for the success of a project may be how closely the particular plan was followed.

SDLC is a systematic approach to problem solving. It is composed of several phases, each comprising multiple steps. A typical SDLC, as practiced in relatively mature software-development organizations, has (at least) the following activities:

- identification of needs and constraints
- elicitation and collection of requirements
- architecture design
- detailed design
- implementation
- testing
- deployment
- maintenance

This list is not exhaustive, and many of these activities can be broken down into sub-activities (for example, most of these activities include a documentation sub-activity and an analysis sub-activity). Also, this list is not to be taken as implying a particular development process, be it spiral, waterfall, agile, or any other. The idea is that these are distinct activities, with their own inputs, outputs, specialists, activities, analysis techniques, and notations that need to be undertaken in the development of any substantial software-intensive project.

The system engineering process models spanning over the product development lifecycle include MIL-STD-499B, EIA IS 632, CMMI, IEEE 1220, and ISO/IEC 15288. An example of program phases, as recommended for the United States Department of Defence (US DoD) programs in MIL 499B [4] is summarized in Figure 2-1. It lists 22 key program tasks, which are conducted during a typical program lifecycle. These tasks provide a broad perspective on the Systems

Engineering process from "Need" to "Disposal". The program phases for commercial firms generally cover the same spectrum of activities, but with different definitions. As an example, a car manufacturer may conduct “man-in-the-street” surveys to collect desirable features for their next generation product. On the other hand, both a military and commercial aircraft company might use similar program phases.

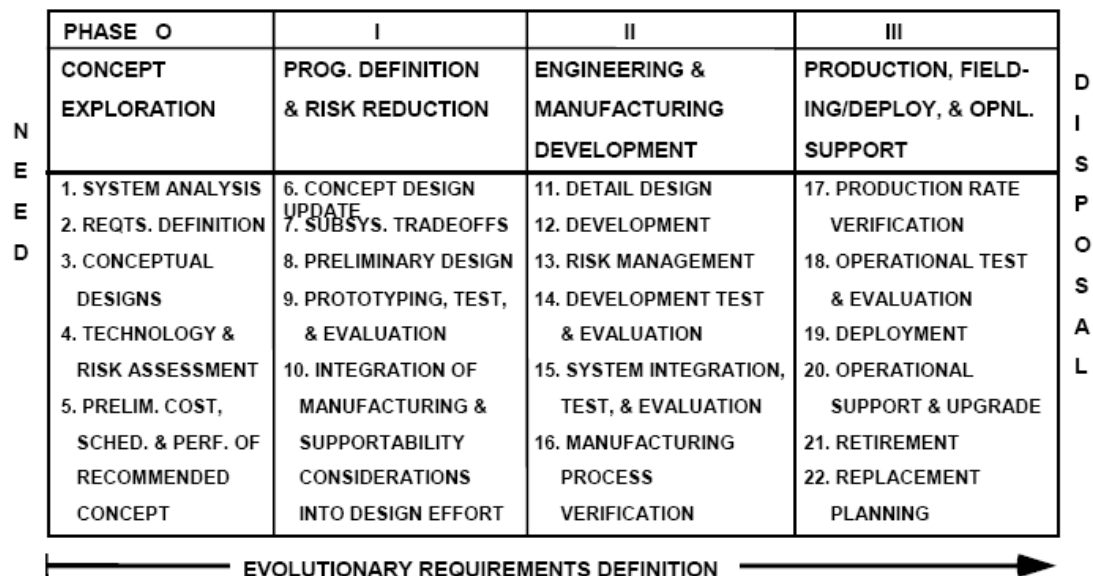


Figure 2-1 US DoD Program Lifecycle

The IEEE 1220 Standard [5] specifies the requirements for the systems engineering process (SEP) and its application throughout the product lifecycle, as presented in Table 2-2.

The definition of system lifecycle phases is only slightly influenced by differences among enterprises, as can be seen from the comparison of system development phases for five different organizations, shown in Figure 2-2. The phases of all five organizations are fundamentally similar in that they move from requirements definition through deployment, operations and support, to deactivation; but they differ in the vocabulary used and nuances within the sequential process.

2 Table -2 IEEE 1220 Standard System Life Cycle

Stage		Required Work Products and Tasks
System Definition		Concept, plans, interfaces, risks, quality factors, specs, baselines, reviews
Subsystem Definition	Preliminary Design	Subsystem definition, plans, interfaces, risks, quality factors, specs, baselines, reviews
	Detailed Design	Component definition, plans, interfaces, risks, quality factors, specs, baselines, reviews
	Fabrication Assembly Integration and Test	System integration and test; Analyze fix and retest; Plans, specs, baselines, reviews and audits
Production – Customer Support		Correct deficiencies; Ensure proper waste handling; Apply SEP on fielded products

Figure 2-3 presents the depth and breadth of three major systems engineering related standards, over a defined lifecycle. ISO/IEC 15288 [16] describes processes that an organization is likely to establish in order to build complete and efficient life cycle models, covering all five phases of the defined lifecycle. EIA 632 – Processes for Engineering a System [6] is an integrated set of fundamental processes for engineering a system, covering three out of the five phases, and followed exactly by INCOSE. Finally, IEEE 1220 – Standard for Application and Management of the Systems Engineering Process [5] is a detailed description of the system development tasks and activities, covering the specific phase of development.

Unlike most other systems engineering management standards, the ISO/IEC 15288 also explicitly accounts for the enterprise processes that "manage the organization's capability to acquire and supply products or services through the initiation, support and control of projects" [16]. These enterprise processes provide resources and infrastructure necessary to support projects and ensure the satisfaction of organizational objectives and established agreements.

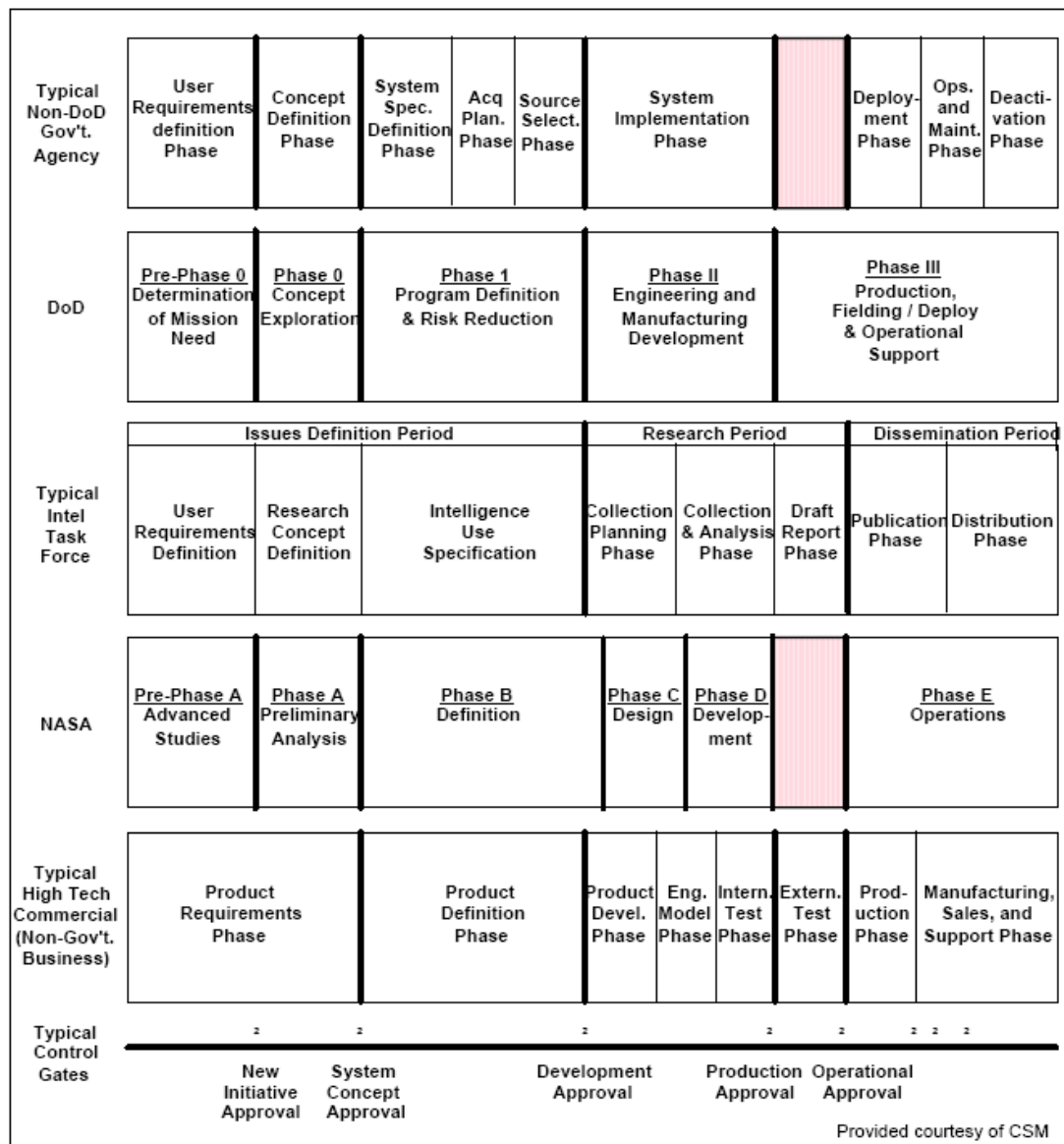


Figure 2-2 INCOSE Comparison of System Life Cycle phases

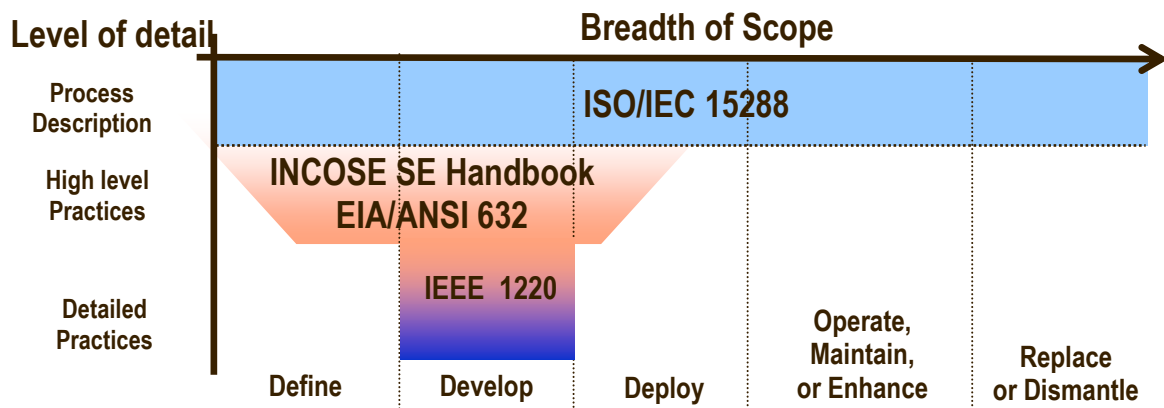


Figure 2-3 ISO/IEC 15288 Comparison of Breadth and Depth of Key Standard

The combined timeline of a typical Project *i*, which delivers Product *i* within a given Enterprise is illustrated in Figure 2-4, showing how the three timelines are intimately intertwined. The life span of the project is a subset of that of the corresponding product, while the enterprise lifespan is normally a superset of its products' life spans. The closure of Project *i* often coincides with the delivery of Product *i* to the prospective customer. The combined project-product-enterprise ensemble comprises tight links between them, which motivate their joint management. An even more complex ensemble is the one involving not just one but many interacting enterprises, each with its own set of project-product pairs. Moreover, a single sizeable project-product pair normally involves many enterprises, one of which leads the project while the rest are its subcontractors. Hence an enterprise can be a leader—a contractor—in a set of project-product pairs and a subcontractor in many others, creating an involved web of value creation (naively called the "value chain"), which Internet-based electronic commerce increasingly facilitates.

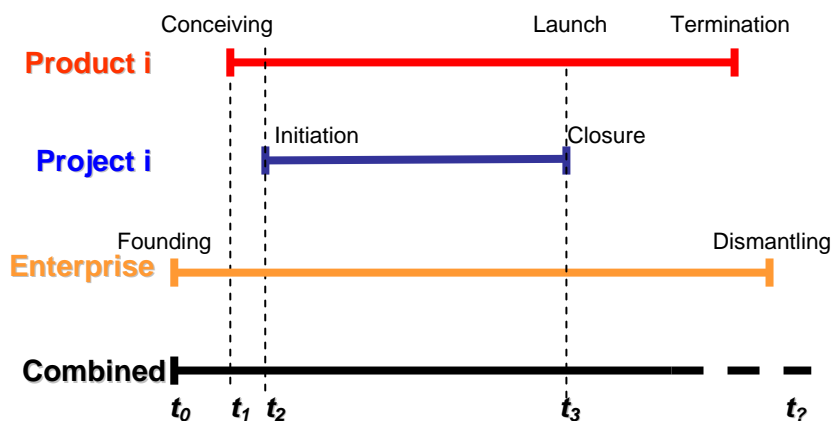


Figure 2-4 The combined timelines of Project *i*, which delivers Product *i* within the Enterprise

Conceptual design is the key initial phase in the project-product lifecycle. To put conceptual design in a lifecycle context, its main steps include needs identification, feasibility analysis, system requirement analysis, system specification, and conceptual design review. Creation of alternative conceptual models, each with its unique future

architecture, and feasibility evaluation of these models, enable selection of the adopted concept. The conceptual design review validates that all the identified product needs are addressed by the conceptual model. The output from the conceptual design is the functional baseline, which constitutes the *product specification*. Conceptual design is followed by preliminary design, which is comprised of functional analysis, architecting, namely high-level design of the system's structure-behavior combination, requirements allocation, trade-off analysis, and synthesis of system options and their evaluation.

The output of the preliminary design phase is the product development specification, which serves as a basis for the next, detailed design phase. For each level of the product's decomposition, the project has to be decomposed and planned properly in order to best support and ensure the development and the integration of the product. Focusing on product-delivering projects, the first step in the planning process requires basic knowledge of the concept of the product that the project is expected to deliver. In systems engineering, the product is usually referred to as *system*, encompassing all its aspects, including performance, over its entire lifecycle span. Therefore, the initial project plan, scope of work, and allocation of resources rely on understanding of at least the top-level system's functionality, architecture, and concept of operation.

The project planning is an iterative process of derivation, refinement, and simulation of the product model, while maintaining traceability and coherence between the product model and the project plan at all levels through the hierarchical structure of both the product and the project, as illustrated in Figure 2-5 [31].

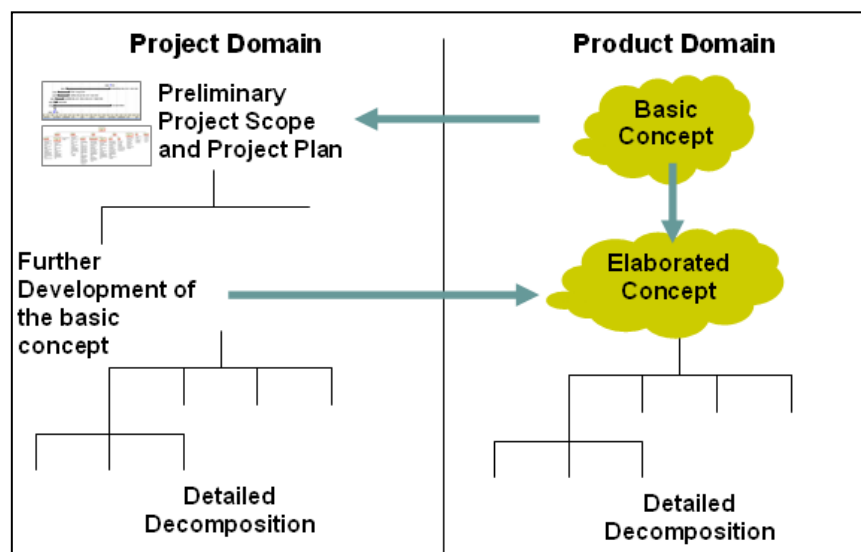


Figure 2-5 Building the project plan iteratively while creating the product model

Systems Engineering Management (SEM) is about handling and solving problems associated with the intricate relationships of the product with the project that delivers it. SE specifies both the product and the process for producing the product, while the overall responsibility for the product on-time and as-specified lies with the project management responsibility. Hence, SEM must be perfectly aligned with PM. The only way to achieve this is to treat SEM and PM as two facets of the same project-product ensemble.

SEM is an iterative process of planning, derivation, refinement, and simulation of both the product model and the management plan, while maintaining traceability and coherence between the product model and the project plan at all levels through the hierarchical structure of both the product and the project. SEM is practiced by continuous iterative zigzagging between the two domains – the systems engineering domain and the project management domain. This zigzagging entails both the cognitive processes the systems engineering manager undergoes, and the actual interactions and actions he or she conducts in order to carry out his or her duty. It starts with the conceptual design of the system, hand in hand with the preliminary management plan, and evolves as the project continues.

The need to concurrently address more than one domain has been identified by Suh [33], who developed the Axiomatic Design (AD) methodology. AD addresses four domains: the customer, functional, physical, and process domains. The output of each domain evolves gradually and hierarchically from an abstract concept into detailed information. The hierarchical decomposition in one domain cannot be performed independently of the other domains, i.e., decomposition requires zigzagging in order to map adjacent domains to each other. The planning process involves the continuous processing of information within and between the four domains. The customer needs are identified in the customer domain and are stated in the form of required functionality of a product in the functional domain. Design parameters that satisfy the functional requirements are defined in the physical domain, and manufacturing variables are defined in the process domain in order to determine how the product is to be produced. The AD physical domain is somewhat analogous to our product domain, while the AD process domain contains elements of our project domain. However, all the four AD domains focus to the product. In our context of SEM and PM, the mental zigzagging between domains entails switching back and forth between the product domain and the project domain.

The integrated Project-Product Lifecycle Management approach and model presented in this work induces traceability, explicating critical relationships between the product and the project. The ability to simultaneously express the required information from both the project and the product domains within a single integrated model-based framework can potentially lead to a more reliable project plan, which is less prone to the need for repeated changes, rework, and corrective actions. The increased robustness of the resulting project plan is attributed to the need to consult the product model integrated into the combined Project-product model while making decisions about the project's "technological order" [34]. The planning process carried out following this approach clarifies the complex relationships between the project and product entities.

In large-scale projects of complex systems and systems-of-systems, the project and the system are decomposed hierarchically, maintaining links between each subsystem (or system in a system-of-systems scenario) and the corresponding project, yielding intricate relationships between the product and the project. For defence and aerospace development projects, the situation is exacerbated by requiring correspondence to baselines, documents, and reviews, along the project-product evolution [35]. Running a project of such magnitude is a huge effort that must be planned and controlled as part of the lead systems engineer's responsibility within the project management endeavor. Creating and executing a project plan that ensures the delivery of a fully functioning system within schedule and budget constraints is the ultimate challenge of both the systems engineering manager and the project manager. What both of them are missing is a common underlying ontology, a conceptual model, and a supporting

software environment to enable the required communication between both domains – the project and the product. Attaining these missing elements will enable the simultaneous expression of the function, structure and behavior of the project and the product. This thesis presents a model-based approach for managing the lifecycle of the product to be developed hand-in-hand with the lifecycle of the project, within the scope of which the product is developed. The cornerstone of this Project-Product Lifecycle Management (PPLM) approach is an underlying holistic conceptual model, supported by software capabilities for an integrated project and product lifecycle environment. The concurrent project-product model, built on common ontological foundations, enables better management, making it possible to directly link entities in one subsystem to those in the other. The expected value of the holistic, integrated conceptual model is the provision of both superior product lifecycle engineering and project management capabilities, yielding significant cut in time to market, reduced risk, and higher product quality.

2.3.2 Choosing the Underlying Language and Methodology

Since a major challenge of integrating the project with the project is to unify data from both domains through a systematic model-based approach, the language and methodology for constructing the project-product model has to be both intuitive and formal. Among the languages examined were UML [36], xUML [37], SysML [38], Modelica [39], and Object-Process Methodology, OPM [1]. The characteristics of the modelling notation we looked for included the ability to represent data in hierarchically organized diagrams, expressiveness of the notation for defining common PPLM conceptual metamodels, formalism, and clear semantics for execution-based model simulation. Since the PPLM potential users come from a wide range of disciplines, the notation should also be simple and intuitive.

UML and xUML were discarded as being software-oriented and not quite appropriate for modelling systems. In contrast, SysML was designed for modelling general systems. The notation modifies a subset of UML diagrams and adds two new diagrams: Requirements Diagram and Parametric Diagram. Like UML, the SysML notation is comprised of multiple diagram types that provide mainly aspect decomposition. Hierarchical breakdown is supported for a subset of the diagram types. The multiple view model of SysML makes it difficult to get a good grasp of the complete system, one of the major tenets of conceptual design. Moreover, unlike xUML, the SysML notation does not use formal activity specifications, and it allows free text in Parametric Diagram for equation descriptions. This informality makes it impossible to define SysML model-based execution without changing the notation significantly. Modelica and Simulink are most fit for the detailed design phase but are too technical for the conceptual modelling phase and for use by a wide user community.

OPM is a holistic, integrated approach to the design and development of systems in general and complex dynamic systems in particular. OPM comprises entities and links. The three entity types are objects, processes (both referred to as "things"), and states. Objects are things that exist and can be stateful (i.e., have states). Processes transform objects: they generate and consume objects, or affect stateful objects by changing their state. Objects and processes are of equal importance, as they complement each other in the single-model specification of the system. Links, which are the OPM elements that connect entities, are of two types: structural and procedural. OPM objects relate statically to each other via structural relations, graphically expressed as structural links. The four fundamental structural relations are

aggregation-participation, generalization-specialization, exhibition-characterization, and classification-instantiation. Objects can also be structurally related to each other by unidirectional or bidirectional tagged relations, similar to association links in UML class diagrams. Structural relations specify relations between any two objects. Due to the object-process symmetry, they can also specify relations between any two processes. Conversely, procedural links connect a process with an object or an object's state to specify the dynamics of the system. Procedural links include (1) transforming links: effect link, consumption link, result link, and the pair of input-output links, (2) enabling links: agent and instrument links, and (3) control links: event, condition, invocation, and time exception links.

An OPM model consists of a set of hierarchically organized Object-Process Diagrams (OPDs) that alleviate systems' complexity. Each OPD is obtained by in-zooming or unfolding of a thing (object or process) in its ancestor OPD. One or more new things (objects and/or processes) can be specified within a thing in an OPD that was refined from a higher-level OPD. Copies of an existing thing can be placed in any diagram, where some or all the details, such as object states or links to other things, which are unimportant in the context of the diagram, need not be shown. It is sufficient for some detail to appear once in some OPD for it to be true for the system in general even though it is not shown in any other OPD.

OPM was selected as the basis for PPLM due to its following features:

- OPM is a visual methodology that incorporates the static-structural and dynamic-procedural aspects of a system into a unifying model, which is presented in its entirety using a single diagram type. This is achieved by treating both objects and processes as equally important things (entities). By using a single model at varying levels of detail, clutter and incompatibilities are likely to be avoided even in highly complex systems.
- OPM is designed to express triggering events, guarding conditions, timing constraints, timing exceptions, and flow-of-control constructs. These features are the basic elements required for exceptional behavior design.
- OPM has proven to be an efficient methodology for modelling complex dynamic behaviors in general and temporal exceptions in particular. OPM was shown to be significantly better in specification quality, compared with OMT, UML's main predecessor (Peleg & Dori, 2000).
- Through its recursive seamless complexity management (scaling, or abstraction/refinement) mechanisms, OPM is highly appropriate for managing systems' complexities. There are three complexity management mechanisms in OPM: (1) unfolding/folding, which is used for refining/abstracting the structural hierarchy of a thing; (2) in-zooming/out-zooming, which exposes/hides the inner details of things within its frame; and (3) expressing/suppressing, which exposes/hides the states of an object. These complexity management mechanisms enable OPM to represent complex systems gradually.
- OPM consists of two semantically equivalent modalities of the same model: graphical and textual. A set of interrelated Object-Process-Diagrams (OPDs) constitute the graphical model, and a set of automatically-generated sentences in a subset of English constitute the Object-Process Language (OPL). In the graphical-visual model, each OPD consists of OPM elements depicted as graphic symbols, while the OPD syntax specifies the consistent and correct ways by which those elements can be managed. Since the corresponding textual model is generated in a subset of English, it is immediately understood by domain experts, who need not learn any special language nor decipher cryptic code.

OPM is currently in the process of becoming an ISO standard and a basis for ISO enterprise standards. When completed, this will enable accelerated dissemination of OPM as a basis for enterprise standards in general and for PPLM in particular.

2.4 Content of the Dissertation

Table 2-3 Table 2-3 lists a summary of the entire dissertation content according to the chapters order.

Table 2-3 Organization of the dissertation

Chapter	Synopsis
Chapter 1 Introduction and Literature Review (Theory)	Introduction to the problem, including a description of the state of affairs in SEM, leading to the need of integrating the project with the product. Review of relevant literature, including a critical discussion of existing standards and methods.
Chapter 2 Motivation Questions, and Framework	Description of research motivation and research questions, followed by arguments for choosing the underlying language and methodology, and concluding with the organization of the dissertation.
Chapter 3 The Combined Project-Product Model (Theory)	Description of the OPM-based combined project-product model, based on a simplified Unmanned Aerial Vehicle (UAV) project-product ensemble, construction of the UAV project-product PPLM model in OPM using OPCAT, facilitating the simulation capability for validation purpose.
Chapter 4 Application of Project Management Methods within Systems Engineering Management (Research)	Following a brief survey of relevant Systems Engineering (SE) and Project Management (PM) standards, focus is placed on seven PM methods used to examine the first two research questions: (1) While conducting the systems engineering management, to what extent do practitioners perceive a notion of a project-domain, a product-domain, and a combined project-product domain? (2) How do systems engineers perceive the extent to which the common PM methods support SEM?
Chapter 5 Extraction of Coherent Project Views from the PPPLM Model-Based Plan (Theory)	Since all the entities and their relations are represented in the PPLM model, automatic procedures can be devised to generate other project views, including Activities Network Plan, Critical Path, Gantt chart, Work Breakdown Structure, and Design Structure Matrix This chapter presents the potential for automatic extraction of these PM representations from the OPM-based PPLM model and provides new and useful model-based representations.
Chapter 6 Perceived Characteristics of SE Tools and Methods (Research)	The characteristics of systems engineering management tools and methods among systems engineers were explored using qualitative research approach in order to answer the following research question: (3) How do systems engineers perceive systems engineering methods and tools?
Chapter 7 The Combined Project-Product Methodology (Theory)	Presentation of PPLM, as a complete lifecycle approach that unitizes the concepts of generic project construct and system builds is presented and demonstrated on a CT-Scanner model.
Chapter 8 Perceived Differences between the Gantt and the PPLM OPM-based model (Research)	Based on conclusions from answers to the first two research questions, a comparison of project planning is made between Gantt chart and the PPLM methodology. The objective here is to answer the fourth research question: (4) How are the differences between a Gantt model plan and a PPLM model plan reflected by systems engineers?
Chapter 9 Error! Reference source not found. (Theory)	Summary of the research, its findings, conclusions, significance, and possible future work.

3 The Combined Project-Product Model

The first phase of planning a project includes identifying the initial project plan, addressing the project's major activities, and determining their order of execution, often referred to as the "technological order" [34]. This chapter presents the PPLM model-based approach for creating the initial project plan, yielding a combined generic project-product model which integrates the product with the project, which does not yet follow the lifecycle methodology guidelines presented in Chapter 7. Using this approach, the model ultimately contains the project activities and the artifacts deemed as required by the project planner. Activities or tasks are OPM processes at various detail levels required for completing the product. Artifacts are objects—product components, resources, and informatical objects, mostly model-based documents. Consistently incorporating this information in the integrated PPLM model eventually yields all the structural relations (among objects) and procedural relations (between objects and processes).

Understanding the product objects' hierarchy hand-in-hand with the project tasks and progress provides the basis for the technological process order and timing. With this understanding, the project planner can model the rationale for ordering the project tasks—the model processes—by identifying the flow of artifacts into and out of each process. Each one of these artifacts results explicitly from a specific process and is used in a subsequent process, either as an instrument (if it is an informatical object), or as a consume (a physical object that is consumed by or embedded into a larger component).

As a case in point, we consider a project of developing a simplified Unmanned Aerial Vehicle (UAV), by an imaginary *New Millennium Aerospace (NMA) Inc.* The OPM-based combined project-product plan was modeled using OPCAT [40], facilitating the simulation capability as a supporting aid for constructing and validating the UAV PPLM plan. The top-level view of the UAV PPLM model is presented in Figure 3-1. The equivalent, automatically generated natural language text, called Object Process Language (OPL) paragraph, is listed in Figure 3-2. The top level diagram contains a single process, as the OPM methodology requires, which represents the combined UAV project-product ensemble. The entire combined ensemble is aimed at delivering the main output - the **Surveillance Data**, yielded by joint project-product ensemble of **UAV Developing & Surveillance Gathering**, while consuming a specified **Budget**. There are seven agents required for the **UAV Developing & Surveillance Gathering** combined ensemble to be executed: **Customer**, **User**, **UAV Pilot**, **Payload Operator**, **Mission Planner**, **Technical Team**, and **NMA Agents set** which is the set of human resources within *NMA Inc.* The **User** is typically some high command and control entity (such as the air force, navy or other army), which makes use of the **Surveillance Data** to its utility. Two instruments are modelled as required at this top level view: The **Requirements Set** and the **NMA Instruments Set which is the set of non-human resources within NMA Inc.**

Figure 3-3 presents the second level diagram of the UAV PPLM model. The equivalent, automatically generated natural language text is listed in Figure 3-4. This level is modeled through hierarchical decomposition, maintaining traceability through the hierarchy levels, using OPM refinement mechanisms, primarily in-zooming. Two processes are modelled at this level, denoted by the two ellipses: **UAV Prototype Developing & Integrating**, which is the entire project execution process, and **Surveillance** process, which is the UAV's product primary mission, the function, or top-level process, that the project is expected to deliver.

The product—the **UAV Prototype**—is the output of the project and is required as an instrument for the **Surveillance** process. The **Surveillance** process requires five agents out of the seven agents defined at the upper level in order to be executed: **User**, **UAV Pilot**, **Payload Operator**, **Mission Planner**, and **Technical Team**. The **Surveillance Data** is the product's main output, as yielded by the **Surveillance** process. The project requires the **NMA Agents set** (human resources), the **NMA Instruments Set** (non-human resources), and the **Budget** (a consumed input or consumee). The **NMA Agents set** is composed of two teams in this view: a **Developing Team** and an **Integration Team**, which are discussed in the sequel.

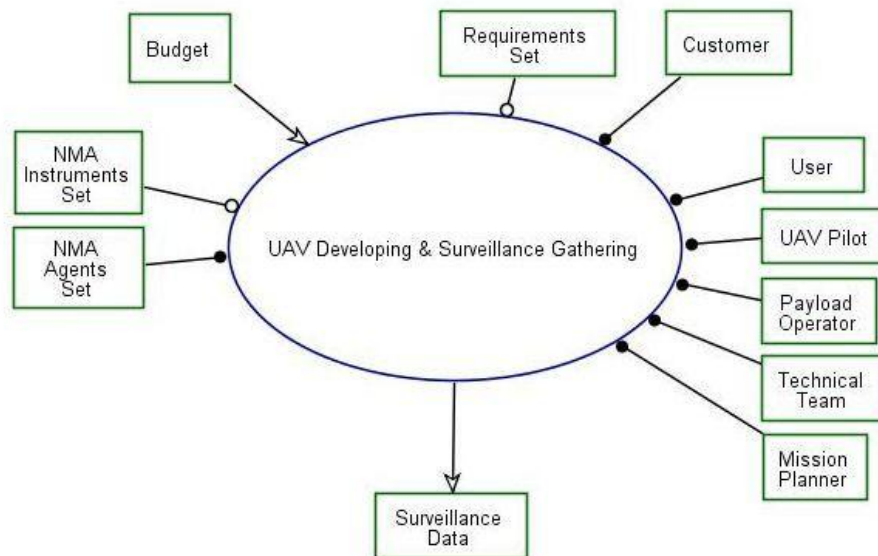


Figure 3-1 The top-level view of the UAV PPLM model

UAV Developing & Surveillance Gathering requires **Requirements Set** and **NMA Instruments Set**.
UAV Developing & Surveillance Gathering consumes **Budget**.
NMA Agents Set handles **UAV Developing & Surveillance Gathering**.
UAV Developing & Surveillance Gathering yields **Surveillance Data**.
User handles **UAV Developing & Surveillance Gathering**.
UAV Pilot handles **UAV Developing & Surveillance Gathering**.
Payload Operator handles **UAV Developing & Surveillance Gathering**.
Technical Team handles **UAV Developing & Surveillance Gathering**.
Mission Planner handles **UAV Developing & Surveillance Gathering**.
Customer handles **UAV Developing & Surveillance Gathering**.

Figure 3-2 The automatically-generated OPL paragraph of the OPD in Figure 3-1

While there is no single "correct" OPM model, the model has to comply with the following requirements:

- The model shall follow guidelines of the methodology part of OPM, including:
 - Starting with one top-level process at the System Diagram (SD, top-level OPD) of the model. As a PPLM specialized guideline, the next level of detail (SD1) shall contain two processes, one representing the project execution and the other—the product's function (main mission).

- Decomposing the project/product hierarchically using OPM refinement mechanisms, primarily in-zooming.
- Following the top-to-bottom ordering of a sequence of sub-processes within an in-zoomed process.
- Maintaining the timeline of subprocesses inside an in-zoomed process, which is vertical, flowing from top to bottom.
- The model shall include all the OPM things (objects and processes) needed for a complete representation of the project and the product.

The project section of the model is constructed iteratively, while zigzagging between the project domain and the product domain, explicitly addressing the artifacts that are generated and required along the project execution process, together with the time aspects for these artifacts, as derived from the relevant process durations. The resulting project plan integrally incorporates the product to be designed, manufactured, delivered, used, serviced, and disposed of, with the project, including all its significant artifacts. The artifacts may include product components, documents related to derived requirements, approvals, simulations, analysis, specifications, and other types of reports. Each one of these artifacts results explicitly from a specific process—a project activity—and is used in a subsequent process, either as an instrument (if it is an informatival object), or as a consumee (a physical object that is consumed by or embedded into a larger component). The artifacts are classified into *Deliverables* or *Resources*, as listed in Table 3-1, providing examples of artifacts for each class.

Table 3-1 Classification of objects in the OPM project plan

Symbol		Description	Examples
Deliverables (Roles)	D	document – a recorded definition of anything related to the product or the project delivering it expressed via an informatival object	Requirements Document, , Design Document, Engineering Drawing, Testing Procedure Document
	G	gate for required approval	Approval of a key document or a physical artifact, often related to a milestone, such as a requirements document, a design document, a prototype
	C	component	System or product part: Engine, Payload, Software
Resources	Inputs	Budget	
		Consumables Set	Raw materials, Energy, Utilities, Un-saved data
	Enablers	A Agent – a human enabler	System Engineer, Software Engineer, Engine Designer, Test Engineer, Technician, Programmer, Team Leader
		I Instrument – a non-human enabler	Design Tool, CAD System, Telemetry Equipment, Laboratory

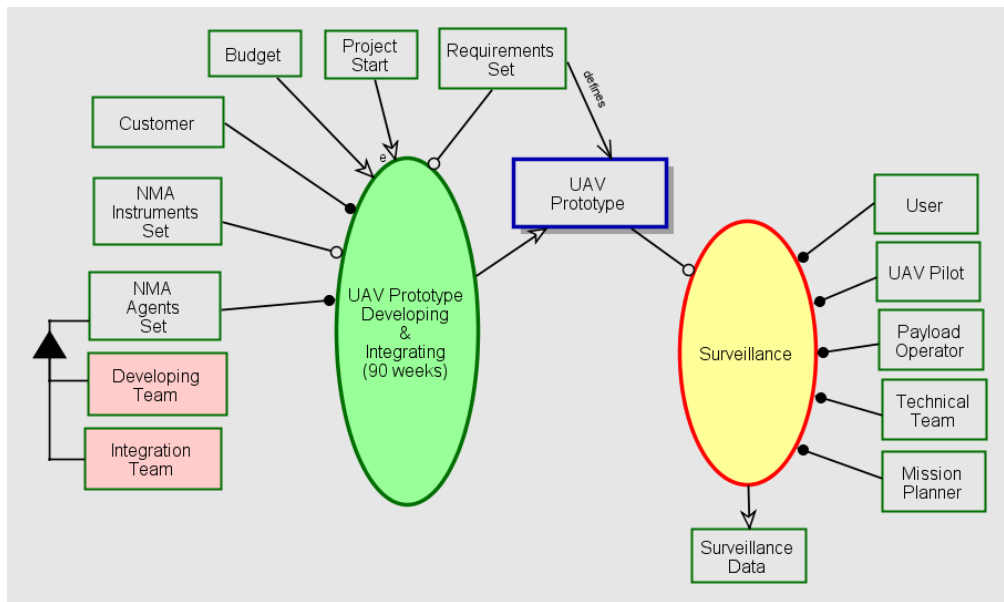


Figure 3-3 OPD of the second level System Diagram (SD1) of UAV PPLM model

Project Start triggers **UAV Prototype Developing & Integrating**.
NMA Agents Set handles **UAV Prototype Developing & Integrating**.
NMA Agents Set consists of **Developing Team** and **Integration Team**.
Customer handles **UAV Prototype Developing & Integrating**.
Developing & Integrating consumes **Project Start** and **Budget**.
UAV Prototype Developing & Integrating requires **NMA Instruments Set** and **Requirements Set**.
Requirements Set defines **UAV Prototype**.
UAV Prototype is physical.
UAV Prototype Developing & Integrating yields **UAV Prototype**.
Surveillance requires **UAV Prototype**.
Surveillance yields **Surveillance Data**.
UAV Pilot handles **Surveillance**.
User handles **Surveillance**.
Payload Operator handles **Surveillance**.
Technical Team handles **Surveillance**.
Mission Planner handles **Surveillance**.

Figure 3-4 The automatically-generated OPL paragraph of the OPD in Figure 3-3

The entire project is modeled through concurrent hierarchical decomposition of processes and objects involved in these processes as inputs, enablers, or deliverables. Using this approach, the model ultimately contains (1) the activities and tasks, which are processes at various detail levels (task is the most detailed, non-decomposable process), required for completing the product, (2) the artifacts (model-based deliverables) created during the project process, and (3) the different resources, including agents (humans), budget, and instruments (facilities and other inanimate resources). Embedding this information consistently in the same unifying model yields all the structural relations—relations between any two objects, and procedural relations—relations between any object and any process.

One advantage of this model-based project-product planning approach is its inherent support of the cognitive zigzagging between the project and product domains, similar to that in Axiomatic Design discussed earlier. At present, experienced professional project planners practice zigzagging using their skills and intuition. Developing and applying PPLM within the framework we propose, this planning process is bound to become more streamlined and standardized, and less dependent on exceptional individual skills. The project-product model enables the project planner to engage in

this zigzagging process in order to come up with a project plan that best addresses the product to be accomplished by the project without the need to leave one model in order to examine the other.

The ability to simultaneously express the required information from both the project and the product domains within a single integrated model-based framework can potentially lead to a more reliable project plan, which is less prone to the need for repeated changes and corrective actions. The increased robustness of the resulting project plan is attributed to the need to make decisions about the project's process order and logic while explicitly addressing the associated product model.

The planning process carried out following this approach clarifies the intricate relationships between the project and product entities. This model-based approach enables the simultaneous expression of the function, structure and behavior of both the project and the product via the same ontological and methodological foundations, maintaining full traceability between the project and product data.

3.1 Extended OPM notation for process sequence and duration

The time line in an OPM is vertical, flowing from top to bottom. The commonly used Finish-to-Start (FS) relationship between tasks A and B, such that B follows A, is expressed in an OPD by depicting the top of B's ellipse below the bottom of A's. OPM does not show process durations graphically, so an extended OPM notation presents process time duration graphically in the OPM project plan model. In this extended OPM project plan model, the height of the ellipse representing the process is roughly proportional to the process duration, which is assumed to be deterministic and is denoted in parentheses below its name. Horizontal dashed lines have been added to specify relationships of Start-to-Start (SS), Finish-to-Start (FS), Finish to Finish (FF), and Float Start-Finish, as shown in Table 3-2. In all four cases, the duration of process A is longer than that of process B, as indicated by the corresponding ellipse heights. In case a, both processes are required to start at the same time, while in case b they both must end at the same time. In case c, process B starts immediately after process A ends. In case d, process B can start at any point in time with or after A, as long as B ends before A ends.

Cases a and c are inherent in OPM, while for cases b and d a new mechanism is needed.

Table 3-2 New OPM constructs defining time notion

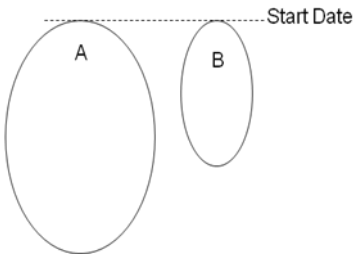
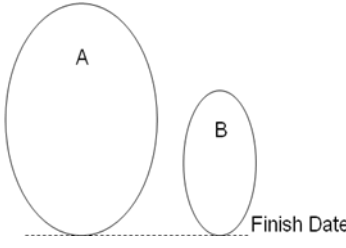
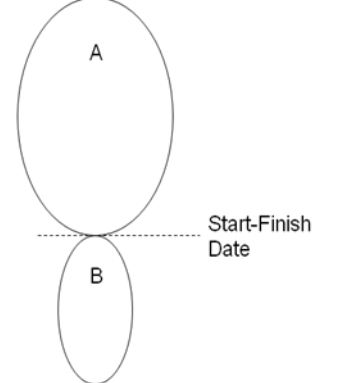
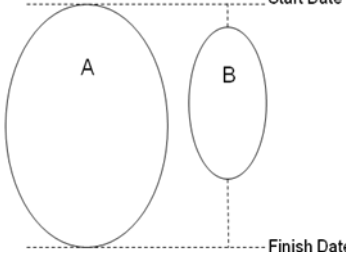
Case	Relationship	Extended OPM notation	Semantics
			OPL Syntax
a	Start-to-Start (SS)		<p>A and B must start at the same time.</p> <p>A and B must start at the same time.</p> <p>A, B, and C must start at the same time.</p>

Table 3-3 New OPM constructs defining time notion (continued)

Case	Relationship	Extended OPM notation	Semantics OPL Syntax
b	Finish-to-Finish (FF)		<p>A and B must finish at the same time.</p> <p>A and B finish concurrently.</p> <p>A, B, and C finish concurrently.</p>
c	Start-to-Finish (SF)		<p>B cannot start before A finishes.</p> <p>B immediately follows A.</p> <p>B and C immediately follow A.</p> <p>B, C, and D immediately follow A.</p>
d	Floating Start & Finish (FSF)		<p>B can start with A or at any time thereafter, as long as it finishes no later than when A finishes.</p> <p>The timing of B floats within the timing of A.</p>

3.2 The UAV Project Domain

3.2.1 Assigning duration to project processes

Leaf-level processes are assigned nominal duration, which is both their minimum and maximum duration according to the project plan. Higher level processes can be assigned maximum durations. In our case, the entire project duration has been assigned a maximum of 90 weeks.

The time duration can serve for validation using the animated simulation built into OPCAT. If a parent process is assigned an upper limit on its duration and invocation relationship is properly modeled, running the simulation will show whether executing its subprocesses exceeds its duration limit. Hazard objects generated by specific project processes if the maximum duration is exceeded can be included in the model.

A hazard object, connected via an event link that triggers a Project Control process, can contain the logic of responding to the event of such time exception.

3.2.2 Specifying resources

Figure 3-5 presents the second level view of the UAV PPLM presented in Figure 3-3, but this view contains more details regarding resources, including **Budget** and two teams: **Developing Team** and **Integration Team**. The allocation of resources in the model can be analyzed using the simulation capability in OPCAT for verification of the project plan. Each resource is utilized in accordance with the data specified in the model. As shown in Figure 3-5, **Budget** is measured in k\$ units and its upper limit is 1000.0. Running the simulation of the model in OPCAT will accumulate the consumed budget, as assigned to each project activity, yielding warnings in the lifespan view of the simulation when the budget exceeds its limit. **Developing Team** and **Integration Team** are measured in person units with a maximum exception rule. Running the simulation of the model in OPCAT will show the utilization of heads, enabling tracking the required amount of persons as a result of executing the project activities. For the sake of simplicity we will discard the resources in the following presentation of the model.

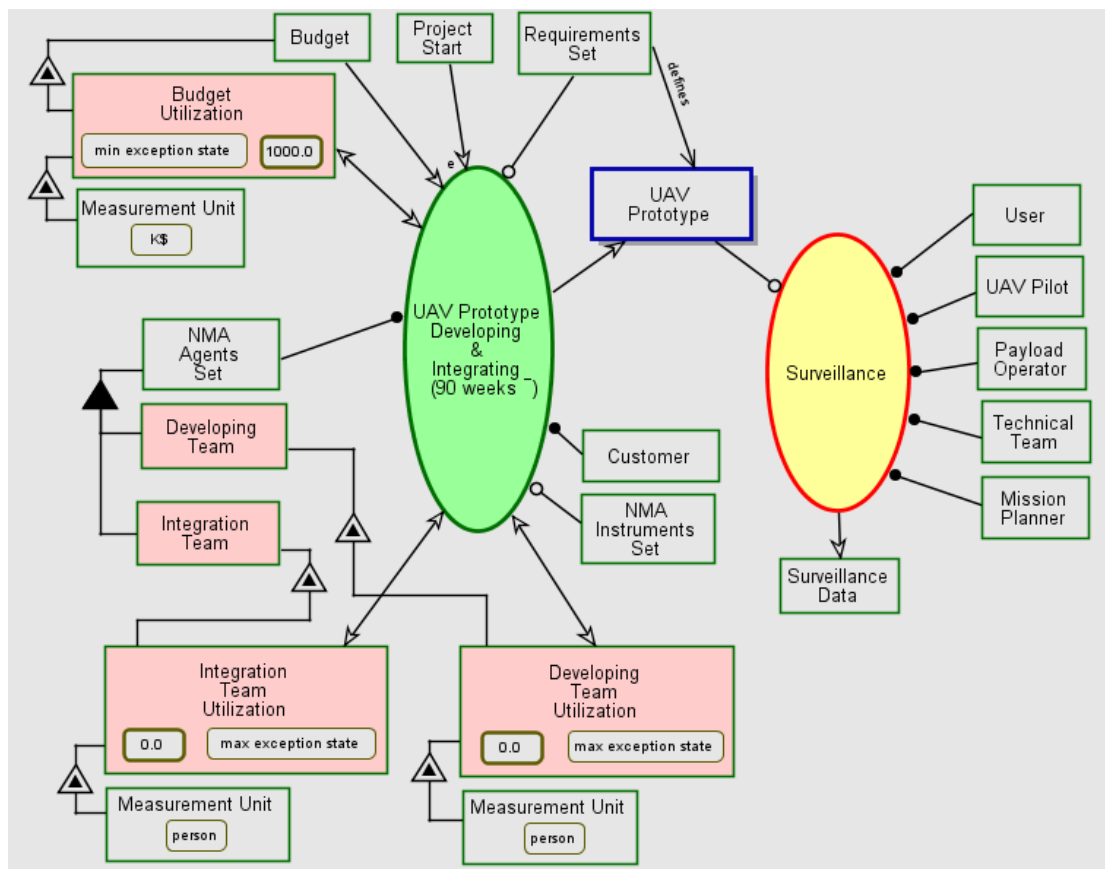


Figure 3-5 UAV PPLM model with assigned resources

3.2.3 Naming conventions of project objects, processes, and process durations

The syntax of the name of each process is such that it ends with parentheses containing the nominal, minimal, or maximal duration and the time unit name. Nominal duration is denoted by a number followed by the time unit name. For example, to denote that a process has a nominal duration of 15 weeks, the process name in the OPD will be followed by (15 weeks). Minimum and Maximum duration is denoted by two number separated by two periods followed by the time unit name. For example, to denote that a process has a minimal duration of 15 weeks and a maximal duration of 18 weeks, the process name in the OPD will be followed by (15..18 weeks). Minimum duration is denoted by a floor hyphen following the time unit name. For example, to denote that a process has a minimum duration of 15 weeks, the process name in the OPD will be followed by (15 weeks_). Similarly, maximum duration is denoted by a ceiling hyphen following the time unit name. For example, to denote that a process has a maximum duration of 15 weeks, the process name in the OPD will be followed by (15 weeks^).

The syntax of the name of each atomic (leaf-level) object is such that it ends with parentheses containing the phrase "outcome of" followed by the task ID.. For example, in Figure 3-6 the **Engine** and the **GFE Payload** are outcomes of the leaf-level processes i and j, respectively, as indicated in parentheses below their names: **Engine** (outcome of i) and **GFE Payload** (outcome of j). These conventions are useful for tracking relationships among leaf-level things (objects and processes), which are often not explicit in a single OPD.

3.2.4 Hierarchical Consistent and Coherent Decomposition

Figure 3-6 presents a third-level OPD of the model, in which the **UAV Prototype Developing & Integrating** process is detailed using the in-zooming refinement mechanism that is built into OPM. This ensures consistency with upper level OPDs of the model. The equivalent, automatically generated OPL paragraph is listed in Figure 3-7. At each detail level, the execution order of sub-processes within the in-zoomed process starts at the top of that process and proceeds downward. Some of the processes are leaf-level, atomic processes, indicated by thin ellipse contours, e.g., **Avionics Delivery & Checkout** and **Software Developing** are already tasks, i.e., simple leaf-level, atomic processes, as indicated by their thin ellipse contours. Other processes, indicated by thick ellipse contours, are non-simple and are further refined in subsequent lower-level OPDs.

In Figure 3-6 there are two pairs of processes modeled with FS relationships (case c in Table 3-2): **Software Developing** immediately follows **UAV Specifying** and **Integrating & Testing** immediately follows **Avionics Delivery & Checkout**. Two processes in the same figure are modeled using the new Floating Start & Finish (FSF) relationship (case d in Table 3-2 New OPM constructs defining time notion): The timing of **Vehicle Developing** floats within the timing of **Payload and Engine Developing**.

The time line in OPM is vertical, flowing from top to bottom. The first subprocess within **UAV Prototype Developing & Integrating** is **UAV Specifying**. When this process ends, **Software Developing** can start, as it is linked with Finish-to-Start relationship, as denoted by the dashed horizontal line between the two corresponding ellipses. The basic semantics of OPM would have been sufficient in this case, since **Software Developing** is modeled beneath **UAV Specifying**. One of the outcomes of the **UAV Specifying** process is the **Software Specification** object. **Software Specification** is assigned the role of document. This document is instrument to the **Software Developing**

process. Therefore, **Software Developing** cannot start before this deliverable is generated by the **UAV Specifying** process. The outcome of **Software Developing** is **Software Approval**, an object which is assigned the role of a gate. This gate is instrument of the **Integrating & Testing** process. Therefore, **Integrating & Testing** cannot start before this deliverable is generated by **Software Developing**.

Figure 3-6 OPD of the UAV Prototype Developing & Integrating In-zoomed

The **Payload & Engine Developing** process can start at some point in time following the start of **UAV Specifying**, provided that the **Engine Specification** object has been created by the **Engine Specifying** process, which is modeled in the OPD in Figure 3-6 as a subprocess of **UAV Specifying**. Similarly, **Payload Specification** is an outcome of **Payload Specifying**. For **Payload & Engine Developing** to start, **ECC Agents Set** and **Government Agents Set** must also be available.

Vehicle Developing can also start following the beginning of **UAV Specifying**, but later than the start of **Payload & Engine Developing**, since it requires **Vehicle Layout**, which results from **Vehicle Layout Designing**, which, in turn, can start after the completion of **Requirements Defining**, as shown in Figure 3-8.

When **Avionics Delivery & Checkout** ends, **Integrating & Testing** can start, as these two processes are linked with FS (Finish-to-Start) relationship, as denoted by the horizontal dashed line between their two corresponding ellipses. Starting the **Integrating & Testing** process requires also that the objects linked to it as instruments be available. These include the four components **Avionics**, **Internal Fittings Set**, **Structural Airframe Prototype**, and **Power System**, as well as one gate - **Software Approval**.

Integrating & Testing is the last process within **UAV Prototype Developing & Integrating**. When **Integrating & Testing** ends, it yields the **UAV Prototype**, built from the **GFE Payload** and **Engine**, which are in turn outcomes of **Payload & Engine Developing**.

The abstraction of processes in the model, i.e., their clustering into higher-level processes, was based on systems engineering practice. Each parent process is a logical cluster in the hierarchy of the project plan model. The processes in each cluster appear in the same OPD, which is a node in the OPD set tree. Different systems engineers would probably suggest somewhat different hierarchical process decomposition (which is equivalent to task clustering). Similarly, different decompositions of

hammocks would be defined by different systems engineers using a Gantt chart. The point to note is that the "real" model is the flattened, all-inclusive model at the most detailed level. Clustering to logical groups helps humans, who are subject to the limited channel capacity, grasp the "big picture". Hence, this decomposition is not necessarily identical in all the models of the same system.

UAV Prototype is physical.
UAV Prototype consists of **Engine (outcome of i)** and **GFE Payload (outcome of j)**.
Engine (outcome of i) is physical.
Engine (outcome of i) plays the role of **component**.
GFE Payload (outcome of j) is physical.
GFE Payload (outcome of j) plays the role of **component**.
Government is environmental and physical.
Government consists of **Government Agents Set**.
Government Agents Set handles **Payload & Engine Developing**.
ECC is environmental and physical.
ECC consists of **ECC Agents Set**.
ECC Agents Set handles **Payload & Engine Developing**.
NMA Agents Set handles **UAV Prototype Developing & Integrating**.
UAV Prototype Developing & Integrating exhibits **Flight Test Approval (outcome of v)**, **Internal Fittings (outcome of m)**, **Structural Airframe Prototype (outcome of r)**, **Power system (outcome of o)**, **Software Approval (outcome of h)**, **Avionics (outcome of p)**, **Vehicle Layout (outcome of e)**, **GFE Avionics Design (outcome of f)**, **Software Specification (outcome of g)**, **Payload Specification (outcome of d)**, **Engine Specification (outcome of c)**.
Vehicle Layout (outcome of e) plays the role of **document**.
GFE Avionics Design (outcome of f) plays the role of **document**.
Software Specification (outcome of g) plays the role of **document**.
Payload Specification (outcome of d) plays the role of **document**.
Engine Specification (outcome of c) plays the role of **document**.
Flight Test Approval (outcome of v) plays the role of **gate**.
Software Developing immediately follows **UAV Specifying**.
Integrating & Testing immediately follows **Avionics Delivery & Checkout**.
UAV Prototype Developing & Integrating consists of **Integrating & Testing**, **Software Developing**, **Avionics Delivery & Checkout**, **Vehicle Developing**, **Payload & Engine Developing**, and **UAV Specifying**.
UAV Prototype Developing & Integrating requires **NMA Instrument Facilities Set**.
UAV Prototype Developing & Integrating affects **New Millennium Aerospace (NMA) Inc.**.
UAV Prototype Developing & Integrating zooms into **UAV Specifying**, **Payload & Engine Developing**, **Vehicle Developing**, **Avionics Delivery & Checkout**, **Software Developing**, and **Integrating & Testing**, as well as **Engine Specification (outcome of c)**, **Payload Specification (outcome of d)**, **Software Specification (outcome of g)**, **GFE Avionics Design (outcome of f)**, **Vehicle Layout (outcome of e)**, **Avionics (outcome of p)**, **Software Approval (outcome of h)**, **Power system (outcome of o)**, **Structural Airframe Prototype (outcome of r)**, **Internal Fittings (outcome of m)**, **Flight Test Approval (outcome of v)**.
Avionics (outcome of p) is physical.
Avionics (outcome of p) plays the role of **component**.
Power system (outcome of o) is physical.
Power system (outcome of o) plays the role of **component**.
Structural Airframe Prototype (outcome of r) is physical.
Structural Airframe Prototype (outcome of r) plays the role of **component**.
Internal Fittings (outcome of m) is physical.
Internal Fittings (outcome of m) plays the role of **component**.
UAV Specifying yields **GFE Avionics Design (outcome of f)**, **Software Specification (outcome of g)**, **Vehicle Layout (outcome of e)**, **Payload Specification (outcome of d)**, and **Engine Specification (outcome of c)**.
Payload & Engine Developing requires **Engine Specification (outcome of c)** and **Payload Specification (outcome of d)**.
Payload & Engine Developing yields **Power system (outcome of o)**, **GFE Payload (outcome of j)**, and **Engine (outcome of i)**.

Figure 3-7 The automatically-generated OPL paragraph of the OPD in Figure 3-6

Vehicle Developing requires **Vehicle Layout (outcome of e)**.
Vehicle Developing yields **Internal Fittings (outcome of m)** and **Structural Airframe Prototype (outcome of r)**.
Avionics Delivery & Checkout requires **GFE Avionics Design (outcome of f)**.
Avionics Delivery & Checkout yields **Avionics (outcome of p)**.
Software Developing requires **Software Specification (outcome of g)** and **GFE Avionics Design (outcome of f)**.
Software Developing yields **Software Approval (outcome of h)**.
Integrating & Testing requires **Structural Airframe Prototype (outcome of r)**, **Internal Fittings (outcome of m)**, **Software Approval (outcome of h)**, **Avionics (outcome of p)**, and **Power system (outcome of o)**.
Integrating & Testing yields **UAV Prototype**.

Figure 3-8 The automatically-generated OPL paragraph of the OPD in Figure 3-6 (continued)

Modelling a deliverable as an outcome of the process producing it is highly valuable for both the project manager and the systems engineering manager. Examining relationships between corresponding tasks in a Gantt chart or an Activities Network Plan establishes their sequence without explicitly accounting for the role that expected deliverables play in the project plan. Milestones are a customary way to overcome this lack of incorporating deliverables into the model. Each milestone represents completion of an important deliverable. The sheer number of deliverables at all levels prevents defining each one of them as a milestone, so a human decision is required regarding whether to associate a milestone to each deliverable. The hierarchy built into the OPM model enables one to potentially define deliverables above a certain level as milestones.

There are four forth level OPDs in the model, represented in Figure 3-6 by thick-contour process ellipses: **UAV Specifying**, **Integrating & Testing**, **Payload & Engine Developing**, and **Vehicle Developing**. **UAV Specifying** is shown in Figure 3-9, and the equivalent, automatically generated OPL paragraph is listed in Figure 3-10. The four deliverables resulting from **UAV Specifying** are identical to the outcomes identified in the upper level OPD, but at this level, they are modeled as outcomes of specific subprocesses of **UAV Specifying**.

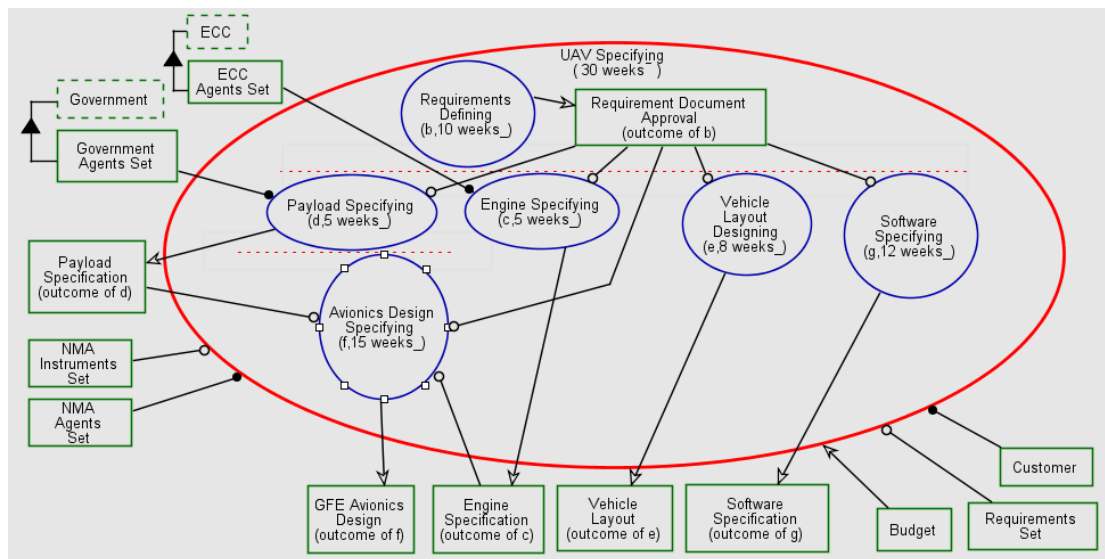


Figure 3-9 OPD of UAV Specifying in-zoomed

Requirements Defining is the first process to be executed under the **UAV Specifying** process cluster. Four processes can start after **Requirements Defining** process ends, as they are FS-related (i.e., related by the Finish-to-Start relationship), as denoted by the dashed horizontal line: **Payload Specifying**, **Engine Specifying**, **Vehicle Layout Designing**, and **Software Specifying**. They will start only if the **Requirements Document Approval Gate**, the outcome of **Requirements Defining**, is accomplished. **Avionics Design Specifying** is FS-related to the two predecessors **Payload Specifying** and **Engine Specifying**. In addition, **Requirements Document Approval** is required for **Avionics Design Specifying** to start, along with the availability of **Payload Specification** and **Engine Specification**.

NMA Agents Set handles **UAV Specifying**.
Government is environmental and physical.
Government consists of **Government Agents Set**.
Government Agents Set handles **Payload Specifying**.
ECC is environmental and physical.
ECC consists of **ECC Agents Set**.
ECC Agents Set handles **Engine Specifying**.
UAV Specifying exhibits **Requirement Document Approval (outcome of b)**.
UAV Specifying consists of **Avionics Design Specifying**, **Software Specifying**, **Vehicle Layout Designing**, **Payload Specifying**, **Engine Specifying**, and **Requirements Defining**.
UAV Specifying requires **NMA Instruments Set**.
UAV Specifying zooms into **Requirements Defining**, **Vehicle Layout Designing**, **Software Specifying**, **Payload Specifying**, **Engine Specifying**, and **Avionics Design Specifying**, and **Requirement Document Approval (outcome of b)**.
Software Specifying, **Vehicle Layout Designing**, **Payload Specifying**, and **Engine Specifying** immediately follow **Requirements Defining**.
Payload Specifying and **Engine Specifying** must finish at the same time.
Avionics Design Specifying immediately follows **Payload Specifying** and **Engine Specifying**.
Requirements Defining yields **Requirement Document Approval (outcome of b)**.
Vehicle Layout Designing requires **Requirement Document Approval (outcome of b)**.
Requirement Document Approval (outcome of b) plays the role of **gate**.
Vehicle Layout (outcome of e) plays the role of **document**.
GFE Avionics Design (outcome of f) plays the role of **document**.
Software Specification (outcome of g) plays the role of **document**.
Payload Specification (outcome of d) plays the role of **document**.
Engine Specification (outcome of c) plays the role of **document**.
Vehicle Layout Designing yields **Vehicle Layout (outcome of e)**.
Software Specifying requires **Requirement Document Approval (outcome of b)**.
Software Specifying yields **Software Specification (outcome of g)**.
Payload Specifying requires **Requirement Document Approval (outcome of b)**.
Payload Specifying yields **Payload Specification (outcome of d)**.
Engine Specifying requires **Requirement Document Approval (outcome of b)**.
Engine Specifying yields **Engine Specification (outcome of c)**.
Avionics Design Specifying requires **Requirement Document Approval (outcome of b)**, **Payload Specification (outcome of d)**, and **Engine Specification (outcome of c)**.
Avionics Design Specifying yields **GFE Avionics Design (outcome of f)**.

Figure 3-10 The automatically-generated OPL paragraph of the OPD in Figure 3-9

3.3 The UAV Product Domain

Figure 2-10 presents a third level OPD of the UAV PPLM model, presenting the **Surveillance** process which is the function of the UAV—the product of our project, yielding **Surveillance Data**. The equivalent OPL is listed in Figure 2-11. The UAV product is modeled simply with synchronous logic, without special conditions, events, loops, or paths. According to the model, **Surveillance** starts with the **Mission Defining**

process which requires the **User** as an agent to yield the **Mission Request**. The **Mission Request** is required both for the next executed process **Preparing** and its successive **Taxi & Takeoff** process. The **Preparing** process requires all agents except for the **User**, while the **Taxi & Takeoff** process requires the **Technical team** and the **UAV Pilot**. While **Cruising**, two more processes are performed in parallel: **Data Collecting** and **Mission Data Processing & Evaluation**. The **Mission Data Processing & Evaluation** yields the UAV ultimate deliverable – **Surveillance Data**. The last executed process is the **Landing & Taxi**.

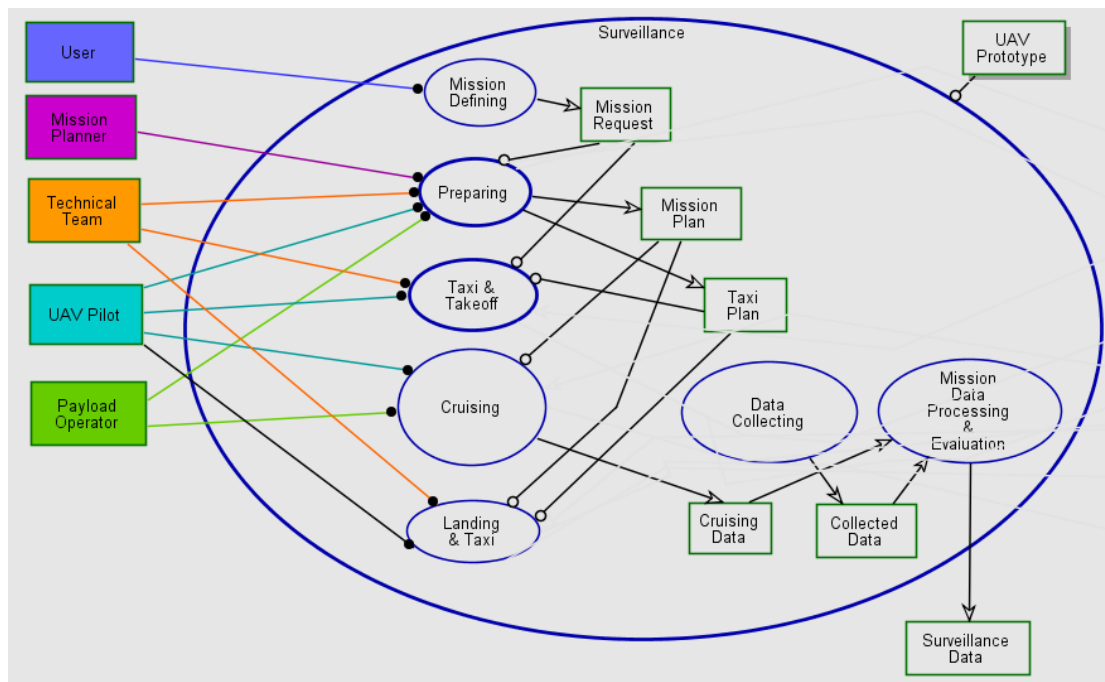


Figure 3-11 OPD of the Surveillance In-zoomed

UAV Prototype is physical.
Surveillance requires **UAV Prototype**.
User handles **Mission Defining**.
UAV Pilot handles **Preparing**, **Taxi & Takeoff**, **Cruising**, and **Landing & Taxi**.
Payload Operator handles **Preparing** and **Cruising**.
Mission Planner handles **Preparing**.
Technical Team handles **Preparing**, **Taxi & Takeoff**, and **Landing & Taxi**.
Surveillance zooms into **Mission Defining**, **Preparing**, **Taxi & Takeoff**, **Cruising**, **Data Collecting**, **Mission Data Processing & Evaluation**, and **Landing & Taxi**, as well as **Collected Data**, **Cruising Data**, **Taxi Plan**, **Mission Plan**, and **Mission Request**.
Mission Defining yields **Mission Request**.
Preparing requires **Mission Request**.
Preparing yields **Mission Plan**, and **Taxi Plan**.
Taxi & Takeoff requires **Taxi Plan** and **Mission Request**.
Cruising requires **Mission Plan**.
Cruising yields **Cruising Data**.
Data Collecting yields **Collected Data**.
Mission Data Processing & Evaluation consumes **Collected Data** and **Cruising Data**.
Mission Data Processing & Evaluation yields **Surveillance Data**.
Landing & Taxi requires **Mission Plan** and **Taxi Plan**.

Figure 3-12 The automatically-generated OPL paragraph of the OPD in Figure 3-10

Figure 3-13 presents the same OPD of the **Surveillance** process as in Figure 3-10 only this time some of UAV states are included in the view. The equivalent, automatically generated OPL is listed in Figure 3-14. When the **Preparing** process ends, it yields the **UAV Prototype** in **on** state or **ready for taxi** state, as further modeled and defined in the lower level OPD in Figure 3-15. The **ready for taxi** state is required for the **Taxi & Takeoff** process which yields the **UAV Prototype** in **ready for takeoff** state or **taking off** state, as further modeled and defined in the lower level OPD in Figure 3-16. The **UAV Prototype** hierarchical decomposition was modeled using the unfold mechanism, as presented in Figure 3-18.

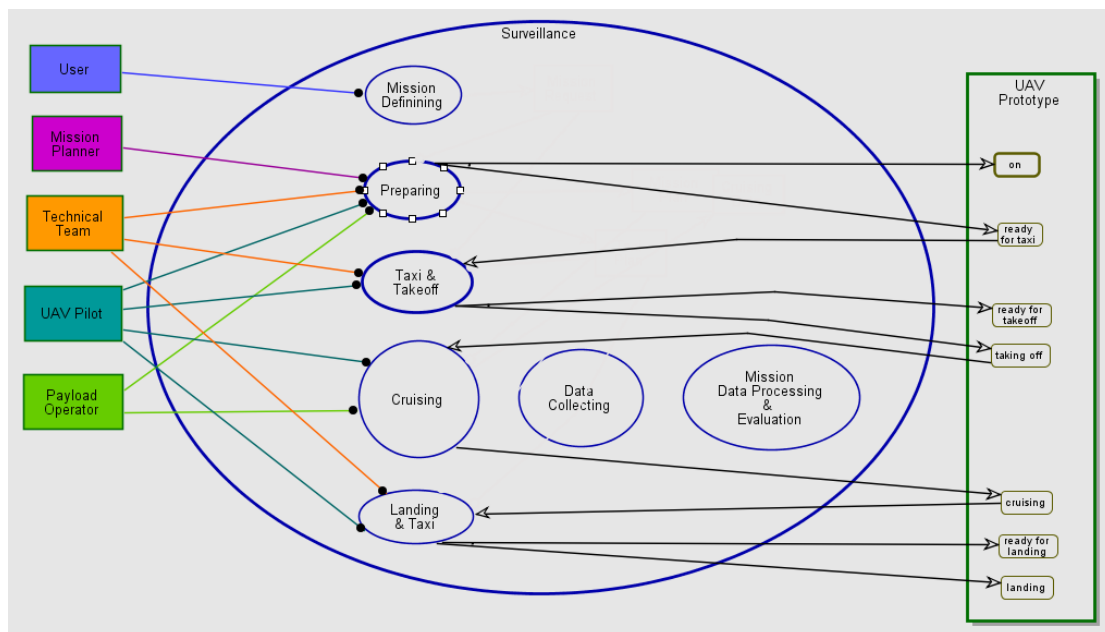


Figure 3-13 OPD of the Surveillance In-zoomed with UAV states

UAV Prototype is physical.
UAV Prototype can be, **ready for taxi**, **ready for takeoff**, **taking off**, **ready for landing**, **cruising**, or **landing**.
on is initial.
Technical Team handles **Preparation**, **Taxi & Takeoff**, and **Landing & Taxi**.
Mission Planner handles **Preparation**.
Payload Operator handles **Preparation** and **Cruising**.
UAV Pilot handles **Preparation**, **Taxi & Takeoff**, **Cruising**, and **Landing & Taxi**.
User handles **Mission Definition**.
Surveillance consists of **Landing & Taxi**, **Cruising**, **Taxi & Takeoff**, **Preparation**, **Mission Definition**, **Data Collecting**, and **Mission Data Processing & Evaluation**.
Preparing yields either **ready for taxi UAV Prototype** or **on UAV Prototype**.
Taxi & Takeoff consumes **ready for taxi UAV Prototype**.
Taxi & Takeoff yields either **taking off UAV Prototype** or **ready for takeoff UAV Prototype**.
Cruising changes **UAV Prototype** from **taking off** to **cruising**.
Landing & Taxi consumes **cruising UAV Prototype**.
Landing & Taxi yields either **ready for landing UAV Prototype** or **landing UAV Prototype**.

Figure 3-14 The automatically-generated OPL paragraph of the OPD in Figure 3-13

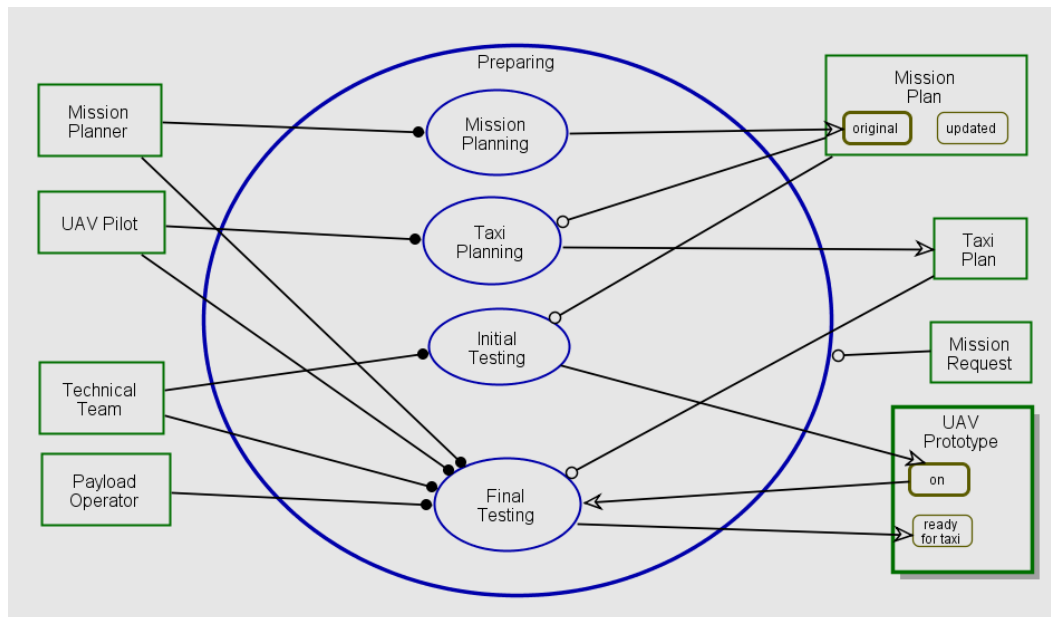


Figure 3-15 OPD of the Preparing In-zoomed

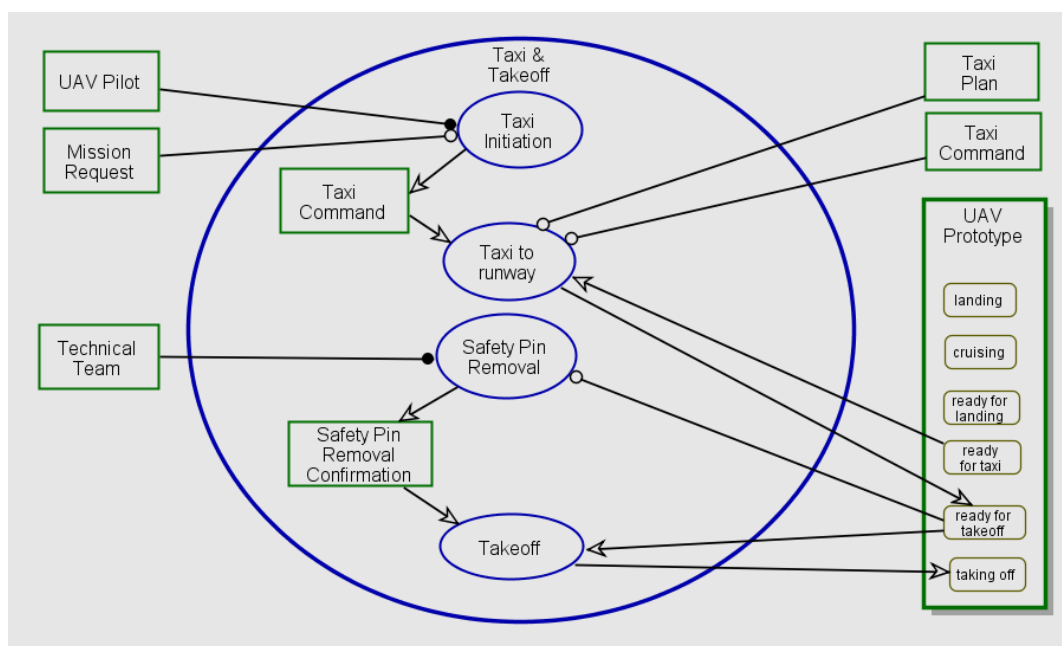


Figure 3-16 OPD of the Taxi & Takeoff In-zoomed

UAV Prototype is physical.
UAV Prototype can be on, ready for taxi, ready for takeoff, taking off, ready for landing, cruising, or landing.
UAV Prototype consists of Data Link (DL), Aerial Segment, and Ground Segment.
Data Link (DL) consists of Aerial DL (ADL) and Ground DL (GDL).
Aerial Segment consists of UAV Vehicle, Payload, and Engine.
UAV Vehicle consists of Fuselage, 2 Wings, and Empennage.
Payload consists of EOIR, EW, SPS, an optional MTI RADAR, an optional Maritime RADAR, and an optional SAR.
Ground Segment consists of Primary GCS, Back Up GCS, and SAR GS.
Primary GCS consists of Ground DL (GDL), Shelter, and Generator.

Figure 3-17 The automatically-generated OPL paragraph of the OPD in Figure 3-18

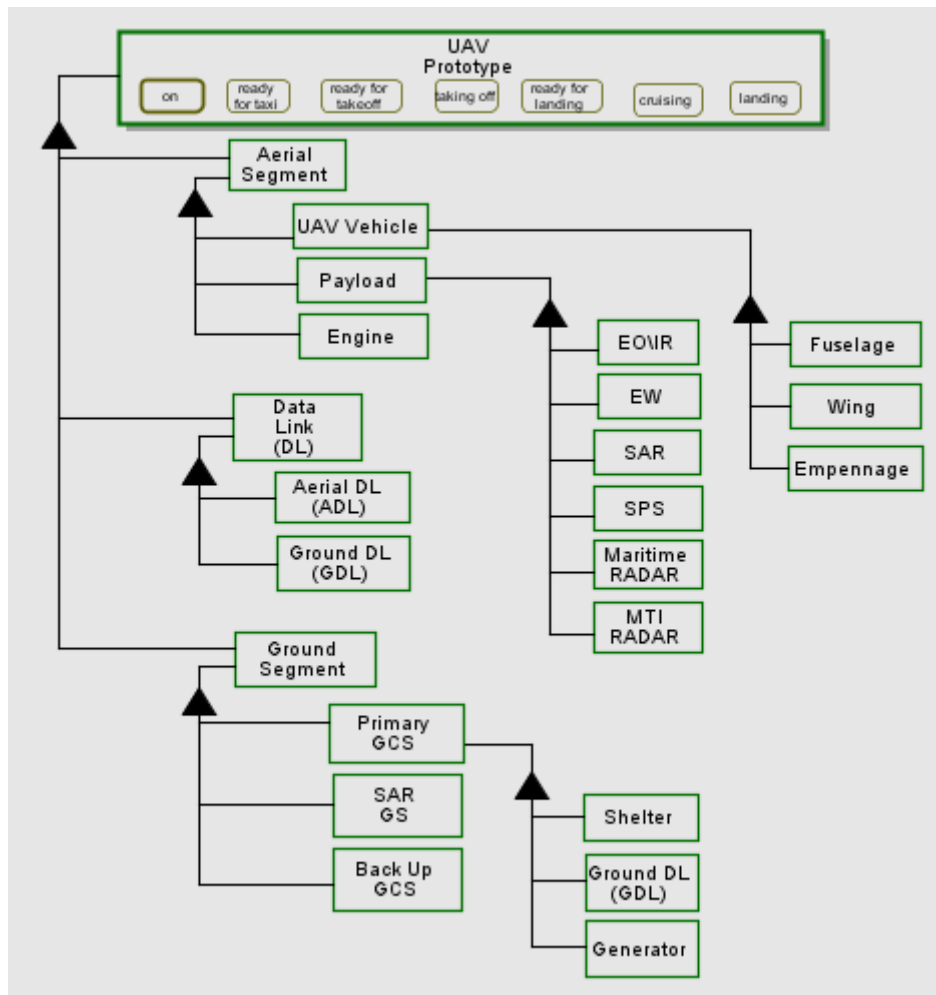


Figure 3-18 OPD of UAV Prototype unfolded

3.4 Simulation-Based Verification of the PPLM Model

Simulation of the OPM PPLM model can be used to validate the project-product plan. The simulation zigzags between the things (object and processes) of the project and those of the product. When a project process that is expected to yield a product component is stuck, the product processes that require this component as an instrument are stuck as well. Any project deliverable required for any other process in either the project or the product domain that is not generated as planned causes the project to halt or the product to fail in providing its expected function.

Using the simulation module of OPCAT, the plan scenarios are visualised for each view created in the model. Such a view is presented in Figure 3-19, showing simulation of the **UAV Prototype Developing** in-zoomed view. In this version of the model, **Project Start** and **Project End** were added with zero duration for exploration. In each simulation view an object is colored if it has existing instances (with indication of number of instances at the bottom right corner). Such are the three agents and the single instrument in Figure 3-19. The experimental **Project Start** process is colored since it is currently executed. The things (processes and objects) in red are on the critical path of the plan, as discussed in Subsection 5.3.

The simulation module enables the planner to do the following: (1) running the simulation forward or backward, (2) step-by-step or continuous mode simulation, (3)

pausing/continuing the simulation (for continuous mode), (4) controlling the simulation speed, and (5) defining and examining breakpoints. The simulation also provides a “lifespan diagram” which graphically describes the state of all the OPM entities at any stage of the simulation. In addition, a “Debug Info” is provided, notifying the planner about possible detected problems. The simulation capability enables saving scenarios, reproducing and inspecting them step-by-step, and analyzing interesting or problematic scenarios.

Figure 3-20 presents the simulation view generated immediately after the view presented in Figure 3-19, in which according to the plan logic, the **Definition** process starts immediately after the **Project Start**. **Requirements Definition** is colored, denoting that it is currently executed. The red dot runs along the link from **Requirements definition** to **Requirements Document Approval**, denoting transfer of control.

The next view is presented in Figure 3-21, showing the **Requirements Document Approval** colored, as it has an existing instance, from which four red dots are transferring control to the next four processes, out of which three are red, indicating that they are on the critical path. Once the four dots reach the respective processes, the four related objects are created. This is shown in Figure 3-22, where **Avionics Design Initiation** is currently executed, since its three instrument enablers—**Requirements Document Approval**, **Payload Specification**, and **Engine Specification**—exist. Once **Avionics Design Initiation** ends, the final outcome of the **Definition** process cluster—the GFE **Avionics Design**, is achieved, as indicated by the red dot just above this object.

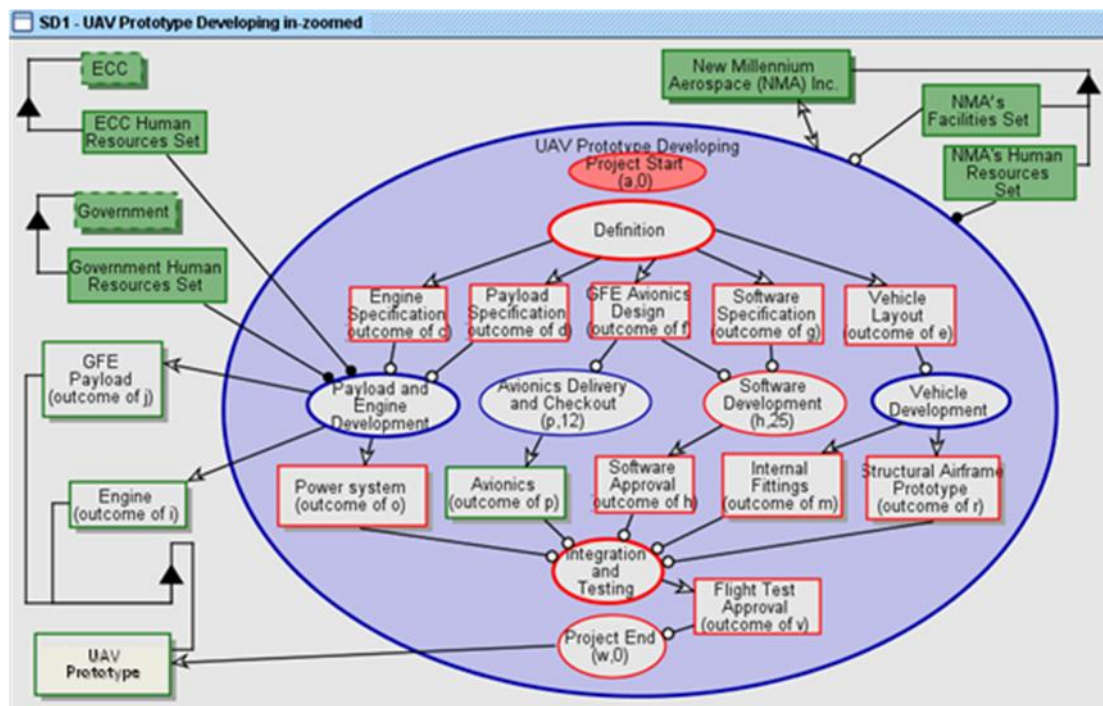


Figure 3-19 UAV Prototype Developing in-zoomed first simulation view

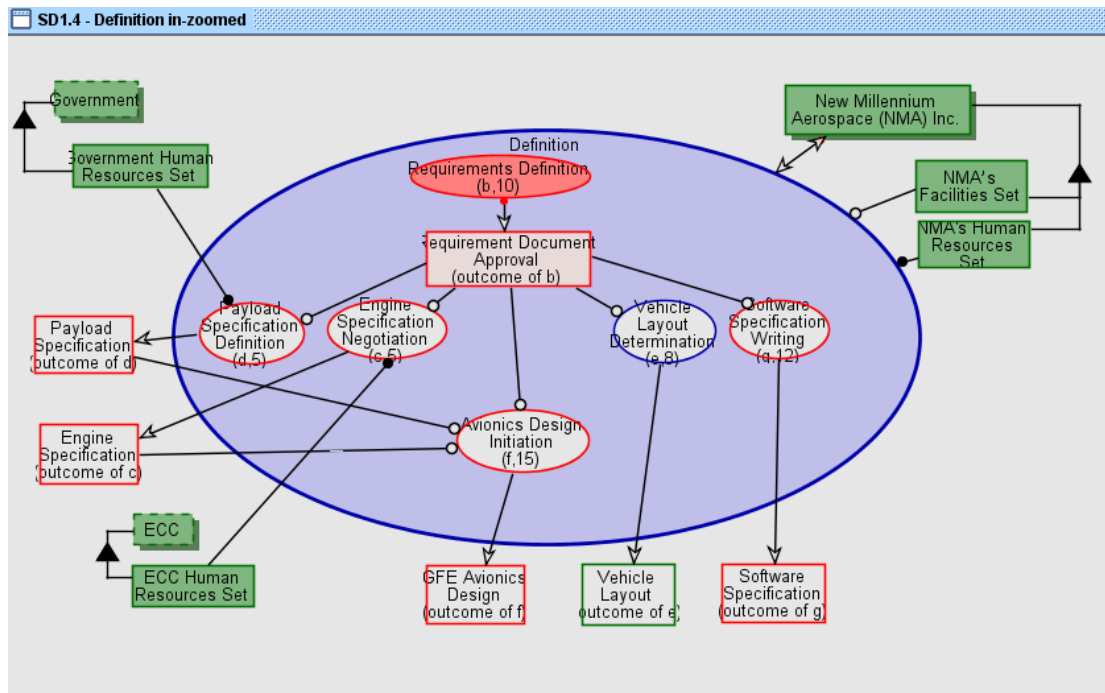


Figure 3-20 Definition process in-zoomed first simulation view

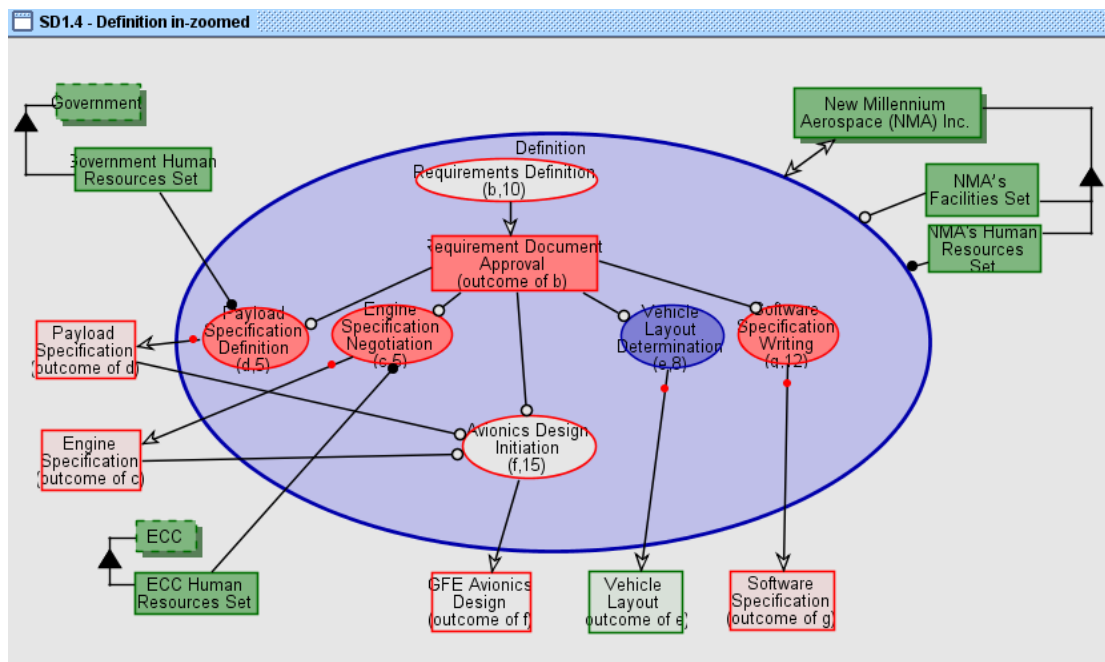


Figure 3-21 Definition process in-zoomed second simulation view

Figure 3-23 presents the first simulation view of the Vehicle development process cluster, in which the first two processes— **Fuselage Design** and **Empennage/Wing Design** start simultaneously. Both are executed since the required instrument **Vehicle Layout**—exists. Each one of the processes yields a corresponding object – the **Fuselage Design Document**, and the **Empennage/Wing Design Document** – both internal to the **Vehicle Development** cluster. Both objects are required for each one of the final processes of the **Vehicle development** cluster, which ultimately yield the **Internal Fittings** and the **Structural Airframe Prototype**, as shown in Figure 3-24. Both are inputs to processes on the critical path, as denoted by their red color. As shown in

Figure 3-25, these two critical objects are the enablers of the two first processes within the **Payload & Engine Development** cluster, which ends when its final process, **Power System Integration**, ends, yielding the final outcome of the cluster, the **Power System** deliverable.

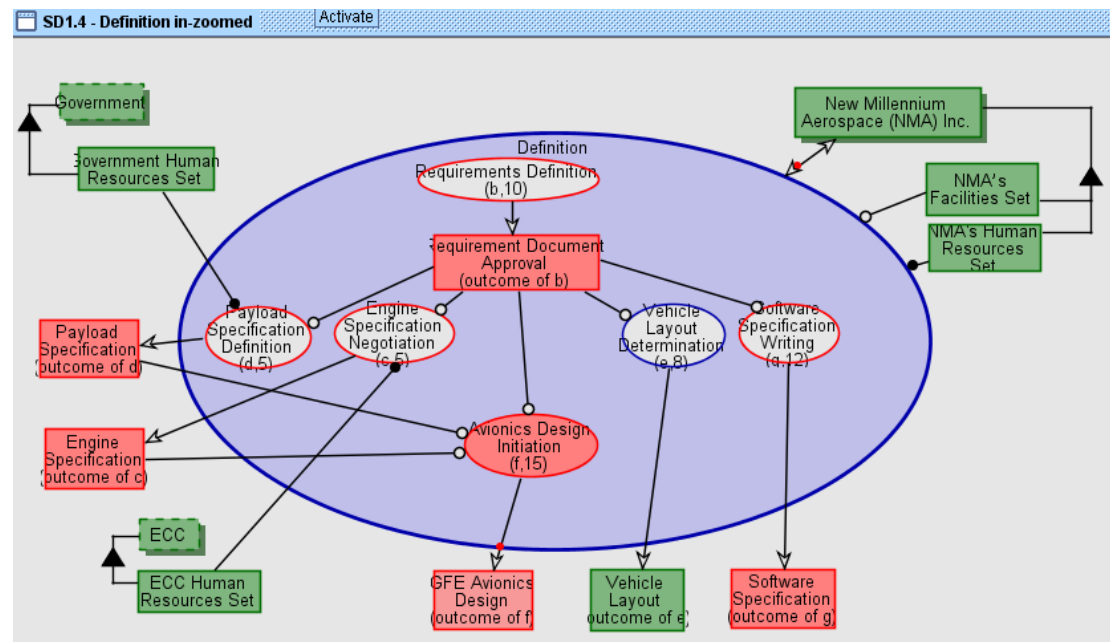


Figure 3-22 Definition process in-zoomed third simulation view

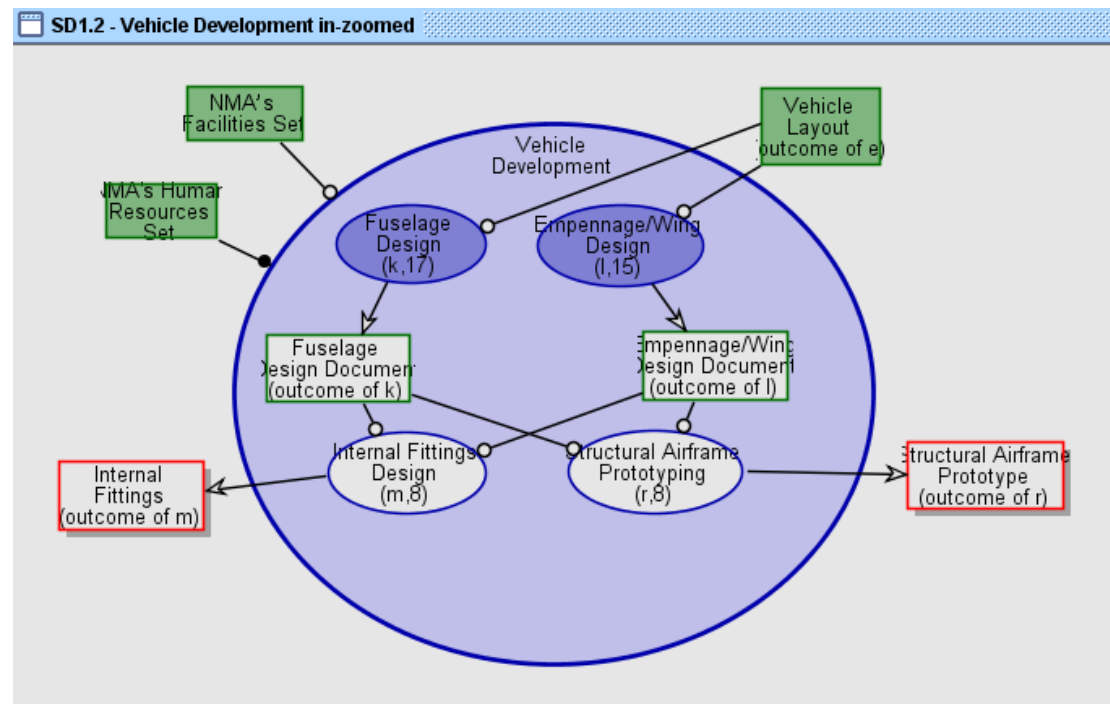


Figure 3-23 Vehicle Development process in-zoomed first simulation view

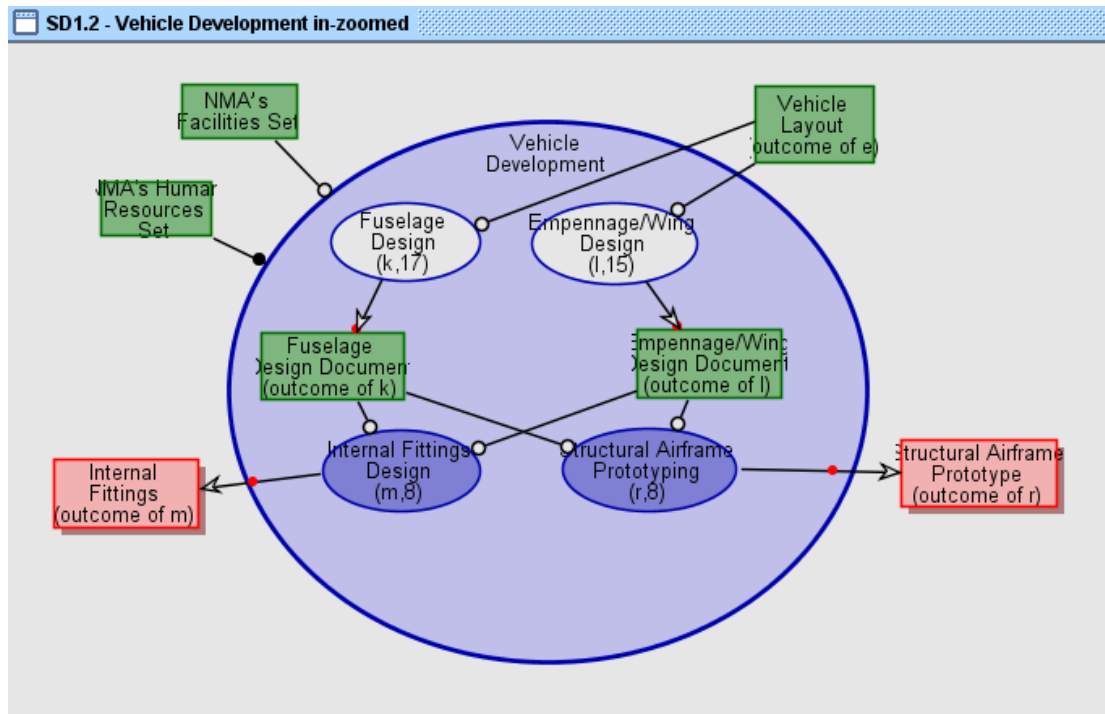


Figure 3-24 Vehicle Development process in-zoomed second simulation view

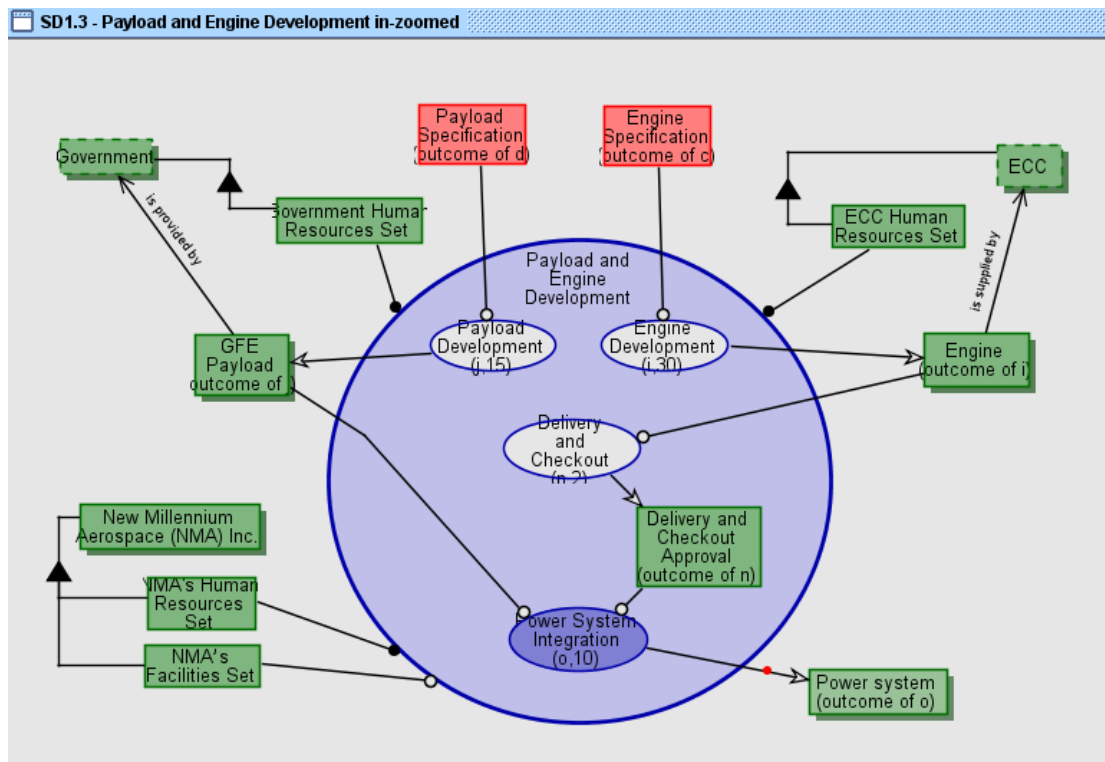


Figure 3-25 Payload and Engine Development process in-zoomed first simulation view

Integration & Testing starts with **Avionics/software Integration**, as shown in Figure 3-26, and it ends when **Flight Test Campaign** ends, yielding the **Flight Test Approval**, as shown in Figure 3-27. As shown in Figure 3-28, this is the last object modeled within the **UAV Prototype Developing** process cluster, and it is required as input to **Project End**, ultimately yielding the **UAV Prototype**.

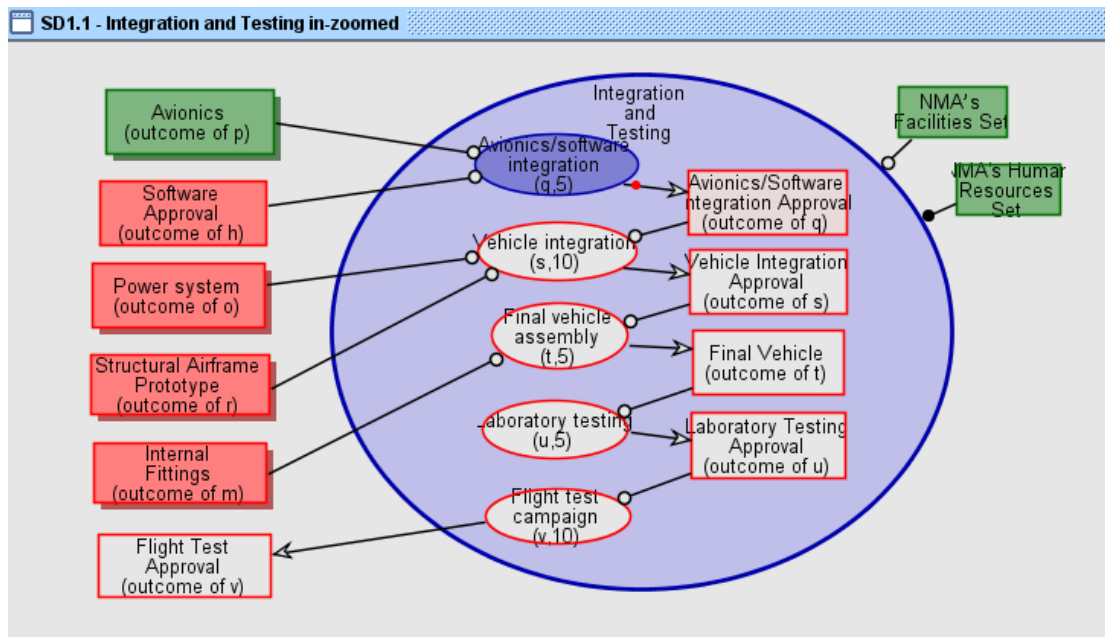


Figure 3-26 Integration and Testing process in-zoomed first simulation view

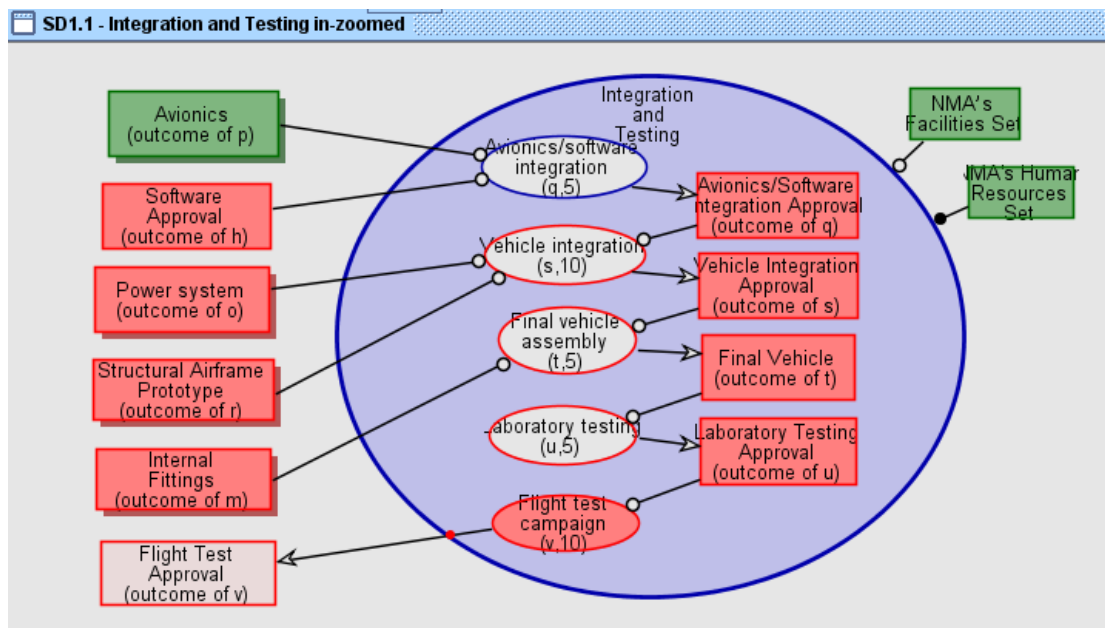


Figure 3-27 Integration and Testing process in-zoomed second simulation view

Some of the OPM constructs, such as control structures of "if then... else," events, and loops, do not commonly appear in the project plan model since they are traditionally not used in project plan logic. However, they can be used as needed in complex project models, as will be further discussed in Paragraph 7.1. Addition of object states is dealt with in Paragraph 5.7.3.

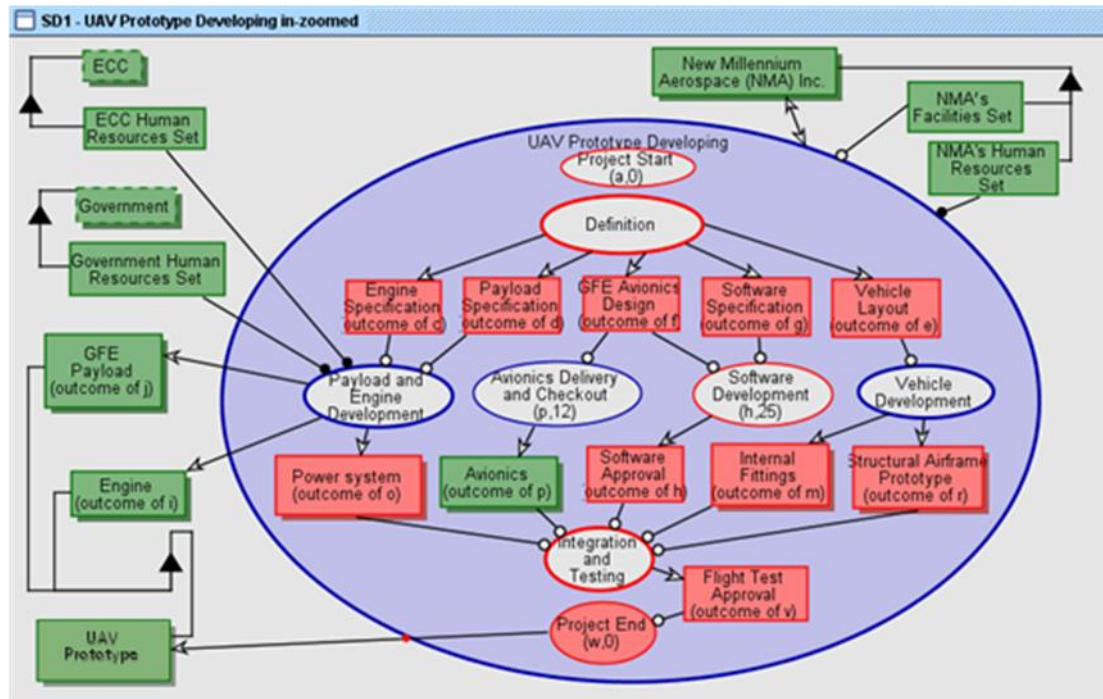


Figure 3-28 UAV Prototype Developing in-zoomed last simulation view

3.5 Summary

The ability to simultaneously express the required information from both the project and the product domains within a single integrated model-based framework can potentially lead to a more reliable project plan, which is less prone to the need for repeated changes and corrective actions. The increased robustness of the resulting project plan is attributed to the need to make decisions about the project's process order and logic while explicitly addressing the associated product model.

The planning process carried out following this approach clarifies the intricate relationships between the project and product entities. This model-based approach enables the simultaneous expression of the function, structure and behavior of both the project and the product via the same ontological and methodological foundations, maintaining full traceability between the project and product data. The use of simulation to validate the combined plan further contributes to its robustness.

The research conducted in order to explore systems engineers' comprehension of the product and project dimensions is presented next in Chapter 4. Since all the entities and their relations are represented in the model, automatic procedures can be devised to generate other project view representations, including Gantt chart, Activities Network Plan, Critical Path, and a corresponding Work Breakdown Structure. This will be further discussed in Chapter 5.

4 Application of Project Management Methods within Systems Engineering Management⁴

4.1 The Compared Project Management Methods

Despite the differences between project management and systems engineering management, the approaches and methods for project planning used by systems engineers are basically the same as those used by project managers. The arsenal of project management methods includes Work Breakdown Structure (WBS) [41], Product Breakdown Structure (PBS) [19], Reliability, Availability, Maintainability (RAM) [42], Critical Chain Project Management (CCPM) by Theory of Constraints (TOC), resource scheduling, procurement techniques, contract bidding techniques, and risk management methods. Other, more traditional methods, are EVM, SDM, SD, CPM/PERT, and Gantt chart. These methods, together with WBS are the most referenced PM methods in systems engineering handbooks, and are most commonly used within SEM. We refer to these methods, together with our seventh method – Object Process Methodology (OPM) – in this chapter in order to gauge the potential use of OPM for project planning and management.

The seven PM methods compared in our research are diversified in terms of the objectives achieved by using each one, as listed in Table 4-1. These methods were compared in order to answer our second research question: how do systems engineers perceive the extent to which, and ways by which, the selected project planning and control methods support Systems Engineering Management. The seven PM methods are described briefly in this section.

Table 4-1 The seven investigated project management methods

Project management method – short name	System Dynamics	Program Evaluation and Reviewing Technique	Critical Path Method	Design Structure Matrix	Earned Value Method	Gantt chart	Object Process Methodology
Project management method – full name	SD	PERT	CPM	DSM	EVM	Gantt	OPM
Main objective	Project planning – Dynamic Modelling	Project planning – Schedule	Project planning – Schedule	Project planning (and product design)	Project control	Project planning – Schedule	Project planning – combined product/activity based

4.1.1 The Critical Path Method and Program Evaluation and Reviewing Technique

The *Critical Path Method* (CPM) is a network model of the project, developed in the 1950's by DuPont for management of maintenance projects in industrial factories. CPM depicts tasks along with dependency information, duration, and the slack time for each activity. CPM chart time is deterministic, resulting in a fixed estimate of the

⁴ A self-contained version of this chapter was submitted to the Journal of Transactions on Systems, Man, and Cybernetics-Part C

time required to complete the project. The method shows the activities that are critical to maintaining the schedule. Delay in an activity on the critical path delays the entire project. To accelerate the project, it is necessary to reduce the total time required for the activities performed in sequence along the critical path. Figure 4-1 [43] depicts an example CPM chart of an Unmanned Aerial Vehicle (UAV) with 23 tasks, their slacks, and the relationships. The three critical paths are denoted by solid lines.

Like CPM, the *Program Evaluation and Reviewing Technique* (PERT) is a network model, developed in the late 1950's as part of the U.S. Navy's Polaris project. Similar to the CPM chart, PERT depicts tasks along with dependency information and duration. This allows for assigning parametric probabilities to task completion times in accordance with optimistic, pessimistic, and likely estimations. Although both CPM and PERT models contain all the information, it is difficult to track the project progress using these network diagrams.

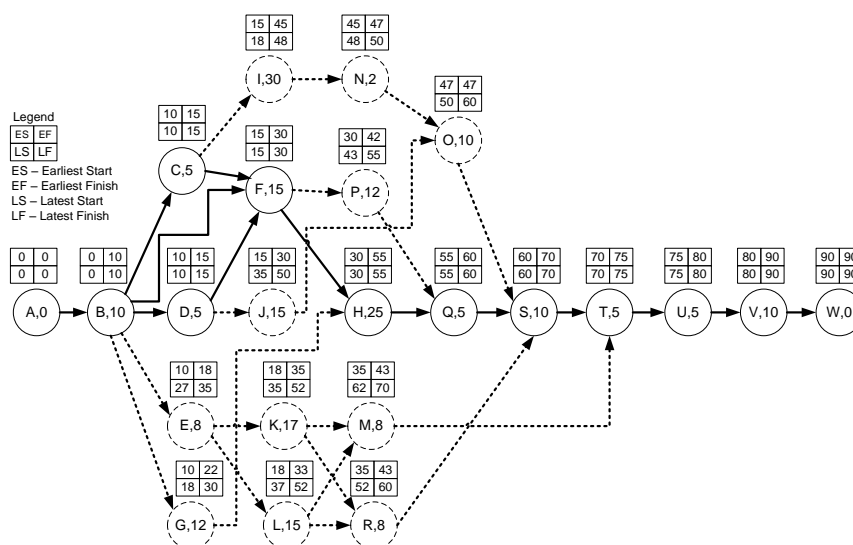


Figure 4-1 Example CPM chart of an Unmanned Aerial Vehicle (UAV) project

4.1.2 The Gantt chart

The *Gantt chart*, shown in Figure 4-2 for the same UAV project, is probably the most widely used method for project management. Gantt chart comprises horizontal scheduling bars with time flowing from left to right, allowing for both planning and tracking of project schedule. In its original form, invented roughly frothy years before PERT and CPM, the Gantt chart showed the timing of tasks without specifying relationships among them. Constraining relationships between tasks, such as start-to-start and finish-to-start, shown as the vertical directed lines in Figure 4-2, were added only in the late 1990's, making it possible to view and understand the impact of a single task delay on the entire project duration. Indeed, in its latest form, Gantt charts are used also for CPM-based project analysis, using software tools to extend the network models representation.

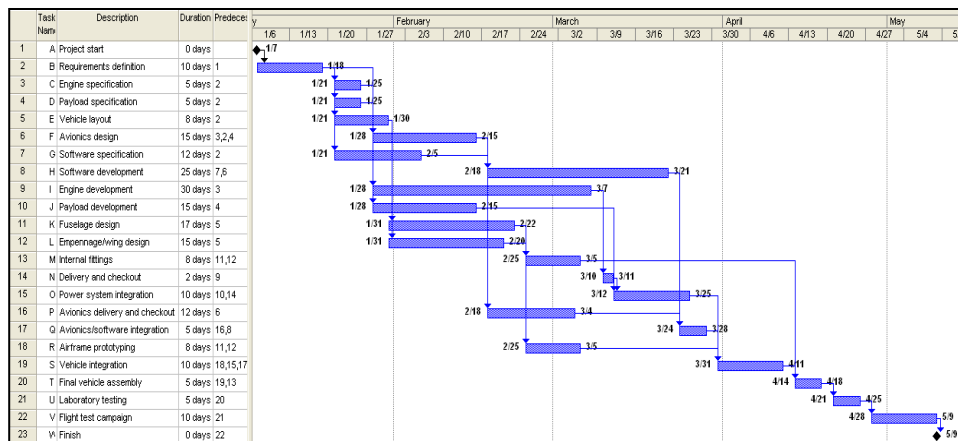


Figure 4-2 Gantt chart of the UAV project

4.1.3 The System Dynamics Method

System Dynamics (SD) was developed initially from the work of Jay W. Forrester [44], [45]. Over the last twenty years, SD has been used to model complex development projects in order to improve their performance [46]. System Dynamics models are created and used for project planning, addressing planned budget, schedule, resources, risks, and past experience. SD can help map out iteration issues and highlight rework loops. During the project planning process, SD models can predict effects of changes for different scenarios on a project's productivity and quality due to various factors. Figure 4-3 is an SD model of the UAV project, depicted in both the PERT network in Figure 4-2 and the Gantt chart in Figure 4-1. In the CPM and Gantt models, the 23 tasks are shown with varying amounts of effort, while in an aggregate SD model, all the tasks should be of approximately the same amount of effort. Therefore, the model of the project consists of approximately 100 equivalent tasks (the exact number does not matter, as the productivity would change correspondingly). Figure 4-3 depicts the graph of amount of work (number of tasks out of 100) done as a function of time for two staffing quality levels: perfect quality (1.0) and quality of 0.7.



Figure 4-3 System Dynamics chart of the UAV project

4.1.4 The Design Structure Matrix

The *Design Structure Matrix* (DSM) is a square matrix representation of interactions among entities in a system. In project management, it enables modelling and analyzing dependencies among components, tasks, or teams in a project. The use of matrices in system modelling can be traced back to Warfield in the 1970's and Steward [47]. In the 1990's, the method received attention and became widespread. Eppinger et al. [48] used a matrix representation to capture both the sequence of and the technical relationships among design tasks, which were then analyzed in order to find alternative sequences and/or definitions of the tasks. Eppinger used DSM also in the context of project management.

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
a	a																						
b	X	b																					
c		X	c																				
d		X		d																			
e		X			e																		
f		X	X	X		f																	
g		X					g																
h						X	X	h															
i			X						i														
j				X						j													
k					X						k												
l					X							l											
m										X	X		m										
n								X						n									
o									X					X	o								
p						X										p							
q							X									X	q						
r									X	X								r					
s															X	X	X	s					
t												X						X	t				
u																			X	u			
v																				X	v		
w																					X	w	

Figure 4-4 Example DSM

Browning [49] suggested four different types of DSM according to the data that the DSM represents: (1) Component-based DSM, which represents relations among objects in the system, for use in systems engineering and architecting, (2) task-based DSM, which represents relations among tasks or activities—processes in the system, for use in project management, (3) parameter-based DSM, which is similar to (2) but at a lower level, and (4) team-based DSM, which interfaces between teams, for use in organizational design and multi-team projects. The task-based DSM provided in Figure 4-4 [43] shows the same 23 tasks and relationships of the UAV project after matrix manipulations that avoid iterations, since all the interactions are below the matrix diagonal.

4.1.5 The Earned Value Management

Used for project performance measurement, the *Earned Value Management* (EVM) is a control method based on conversion of scope of work to budget terms. The underlying concept of EVM has been in use for over a century, and the US department of defense has used EVM in a modest measure back in the 1960's. EVM was elaborated in recent years and became an ANSI standard, which provides a forecast to a project's end along with indications of exceptions to the plan. The basic idea behind EVM is comparing planned and actual work in order to determine budget

and schedule propagation. The earned value can only be calculated after a project's actual start, and it is always defined in reference to a specific control point in time. The different EVM parameters are calculated at the task level and are summed up in an aggregate manner for the entire project. Figure 4-5 is an example of an EVM plot, depicting for the same project shown in Figures 4-2, 4-3, and 4-4, the Curves of Planned Value (PV), Actual Cost (AC), and Earned Value (EV) against each other.

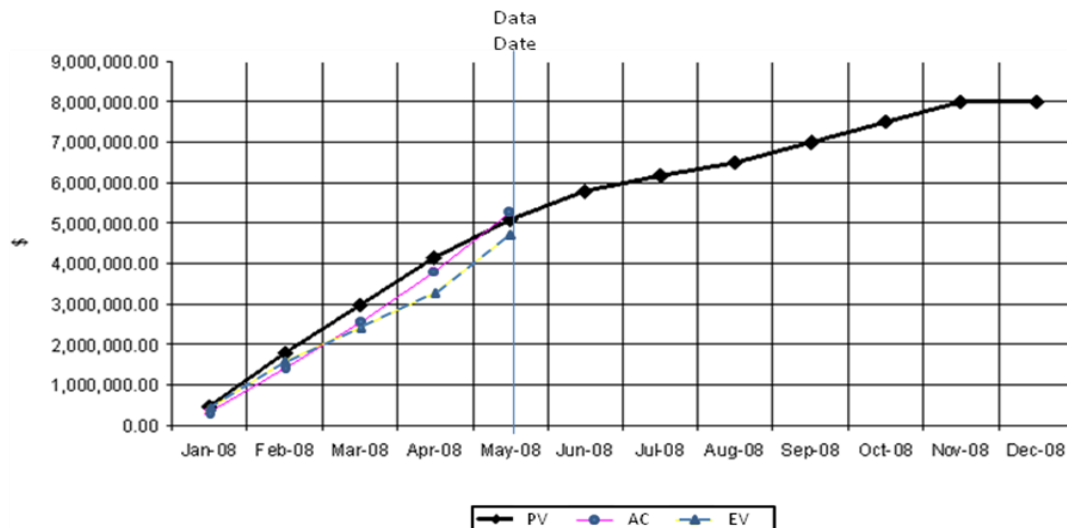


Figure 4-5 EVM chart of the UAV project

4.2 Research Population and Setting

The research population consisted of 24 mid-career systems engineers from companies across the USA with 5-8 years of practice, who were among about 80 graduate students in the Systems Project Management course. This course is one of three core mandatory courses in the Systems Design and Management (SDM) program at MIT's Engineering Systems Division. During the spring 2008 semester course (see Appendix A), the research participants studied the seven project management methods surveyed above and practiced them through targeted homework assignments, which are listed in Table 4-2. The table also presents the quantity of class lessons devoted to each one of the project management methods (see Appendix B). Gantt and OPM were not thought as part of the course, since all students were supposed to have former acquaintances with both these methods. A short reminder has been reviewed, followed by teaching of the model-based project planning approach and introduction of the PPLM research.

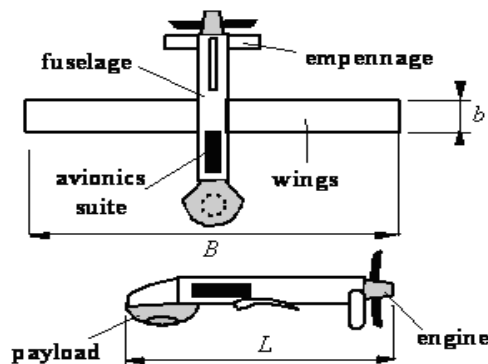
The entire student population was informed about HW5 being part of a research and non-mandatory. The 24 respondents elected to do HW5 and participate in the study. Some of their motivation was the option they were given of having their final grade based on the best five out of six homework assignments.

Table 4-2 The seven investigated project management methods

Project management method – short name	System Dynamics	Program Evaluation and Reviewing Technique	Critical Path Method	Design Structure Matrix	Earned Value Method	Gantt chart	Object Process Methodology
Project management method – full name	SD	PERT	CPM	DSM	EVM	Gantt	OPM
Homework assignment	HW1	HW2	HW2	HW3	HW4	HW5	HW5
Quantity of class sessions	6	1	1	2	1	Only reviewed prior to presenting PPLM approach*	Only reviewed prior to presenting PPLM approach*

* The PPLM approach was presented during 1.5 class sessions.

An Unmanned Aerial Vehicle (UAV) case study served as a running case study for all of the homework assignments. This case study concerns a project of developing a UAV by a fictitious government-contracted leading UAVs manufacturer, *New Millennium Aerospace (NMA) Inc.* A rough specification and sketch of the UAV “pusher” vehicle concept, shown in Figure 4-6, was given to the students. The UAV's payload in this case study is provided by the government as "modified government furnished equipment" (GFE), while the engine is supplied by an established commercial company (ECC) under a subcontract.



UAV concept, Specifications:
 $L=2000$ mm, $B=3500$ mm, $b=500$ mm

Figure 4-6 A rough specification and sketch of the UAV vehicle concept

For their first homework (HW1), all the students were tasked with creating a simple SD model and exploring its behavior. They examined the impact of uncertainties in project assumptions on cost and schedule. In HW2, they created a project plan using the Critical Path Method (CPM), drew a project graph, estimated the early finish (EF) time of the project and identified the critical path and slack times. Using PERT, they had to analyze the impact of changes in individual task times on the critical path and

consider probability distributions of task times and their effect on the project schedule.

HW3 called for applying DSM. Students first translated the project graph from the previous assignment to a DSM representation. Next, they added iterations to the project and analyzed their effect on the previous task sequence. They then had to consider partitioning the DSM to reveal meta-tasks. Finally, they estimated the effect of these changes on the critical path and estimated project completion time.

For HW4, the students focused on tracking projects and computing the various metrics defined in EVM terms of cost and schedule in order to assess the overall performance of the project and to critically analyze and interpret the results.

Finally, based strictly on the text given in a previous homework assignment (HW2), HW5 called for creating two project plan versions, one using a Gantt chart model and the other using OPM. At the second part of the homework the students were asked to conduct a comparison of all the seven project management methods they had studied in the course with respect to a set of 14 project management factors, as described in the next section.

4.3 Research Methodology

The research was aimed at exploring answers to our first two research questions: (1) while conducting the systems engineering management, do practitioners perceive a notion of a project-domain, a product-domain, and a combined project-product domain? (2) How do systems engineers perceive the extent to which selected seven PM methods support SEM?

Since the investigated project management methods were taught in the course during lectures and practiced through homework assignments, we assumed that the participants had identical knowledge of, and training level in, these methods. Furthermore, since the same system project case study—an Unmanned Aerial Vehicle—served as the basis for all the assignments throughout the entire course (except for the final projects), the experience students gained in applying all seven methods can be taken to be free of system-specific bias.

4.3.1 The 14 project management factors and their latent dimensions

Recognizing that systems engineering management entails both the product and the project viewpoints, we defined 14 factors that account for both major classical project management issues and aspects of the joint project-product ensemble, which is at the focus of Systems Engineering Management. These 14 factors were introduced to all the participants in a random order listed in Table 4-3.

Four of the 14 SEM factors belong to the "classical" project management domain and are indeed addressed by common project management methods. These include: (1) budget/schedule measurement/ tracking, (2) budget/schedule forecasting, (3) resource management, and (4) iterations management. These four factors are categorized in the project latent dimension.

Four other factors fit in the product domain: (1) product planning, (2) product measurement/tracking, (3) product quality, and (4) performance quality. The remaining six factors are common to the combined product-project domain, as they cannot be uniquely associated with either the product alone or the project alone. These four factors are categorized in the product latent dimension.

The remaining six factors are categorized in the project-product latent dimension. With respect to risk management, we adopted NASA's viewing it as being common to both the project and the product domains [3]. This approach is founded on the premise that there are technical risks, which are mostly in the product domain, and managerial risks, which are mostly in the project domain, This is contrary to the approach of leading standards [5], [6], which view risk management primarily as a managerial issue and therefore relate to it as project domain issue.

Table 4-3 The 14 Systems Engineering Management Factors

SEM Factor		Latent Dimension
1.	Budget/Schedule measurement/tracking	Project
2.	Budget/Schedule forecasting	Project
3.	Inter-relationships (process & product)	Project-Product
4.	Resource management	Project
5.	Stakeholders/agents tracking	Project-Product
6.	Performance quality	Product
7.	Product quality	Product
8.	Product planning	Product
9.	Product measurement/tracking	Product
10.	Risk management	Project-Product
11.	Iterations management	Project
12.	Information resolution level	Project-Product
13.	Ease of communication	Project-Product
14.	Change management	Project-Product

4.3.2 The survey questions and their analysis method

The 24 research participants were instructed to rank each one of the 14 factors for each one of the seven systems engineering management methods using a Likert scale [50] of 1 to 5, where 1 is poor, 2 is fair, 3 is good, 4 is very good, and 5 is excellent. N/A was denoted by 0.

The question posed to the participants was phrased as follows: *"Please compare the project models or representations you have done so far as homeworks, with respect to the 14 Project Management considerations. Wherever you believe a correlation exists between a model and a PM consideration, provide a short written explanation of the relationship and grade its strength numerically (between 1 and 5 as specified)."*

Since the participants were practicing systems engineers, their views of the project management tools tended to reflect the application of these methods in systems engineering management more than in project management. To examine the participants' views of each project management method with respect to each factor, we compared the responses for each one of 14 factors with respect to each one of the seven PM methods.

The students were not instructed in any way to think specifically of the considerations as related to "project," "product," or "project-product" dimensions. Our aim was to explore whether their unguided perceptions towards the 14 different factors would

reflect recognition of these factors as related to our three predefined latent dimensions of "project," "product," and "project-product." To avoid any potential influence on the responses, in the instructions we elected to use the phrase "Project Management (PM) considerations" rather than "Systems Engineering Management factors," which might have diverted the respondents to go in the SEM direction.

To determine whether our classification of the 14 factors into the three latent domains can be verified by the research participants' responses, we first analyzed the grades they had given for each factor and method combination. Using Alpha Cronbach coefficient [51] we determined whether the domain-categorized factors can be considered a dimension, namely project dimension, product dimension, and project-product dimension.

The Alpha Cronbach coefficient is used for estimating how well a set of variables measures a single uni-dimensional underlying construct. It determines the internal consistency of items within a single test, indicating reliability. The reliability is in terms of the ratio between the true score variance of the "underlying construct" and the observed score variance of that uni-dimensional construct, where the construct is the hypothetical variable that is being measured [52]. The Alpha Cronbach coefficient ranges in value from 0 to 1, when 0.70 is defined [53] as the cutoff value to be an acceptable reliability. It increases when the average inter-variables correlation increases. Therefore, high values of Alpha Cronbach provide evidence that the variables included in its calculation measure the same underlying construct. For this reason, Alpha Cronbach is often used in order to probe underlying constructs that the researcher wants to measure, as part of developing predicting variables and objective scales in surveys.

The sum of all the participants' Likert scale rankings for each factor was calculated, and the sum of all 14 factors for each PM method was taken as that method's score. The variables for the Alpha Cronbach coefficients for each PM method were calculated from the Likert scale results for each group of factors defined for each domain: (a) The project domain, consisting of factors 1, 2, 4, and 11, (b) The product domain, consisting of factors 6, 7, 8, and 9, and (c) The project-product domain, consisting of factors 3, 5, 10, 12, 13, and 14. Additionally, we calculated the Alpha Cronbach coefficient also for a fourth potential dimension—the *combined* project-product domain, which is the combination of eight factors: the four project factors 1, 2, 4 and 11 and the four product factors 6, 7, 8, and 9.

4.4 Results and Analysis

In this paragraph we present and discuss the results of the participants' comparison of the seven project management methods they had studied, listed in Table 4-2, with respect to the 14 project management factors, listed in Table 4-3.

4.4.1 Methods Comparison by Factors

Alpha Cronbach coefficient serves as a basis for comparing between the methods, initially using all 14 factors. The Alpha Cronbach coefficients, presented in Table 4-4, are higher than 0.70 for all but the Design Structure Matrix (DSM) method. Therefore we can use the participants' rankings for all the 14 factors for the sake of comparison between the six PM methods, from which DSM is excluded. When excluding two factors which are in the Project-Product latent domain—factor 12 (Information Resolution Level), and factor 3 (Inter-relationships, process & product)—DSM exceeds an Alpha Cronbach coefficient value of 0.7. Excluding these

two factors for all the seven PM methods, we are left with a set of 12 factors that can be reliably used for the comparison of all the seven PM methods.

Figure 4-7 represents by the dark bars the sum of scores of the 14 factors participants assigned for each method. OPM scored the maximum sum, 885 points. A cutoff value of 664 points, which is 75% of this maximum score, leaves us with three methods: OPM, SD, and EVM.

The light grey bars in Figure 4-7 represent the sums of rankings of the 12 factors (where factors 3 and 12 are excluded). With these 12 factors, SD scored the maximum sum, 769 points. Assigning a cutoff of 577, which is 75% of this maximum, leaves four methods in the game: OPM, SD, EVM, and DSM. The sum of all the participants' rankings for each factor was calculated, and the sum of the 12 factors for each PM method was taken as that method's final score.

Table 4-4 All Factors Set Reliability

Project Management Method	SD	PERT	CPM	DSM	EVM	Gantt	OPM
Full name	System Dynamics	Program Evaluation and Reviewing Technique	Critical Path Method	Design Structure Matrix	Earned Value Method	Gantt Chart	Object Process Methodology
Cronbach's Alpha	.743	.793	.754	.640	.757	.760	.855
Best Improved	-	-	-	.702 ⁽¹⁾	-	-	-

(1) Improved by deletion of factor 12 – Information Resolution Level and factor 3 - Inter-relationships (process & product)

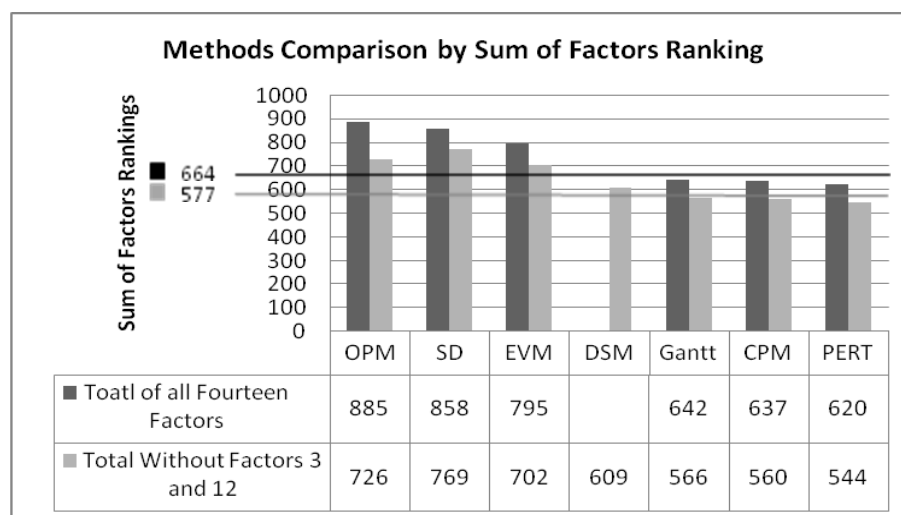


Figure 4-7 Project management Methods Comparison by Sum of Factors Rankings

4.4.2 Methods' perceived suitability with respect to factors

Figure 4-8 a through d depicts the suitability of each one of the seven PM methods to handle four of the 14 factors. In view of our analysis above, factors 12 and 3 were not examined for DSM. The four charts illustrate the respondents' views on how each one of these four factors is accounted for, or "covered," by each method. The scores

reflect the strengths and weaknesses of each PM method with respect to the pertinent factor, as perceived by the research participants.

EVM scored highest for factor 1 – budget/schedule measurement tracking, and factor 2 – budget/schedule forecasting (see Figure 4-8 a). Figure 4-8 b shows that for factor 3 – inter-relationships, OPM scored highest, while EVM and CPM gained the lowest scores for this factor. For factor 11 – iterations management (see Figure 4-8 c), the DSM method scored the highest. Factors 6 – performance quality, 7 – product quality, 8 – product planning, 9 – product measurement/tracking, and 11 – iterations management, scored low in most methods in comparison to other factors. Figure 4-8 d, which represents the suitability of each method to handle factor 13 – ease of communication, shows that the Gantt method scored the highest for this factor, followed by OPM.

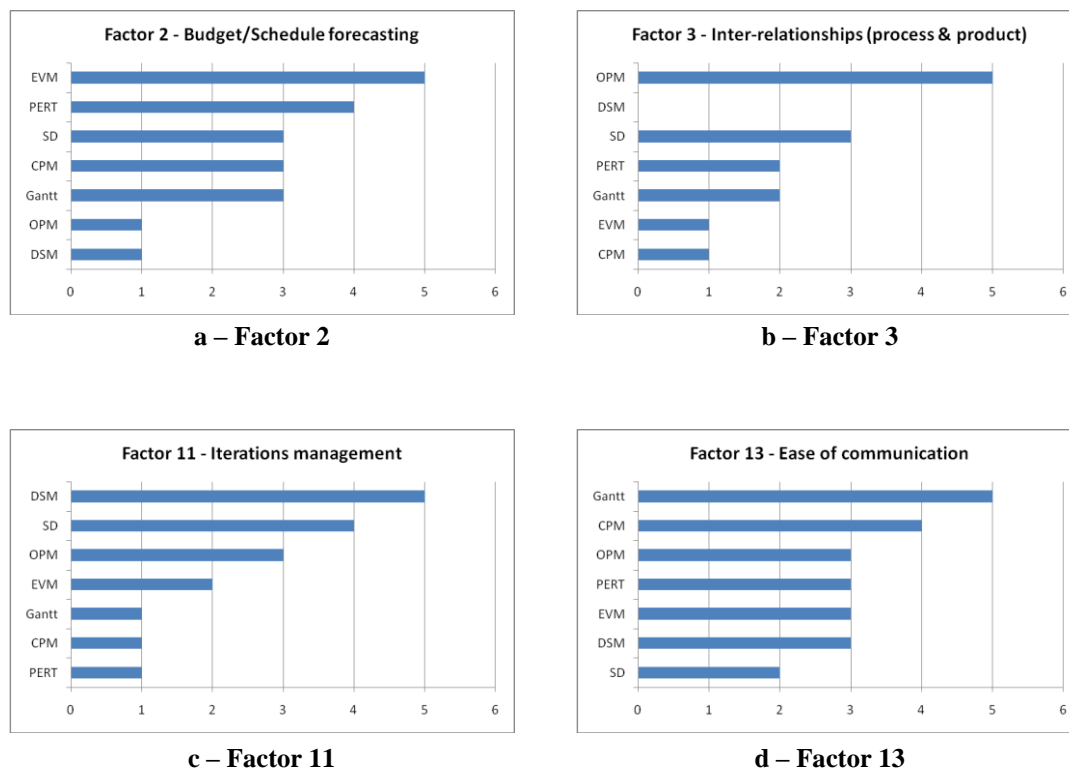


Figure 4-8 The suitability of each one of the seven PM methods to handle four of the 14 factors

4.4.3 The project dimension

Alpha Cronbach coefficient for each PM method based on the participants' rankings for the four factors, 1, 2, 4, and 11, of the project latent domain is presented in Table 4-5. Since we evaluated project management methods, we expected an Alpha Cronbach coefficient with value of .70 or higher to be obtained for all the seven methods, indicating that the factors in the underlying project domain are handled by all the seven PM methods. Surprisingly, however, as Table 4-5 shows, such above-the-cutoff values were found only for four PM methods: SD, PERT, DSM, and OPM. The remaining three methods, namely CPM, EVM, and Gantt, did not pass the 0.70 Alpha Cronbach Coefficient cutoff acceptance value. Even for the best improved result, which was obtained by removing factor 11 – iterations management, these three methods still remain below the 0.70 cutoff value (see bottom line of Table 4-5).

Table 4-5 The Project Dimension (factors 1, 2, 4, and 11)

Project Management Method	SD	PERT	CPM	DSM	EVM	Gantt	OPM
Full name	System Dynamics	Program Evaluation and Reviewing Technique	Critical Path Method	Design Structure Matrix	Earned Value Method	Gantt Chart	Object Process Methodology
Cronbach's Alpha	.738	.700	.422	.744	.505	.511	.731
Best Improved	-	-	.608 ⁽¹⁾	-	.512 ⁽¹⁾	.613 ⁽¹⁾	-

(1) Improved by deletion of factor 11 – Iterations Management

4.4.4 The product dimension

Applying a similar analysis for the product domain, we calculated the Alpha Cronbach coefficient for the four factors 6, 7, 8, and 9 of the product latent domain for each project management method. As Table 4-6 shows, the product dimension was found for three methods: SD, OPM, and DSM. The latter got in as having a product dimension only after removing factor 8 – product planning.

Table 4-6 The Product Dimension (factors 6, 7, 8, and 9)

Project management method	SD	PERT	CPM	DSM	EVM	Gantt	OPM
Full name	System Dynamics	Program Evaluation and Reviewing Technique	Critical Path Method	Design Structure Matrix	Earned Value Method	Gantt Chart	Object Process Methodology
Alpha Cronbach	.775	.472	.402	-.343 ⁽¹⁾	.414	.655	.746
Best improved	-	.486 ⁽³⁾	.601 ⁽²⁾	.725 ⁽²⁾	.560 ⁽⁴⁾	.678 ⁽⁵⁾	-

(1) The value is negative due to a negative average covariance among items.

(2) Improved by deletion of factor 8 – Product Planning

(3) Improved by deletion of factor 7 – Product Quality

(4) Improved by deletion of factor 9 – Product measurement/tracking

(5) Improved by deletion of factor 6 – Performance Quality

4.4.5 The project-product dimension

The project-product dimension was found to characterize only two methods: EVM and OPM (see Table 4-7). The results reflect the participants' perception that only these two methods have an underlying project-product latent dimension.

Table 4-7 The Project-Product Dimension (factors 3, 5, 10, 12, 13, and 14)

Project Management Method	SD	PERT	CPM	DSM	EVM	Gantt	OPM
---------------------------	----	------	-----	-----	-----	-------	-----

Full name	System Dynamics	Program Evaluation and Reviewing Technique	Critical Path Method	Design Structure Matrix	Earned Value Method	Gantt Chart	Object Process Methodology
Cronbach's Alpha	.251	.535	.624	.523	.723	.605	.706
Best Improved	.435 ⁽¹⁾		.687 ⁽¹⁾	.579 ⁽¹⁾	-	.651 ⁽¹⁾	-

(1) Improved by deletion of factor 5 – Stakeholders/ agents tracking

4.4.6 The combined project-product dimension

In view of the small number of methods found for the project-product dimension, we also examined the combined project-product domain, namely the combination of four project factors 1, 2, 4 and 11 with the four product factors 6, 7, 8, and 9. While the "original" project-product domain is based on six dual-domain factors, the latter is a combination of eight factors of which four are "purely" from the project domain and four—from the product domain. Although no such underlying dimension was predefined for the survey, the participants' rankings that yields Alpha Cronbach of 0.70 or higher might potentially reflect that the project and product dimensions are cognitively inseparable.

Table 4-8 The Combined Project-Product Dimension (factors 1, 2, 4, and 11 combined with factors 6, 7, 8, and 9)

Project Management Method	SD	PERT	CPM	DSM	EVM	Gantt	OPM
Full name	System Dynamics	Program Evaluation and Reviewing Technique	Critical Path Method	Design Structure Matrix	Earned Value Method	Gantt Chart	Object Process Methodology
Cronbach's Alpha	.807	.655	.565	.703	.524	.451	.826
Best Improved	-	.734 ⁽¹⁾	.690 ⁽¹⁾	-	-	.570 ⁽²⁾	-

(1) Improved by deletion of factor 8 – Product Planning

(2) Improved by deletion of factor 2 – Budget/Schedule forecasting

The combined project-product dimension, presented in Table 4-8, was found for OPM, SD, DSM, and PERT. The latter became acceptable after elimination of factor 8 – product planning, even though for the product dimension it had not reached the cutoff value.

4.4.7 Methods comparison by dimension

Comparison of the seven PM methods by dimension can be conducted with the methods for which all dimensions were found – System Dynamics (SD), Design Structure Matrix (DSM), and Object Process Methodology (OPM). The comparison, presented in Table 4-9, is based on the sums of scores participants assigned to each factor for each project management method. Each sum was divided by the quantity of factors used for calculating that sum. The sum of all fourteen factors is not applicable for DSM since only by excluding factors 12 and factor 3 it passed the 0.70 threshold.

The same rationale was followed for calculating the sums for the project-product and the combined project-product factors. The sum of the product factors was calculated for all three methods without factor 8 – product planning, since DSM got in as having the product dimension only after removing this factor.

Table 4-9 Sums calculations for comparison of methods by dimensions

		OPM	SD	DSM
Sum of all 14 Factors	Total Sum	885	858	N/A
	Divided by 14	63.2	61.3	N/A
Sum without Factors 3 and 12	Total Sum	726	769	609
	Divided by 12	60.5	64.1	50.8
Sum of Project factors (1, 2, 4, and 11)	Total Sum	214	299	226
	Divided by 4	53.5	74.8	56.5
Sum of Product factors (6, 7, and 9; without 8)	Total Sum	122	181	87
	Divided by 3	40.7	60.3	29.0
Sum of Project-Product factors (5, 10, 13, and 14; Without 3 and 12)	Total Sum	267	242	226
	Divided by 4	66.8	60.5	56.5
Sum of Combined Project-Product factors (1, 2, 4, 11 and 6, 7, 9)	Total Sum	336	480	313
	Divided by 7	48.0	68.6	44.7

Based on the data presented in Table 4-9, Figure 4-9 shows for each of the three methods in Table 4-9 the sum of scores participants assigned to that method for each one of the three dimensions. The values on the vertical axis represent the normalized total sum, which is the total sum divided by number of factors (see Table 4-9), for each project management method.

The project dimension scores (darkest grey bars) are higher than the product dimension scores (lightest grey bars) for all the three PM methods. The scores of the combined project-product dimension (dark grey bars) are reasonably situated between the project and the product dimension scores. OPM scored the highest in three dimensions – project dimension, product dimension, and the combined project-product dimension. While for SD and DSM, the project-product dimension scores are higher than those of the combined project-product dimension, the result is reverse for OPM. This may indicate that the dimensions perception is more complex: for OPM, a project-product dimension is acceptable, but the separate project dimension and product dimension are not only acceptable, but also rank higher. For SD and DSM the perception of dimensions is reversed.

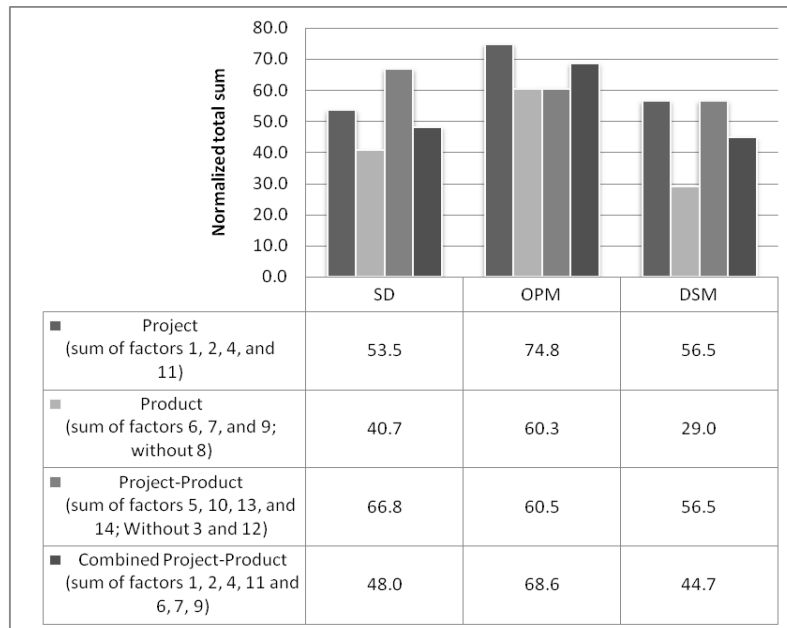


Figure 4-9 Project management methods comparison by dimensions

4.5 Discussion

Based on their practice and experience, practitioners tend to use the six traditional project management methods – SD, PERT, CPM, DESM, EVM, and Gantt chart – in practice for different purposes and in different contexts. Model-based project planning is still not in use, and OPM is not recognized yet as a means for constructing a model-based project plan. In order to conduct the presented research, there was a need to find a group of systems engineers who are familiar not only with the traditional PM methods, but also with OPM. In addition, because of the diversity of the SEM vocation and the wide range of practitioners' training and experience, it is very difficult, if at all possible, to find a group of systems engineers who are homogeneous in their knowledge and application of the traditional PM methods. Therefore, a reasonable research population for our purpose would be students at a graduate program, who are also practicing systems engineers, and are at about the same stage of their graduate studies at systems and management programs, and have equal knowledge about traditional PM methods, as well as knowledge about OPM. Such is our research group of 24 mid-career systems engineers studying in the Systems Design and Management graduate program at MIT. Furthermore, in the specific course in which the research was conducted, all the investigated project management methods were taught using the same system project case study—an Unmanned Aerial Vehicle—as the basis for all the assignments. This enabled us to assume identical knowledge of, and training level in, the examined methods, as well as non system-specific bias. Due to the required profile of the students in this program, the participants are not only students, but also practicing systems engineers in companies across the USA with 5-8 years of practice. The students were encouraged to reflect and provide criticism, and so they did. The majority of students claimed that using OPM was not easy and that the PPLM approach is not yet mature in their minds. Nevertheless, the results indicate that when the participants were required to rank the PM methods, in regard to the given set of factors, their judgment reflected more subtle attitude towards the PPLM. Furthermore, by examining their reflections, we learned that the majority of participants though the duration of performing the tasks of

modelling using Gantt and OPM, was the researched parameter, since they were asked to record the durations. Given all these, we exclude potential bias of the results and consider the results to be reflecting the systems engineering management practice in a larger context.

The set of factors have we used for comparison of the PM methods, was found to be reliable in regard to the seven methods. Therefore we believe it can be used, at least as preliminary set for furthermore refinement, for future research of PM methods in the context of SEM.

Table 4-10 Summary of methods comparison findings

Project Management Method	SD	PERT	CPM	DSM	EVM	Gantt	OPM
Full name	System Dynamics	Program Evaluation and Reviewing Technique	Critical Path Method	Design Structure Matrix	Earned Value Method		Object Process Methodology
Pass of 75% cutoff for all 14 factors	✓	-	-	N/A	✓	-	✓
Pass of 75% cutoff for 12 factors	✓	-	-	✓	✓	-	✓
Project Dimension	✓	✓	-	✓	-	-	✓
Product Dimension	✓	-	-	✓	-	-	✓
Project-Product Dimension	-	-	-	-	✓	-	✓
Combined Project-Product	✓	✓	-	✓	-	-	✓

Table 4-10 presents the summary of all methods comparison findings. The cases where the latent dimensions were found acceptable indicate that the participants' rankings of the given factors reflect their perception of the factors as related to the defined dimensions. Although the participants were not instructed to group the factors in any specific way, they cognitively combined the factors in their minds in a way that three out of the seven examined methods—SD, DSM, and OPM—were found to handle well both the project dimension and the product dimension. The project-product dimension was also found acceptable for OPM, as well as for EVM. This dimension is composed of six factors which were defined as not directly associated with either the product alone or the project alone. For EVM, only the project-product dimension was found acceptable, but neither the project dimension nor the product dimension alone was found acceptable. Examining the elaborate responses participants provided along with their rankings, reveal that they consider EVM to be a project tracking method, while the other methods were perceived more suitable for project planning and less for progress tracking. While the three project management methods—SD, DSM, and OPM—have passed the 75% cutoff value for the sum

rankings, for the defined factors, EVM also scored high enough to pass the 75% threshold.

The "big" picture reflected by the results is that SD, DSM, EVM, and OPM were found to address SEM better than the other PM methods examined. These four methods can be considered as being project-product oriented rather than just project-oriented. Therefore, they are likely to have greater utility as methods for systems engineering management.

The fact that out of the four methods, only OPM was found suitable in all the three examined dimensions, is an indication that OPM might potentially become a common paradigm and language for communication among stakeholders and management of multidisciplinary teams of experts who are partners in the systems engineering management process [2].

4.6 Summary

This research has examined the suitability of seven project management (PM) methods for systems engineering management (SEM), as perceived by systems engineers, with respect to 14 factors. Focusing on application of PM methods within SEM, the seven examined methods contained six traditional project management methods – SD, PERT, CPM, DESM, EVM, and Gantt chart – the new model-based project planning using OPM which is the infrastructure for applying the Project-Product Lifecycle Management (PPLM) for SEM. Since SEM is about handling and solving problems associated with the intricate relationships of the product with the project that delivers it, we classified the 14 factors into three domains: the project domain, the product domain, and a holistic project-product ensemble domain. Our research population, a group of 24 mid-career systems engineers studying in the Systems Design and Management graduate program at MIT, ranked the adequacy of each one of the seven examined project management methods to tackle each one of the 14 factors. The set of fourteen factors was found reliable for comparison of six out of the seven examined project management methods. After excluding two factors, a set of 12 factors was reliably used for comparison of all seven PM methods.

Using the participants' rankings, the three predefined dimensions were analyzed using Alpha Cronbach coefficient to examine the extent to which the participants perceived the 14 factors as domain-related. The findings support the notion of the project and the product as being two complementary facets involved in systems engineering management. Four project management methods—SD, DSM, EVM, and OPM—were found more suitable than the others for use in systems engineering management. These four methods were found to address the defined domains better than the other examined methods.

OPM was found the most suitable method both by dimensions analysis and ranking comparison analysis. The results may imply that OPM should be favorably considered as a suitable method for managing product-project ensembles within systems engineering management. Applying the Project-Product Lifecycle Management (PPLM) methodology, it might become the actual bridge between systems engineering and project management, enabling the simultaneous expression of the function, structure and behavior of both the project and the product within a holistic integrated conceptual model.

5 Extraction of Coherent Project Views from the PPPLM Model-Based Plan⁵

Current SEM and project management (PM) utilize a host of planning methods and techniques, including the Critical Path Method (CPM), Gantt chart, Work Breakdown Structure (WBS), and DSM. These methods focus almost exclusively on activities and tasks to be performed, along with their relationships and durations, while the deliverables—the end product or system to be developed, their components constructed along the way, and associated enabling products—are mostly expressed only implicitly. Using our running example of the unmanned aerial vehicle, we review these PM approaches and discuss problems stemming from the lack of explicit representation of the product facet in current project management methods. Based on this observation, we propose the comprehensive product-project integrated model as the basis for the various SEM and PM views. Utilizing Object Process Methodology (OPM), the PPLM model integrates the project domain with the product domain via a shared ontology that explicitly relates project entities to product entities within a unifying frame of reference. This model is the exclusive source of the various enhanced PM representations. In their new PPLM-based versions, these representations are augmented with critical product-related information gleaned from the common underlying model, facilitating a shift from activities-based to deliverables-based project management. Compared with the traditional CPM, our deliverables-oriented method augments the critical path activities list with activities that yield deliverables required for initiating critical activities. Ignoring those additional "sub-critical" activities can delay a project in spite of closely monitoring its CPM activities.

5.1 Researched PM Methods and Views

We focus on four widely used PM methods: (1) Critical Path Method (CPM), which is based on the project's Activities Network Plan (ANP), (2) Gantt chart, (3) Work Breakdown Structure (WBS), and (4) Design Structure Matrix (DSM). These four methods, briefly reviewed in this paragraph, were developed over the last decades, and are still widely used with little changes, if any. For each method, we propose the model-based counterparts and demonstrate their advantages.

CPM was developed in the 1950's by DuPont for management of maintenance projects in industrial factories. CPM depicts activities⁶ along with their dependencies, duration, and slack time, showing which activities are critical to maintaining the schedule. Delay in an activity on the critical path delays the entire project. To accelerate the project, it is necessary to reduce the total time required for the activities along the critical path. CPM chart time is deterministic, resulting in a fixed estimate of the time required to complete the project.

The Gantt chart provides a graphical representation of horizontal bars for both planning and tracking of project schedule. In its current form, is probably the most

⁵ A self-contained version of this chapter was submitted to the Journal IEEE Transactions on Engineering Management.

⁶ While *activity* and *task* are commonly used as alternating terms, in this paper we use the definition of *task* as an *atomic, leaf-level activity*. In UML and SysML Activity Diagrams, a *task* is referred to as an *action*.

widely used method for project management. Originally, when the Gantt chart was invented roughly forty years before CPM, it showed the timing of tasks without specifying relationships. Only in the late 90's, constraining relationships between tasks, such as Start-to-Start and Finish-to-Start, were added, making it possible to view and understand the impact of delaying a single task on the project duration. In its latest form, the Gantt chart is used also for CPM-based project analyses.

The Work Breakdown Structure (WBS) was initially developed as a product-oriented family tree by the United States Department of Defense (DoD), which in 1968 issued the "Work Breakdown Structures for Defense Materiel Items" (MIL-STD-881). WBS captures the work content within a project's scope in a hierarchical structure. The development of the WBS model involves breaking the overall work involved in a project into progressively smaller pieces down to the level of "work packages". These work packages are the elements for which required resources, budget, and duration can be estimated with relative confidence. The WBS can be presented graphically as a hierarchical tree or as an indented list. It is usually accompanied with a dictionary describing each WBS element. The WBS is usually modeled in one of three approaches: product-oriented, process-oriented, or a hybrid approach. The WBS model is usually developed before identifying the dependencies between activities and estimating their durations. Therefore, it is often used to identify the tasks required in the Activities Network Plan for applying CPM.

The Design Structure Matrix (DSM) is used for modelling and analyzing dependencies among components, tasks, or teams in a project. Table 5-1 Different types of Dependency Structure Matrix (DSM) lists the three types of DSM that are in common use in the context of PM. Out of the three the Process-based DSM is the most popular one, used for capturing the sequence of and the technical relationships among design tasks, which are then analyzed in order to resolve design loops and iterations.

Table 5-1 Different types of Dependency Structure Matrix (DSM)

DSM Data Types	Representation	Application	Analysis Method
Process-based (Task-based, Activity-based)	Activity input/output relationships	Project planning	Sequencing & Partitioning
Organizational-based	Multi-team interface characteristics	Organizational design, interface management, team integration	Clustering
Component-based	Multi-component relationships	System architecting, engineering and design	Clustering

Using any of these methods, namely CPM, Gantt, WBS, or DSM, produces a project plan in which the product model is written all over, albeit not explicitly. Regardless of the way the systems engineering manager chooses to model and analyze the SEM plan, he or she must constantly and closely consult the product model, which is the ultimate project goal. Striving to reflect the product aspect in the project plan, task names often contain product components names, some of the milestones represent completion of product-related issues, and the order of activities in the plan follows the

technological order, in which the product evolves as the project activities are executed.

5.2 Activities Network Plan and Object & Activities Network Plan

The Activities Network Plan (ANP) view for our UAV project is generated based on the dependencies between processes in the OPM model-based project plan. We generate the Activity-on-Arrow form of ANP, where each process is an activity node. Based on the model, the predecessors of each activity are identified and automatically converted into adequately linked nodes while maintaining their names. Figure 5-1 presents such ANP for our case study, and Table 5-2 lists its 23 activities with their predecessors. Size limitations of real-life projects require breaking the top-level diagram into smaller, manageable parts, which, if not model-based, are not necessarily coherent with each other, or with the entire project network. The automatically-generated ANP is coherent and it enables zooming in or out as necessary in order to fully comprehend the entire network.

Assigning probabilistic durations to the processes in the OPM model-based project plan, followed by applying the OPCAT simulation mechanism to the project model, will provide statistical estimations of the project duration processes and their corresponding budget allocations.

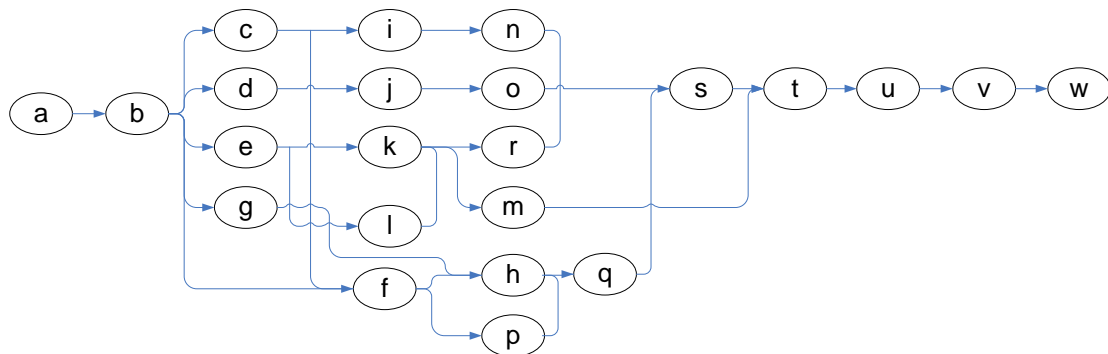


Figure 5-1 The Automatically Extracted Activities Network Plan

Table 5-2 List of 23 Activities of the UAV project

Activity	Description	Predecessors
a	Project start	-
b	Requirements Defining	a
c	Engine Specifying	b
d	Payload Specifying	b
e	Vehicle Layout Designing	b
f	Avionics Design Specifying	b, c, d
g	Software Specifying	b
h	Software Developing	g, f
i	Engine Developing	c
j	Payload Developing	d

Table 5-3 List of 23 Activities of the UAV project (continued)

Activity	Description	Predecessors
k	Fuselage Designing	e
l	Empennage/Wing Designing	e
m	Internal Fittings Designing	k, l
n	Delivery & Checkout	i
o	Power System Integrating	j, n
p	Avionics Delivery and Checkout	f
q	Avionics/Software Integrating	h, p
r	Airframe Prototyping	k, l
s	Vehicle Integrating	o, q, r
t	Final Vehicle Assembling	s, m
u	Laboratory Testing	t
v	Flight Test Campaigning	u
w	Project Finish	v

The standard ANP contains all the processes without data and material flow, but the OPM model enables automatic extraction of a richer version, Objects & Activities Network Plan (OANP). This is a PPLM view in which the processes and the objects are simultaneously displayed, along with their inputs and outputs. Figure 5-2 shows this OANP, where the thin arrows are the same as in Figure 5-1, while the thick arrows represent flow of objects from and to activities. The OANP is similar to the Enhanced Functional Flow Block Diagram (EFFBD) [54], where functions are OANP activities. Using an object hide/show software function, one can view the ANP as in Figure 5-1 or the OANP as in Figure 5-2.

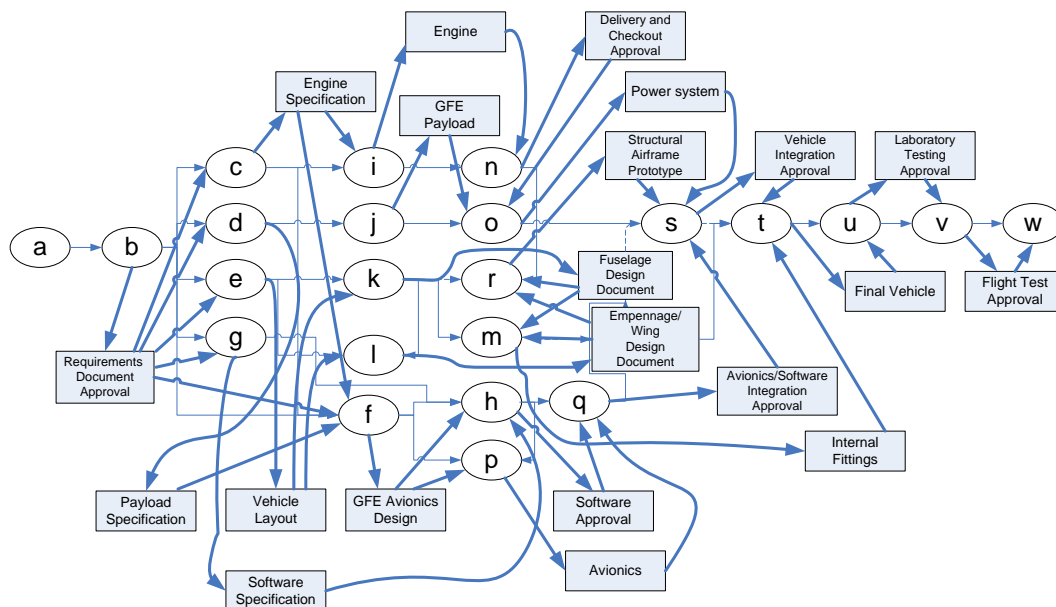


Figure 5-2 The Objects & Activities Network Plan (OANP) PPLM view of the network in Figure 5-1

5.3 CPM, Object-enhanced CPM, and Critical Objects

The Activities Network Plan (ANP) is the basis for CPM. For constructing the CPM diagram, Early Start (ES), Late Start (LS), Early Finish (EF), and Late Finish (LF) can be computed from process duration and dependencies in the OPM project model, and the critical path can be identified and presented in a standard CPM view. Figure 5-3 is a CPM diagram that can be produced automatically from the OPM project model. The diagram contains the activities network, the information about the timing at each node, and the critical paths – three in this case – denoted by thick lines.

Like the ANP, the standard CPM view lacks the objects. Automatic extraction of the CPM view from the OPM project model enables the creation of an Object-enhanced Critical Path (OCPM) PPLM view, in which the processes and objects are presented simultaneously.

Figure 5-4 shows the OCPM for the UAV. The processes on the critical path are denoted by solid ellipses, and the objects required as inputs to these processes are denoted by rectangles. This is an extremely valuable view for the project planner and manager, since it explicitly shows the deliverables which are critical for starting processes along the critical path.

Figure 5-4 shows that the **Software Specification** is required as input to the critical process **H (Software Developing)**, although it is produced by the non-critical process **G (Software Specification Writing)**. The slack of **G** is 8 weeks, which is less than 10% of the entire project duration (90 weeks). Although sub-critical, a delay of 8 weeks or more in producing the **Software Specification** will cause a delay of the critical process **H**.

As this example shows, the OCPM view is most helpful for understanding the potential impact of the outcomes of "innocent" non-critical processes. These processes are aimed at producing deliverables which are required as inputs to processes on the critical path. Planning a project based on the conventional critical path method ignores the potential of delay in the start of the **Software Developing** process, due to required artifacts which are produced by non-critical processes. In Figure 5-4 there are four more such deliverables, all of which are physical objects: (1) **Avionics** which is produced by the non-critical process **P (Avionics Delivery and Checkout)** and required as input to the critical process **Q (Avionics/Software Integrating)**, (2) **Power System** which is produced by the non-critical process **O (Power System Integration)** and required as input to the critical process **S (Vehicle Integrating)**, (3) **Structural Airframe Prototype** which is produced by the non-critical process **R (Airframe Prototyping)** and required as input to the critical process **S (Vehicle Integrating)**, and (4) **Internal Fittings** which is produced by the non-critical process **M (Internal Fittings Designing)** and required as input to the critical process **T (Final vehicle assembling)**. The slacks of these non-critical processes are 13, 17, 13, and 27 weeks, respectively. Setting 10% of the entire project duration (9 of 90 weeks) as a threshold eliminates the potential influence of these non-critical processes on the critical path as the slack of these processes is sufficiently large to exclude them from presenting a potential scheduling risk. A parametric scale of slacks thresholds can provide a more fine-tuned method for quantifying the potential influence of the non-critical processes on the critical path.

In general, whenever a process is a predecessor to a critical process, it contains a potential risk in case the outcome required by the critical process is delayed. Some of the processes that produce objects that are inputs to the processes along the conventional critical path may have a large slack so they are not likely to influence

the critical path. Other processes, whose slack is small, can be considered potential risks to the critical path. Such potentially risky processes can be used to determine the level of confidence of project on-time completion. This issue needs to be further researched. In future work we plan to propose a method for planning and controlling the project duration, which accounts for the contribution of sub-critical processes to the critical path.

By explicitly showing the deliverables which are inputs to critical processes, Figure 5-4 provides additional valuable information to the project planner and manager. For example, **Requirements Document Approval** is needed as input to three critical processes: C (**Engine Specifying**), D (**Payload Specifying**), and F (**Avionics Design Specifying**). It is also required as input to the two non-critical processes E (**Vehicle Layout Designing**) and G (**Software Specification Writing**). Provided with this information, the project planner can prioritize the parts of the Requirements Document required for processes C, D, and F over the parts required for processes E and G. There are two similar cases in Figure 5-4: (1) **Engine Specification**, which is required for the critical process F, as well as for the non-critical process I, and (2) **GFE Avionics Design**, which is required for the critical process H, as well as for the non-critical process P.

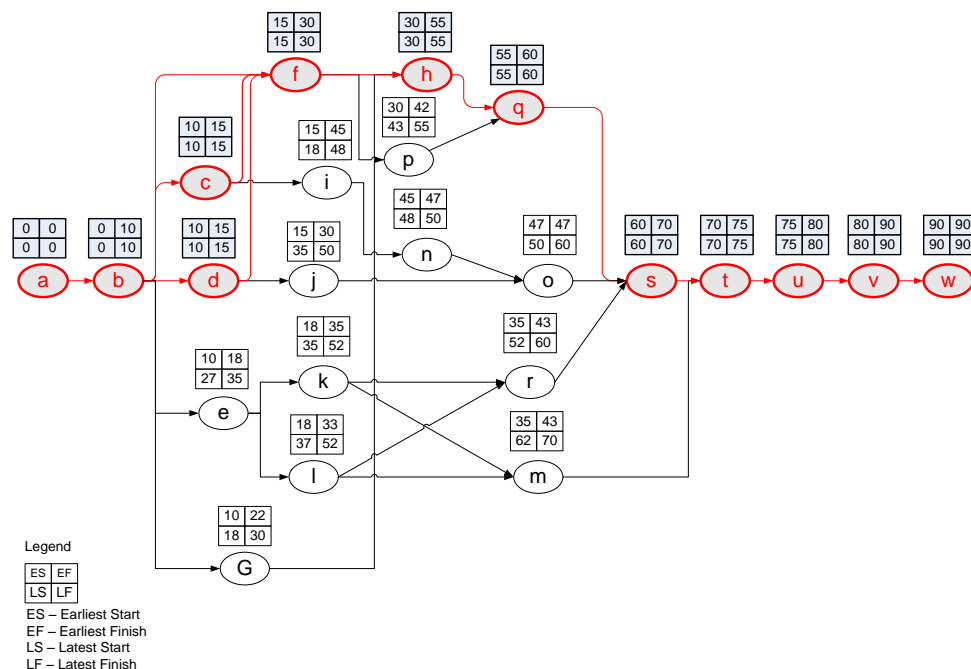


Figure 5-3 The automatically extracted Critical Path

The OCPM view enables the project manager to detect not just the critical path, but also the **critical objects**—the objects required as inputs to the processes along the critical path. Knowing what these objects are reduces the scheduling risk by focusing on managing them with high priority, adding value to the project management process. Rather than inquiring about the processes designed to deliver the critical objects, the project manager can monitor the status of the critical objects themselves. These critical objects may include documents, approvals, simulation outcomes, analyses, specifications, and reports.

This capability, in turn, provides the project manager with the flexibility of treating the processes not as an end in itself, but rather as vehicles for obtaining the critical objects. This way, processes are managed to *achieve the critical objects rather than*

complete the critical tasks. In other words, the ability to describe the critical path in terms of objects (products), rather than in terms of the processes that yield them, is highly valuable for the project manager and the systems engineering manager, since it directs them to focus on precisely those activities that advance the project towards timely completion of the required deliverables.

Viewing the Model-Based Critical Path (MBCP) is easier in the OPM project plan views, enabling planning and tracking of the combined Process-Object critical path through the entire project plan hierarchy. These views for the UAV project, presented in Figure 5-5 through Figure 5-8, enable better understanding of the critical path in terms of outcomes – the project deliverables, rather than processes.

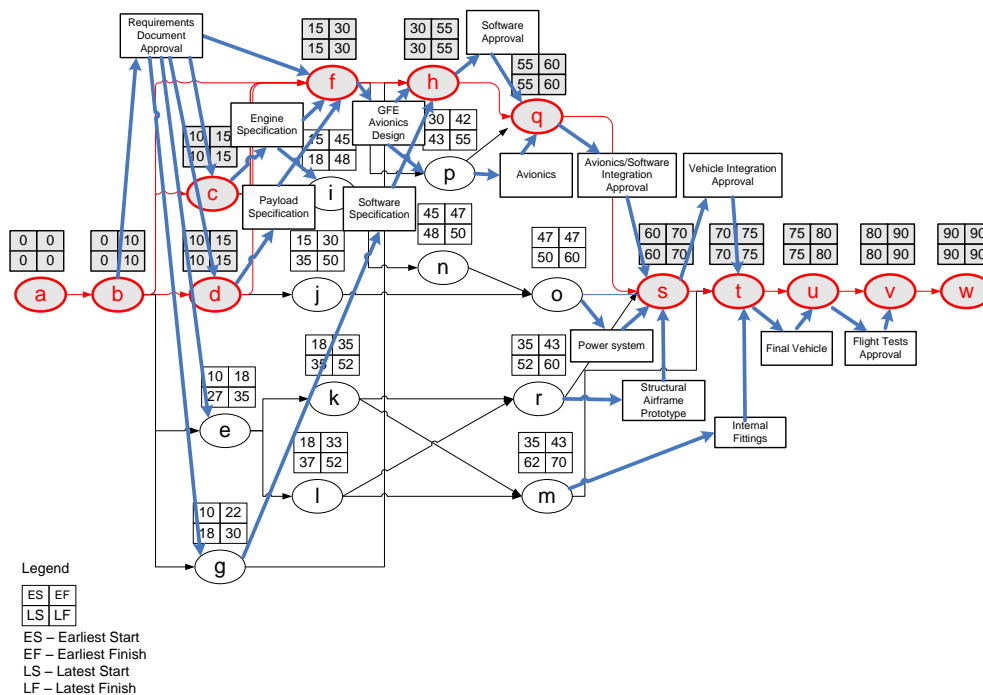


Figure 5-4 The Object-enhanced Critical Path Method (OCPM) PPLM view

In Figure 5-5, **Error! Reference source not found.** that specifies **UAV Prototype Developing & Integrating**. The processes on the critical path are denoted by solid ellipses. Whenever a "parent" process contains at least one subprocess that is on the critical path, it is denoted as critical in the OPD in which it appears. The **UAV Specifying** process appears as a solid ellipse due to fact that 4 out of its 6 subordinate processes are on the critical path (see Figure 5-6). The **Integrating & Testing** also appears as a solid ellipse. In this case, all the five subprocesses are on the critical path. Whenever a deliverable is required as input to at least one process which is on the critical path, it is defined as a critical object. All the critical objects are denoted by solid rectangles.

Figure 5-7 shows that the two physical objects **Internal fittings** and **Structural Airframe prototype** are critical. This is due to the fact they are required as instruments for **Integrating & Testing**, which is defined as critical. This is an important observation, since these two artifacts are outcomes of the **Vehicle Developing** process, which in itself is not critical (see Figure 5-8). Figure 5-8 shows the **Vehicle Developing** process, which, although non-critical, produces the **Internal fittings** and the **Structural Airframe prototype**, both critical since they are required for critical processes. This view is most helpful for understanding the true potential impact of the outcomes of the "innocent"

non-critical **vehicle developing** process, aimed at producing these two specific objects, according to the project plan.

A similar case is the physical **Power System** (see Figure 5-5), which is also required for the **Integrating & Testing** process, and is one of the outcomes of the non-critical **Payload & Engine Developing** process. Figure 5-7 shows the **Payload and Engine Developing** process, which, although non-critical, produces the critical **Power System**. In addition, it shows that although the **Payload & Engine Developing** process in non-critical it requires two critical inputs – **Engine Specification** and **Payload Specification**. They are critical due to the fact that they are required as inputs to critical process under the **UAV Specifying** process (see Figure 5-6). A delay in delivering these objects can affect the start of the **Payload & Engine Developing** process as well. Seemingly, it is not important, because this process is not critical, but knowing that it produces the critical **Power System**, the project planner should be aware of its potential to influence the project duration.

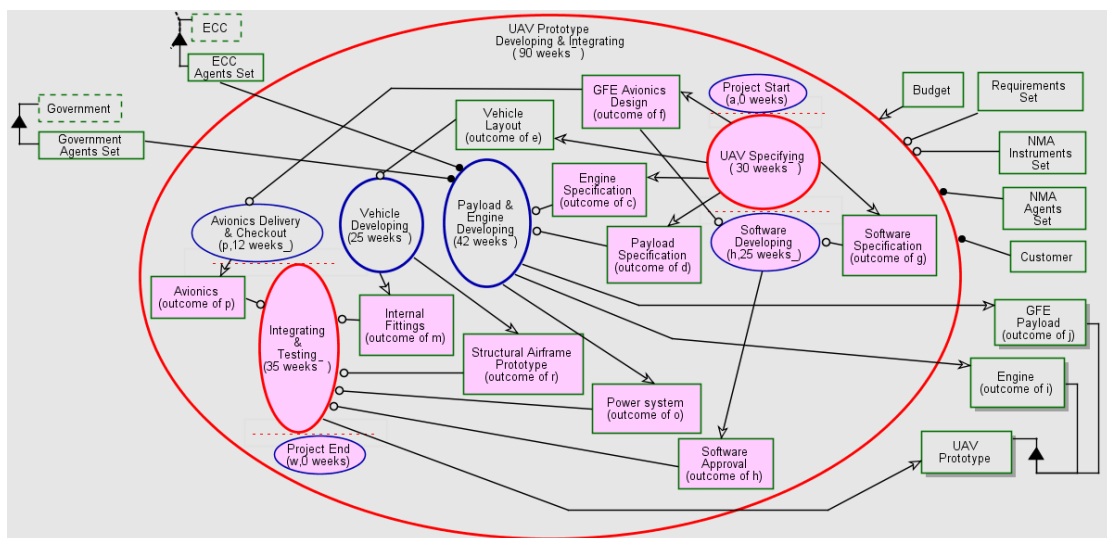


Figure 5-5 The Critical Path Depicted in the OPM Model – OPD of UAV Prototype Developing & Integrating

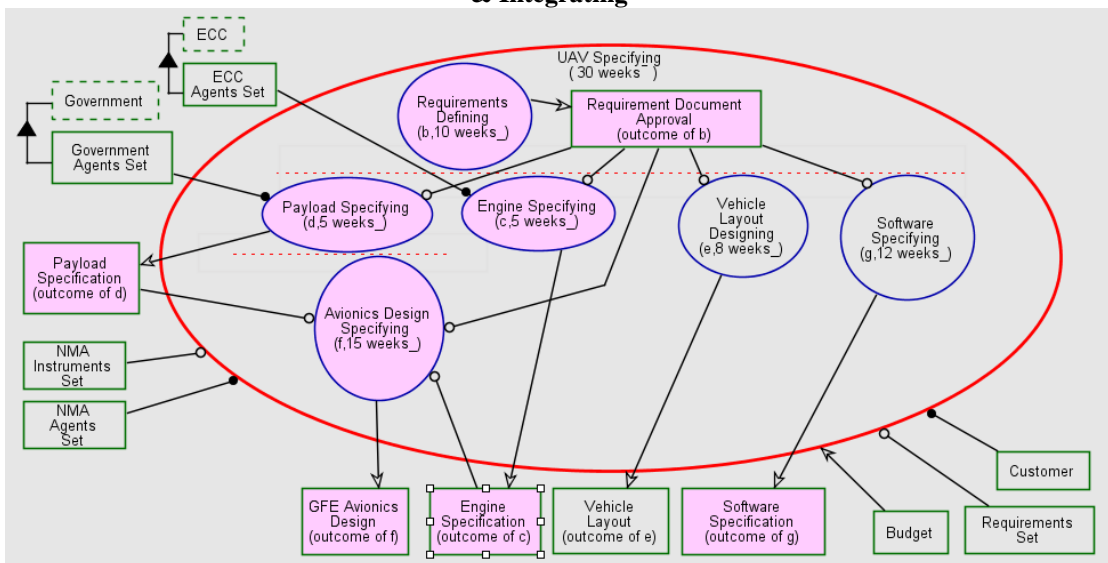


Figure 5-6 The Critical Path Depicted in the OPM Model – OPD of UAV Specifying

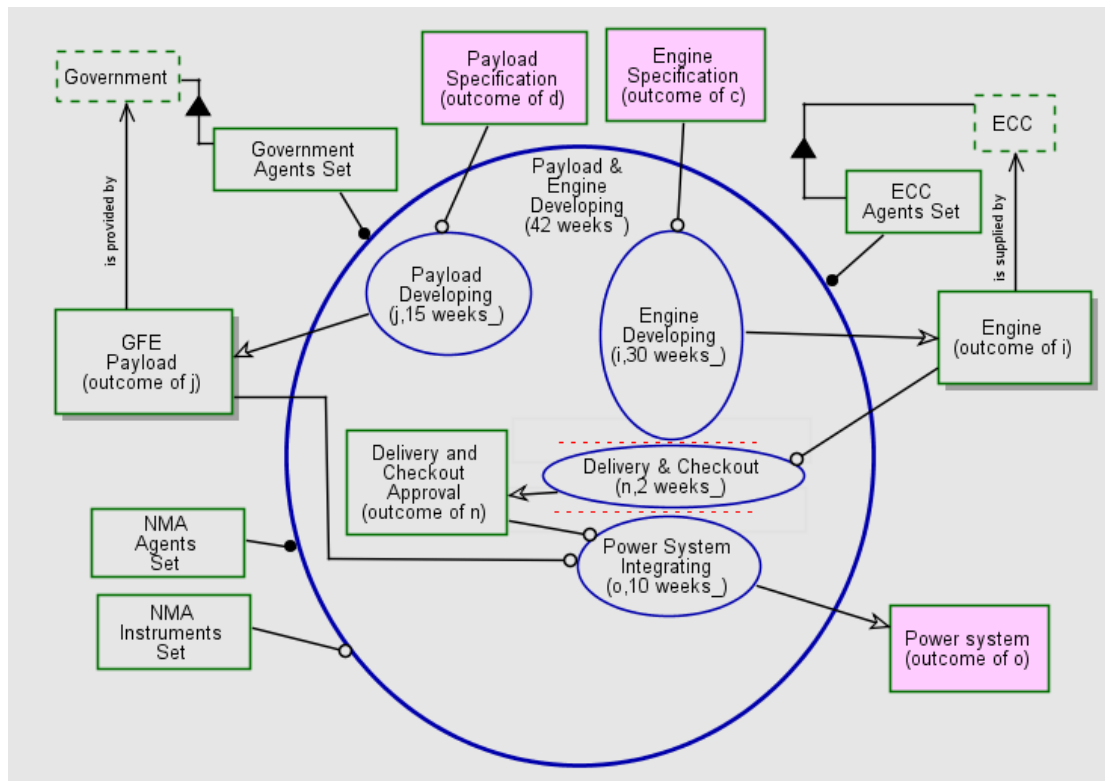


Figure 5-7 The Critical Path Depicted in the OPM Model – OPM of Payload & Engine Developing

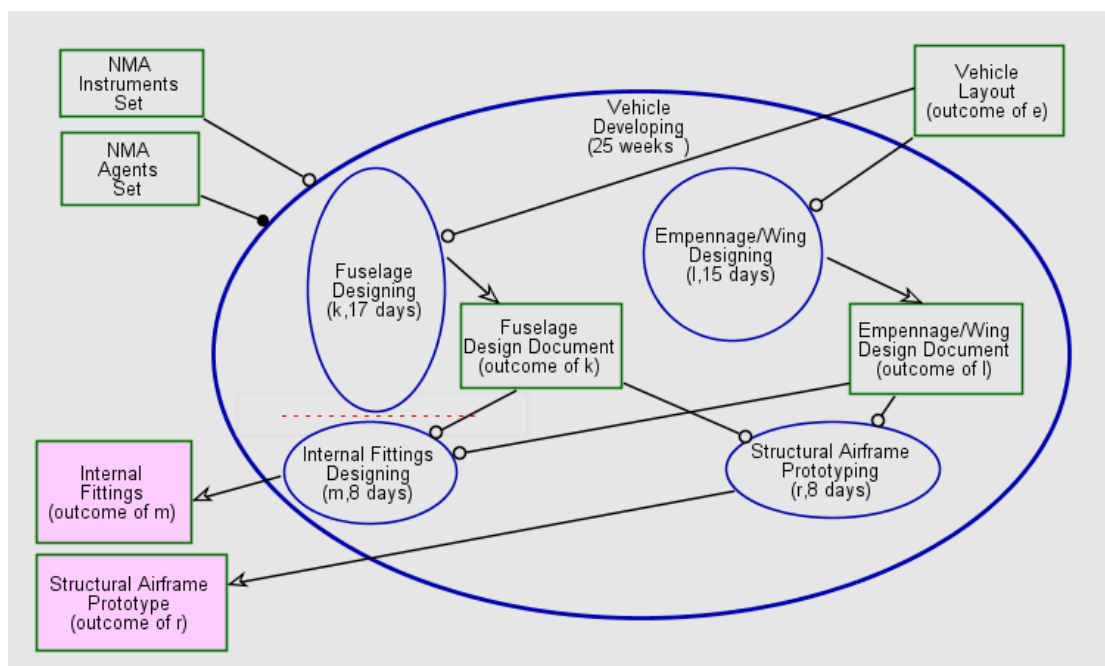


Figure 5-8 The Critical Path Depicted in the OPM Model – OPD of Vehicle Developing

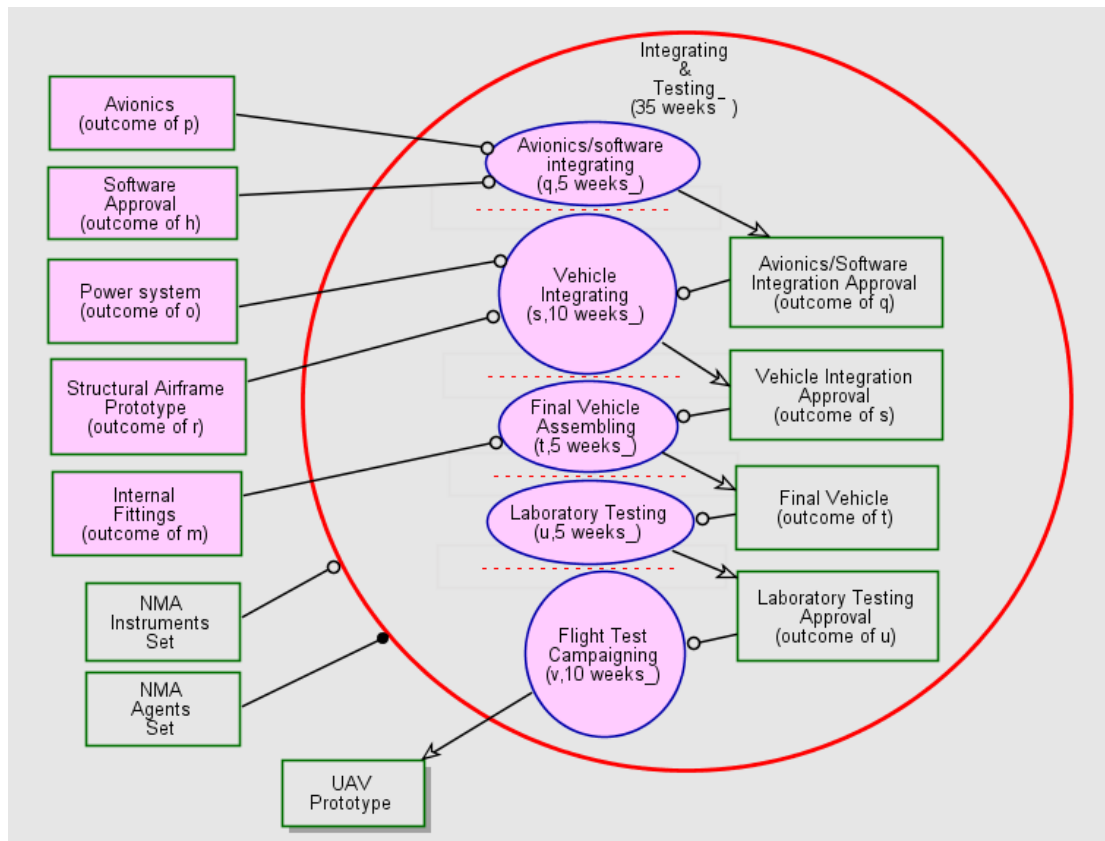


Figure 5-9 The Critical Path Depicted in the OPM Model – OPD of Integrating & Testing

5.4 Gantt chart and Model-Based Gantt chart

A Gantt chart containing all the processes in their hierarchical structure can be automatically generated from the OPM project model. Figure 5-10 presents the Gantt PPLM view that can be generated automatically from the OPM model, containing the entire project plan hierarchy. Each atomic OPM process is a task in the Gantt chart. This Gantt chart contains all the 23 project tasks along with their dependencies and durations. The OPM vertical time line is horizontal in the Gantt chart, advancing from left to right. All the process predecessors are identified and automatically converted into adequately linked tasks while maintaining their names and relationships as identified in Table 5-2, with supplements for handling imposed objects relationships where objects are modeled between processes. Each hammock is derived from a parent node process in the OPM model, representing a logical activity cluster, whose OPM model name appears to the left of the hammock bar.

5.5 Milestones Breakdown Structure

Some project planning software packages (e.g., MS-Project) treat a milestone as a zero-duration task. However, semantically, a milestone is an event—a point in time at which some important happening in the project system occurs. To remedy this semantic incompatibility, we define a new construct, the milestone construct, which consists of a process and one or more of its deliverables having the role of a milestone. An example of a milestone role is depicted in Figure 5-11, where the milestone construct includes the process **Processing** and its result **Outcome 1**.

The decision whether a specific process and its outcome constitute a milestone construct is up to the project planner to decide. If it is a milestone, a milestone role shall be recorded (as usual, at the upper left corner of the object). This notation indicates that a milestone needs to be recorded in the Gantt chart PPLM view. Similarly, a milestone role is generated automatically when a Gantt chart is transformed into an OPM PPLM model.

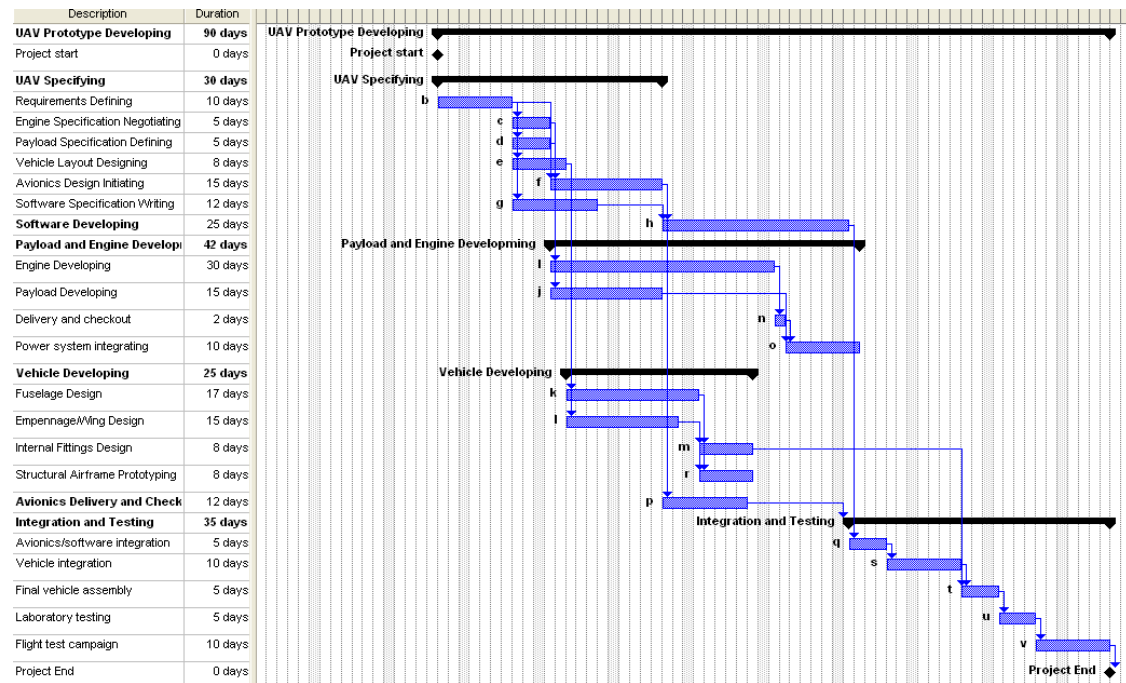


Figure 5-10 The Gantt PPLM view

The milestone role serves also for automatically generating a novel view, which we call. This view presents the hierarchy of all the milestones planned within the project plan, each with the process delivering it. MBS enables the project planner to clearly view and review the exact planned milestones at any hierarchy level of the planned processes of the project. Figure 5-12 presents a four-level MBS for the UAV project, where 15 outcomes of the tasks delivering them were defined as milestone constructs. At each milestone level, milestone deliverables are presented. Beneath each deliverable is the process for which the milestone deliverable is required. Each milestone deliverable is placed at the level in which it can be first achieved according to the PPLM model. The time in weeks denoted in Figure 5-12 on the horizontal dashed line at each level is the time from the project start required to achieve all the milestones at the respective level. It is computed by adding the maximal time required to achieve any milestone at that level to the time required to get to that level. For example, 22 weeks are needed to achieve all the 2nd level milestones: 10 for the first level plus 12—the maximal process duration of 2nd level processes. This display can be fine-tuned to present the time for each milestone separately.

Looking at the MBS PPLM view, which can be extracted automatically, the project manager can trace the hierarchical milestones flow induced in the model by the defined deliverables, which now becomes explicit and supports verification of the planning rationale. He or she can also explore the different types of deliverables in accordance with Table 3-1, to determine gates, documents, and components.

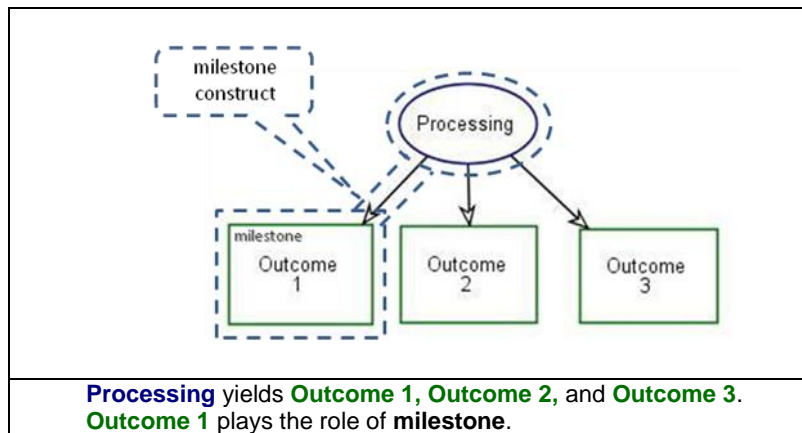


Figure 5-11 The milestone construct

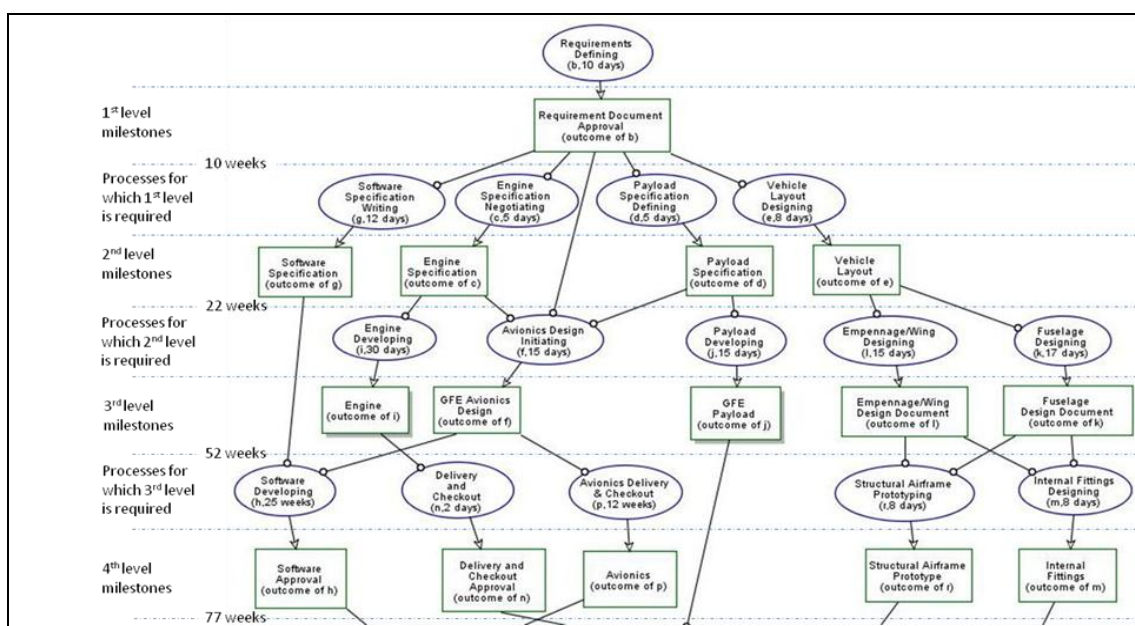


Figure 5-12 First four MBS levels of the UAV project

5.6 Process WBS and Model-Based WBS

Figure 5-14 and Figure 5-14 show automatically extracted process WBS diagrams for two intermediate-level processes using the OPM unfolding mechanism. The first process WBS is for the **UAV Specifying** process (Figure 5-14), and the second—for the **Integrating & Testing** (Figure 5-14). They look like common hierarchical WBS views, except that the nodes, which are OPM processes, are denoted by ellipses instead of the commonly used rectangles. In both WBS views, each ellipse represents a process in the hierarchy and it contains information regarding the process duration, allocated resources, and budget, if inserted into the model. The resources (not shown in these views for simplicity) are denoted by rectangles—objects that are related to the various processes, as members of the **Agents Set** for human resources or of the **Instruments Set** for all other types of resources, except for budget, which has its own dedicated object, as argued earlier.

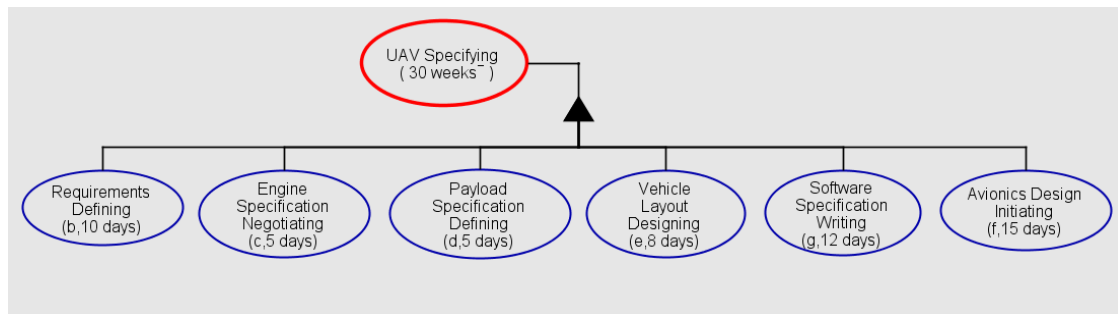


Figure 5-13 Automatically extracted process WBS for UAV Specifying

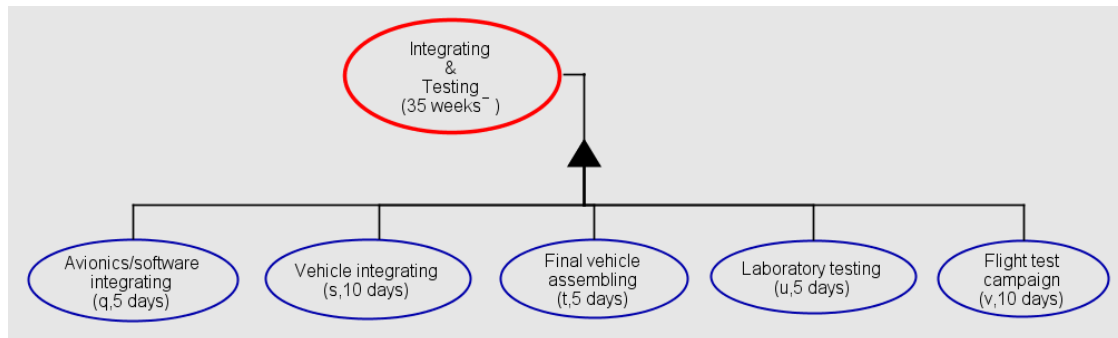


Figure 5-14 Automatically extracted process WBS for Integrating & Testing

While the process WBS may be sufficient for classical project management, systems engineering management has to plan and control the correspondence between the project processes and the product deliverables. To this end, we suggest a new PPLM view, Object WBS (OWBS), which adds the involved objects (deliverables) to the processes hierarchy of the conventional process WBS views. A partial OWBS view is presented in Figure 5-15 for the same **UAV Specifying** process shown in Figure 5-14. A complete OWBS view is presented in Figure 5-16 for the same **Integrating & Testing** process shown in Figure 5-14. Both Figure 5-15 and Figure 5-16 contain only two levels since they were produced for nodes which are one level up from the task level, the lowest (leaf) level in a PPLM model.

The OWBS view is constructed according to the following layout. The node for which the OWBS is extracted is placed at the top – this is the parent process, the top-level process in this hierarchy. *Internal processes* are processes that are contained in the in-zoomed parent process or are its parts. These internal processes are placed horizontally beneath the parent process and are connected with the parent by an aggregation-participation link (the solid triangle). Beneath each one of these internal processes are the exclusively internal deliverables of that process, along with their links to the process. An *exclusively internal deliverable* is a deliverable that is required only between internal processes. All the other deliverables are external—they are (also or only) required by processes external to the OWBS. External deliverables are placed at the bottom of the OWBS view.

A deliverable of a OWBS can be a *mixed deliverable*—a deliverable to at least two processes, of which at least one is internal and another—external. Like internal deliverables, mixed deliverables are solid, but they are placed at the bottom of the OWBS view, along with the external deliverables. We demonstrate this by the six deliverables produced by subprocesses of the **UAV Specifying** node in the OWBS at the top of Figure 5-15: **Requirements Document Approval**, **Vehicle Layout**, **Software Specification**, **GFE Avionics Design**, **Payload Specification**, and **Engine Specification**.

Requirements Document Approval is an internal deliverable, as it is generated and required only by internal processes—subprocesses of **UAV Specifying**. **Requirements Document Approval** is therefore marked by a solid rectangle and placed at the first level under the node's subprocesses. **Engine Specification** and **Payload Specification** are mixed deliverables; they are required by internal processes, but based on the PPLM model, they are also inputs to external processes. Being mixed variables, **Engine Specification** and **Payload Specification** are solid rectangles, but they are placed at the bottom of the OWBS view together with the remaining three deliverables, **Vehicle Layout**, **Software Specification**, and **GFE Avionics Design**, which are external, as indicated by their blank rectangles.

The OWBS view is valuable for both the project manager and the systems engineering manager, as it provides a clear picture of the deliverables and how they flow amongst the planned WBS process nodes; it enables the comprehension of dependencies between nodes through the WBS decomposition in terms of deliverables.

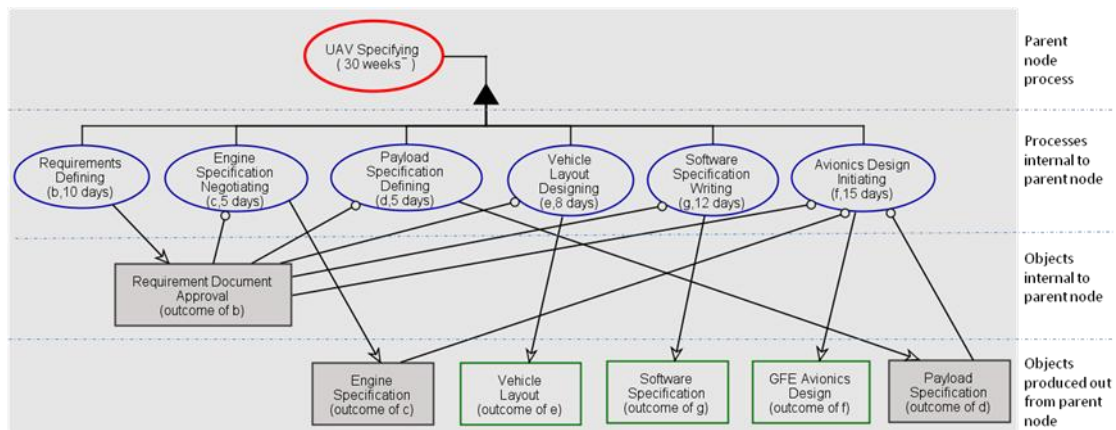


Figure 5-15 An automatically extracted partial OWBS for UAV Specifying

While Figure 5-15 shows a partial OWBS view, Figure 5-16 presents a complete OWBS view for the **Integrating & Testing** node. This complete OWBS view shows, in addition to what is shown in the partial OWBS view, also five objects: (1) **Internal Fittings**, which results from the **Internal Fittings Designing** process, (2) **Avionics**, which results from the **Avionics Delivery & Checkout** process, (3) **Structural Airframe Prototype**, which results from the **Structural Airframe Prototyping** process, (4) **Power System**, which results from the **Power System Integrating** process, and (5) **Software Approval**, which is accomplished at the end of the **Software Developing** process. These objects constitute the *preprocess object set* of **Integrating & Testing**, the parent OWBS node. They are required for executing this process. These objects, along with the processes that yield them, are within the dash-dotted area denoted by “Feeding to parent node subprocesses” at the top left of Figure 5-16. These five deliverables are the same five inputs required for **Integrating & Testing** in all the other views extracted from the PPLM model. For each object in the preprocess object set, Figure 5-16 also shows the subprocess for which that object is an instrument.

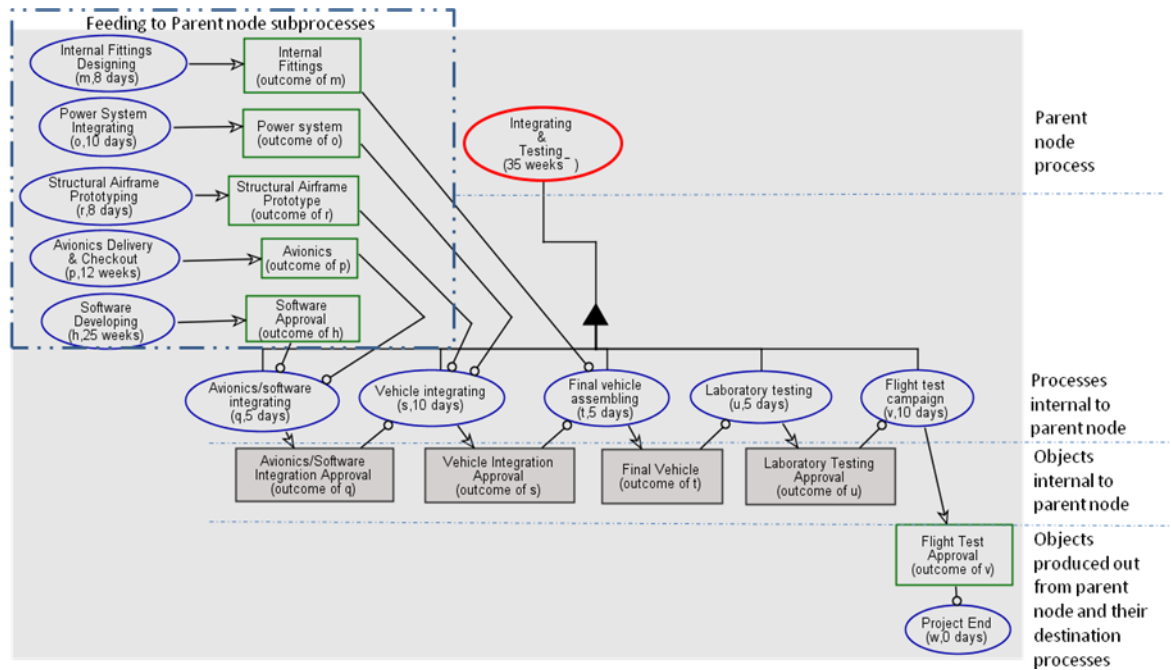


Figure 5-16 An automatically extracted OWBS for Integrating & Testing

5.7 Process-Based DSM and Object-Process Model-Based DSM

The three different types of DSM presented in Table 5-1 can be automatically generated from the OPM model. In this section we focus on Process-based DSM, which can be obtained from the relationships defined in the OPM model. We will examine the generated DSM, identify blocks of coupled activities, iterations and re-sequencing product development tasks. Since the OPM model includes the project and product objects in addition to the processes, a combined Object-Process DSM (OPDSM) can be automatically generated, based on the relationships established in the OPM project-product model.

5.7.1 Constructing the Process-based DSM

Based on the OPM model, we construct the 23×23 process-based DSM, shown in Figure 5-17, for the project tasks (leaf-level processes). It includes the project tasks a through w (including the dummy tasks "start" and "finish"). $DSM[i, j] = X$ means that task i requires an informational object (deliverable or artifact) or physical object (part or assembly) from task j . All the entries in the matrix are below the diagonal. This means that there are no iterations in this project plan. In other words, downstream tasks depend only on upstream tasks and not on tasks that are further downstream.

The resulting DSM contains all the leaf-level processes in the first row and first column, as well as the triangle, as shown in Figure 5-17. The upper level processes, above the leaf-level ones, are termed clusters for further discussion.

Table 5-4 lists the leaf-level information extracted from the project model, which serves as the input to the process pairs' comparison for automatic generation of the DSM. Each leaf-level process is extracted along with its involved objects set and related processes – its parent process and invoking process, if it exists. The role is also extracted, but not presented in the table for the involved objects. Each pair of processes is examined to isolate the common objects involved internally. Each such object indicates a relationship in the corresponding DSM, as shown in Figure 5-17. If two processes are in the same cluster (have the parent process) the relationship can be viewed in a single OPD, otherwise, no single OPD presents the relationship. The project tasks are extracted from the OPM-based project plan, provided that no bi-directional *effect* links are modeled between objects and corresponding leaf-level processes; all the objects involved in leaf-level processes must be explicitly modeled as either belonging to the preprocess object set or to the post-process object set. The naming convention we have used makes it easier to follow object-task relationships, through the OPDs, OPLs, and extracted data tables.

That task **b** depends on task **a** is evident from the OPD of **UAV Prototype Developing & Integrating** (see Figure 5-5), where **Project Start** (a,0) invokes **Requirements Definition** (b,10). The dependency of tasks **c**, **d**, **e**, **f**, and **g** on task **b** is discerned from the OPD of **UAV Specifying** (see Figure 5-6), where all first require the **Requirement Document Approval**, which is the output of **Requirements Definition** (b,10). The dependency of task **f** on task **d** and task **c** is also explicit in this OPD.

The dependency of task **h** on task **f** is not explicit in any single OPD, as there is no OPD in which both **h** and **f** are explicitly linked. Here we make use of the convention of including the object's source process (in parentheses). Indeed, in Figure 5-5 we observe that **GFE Avionics Design** (Product of **f**) is required for **Software Development** (h,25), implying that task **h** depends on task **f**. In a similar manner we find the dependency of task **h** on task **g**, and the dependency of task **p** on task **f**. Both can be concluded from SD1 (Figure 5-5) thanks to the naming convention. The same logic is used for determining the dependency of task **j** on task **d**, which is not explicit in any single OPD, as there is no OPD in which both **j** and **d** are explicitly linked. making use of the naming convention we can observe in Figure 5-7 that **Payload Specification** (outcome of **d**) is required for the **Payload Development** (j,15), implying that task **j** depends on task **d**. In a similar manner we find the dependency of task **i** on task **c**. Figure 5-8 contains many implicit relationships among leaf-level processes, created by the involved items; all the *instrument* objects outside the **Integrating & Testing** process ellipse, which are required for the inner processes, are outcomes of leaf-level processes which are not included in this OPD, but can be followed using the naming convention. All these relationships are explicit in the extracted DSM (Figure 5-17).

Table 5-4 OPM to DSM Extraction Data

Leaf-level Process Name		Role	Parent Process (cluster)	Invoking Process	Involved Objects				
Short	Full				Pre-process				Post-process
					Consumables	Enablers	Instruments	Agents	Results
a	Project Start (a,0)	Dummy	UAV Prototype Developing & Integrating						
b	Requirements Defining (b,10)		UAV Specifying	Project Start (a,0)					Requirement Document Approval (outcome of b)
c	Engine Specifying (c,5)		UAV Specifying				Requirement Document Approval (outcome of b)	ECC Agents Set	Engine Specification (outcome of c)
d	Payload Specifying (d,5)		UAV Specifying				Requirement Document Approval (outcome of b)	Government Agents Set	Payload Specification (outcome of d)
e	Vehicle Layout Designing (e,8)		UAV Specifying				Requirement Document Approval (outcome of b)		Vehicle Layout (outcome of e)
f	Avionics Design Specifying (f,15)		UAV Specifying				Requirement Document Approval (outcome of b)		GFE Avionics Design (outcome of f)
							Engine Specification (outcome of c)		
							Payload Specification (outcome of d)		
g	Software Specifying (g,12)		UAV Specifying				Requirement Document Approval (outcome of b)		Software Specification (outcome of g)
h	Software Developing (h,25)		UAV Prototype Developing & Integrating				GFE Avionics Design (outcome of f)		Software Approval (outcome of h)
							Software Specification (outcome of g)		
i	Engine Developing (i,30)		Payload & Engine Developing				Engine Specification (outcome of c)		Engine (outcome of i)
j	Payload Developing (j,15)		Payload & Engine Developing				Payload Specification (outcome of d)		GFE Payload (outcome of j)
k	Fuselage Designing (k,17)		Vehicle Developing				Vehicle Layout (outcome of e)		Fuselage Design Document (outcome of k)
l	Empennage/Wing Designing (l,15)		Vehicle Developing				Vehicle Layout (outcome of e)		Empennage/Wing Design Document (outcome of l)
j	Payload Developing (j,15)		Payload & Engine Developing				Payload Specification (outcome of d)		GFE Payload (outcome of j)
k	Fuselage Designing (k,17)		Vehicle Developing				Vehicle Layout (outcome of e)		Fuselage Design Document (outcome of k)
l	Empennage/Wing Designing (l,15)		Vehicle Developing				Vehicle Layout (outcome of e)		Empennage/Wing Design Document (outcome of l)

Table 5-5 OPM to DSM Extraction Data (continued)

Leaf-level Process Name		Role	Parent Process (cluster)	Invoking Process	Involved Objects				
Short	Full				Pre-process				Post-process
					Consumables	Enablers	Instruments	Agents	Results
m	Internal Fittings Designing (m,8)		Vehicle Developing				Fuselage Design Document (outcome of k)		Internal Fittings (outcome of m)
							Empennage/Wing Design Document (outcome of l)		
n	Delivery and Checkout (n,2)		Payload & Engine Developing				Engine (outcome of i)		Delivery and Checkout Approval (outcome of n)
o	Power System Integrating (o,10)		Payload & Engine Developing				GFE Payload (outcome of j)		Power system (outcome of o)
							Delivery and Checkout Approval (outcome of n)		
p	Avionics Delivery &Checkout (p,12)		UAV Prototype Developing & Integrating				GFE Avionics Design (outcome of f)		Avionics (outcome of p)
q	Avionics/software integrating (q,5)		Integrating & Testing (SD1.2)				Software Approval (outcome of h)		Avionics/Software Integration Approval (outcome of q)
							Avionics (outcome of p)		
r	Structural Airframe Prototyping (r,8)		Vehicle Developing (Fuselage Design Document (outcome of k)		Structural Airframe Prototype (outcome of r)
							Empennage/Wing Design Document (outcome of l)		
s	Vehicle integrating (s,10)		Integrating& Testing				Power system (outcome of o)		Vehicle Integration Approval (outcome of s)
							Avionics/Software Integration Approval (outcome of q)		
							Structural Airframe Prototype (outcome of r)		
t	Final vehicle assembling (t,5)		Integrating & Testing				Internal Fittings (outcome of m)		Final Vehicle (outcome of t)
							Vehicle Integration Approval (outcome of s)		
u	Laboratory testing (u,5)		Integrating & Testing				Final Vehicle (outcome of t)		Laboratory Testing Approval (outcome of u)
v	Flight test campaigning (v,10)		Integrating & Testing				Laboratory Testing Approval (outcome of u)		Flight Test Approval (outcome of v)
w	Project End (w,0)	Dummy	UAV Prototype Developing & Integrating				Flight Test Approval (outcome of v)		Completed Prototype UAV

to From	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
a	a																						
b	X	b																					
c		X	c																				
d		X		d																			
e		X			e																		
f		X	X	X		f																	
g		X					g																
h						X	X	h															
i			X						i														
j				X						j													
k					X						k												
l					X							l											
m											X	X	m										
n									X					n									
o										X					X	o							
p						X										p							
q								X								X	q						
r											X	X						r					
s															X		X	X	s				
t													X						X	t			
u																				X	u		
v																					X	v	
w																						X	w

Figure 5-17 The Project Plan Activities DSM

5.7.2 Constructing the Model-based Object-Process DSM

Since the DSM is derived from the project-product OPM model, we can glean additional knowledge about task dependencies represented in the DSM and upgrade it to a model-based Object-Process DSM (OPDSM), according to the classification of object in the OPM-based PPLM model (Table 3-1) – D, G, or C. X is left as is whenever a process is invoked by another process with no intermediate object. For all the objects involved with leaf-level processes, the type is designated in the upper left corner of the object rectangle, using the role attribute.

Starting with the DSM in Figure 5-17 and replacing where possible X's by D, G, or C and the names of the intermediary objects, we derive the OPDSM in Figure 5-19. For example, "G_{Req. Doc. Approval}" is inserted in the cell (4, 2) of the OPDSM because, as shown in Figure 5-6, the informatical object **Requirements Document Approval** is the node of type object in the bipartite graph underlying the OPM model, which connects the process **Requirements Defining** to the process **Payload Specifying**.

Since the project model is simplified with a single outcome from each process, and with no states, the relationships found in each column are the same. This implies that there is a single object as an outcome, which is required for one or more following tasks.

Comparing the DSM in Figure 5-17 with the OPDSM in Figure 5-19, it is obvious that the latter provides more information and deeper understanding of the relationships among the project and product entities.

In medium-scale industrial projects, there may be more than a few dozen leaf-level processes, therefore a sliding view mechanism would be used to enable focusing on the OPDSM region which is most interesting for the project planner. Such a view is provided in Figure 5-18, where the focus is on the first seven processes of the project. This sliding OPDSM view does not include any C's, since it contains only the first seven tasks in the project, hence no components are delivered yet.

Task (from) \ Depends on task (to)	a	b	c	d	e	f	g
Project Start - a							
Requirements Definition - b	X						
Engine Specification Negotiation - c		G Req. Doc. Appro val					
Payload Specification Definition - d		G Req. Doc. Appro val					
Vehicle Layout Determination - e		G Req. Doc. Appro val					
Avionics Design Initiation - f		G Req. Doc. Appro val	D Eng. Spec.	D PL Spec.			
Software Specification Writing - g		G Req. Doc. Appro val					G

Figure 5-18 OPDSM Sliding View for the First Seven Processes

to from	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u
a	a																				
b	x	b																			
c		G Req. Doc.	c																		
d		G Req. Doc.		d																	
e		G Req. Doc.			e																
f		G Req. Doc.	D Eng. Spec.	D PL. Spec.		f															
g		G Req. Doc.					g														
h						D Av.Des.	D SWSpec	h													
i			D Eng. Spec.						i												
j				D PL. Spec.						j											
k					D Veh. Lay						k										
l					D Veh. Lay							l									
m											D Fus. DD	D Emp. DD	m								
n									C Eng.					n							
o										C PL					G Del	o					
p						D Av.Des.										p					
q								G SW Approva									C Avionics	q			
r											D Fus. DD	D Emp. DD						r			
s															C PS.		G Int Approva	C SAP	s		
t													C Int. Fit.						G Veh Int.	t	
u																				C FV	u
v																					G Lab Approva
w																					

Figure 5-19 The Project Plan OPDSM

5.7.3 Improving the OPM-based Project Model by Inspecting the DSM clustering

We can improve the OPM-based Project Model by inspecting the DSM clustering. First we need to include some task interdependencies in our project plan. Let us review them one by one, referring to the updated project model in Figure 5-20 through **Error! Reference source not found.**

- **Software Specifying** (g) and **Software Developing** (h) are actually iterative. This can be seen in is the new OPD of **UAV Prototype Developing** in-zoomed (Figure 5-20). States were added to **Software Specification** (outcome of g) so it can be either *initial* or *final*. The **Definition** process, which is the parent of **Software Specifying** (g,12), yields **Software Specification** (outcome of g), in its *initial* state. **Software Developing** (h,25) requires **Software Specifying** (outcome of g) in its *initial* state, and it yields *final* **Software Specification** (outcome of g).
- **Avionics Designing** (f) requires information from **Software Developing** (h) in terms of properly sizing the avionics bus and controllers for processing speed, memory and communications bandwidth. This addition can be seen in the new OPD of **UAV Prototype Developing** in-zoomed (Figure 5-20). States were added to the **GFE Avionics Design** (outcome of f) so it can be either *initial* or *final*. The **Defining** process, which is parent of **Avionics Design Initiating** (f,15), yields **GFE Avionics Design** (outcome of f), in its *initial* state. **Software Development** (h,25) requires **GFE Avionics Design** (outcome of f) in its *initial* state, and yields *final* **GFE Avionics Design** (outcome of f).
- **UAV Engine Developing** (i) and **Payload Developing** (j) require data from each other as they are concurrent and interdepnedent. This addition can be seen in the new OPD of **Payload & Engine Developing** in-zoomed (Figure 5-26); a bidirectional relationship tagged information exchange was added between **Payload Developing** (j,15) and **Engine Developing** (i,30).
- In previous vehicle projects there were structural misalignment problems between the fuselage, wing and empennage. In order to avoid this in the current UAV project, tasks k and l must feed each other with information. This is modeled in the OPD of **Vehicle Developing** in-zoomed (Figure 5-28); a bidirectional relationship tagged information exchange was added between **Fuselage Designing** (k,17) and **Empennage/Wing Designing** (l,15).
- Experience from other programs has shown that significant errors and non-conformances are often discovered during **Avionics/Software integrating** (q). This can require substantial rework from this task q to **Software Developing** (h). This is modeled in the new OPD of **UAV Prototype Developing** in-zoomed (Figure 5-20), where states were added to the **Software Approval** (outcome of h) so it can be either *conditional* or *final*. **Software Developing** (h,25) yields **Software Approval** (outcome of h), in its *conditional* state. **Integrating & Testing**, which is the parent process of **Avionics/Software Integrating** (q,5), requires **Software Approval** (outcome of h) in its *conditional* state, and yields *final* **Software Approval** (outcome of h).
- Integrating the vehicle s is never a single step process. Data from past projects indicates that adjustments must be made to **Power System Integrating** (o) due to problems discovered during s. This is modeled in the new OPD of **UAV Prototype Developing** in-zoomed (Figure 5-20). States were added to the **Power System** (outcome of o) so it can be either *subject to adjustments* or *final*. The **Payload & Engine Developing**, which is parent process of **Power System**

Integrating (o,10), yields **Power System** (outcome of o) in its *subject to adjustments* state. The **Integration & Testing** process, which is the parent of **Vehicle Integrating** (s,10), requires **Power System** (outcome of o) in its *subject to adjustments* state, and yields *final* **Power System** (outcome of o).

- Finally, despite early efforts to avoid collisions between fuel lines, electric wire harnesses, air ducts, and trim panels, frequently, **Internal Fittings Designing** (m) needs to be adjusted based on interferences discovered during **Final Vehicle Assembling** (t). This is modeled in the new OPD of **UAV Prototype Developing** in-zoomed (Figure 5-20). States were added to **Internal Fittings** (outcome of m) so it can be either *subject to adjustments* or *final*. **Vehicle Developing**, which is the parent of **Internal Fittings Designing** (m,8), yields **Internal Fittings** (outcome of m) in its *subject to adjustments* state. The **Integration & Testing** process, which is the parent of **Final Vehicle Assembling** (t,5), requires **Internal Fittings** (outcome of m) in its *subject to adjustments* state, and yields *final* **Internal Fittings** (outcome of m).

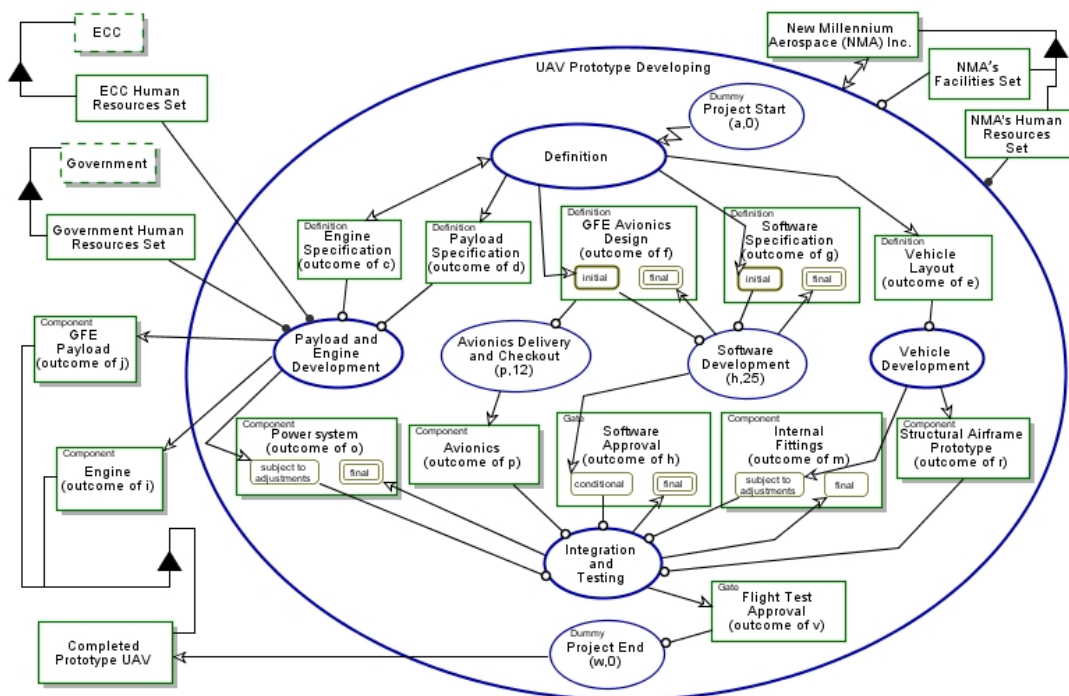


Figure 5-20 OPD of SD1 - UAV Prototype Developing in-zoom

New Millennium Aerospace (NMA) Inc. is physical.
New Millennium Aerospace (NMA) Inc. consists of **NMA's Facilities Set** and **NMA's Human Resources Set**.
NMA's Human Resources Set handles **UAV Prototype Developing**.
ECC is environmental and physical.
ECC consists of **ECC Human Resources Set**.
ECC Human Resources Set handles **Payload and Engine Development**.
Government is environmental and physical.
Government consists of **Government Human Resources Set**.
Government Human Resources Set handles **Payload and Engine Development**.
Completed Prototype UAV is physical.
Completed Prototype UAV consists of **GFE Payload (outcome of j)** and **Engine (outcome of i)**.

Figure 5-21 The automatically-generated OPL paragraph of the OPD in Figure 5 20

GFE Payload (outcome of j) is physical.
 GFE Payload (outcome of j) plays the role of **component**.
 Engine (outcome of i) is physical.
 Engine (outcome of i) plays the role of **component**.
 UAV Prototype Developing requires **NMA's Facilities Set**.
 UAV Prototype Developing affects **New Millennium Aerospace (NMA) Inc..**
 UAV Prototype Developing zooms into **Project Start (a,0)**, **Definition**, **Payload and Engine Development**, **Avionics Delivery and Checkout (p,12)**, **Software Development (h,25)**, **Vehicle Development**, **Integration and Testing**, and **Project End (w,0)**, as well as **Flight Test Approval (outcome of v)**, **Internal Fittings (outcome of m)**, **Structural Airframe Prototype (outcome of r)**, **Power system (outcome of o)**, **Software Approval (outcome of h)**, **Avionics (outcome of p)**, **GFE Avionics Design (outcome of f)**, **Software Specification (outcome of g)**, **Vehicle Layout (outcome of e)**, **Payload Specification (outcome of d)**, and **Engine Specification (outcome of c)**.
 Internal Fittings (outcome of m) is physical.
 Internal Fittings (outcome of m) can be **subject to adjustments** by default or **final**.
 Internal Fittings (outcome of m) plays the role of **component**.
 Structural Airframe Prototype (outcome of r) is physical.
 Structural Airframe Prototype (outcome of r) plays the role of **component**.
 Power system (outcome of o) is physical.
 Power system (outcome of o) can be **subject to adjustments** by default or **final**.
 final is final.
 Power system (outcome of o) plays the role of **component**.
 Software Approval (outcome of h) can be **conditional** by default or **final**.
 final is final.
 Software Approval (outcome of h) plays the role of **gate**.
 Avionics (outcome of p) is physical.
 Avionics (outcome of p) plays the role of **component**.
 GFE Avionics Design (outcome of f) can be **initial** or **final**.
 initial is initial.
 final is final.
 GFE Avionics Design (outcome of f) plays the role of **definition**.
 Software Specification (outcome of g) can be **initial** or **final**.
 initial is initial.
 final is final.
 Software Specification (outcome of g) plays the role of **definition**.
 Project Start (a,0) invokes **Definition**.
 Definition affects **Engine Specification (outcome of c)**.
 Definition yields **initial** **GFE Avionics Design (outcome of f)**, **initial** **Software Specification (outcome of g)**, **Payload Specification (outcome of d)**, and **Vehicle Layout (outcome of e)**.
 Payload and Engine Development requires **Payload Specification (outcome of d)** and **Engine Specification (outcome of c)**.
 Payload and Engine Development yields **subject to adjustments** **Power system (outcome of o)**, **Engine (outcome of i)**, and **GFE Payload (outcome of j)**.
 Avionics Delivery and Checkout (p,12) requires **GFE Avionics Design (outcome of f)**.
 Avionics Delivery and Checkout (p,12) yields **Avionics (outcome of p)**.
 Software Development (h,25) requires **initial** **GFE Avionics Design (outcome of f)** and **initial** **Software Specification (outcome of g)**.
 Software Development (h,25) yields **conditional** **Software Approval (outcome of h)**, **final** **GFE Avionics Design (outcome of f)**, and **final** **Software Specification (outcome of g)**.
 Vehicle Development requires **Vehicle Layout (outcome of e)**.
 Vehicle Development yields **subject to adjustments** **Internal Fittings (outcome of m)** and **Structural Airframe Prototype (outcome of r)**.
 Integration and Testing requires **subject to adjustments** **Internal Fittings (outcome of m)**, **subject to adjustments** **Power system (outcome of o)**, **conditional** **Software Approval (outcome of h)**, **Avionics (outcome of p)**, and **Structural Airframe Prototype (outcome of r)**.
 Integration and Testing yields **final** **Internal Fittings (outcome of m)**, **final** **Power system (outcome of o)**, **final** **Software Approval (outcome of h)**, and **Flight Test Approval (outcome of v)**.
 Project End (w,0) requires **Flight Test Approval (outcome of v)**.
 Project End (w,0) yields **Completed Prototype UAV**.

Figure 5-21 The automatically-generated OPL paragraph of the OPD in Figure 5-20 (continued)

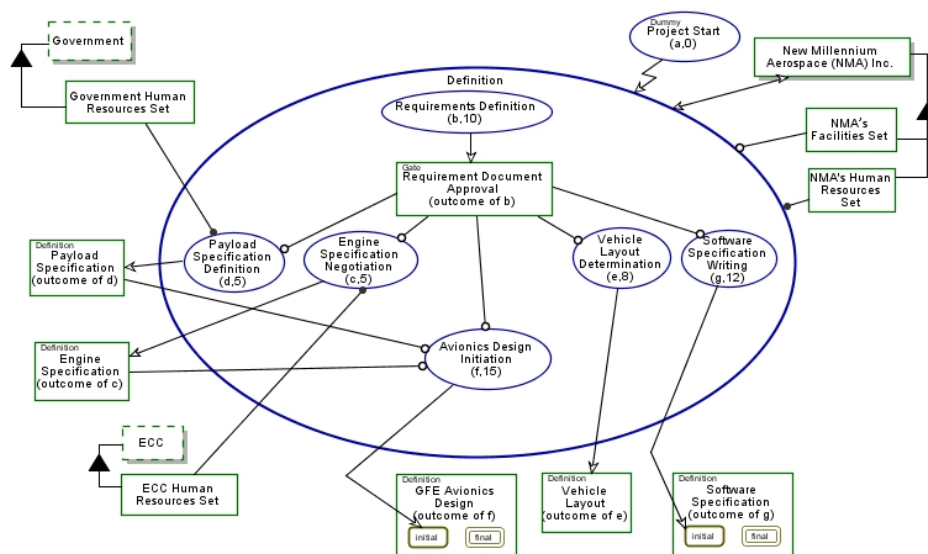


Figure 5-22 OPD of SD1.1 - Definition in-zoomed

New Millennium Aerospace (NMA) Inc. is physical.
New Millennium Aerospace (NMA) Inc. consists of **NMA's Facilities Set** and **NMA's Human Resources Set**.

NMA's Human Resources Set handles **Definition**.

ECC is environmental and physical.

ECC consists of **ECC Human Resources Set**.

ECC Human Resources Set handles **Engine Specification Negotiation (c,5)**.

Government is environmental and physical.

Government consists of **Government Human Resources Set**.

Government Human Resources Set handles **Payload Specification Definition (d,5)**.

Software Specification (outcome of g) can be **initial** or **final**.

initial is initial.

final is final.

Software Specification (outcome of g) plays the role of **definition**.

GFE Avionics Design (outcome of f) can be **initial** or **final**.

initial is initial.

final is final.

GFE Avionics Design (outcome of f) plays the role of **definition**.

Project Start (a,0) invokes **Definition**.

Definition requires **NMA's Facilities Set**.

Definition affects **New Millennium Aerospace (NMA) Inc.**

Definition zooms into **Requirements Definition (b,10)**, **Vehicle Layout Determination (e,8)**, **Software Specification Writing (g,12)**, **Engine Specification Negotiation (c,5)**, **Payload Specification Definition (d,5)**, and **Avionics Design Initiation (f,15)**, as well as **Requirement Document Approval (outcome of b)**.

Requirements Definition (b,10) yields **Requirement Document Approval (outcome of b)**.

Vehicle Layout Determination (e,8) requires **Requirement Document Approval (outcome of b)**.

Vehicle Layout Determination (e,8) yields **Vehicle Layout (outcome of e)**.

Software Specification Writing (g,12) requires **Requirement Document Approval (outcome of b)**.

b).

Software Specification Writing (g,12) yields **initial Software Specification (outcome of g)**.

Engine Specification Negotiation (c,5) requires **Requirement Document Approval (outcome of b)**.

b).

Engine Specification Negotiation (c,5) yields **Engine Specification (outcome of c)**.

Payload Specification Definition (d,5) requires **Requirement Document Approval (outcome of b)**.

b).

Payload Specification Definition (d,5) yields **Payload Specification (outcome of d)**.

Avionics Design Initiation (f,15) requires **Engine Specification (outcome of c)**, **Payload Specification (outcome of d)**, and **Requirement Document Approval (outcome of b)**.

Avionics Design Initiation (f,15) yields **initial GFE Avionics Design (outcome of f)**.

Figure 5-23 The automatically-generated OPL paragraph of the OPD in Figure 5-22

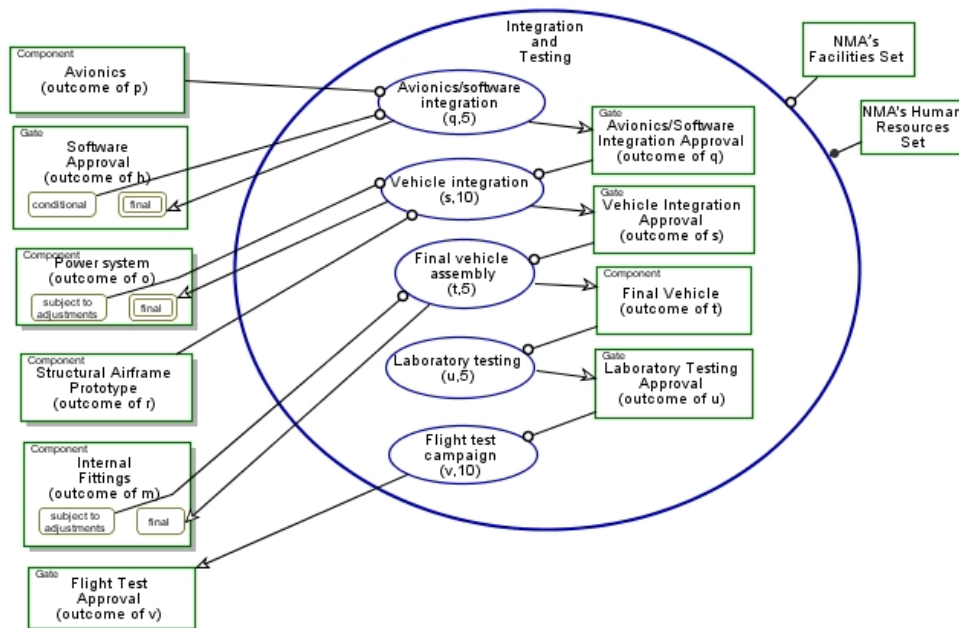


Figure 5-24 OPD of SD1.2 - Integration and Testing in-zoomed

Avionics (outcome of p) is physical.
 Avionics (outcome of p) plays the role of **component**.
 Software Approval (outcome of h) can be **conditional** by default or **final**.
 final is final.
 Software Approval (outcome of h) plays the role of **gate**.
 Power system (outcome of o) is physical.
 Power system (outcome of o) can be **subject to adjustments** by default or **final**.
 final is final.
 Power system (outcome of o) plays the role of **component**.
 Structural Airframe Prototype (outcome of r) is physical.
 Structural Airframe Prototype (outcome of r) plays the role of **component**.
 Internal Fittings (outcome of m) is physical.
 Internal Fittings (outcome of m) can be **subject to adjustments** by default or **final**.
 Internal Fittings (outcome of m) plays the role of **component**.
 NMA's Human Resources Set handles **Integration and Testing**.
 Integration and Testing requires **NMA's Facilities Set**.
 Integration and Testing zooms into **Avionics/software integration (q,5)**, **Vehicle integration (s,10)**, **Final vehicle assembly (t,5)**, **Laboratory testing (u,5)**, and **Flight test campaign (v,10)**, as well as **Laboratory Testing Approval (outcome of u)**, **Final Vehicle (outcome of t)**, **Vehicle Integration Approval (outcome of s)**, and **Avionics/Software Integration Approval (outcome of q)**.
 Avionics/software integration (q,5) requires **conditional** **Software Approval (outcome of h)** and **Avionics (outcome of p)**.
 Avionics/software integration (q,5) yields **final** **Software Approval (outcome of h)** and **Avionics/Software Integration Approval (outcome of q)**.
 Vehicle integration (s,10) requires **subject to adjustments** **Power system (outcome of o)**, **Structural Airframe Prototype (outcome of r)**, and **Avionics/Software Integration Approval (outcome of q)**.
 Vehicle integration (s,10) yields **final** **Power system (outcome of o)** and **Vehicle Integration Approval (outcome of s)**.
 Final vehicle assembly (t,5) requires **subject to adjustments** **Internal Fittings (outcome of m)** and **Vehicle Integration Approval (outcome of s)**.
 Final vehicle assembly (t,5) yields **final** **Internal Fittings (outcome of m)** and **Final Vehicle (outcome of t)**.
 Laboratory testing (u,5) requires **Final Vehicle (outcome of t)**.
 Laboratory testing (u,5) yields **Laboratory Testing Approval (outcome of u)**.
 Flight test campaign (v,10) requires **Laboratory Testing Approval (outcome of u)**.
 Flight test campaign (v,10) yields **Flight Test Approval (outcome of v)**.

Figure 5-25 The automatically-generated OPL paragraph of the OPD in Figure 5-24

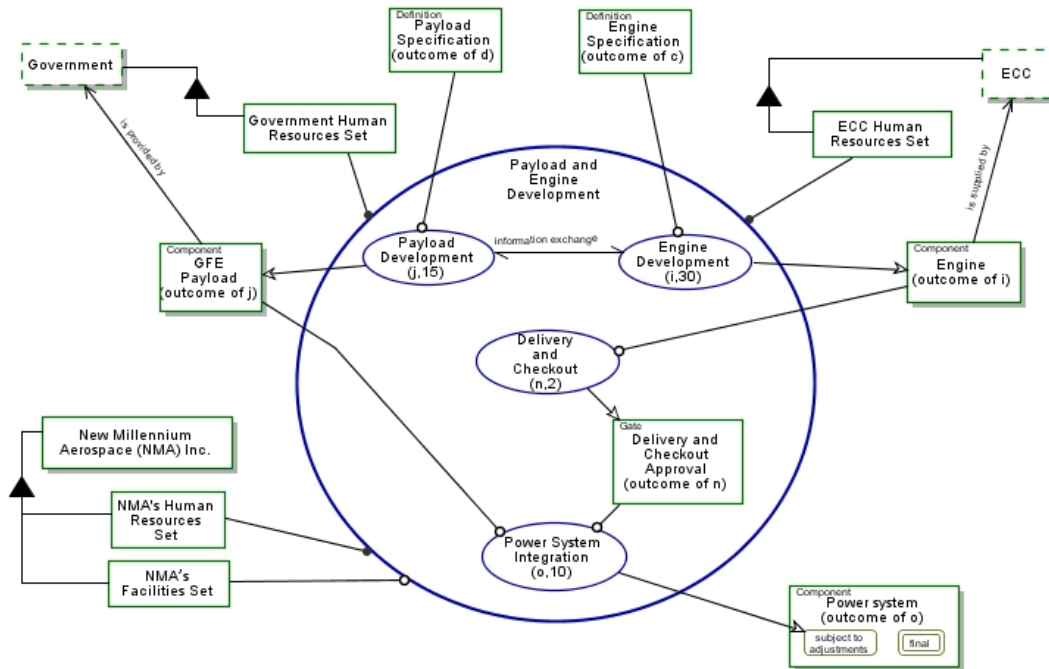


Figure 5-26 OPD of SD1.3 - Payload and Engine Development in-zoomed

New Millennium Aerospace (NMA) Inc. is physical.
New Millennium Aerospace (NMA) Inc. consists of **NMA's Facilities Set** and **NMA's Human Resources Set**.
NMA's Human Resources Set handles **Payload and Engine Development**.
GFE Payload (outcome of j) is physical.
GFE Payload (outcome of j) is provided by **Government**.
GFE Payload (outcome of j) plays the role of **component**.
ECC is environmental and physical.
ECC consists of **ECC Human Resources Set**.
ECC Human Resources Set handles **Payload and Engine Development**.
Government is environmental and physical.
Government consists of **Government Human Resources Set**.
Government Human Resources Set handles **Payload and Engine Development**.
Engine (outcome of i) is physical.
Engine (outcome of i) is supplied by **ECC**.
Engine (outcome of i) plays the role of **component**.
Power system (outcome of o) is physical.
Power system (outcome of o) can be **subject to adjustments** by default or **final**.
final is final.
Power system (outcome of o) plays the role of **component**.
Payload and Engine Development requires **NMA's Facilities Set**.
Payload and Engine Development zooms into **Payload Development (j,15)**, **Engine Development (i,30)**, **Delivery and Checkout (n,2)**, and **Power System Integration (o,10)**, as well as **Delivery and Checkout Approval (outcome of n)**.
Payload Development (j,15) requires **Payload Specification (outcome of d)**.
Payload Development (j,15) yields **GFE Payload (outcome of j)**.
Engine Development (i,30) and **Payload Development (j,15)** exchange information.
Engine Development (i,30) requires **Engine Specification (outcome of c)**.
Engine Development (i,30) yields **Engine (outcome of i)**.
Delivery and Checkout (n,2) requires **Engine (outcome of i)**.
Delivery and Checkout (n,2) yields **Delivery and Checkout Approval (outcome of n)**.
Power System Integration (o,10) requires **GFE Payload (outcome of j)** and **Delivery and Checkout Approval (outcome of n)**.
Power System Integration (o,10) yields **subject to adjustments Power system (outcome of o)**.

Figure 5-27 The automatically-generated OPL paragraph of the OPD in Figure 5-26

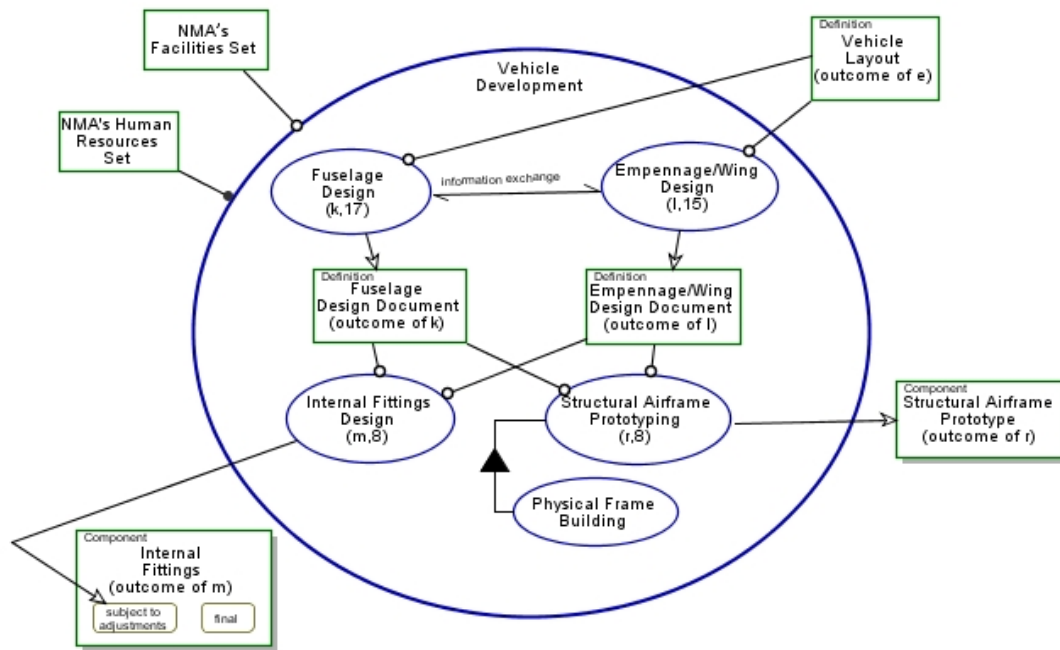


Figure 5-28 OPD of SD1.4 - Vehicle Development in-zoomed

Structural Airframe Prototype (outcome of r) is physical.
Structural Airframe Prototype (outcome of r) plays the role of **component**.
Internal Fittings (outcome of m) is physical.
Internal Fittings (outcome of m) can be **subject to adjustments** by default or **final**.
Internal Fittings (outcome of m) plays the role of **component**.
NMA's Human Resources Set handles **Vehicle Development**.
Vehicle Development requires **NMA's Facilities Set**.
Vehicle Development zooms into **Empennage/Wing Design (l,15)**, **Fuselage Design (k,17)**, **Structural Airframe Prototyping (r,8)**, **Internal Fittings Design (m,8)**, and **Physical Frame Building**, as well as **Empennage/Wing Design Document (outcome of l)** and **Fuselage Design Document (outcome of k)**.
Empennage/Wing Design (l,15) and **Fuselage Design (k,17)** exchange information.
Empennage/Wing Design (l,15) requires **Vehicle Layout (outcome of e)**.
Empennage/Wing Design (l,15) yields **Empennage/Wing Design Document (outcome of l)**.
Fuselage Design (k,17) requires **Vehicle Layout (outcome of e)**.
Fuselage Design (k,17) yields **Fuselage Design Document (outcome of k)**.
Structural Airframe Prototyping (r,8) consists of **Physical Frame Building**.
Structural Airframe Prototyping (r,8) requires **Empennage/Wing Design Document (outcome of l)** and **Fuselage Design Document (outcome of k)**.
Structural Airframe Prototyping (r,8) yields **Structural Airframe Prototype (outcome of r)**.
Internal Fittings Design (m,8) requires **Fuselage Design Document (outcome of k)** and **Empennage/Wing Design Document (outcome of l)**.
Internal Fittings Design (m,8) yields **subject to adjustments Internal Fittings (outcome of m)**.

Figure 5-29 The automatically-generated OPL paragraph of the OPD in Figure 5-28

The updated information extracted from the project model, presented in

Table 5-6, can serve as input to the process pairs comparison during automatic generation of the DSM updated with iterations. The changes from Table 5-4 are underlined (and colored in red if color is visible). The iterative nature of **Software Specifying** (g) and **Software Developing** (h) is underlined both in row g and h. The result from **Software Specifying** (g,12) is now **Software Specification** (outcome of g) in initial state, which is required for **Software Developing** (h,25). The latter is updated with a new result – **Software Specification** (outcome of g *in initial state*) in final state. The same approach was used for the dependency of task h on task f, task q on task h, task t on task m, and task s on task o. These dependencies are extracted as underlined (and red) X's, above the DSM diagonal, as marked in Figure 5-30. If iteration is defined directly between processes, without explicit objects, the information appears in the 5th column - Invoking or other involved Process. The latter are extracted as underlined bold X's in symmetrical locations with respect to the DSM diagonal, as seen in Figure 5-30.

The corresponding DSM is shown in Figure 5-30. We now have some entries in the upper right part of the DSM, above the main diagonal. These interdependencies turn the project graph from an acyclic (having no loops) to a cyclic. These interdependencies can be interpreted as iterations. For example, task j depends on i, but i also depends on j. The new task dependencies in the **UAV Developing & Integrating** project can be interpreted as follows.

Planned Iterations (amounting to work refinement):

$h \rightarrow f$: iteration between software and avionics design

$i \rightarrow j, j \rightarrow i$: iteration between engine development and payload development

$k \rightarrow l, l \rightarrow k$: iteration between fuselage design and empennage/wing design

$g \rightarrow h, h \rightarrow g$: gradual refinement of software specification as coding proceeds along

Unplanned Iterations (amounting to rework)

$q \rightarrow h$: troubleshooting due to problems discovered during avionics/software integration

$s \rightarrow o$: troubleshoot power system due to problems discovered during vehicle integration

$t \rightarrow m$: internal fittings rearrangement due to interference problems downstream

Table 5-6 Updated OPM to DSM Extraction Data

Leaf-level Process Name		Role	Parent Process (cluster)	Invoking <u>or</u> <u>other involved</u> Process	Involved Objects				
Short	Full				Pre-process				Post-process
					Consumables	Enablers	Instruments	Agents	Results
a	Project Start (a,0)	Dummy	UAV Developing & Integrating						
b	Requirements Defining (b,10)		Defining	Project Start (a,0)					Requirement Document Approval (outcome of b)
c	Engine Specifying (c,5)		Defining				Requirement Document Approval (outcome of b)	ECC Agents Set	Engine Specification (outcome of c)
d	Payload Specifying (d,5)		Defining				Requirement Document Approval (outcome of b)	Government Agents Set	Payload Specification (outcome of d)
e	Vehicle Layout Designing (e,8)		Defining				Requirement Document Approval (outcome of b)		Vehicle Layout (outcome of e)
f	Avionics Designing (f,15)		Defining				Requirement Document Approval (outcome of b)		GFE Avionics Design (outcome of f) <u>in initial state</u>
							Engine Specification (outcome of c)		
							Payload Specification (outcome of d)		
g	Software Specifying (g,12)		Defining				Requirement Document Approval (outcome of b)		Software Specification (outcome of g) <u>in initial state</u>
h	Software Developing (h,25)		UAV Developing & Integrating				GFE Avionics Design (outcome of f) <u>in initial state</u>		Software Approval (outcome of h) <u>in conditional state</u>
							Software Specification (outcome of g) <u>in initial state</u>		Software Specification (outcome of g in initial state) <u>in final state</u>
i	Engine Developing (i,30)		Payload & Engine Developing	<u>Payload Development (i,15)</u>			Engine Specification (outcome of c)		Engine (outcome of i)
j	Payload Developing (j,15)		Payload & Engine Developing	<u>Engine Development (i,30)</u>			Payload Specification (outcome of d)		GFE Payload (outcome of j)
k	Fuselage Designing (k,17)		Vehicle Developing	<u>Empennage/Wing Design (l,15)</u>			Vehicle Layout (outcome of e)		Fuselage Design Document (outcome of k)
l	Empennage/Wing Designing (l,15)		Vehicle Developing	<u>Fuselage Design (k,17)</u>			Vehicle Layout (outcome of e)		Empennage/Wing Design Document (outcome of l)

Table 5 6 Updated OPM to DSM Extraction Data (continued)

Leaf-level Process Name		Role	Parent Process (cluster)	Invoking <u>or</u> <u>other involved</u> Process	Involved Objects				
Short	Full				Pre-process			Agents	Post-process Results
Consumables	Enablers	Instruments							
m	Internal Fittings Designing (m,8)		Vehicle Developing				Fuselage Design Document (outcome of k) Empennage/Wing Design Document (outcome of l)		Internal Fittings (outcome of m) <u>in subject to adjustments state</u>
n	Delivery and Checkout (n,2)		Payload & Engine Developing				Engine (outcome of i)		Delivery and Checkout Approval (outcome of n)
o	Power System Integrating (o,10)		Payload & Engine Developing				GFE Payload (outcome of j) Delivery and Checkout Approval (outcome of n)		Power system (outcome of o) <u>in subject to adjustments state</u>
p	Avionics Delivery and Checkout (p,12)		UAV Developing & Integrating				GFE Avionics Design (outcome of f)		Avionics (outcome of p)
q	Avionics/software integrating (q,5)		Integrating & Testing				Software Approval (outcome of h) <u>in conditional state</u> Avionics (outcome of p)		Avionics/Software Integration Approval (outcome of q) Software Approval (outcome of h <u>in conditional state</u>) <u>in final state</u>
r	Structural Airframe Prototyping (r,8)		Vehicle Developing				Fuselage Design Document (outcome of k) Empennage/Wing Design Document (outcome of l)		Structural Airframe Prototype (outcome of r)
s	Vehicle integrating (s,10)		Integrating & Testing				Power system (outcome of o) <u>in subject to adjustments state</u> Avionics/Software Integration Approval (outcome of q) Structural Airframe Prototype (outcome of r)		Vehicle Integration Approval (outcome of s <u>in subject to adjustments state</u>) <u>in final state</u>
t	Final vehicle assembling (t,5)		Integrating & Testing				Internal Fittings (outcome of m) <u>in subject to adjustments state</u> Vehicle Integration Approval (outcome of s)		Final Vehicle (outcome of t <u>in subject to adjustments state</u>) <u>in final state</u>
u	Laboratory testing (u,5)		Integrating & Testing				Final Vehicle (outcome of t)		Laboratory Testing Approval (outcome of u)
v	Flight test campaigning (v,10)		Integrating & Testing				Laboratory Testing Approval (outcome of u)		Flight Test Approval (outcome of v)
w	Project End (w,0)	Dummy	UAV Developing & Integrating				Flight Test Approval (outcome of v)		Completed Prototype UAV

to From	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
a	a																						
b	X	b																					
c		X	c																				
d		X		d																			
e		X			e																		
f		X	X	X		f		X															
g		X					g	X															
h						X	X	h									X						
i			X						i	X													
j				X					X	j													
k					X						k	X											
l					X					X	l												
m											X	X	m								X		
n									X					n									
o										X				X	o				X				
p						X										p							
q								X								X	q						
r											X	X						r					
s															X		X	X	s				
t												X							X	t			
u																			X	u			
v																					X	v	
w																						X	w

Figure 5-30 The Project Plan Activities DSM with iterations

Applying a partitioning or clustering algorithm by swapping columns and rows of the DSM to improve the task sequence and minimize loops can be done manually or via an automated algorithm. We apply the partitioning algorithm manually as follows:

1. Move all tasks that don't require inputs from other tasks to the top
2. Move all tasks that don't produce outputs for other tasks to the bottom
3. Group coupled tasks together until a group has either no input or no output requirement
4. Repeat steps 1 or 2 for each group as appropriate

5. Finish when tasks are clustered such that the sum of the distance of the off-diagonal terms above the diagonal is minimized

Figure 5-31 shows the results of the manual clustering work. Coupled meta-tasks are highlighted by enclosing them in a highlighted box.

	a	b	c	d	e	g	f	h	p	i	k	l	r	i	j	n	o	s	m	t	u	v	w
a	a																						
b	X	b																					
c		X	c																				
d		X		d																			
e		X			e																		
g		X				g		X															
f		X	X	X			f	X															
h						X	X	h		X													
p							X		p														
q								X	X	q													
k				X							k	X											
l				X							X	l											
r											X	X	r										
i			X											i	X								
j				X										X	j								
n														X		n							
o															X	X	o	X					
s									X			X					X	s					
m										X	X							X	m	X			
t																		X	X	t			
u																				X	u		
v																					X	v	
w																						X	w

Figure 5-31 Manually-clustered DSM matrix for the UAV project

The manual clustering leads to a cleaner project structure with the following task clusters (macro-tasks, or DSM blocks):

1. Requirements Definition (b, c, d)
2. Layout (e)
3. Software and Avionics Design (g, f, h, p, q)
4. Structural Airframe Design (k, l, r)
5. Engine and Payload Design (i, j, n, o)
6. Assembly & Integration (s, m, t)
7. Testing (u, v)

Within these 7 meta-tasks, requirements definition (1) must occur first followed by vehicle layout (2). This is followed by the three major blocks: Software and Avionics

Design (3) Structural Airframe Design (4), and Engine and Payload Design (5) which can proceed in parallel. Only Structural Airframe Design (4) explicitly depends on Layout (2). Each of the meta-tasks are highly coupled internally, but loosely coupled with other meta-tasks. Finally, once the design work is done in those three meta-tasks (3, 4, and 5), assembly & integration (6) and testing (7) can occur.

The only iteration that remains outside one of the blocks is $s \rightarrow o$. This iteration cannot be avoided, since it would only be discovered downstream during integration of the power system into the unmanned air vehicle.

Table 5-7 presents the comparison of the resulted clustering with the original clustering in the OPM model.

Table 5-7 Comparisons of DSM Clustering with OPM Clustering

New DSM Clustering			OPM Original Clustering		
#	Cluster Name	Contained Tasks	Cluster Name	Contained Tasks	#
1	Requirements Definition	b, c, d	Defining	b, c, d,	1
2	Layout (e)	{Ø}		e,	
3	Software and Avionics Design	g, f,	Software Developing (h)	g, f	2
		h,		{Ø}	
		p, q		{Ø}	
		u, v	Integrating & Testing	q,	4
4	Testing	u, v	Vehicle Developing	u, v,	
5	Assembly & Integration	s, t, m		s, t	5
6	Structural Airframe Design	k, l, r	Payload and Engine Developing	m,	
7	Engine and Payload Design	i, j, n, o		k, l, r	6
				i, j, n, o	

5.8 Combined views

5.8.1 Combined WBS-CPM

Combining the WBS and the CPM views, as shown in Figure 5-32, we get a combined WBS-CPM view, which represents both the things (objects and processes) and their criticality. This combined view enables focusing on the delivery of the critical objects. Figure 5-32 is the same as Figure 5-15 with the addition of indication of critical things by a double contour. The criticality of processes is determined by CPM—these are all the processes with zero slack. The criticality of the objects is determined from their relationships in the PPLM model with critical or sub-critical processes, based on the OCPM, as explained earlier.

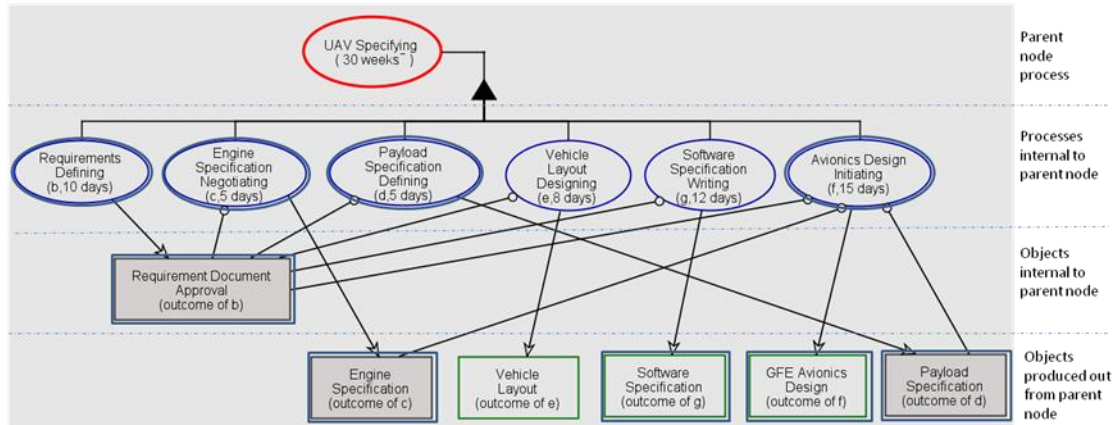


Figure 5-32 Automatically Extracted Combined WBS-CPM for the UAV Specifying node

Examining Figure 5-32, one can see that four of the six subprocesses of **UAV Specifying** are on the critical path, as denoted by their double contour ellipses. There are also five critical objects, denoted by double contour rectangles. They are critical since they are required for critical processes. The objects **Requirements Document Approval**, **Payload Specification**, and **Engine Specification** are critical as they are required as inputs to critical internal processes. The two other objects, **Vehicle Layout** and **Software Specification**, are critical based on the PPLM model.

This combined view is useful, for example, in case **UAV Specifying** is delayed or if, for some reason, there is a need to reduce the effort invested in **UAV Specifying**, based on this view, the manager can decide to mobilize resources that are occupied in executing the **Vehicle Layout** subprocess, since it is the only non-critical object.

Figure 5-33 shows **Integration & Testing** as the parent of five subprocesses, all of which are critical. Therefore, all the objects required as inputs to these subprocesses are also critical. **Flight Test Approval** is critical since it is in the preprocess object set of **Project End**—a process which is on the critical path. Based on the CPM view, four of the five processes that yield objects feeding into subprocesses of **Integration & Testing** are not on the critical path. Therefore, each of these for processes has a non-zero slack. However, this slack can still be small. If this is the case, although not critical, if delayed, one or more of these processes can potentially delay the start of **Integration & Testing**. And since all five **Integration & Testing** subprocesses are critical, **Integration & Testing** as a whole will be delayed.

5.8.2 Critical DSM: Combining DSM and CPM

A combined DSM-CPM view, which we call the Critical DSM, shown in Figure 5, presents in the dependency matrix both the things (objects and processes) and their criticality. Figure 5 is the same as Figure 5-19 with the addition of indication of critical things underlined (and by red) color.

Examining the dependency types enables us to gain deeper understanding of the way project processes depend on each other via critical processes and identified critical objects. For example, a quick look at the OPDSM in Figure 5 reveals that the Gate of **Requirements Document Approval** is a critical object, as it precedes and is a prerequisite for five critical tasks. We can also track critical Ds (Documents) in the project; design documents, specifications, layouts, etc. which are critical in terms of influence on other project task. Furthermore, we can identify the critical Cs

(Components), and evaluate their potential impact on the corresponding critical processes.

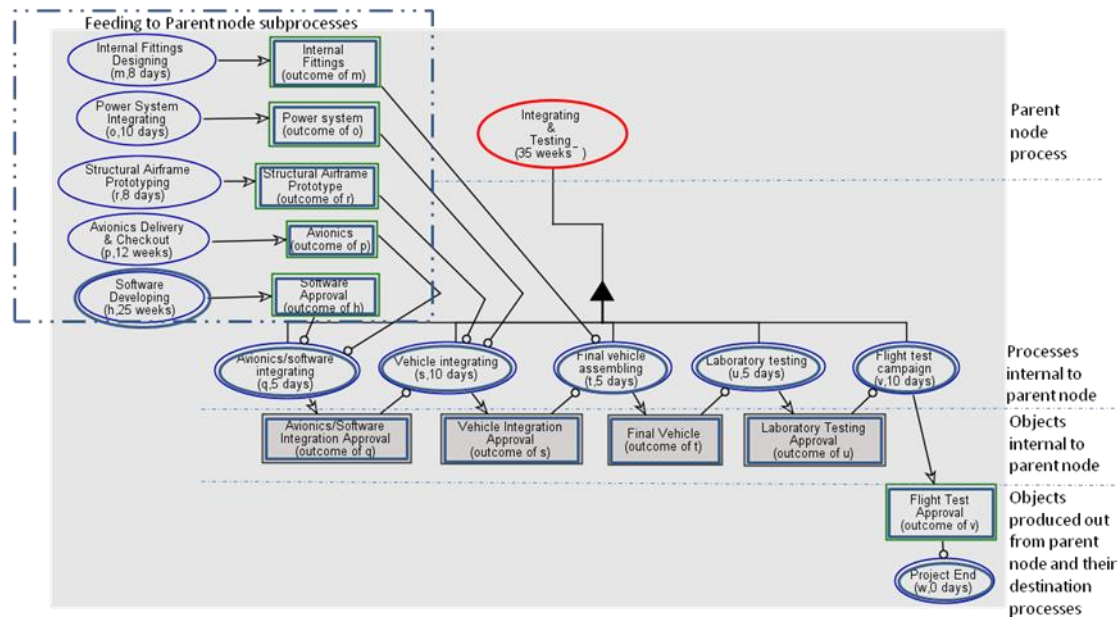


Figure 5-33 Automatically extracted Combined WBS-CPM for the Integration & Testing node

5.9 Summary

Current systems engineering management and project management utilize a host of methods and techniques for devising a systems engineering management plan, including Critical Path Method (CPM), Gantt chart, Work Breakdown Structure (WBS), and Design Structure Matrix (DSM). These methods focus almost exclusively on activities and tasks to be performed, along with their relationships and durations, while the deliverable—the product or system to be developed—is implicit. Using a running example of an unmanned aerial vehicle, we reviewed these representations and discussed problems stemming from the lack of explicit representation of the product facet in current project management methods. Based on this observation, we proposed an integrated conceptual model-based combined project-product lifecycle approach for systems engineering management.

Starting with construction of an OPM model-based plan of the project and the product it is expected to deliver, we obtain a comprehensive model that serves as a basis for deriving the customary repertoire of project management tools. However, rather than being constructed separately and independently, which is a certain source of mismatches and incompatibilities with potential detrimental consequences to the project success, these project views—Activities Network, CPM, Gantt, WBS, and DSM—are derived reflections of various aspects of a comprehensive underlying OPM model and are therefore consistent and coherent. Any change in the project model is automatically reflected in the various project management views. Moreover, any change in the product to be delivered—the ultimate output of the project—can be modeled in the joint project-product model and its implications on the project parameters can be directly inspected and assessed.

The fact that the model integrates the process view of project activities and tasks with the object view of the product deliverables with its components at all levels and their

associated artifacts enables the project manager to focus on advancing the completion of the output deliverables rather than on just performing processes without direct reference to their anticipated outcomes. Knowing the critical path and the critical artifacts—those generated by processes along the critical path—enables to closely manage and monitor them.

to From	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
a	a																						
b	X	b																					
c		<u>G</u> Req. Doc. Appr oval	c																				
d		<u>G</u> Req. Doc. Appr oval		d																			
e		<u>G</u> Req. Doc. Appr oval			e																		
f		<u>G</u> Req. Doc. Appr oval	<u>D</u> Eng. Spec.	<u>D</u> PL Spec.		f																	
g		<u>G</u> Req. Doc. Appr oval					g																
h						<u>D</u> Av.De s.	<u>D</u> SWSp ec.	h															
i			<u>D</u> Eng. Spec.						i														
j				<u>D</u> PL Spec.						j													
k					<u>D</u> Veh. Lav						k												
l					<u>D</u> Veh. Lav							l											
m											<u>D</u> Fus. DD	<u>D</u> Emp. DD	m										
n									C Eng.							n							
o										C PL					G Del	o							
p						<u>D</u> Av.De s.											p						

Figure 5-34 The Project Plan OPDSM with designated critical things (processes and objects)

to From	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	
q								G SW Appr oval								C Avionics	q							
r											D Fus. DD	D Emp. DD						r						
s															C PS.		G Int Appr oval	C SAP	s					
t													C Int. Fit.						G Int. proval	t				
u																			C FV		u			
v																					G Lab proval		v	
w																						G FT proval		w

Figure 5The Project Plan OPDSM with designated critical things (processes and objects)
(continued)

6 Perceived Characteristics of SE Tools and Methods

6.1 Introduction

The survey examined systems engineers in a large enterprise, in which several initiatives have been (and still are) conducted to implement methods and tools for facilitating systems engineering, over the last fifteen years. The enterprise produces large-scale complex systems and employs a few thousands of employees, of whom several dozen are systems engineers. The recognition in the necessity of methods and tools for systems engineering has matured in the enterprise, hand in hand with the development of these over the rest of the industrial world [2], [3], [17]. The common belief is that using methods and tools for systems engineering would improve the products and its deliverables in projects where these techniques and tools are implemented [24], [25]. The premise for this belief is that investing in methods and tools, especially at the early phases of the project's lifecycle, will result in savings of other consumed resources at successive phases of the project, particularly during integration and testing [3], [55]. Based on this, many enterprises have developed in-house tools and methodologies. At the same time, different tools and methodologies have started to be developed by software vendors in order to offer potential solutions in accordance with different standards and approaches. Despite this tendency, the actual implementation of methods and tools for systems engineering has met with resistance to change and rejection by the majority practitioners' roles [56].

6.2 Research Population and Setting

The research focused on ten systems engineers (7 men and 3 women) from different divisions of the large enterprise. Their vocational experience in systems engineering ranged from 5 to 25 years of professional experience. The former are considered "beginners" and are usually members in a systems engineering group of a specific project, while the latter are positioned as leading senior systems engineers in specific projects or heads of systems engineering departments. All the participants have been serving in a variety of roles of systems engineering over the years and all of them also took part in methods and tools implementation across the enterprise. Their origin disciplines are diversified, as can be expected of systems engineers, and they belong to different projects, different in nature, scope, and lifecycle phase in which the project is at.

The majority of current systems developed in the enterprise cannot be handled by a single systems engineer, so cooperation between individuals and teams is mandatory. The systems engineers in the enterprise are responsible for the systems engineering in a specific project, whether they belong to the project itself or positioned in the project in a matrix structure.

Systems engineering exists as part of the projects, therefore a senior systems engineer in a project must work in tight cooperation with the project manager in order to develop and deliver the system to the customer, in accordance with the agreed requirements. This can be achieved through a large range of methods and tools that can be applied in a project, and it is up to the project leaders to choose the ways of action in each project. The majority of tools that have been used by systems engineers in the enterprise's projects are aimed for technical management. In general, these are databases in which the system is documented. They contain requirements, functions,

architecture and the relationships among them, enabling documents and reports extraction. Only in few projects methods and tools are used extensive used. There is no enterprise directive for this issue. When methods and tools are used, it is a result of a local initiative, arising from the recognition of the need or expected utility for the project.

6.3 Research Methodology

The research conducted as part of this dissertation was an introspective research [57], designed to investigate the characteristics of methods and tools as perceived by systems engineers. The research population consisted of ten systems engineers in the large enterprise described above. Using interviews, observations and analysis of relevant documents, we explored the implications and interactions among different elements related to our research questions.

The observations were conducted during meetings between systems engineers and systems engineering facilitators, conducted as part of the overall initiative across the enterprise. A few dozen documents were analyzed, including internal directives of the enterprise, correspondence among people, and other relevant documents.

The open interviews were guided by a set of questions, validated in an iterative process by two high ranking systems engineers in the enterprise, both of whom were interested in the research results for further decision making regarding systems engineering methods and tools across the enterprise. Interviews duration ranged from 60 to 90 minutes each and they were all recorded and transcribed. The first part of each interview was devoted to the exploration of systems engineering characteristics in general, followed by a second part, in which the use of methods and tools was more specifically gauged. This way of action was planned in order to focus on understanding the root cause of each systems engineer in the broader context – how he or she understands his or her role, its characteristics and perceptions of systems engineering in general. This was used later on to relate to the more specific characteristics of the methods and tools they have been exposed to. All participants are systems engineers who are involved, in one way or another, in promoting systems engineering issues, inside and outside the enterprise, and they gladly agreed to take part in the research.

The results and analysis were intertwined with the data collection process, in an iterative investigation manner. The texts analysis was conducted in a thematic approach [58] rather than using words and expression as analysis units.

6.4 Results and Analysis

The central findings relate to the concept of "utility" gained from using methods and tools that has emerged from the research field by the practitioners. The analysis of the data revealed three domains for which the perceived utility was expressed and negotiated: the organizational hierarchy domain, the system complexity domain, and the professional hierarchy domain. Although none of these terms has been used by the researchers, all practitioners used them to express their notions about using methods and tools in their systems engineering practice.

6.4.1 The utility in the organizational hierarchy domain

The organizational hierarchy domain has emerged from the collected data. As shown in Figure 66-1, the spectrum starts at the bottom with the single project level, going

through the Plant level, followed by the Division level, and ends with entire Enterprise level. The different levels arose from the collected data and each level has been addressed in relation to utility of using methods and tools for systems engineering.

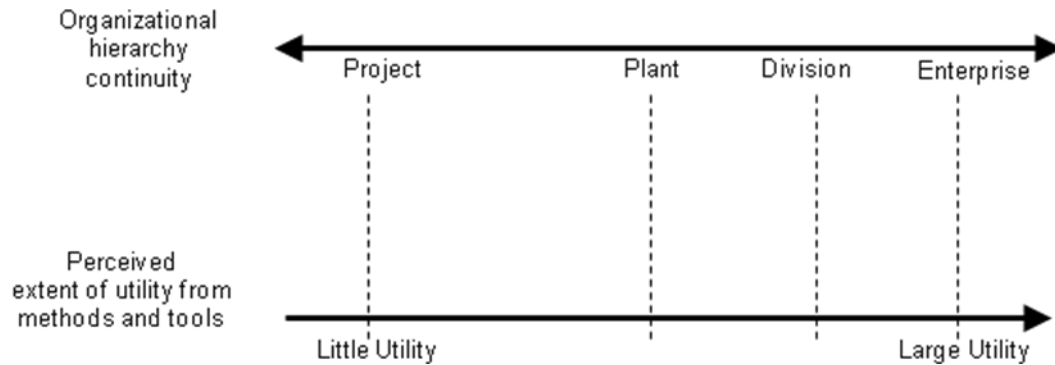


Figure 66-1 The utility span over the organizational hierarchy domain

At the Project level, the systems engineers perceive the methods and tools as a burden which is time consumable and prevents them from working on the real engineering activities required in order to achieve the project goals, as one of them noted:

"Although there is a will to manage things in an orderly manner and do the methodological process and everything, the reality dictates the necessity to run forward and...achieve the results in such a way that does allow so much for tidy methodological work...no one suggests that a project will delay milestones or important events during the project because someone wants to put everything in order...".

The systems engineers grasp the use of methods and tools as a factor that causes project delays: *"the difficulty in my opinion, in all projects, rises from the fact a project lives and breathes and...usually the process ... if done in an orderly and methodological manner, will delay the project. I mean, if people will be forced to work orderly and methodologically... the project will surely be delayed"*. When this systems engineer was asked if he thought there were any systems engineers who recognized any benefit from using methods or tools for systems engineering in project, he answered: *"On one hand the benefit is visible, on the other hand they realize that...the investment is very large and sometimes you don't want to get into it. Ah...since they say they don't have time or don't have enough people to invest in all the work associated with this maintenance, they prefer not to get into it at all. Because if it isn't maintained properly they what do I need it for. There are people that don't see at all...any ah...advantage in working with such a thing, on the contrary, only a resource consuming..."*. Intensification of this view among systems engineers is manifested by the reaction of one, who, when asked by his manager to update data in the systems engineering database of the project, in which ordered methodologies and tools are used, said: *"The SRS [Systems Requirements Specification] and SDD [System Design Document] still require a lot of work but right now the priority is on flight tests. Therefore, we don't update anything in the database. When the right time comes, we will update the relationships"*.

While the utility at the project level is perceived as low, there is a perception that from the next project and on, there will be tangible utility from the use of methods

and tools in the first project. In this sense, the utility is perceived as high at the enterprise level since all projects are under the enterprise responsibility, rather than a single project: *"really, from experience ah...usually from failures...in order to prevent a failure in the next project or adjacent project. Now, the more tools, the more scripts for data quality checks and deliverables quality, the smaller the chance for mistakes and less things slip...things due to lack of coordination between groups will get smaller as we will enhance automation ah...of the things and use of tools"*.

The difference between the perceived utility at the project level and perceived utility at the enterprise level is salient in the things one of systems engineers said: *"managers don't think it is something that is required for their project. They don't see the benefit of the...it's not something like you can push a button and see the result of it immediately. This is something that can be reached not even in the project you are working in. It can be reached only in the next project or even in following ones, in a family of products, in Reuse of projects...at the first time it will only holdup, it will only oppress, it will only ah...it will not show you have saved anything, on the contrary maybe you only spent..."*.

Another system engineer also addressed the utility gained by the enterprise at the next projects from investment in a previous project as a motivation for using tools: *"the proficiency you need to use afterwards in additional projects that relay on a former project ah...on a basis project is one thing. You design a system using the tools and then you have a repository that you can use afterwards a lot easier and it is more available for the next projects...and that is the reason for choosing to implement these tools"*.

6.4.2 The utility span over the system complexity domain

The system complexity continuity (see Figure 62-), as perceived by practitioners was extracted from the data, starting with a simple system, followed by a complex system, and ending with system of systems (SoS). The practitioners associated different utilities to the use of methods and tools with respect to the system complexity domain.

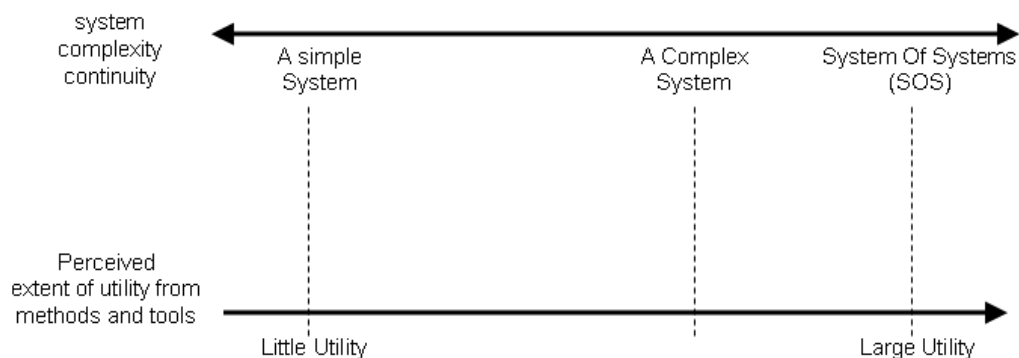


Figure 6-2 The utility span over the system complexity continuity

The potential utility from using tools in different scales of complexity was described by one systems engineers: *"This is the kind of project that...that...in the classical view when presenting tools you talk about a system that has, for the sake of the matter ah...examples of five, six, ten components and the interfaces among them, so this is equivalent to a project that for the sake of the matter is something like one box in a*

complex system OK?...Not something special like the other large scale project. This is why working with a tool in this project let's say is not something that I see today as something that would advance or improve or fix some problem that has happened. As for the future, if there would be, I really hope so, a full scale development, I believe we will work with tools".

The conception is that in a large scale projects, in contrast to small scale projects, it is impossible to meet the goals without using methods and tools: *"In this kind of projects, if you don't work with such a tool where you centralize the requirements, and the design...the project is simply...without such a tool in the project, I would expect it will crash..."*.

The practitioners refer to the potential utility, while describing a tight coupling between the level of system complexity and the project's size: *"as the projects become larger, more complex, more complicated and the technology enables for example by using optical fibres from here to there, to transmit endless information in different timings etc...the heroes [senior systems engineers] cannot solve all the problems...and the use of tools, again, enables the early reveal of things on one hand, and coordinating expectations with the project manager and the customer on the other hand, when things are revealed a lot earlier than at the final stages of delivery".* *"When the project has a more so called start-up nature, then the tendency to work with tools is much smaller. I mean the tendency is not to waste time...time or resources on that thing. In a large project...in which the work is spread over not so short period of time ah...let's say, there is a chance for a mess if there won't be order, there can be a larger tendency towards working with tools".*

"In large systems, the larger the system the more complex it is and you need to know how to base on methodologies and tools that would give the systems engineer the relevant information without the need for him to start reading the entire ah...each sub sub document of ah...down until the last detail because at the end he or she will turn again being an expert instead...instead of practicing systems engineering".

6.4.3 The utility span over the professional hierarchy domain

The professional hierarchy domain (see Figure 6-3) has emerged from the collected data, including the systems engineer positioned with responsibility for a subsystem within a project, on one end. At the other end of the professional hierarchy spectrum is the senior systems engineer, whose role is managerial. The difference between the two, as perceived by the practitioners, can be understood from the following description: *"The head of systems engineering group has technical and managerial responsibility for all the systems engineering activities in the project. He is actually the technical manager of the project de facto. He is in charge of the systems engineer and delegates technical and managerial responsibility for a subsystem or LRU [Line Replaceable Unit]. This is the case in general. A senior systems engineer, head of systems engineering group, is actually the one who is charge of all the phases...and he is the one...he is the focal point or let's call it the interaction point with other disciplines that exist within the project. These include facilities, ah...software engineering, electrical engineering, ah...the project's management etc."*

Others also see management as an integral part of the senior systems engineering role: *"I view the project manager and senior systems engineer as a single entity...in general I would divide the entity so that the project manager is more responsible for...external relations...leaving the technical decisions, the technical management ah...in the hands of the systems engineer".*

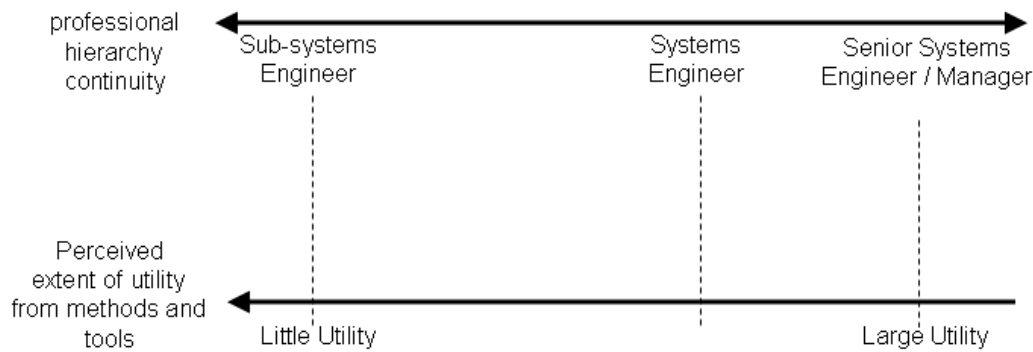


Figure 6-3 The utility span over the professional hierarchy domain

The perceived utility from using methods and tools is low for senior systems engineering managers, while for systems engineers in charge of a subsystem the utility is perceived as high. The senior systems engineers are usually knowledgeable and due to their rich professional experience they do not grasp the use of methods and tools as beneficial: *"they don't like to be instructed, they don't like being checked, after all they did OK yesterday so who can teach them how to do things today..."*. One practitioner described the tools implementation in a specific project: *"the choice to implement these tools in this new technology project and other projects is...there are old dinosaurs [senior systems engineers] in the other projects. They work mainly based on their knowledge and ah...more or less what they have been doing for years...I defined in advance that ah...the use of tools would be conducted only in projects where there are young people. It is no wonder that the older are left in the old technology domain projects where all are dinosaurs who work in the same way for years. They will never use tools. They are reluctant. It is not part of what they are used to do. They have already been used to work without"*.

One possible reason for the perceived low utility for experienced senior systems engineers is that they consider the use of methods and tools as reducing the need for them in the enterprise. One practitioner refers to them as heroes and says: *"the more tools are used...for different managerial levels, there is a capability of quality check [of the systems engineering data], without having to be an expert so...it reduces the need for heroes. Less dependency...you can see things that without the tools only heroes could see"*.

6.4.4 Additional Findings

Three additional findings emerged from the collected data:

(a) Differentiation between orderly work per se and use of tools

Although systems engineers don't hurry into adopting methods and tools, they favor working in an orderly fashion, especially in work concerning systems engineering documents generation. Following are three examples:

"It took a lot of efforts to establish a common language. A language that is clear. It starts with a dictionary terms that has to be clear to all and it is not taken for granted and it continues to working methods: when you update a document, how you work with track changes or don't work with track changes, there are versions, who hands

what to whom, who suggests corrections, who approves corrections, who holds the document sources, what documents at all, who will write each document. So this is the first stage of...creating the work method".

"So there is also a template for this and there is ah...experience that was gained in other projects. There are example documents and ah...there is what is called accepted methods in which you conduct integration for example so you don't invent the wheel. If you get stuck than you improvise or develop some change to the method but usually there are acceptable methods that prove themselves and you make an effort to act in accordance to those".

"We started doing a first step toward this subject of tools...but other than that [referring to requirement management] we don't work with tools. What we do, is we make an effort to work with ordered documents. We make efforts to release engineering documents as required ah...alternatives to the different subsystems have been defined as required by ordered method and...the selection of alternative for each sub system has been conducted in an ... ordered manner. Inside the department [the systems engineering department] itself there is knowledge transfer ah...we used to conduct full day tutorials on different subjects of systems engineering, presenting examples of ah...how to write specific documents or specifications...in more ordered manner. In addition we have meetings. This is one of the advantages from that all systems engineers sit together in the same space so you sometimes consult: 'do you happen to have this or that template?' ".

(b) The professional training is perceived as a crucial incentive

Systems engineers that have been exposed to use of methods and tools during professional training of systems engineering are the ones who adopt those in projects, as one of the practitioners says: *"giving the different systems engineers the right training and...emphasizing the advantages of using tools usually is a success and than leveraging in a second project, or specific enterprise...".* The contribution of exposure to methods and tools during training to increasing the willingness to adopt them in projects arose in one of the meetings between the division representative and a senior systems engineer in a project in which tool are used: *"I think that both you and your colleague are graduates of the systems engineering course [an internal training at the enterprise]. Does this seem meaningful to you? – Yes, absolutely. We have been exposed to tools and their importance and in this project I realized that we must manage the requirements in a tool".*

(c) Management commitment is perceived as a necessary condition

Management commitment is perceived as a crucial key factor to the willingness to use tools, as one systems engineer who also conducts tools implementation in a large project within the enterprise testifies: *"As I said, at the end of the day they [systems engineers] get some management commitment. I mean eventually ah...they receive the hours required for this [use of tools in the project] ah...paying the hours of my efforts in this, and all related required investments".* Others explicitly said: *"they [systems engineers] don't like to work with tools...it has to be enforced by the management. There should be a very clear management involvement ...requiring the use of tools."* *"One of the problems is that it is very hard to measure how much is saved by the use of tools in comparison to not using tools ah...so there has to be management commitments and management investment...it is critical".*

6.5 Discussion and summary

Although the common belief is that the use of methods and tools for systems engineering in a project leads to process improvement and higher quality of deliverables, the results suggest that for a given project the utility is perceived as low by systems engineers. Yet, they ascribe high utility to the use of methods and tools at the enterprise level. These findings can provide a possible explanation to the fact that the majority of systems engineers do not voluntarily adopt methods and tools in a specific project.

It might be the case that in projects in the enterprise where methods and tools are used by systems engineers, it is due to a local figure with strong enterprise consciousness that drives him or her to initiate the use of methods and tools, adopting the enterprise point of view. The lack of enterprise enforcement to use methods and tools because of the conception that systems engineers cannot be forced to do so is not supported by the results. On the contrary, the results indicate that systems engineers perceive the utility as high at the enterprise level; therefore the enterprise leaders can adopt this notion and enforce the use of methods and tools by systems engineers. Since the systems engineers understand the potential benefit, they are likely to be cooperative, once they realize that upper management expects them to do so.

The finding of the system complexity domain can provide an explanation to the fact that the majority of systems engineers are reluctant to use methods and tools. The perceived utility from using methods and tools in medium and small scale project is low, but this is exactly the scale of projects for which the majority of practitioners are assigned. In contrast, they perceive the utility as high for large scale projects dealing with complex systems, and this is correlated with the fact that the local initiatives for using methods and tools in the enterprise are actually in large scale projects. These two findings are complementary: for small or medium scale projects, the perceived utility is high at the enterprise level. A large scale project can be considered as an enterprise in itself; therefore the perceived utility from using methods and tools in such a project is high.

These findings give rise to EPPLM – Enterprise PPLM, a possible future work to enlarge the scope of the PPLM research to include the enterprise as the execution agency of multiple projects, each delivering a product or a product line.

7 The Combined Project-Product Methodology

7.1 The Generic Project Construct

The PPLM model is constructed using a designated template called the Generic Project Construct (GPC), which is presented in Figure 7-1. The GPC comprises the process **Task Execution**, which takes an expected duration and is aimed at achieving a specific **Deliverables Set**, using a **Resources Set** that specializes into **Agents Set**, **Budget**, and **Instruments Set**. **Agents Set** is a set of required human enablers, **Instruments Set** is a set of required non-human enablers, and **Budget** is a consumed monetary input. Although **Budget** could have been considered to be member of the **Instruments Set**, it has been assign its own dedicated object, since it is a universal generic resource that enables attainment of all the other resources, and all other resources consumption can be expressed by or converted into units of **Budget**.

Each resource type is utilized as defined by its **Utilization** attribute, which is measured using specified **Measurement Units**. The **Task Execution** process has an associated **Duration** attribute, which is defined by **Minimum Activation Time**, **Maximum Activation Time**, and **Units**. **Related Information Set** is linked to each specific **Task Execution** process, pertaining to specific **Requirements Set** and **Risk Set**. The **Deliverables Set**, which is the outcome of each **Task Execution**, may include deliverables such as product components, documents related to derived requirements, approvals, simulations, analyses, specifications, and other types of reports. These deliverables are classified under three roles: **document**, **component**, or **gate**. Each one of the deliverables results explicitly from a specific process and is used in a subsequent process, either as an instrument (usually if it is an informatical object), or as a consumee (a physical object that is consumed by or embedded into a larger component). The **Resources Set** and the **Deliverables Set** are described and exemplified in Table 3-1.

The GPC facilitates and streamlines the modelling of the entire project, as it provides a means for checking all the possible deliverables that can be generated by each activity and are required by one or more subsequent activities. The time for obtaining each deliverable can be calculated from the relevant process durations. The resulting project plan incorporates the product to be designed, manufactured, delivered, used, serviced, and disposed of, with the project, including all its significant deliverables, such as resources and their allocation, timetable, limits and milestones, evaluation of project constraints, and assertions related to its cost and duration.

7.1 Planning to System Builds

Planning the complete combined project-product includes the definition of Systems Builds (SBs), i.e., specific configurations of the system, planned to be achieved along the project lifecycle span. The methodology enables the planner to plan for a specific functionality (which is related to a set of requirement) or even a specific single requirement to be achieved in a certain SB.

There are two known strategies which are related to our proposed SBs approach: the *incremental* strategy and the *evolutionary* strategy. The incremental strategy [59] (called "Pre-planned Product Improvement" in DOD 5000.2), determines user needs and defines the system requirements, then performs the rest of the development in a sequence of builds. The first build incorporates part of the planned capabilities; the next build adds more capabilities, and so on, until the system is complete.

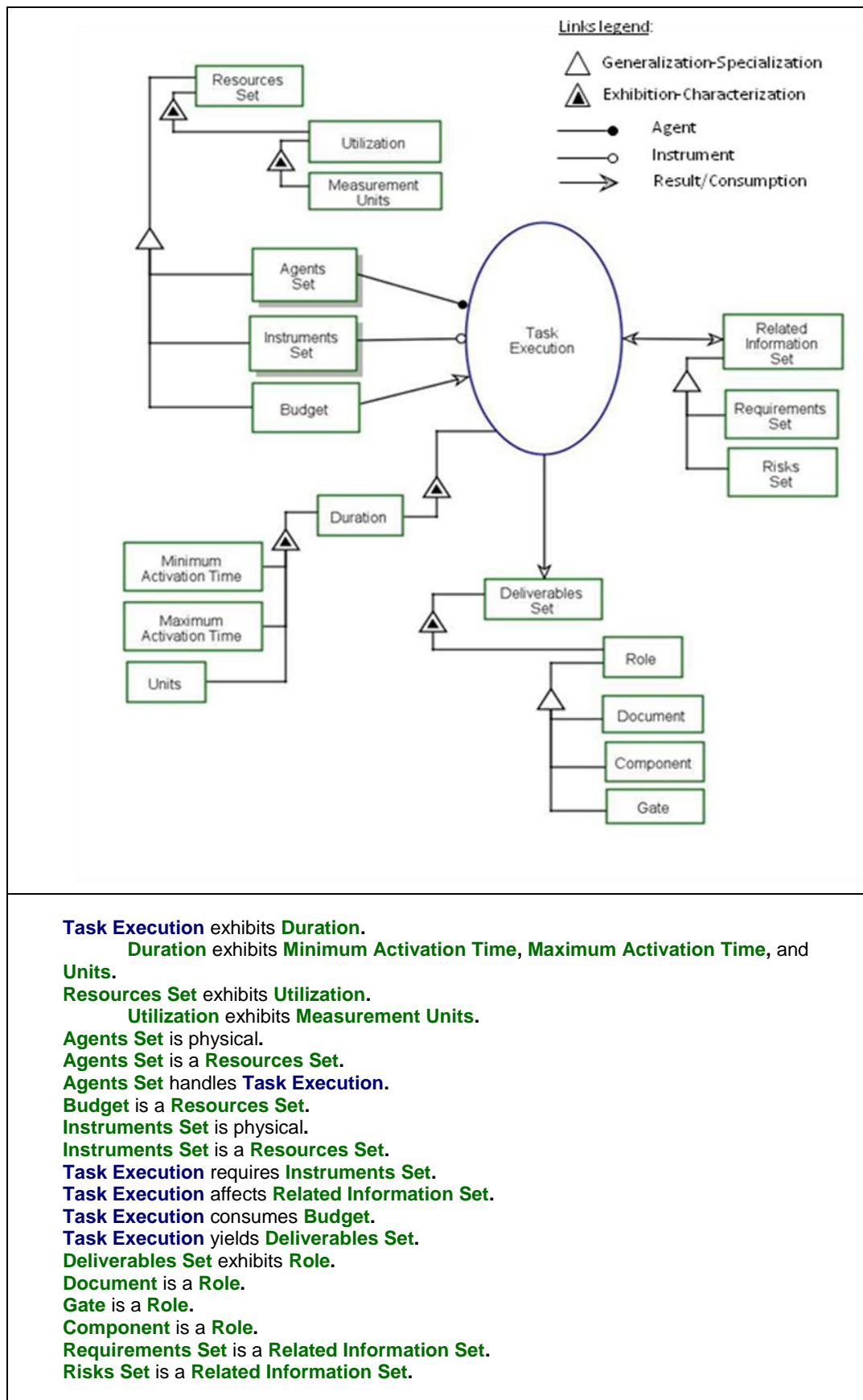


Figure 7-1 The Generic Project Construct (GPC) of the Model-Based Project Plan

The "evolutionary" strategy (called "evolutionary" in DOD Instructions) also develops a system in builds, but differs from the incremental strategy in acknowledging that the user need is not fully understood and all requirements cannot be defined up front. In this strategy, user needs and system requirements are partially defined up-front, and then refined in each succeeding build.

Our proposed SBs strategy resembles the incremental strategy, with one major difference: while in the "incremental" strategy the capabilities are built up on top of each other for each build, the SBs strategy facilitates the planning of capabilities to be achieved, not necessarily in an cumulative manner, but rather in a way that best serves the technological order and risk mitigation in each project.

The number of SBs planned for a project depends on the planner's technical planning experience and judgment. The SBs planning is derived from the system requirements, functionality, and architecture, accounting, as noted, for technological capabilities and risk mitigating. Some SBs may be planned to achieve certain functionality to be demonstrated due to binding milestones, while other SBs may be planned for mitigating technological risks along the planned development of the entire system to be delivered. In any case, the last SB is the complete product to be supplied by the project. The Product to be delivered is the unification of all capabilities previously proven successful in former SBs, reflecting all the system requirements and functions that were realized in each one of the System Builds in the SB set. The set of SB-related definitions for the combined Product-Project methodology are listed in Table 7-1.

Table 7-1 The SB-related definitions for the combined Product-Project Methodology

	Term	Definition
1	Product (originally defined)	The final system as delivered to the customer.
2	Functionality	The set of requirements and their corresponding allocated set of functions.
3	Product Functionality	The functionality that needs to be realized in the delivered product.
4	System Build (SB)	A variant of the product (possibly also the delivered product), which realizes a certain subset of the product functionality, and has a lifecycle as a system in its own right.
5	System Build Functionality	The subset of Product Functionality that is realized by the System Build.
6	System Build Set	The set of all the SBs , the union of the functionalities of which realizes the product functionality.
7	Product (redefined with SB)	The ultimate SB, which embodies the union of the individual SB functionalities, such that it delivers the product functionality.

Each SB is composed of specific Software (SW) Builds and specific Hardware (HW) builds, so as to provide the defined functionality to be achieved in each planned SB, as depicted in Figure 7-2.

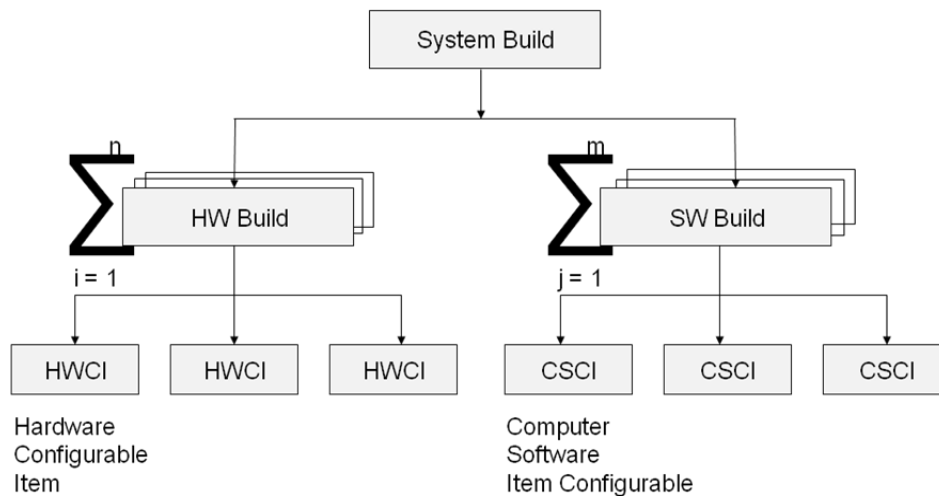


Figure 7-2 The combined product-project model System Build (SB) hierarchy

Figure 7-3 depicts the generic life cycle of a system build. Each SB is planned with its complete own lifecycle, which contains five phases: (1) Defining, (2) Developing, (3) Deploying, (4) Operating & maintaining, and (5) Retiring. **Error! Reference source not found.** presents an example of project plan containing five planned SBs, each with its own five phases. In such a multiple SBs plan, usually the *Retiring* phase of a specific SB is related to the *Deploying* phase of a successive SB. For example, the retiring phase of SB#1 can be installation in a specific lab site as an infrastructure for SB#5.

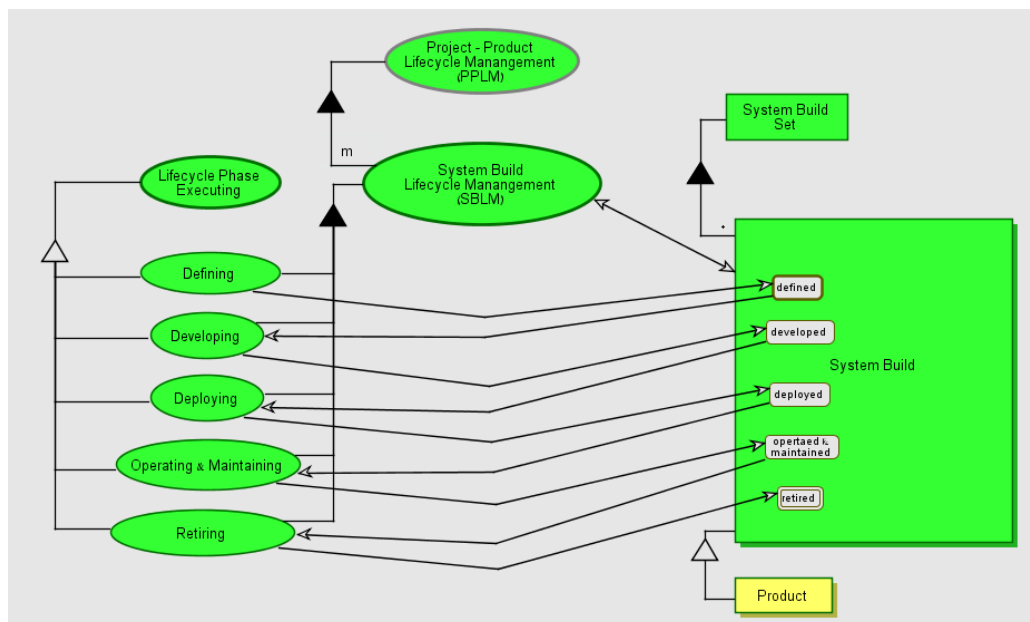


Figure 7-3 The generic lifecycle of a System Build

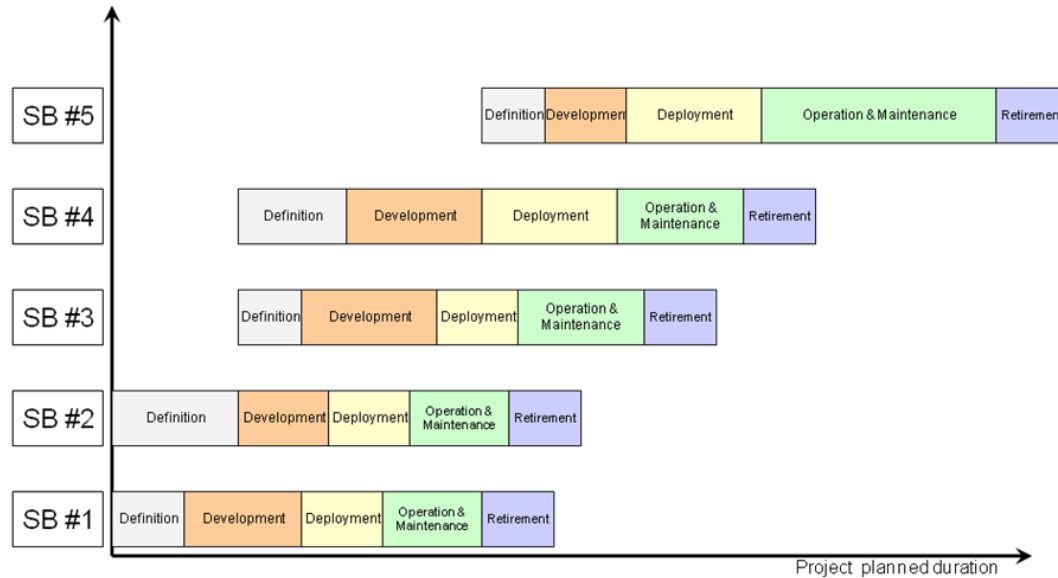


Figure 7-4 Five planned SBs spread over the project lifecycle

In the OPM-based PPLM model, each SB is defined by specific artifacts states, composing the exact functionality that is planned to be obtained by each cycle of developing & integrating as defined by the project processes modelling. Figure 7-5 depicts the OPM view of the PPLM model for a general **Developing** process. The **Developing** process is aimed to produce the three subsystems **Subsystem A**, **Subsystem B**, and **Subsystem C**, each in its complete state. All three subsystems comprise the **System Under Development (SUD)**, which exhibits five **System Functions**. The SUD is planned to be accomplished by four SBs: (1) **SUD SB 1**, planned for obtaining **System Function 1** and **System Function 4**, (2) **SUD SB 2**, planned for obtaining **System Function 2** and **System Function 3**, (3) **SUD SB 3**, planned for obtaining **System Function 5**, and (4) **SUD SB 4**, which is the final build, namely the complete delivered system, which has all five required **System Functions**. The **Developing** process starts on January 1 2009 and ends on December 31 2009. It begins with two simultaneously starting processes: **Developing Subsystem B** and **Developing Subsystem A**. The latter takes longer and ends at the end of the first year's quarter. When **Developing Subsystem A** ends, it transforms **Subsystem A** to its complete state, and when **Developing Subsystem B** ends, it transforms **Subsystem B** to its complete state. The complete states of both of these subsystems are required for **Integrating A + B** to start. **Integrating A + B** starts simultaneously with **Developing Subsystem C**, at the end of the first year's quarter, as they are both related to **Developing Subsystem A** in a Finish-to-Start relationship, as denoted by the horizontal dashed line. When **Integrating A + B** ends, the first system build – **SUD SB 1** – is completed, having **System Function 1** and **System Function 4** both achieved according to the plan. The dashed red line denoted as NOW indicates the current time.

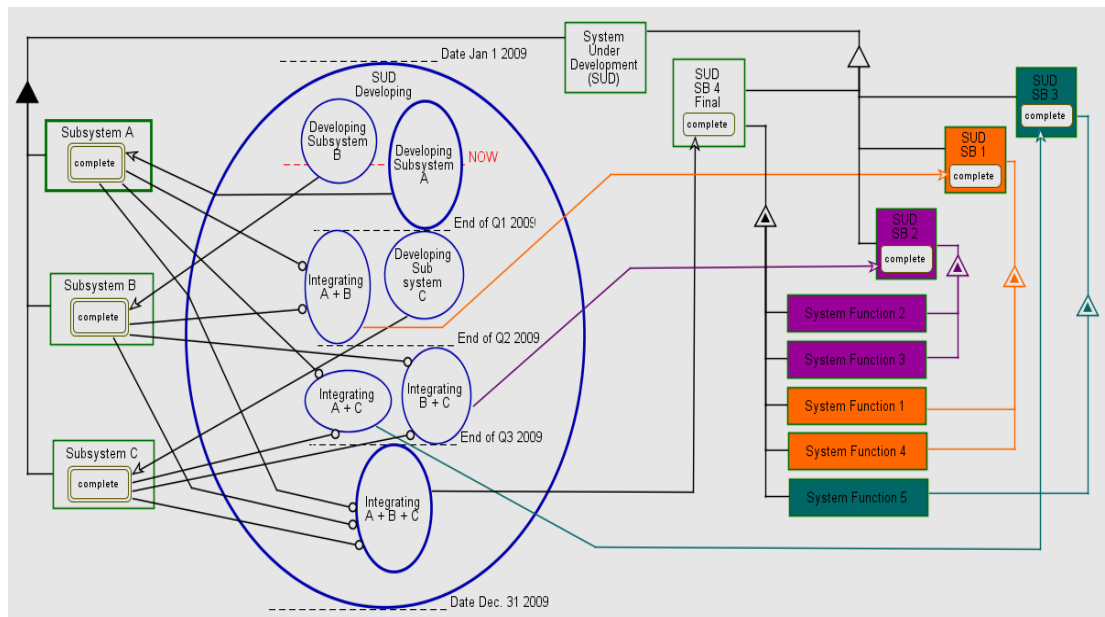


Figure 7-5 PPLM model of a SUD Developing process

Integrating B + C starts immediately when **Integrating A + B** ends, at the end of the second year's quarter, as denoted by the horizontal dashed line. For **Integrating B + C** to start, both **Subsystem A** and **Subsystem B** must be in their complete states. When **Integrating B + C** ends, the second system build – **SUD SB 2** – is complete, exhibiting **System Function 2** and **System Function 3** according to the plan.

Having the information of both the product and the project in the combined plan enables the following observation: Since **Integrating B + C** requires complete **Subsystem A** and complete **Subsystem B**, without them being integrated there is actually no need for the **Integrating B + C** to be executed after **Integrating A + B** ends. It can be planned to start earlier.

Integrating A + C floats within the duration of **Integrating B + C** (second year's quarter). **Integrating A + C** requires that both **Subsystem A** and **Subsystem C** be in their complete states. When **Integrating A + C** ends, the third system build – **SUD SB 3** – is completed, exhibiting **System Function 5**.

In this example each system build is planned for obtaining specific functions in a non-cumulative manner, i.e., it is not based on an incremental strategy.

Integrating A + B + C is planned to be executed as soon as **Integrating B + C** ends, at the end of the third year's quarter, as denoted by the horizontal dashed line. **Integrating A + B + C** start requires that all three **Subsystems** be complete. When **Integrating A + B + C** ends, the final system build – **SUD SB 4** – is completed, exhibiting all five **System Functions**.

Figure 7-6 presents the in-zoomed view of **Developing Subsystem A**, which is planned to be executed at the beginning of the **Developing** process shown in Figure 7-5. **Subsystem A** consists of a **Computer**, a **laser**, and a **Tracker**. The process of **Developing Subsystem A** begins with the simultaneous start of **Computer Developing** together with **Laser Developing**. The end of **Computer Developing** transforms the state of **Computer** from degraded to complete. When **Laser Developing** ends, it yields the **Laser**, just as **Tracker** is generated when **Laser Developing** ends.

The timing of the entire **Developing Subsystem A** process in Figure 7-6 is coherent with the upper level presented in Figure 7-5. The current time of the project, indicated

by the dashed red line denoted as NOW, is also compatible with upper level of this PPLM model.

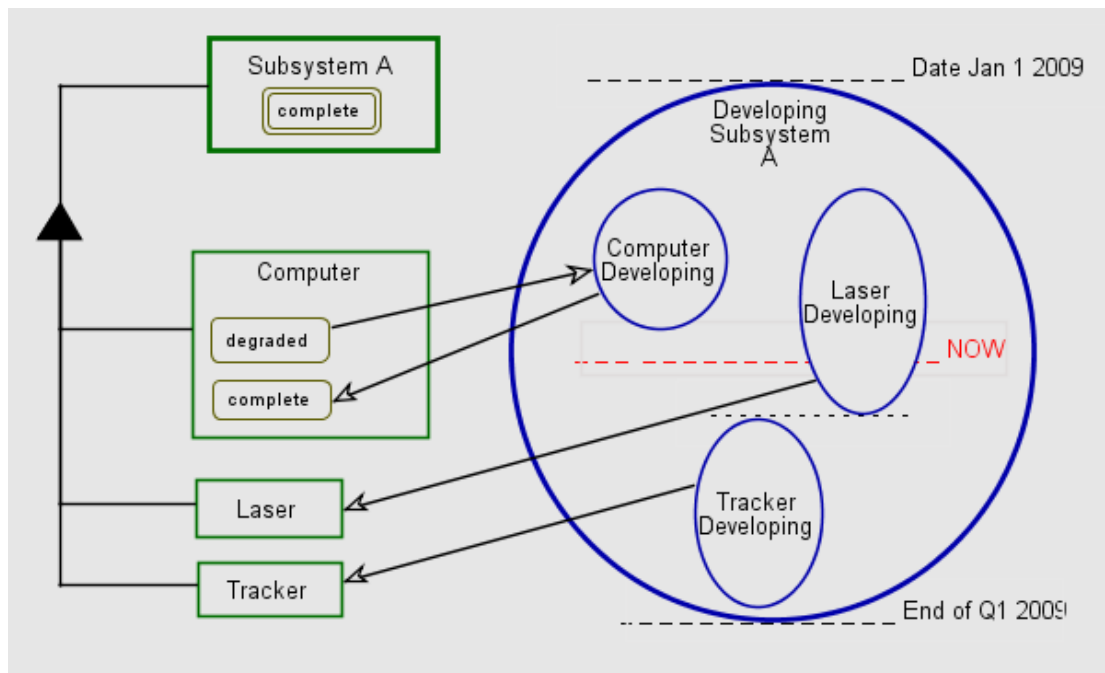


Figure 7-6 PPLM plan of Developing Subsystem A

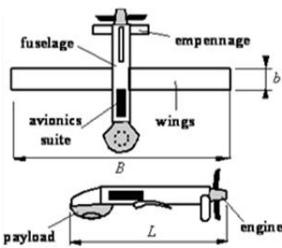
8 Perceived Differences between the Gantt and the PPLM OPM-based model

This research included three stages, as presented in Table 8-1:

- (1) Gantt-PPLM comparison among first participants group, based on a simplified UAV model which contains the project and the product in a freely combined manner,
- (2) Same as 1, with a second group and improved test based on lessons learned from (1), and
- (3) Elaborated Gantt-PPLM comparison among the second group, based on a simplified CT scanner model in which the project and product aspects were combined according to PPLM methodology.

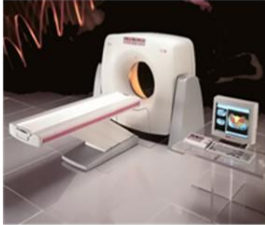
While in the first two stages the participants were asked to create the models, based on a given text specifying 23 tasks and their relationships, in the third stage the models – a PPLM model and a Gantt chart model – were given to the participants, who were asked to answer questions based on the provided models.

Table 8-1 The three research stages

Stage	Research Method
Stage (1) + Stage (2)	<p style="text-align: center;">Given UAV Specification</p>  <p style="text-align: center;">UAV concept, Specifications: L=2000 mm, B=3500 mm, b=500 mm</p> <p style="text-align: center;">23 activities specified with relationships</p> <div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;"> <p>↙</p> <p>Gantt Model created by participants</p> </div> <div style="text-align: center;"> <p>↘</p> <p>Project-Product OPM Model created by participants</p> </div> </div> <p style="text-align: center;">Project and product combined in a <u>free manner</u></p>

56

Table 8-2 The three research stages (continued)

Stage	Research Method
Stage (3)	<div style="text-align: center;"> <p>Given</p> <div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;"> <p>Gantt Model created by researcher</p> </div>  <div style="text-align: center;"> <p>Project-Product OPM Model created by researcher</p> </div> </div> <p>5 development cycles for 5 System Builds (SBs)</p> <p>↓</p> <p>30 questions – understanding of plan content answered by participants</p> <p>Project and product combined <u>according to the PPLM methodology</u></p> </div>

8.1 First Stage and Second Stage

8.1.1 Research Population and Setting

There were two research populations in this part of the research – one for each stage as presented in Table 8-3.

The first stage research population are the same 24 mid-career systems engineers, who were graduate students in the Systems Project Management course (see Appendix A and Appendix B), as detailed in paragraph 4.2. This first stage of the research was conducted within the fifth homework (HW5) in the course, same as detailed in paragraph 4.2. In this part of the homework (see Appendix C) the participants were requested to create two project plan versions by using two different methods: A Gantt chart model and an Object Processes Methodology (OPM) model, based strictly on the text given in former homework (HW2). The students were divided randomly into two groups, similar in number of participants; in one group the Gantt chart model has been created before the Object Processes Methodology (OPM) model, while in the second group the order was reversed. The students have been asked to do this in a specific order without backtracking and document their reflections on the way they did their project planning.

The second stage research population were 32 mid-career systems engineers from companies across Israel with 3-15 years of practice, who were graduate students at the systems engineering program of the Technion during the fall 2009. This second stage research group participants have been given a three hours lecture on the PPLM approach, followed by homework. In this homework (see Appendix D) they were

given the exact text given to the first stage group, and had been asked to create three project plan versions by using three different representations: (1) an Activities Network Plan (AON type) model, (2) a Gantt chart model, and (3) an Object Processes Methodology (OPM) model. The order for creating the models was not imposed. The participants were instructed to write their reflection of the process they had undergone while creating each model, and record their thoughts, assumptions and decisions during the modelling process.

Table 8-3 Research population and settings for stage (1) and stage (2)

Research Question	Stage	Research Population	Research Method
How are the differences between a Gantt model plan and a PPLM model plan reflected by systems engineers?	<u>Stage (1)</u>	24 participants graduate students at MIT	Structured questionnaires. Participants asked to model (1) OPM (2) Gantt chart Based on given text of UAV Project and product combined in a <u>free manner</u>
	<u>Stage (2)</u>	32 participants graduate students at the Technion	<div style="text-align: center;"> ↓ Lessons learned </div> <ul style="list-style-type: none"> • Time recording during modelling process excluded • OPM time notion emphasized

8.1.2 Research Methodology

The research was aimed to explore answers to our fourth research question: How are the differences between a Gantt model plan and a PPLM model plan reflected by systems engineers?

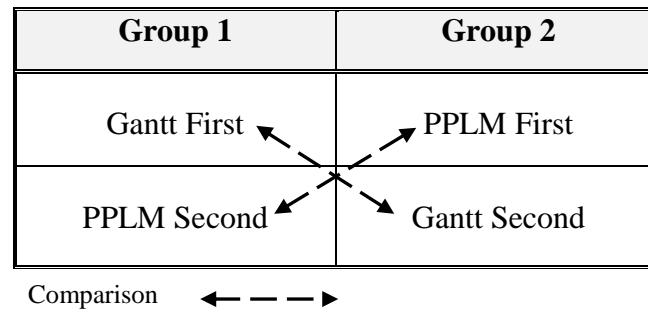
8.1.2.1 The designed comparisons

The first stage of the research was conducted using a structured assignment designed to explore the reflections of systems engineers regarding differences between a Gantt model plan and an OPM-based PPLM model. The participants were divided randomly into two groups. Group 1 was instructed to model the Gantt chart first, and then to create the PPLM model. Group 2 was instructed to do the same, but in reverse order. The assignment was designed to enable analysis by two comparisons of models between the two groups, as described in **Error! Reference source not found.:** Gantt to Gantt Comparison, and PPLM to PPLM Comparison, seeking for similarities and differences, some of which might be related to the order of model construction.

The influence of the order of model construction was also investigated based on the reflections that the participants were asked to document on what prompted them to introduce changes, additions or deletions in the second model (PPLM or Gantt chart)

with respect to the model they had prepared previously (Gantt chart or PPLM respectively) and record the time required for each stage of the modelling. The records participants made served for further investigation of their modelling process.

Table 8-4 Stage 1 participants group partition



The second stage of the research was conducted with two different groups of participants with the structured assignment used in the first stage, to which some changes were made, based on lessons learned from analyzing the results obtained in the first stage. The differences and similarities between the groups of the two stages are summarized in Table 8-5.

8.1.2.2 Pre-assignment knowledge among participants

Both stages' groups were introduced with the PPLM approach during class hours, although with different amounts. The UAV case study which was the basis for the structured questionnaires in both stages' groups was familiar to the first stage participants, since it was their fifth homework in the course, based on the specific case study used during the semester. The UAV case study was new to the second stage participants. In order to moderate this difference between the groups, the second stage group was instructed to conduct two tasks prior to the Gantt chart modelling and PPLM modelling – they were asked to construct a task table from the project description in “technological order”, and they were asked to create the project Activities Network Plan (ANP) graph (AON type) model. Both these tasks were contained in the second homework given during the course of the first stage group. Both groups did not receive any guidance regarding Gantt chart modelling, assuming this is a common planning tool used in their practice.

Table 8-5 Comparison between Stage 1 and Stage 2 research populations

Stage	Class hours devoted to introducing the PPLM approach	Prior knowledge of the UAV case study	Prior knowledge of OPM	Prior knowledge of Gantt chart	Knowledge of PPLM objects typology*	Research Method
Stage 1	90 min + 90 min Within the course titled “System Project Management”	Served as the basis for all the assignments throughout the entire course	Have prior acquaintance from a preceding course; Given short reminder	Assumed former acquaintance from practice	None	<ul style="list-style-type: none"> • Time recording required during modelling process • OPM time notion not emphasized • PPLM Objects typology not introduced

Table 8-6 Comparison between Stage 1 and Stage 2 research populations (continued)

Stage	Class hours devoted to introducing the PPLM approach	Prior knowledge of the UAV case study	Prior knowledge of OPM	Prior knowledge of Gantt chart	Knowledge of PPLM objects typology*	Research Method
Stage 2	~ 100 min Within the course titled "Integration and design verification of system product"	Non Given two pre-tasks which were included in former homework of stage(1) group	No prior acquaintance Given ~ 30 min introduction	Assumed former acquaintance from practice	None	<div> ↓ Lessons learned </div> <ul style="list-style-type: none"> • Time recording during modelling process excluded • OPM time notion emphasized • PPLM Objects typology not introduced

* See Table 3-1

8.1.2.3 The expected PPLM model

The text given to the participants (see Appendix C) explicitly contained a description of the project tasks, including the dependencies between tasks. The **Things** to be added to the model by the participants refer to both additional **processes** and required **objects**. Each one of the **objects** to be added, such as components, documents, approvals, simulations, analysis, specifications, etc., should result explicitly from a specific process and used (as instruments, or consumees) in a subsequent process. No instructions were given for specific use or non-use of states of objects in the model. The participants were not instructed to model in a specific methodology, but rather freely use their own sense to create a model in which the project and the product are combined.

Both groups were deliberately not introduced with the complete PPLM methodology, including the PPLM objects typology (given in Table 3-1). They added objects in the PPLM model, according to their own judgment, based on the given text, after of before creating the Gantt model.

There is no one single "correct" solution for the PPLM model. However, the model had to comply with the following basic requirements:

- Containing all the information provided in the text.
- Following the basic OPM rules, including:
 - Starting with a top level process at the first System Diagram (SD) of the model, representing the entire project.
 - Hierarchical decomposing using the in-zooming mechanism.
 - At each level, the processes should be laid from top downwards, indicating the technological order of sub-processes.
- Containing additional OPM things (objects and processes), as required for a complete model.

8.1.2.4 Objects classification and count

The participants were not introduced to the PPLM object types (see Table 3-1) since our goal was to gauge the “natural” systems engineers approach for adding objects in the model rather than enforcing our objects typology. The analysis of the PPLM models produced by the participants was based on classifying the objects contained in each model according to the typology (see Table 3-1), with each object classified into a single type.

Since no specific instructions were given for using states of objects, some of the participants have used states in their model, while others have not. Therefore, whenever states were used we counted each state as a separate equivalent object, as depicted in Figure 8-1. The figure shows on the left an **Engine** which can be in one of four states: designed, developed, tested, or integrated. The equivalent is presented on the right – four separate engine objects, each named in accordance with the states on the left.

The major expected results are presented in Table 8-7.

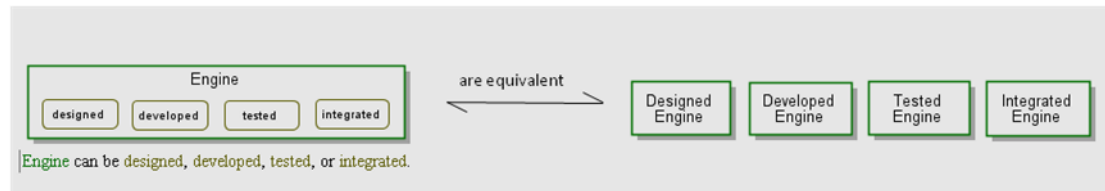


Figure 8-1 Object’s states and equivalent objects

Table 8-7 Expected results

	Expected result	Rationale
1	<u>Time to produce the models</u> : Longer time to produce the PPLM model compared with time to produce the Gantt chart.	Either before or after Gantt chart modelling, OPM modelling requires adding information - both additional processes and required objects.
2	<u>Details level</u> :	
2.1	Gantt chart modeled after PPLM will contain more details than Gantt chart modeled before PPLM – milestones, cluster hammocks.	Influenced by the PPLM modelling, which enforces adding information on top of the given 23 tasks.
2.2	In both Gantt and PPLM: at least 21 processes.	Excluding the first task (project start) and the last task (project end) → 21 processes. Possible additional parent processes – clusters/hammocks.
2.3	In PPLM model: at least 21 objects.	Excluding the first task (project start) and the last task (project end) → 21 processes. Each one should have at least one deliverable of either type (D, C, or G), otherwise the process is not productive.
2.4	In PPLM model: more G type objects than other types*.	As equivalents to commonly used milestones in Gantt charts.

* See Table 3-1

8.1.3 Results and Analysis

8.1.3.1 Stage 1

The results and analysis presented in this paragraph follow the order of the expected results, as listed in Table 8 5.

8.1.3.1.1 *Time to produce the models*

The time to produce each one of the models – the PPLM model and the Gantt chart model –was reported by the participants as part of the assignment. Figure 8-2 shows the maximal time reported for each model, as well as the minimal reported times. The figure also shows calculated median and standard deviation (SDT) of the reported times for each model. The time for producing the PPLM model varies between 7.5 hours (max value) and 0.75 hours (min value), while the time for producing the Gantt chart varies between 1.5 hours (max value) and 0.33 hours (min value). As expected, based on the medians results, the time it took the participants to produce the PPLM model is significantly longer than the time it took them to produce the Gantt chart model, based on the same given text: while the median of times to produce the PPLM model is 2.5 hours, the median of times to produce the Gantt chart is 0.75 hours. The results for the standard deviation of the reported times is also not surprising: while the SDT of the times for producing the PPLM model is 1.8 hours, the SDT for the Gantt chart times is 0.3 hours. The SDT results show that the times to model the provided text into Gantt chart does not vary much from one participant to another. The fact that it took about the same time for all participants to produce the Gantt chart is explained by the reflections provided by the participants on the process of producing the two models: the majority of participants wrote that producing the Gantt chart was “pretty easy” and “straightforward” from the text, “without any assumptions to be made”, removing the reason for much time variability.

Comparing the time results for the groups, Group 1 – PPLM model first followed by Gantt chart, and Group 2 – Gantt chart first followed by PPLM model, showed no significant difference between the two groups, i.e., the results presented in Figure 8-2 are applicable for both groups, with no difference due to modelling order.

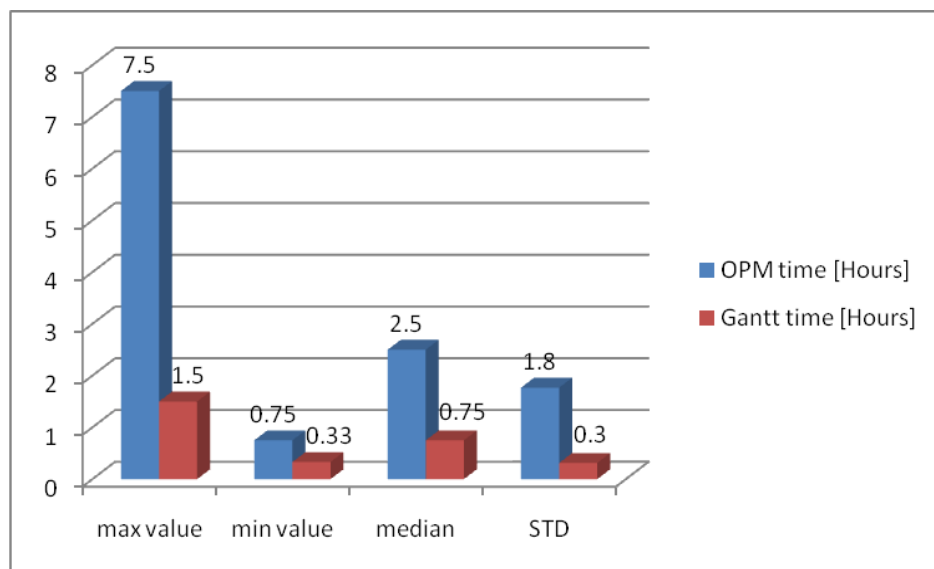


Figure 8-2 Time to produce Gantt chart and PPLM models

8.1.3.1.2 Detail level

The results and analysis of the detail level are presented in this paragraph, first for the Gantt chart model, followed by the PPLM model – both in regard to the expected results, as listed in Table 8 5. In addition, a comparison of Stage 1 vs. Stage 2 is discussed, concluding with identified required changes for Stage 3.

8.1.3.1.2.1 Gantt chart model

We expected that the Gantt chart modeled after the PPLM modelling would contain more details compared to the Gantt chart modeled first. Since the Gantt chart can be produced easily based on the given text, we expected that the Gantt chart modeled first will only contain the given 21 tasks and their relationships, and two milestones – *project start* and *project end* (both assigned zero duration). On the other hand, we expected that the Gantt chart produced after modelling the same text into OPM, will be influenced by the PPLM model: Added cluster processes in OPM would be turned into hammocks in the Gantt chart, while added objects in OPM would be reflected as milestones in the Gantt chart. However, all Gantt charts produced by the participants were identical. They all contained the given 21 tasks and their relationships, and the two milestones.

Based on the reflections provided by the participants on the Gantt chart modelling process, they felt the provided text contained all the required information for creating the Gantt chart. Therefore, no assumption was made nor was data added to the Gantt chart other than what was given in the text.

8.1.3.1.2.2 PPLM model

The number of processes and objects in the PPLM model are depicted in Figure 8-3. The total number of processes varies between 31 and 14 processes, while the median result is exactly 23 processes. Indeed, in most of the cases, the PPLM models contained the 23 tasks which were provided in the text. Only part of the participants included abstraction by clustering processes (using the OPM in-zoom mechanism). A small part of the participants did not include the minimum expected 21 tasks in the model. As expected, some of the participants did not model the first task (project start) and the last task (project end) as processes but rather used objects for modelling them.

The total number of objects in the PPLM model varies between 44 and 2 objects, while the median result is 21.5 (see Figure 8-3). The median indicated that the majority of participants did model at least 21 objects, as expected. Each one of the 21 processes (remaining processes after excluding the project start and project end) should have at least one deliverable, otherwise the process is not productive. There were very few cases in which only few objects were modeled. In some cases, the model contained twice as much as 21 objects. In these cases, the high number of objects is not because more than a single object was modeled for some of the processes, but because some of the components were modeled as stateful. Each stateful object was counted as several objects, as explained in Subsection 8.1.28.1.2.4.

The standard deviation (STD in Figure 8-3) of the objects number (10.6) is almost three times bigger than the standard deviation of the processes number (3.9). This outcome is in accord with the previously presented results: while there was little

difference in modelling the processes, modelling the object in the PPLM model was much more diversified in terms of number of objects modelled.

The distribution of object types in the model is presented in Figure 8-4. The percentage of each object type is calculated based on the averages obtained for each type, calibrated such that 100% is the sum of averages of all objects types (17 objects). *Equivalent* in the legend denotes the total number of objects from specific type, when objects are counted as explained in Paragraph 8.1.28.1.2.4. The majority of objects are C (Component) type – 48%, followed by D (Document) type – 34%. Two of the participants included a single Budget type object in their models (0.1%). A (Agent) type objects are 6% of the entire objects, while I (Instruments) type are 5%. The Budget together with the A type and I type are the resources (see Paragraph 7.1), which together account for 11% of the overall objects number. The G (Gates) type objects, which we expected to be dominant, were found to be only 7% of the total number of objects.

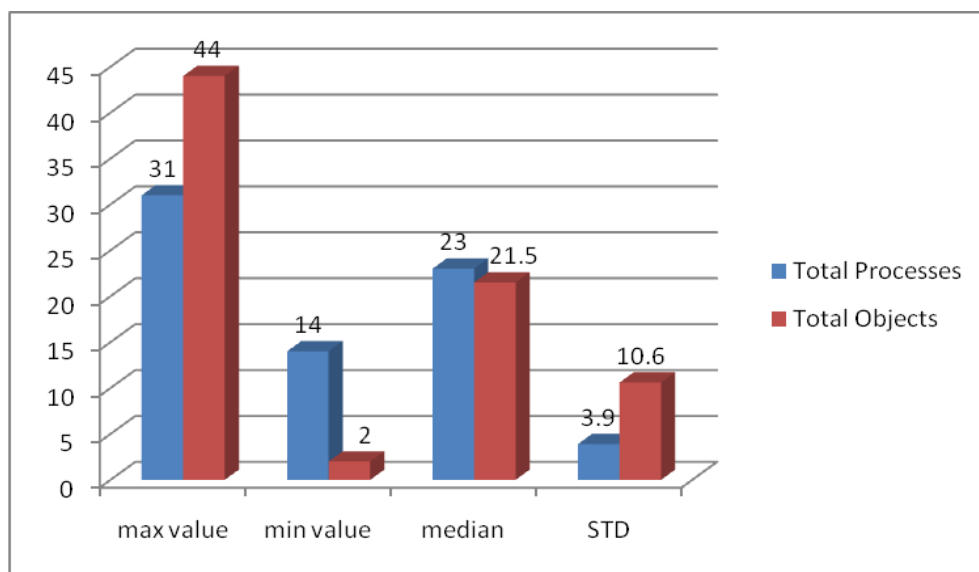


Figure 8-3 Number of processes and objects in the PPLM model – Stage 1

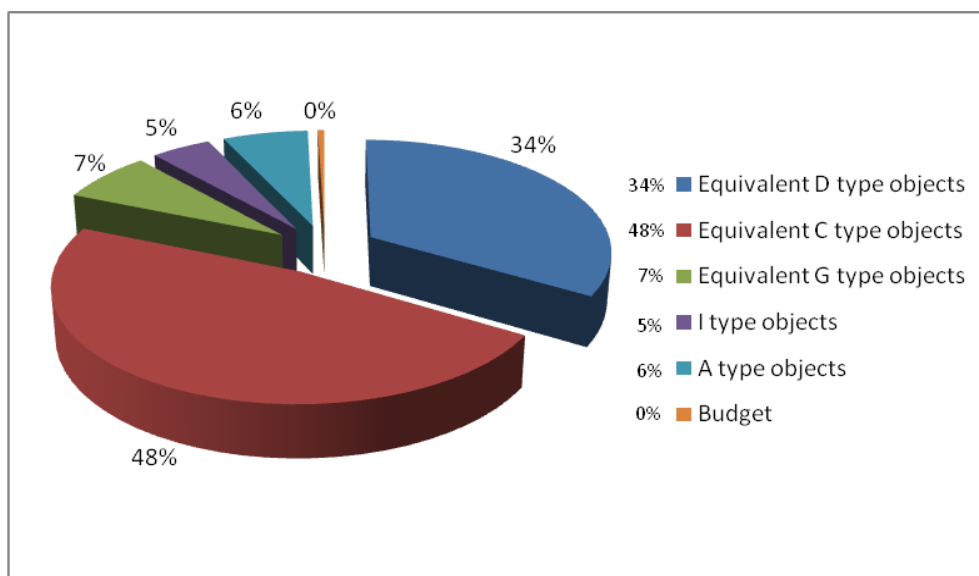


Figure 8-4 Averages of objects quantity distribution by types – Stage 1

One third of the participants (8 out of the 24) in Stage 1 used states in their PPLM model. These participants modeled states of C type objects, and half of them modeled also states to D type objects. Only one participant modeled G type stateful objects. About third of the participants (9 out of 24) created the PPLM model using the in-zoom. However, the use of the in-zoom mechanism does not comply with the OPM methodology.

8.1.3.1.3 General conclusions from participants' reflections

Some selected reflections provided by the participants, are presented in this paragraph: first for the Gantt chart model, followed by the PPLM model, through PPLM vs. Gantt reflection, and concluding with reflections related to lack of adequate visualization of time in the OPM model.

8.1.3.1.3.1 The Gantt chart model

All the participants reported on the ease of creating the Gantt chart, based on the given text, in which the tasks along with their durations and relationships were included:

"While building the [Gantt] model, the most critical aspect was figuring out task dependencies. I did not feel that fully understanding the nature of the task or its end results were very significant for building the model. Overall, there were three items I needed for each task: 1 - task duration, 2 - task dependencies, 3 - task resources (though not very significant for the model's structure).

When building the model I was only interested in the task itself and what it depended on. It was not very important to understand where the task fits within the overall project." [Subject 17]

"It took me 0.5 hours to develop the Gantt chart in MS project. Per the given information, the step and time duration of each step, as well as the dependency are very clear, and easily organized into the Gantt chart... I did not make any additional assumption beside the given information." [Subject 18]

"The [Gantt] model was very easy to create, no need for intermediate objects to be created, only the sequence matter and Microsoft Project worked very well for it." [Subject 24]

Some of the participants provided judgment of the Gantt chart utility, as contained in the following phrasing:

"It [the Gantt chart] is a general timeline which helps assessing the duration of the project, and the general outline." [Subject 11]

8.1.3.1.3.2 The PPLM model

Many participants indicated the longer time spent on producing the PPLM model, but reported added value to the effort of including both the processes and the objects in the combined PPLM model:

"While the [PPLM] model took significantly more time, I found it quite interesting that the end result gave me an insight on both the project process and the system

being built (product). Throughout the process I was also very much engaged in understanding the outcome of each task and the resulting states. Overall it is interesting to observe that through OPD I was able to both model the project flow and the outcomes.” [Subject 17]

“It took significantly longer to make an OPM model of the project (probably partly because I was not close to as comfortable with OPCAT as I was with Project)...I had to make some more assumptions when using OPCAT. I decided to model the informational and physical aspects of the project as separate objects (i.e. the “software” was treated as one physical object, while the “software design” was a separate informational object). I also had to make assumptions for the state and process for each task “grouping.” In the end it generally ended up that I modeled each task from HW2 as a set of an object with two states and a process that changed the object from one state to the next. When viewing the completed project model, there is more information presented in the OPM model as far as interactions and statuses, however the Gantt chart better shows the timing of tasks since those are contained within the properties box on OPCAT.” [Subject 12]

“It took me about 3 hours to complete the OPM of the UAV product System as shown above. Clearly, the project plan has been divided into five high level processes...Each of these high level processes are further subdivided into processes as shown in the OPM above. Each sub-process, for example “Develop Engine Specification” depends on a deliverable (UAV Requirements document in this case as a result of previous process) and results in a deliverable namely, Engine Specification in the above model. All tasks in the project are thus mapped into process and object deliverables.” [Subject 14]

“It took me 5 hours to develop OPM model. I regrouped the tasks on the basis of task content such as design spec defining, designing and developing, integrating and assembling, as well as testing. These meta-groups have clear sequences. The previous group tasks are done, and then the next group tasks can begin.” [Subject 18]

Some of the participants explicitly phrased the added value, in their opinion, from the combined OPM based project-product plan they had produced with the OPCAT software:

“In the process of writing the OPM, you need to think of operators like ECC and states of objects in a project... I used the creation of objects like Engine Specification or Payload Specification as a deliverable of an activity to later be used by consecutive process as a pre-requisite. The good thing about this diagram is that it pictures what process affects on a particular object (deliverable) and which processes depend on those objects (previous activity deliverable).” [Subject 24]

“What I did like about this tool [OPCAT], is the strong relationship between objects and processes. I think it simplifies the way one thinks about a project. It makes the stages more clear. You know that at the end of each process there is an object to create, and it feels like little milestones, that helps define the step by step execution of the project.” [Subject 11]

8.1.3.1.3.3 PPLM vs. Gantt

Many participants included a comparison of the two models they had produced, as part of their reflection on the modelling process, expressing their criticism on both the modelling process and the produced models. A single participant was entirely in favor of the Gantt chart:

“After my experiencing with both softwares, I believe the MS Project can demonstrate most critical factors of a project much more explicitly than OCPAT.”
[Subject 18]

All the other participants provided a more balanced comparison, overall favoring OPM and indicating its advantages over Gantt. Following are a few excerpts of students' reflections.

“The OPM model adds another perspective to the CPM model in HW2. Despite the time to create, the OPM is quite useful and best describes the project plan and its resulting objects... the OPM model is a much more of a comprehensive representation of the project and its corresponding process...The Gantt chart should only be used for analyzing higher level tasks when schedule is the managers only concern.” [Subject 10]

“OPM also triggers the need to identify the output or object, and incorporate that as a deliverable, this can be reflected in the Gantt chart. By doing so, all stakeholders in the Gantt chart will understand the deliverable being received and also the deliverable being delivered downstream...OPM can assist to reflect which task should be completed first before engaging in subsequent task. In this example, the task list in diagram 3 would be completed first, and then the link will continue in diagram 2. Diagram 2 would of course be initiated first due to Requirement Definition task, but the next step would be to branch into this task, and as the micro task of Requirement Definition is completed, then the Gantt chart should refer back to Diagram 2 for subsequent tasks.” [Subject 9]

“I guess the concept of these 2 tools is very different, and concentrate on different aspects of a project. While Gantt gives you a general idea about the project schedule, the OPM model represents the project as a relationship between objects and processes that need to be accomplished. It helps understanding the different stages of development, and makes it simpler... It makes sense to me to have an OPM of the product first, specify each component, and its function and then map it directly to the project OPM, making sure that the product is being built correctly, timely manner.” [Subject 11]

“The Gantt was directly created from the Task details in HW2, using the task durations and Early Start and Early Finish times for the project. It's an interesting observation that, remarkably enough the OPM diagram (above) has captured much more detail compared to the Gantt (below). In the Gantt the object and process relationships are really embedded within each task, there is no clear way to differentiate what really are the deliverables at each stage of the project... There is also no straight way to figure out which tasks can be grouped together for iterations.” [Subject 14]

So far, from Figure 6 [OPM], it does not appear that I can visualize the project timeline, nor the start dates. If you recall, I indicated that once I entered the project start date, and the task dependencies and durations, the Gantt chart calculated the ES and EF of each of my tasks, and showed them graphically with blue bars. It is helpful for me to visualize the schedule part of the iron triangle (cost, schedule, scope) in that way. However, it is better for me to visualize the scope part of the iron triangle through the OPCAT model in figure 1. Displaying the tasks and resources separately makes it very easy to see who/what is needed and when. ... The major change from the Gantt to the OPM is that I created objects that signify the end of a process. All tasks in a real project have a deliverable. Using these objects to indicate the deliverables, what creates them and who requires them is extremely useful.” [Subject 13]

“The change that I made to the OPM model that I did not have in the Gantt chart was the NMA facilities. One explanation would be that I just picked out something in the second reading of the project description that I did not see in the previous reading when I created the Gantt chart. However, I would suggest that this was not the only reason. Making a pictorial distinction between the objects and processes enabled me to think harder as to what other ‘resources’ could be needed. When I thought about this, I then noticed that it said ‘NMA Facility’ in the project description.” [Subject 23]

8.1.3.1.3.4 Insufficient visualization of process durations in the PPLM model

Several participants related to lack of adequate visualization of time in the PPLM model.

“The OPM diagram did not seem to show the project flow well in the time domain.” [Subject 3]

“There is no duration task list to determine total duration or estimated duration in OPCAT.” [Subject 9]

“In the OPM...the main difficulty is to identify the flow just by viewing the diagram, there is no sequential order easily seen.” [Subject 24]

“I found that even for each process, OPCAT can track the start time and end time, but, compared with the MS Project, it is difficult to use OPCAT to demonstrate critical path, early finish (EF) time, late finish (LF) time, as well as the total slack (TS) and the free slack (FS).” [Subject 18]

8.1.3.1.4 Discussion and Identified required changes for stage 2

As expected, the time to produce from the same text of project specification the PPLM model was significantly longer than the time to produce the Gantt chart. The general sense from the participants’ reflection, though, is that the majority of them felt there was added value to the project plan ascribed to the larger time spent in order to produce the PPLM model.

The number of processes in the Gantt chart, 23, is identical to the number of the processes in the PPLM model and identical to the number of tasks in the provided text, indicating that, at least for the specific UAV assignment, neither one of the modelling processes had promoted the enrichment of the model beyond the given text. The comprehension of the project presented in either the Gantt chart or the PPLM model would likely be the same, assuming one is familiar with both representations. Both contain the same information with respect to the project processes.

In the PPLM model, we expected the G (Gates) type objects, which are equivalent to the commonly used milestones in Gantt charts, to be dominant in quantity over the other objects types. However, only 7% of the objects in the PPLM models produced by the participants were of G type. Instead, the C (Components) type were found dominant - 48%, followed by D type (Document) - 34%. In other words, the majority of objects participants added were Components and Documents – modeled as deliverables, required by subsequent processes in the model. Furthermore, Components and Documents were the only object types modeled with states (except for a single case of G type stateful object). This surprising result is very encouraging since it suggests that the combination of the project domain – the processes in the model, with the product domain – the objects in the PPLM model, is more powerful than we expected. This is in line with the findings presented in Section 4.6, which support the notion of the project and the product as two complementary facets of systems engineering management.

Two findings of the first stage triggered changes in the second stage. The first is that no differences were found between the Gantt charts of the two groups, therefore the partition into two groups (shown in Table 8-5) was abandoned. The second is that examining the written reflections of the participants revealed that they had thought that time to complete the assignment was the parameter tested in our research. Therefore, in the second stage we removed the request to record the time. Examination of the PPLM models of the first stage revealed that the participants did not use the OPM time notion properly, which is an essential characteristic of an OPM model in general, and a PPLM OPM-based model in particular. In order to cope with the first potential reason, while teaching OPM in the second stage of the research, we emphasized the in-zoom mechanism and its use with respect to expressing time concepts.

In the first stage, participants did not receive any guidance regarding Gantt chart modelling, assuming this is a common planning tool used in their practice. The same was followed in the second stage group.

In the first stage a short reminder of OPM was given in the class, since all the students were supposed to have prior acquaintance with the method. Later on, this was found to be the case for the majority of the participants. Therefore, a 30 min introduction to OPM was given to the second stage group.

8.1.3.2 Stage 2

8.1.3.2.1 Detail level

The results and analysis of the detail level are presented in this paragraph, first for the Gantt chart model, followed by the PPLM model – both in regard to the expected results, as listed in Table 8 5. In addition, a comparison of Stage 1 vs. Stage 2 is discussed concluding with identified required changes for Stage 3.

8.1.3.2.1.1 Gantt chart model

Based on the results of stage 1, we expected the Gantt chart models of the participants to be the same, containing the given 21 tasks and their relationships, and two milestones – *project start* and *project end* (both assigned zero duration). The results matched the expectations. The participants did not add any information in the model such as hammocks or additional milestones. Based on the reflections provided by the participants, they felt the creation of the Gantt chart from the provided text is a straightforward assignment. Therefore, as in stage 1, no assumption was made nor was data added to the Gantt chart other than what was given in the text.

8.1.3.2.1.2 PPLM model

The numbers of processes and objects in the OPM model are depicted in Figure 8-5. The total number of processes varies between 32 and 6 processes, while the median result is exactly 23 processes. Indeed, in most of the cases, as in stage 1, the PPLM models contained the 23 tasks which were provided in the text. Only part of the participants included abstraction by clustering processes (using the OPM in-zoom mechanism). Few participants did not include the minimum expected 21 tasks in the model. As expected, some of the participants did not model the first task (project start) and the last task (project end) as processes but rather used objects for modelling them.

The total number of objects in the PPLM model varies between 35 and 2 objects, while the median is 20 (see Figure 8-5). In cases where more than 20 objects are included in the model, the number of objects is a result of objects that were modeled as stateful; each stateful object was counted as several objects, as explained in Subsection 8.1.28.1.2.4.

The standard deviation (STD in Figure 8-5) of the objects number (7.6) is only slightly bigger than the standard deviation of the processes number (6.0). The distribution of object types in the model is presented in Figure 8-6. The percentage of each object type is calculated based on the averages obtained for each type, calibrated such that 100% is the sum of averages of all objects types (27 objects).

Equivalent in the legend denotes the total number of objects of a specific type, when objects are counted as explained in Paragraph 8.1.2.4. The majority of objects are C (Component) type – 42%, followed by D (Document) type – 21%. Budget type objects account for 5% of the total number of objects. Five participants included a single Budget type object in their models, and one of them modeled it as stateful. A (Agent) type objects account for 11% of the entire number of objects, while I (Instrument) type are 7%. The Budget together with A type and I type objects are the resources (see Paragraph 7.1), which together account for 23% of the overall objects number. The G (Gates) type objects, which we expected to be dominant, were found to make up only 14% of the total number of objects.

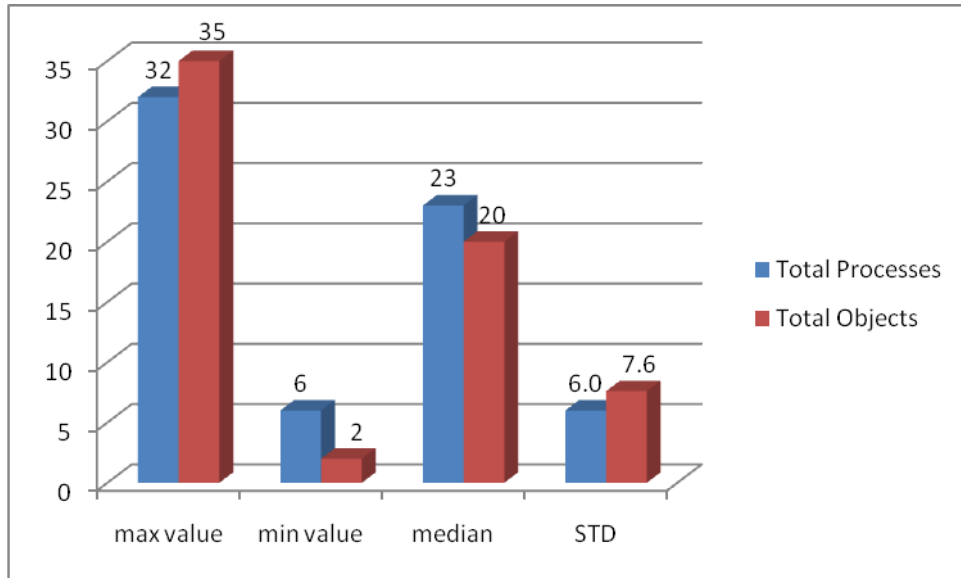


Figure 8-5 Number of processes and objects in the PPLM model – Stage 2

About half of the participants in Stage 2 (52%) used states in their PPLM model. These participants modeled states of C type objects, and one third of them modeled also states to D type objects. Only one participant modeled a Budget type stateful object.

85% of the participants used the in-zooming mechanism in their PPLM models. However, the use of the in-zoom mechanism often does not comply with the OPM methodology.

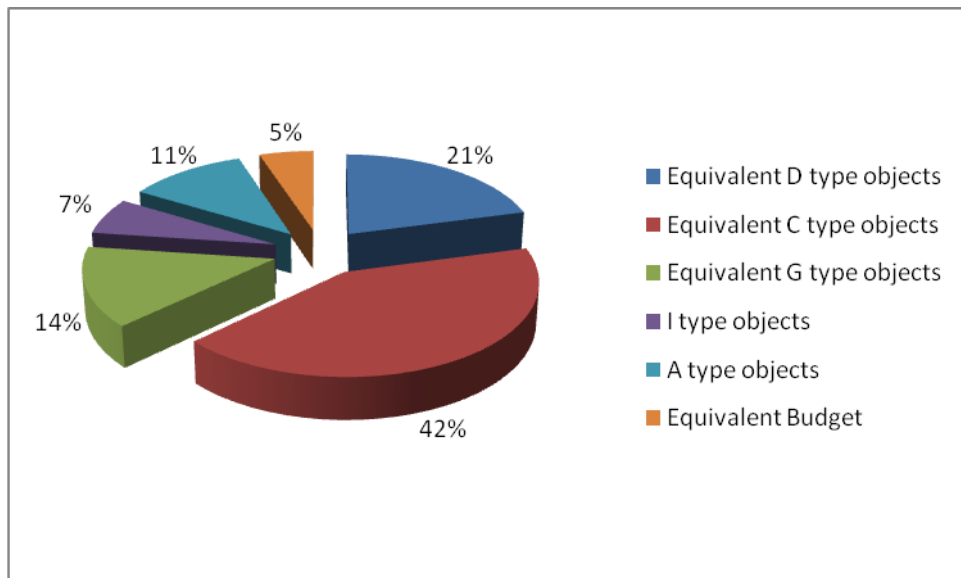


Figure 8-6 Medians of objects quantity distribution by types – Stage 2

8.1.3.3 Stage 1 vs. Stage 2 discussion and Identified required changes for Stage 3

In both stages the number of entities (tasks and milestones) in the Gantt chart is 23, which is identical to the number of tasks in the text provided to the participants. Based on the participants' reflections, this can be explained by the fact that the text contained the tasks and their relationships in a straightforward manner, which did not encourage the participants to question or investigate deeper. The assignment enabled the coverage of the processes in both models with little effort. Based on our professional experience in the field of project planning, we suspect that a more profound reason for this is the common belief that capturing the tasks and their relationships, if conducted thoroughly and correctly, guarantees the construction of a "good" project plan.

The PPLM model results for the two stages can be compared based on the processes and on the objects. Figure 8-7 shows the differences between the processes in the two stages. The maximal number of processes in both stages is almost the same (32 in stage 2 vs. 31 in stage 1), while the minimal number of processes is very different: 6 in stage 2 vs. 14 in stage 1. However, the median is identical in both stages – 23, and is exactly the total number of tasks contained in the text provided to the participants.

The standard deviation (STD in Figure 8-7) of the processes number in stage 2 (6.0) is almost twice as much as the standard deviation in stage 1 (3.9). This outcome is in accord with the minimal values (6 in stage 2 vs. 14 in stage 1). This result is due to the fact that more participants in stage 2 did not include in their PPLM model the minimal expected number of processes compared with stage 1.

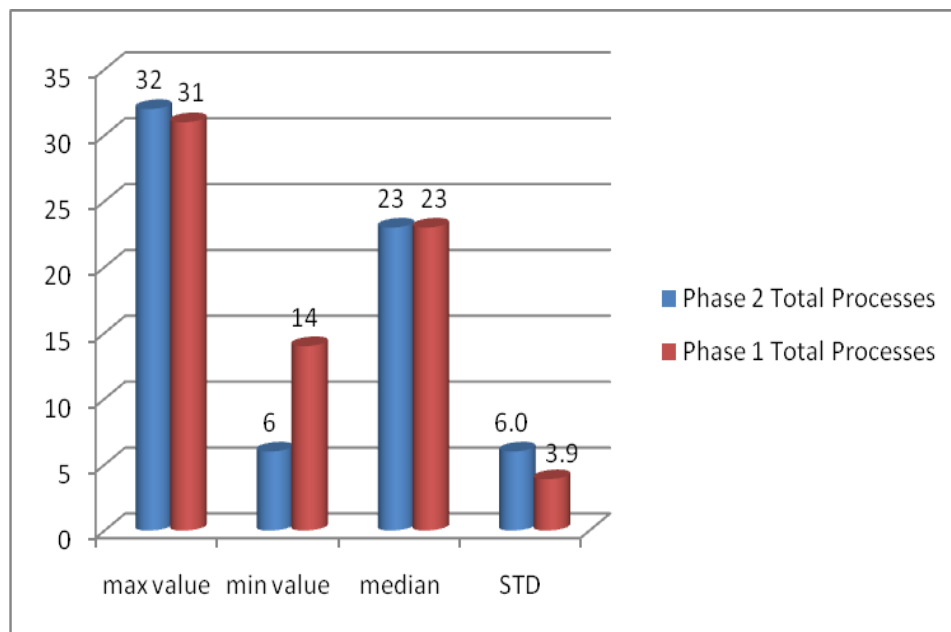


Figure 8-7 Total processes number comparison - Stage 2 against Stage 1

Figure 8-8 shows the differences between the objects in the two stages. The maximal number of objects in stage 2 is about 20% lower than that in stage 1, (35 vs. 44, while the minimal number of objects, 2, is the same in both. The medians in both stages are close: 20 in stage 2 vs. 21.5 in stage 1, and are close to the minimal expected number of objects, 21.

The standard deviation (STD in Figure 8-8) of the objects number in stage 2 (7.6) is about 30% lower than the standard deviation in stage 1 (10.6). This outcome is in accord with the maximal values (35 in stage 2 vs. 44 in stage 1).

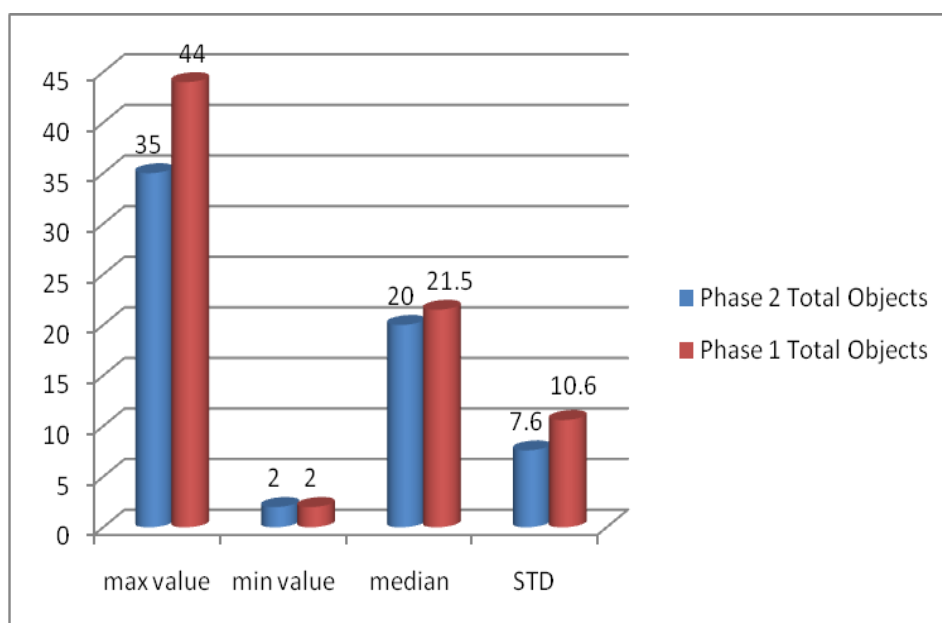


Figure 8-8 Total objects number comparison - Stage 2 against Stage1

Figure 8-9 shows the percentage (values indicated next to each bar) of objects types in stage 2 against stage 1. In both stages the majority of objects are C (Component) type. The difference in percentage of C type components in both stages is small – 42% in stage 2 vs. 48% in stage 1. In both stages the second higher percentage is of the D (Document) type – 21% in stage 2 vs. 34% in stage 1, although the difference between the two stages is higher for the D type than for the C type objects. The participants in stage 2 modeled less D type objects than the participants in stage 1. The percentage of I (Instruments) type objects is very similar in both stages, 7% in stage 2 vs. 5% in stage 1. The A (Agent) type objects in stage 2 are almost twice as much as those in stage 1, 11% in stage 2 vs. 6% in stage 1. Budget type objects in stage are negligible, 0%, while their percentage in stage 2 is 5%. The Budget together with the A type and I type are the resources (see Paragraph 7.1), which together account for 23% in stage 2 vs. 11% in stage 1. This indicates that the participants in stage 2 modeled more resource objects than the participants in stage 1. The G (Gates) type objects in stage 2 (14%) are double than those in stage 1 (7%). Nevertheless, in both stages the percentage of G type objects is low and does not match our expectation. This consistent result suggests that the combination of the project domain – the processes in the model, with the product domain – the objects in the PPLM model, is more powerful than we expected; the participating systems engineers chose to add objects to the given processes according to their own intuitive judgment. While we expected them to add more G type components, as equivalents to the milestones used in Gantt chart, the majority of objects added by the participants were of Component type. We interpret this result as a reflection of the systems engineers mental processes while modeling the PPLM plan – continuous iterative zigzagging between the two domains – the systems engineering domain and the project management domain. The participants did not choose to add milestones to the Gantt

chart, as equivalents to the Component objects they added in their PPLM models, even when asked if they have any changes they like to conduct to the Gantt chart *after* modeling the PPLM model. Therefore we claim that the OPM-based environment not only enabled them, but encouraged them, to freely express their intuitive linking between the processes, which belong to the *project domain* and the components which reside in the *product domain*.

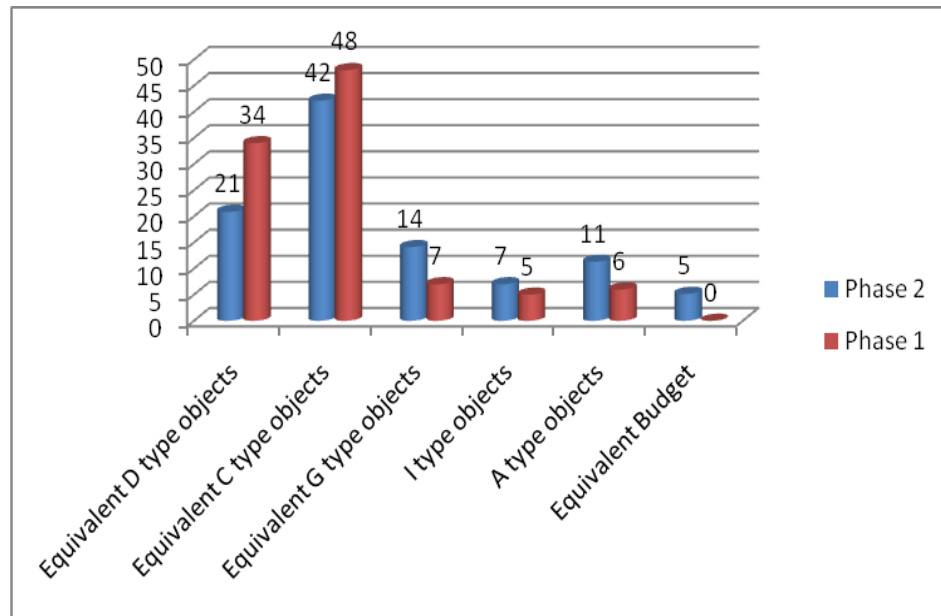


Figure 8-9 Percentage of objects types comparison - Stage 2 against Stage 1

The diversity of the PPLM models within and between the stages can be related to the fact that the participants modeled the processes (project domain) in the model, with the objects (product domain), in a free manner, following our goal to gauge the “natural” systems engineers approach. Modelling the combined project-product plan according to the PPLM methodology will probably result in less diverse models. This could be examined in stage 3 in two ways: (1) teaching the PPLM methodology in detail to participants prior to their modelling, or (2) providing a text which explicitly contains the objects in addition to the processes. The first option was not possible since we did not have an opportunity to conduct such an experiment. The second option was not used because the participants’ reflections in both stages revealed their inconvenience to model the durations in OPM. This was also apparent in the incorrectness of time notion in the PPLM models they produced. Therefore, for the third stage we chose instead to generate the models in accordance with the PPLM methodology and administer a comprehension questionnaire on the model provided.

8.2 Third Stage

This section describes the third stage of the research: starting with the research population and settings, through the research methodology for this stage, and concluding with the stage results and analysis if the PPLM vs. Gantt chart analysis.

8.2.1 Research Population and Setting

The participants in the third research stage were the same students of the ones in the second stage, namely the 32 mid-career systems engineers who were graduate students at the systems engineering program of the Technion. While in the former two stages the participants were asked to create the models based on a given text, in this stage the PPLM model and the Gantt chart model were provided to the participants. This stage was conducted through a class assignment, given after reviewing the solution of the homework which had been given a week earlier and used for the second stage of our research. The class assignment was given individually, allotting one hour for its completion.

8.2.2 Research Methodology

The research conducted in this stage was designed to answer the same research question as in the former two stages, namely how systems engineers view the differences between a Gantt model plan and a PPLM model plan. However, this time, the examination was elaborated to investigate project and product aspects when these are integrated according to the PPLM methodology.

Since the participants in this stage were the same as the ones in the second stage, there was a need to use a new model, different from the UAV model used in the first two stages. The model we used in this stage was that of a simplified CT scanner. Appendix G includes the provided CT scanner PPLM model, in which the project and product aspects were combined according to the Systems Builds (SBs) notion of the PPLM methodology, as described in Section 7.1.

The participants were given 33 questions, based on both models.

Table 8-8 shows the first question as an example. The entire questionnaire is provided in Appendix E.

The assignment was divided into two parts, as presented in Table 8-9. In the first part the participants were given the Gantt chart (see Appendix F) of the CT scanner project plan, together with the product section of the entire OPM-based PPLM model. The information contained the project data, given in the Gantt chart, as well as the product data, given in the OPM-based model. They were instructed to answer all 33 questions (see Appendix E – Stage (3) participants CT scanner assignment) based on the given material, using the two right columns on the questionnaire pages titled “Answer based on Gantt Model” in

Table 8-8.

In the second part, the participants were given the entire PPLM model, which contained the combined project-product information, while leaving the Gantt chart for them as possible reference. They were instructed to answer all 33 questions again based on the new given material, and to indicate the diagrams upon which each answer is based in the two left columns of the questionnaire pages titled “Answer based on PPLM Model” in

Table 8-8. In addition, the participants were instructed to check their previous answers “based on Gantt Model”, and in case they wish to change the former answer – do so, while denoting the first answer by (1) and the second answer by (2) in the right columns of the questionnaire pages (“based on the Gantt model”).

Table 8-8 The structure of the questionnaire used in Stage 3

	Question	Answer based on PPLM Model		Answer based on Gantt Model	
		Mark one option	Full answer	Mark one option	Full answer
1	How long is the project planned to take?	(a) can be easily found out (b) can be found out (c) can be hardly found out (d) Impossible to find out	Based on diagrams:	(a) can be easily found out (b) can be found out (c) can be hardly found out (d) Impossible to find out	

Table 8-9 The two assignment parts in stage 3

	1 st part	2 nd part
Given	Gantt chart model + <u>Product</u> section of the entire PPLM model	Entire PPLM model + Gantt chart left also
Allotted time	30 minutes	30 minutes
Assignment instructions	Answer all 33 questions based on the given material. Use the two right columns on the answering pages titled “Answer based on Gantt Model”	Answer all 33 questions again based on the given material. Use the two left columns on the answering pages titled “Answer based on PPLM Model”. In case you wish to change your previous answer in the right column (based on the Gantt model) do so, while denoting your first answer by (1) and second answer by (2).

The order of the two parts – Gantt followed PPLM – was deliberate; since the participants were more familiar with Gantt chart representations for the project domain, they were given the Gantt chart first with the product section of the entire PPLM model, in order to easily familiarize themselves with the CT scanner project-product information. Later on, they were given the entire information in a PPLM OPM-based model.

The information provided in the first part of the assignment is equivalent to the information provided in the second part of the assignment, except for a few deliberate non-equivalent data, as explained below. In both parts, the information contained the same tasks planned for developing and integrating the five planned System Builds (SBs) of the CT scanner. The objects of the SBs in the OPM-based model were represented by equivalent milestones in the Gantt chart. The equivalence is complete except for deliberately omitting or changing data in each model, to be used as check questions, as listed in **Error! Reference source not found..**

In addition, for both models, the participants were asked “What are the three most important things you find missing in the plan?” (Question 36).

Table 8-10 Designed non-equivalence in the check questions

Issue	Related Question	In PPLM	In Gantt
Built-in Test (BIT)	14 and 15	Completely missing	Fully contained
SB#4	4 and 5	Contained	Completely missing
SB#5	4, 5, and 30	Missing specific in-	Fully contained
<i>Integration</i> in System Build #3	25	Contained	Missing

8.2.3 Results and Analysis

This paragraph describes the results and analysis of the third stage of the research: starting with the approach for results handling, through the obtained results in this stage, ending with deeper comparison of PPLM vs. Gantt chart by individual questions, revealing differences between the PPLM and the Gantt chart model in more detail.

8.2.3.1 Results handling

The participants answered the questions using the following scale regarding their view on the ease of finding an answer to each question: (a) can be easily found out, (b) can be found out, (c) can be hardly found out, and (d) Impossible to find out. We converted the answers to quantifiable data as follows: (a) – worth 4 points, (b) – worth 3 points, (c) – worth 2 points, and (d) – worth 1 point. This grading maintains the meaning of the scale where (a) is the best situation and (d) is the worst.

A question for which an answer was missing for both models – PPLM *and* Gantt – was left out. There were only 11 such missing answers out of 1056 (33 Questions times 32 participants) answers, so their influence on the results is negligible.

In a question for which an answer was missing for one of the models – PPLM *or* Gantt – we replaced the missing answer with an answer identical to the answer that was given to answered model. This way, we added to the cumulative rankings of each model to gain balance of calculated relevant sums (as discussed in the continuation), maintaining evenness between the two models.

8.2.3.2 PPLM vs. Gantt total grading results

The results of the total grading for each question in each model were calculated according to the following equation. Each total grade for each one of the 33 questions is based on the quantity (count) of a, b, c, and d found among the 32 participants' answers.

$$Total\ Grade = 4 * \sum_{1}^{32} count(a) + 3 * \sum_{1}^{32} count(b) + 2 * \sum_{1}^{32} count(c) + \sum_{1}^{32} count(d)$$

The results of the t-test comparison between the PPLM and the Gantt chart are presented in presented in

Table 8-11 and

Table 8-12. On average PPLM significantly scored higher ($M=94.7$, $SE=3.466$), than Gantt chart ($M=77$, $SE=4.83$, $p < 0.05$).

Table 8-11 Paired samples statistics for PPLM and Gantt chart

Pair	Mean	N	Std. Deviation	Std. Error Mean
PPLM	94.70	33	19.911	3.466
Gantt	77.00	33	27.749	4.830

Table 8-12 Paired Samples Test for PPLM and Gantt chart

Pair	Paired Differences					t	df	Sig. (2-tailed)
	Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
				Lower	Lower			
PPLM - Gantt	17.697	27.341	4.759	8.002	27.392	3.718	32	.001

Taking into account all 33 questions, the PPLM model median of grades is about 20% higher than the median of the Gantt model grades. This is a very encouraging result, indicating that, at least for the examined CT scanner case, the answers to the comprehension questions on the information contained in both models, were more easily found based on the PPLM than on the Gantt. Since Gantt charts are commonly used with consensus and the participants became familiar with OPM only a week before the given assignment, we expected to find the PPLM scores close or only slightly lower than the Gantt chart scores. This would have been a positive result in our mind. But, the results exceeded our expectations regarding the PPLM model. This result may also indicate that OPM is easily learned at least for the purpose of “reading” PPLM models (not necessarily for “writing” them).

The results of the standard deviations of the total grading of all 33 questions are surprising. We expected a smaller range of results for the Gantt model than for the PPLM model, based on the assumption that the Gantt is more common and that it contains the entire plan information in a single view. The results are not in line with our expectations, and may indicate that, at least for the examined CT scanner case, the comprehension of the Gantt model, in regard to our 33 questions, is not as straightforward as we expected.

The distribution of the answers to all 33 questions is presented in Figure 8-10. The numbers next to each bar indicate the total count of a, b, c, or d found across all 32 participants. The sum of counts for each model – PPLM or Gantt – is 1045 (33 questions, 32 participants, 11 missing answers). Item (a) which refers to “can be easily found out” is found in about 40% more cases for the PPLM model (519) than for the Gantt chart model (337). Item (b) which refers to “can be found out” is found in about 20% more cases for the PPLM model (225) than for the Gantt chart model (171). Item (c) which refers to “can be hardly found out” is found in about 50% more cases for the Gantt chart model (143) than for the PPLM model (73). Item (d) which refers to “Impossible to find out” is found in about 40% more cases for the Gantt chart model (394) than for the PPLM model (228).

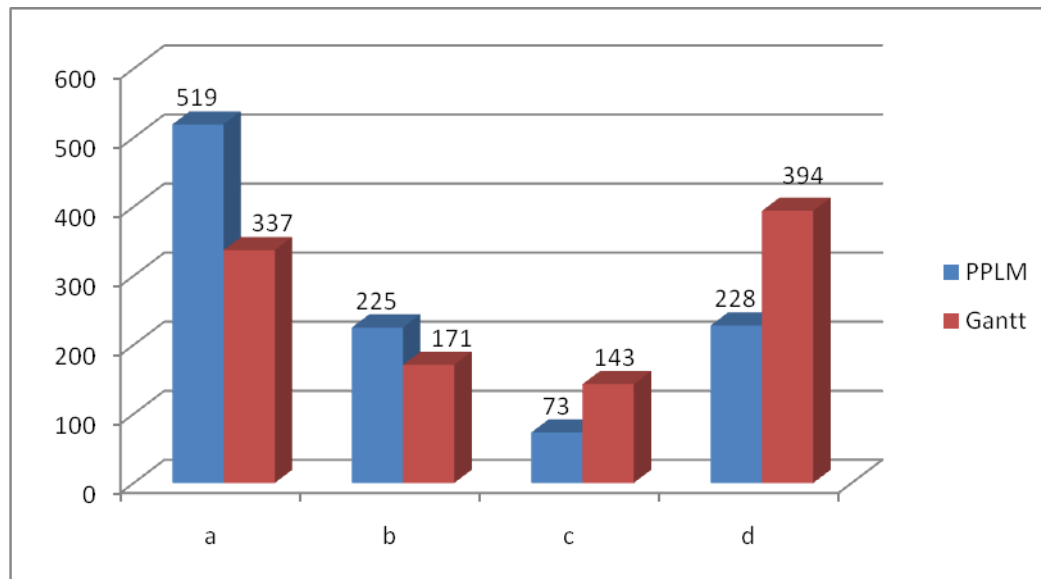


Figure 8-10 The distribution of answers to all 33 questions

8.2.3.3 PPLM vs. Gantt by question

Examining the results for individual question reveals differences between the PPLM and the Gantt chart model in more detail. The results of grades obtained – for each on the 33 questions and for PPLM and Gantt chart are depicted in Figure 8-11. The horizontal axis indicates the question number (33; only odd number are plotted), while the perpendicular axis indicates the total grade, calculated according the equation in subsection 8.2.3.2. The questions were designed in a gradually evolving manner starting with “pure” duration questions, followed by general completeness questions and continuing with project-product completeness questions. The results in Figure 8-11 indicate the following.

- For questions 1 and 2, which are related exclusively to duration, both models are close with slight superiority advantage of the Gantt over the PPLM model.
- For questions 3 to 13, which are general completeness and project-product completeness questions, the outcomes for the two models are close with a slight advantage of PPLM over Gantt.
- For questions 14 to 15, which are two check questions, where the Built-in Test (BIT) cluster was omitted from the PPLM model but contained in the Gantt, the Gantt is much better than the PPLM.
- For questions 15 to 33, except for 21, 23, and 30, which are general completeness and project-product completeness questions, the two models scored close, but PPLM had advantage over Gantt.

Table 8-13 contains the detailed results for each question with specific discussion. $P < 0.05$ appears in the “*T-Test result*” column, denoting when the T-test results of the comparison between the PPLM and the Gantt chart, for each question, indicate a significant difference, otherwise the cell in this column is left blank.

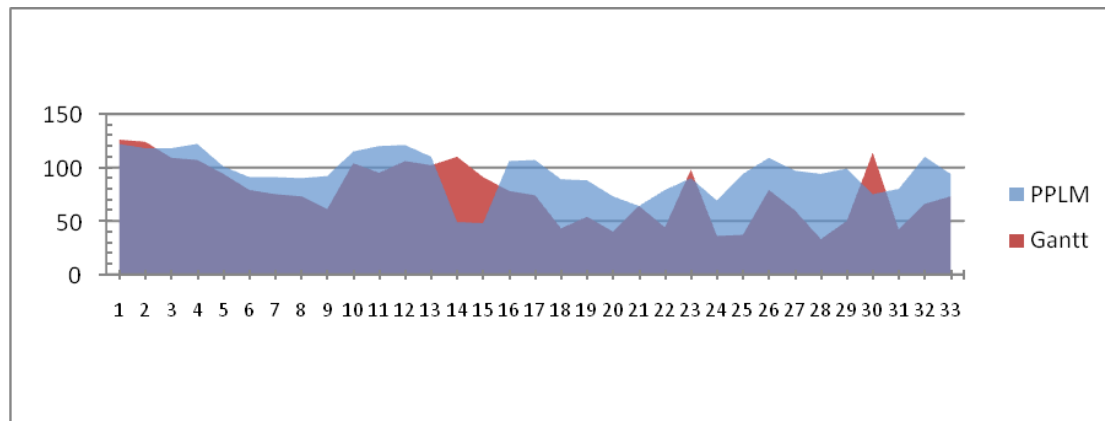


Figure 8-11 PPLM vs. Gantt over the 33 questions span

Table 8-13 Stage 3 detailed results for each question

#	Question Phrase	Results		T-Test result	Discussion
		PPLM	Gantt		
1	How long is the first development cycle planned to take?	122	126		The difference between Gantt and PPLM is not significant. This result indicates that in this stage, where we provided the models, we overcame the problem of the inconvenience of the presentation of process durations in the PPLM model. It also suggests that with proper training, one can produce both models – PPLM and Gantt – with little difference if any with respect to presentation of process durations.
2	How long is the first development cycle planned to take?	118	124		The difference between Gantt and PPLM is not significant. The discussion in question 1 is applicable here as well.
3	When does the 1st cycle start in regard to <i>Definition</i> ? What is the reason?	118	109	P<0.05	PPLM is about 8% better than the Gantt. The reason lays in the asking for the reason in the question phrasing. The reason is explicit only in the PPLM model, while the relationship itself between the 1st cycle and the <i>Definition</i> is explicit in both models. This result is an example for the strength of the PPLM model – it provides the rational for the relationship, which is actually the deliverables that are outputted from the 1st cycle and required for the <i>Definition</i> .
4	How many System Builds are planned?	122	107	P<0.05	PPLM is about 12% better than the Gantt. We expected the results for both models to be close, since they both do not contain all SBs (see Table 8-10) but they both show the intent of including 5 SBs in the plan. We expected that when a SB is clearly missing, it would be graded as (a) indicating that the fact that it is missing “can be easily found out”.

					But, many participants chose other answers as a reflection to the missing SBs.
5	Are there system builds that are not covered by the plan?	101	94		The difference between Gantt and PPLM is not significant. The discussion in question 4 is applicable here as well.
6	Is the plan complete?	91	79	P<0.05	PPLM is about 13% better than the Gantt. The discussion in question 4 is applicable here as well.
7	Are there components of the CT scanner that are not covered by the plan?	91	75	P<0.05	PPLM is about 17% better than the Gantt. For answering this question based on the Gantt and the provided product model, the participants had to alternate between the two to find out whether there are components of the CT scanner that are not covered by the plan. This information is clearer in the combined project-product PPLM model.
8	Are there components of the CT scanner that are not required according to the plan?	90	73	P<0.05	PPLM is about 19% better than the Gantt. The discussion in question 7 is applicable here as well.
9	Is there a parameter of the CT scanner that is not covered by the plan?	92	61	P<0.05	PPLM is about 34% better than the Gantt. The discussion in question 7 is applicable here as well.
10	What specifications are required for the first development cycle?	115	104	P<0.05	PPLM is about 10% better than the Gantt. This result is an example for the strength of the PPLM model – it provides the rational of the plan in terms of explicit deliverables: which are actually the deliverables that are outputted from each process and which deliverables are required for each process in order to start.
11	What specifications are required for the second development cycle?	120	95	P<0.05	PPLM is about 21% better than the Gantt. We expected the results for both models to be close, since all five specifications are contained in both models – as object in the PPLM and as milestones in the Gantt. The result can be derived from a “halo effect” of the fact that the all deliverables are in general more explicit in the PPLM model.
12	What is the outcome of the Definition activity?	121	106	P<0.05	PPLM is about 12% better than the Gantt. The discussion in question 10 is applicable here as well.
13	How many specifications are contained in the plan?	110	102		PPLM is about 7% better than the Gantt. We expected the results for both models to be close, since they both contain the same information regarding to the specifications contained in the plan – object in the PPLM

					and milestones in the Gantt. The result can be derived from a “halo effect” of the fact that the all deliverables are in general more explicit in the PPLM model.
14	When is the BIT planned?	49	110	P<0.05	Gantt is about 56% better than the PPLM. We expected these results since the BIT is very clear in both – it is clearly completely missing in the PPLM model while clearly fully contained in the Gantt model (see Table 8-10).
15	What is required for the BIT?	48	91	P<0.05	Gantt is about 47% better than the PPLM. The discussion in question 14 is applicable here as well.
16	What product processes are planned to be achieved by the end of the 2 nd development cycle?	106	78	P<0.05	PPLM is about 26% better than the Gantt. The discussion in question 10 is applicable here as well.
17	What product processes are planned to be achieved by System Build 3?	107	74	P<0.05	PPLM is about 31% better than the Gantt. The discussion in question 10 is applicable here as well.
18	What product processes are planned to be achieved by System Build 4?	89	43	P<0.05	PPLM is about 52% better than the Gantt. The discussion in question 10 is applicable here as well.
19	How many product processes?	88	54	P<0.05	PPLM is about 39% better than the Gantt. The discussion in question 10 is applicable here as well.
20	How many product requirements?	73	40	P<0.05	PPLM is about 45% better than the Gantt. Actually neither one of the models contains the product requirements. The result can be derived from a “halo effect” of the fact that the all deliverables are in general more explicit in the PPLM model
21	How many design reviews are planned?	64	64		Both models score the same. We expected even lower grades since actually neither one of the models contains planned design reviews.
22	Are all the product processes covered by the plan?	79	44	P<0.05	PPLM is about 44% better than the Gantt. The discussion in question 10 is applicable here as well.
23	If the <i>table specification</i> is delayed, how is the plan affected?	90	98		The difference between Gantt and PPLM is not significant. The results for both models are close and match our expectations, based on the fact that all five specifications are contained in both

					models – as object in the PPLM and as milestones in the Gantt.
24	If a <i>software specification</i> is delayed, how is the plan affected?	69	36	P<0.05	PPLM is about 48% better than the Gantt. There are three identical <i>software specifications</i> in both models. But, the notion of them being <i>software specifications</i> can be derived only from the product model which specifically contains the three objects under the software component, and are named identical to the corresponding <i>software specification</i> s. For answering this question based on the Gantt and the provided product model, the participants had to alternate between the two to find out the answer. This information is clearer in the combined project-product PPLM model.
25	What is required in order to achieve <i>Data acquisition</i> ?	94	37	P<0.05	PPLM is about 61% better than the Gantt. The discussion in question 10 is applicable here as well.
26	What is required for <i>Image reconstruction</i> in System Build #3?	109	79	P<0.05	PPLM is about 28% better than the Gantt. The discussion in question 10 is applicable here as well.
27	What software module is required for <i>Image correction</i> in System Build #3?	97	60	P<0.05	PPLM is about 38% better than the Gantt. The discussion in question 10 is applicable here as well.
28	How is <i>scanning velocity</i> defined in regard to System Build #4?	94	33	P<0.05	PPLM is about 65% better than the Gantt. The discussion in question 10 is applicable here as well.
29	How long does the <i>Integration</i> in System Build #3 take?	99	50	P<0.05	PPLM is about 50% better than the Gantt. We expected these results since the <i>Integration</i> in System Build #3 is very clear in both – it is missing in the Gantt model while it is contained in the PPLM model (see Error! Reference source not found.).
30	How long does System Build #5 take?	75	114	P<0.05	Gantt is about 34% better than the PPLM. We expected these results since SB#5 is missing in the PPLM model while it is contained in the Gantt model (see Error! Reference source not found.).
31	How is the X-ray tube treated in the	80	42	P<0.05	PPLM is about 48% better than the Gantt. The discussion in question 10 is applicable here as well.

	1 st development cycle?				
32	What product characteristics are planned to be achieved by the end of the 2 nd development cycle?	110	66	P<0.05	PPLM is about 40% better than the Gantt. The discussion in question 10 is applicable here as well.
33	What product characteristics are planned to be achieved by the end of the 3 rd development cycle?	94	73	P<0.05	PPLM is about 22% better than the Gantt. The discussion in question 10 is applicable here as well.

9 Summary and suggested future work

9.1 Summary

The goal of this research was to develop an approach and a conceptual model for fusing the domain of the *product* be developed with the domain of the *project* which is expected to deliver the product in order to support systems engineering and project management. The combined project-product model, the Project Product Lifecycle Management (PPLM) model is based on Object Process Methodology (OPM). This model-based approach enables the simultaneous expression of the function, structure and behavior of both the project and the product via the same ontological and methodological foundations, maintaining full traceability between the project and product data.

9.1.1 The first research phase

The first research phase was conducted in order to explore systems engineers' comprehension of the product and project dimensions, as presented in Chapter 4. Based on the model that was developed at the time of the first phase of the research (see Chapter 3), seven project management (PM) methods were examined in order to answer the first two research questions: (1) While conducting the systems engineering management (SEM) process, to what extent do practitioners perceive a notion of a project domain, a product domain, and a combined project-product domain? (2) How do systems engineers perceive the extent to which the seven PM methods support SEM?

The suitability of the seven PM methods for SEM, as perceived by systems engineers was examined with respect to 14 factors. Focusing on application of PM methods within SEM, the seven examined methods included six traditional project management methods – System Dynamics (SD), Program Evaluation and Review Technique (PERT), Critical Path Method (CPM), Design Structure Matrix (DSM), Earned Value Method (EVM), and Gantt chart – and the new PPLM model-based project planning methodology. Since SEM is about handling and solving problems

associated with the intricate relationships of the product with the project that delivers it, we classified the 14 factors into three domains: the project domain, the product domain, and a holistic project-product ensemble domain. Our research population, a group of 24 mid-career systems engineers studying in the Systems Design and Management graduate program at MIT, ranked the adequacy of each one of the seven examined project management methods to tackle each one of the 14 factors. The set of 14 factors was found reliable for comparison of six out of the seven examined project management methods. After excluding two factors, a set of 12 factors was reliably used for comparison of all seven PM methods.

The findings support the notion of the project and the product as two complementary facets involved in systems engineering management. Four project management methods - System Dynamics (SD), Design Structure Matrix (DSM), Earned Value Method (EVM), and PPLM—were found more suitable than the others for systems engineering management. PPLM was found the most suitable method both by dimensions analysis and ranking comparison analysis. The results may imply that OPM-based PPLM should be favorably considered as a suitable method for managing product-project ensembles within systems engineering management. The Project-Product Lifecycle Management (PPLM) methodology might become the bridge between systems engineering and project management, enabling the simultaneous expression of the function, structure and behavior of both the project and the product within a holistic integrated conceptual model.

9.1.2 The second research phase

Since all the entities and their relations are represented in the PPLM model, automatic procedures can be devised to generate other project views, including Activities Network Plan, Critical Path Method (CPM), Gantt chart, Work Breakdown Structure (WBS), and Design Structure Matrix (DSM). The second research phase was devoted to the development of potential automatic extraction of these PM representations from the OPM-based PPLM model for providing new, useful model-based representations. Using a running example of an unmanned aerial vehicle, we reviewed the various project management (PM) representations and discussed problems stemming from the lack of explicit representation of the product facet in current PM methods. Based on this observation, we proposed an integrated conceptual model-based combined project-product lifecycle approach for systems engineering management.

Starting with construction of an OPM model-based plan of the project and the product it is expected to deliver, we obtain a comprehensive model that serves as a basis for deriving the customary repertoire of project management tools. However, rather than being constructed separately and independently, which is a certain source of mismatches and incompatibilities with potential detrimental consequences to the project success, these project views—Activities Network, CPM, Gantt, WBS, and DSM—are derived reflections of various aspects of a comprehensive underlying OPM model and are therefore consistent and coherent. Any change in the project model is automatically reflected in the various project management views. Moreover, any change in the product to be delivered—the ultimate output of the project—can be modeled in the joint project-product model and its implications on the project parameters can be directly inspected and assessed.

The fact that the model integrates the process view of project activities and tasks with the object view of the product deliverables with its components at all levels enables the project manager to focus on advancing the completion of the output deliverables rather than on just performing processes without direct reference to their anticipated

outcomes. Knowing the critical path and the critical artifacts—those generated by processes along the critical path—enables their close monitoring and management .

9.1.3 The third research phase

The third research phase was conducted in order to explore the characteristics of systems engineering management tools and methods among systems engineers, applying a qualitative research approach designed to answer the third research question: How do systems engineers perceive systems engineering methods and tools? Although the common belief is that the use of methods and tools for systems engineering in a project leads to process improvement and higher quality of deliverables, the results we found suggest that for a given project the utility is perceived as low by systems engineers. Yet, they ascribe high utility to the use of methods and tools at the enterprise level. These findings can provide a possible explanation to the fact that overall, the majority of systems engineers do not voluntarily adopt methods and tools in projects.

The finding of the system complexity might explain why the majority of systems engineers are reluctant to use methods and tools. The perceived utility gained from using methods and tools in medium- and small-scale project is low, but this is exactly the scale of projects for which the majority of practitioners are assigned. In contrast, they perceive the utility as high for large-scale projects dealing with complex systems, and this is correlated with the fact that local initiatives of using methods and tools in the enterprise are in large scale projects. These two finding are complementary: for small or medium scale projects, the perceived utility is high at the enterprise level. A large scale project can be considered as an enterprise in itself; therefore the perceived utility from using methods and tools in such a project is high.

These findings give rise to EPPLM – Enterprise PPLM, a possible future work to extend the scope of the PPLM research to include the enterprise as the agency that is in charge of the execution of multiple projects, each delivering a product or a product line, which compete for the enterprise resources.

9.1.4 The fourth research phase

Based on conclusions from the previous research phases, the objective of the fourth phase was to answer the fourth research question: How are the differences between a Gantt model plan and a PPLM model plan reflected by results obtained from their use by systems engineers?

This phase included three stages:

- Stage (1) Gantt-PPLM comparison among first participants group, based on a simplified Unmanned Aerial Vehicle (UAV) model which contains the project and the product in a freely combined manner,
- Stage (2) Same as 1, with a second group and improved test based on lessons learned from (1), and
- Stage (3) Elaborated Gantt-PPLM comparison among the second group, based on a simplified CT scanner model in which the project and product aspects were combined according to PPLM methodology.

While in the first two stages the participants were asked to create the models, based on a given text specifying 23 tasks and their relationships, in the third stage the models – a PPLM model and a Gantt chart model – were given to the participants, who were asked to answer 33 questions based on the provided models.

The Unmanned Aerial Vehicle (UAV) assignment in stages 1 and 2 enabled the coverage of the processes in both models with little effort. While the Gantt charts provided by the participants were identical, containing the exact processes identified in the provided text, their PPLM models, which included additional objects, were diverse. The objects types that were added in both stages were analyzed according to the PPLM objects typology. The majority of added objects were C (Component) - 42% in stage 2 vs. 48% in stage 1, followed by D (Document) type – 21% in stage 2 vs. 34% in stage 1. The percentage of added G (Gates) type objects in both stages was low - 14% in stage 2 vs. 7% in stage 1 (7%). This result suggests that the combination of the project domain – the processes in the model, with the product domain – the objects in the PPLM model, is more powerful than we expected; the participating systems engineers chose to add objects to the given processes according to their own intuitive judgment. While we expected them to add more G type components, as equivalents to the milestones used in Gantt chart, the majority of objects added by the participants were of Component type. We interpret this result as a reflection of the systems engineers mental processes while modeling the PPLM plan – continuous iterative zigzagging between the two domains – the systems engineering domain and the project management domain. The participants did not choose to add milestones to the Gantt chart, as equivalents to the Component objects they added in their PPLM models, even when asked if they have any changes they like to conduct to the Gantt chart *after* modeling the PPLM model. Therefore we claim that the OPM-based environment not only enabled them, but encouraged them, to freely express their intuitive linking between the processes, which belong to the *project domain* and the components which reside in the *product domain*.

The CT scanner assignment in stage 3 was designed to explore a deeper comparison of PPLM model vs. Gantt chart, based on 33 questions. The comparison of both models using t-test showed that on average PPLM significantly scored higher ($M=94.7$, $SE=3.466$), than Gantt chart ($M=77$, $SE=4.83$, $p < .05$). This is a very encouraging result, indicating that, at least for the examined CT scanner case, the answers to the comprehension questions on the information contained in both models, were more easily found based on the PPLM than on the Gantt. Since Gantt charts are commonly used with consensus and the participants became familiar with OPM only a week before the given assignment, we expected to find the PPLM scores close or only slightly lower than the Gantt chart scores. This would have been a positive result in our mind. But, the results exceeded our expectations regarding the PPLM model. This result may also indicate that OPM is easily learned at least for the purpose of “reading” PPLM models (not necessarily for “writing” them).

9.2 Contributions of this work and suggested future work

We identified five general topics in which the complete combined Project-Product Lifecycle Model (PPLM) developed and researched in this work, has a potential contribution, as presented in Table 9-1.

Table 9-1 Issues of potential contribution of this research

Issue	Detailed potential contribution
Enhanced Confidence level in designed	The ability to simultaneously express the required information from both the project and the product domains within a single integrated model-based framework can potentially lead to a more

relationships between activities in project plans	reliable project plan, which is less prone to the need for repeated changes and corrective actions. The increased robustness of the resulting project plan is attributed to the need to make decisions about the project's process order and logic while explicitly addressing the associated product model.
Enhanced Change Management Capability	Change management of the project-product plan can be easily conducted since the PPLM plan contains the details of both the project activities, linked to specific product entities, it is easy to track down <i>product</i> processes or objects that are in jeopardy, in case some specific <i>project</i> activities are delayed. This kind of information facilitates the system engineering manager with the ability to better cope with changes by following the explicitly modeled relationships which are the potential paths of project-product impacts. This kind of combined project-product knowledge cannot be extracted from common used project plans models.
Enhanced Plan Realism	<p>Current systems engineering management utilizes a host of methods and techniques for devising a systems engineering management plan, which focus almost exclusively on activities and tasks to be performed, along with their relationships and durations. The relationships defined in the common planning methods are of precedence, modelling only synchronous planning.</p> <p>The use of OPM-based PPLM models enables the modelling of a more realistic project plan, containing asynchronous triggering of processes by events and loops. Since the PPLM contains the artifact produced, processes are triggered by the artifacts that are linked to them as events. These triggering artifacts can be aggregated into project control logic which is contained in the complete PPLM model. The project control logic is modeled to react to "what-if" scenarios contained in the model, especially for handling the creation of Systems Builds (SBs) which are part of the PPLM methodology. The SBs are defined configurations of the system, planned to be achieved along the project lifecycle span, by pre-planned developing & integrating cycles. Using OPM, this planning contains the logic to answer which product functionality is to be achieved by the project-activities processes, at each cycle. The model actually conations, instead of one synchronous path as usually established in a Gantt chart, multiple event-driven paths, as defined by the planner. Instead of one deterministic Gantt or Project Activities Network, the OPM based plan is a conglomerate of project plan instances that can be verified using the OPCAT simulation. Based on a large number of simulation runs, statics results can also be obtained for specific durations of interest for the planner.</p>
A single source for coherent Project Views and model-based	The underlying PPLM model can serve as the basis for and source of the various project-oriented representations. Since all the entities and their relations are represented in the PPLM model, automatic procedures can be devised to generate other project view representations, including Activities Network Plan, Gantt

<p>enhancements</p>	<p>chart, Critical Path, Work Breakdown Structure, and Design Structure Matrix. Furthermore, new versions of the common project views are enhanced with critical product-related information gleaned from the common underlying model, facilitating a shift from activities-based to deliverables-based project management.</p> <p>Activities Network Plan: The classical Activities Network contains all the processes without dataflow, but the automatic extraction from the OPM model-based Project Plan enables viewing an Enhanced Activities Network Plan, in which the processes and the objects are simultaneously displayed, along with their input, output, agent, and instrument relationships. A simple mechanism can show or hide the dataflow of objects among the processes, at each level of detail in the Enhanced Activities Network.</p> <p>Critical Path Method (CPM): Automatic extraction of the CPM view from the OPM project model enables the creation of an Object-enhanced Critical Path (OCPM) PPLM view, in which the processes and objects are presented simultaneously. The OCPM view enables the project manager to detect not just the critical path, but also the <i>critical objects</i>—the objects required as inputs to the processes along the critical path. Knowing what these objects are reduces the scheduling risk by focusing on managing them with high priority, adding value to the project management process. Rather than inquiring about the processes designed to deliver the critical objects, the project manager can monitor the status of the critical objects themselves. These critical objects may include documents, approvals, simulation outcomes, analyses, specifications, and reports.</p> <p>This capability, in turn, provides the project manager with the flexibility of treating the processes not as an end in itself, but rather as vehicles for obtaining the critical objects. This way, processes are managed to <i>achieve the critical objects rather than complete the critical tasks</i>. In other words, the ability to describe the critical path in terms of objects (products), rather than in terms of the processes that yield them, is highly valuable for the project manager and the systems engineering manager, since it directs them to focus on precisely those activities that advance the project towards timely completion of the required deliverables.</p> <p>Design Structure Matrix (DSM): The Process-based DSM is obtained from the relationships defined in the PPLM model. Examining the generated DSM, identifying blocks of coupled activities, iterations and re-sequencing product development tasks to minimize budget and duration can be integrated into PPLM. Since the OPM model includes the project and product objects in addition to the processes, a combined Object-Process DSM (OPDSM) can be automatically generated, based on the relationships established in the OPM project-product model.</p>
----------------------------	---

Based on the research we conducted we propose several topics for future research as follows:

- Devising algorithms for automatically updating the various project management views in response to changes in the project plan or product specification made on the joint project-product OPM model.
- Enabling real-time instance and data-based execution of the project model for simulation, exploration, and forecasting purposes. Using the enhanced execution capabilities of OPCAT, it might be possible to simulate and adapt the project and product parameters for feasibility and in some cases even optimality of the project's designed timeline, resource allocation, and tradeoff of product functionality and deliverables.
- Developing mechanisms for effectively assessing and managing risks that relate to processes and their resulting objects along the critical path by monitoring them more closely than other, less critical risks, and mitigating them by allocating more resources to reduce the probability of their occurrence.
- Enabling concurrent monitoring and controlling of both the project and product evolution in order for the product manager to be able to take corrective actions in real time as needed if potential deviations are detected. This can be done via execution-based simulation or, if actual deviations are detected, via close monitoring of the project lifecycle within that of the product in the context of the enterprise or in relation to subcontractors' deviations from their schedules.
- Developing mechanisms for answering "what-if" questions that enable executives to assess the impact of changes requested by the customer or are about to be introduced to the product during its design due to technological, legislative, or economical considerations, in terms of impacting cost, risk, and time-to-market. What-if analysis based on possible architecture models can be conducted to help achieve objectives and rationalized decisions regarding the appropriate architecture from the set of potential candidates. Such analyses can reveal contradicting requirements and provide basis for suggesting corrective actions in order to improve the product architecture and required changes in the project delivering it.
- Developing a methodology and a supporting mechanism for planning and controlling the project duration, which accounts for the contribution of critical objects and sub-critical activities. This methodology can also be further developed to handle critical resources – Budget, Agents (human resources), and Instruments (non-human resources), which are all contained in the complete PPLM model.
- Devising algorithms for automatic extraction of more DSM types. Since the PPLM model includes the project and product objects in addition to the processes, the automatic generation of the component-based DSM can be extracted based on the relationships established in the OPM-based model. As a result of the inclusion of all agent human resources in the OPM model-based project plan, it is also possible to automatically produce the Organizational-based DSM and investigate the social network issues, as revealed from the assignment of the humans in the model.
- Developing a methodology for EPPLM – Enterprise PPLM, a possible future work to extend the scope of the PPLM research to include the enterprise as the agency that is in charge of the execution of multiple projects, each delivering a product or a product line, which compete for the enterprise resources.

References

- [1] D. Dori, *Object-Process Methodology – A Holistic Systems Paradigm*. Springer Verlag, Berlin, Heidelberg, New York, 2002.
- [2] *INCOSE Systems Engineering Handbook*, INCOSE-TP-2003-016-02, Version 2a, 2004.
- [3] *NASA Systems Engineering Handbook*, NASA-SP-6105. SP-6105, 1995).
- [4] *Engineering Management*, Military Standard Mil-Std-499B 2.50T-1993.
- [5] *Standard for Application and Management of the Systems Engineering Process*, IEEE Standard 1220-1994.
- [6] *Processes for Engineering a System*, ANSI/EIA Standard 632-1999.
- [7] J. A. A Zachman, framework for information systems architecture. IBM Systems Journal, 26 (3), pp. 276-292.
- [8] DoD Architecture Framework: Volume 1: Definitions and Guidelines Version 1.0, DoD Architecture Framework Working Group.
- [9] MOD Architecture Framework Overview, Version 1.0, 31 August 2005, MODAF-M09-002 (www.modaf.com).
- [10] TOGAF (2007). The Open Group Architecture Framework. <http://www.opengroup.org/architecture/togaf8-doc/arch/>
- [11] D. Delaurentis, "System of Systems Definition and Vocabulary," School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, 2007.
- [12] M. W. Maier, "Architecting Principles for Systems of Systems," Systems Engineering, Vol. 1, No. 4, 1998, pp. 267-284. 1997.
- [13] http://www.nasa.gov/offices/oce/appel/pm-development/pm_se_competency_framework.html
- [14] *Interim Defense Acquisition Guidebook*, Department of Defense, 15 June 2009. <https://acc.dau.mil/dag>
- [15] M. Frank, "Cognitive and Personality Characteristics of Successful Systems Engineers," presented at the INCOSE International Symposium, Minneapolis, MN, 2000.
- [16] *Systems and software engineering—System life cycle processes*, International Standard ISO/IEC 15288:2008(E)-2008.
- [17] *INCOSE Systems Engineering Handbook*, INCOSE, Version 3.0, 2009.
- [18] *Improving processes for better product*, CMMI® for Development, Version 1.2, CMU/SEI-2006-TR-008-2006.

- [19] *Managing Successful Projects with PRINCE2*, Office of Government Commerce, 2005.
- [20] The PRINCE2 Definitive Resource. <http://www.non-uk.prince2.com/whatisp2english.html#intro>
- [21] *PMBok*, Project Management Institute, 2008.
- [22] *Guide to Integrated Product and Process Development*, Department of Defense, Version 1.0, February 5, 1996.
- [23] S.C. Cook, J.E. Kasser, and T.K.J. Ferris, "Elements of a Framework for the Engineering of Complex Systems," in *Proc. 9th ANZSYS Conference, Systems in Action*, Melbourne, Australia, 2003, Paper 3000079.
- [24] D.K. Hitchins, (2003). *World class systems engineering – the five layer model*. Available: <http://www.hitchins.net/5layer.html>
- [25] S. Eppinger and V. Salminen, "Patterns of Product Development Interactions," in *Proc. International Conference On Engineering Design, ICED01*, Glasgow, August 21-23, 2001, pp 283-290.
- [26] *DSMC Systems Engineering Fundamentals*, Defense Systems Management College, Fort Belvoir, Virginia, 1999.
- [27] *Assessments of Selected Major Weapon Programs*, United States Government Accountability Office, GAO-06-391, March 2006.
- [28] *Integrated Master Plan and Integrated Master Schedule Preparation and Use Guide*, Department of Defense, Version 0.9, 21 October 2005.
- [29] A. Shein. *Organizational culture and leadership*. Jossey Bass, 1997
- [30] A. Sharon, D. Dori, and O. de Weck, "Is there a Complete Project Plan? A Model-Based Project Planning Approach," *Proc. 19th Annual International INCOSE Symposium*, Singapore, 2009.
- [31] A. Sharon A., V. Perelman, and D. Dori, "A Project-Product Lifecycle Management Approach For Improved Systems Engineering Practices," *Proc. INCOSE 18th Annual International Symposium, 6th Biennial European Systems Engineering Conference*, Utrecht, the Netherlands, 2008.
- [32] International Journal of Engineering Management and Economics (IJEME)
ISSN (Online): 1756-5162 - ISSN (Print): 1756-5154
<http://www.inderscience.com/browse/index.php?journalID=299>
- [33] SuSuh Nam Pyo, *The Principles of Design*, Oxford University Press, 1990.
- [34] Levy K.F., Thompson G. L and Wiest J.D. (1963). The ABCs of the Critical Path Method, *Harvard Business Review*, #63508, *Harvard Business School Publishing*.
- [35] O.C. John, Systems engineering and software engineering processes, products, and people from a standards perspective. *Hampton Roads Area and Southern Maryland Chapters of INCOSE Systems Engineering Seminar*. 2003.

- [36] Unified Modelling Language. <http://www.uml.org/>
- [37] S.J. Mellor, M.J. Balcer, Executable UML. Addison-Wesley, 2002.
- [38] Systems Modelling Language. <http://www.omgsysml.org/>
- [39] Modelling of Complex Physical Systems – The Modelica Portal. 2008
<http://www.modelica.org/>
- [40] OPCAT. <http://www.opcat.com/>
- [41] *Work Breakdown Structures for Defense Material Items*, Military Standard Mil-Std-881-1968.
- [42] *Department of Defense Reliability, Availability, Maintainability, and Cost Rationale Report Manual*, Department of Defense, Washington, DC: Office of the Secretary of Defense. 2009.
- [43] O. de Weck and J.M. Lyneis, *MIT ESD.36 System Project Management course assignments*, 2008.
- [44] J.W. Forrester, *Industrial Dynamics*. Productivity Press, Cambridge, MA, 1961.
- [45] J.W. Forrester, *World Dynamics*. Productivity Press, Cambridge, MA, 1973
- [46] J.M. Lyneis and D.N. Ford, "System dynamics applied to project management: a survey, assessment, and directions for future research," *System Dynamics Review*, Volume 23, 2007, Issue 2-3, Pages 157 – 189. Special Issue: Exploring the Next Great Frontier: System Dynamics at 50 Guest Editor: John D. Sterman. John Wiley & Sons, Ltd. , 2007.
- [47] D.V. Steward, (1981). *Systems Analysis and Management: Structure, Strategy, and Design*, Petrocelli Books, New York, NY. 1981.
- [48] S.D. Eppinger, D.E. Whitney, R.P. Smith, and D.A. Gebala, "A Model-based Method for Organizing Tasks in Product Development," *Research in Engineering Design*, Vol. 6, No. 1, pp. 1-13, 1994.
- [49] T. Browning, "Applying the Design Structure Matrix to System Decomposition and Integration problems: A Review and New Directions," *IEEE Transactions on Engineering Management*, Vol. 48, No. 3, pp. 292-306, 2001
- [50] R. Likert, "A Technique for the Measurement of Attitudes," *Archives of Psychology*, 140: 1–55, 1932.
- [51] L. J. Cronbach, "Coefficient alpha and the internal structure of tests," *Psychometrika*, 16, 297-334, 1951.
- [52] L. Hatcher, *A step-by-step approach to using the SAS(R) system for factor analysis and structural equation modelling*. Cary, NC: SAS Institute, 1994.
- [53] J. Nunnally, *Psychometric theory*. New York: McGraw-Hill, 1978.
- [54] J.E. Long, "Relationships between Common Graphical Representations Used in System Engineering," *Proceedings of the SETE2000 Conference*. 2000.

- [55] James N. Martin, *Systems Engineering Guidebook: A Process for Developing Systems and Products*. CRC Press, 1997.
- [56] S. A. Sheard, Twelve Systems Engineering Roles. *Software Productivity Consortium*, 1996.
- [57] A. Coffey, and P. Atkinson, *Making Sense of Qualitative Data*, Sage Publications. 1996.
- [58] N. K. Denzin, and Y. S. Lincoln, *Handbook of Qualitative Research* – 2nd edition, Sage Publications, Inc. 2000.
- [59] *Software Development and Documentation*, Military Standard Mil-Std-498 AMSC NO. N7069-1994.

Appendix A – MIT ESD Course Syllabus

MASSACHUSETTS INSTITUTE OF
TECHNOLOGY

S Y S T E M D E S I G N A N D M A N A G E M E N T
P R O G R A M

ESD.36
System Project Management (SPM)

Syllabus and Schedule

Units (3-0-9)
Fall 2008

Instructors:
Prof. Olivier de Weck and Dr. James Lyneis

Teaching Assistants:
Athar Syed, Takuto Ishimatsu

Lectures:

Tuesday and Thursday, 3:00-4:30pm

MIT Room 1-390 (Bechtel)

Official Course Description

(from MIT online course listing⁷)

ESD.36 System Project Management



Prereq: Permission of instructor

Units: 3-0-9



Lecture: TR3-4.30 ([1-390](#))

Subject focuses on management principles, methods, and tools to effectively plan and implement successful system and product development projects. Material is divided into four major sections: project preparation, planning, monitoring, and adaptation. Brief review of classical techniques such as CPM and PERT. Emphasis on new methodologies and tools such as Design Structure Matrix (DSM), probabilistic project simulation, as well as project system dynamics (SD). Topics are covered from strategic, tactical, and operational perspectives. Industrial case studies expose factors that are typical drivers of success and failure in complex projects with both hardware and software content. Term projects analyze and evaluate past and ongoing projects in student's area of interest. Projects are used to apply concepts discussed in class.

O. de Weck, J. Lyneis

⁷ URL: <http://student.mit.edu/catalog/index.cgi>

Course Synopsis:

The course System Project Management (SPM) is focused on teaching methods and tools for planning and managing complex product and system development projects. We assume that the enterprise has already chosen what product or system to develop⁸, so that the course can focus on the preparation, planning, monitoring and adaptation of projects. The course is organized into six loosely interwoven modules.

The first module covers traditional and new project planning and simulation techniques such as PERT/CPM, design structure matrices (DSM) and critical chain (CC). This planning-centric view of project management exposes not only the capabilities, but also the limitations of traditional PM methods and tools.

The second module introduces system dynamics (SD) in the context of large projects. Unfortunately, real projects rarely unfold exactly as they are planned. System dynamics shows how the evolution of projects can be simulated by modelling the rework cycle and the dynamic effects of various external parameters as well as the impact of corrective actions taken during the project.

The third module presents four real world case studies to illustrate the issues associated with complex projects. The projects are chosen from a variety of industries (automotive, aerospace, construction, oil & gas exploration, software) to allow the students to see the previously discussed methods in action, but also to start appreciating the importance of strategically and tactically managing projects with emphasis on project risks.

The fourth module is focused on ways in which projects that are already underway can be monitored and tracked in terms of cost, schedule and technical progress. Risk management techniques for identifying, tracking and mitigating risks are discussed. Uncertainty can also be turned into opportunity by embedding real options in projects to maximize project value.

The fifth module discusses various forms of project organizations, the challenges of managing international projects with geographically dispersed teams as well as human aspects of project work.

Finally, in the sixth module we provide pointers to important resources for cost estimation, project management software tools as well as a list of empirical factors that are known to affect project success and failure. These will be presented and debated in class. Student project presentations will round out the course and offer an opportunity to tap into the collective experience and insights of the participants.

⁸ Such upfront issues are covered in ESD.34 System Architecture, among other courses.

Learning Objectives:

The course is specifically designed for students in the System Design and Management (SDM) program and therefore assumes that you already have a basic knowledge of project management. Ideally, you will have already managed one or more projects yourself and will therefore understand the fundamental tensions between technical scope, cost, schedule and risk.

The overall objective of this course is to introduce advanced principles, methods and tools for project management in a realistic context, such that they can be taken back to the workplace to improve your ability to manage complex product and system development projects.

The detailed learning objectives are to:

1. Introduce advanced methods and tools of project management :
 - a. CPM/PERT
 - b. Design Structure Matrix
 - c. System Dynamics
 - d. Critical Chain
 - e. Discrete Event Simulation
 - f. Earned Value Management
2. Understand realistic application of methods (strengths, limitations) and strategic issues
 - a. Industry Examples (interspersed)
 - b. Case Studies
 - c. Risk Management
 - d. Real Options in Projects
3. Obtain an appreciation for organizational and human aspects in
 - a. Project Organizations
 - b. International Project Management (dispersed teams)
 - c. Project Manager soft skills and typical profiles
4. Learn from each other
 - a. Class Discussions
 - b. Project Assignments
 - c. Homeworks

Disclaimer:

This is not a course on how to use commercial project management software (e.g. Microsoft Project). This course does not lead to an official certification as a project manager. Such courses are available from professional societies such as the Project Management Institute (PMI).

In contrast to traditional courses on the subject of project management we will emphasize strategic issues and scenarios that cannot be fully predicted such as task iterations, unplanned rework, perceived versus actual progress and misalignments between work breakdown structures, product architectures and organizations. As an SDM alumnus/alumna you will likely be in a leadership position where you will spend more time thinking about strategic and tactical issues than perform detailed operational planning yourself.

Discussions

We plan to include discussion related to your own project management experiences. We hope to accomplish this in several ways:

1. Class discussions will revolve around your experiences in industrial practice. Several times during a lecture we will stop for discussion points. The key points of this discussion will be recorded by the TA and posted on stellar. Your level of activity during discussions will affect your participation grade.
2. We would like to steer the lecture content and focus on those aspects that interest a majority of the class beyond the standard, planned material. For this purpose the TA will conduct a number of informal polls several times during the semester.
3. We welcome SDM student contributions to specific aspects of the course where you feel particularly competent or where you wish to share unusual project management experiences with your peers. Such contributions can take the form of short prepared speeches, additional readings or mini-presentations throughout the term. Please contact the instructors or TA if you wish to contribute in this manner.
4. The project assignment will allow you to directly apply one of the advanced methods such as DSM, SD or CC to a development situation at your company. Alternatively, you may want to analyze in-depth the reasons for failure or success of past or ongoing complex product development projects.

Homework Assignments:

There will be a total of five (5) homework assignments throughout the term. The assignments are designed to reinforce key concepts from class and focus on applications of specific methods and tools. Our objective is not to make you an expert user of any particular method (e.g. PERT/CPM, DSM, SD) or particular tool (e.g. Microsoft Project, Vensim). We are assuming that you will be in a leadership position at your company and that others will carry out the mechanics of maintaining project plans and documents under your supervision. Therefore, it is important that you understand the basic workings of the different methods and tools and grasp their relative advantages and limitations.

The homeworks are a pedagogic means of ensuring some uniformity in achieving the learning objectives across the class that would not be guaranteed by the term projects alone.

- Homeworks are intended to be solved individually. Verbatim copying from others is not allowed. MIT's standard rules of academic honesty apply: <http://web.mit.edu/policies/10.2.html>
- If students want to cooperate on the homework, they can do so, provided that they properly reference the contributions that others have made on the first page. There will be no deduction if cooperation is properly referenced.
- Each student needs to upload their own solution
- We might gather and publish anonymous time-spent statistics for assignments.
- In the past, some students have shown a tendency towards perfectionism and spent up to 20-30 hours on a single homework. That is excessive. Our intent is that a homework assignment should be able to be solved within a maximum 9-10 hours of effort.
- A master solution will be worked out, posted and discussed for each homework within one week after the due date. The master solution will be delayed if extensions have been granted to some students.
- The homework topics and due dates are summarized in the table below.

HW	Topic	Out	Due
1	Set up Project System Dynamics Model	9/9	9/30
2	Create Project CPM/PERT Plan	9/16	10/9
3	Task-based Design Structure Matrix (DSM)	9/30	10/23
4	Project Monitoring: Earned Value Management	10/9	11/4
5	Identify and Evaluate Real Options in Projects	10/30	11/20

In 2008 we will use an **Unmanned Aerial Vehicle (UAV)** project as the common context for all the homeworks.

Class Project Assignment:

The intent of the Class Project Assignment is to allow you to explore one particular aspect of system project management in-depth in the context of your company or in a general industrial setting. There is relatively large freedom in the selection of topic, choice of research method and team composition.

Some examples of acceptable topics are given below:

- **DSM Project:** Create and analyze a DSM model of a product development project of your choosing. You must identify a project to study, collect the data, conduct the interviews and analysis, and suggest ways in which the process can be improved based on your findings.
- **SD Project:** Identify the dynamics and drivers of a real or hypothetical project. Build a system dynamics model of the project including causal loop diagrams and governing equations. Quantitatively simulate the evolution of the project and explore “what-if” scenarios. This SD model has to be different from the one used in the homeworks, for example by adding feedback effects, replicating sectors, and/or calibrating parameters to a real case.
- **Survey of Methods and Tools:** Conduct a survey of actually used methods and tools (software) for project management in a particular organization. Compare advantages and disadvantages and distill lessons learned. This can also take the form of a comparison of project management practices across multiple firms.
- **Success and Failure of a Past Project:** Analyze in-depth the preparation, planning and execution of a large-scale past development project. Study historical data and conduct stakeholder interviews. Assess the degree of project success or failure against the original project objectives and identify key factors.

This list is not meant to be comprehensive. We are open to other types of projects, provided that there is a clear link to the class objective and contents.

Project Administration:

- The class project is expected to require approximately **50 hours** of cumulative work per person over the course of the entire semester.
- The formal project assignment will be handed out during the first class.
- Two page project proposals are due at the end of the third week.

- Faculty feedback and approval of the project proposals will occur during the 4th week.
- At the time of project approval, each team will be assigned a faculty or staff mentor. The role of the mentor is to help guide the project by pointing out resources, previous work and shape the direction of the research.
- A written progress report (~1000 words as pdf uploaded to SloanSpace) is due in the middle of the semester ('Project Update'). We encourage especially those teams that have both on-campus and off-campus students to meet during the business trip week to make progress on their term projects.
- Final project presentations are given on either 12/2 or 12/4
- Teams must be formed by the third week and be indicated on the project proposal. The nominal team size is four (4) students. Deviations from this guideline can be approved by the instructors.
- The final project deliverable is the set of annotated viewgraphs used during the final presentation, no separate written report is required.
- This is essentially a three-month assignment (September-November) requiring some project management skills to complete on time.
- Project grades are based on the quality of the proposal, mid-term update, in-class presentation and on the insights presented to the class.

Grading:

	Component	Contribution
1	5 Homeworks (10% each)	50%
2	Term Project	40%⁹
3	Active Participation	10%
Total		100%

Active participation means regular attendance of lectures, offering suggestions or questions during lectures and relaying of personal experiences to others in the context of the class.

The class is graded according to the letter system A-F. Standard MIT grading policy applies.

Please bring and use your name cards during every lecture.

Pay attention during class and minimize the use of your laptop for non-class related activities during lectures.

⁹ All team members will receive the same grade for their joint term project.

Appendix B – MIT ESD course schedule

ESD.36 Class Schedule – Fall Term
2008 (13 Tuesdays, 13Thursdays)

Tuesday

Thursday

		Sep 4 L1: Course Introduction dWo, JL Project Assignment out
Sep 9 L2: Intro to Project System Dynamics JL <i>HW1 out</i>		Sep 11 L3: Feedback Loops and Rework JL
Sep 16 L4: Critical Path Method dWo <i>HW2 out</i>		Sep 18 L5: Probabilistic Project Scheduling dWo <i>Project Proposal due</i>
Sep 23 L6: Project Dynamics Simulation JL		Sep 25 L7: Strategic Project Management JL <i>Project Approval given</i>
Sep 30 L8: Design Structure Matrix Method dWo <i>HW3 out</i>		Oct 2 L9: Iteration Models for Projects dWo <i>HW1 due</i>
Oct 7 L10: SD Cases, Lessons, Extensions JL		Oct 9 L11: Critical Chain Method dWo <i>HW2 due</i>
Oct 14 L12: Earned Value Management dWo <i>HW4 out</i>		Oct 16 Case 1: BAE Denver Airport dWo
Oct 21 <i>Business Trip Week</i> Case 2: Globalstar/Iridium Constellation JS		Oct 23 <i>Business Trip Week</i> L13: Introduction to OPM DD <i>HW3 due</i>
Oct 28 L14: Project-Product Lifecycle Management AS <i>Project Update due, HW5 out</i>		Oct 30 Case 3: Microsoft Office 2000 DD <i>HW5 out</i>
Nov 4 L15: Project Risk Management dWo <i>HW4 due</i>		Nov 6 Case 4: Mission to Mars G <i>HW6 out</i>
Nov 11 No Class, Veteran's Day Vacation		Nov 13 L16: Managing International Projects RM <i>HW5 due</i>
Nov 18 L17: Project Management Resources PH		Nov 20 L18: Human Aspects of Project Management dWo
Nov 25 L19: Cost Estimation in Projects RV <i>HW6 due</i>		Nov 27 No Class, Thanksgiving Vacation
Dec 2 (3 hr class) Project Presentations all		Dec 4 (3 hr class) Project Presentations all <i>Project due</i>
Dec 9 L20: Class Summary – Success Factors dWo		

Instructors: dWo: Olivier de Weck, JL: Jim Lyneis, DD: Dov Dori, AS: Amira Sharon, JS: Joel Schindall, RM: Rolf Maisch, PH: Pat Hale, RV: Ricardo Valerdi, G: Guest

Appendix C – Stage (1) participants UAV assignment

ESD.36 System Project Management

Fall 2008

System Design and Management Program (SDM)

Prof. Olivier de Weck

deweck@mit.edu

Homework 5

2008

Out: October 30,

Model-Based Project Management

13, 2008, 3pm

Due: November

Learning objectives

In this homework you are requested to create two project plan versions by using two different representations: A Gantt chart model and an Object Processes Methodology (OPM) model, based strictly on the text given in HW2. You will be asked to do this in a specific order without backtracking and document your reflections on the way you did your project planning.

Readings (on Stellar)

- [1] Dov Dori, *Object-Process Methodology Basics*
- [2] Amira Sharon, Valeria Perelman, and Dov Dori (2008). *A Project-Product Lifecycle Management approach for improved systems engineering practices*. INCOSE 2008 Symposium.

Other Resources

- [3] Dov Dori, [Object-Process Methodology - A Holistic Systems Paradigm](#), Springer Verlag, Berlin, Heidelberg, New York, 2002 (ISBN 3-540-65471-2).
- [4] Additional papers on OPM in <http://www.objectprocess.org/>

Instructions

- The OPM model of the project plan requires using OPCAT 4.0 software version. It is a Beta Product for evaluation and academic use only. See Course Site for download instructions and tutorial.
- The Gantt chart model can be done manually or by using software tools such as Microsoft Project or Excel.

Task 1 Plan the Project

Based strictly on the project description given in HW2:

Do this if your last name starts with A-J

Please read carefully the instruction up to the end of section 3 before you start answering!

1. Gantt chart Model

- 1.1 Record you start time. Create the Gantt chart model of the project plan containing all the required work, dependencies, and allocated resources.
- 1.2 Record the time it took you to accomplish the model.
- 1.3 Please reflect on the processes of the model creation and record your thoughts, assumptions and decisions during this process.

2. OPM Model

- 2.1 Record you start time. Create the OPM model of the project plan containing all the required work, dependencies, and allocated resources.
- 2.2 Record the time it took you to accomplish the model.
- 2.3 Please reflect on the processes of the model creation and record your assumptions and decisions during modelling.

Maintaining the order, without backtracking is most important, so please do not go back and forth editing the two models. It is not necessary for the two to be consistent. Rather, you are asked next to reflect on the evolution of your considerations as you move from the Gantt chart model to the OPM model.

3. Reflect on what prompted you to introduce changes, additions or deletions in the OPM model with respect to the previously prepared Gantt chart model.

Do this if your last name starts with K-Z

Please read carefully the instruction up to the end of section 3 before you start answering!

1. OPM Model

Record you start time. Create the OPM model of the project plan containing all the required work, dependencies, and allocated resources.

Record the time it took you to accomplish the model.

Please reflect on the processes of the model creation and record your thoughts, assumptions and decisions during this process.

2. Gantt chart Model

- 2.1 Record you start time. Create the Gantt chart model of the project plan containing all the required work, dependencies, and allocated resources.
- 2.2 Record the time it took you to accomplish the model.
- 2.3 Please reflect on the processes of the model creation and record your assumptions and decisions during modelling.

Maintaining the order, without backtracking is most important, so please do not go back and forth editing both models. It is not necessary for the two to be consistent. Rather, you are asked next to reflect on the evolution of your considerations as you move from the OPM model to the Gantt chart model.

3. Reflect on what prompted you to introduce changes, additions or deletions in the Gantt chart model with respect to the previously prepared OPM model.

- 4 Please compare the project models or representations you have done so far as homeworks, with respect to the following Project Management Considerations. Utilize the excel file entitled HW5 Q4 for this purpose. Wherever you believe a correlation exists between a model and a PM consideration, provide a short written explanation of the relationship and grade its strength numerically (between 1 and 5).

- 5 Unmanned Aerial System (UAS)
 - a. Based on the UAV product presented in HW2, create an OPM Product model for the UAS. Note that the Unmanned Aerial System (UAS) is an upper-level view of the entire system, which consists of an Aerial Segment (containing the UAV and the payload), a Ground Segment, and Data Link (consists of aerial components installed on the Aerial Segment and corresponding equipment installed on the Ground Segment) enabling the communication between the two segments. One Ground Segment can control up to four UAVs and contains a primary Ground Control Station (GCS) along with a Backup GCS (BGCS); each one has a separate shelter with a dedicated generator. The system's primary mission, **Surveillance**, is the function of the system. It is represented as the sole systemic process in SD, the System Diagram, OPM's top level OPD. In order for the system to carry out its primary mission, the entire system should also entail the human constituents: the UAV pilot, payload operator, mission planner, and ground staff. The last two are required for the preparations of the UAS prior to each sortie. The user is some high command and control entity (such as the air force, navy or other army), which makes use of the **Surveillance** deliverables.

 - b. Reflect on the processes of the model creation and record the assumptions and decisions you made while you were engaged in modelling the UAS.

 - c. Now, having modeled the UAS product, please specify and argue for what, if anything, would you change, add, or delete from the project model?

- 6 Propose at least 10 different relationships that can and should be established between entities of the OPM product model and entities of the OPM project model to enhance traceability between the product and the project model for better project planning and control. Please provide the rationale for each proposed relationship and indicate if, and if so where, this relationship is usually maintained.

Challenge Questions

- 7 Is the critical path obtained based on the Gantt and by the OPM (questions 1 and 2) identical in both cases? Please elaborate.
- 8 You have now been promoted to be the manager of a portfolio of UAS product line. What would you change in your UAS project plan?

Appendix D – Stage (2) participants UAV assignment

Integration and design verification of system product

Winter 2009

Dr. Itzhack Glazer

**Planning Using a Unified Product and Project Lifecycle Model (PPLM)
for Systems Engineering**

Amira Sharon

Homework:

11, 2009

Out: November

Model-Based Project-Product Planning

18, 2009, 5pm

Due: November

Learning objectives

In this homework you are requested to create three project plan versions by using three different representations: (1) an Activities Network Plan (AON type) model, (2) a Gantt chart model, and (3) an Object Processes Methodology (OPM) model, and document your reflections on the way you did your project planning.

Instructions

- The project activities network model and the Gantt chart model can be done manually or by using software tools such as Microsoft Project or Excel.
- The OPM model of the project plan requires using OPCAT software version. See FTP Site for download instructions and tutorial.

Situation

You have recently been promoted to Project Manager at *New Millennium Aerospace (NMA) Inc.*, a leading manufacturer of unmanned aerial vehicles (UAVs) for the government. Your new job is to plan and execute the development project for a UAV, to be used for surveillance purposes. A rough specification and sketch of the new vehicle is shown in Figure 1. The payload is provided by the government as modified GFE (government furnished equipment), while the engine will be supplied by a well-established commercial company (ECC) under a subcontract. The remainder of the vehicle, including integration and testing is NMA's - and therefore your - responsibility. Your task today is to create a project schedule, find the critical path and to estimate the finish time of the project. The subsequent project description is hypothetical, but will help you establish the plan.

UAV Project Description (NMA-X1)

The UAV “pusher” vehicle concept is shown in Figure 1. In a pusher aircraft, the engine is rear-mounted which can lead to higher propulsive efficiency. The vehicle can be decomposed into the following assemblies: fuselage (houses the avionics suite), wings, empennage, payload (a visual and an IR camera, incl. transmitter) and the engine (incl. propeller).

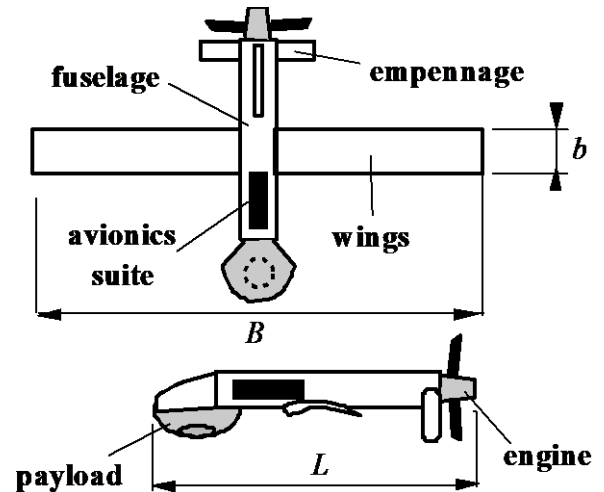


Fig 1. UAV concept, Specifications:
L=2000 mm, B=3500 mm, b=500 mm

What follows is a description of the tasks necessary to develop the vehicle, incl. the dependencies between tasks. The task (“job”) descriptions are underlined the first time they are mentioned, and the task ID and normal duration in working days are given in parentheses. Example: engine integration (x,17), means that there is a task called “engine integration”, identified by the symbol “x”, whose nominal duration is 17 working days.

After the project start (a,0) you first have to complete the overall requirements definition (b,10) step. Once this is accomplished you can carry out the following jobs in parallel: negotiate the engine specification (c,5) with ECC, define your payload specification (d,5), determine the vehicle layout (e,8) and write the software specification (g,12). You can initiate(GFE) avionics design (f,15) after (b,10), however the tasks (c,5) and (d,5) must also have been completed before the GFE design can be started, so that the avionics will be able to control both the engine and payload in a synchronized fashion.

Once the engine specs (c,5) have been defined, the supplier (ECC) informs you that it will take 30 days for engine development (i,30) based on experience with a previous variant. Once engine development is complete, delivery and checkout (n,2) can take place at NMA’s facilities. After (d,5) is done, payload development (j,15) can take place in parallel with engine development. Once the payload is developed (j,15) and the engine delivered (n,2), both the engine and payload are integrated (electrically) in the power system integration (o,10) step.

Fuselage design (k,17) and empennage/wing design (l,15) begin in parallel after the vehicle layout (e,8) has been established. Internal fittings (m,8) can be designed after these two jobs are completed. Also, structural airframe prototyping (r,8) consists of building a physical frame for the vehicle after jobs (k,17) and (l,15) are completed.

Once avionics design (f,15) has been completed, this leads to avionics delivery and checkout (p,12) and subsequent avionics/software integration (q,5). Obviously, in order for this last step to take place, software development (h,25) which depends both on (g,12) and (f,15) must have also been completed.

The project is continued by performing vehicle integration (s,10) which requires prior completion of power system integration (o,10), airframe prototyping (r,8) and avionics/software integration (q,5). After vehicle integration (s,10) and internal fitting design (m,8) have been achieved, final vehicle assembly (t,5) can begin. After final

assembly, the completed vehicle is subjected to laboratory testing (u,5), followed by an outdoor flight test campaign (v,10) leading to completion of the prototype development project, finish (w,0).

Notes:

- task descriptions are underlined
- (n,25) means that the task is tagged as “n” and is expected to take 25 work days
- task descriptions are hypothetical, but in a notionally meaningful sequence
- task durations are hypothetical (on the short side)

Plan the Project

1. Construct a task table from the NMA-X1 project description. Clearly designate each task with its tag, description and identify immediate predecessors and nominal task completion times. Try to arrange the task table in “technological order”.
2. Create a project Activities Network Plan (ANP) graph (AON type) model by hand or using a computer program.
Write your reflection of the process you undergo while creating the model, and record your thoughts, assumptions and decisions during this process.
3. Create the Gantt chart model of the project plan by hand or using a computer program.
Write your reflection of the process you undergo while creating the model, and record your thoughts, assumptions and decisions during this process.
4. Create the OPM model of the project plan containing all the required additional objects in the model.
Write your reflection of the process you undergo while creating the model, and record your thoughts, assumptions and decisions during this process.

Please submit your homework answer as follows:

- (1) The provided rubric file, filled.***
- (2) Answer to Question 1 - Task table – as a WORD document.***
- (3) Answer to Question 2 - Activities Network Plan model – as a WORD document.***
- (4) Answer to Question 3 - Gantt chart model – as a WORD document, and additional corresponding file if you have used a software tool.***
- (5) Answer to Question 4 – OPM model – as a WORD document and corresponding OPCAT (*.opz) file as well.***
- (6) All of the above shall be submitted electronically, and a hard copy of all the above shall be submitted as well.***

Appendix E – Stage (3) participants CT scanner assignment

Development and Integration of CT scanner

The Product

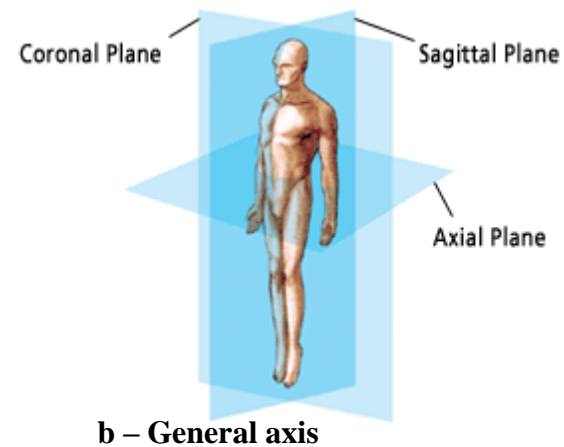
A **Computed Tomography (CT) scanner** is a medical imaging device used to generate 3D images of the inside of a patient body (or of an object in general) in a non-invasive manner. A series of 2D X-ray images that are taken around a single axis of rotation are used to create an anatomical 3D image.

The CT scanner is built from the following **main** components (see figure 1a):

- **Table** (bed), on which the patient is lying.
- **Gantry** which contains under its covers (see figure 2a), **X-ray tube** and **detector** on a circular rotor. Slip rings inside the gantry allow continuous gantry rotation (see figure 2b). The gantry has a big “hole” in the middle of it; this is the gantry **bore**, and the patient (through the table) can pass through it.
- **Console** (acquisition station) which is used to control the scanner, as well as process and display the data.

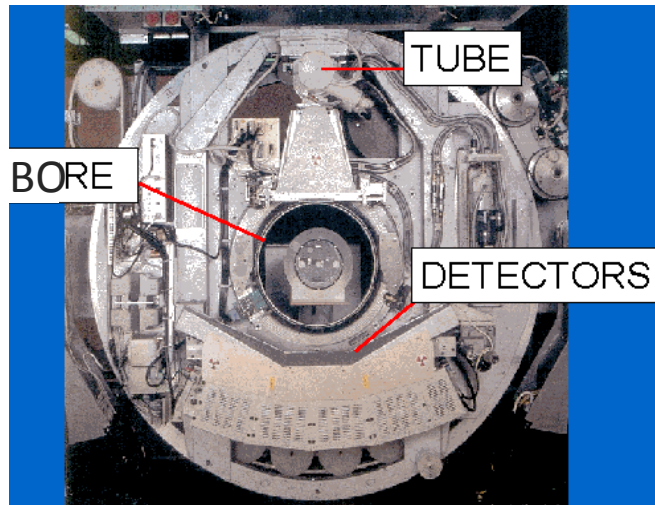


a - General CT scanner components

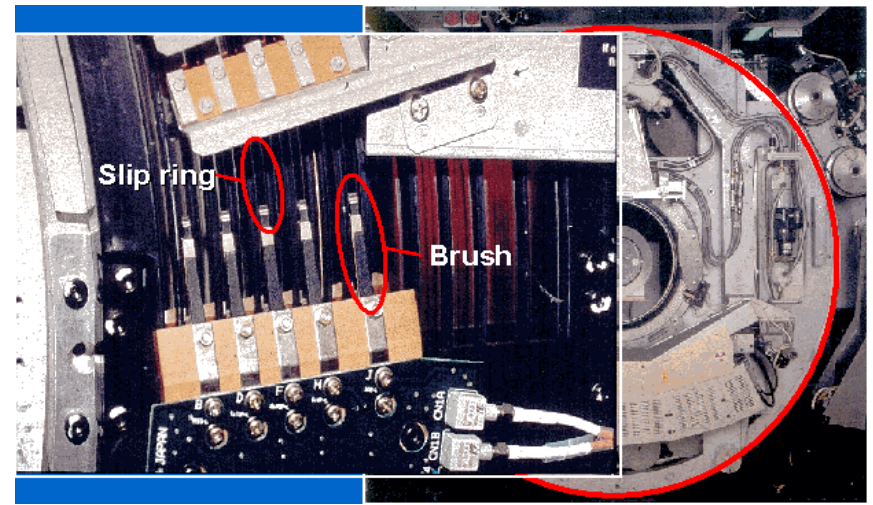


b – General axis

Figure 1 – General CT scanner



a – Main gantry components



b – Gantry rotation

Figure 2 - Gantry inside

A certified **technician** is the operator of the CT scanner. The technician is responsible to scan the patient and produce **diagnostic images**, which are later transferred to a physician for diagnosis and treatment recommendations.

The **gantry** (which contains the X-ray tube and detector) is required to **rotate** through the whole scan, in order to acquire the patient body from different angles. The **table** is also moving during the entire scan in the **longitudinal** direction, to allow scanning the required length of the patient body.

After the data was acquired, it should be processed on the console, to produce the final diagnostic images. These images are composed of "cuts" in the 3 main planes: **axial**, **coronal** and **sagittal** (see figure 1b).

Assignment	
Next you will find diagrams of two equivalent models:	
(1) A Gantt chart model of the CT scanner and an OPM-based product model. (2) A combined OPM-based Project-Product model of the CT scanner.	
•	Start by reviewing the diagrams; comprehension questions of the models will follow. It is recommended to go over across all the diagrams and get an overall point of view of the project-product system models, before answering questions.
•	For each question, answer for each one out of the two models as follows:
	(1) Mark one of the options: (a) can be easily found out, (b) can be found out, (c) can be hardly found out, or (d) impossible to find out.
	(2) Provide a short written answer.
	(3) Make up your own two questions and answer them.
•	Answer the last question.

	Question	Answer based on PPLM Model		Answer based on Gantt Model	
		Mark one option	Full answer	Mark one option	Full answer
1	How long is the project planned to take?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
2	How long is the first development cycle planned to take?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
3	When does the 1st <i>cycle</i> start in regard to <i>Definition</i> ? What is the reason?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
4	How many System Builds are planned?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	

	Question	Answer based on <u>PPLM Model</u>		Answer based on <u>Gantt Model</u>	
		Mark one option	Full answer	Mark one option	Full answer
5	Are there system builds that are not covered by the plan?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
6	Is the plan complete?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
7	Are there components of the CT scanner that are not covered by the plan?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
8	Are there components of the CT scanner that are not required according to the plan?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	

	Question	Answer based on PPLM Model		Answer based on Gantt Model	
		Mark one option	Full answer	Mark one option	Full answer
9	Is there a parameter of the CT scanner that is not covered by the plan?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
10	What specifications are required for the first development cycle?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
11	What specifications are required for the second development cycle?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
12	What is the outcome of the Definition activity?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	

	Question	Answer based on PPLM Model		Answer based on Gantt Model	
		Mark one option	Full answer	Mark one option	Full answer
13	How many specifications are contained in the plan?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
14	When is the BIT planned?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
15	What is required for the BIT?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
16	What product processes are planned to be achieved by the end of the 2 nd development cycle?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	

	Question	Answer based on PPLM Model		Answer based on Gantt Model	
		Mark one option	Full answer	Mark one option	Full answer
17	What product processes are planned to be achieved by System Build 3?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
18	What product processes are planned to be achieved by System Build 4?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
19	How many product processes?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	
20	How many product requirements?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out (c) can be hardly found out (d) Impossible to find out		(b) can be found out (c) can be hardly found out (d) Impossible to find out	

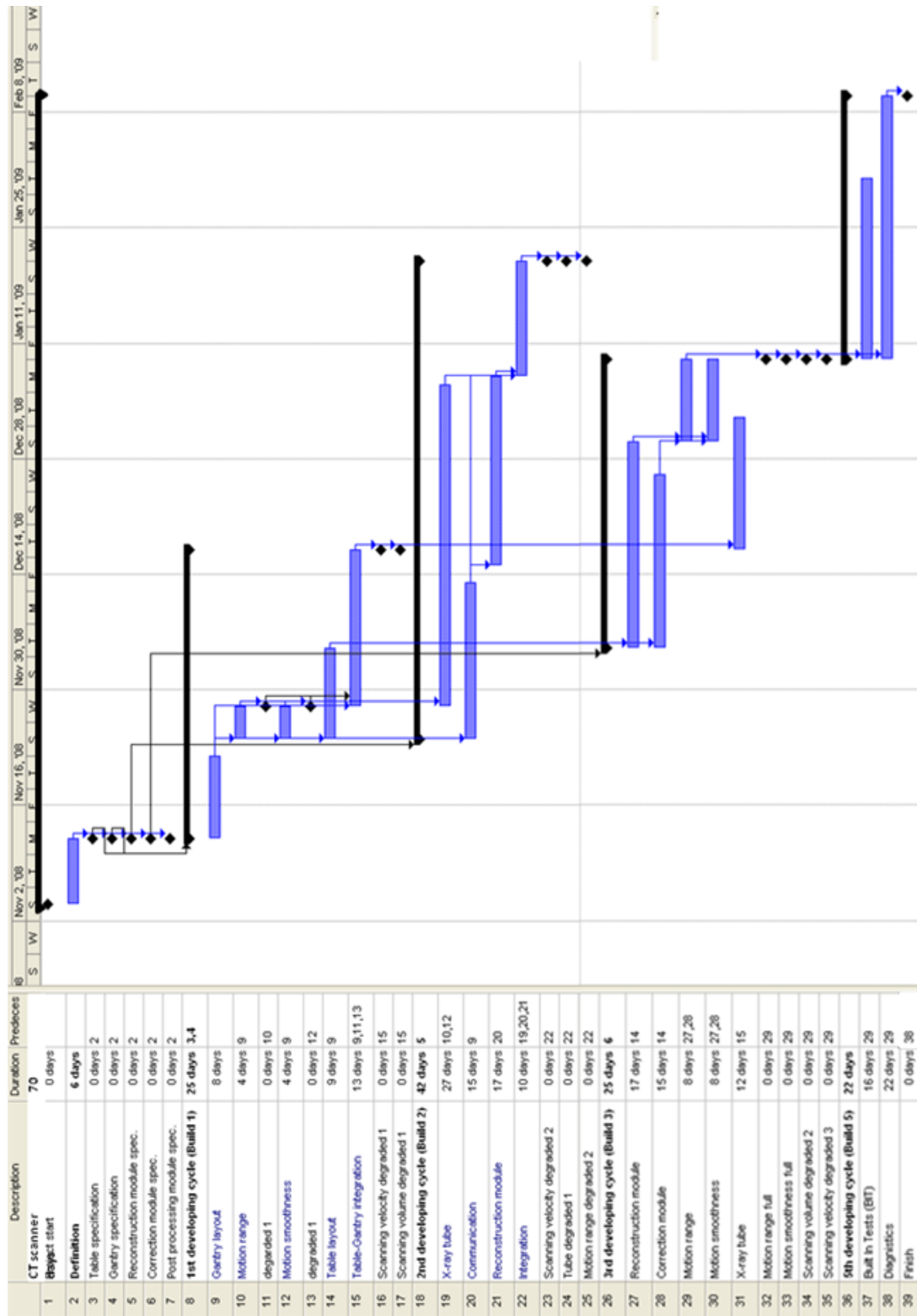
	Question	Answer based on PPLM Model		Answer based on Gantt Model	
		Mark one option	Full answer	Mark one option	Full answer
21	How many design reviews are planned?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
22	Are all the product processes covered by the plan?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
23	If the <i>table specification</i> is delayed, how is the plan affected?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
24	If a <i>software specification</i> is delayed, how is the plan affected?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	

	Question	Answer based on PPLM Model		Answer based on Gantt Model	
		Mark one option	Full answer	Mark one option	Full answer
25	What is required in order to achieve <i>Data acquisition</i> ?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
26	What is required for <i>Image reconstruction</i> in System Build #3?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
27	What software module is required for <i>Image correction</i> in System Build #3?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
28	How is <i>scanning velocity</i> defined in regard to System Build #4?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
		(c) can be hardly found out		(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	

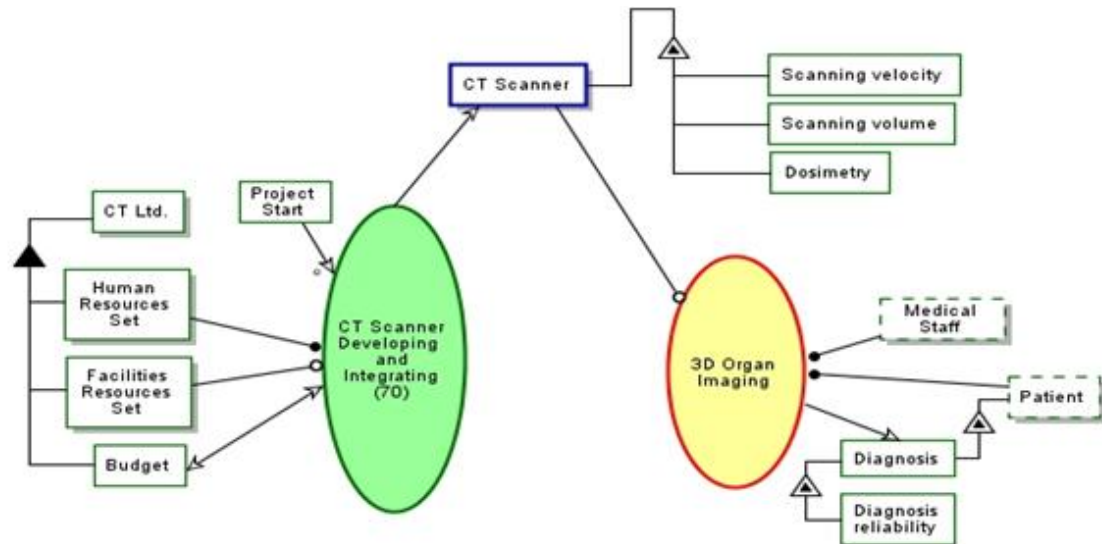
	Question	Answer based on PPLM Model		Answer based on Gantt Model	
		Mark one option	Full answer	Mark one option	Full answer
29	How long does the <i>Integration</i> in System Build #3 take?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
30	How long does System Build #5 take?	(c) can be hardly found out	Based on diagrams:	(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	
31	How is the X-ray tube treated in the 1 st development cycle?	(a) can be easily found out	Based on diagrams:	(a) can be easily found out	
		(b) can be found out		(b) can be found out	
32	What product characteristics are planned to be achieved by the end of the 2 nd development cycle?	(c) can be hardly found out	Based on diagrams:	(c) can be hardly found out	
		(d) Impossible to find out		(d) Impossible to find out	

	Question	Answer based on PPLM Model	Answer based on Gantt Model		
		Mark one option	Full answer	Mark one option	Full answer
33	What product characteristics are planned to be achieved by the end of the 3 rd development cycle?	(a) can be easily found out (b) can be found out (c) can be hardly found out (d) Impossible to find out	Based on diagrams:	(a) can be easily found out (b) can be found out (c) can be hardly found out (d) Impossible to find out	
34		(a) can be easily found out (b) can be found out (c) can be hardly found out (d) Impossible to find out	Based on diagrams:	(a) can be easily found out (b) can be found out (c) can be hardly found out (d) Impossible to find out	
35		(a) can be easily found out (b) can be found out (c) can be hardly found out (d) Impossible to find out	Based on diagrams:	(a) can be easily found out (b) can be found out (c) can be hardly found out (d) Impossible to find out	
36	What are 3 most important things you find missing in the plan?	Based on diagrams:			

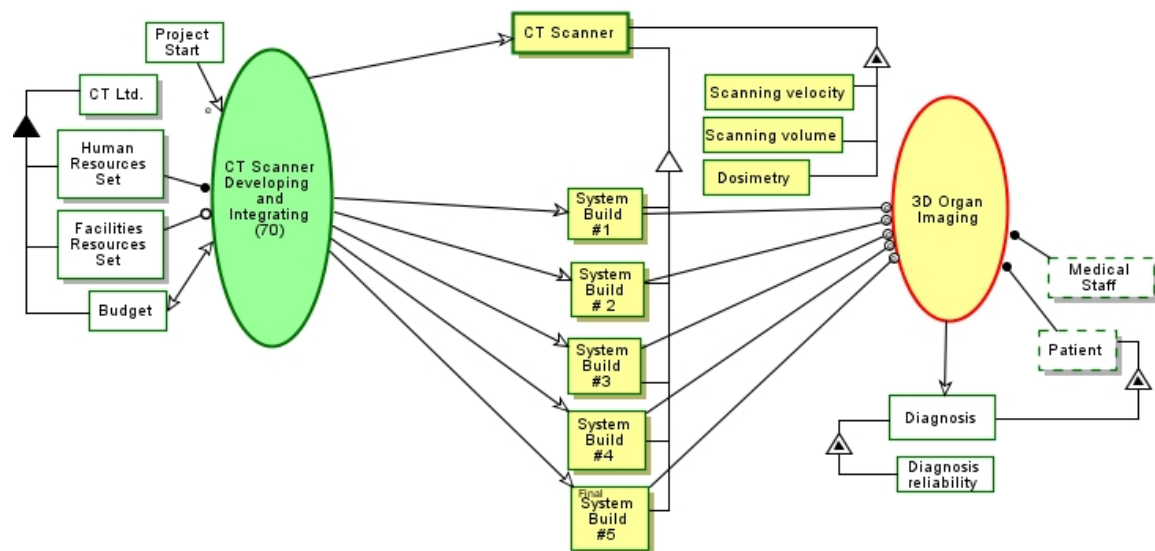
Appendix F – The provided Gantt chart of the CT scanner plan



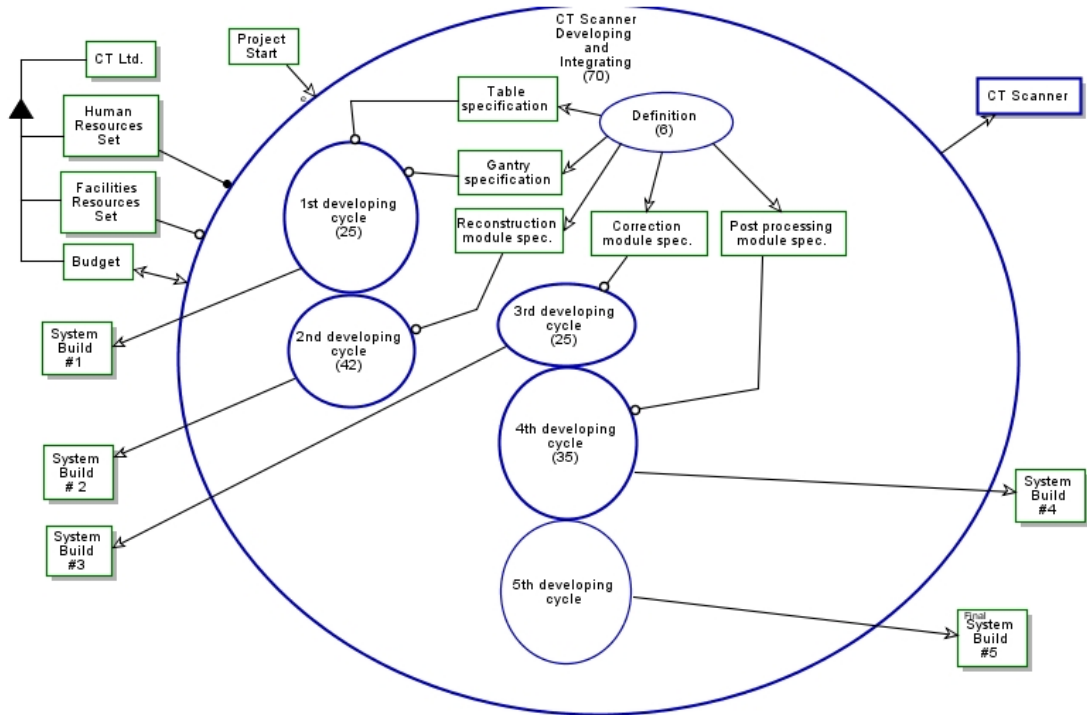
Appendix G – The provided CT scanner PPLM Model



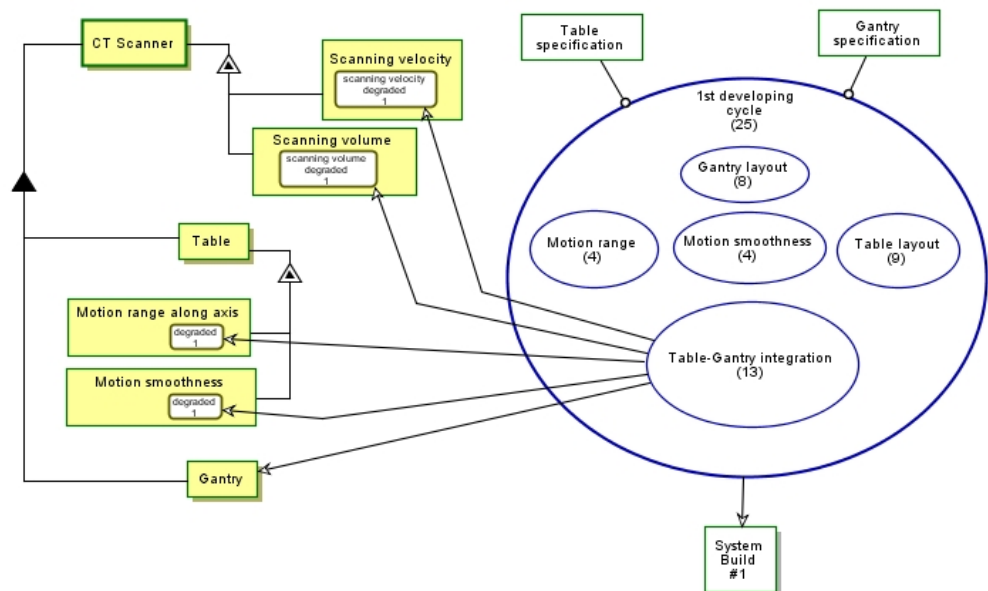
SD1 - CT scanner Project-Product Lifecycle Management in-zoomed (1)



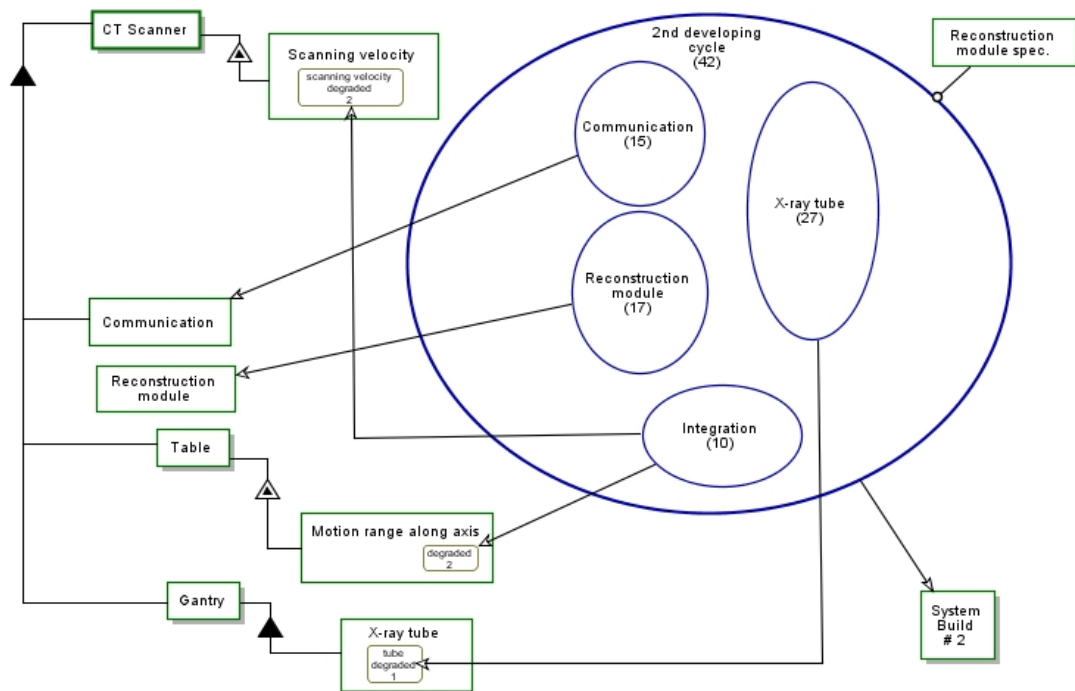
SD1 - CT scanner Project-Product Lifecycle Management in-zoomed (2)



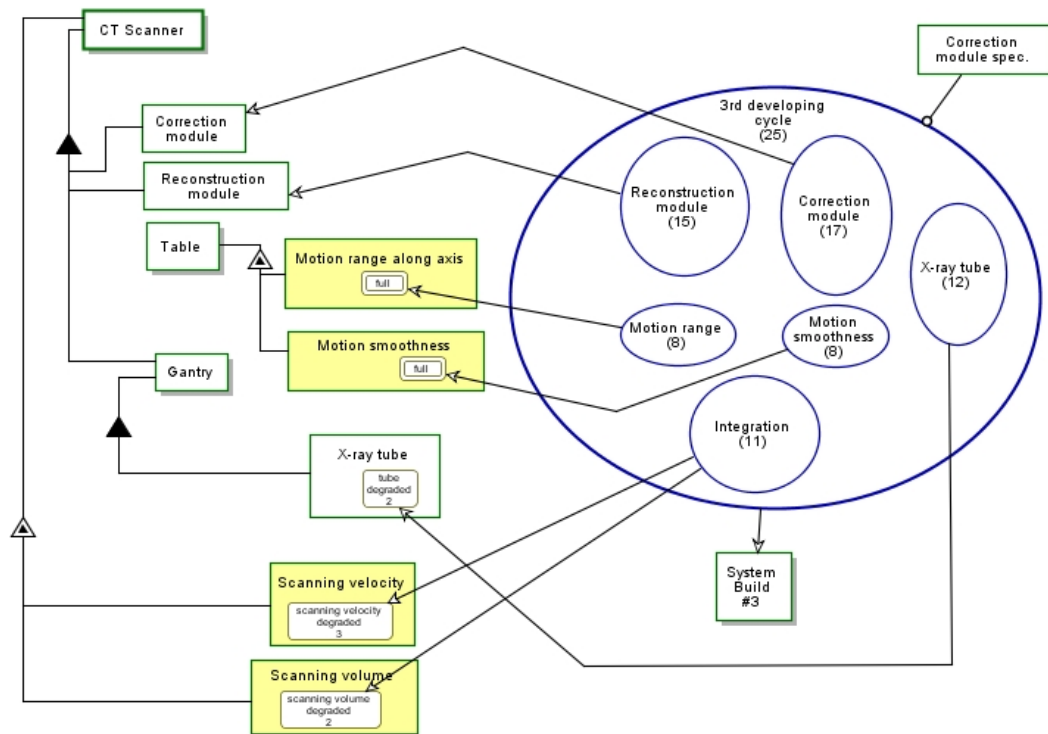
SD1.1 - CT scanner Developing and Integrating in-zoomed



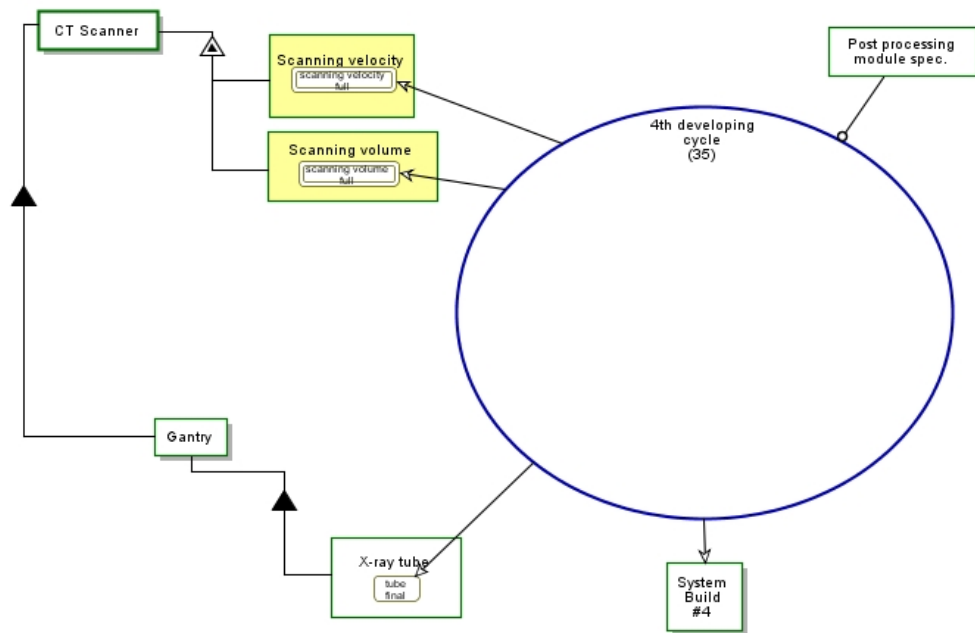
SD1.1.1 - 1st developing cycle in-zoomed



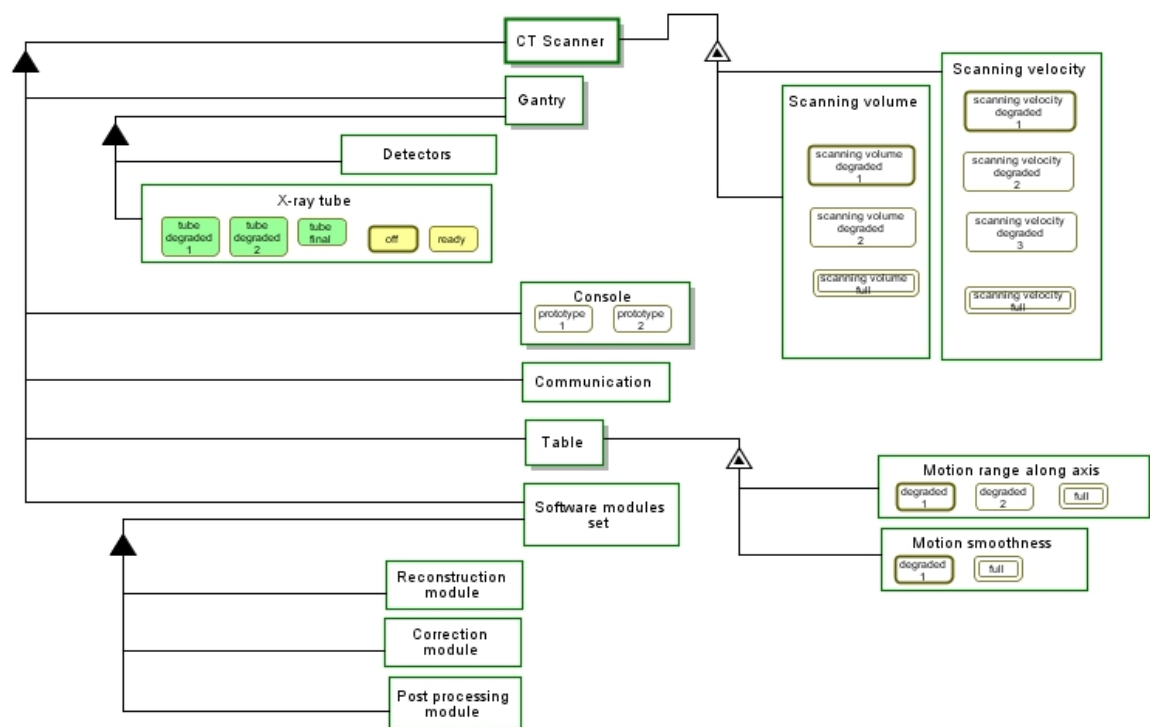
SD1.1.2 - 2nd developing cycle in-zoomed



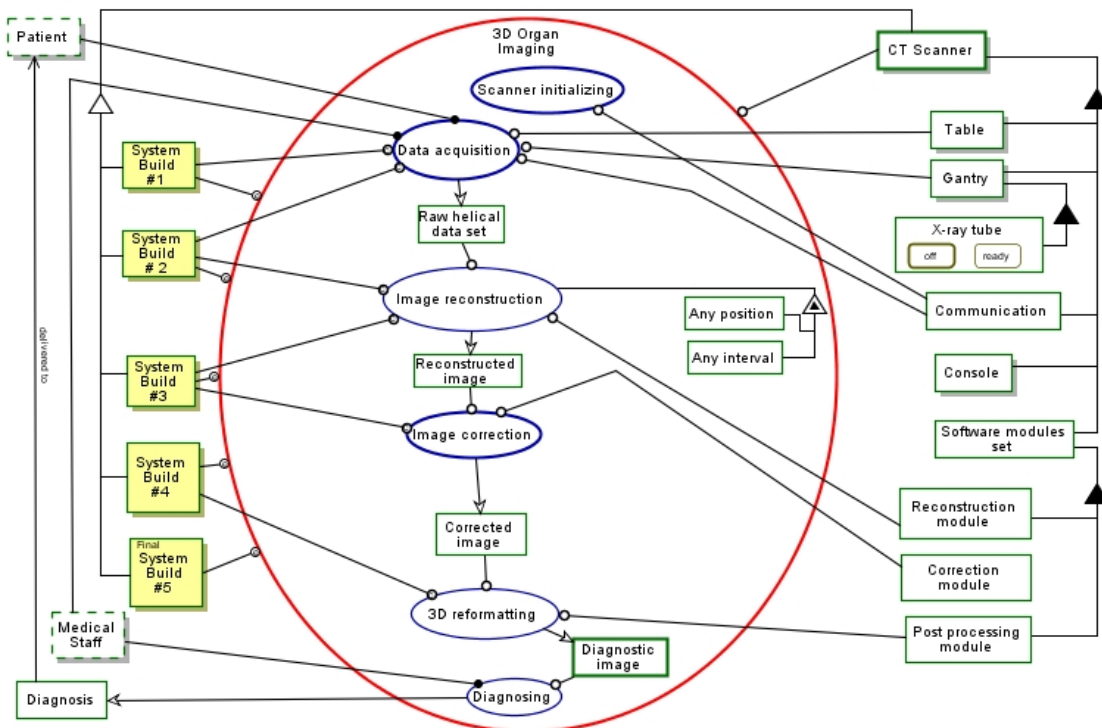
SD1.1.3 - 3rd developing cycle in-zoomed



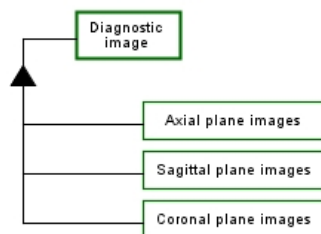
SD1.1.4 - 4th developing cycle in-zoomed



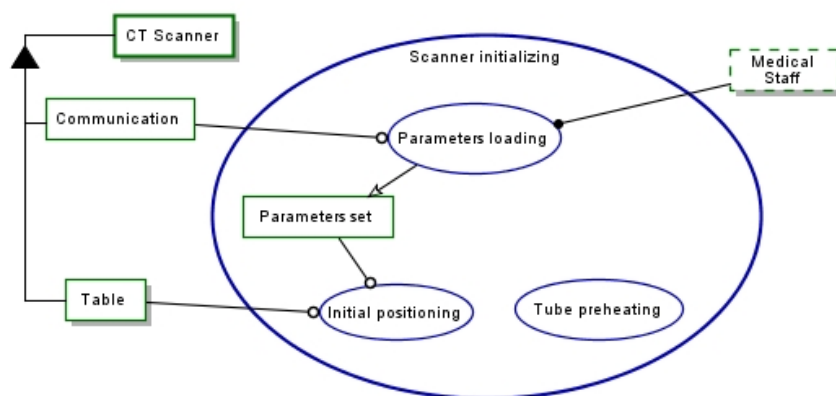
UD5 - CT scanner unfolded



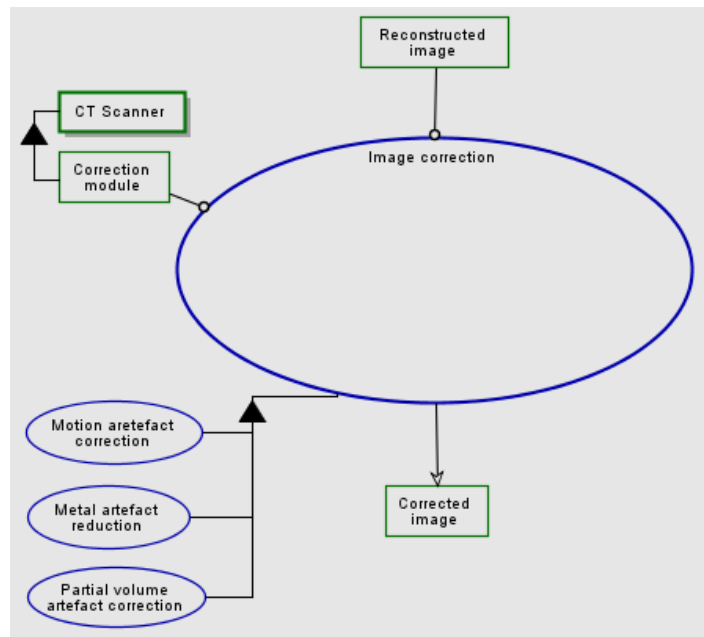
SD1.2 - 3D Organ Imaging in-zoomed



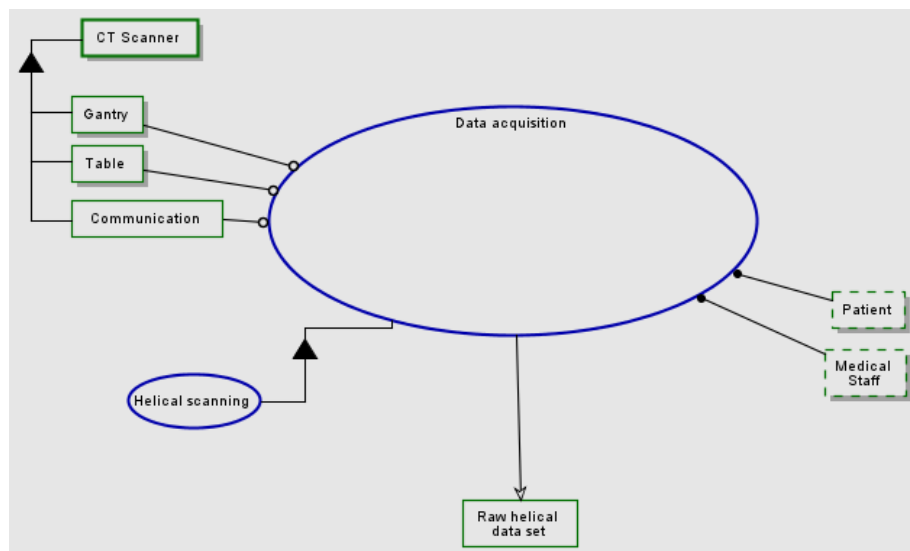
UD4 - Diagnostic image unfolded



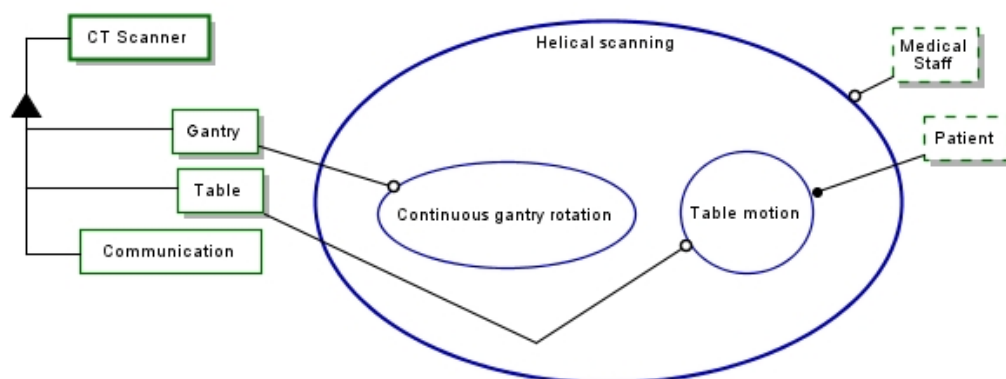
SD1.2.1 - Scanner initializing in-zoomed



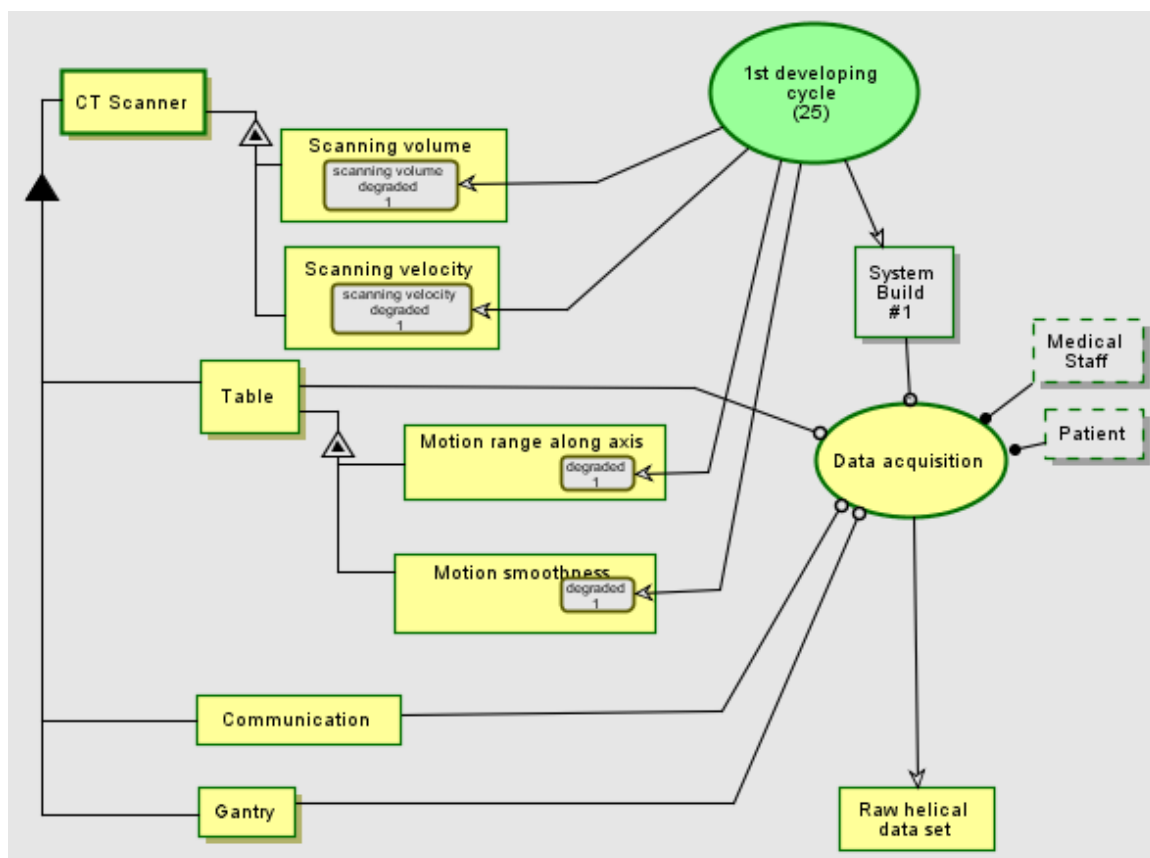
SD1.2.2 - Image correction in-zoomed



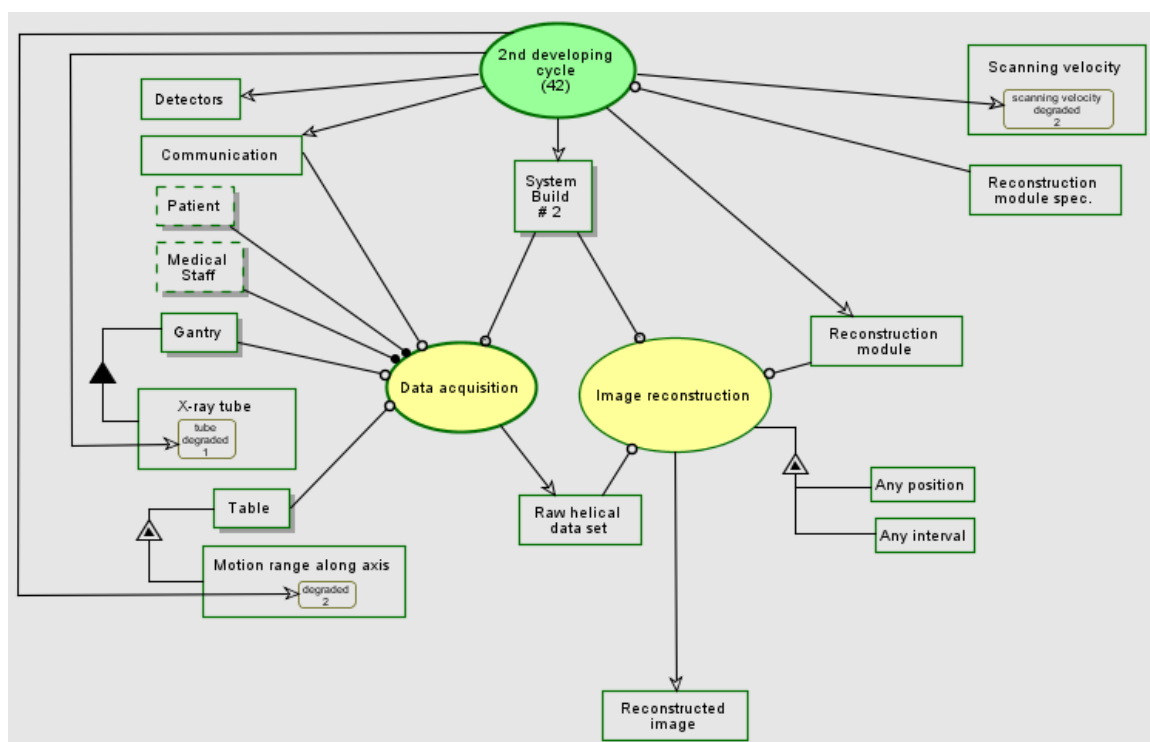
SD1.2.3 - Data acquisition in-zoomed



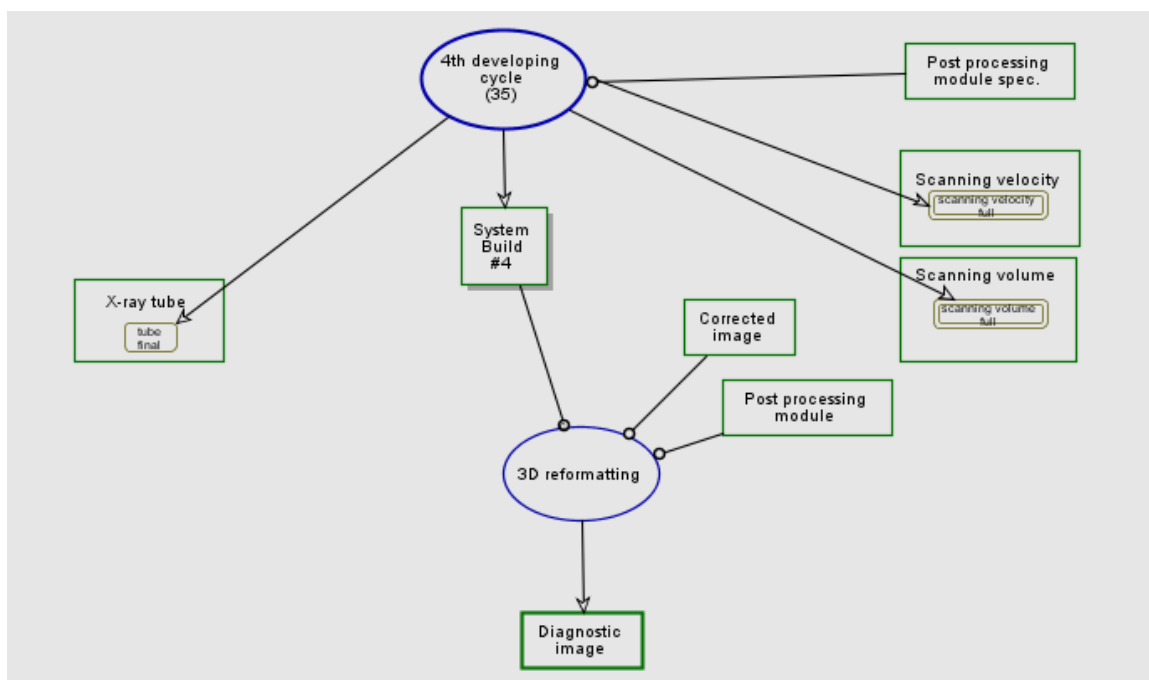
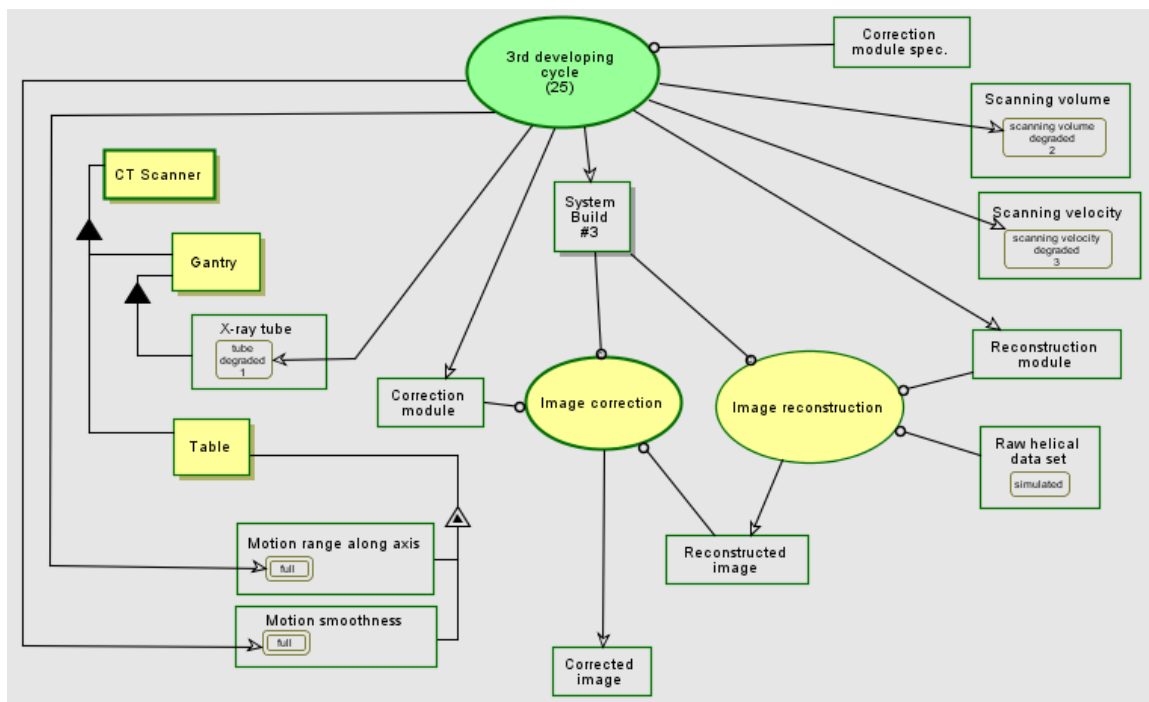
SD1.2.3.1 - Helical scanning in-zoomed



System Build #1 view



System Build #2 view



Appendix H – OPM Syntax and Semantics

ENTITIES

STRUCTURAL LINKS & COMPLEXITY MANAGEMENT



ENABLING AND TRANSFORMING PROCEDURAL LINKS



EVENT, CONDITION, AND INVOCATION PROCEDURAL


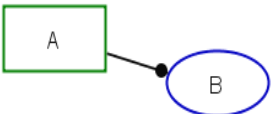
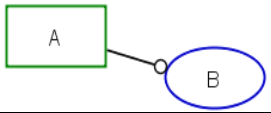
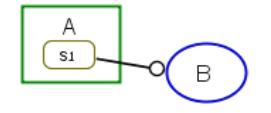
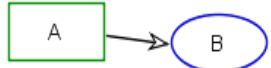
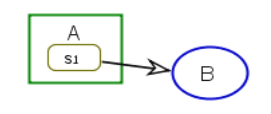
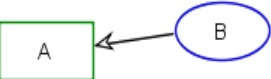
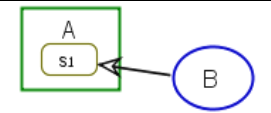
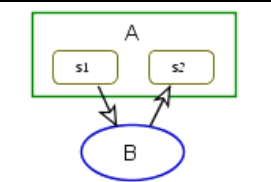
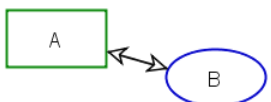



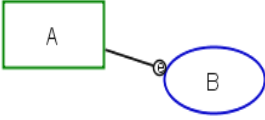
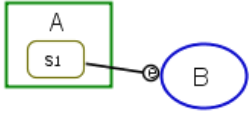
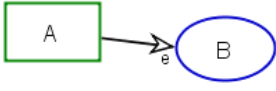
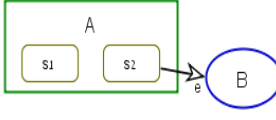
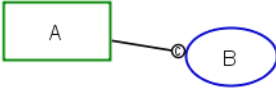
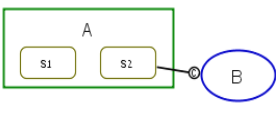

LINKS



ENTITIES				
Name		Symbol	OPL	Definition
Things	Object		B is physical. (shaded rectangle)	An object is a thing that exists. A process is a thing that transforms at least one object. Transformation is object generation or consumption, or effect—a change in the state of an object.
	Process		C is physical and environmental. (shaded dashed rectangle) E is physical. (shaded ellipse) F is physical and environmental. (shaded dashed ellipse)	
State			A is s1 . B can be s1 or s2 . C can be s1 , s2 , or s3 . s1 is initial. s3 is final.	A state is situation an object can be at or a value it can assume. States are always within an object. States can be initial or final.

9.2.1 STRUCTURAL LINKS & COMPLEXITY MANAGEMENT				
Name		Symbol	OPL	Semantics
Fundamental Structural Relations	Aggregation-Participation		A consists of B and C.	A is the whole, B and C are parts.
			A consists of B and C.	
	Exhibition-Characterization		A exhibits B, as well as C.	Object B is an attribute of A and process C is its operation (method). A can be an object or a process.
			A exhibits B, as well as C.	
	Generalization-Specialization		B is an A. C is an A.	A specializes into B and C. A, B, and C can be either all objects or all processes.
			B is A. C is A.	
	Classification-Instantiation		B is an instance of A. C is an instance of A.	Object A is the class, for which B and C are instances. Applicable to processes too.
	Unidirectional & bidirectional tagged structural links			A relates to B. (for unidirectional) A and C are related. (for bidirectional)
In-zooming		A exhibits C. A consists of B. A zooms into B, as well as C.	Zooming into process A, B is its part and C is its attribute.	
		A exhibits C. A consists of B. A zooms into B, as well as C.	Zooming into object A, B is its part and C is its operation.	

9.2.2 ENABLING AND TRANSFORMING PROCEDURAL LINKS				
Name	Symbol	OPL	Semantics	
Enabling links	Agent Link		A handles B .	Denotes that the object is a human operator.
	Instrument Link		B requires A .	"Wait until" semantics: Process B cannot happen if object A does not exist.
	State-Specified Instrument Link		B requires s1 A .	"Wait until" semantics: Process B cannot happen if object A is not at state s1.
Transforming links	Consumption Link		B consumes A .	Process B consumes Object A.
	State-Specified Consumption Link		B consumes s1 A .	Process B consumes Object A when it is at State s1.
	Result Link		B yields A .	Process B creates Object A.
	State-Specified Result Link		B yields s1 A .	Process B creates Object A at State s1.
	Input-Output Link Pair		B changes A from s1 to s2 .	Process B changes the state of Object A from State s1 to State s2.
	Effect Link		B affects A .	Process B changes the state of Object A; the details of the effect may be added at a lower level.

9.2.3 EVENT, CONDITION, AND INVOCATION PROCEDURAL LINKS							
Name	Symbol	OPL	Semantics				
Instrument Event Link		A triggers B . B requires A .	Existence or generation of object A will attempt to trigger process B once. Execution will proceed if the triggering failed.				
State-Specified Instrument Event Link		A triggers B when it enters s1 . B requires s1 A .	Entering state s1 will attempt to trigger the process once. Execution will proceed if the triggering failed.				
Consumption Event Link		A triggers B . B consumes A .	Existence or generation of object A will attempt to trigger process B once. If B is triggered, it will consume A. Execution will proceed if the triggering failed.				
State-Specified Consumption Event Link		A triggers B when it enters s2 . B consumes s2 A .	Entering state s2 will attempt to trigger the process once. If B is triggered, it will consume A. Execution will proceed if the triggering failed.				
Condition Link		B occurs if A exists.	Existence of object A is a condition to the execution of B. If object A does not exist, then process B is skipped and regular system flow continues.				
State-Specified Condition Link		B occurs if A is s2 .	Existence of object A at state s2 is a condition to the execution of B. If object A does not exist, then process B is skipped and regular system flow continues.				
Invocation Link		B invokes C .	Execution will proceed if the triggering failed (due to failure to fulfill one or more of the conditions in the precondition set).				