

DISSERTATION

COST OPTIMIZATION IN REQUIREMENTS MANAGEMENT FOR SPACE SYSTEMS

Submitted by

Tami E. Katz

Department of Systems Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2021

Doctoral Committee:

Advisor: Steve Simske

Ron Sega

Erika Miller

John Macdonald

Copyright by Tami E. Katz 2021

All Rights Reserved

ABSTRACT

COST OPTIMIZATION IN REQUIREMENTS MANAGEMENT FOR SPACE SYSTEMS

When producing complex space systems, the transformation of customer needs into a realized system includes the development of product requirements. The ability to generate and manage the requirements can either enable the overall system development or drive significant cost and schedule impacts. Assessing practices in the industry and publications, it is observed that there is a substantial amount of documented approaches to address requirement development and product verification, but only a limited amount of documented approaches for requirements management. A complex system can have tens of thousands of requirements across multiple levels of development which, if not well managed, can lead to hidden costs associated with missed requirements and product rework. With current space system projects being developed at a rapid pace using more cost constrained approaches such as fixed budgets, an investigation into more efficient processes, such as requirements management, can yield methods to enable successful, cost effective system development.

To address the optimal approach of managing requirements for complex space systems, this dissertation assesses current practices for requirements management, evaluates various contributing factors towards optimization of project costs associated with this activity, and proposes an optimized requirements management process to utilize during the development of space systems. Four key areas of process control are identified for requirements management optimization on a project, including utilization of a data focused requirements management approach, development (and review) of requirements using a collaborative software application, ensuring the requirement set is a consolidated with an appropriate amount of requirements for the project, and evaluating when to officially levy requirements on the product developers based on requirement maturation stability.

Multiple case studies are presented to evaluate if the proposed requirements management process yields improvement over traditional approaches, including a simulation of the current state and proposed

requirements management approaches. Ultimately, usage of the proposed optimized set of processes is demonstrated to be a cost effective approach when compared against traditional processes that may adversely impact the development of new space systems.

ACKNOWLEDGEMENTS

Research and generation of new material is not a solo activity, and significant appreciation goes to those that helped me produce this dissertation. The support from those that gave of their time and shared their experiences working on projects has been invaluable, and greatly added to the depth of this research. Thank you to Lou Wheatcraft, Katie Trase, Ken Eastman, Raymond Wolfgang, Joel Knapp, Kevin Orr, David Hill, Angie Wise and John Curry.

Those that passionately seek to improve the field of requirements in a voluntary capacity have also contributed to this project; thank you to the INCOSE Requirements Working Group co-chairs, Lou, Kevin, Mike Ryan and Rick Zinni for helping me explore better ways to address requirements on a project. Additional appreciation goes to Saulius Pavalkis at Dassault Systèmes for his support during the development of the requirements management model and simulation, his advice and troubleshooting recommendations helped to keep the modeling progress on course.

I would like to acknowledge the companies that I have worked for during this effort, Sierra Nevada Corporation and Ball Aerospace, both of which supported my research and enabled me to continue my education while I was employed full time. The support and advice of my colleague, Erik Wilkinson, has significantly enabled my ability to question established methods to seek more optimal ways. I will always be grateful for his support and inspiration to try new things.

Significant support was provided by Dr. Steve Simske, my advisor and guide for pursuing this research, thank you for challenging me to push my ideas further.

And finally, my love and appreciation to my husband, Mike Katz, and children Janelle and Jason Katz, for supporting my long nights and weekends researching requirements engineering and writing this paper, thank you for your incredible encouragement!

TABLE OF CONTENTS

ABSTRACT.....	ii
ACKNOWLEDGEMENTS.....	iv
LIST OF TABLES.....	x
LIST OF FIGURES.....	xii
Chapter 1: INTRODUCTION.....	1
1.1 The Challenge with Requirements.....	1
1.2 Dissertation Organization.....	3
Chapter 2: CONCEPT OF REQUIREMENTS ENGINEERING.....	5
2.1 The Systems Engineering Process and the Project Life Cycle.....	5
2.2 Overview of Requirements Engineering.....	8
2.2.1 Terminology and Authoritative Publications.....	8
2.2.2 The Requirements Development Processes.....	12
2.2.3 Requirements Development Among Different Organizations.....	20
2.2.4 Concept of Requirements Traceability.....	22
2.2.5 Results of Poor Requirements Engineering.....	26
2.3 Requirements Engineering Conclusions.....	29
Chapter 3: INVESTIGATION INTO REQUIREMENTS MANAGEMENT.....	30
3.1 Current Approaches to Requirements Management.....	30
3.1.1 Definition of Requirements Management.....	30
3.1.2 Requirements Management Process Models.....	31
3.1.3 Why Requirements Change on a Project.....	46
3.1.4 Requirement Stability Using TBX Management.....	50
3.1.5 Requirements Management for Product Lines.....	53
3.1.6 Ensuring Requirements Quality through Requirements Management.....	54
3.1.7 Data Attributes for Requirements Management.....	56
3.1.8 Conclusion on Current Approaches to Requirements Management.....	56
3.2 Newer Requirements Management Trends.....	57
3.2.1 Trends Based on the INCOSE Requirements Working Group Efforts.....	57
3.2.2 Trends Based on Author Published Research Efforts.....	60
3.2.3 Trends in Various Industries.....	61

3.2.4	Requirements Management (RM) Tools.....	64
3.2.5	Observations from Requirements Engineering Practitioners	80
3.2.6	Requirement Trends Conclusions	94
Chapter 4:	CHALLENGES ASSOCIATED WITH SPACE SYSTEMS	96
4.1	Overview of the Space Industry.....	96
4.1.1	Complexity in Space Systems.....	96
4.1.2	Definition of a Successful Space System Project	103
4.1.3	Space System Cost Calculations as a Function of System Requirements.....	105
4.2	Space Project Example 1: MAVEN.....	108
4.2.1	MAVEN Overview	108
4.2.2	MAVEN Requirements Approach	109
4.2.3	MAVEN Project Outcome	111
4.2.4	Assessment of MAVEN Requirements Approach.....	113
4.3	Space Project Example 2: Mars Science Laboratory, MSL (Mars Curiosity Rover).....	114
4.3.1	MSL Overview.....	114
4.3.2	MSL Requirements Approach	116
4.3.3	MSL Outcome.....	117
4.3.4	Assessment of MSL Requirements Approach	120
4.4	Space Project Example 3: GOES-R Weather Satellites.....	122
4.4.1	GOES-R Series Overview.....	122
4.4.2	GOES-R Requirements Approach	125
4.4.3	GOES-R Outcome.....	127
4.4.4	Assessment of GOES-R Requirements Approach	130
4.5	Space Project Example 4: NASA Constellation	131
4.5.1	Constellation Overview.....	131
4.5.2	Constellation Requirements Approach	134
4.5.3	Constellation Project Outcome	137
4.5.4	Assessment of Constellation Requirements Approach	141
4.6	Space Project Example5: NASA Artemis Human Landing System (HLS).....	146
4.6.1	Artemis HLS Overview	146
4.6.2	Artemis HLS Requirements Approach	151
4.6.3	Artemis HLS Outcome.....	152
4.6.4	Assessment of Artemis HLS Requirements Approach	153

4.7	Assessment of Requirements Management Approach of the Space Example Projects	153
Chapter 5:	RECOMMENDED REQUIREMENTS MANAGEMENT PROCESS UPDATES.....	156
5.1	Factors Used to Evaluate Optimized Requirements Management Approaches	156
5.1.1	Cost to Apply Process Updates	156
5.1.2	Duration and Labor Cost of Requirements Management Processes	157
5.1.3	Direct Cost of Requirements Management Processes	158
5.1.4	Cost Associated with Schedule Delay.....	158
5.1.5	Cost Associated with Number of Requirements	159
5.2	Proposed Process Updates for Requirements Management.....	159
5.3	Process 1: Data Focused Requirements Management Approach.....	161
5.3.1	Document Centric Requirements Management Approach.....	161
5.3.2	Current Trend Towards Model Based Systems Engineering	163
5.3.3	Current Trend Towards Information Based Requirements Management	164
5.3.4	Usage of a Data Centric Requirements Management Approach.....	165
5.3.5	The Benefit of Requirement Reuse	166
5.3.6	Cost Optimization using a Data Focused Requirements Management Approach	169
5.3.7	Data Focused Requirements Management Design Pattern	169
5.4	Process 2: Use of a Collaborative Requirements Management Tool.....	170
5.4.1	Rationale for Requirement Collaboration	171
5.4.2	Options for Collaboration During Requirements Management	172
5.4.3	Cost Optimization with a Collaborative Requirements Management Tool	173
5.4.4	Collaborative Requirements Management Tool Usage Design Pattern.....	174
5.5	Process 3: Minimize and Consolidate the Requirements	175
5.5.1	Achieving Balance in Requirement Quantity	176
5.5.2	Requirement Document Quantity as a Function of Cost.....	178
5.5.3	Assessment to Ensure Minimized Set of Good Quality Requirements.....	179
5.5.4	Cost Optimization with Minimization and Consolidation of Requirements Approach	180
5.5.5	Minimize and Consolidate Requirements Design Pattern.....	180
5.6	Process 4: Requirement Stability and Enforcement.....	181
5.6.1	Requirement Volatility Impact in Cost Calculations	182
5.6.2	Requirement Instability Assessment.....	183
5.6.3	Impact of Enforcement of Requirements with High Instability	184
5.6.4	Cost Optimization with Levying Requirements vs. Maturing Requirements Approach... 185	

5.6.5	Requirement Stability and Enforcement Design Pattern.....	187
5.7	Impact to the Requirements Management Model	188
Chapter 6:	DEVELOPMENT OF AN OPTIMIZED REQUIREMENTS MANAGEMENT EXECUTABLE MODEL	190
6.1	Generation of a Requirements Management Executable Model.....	190
6.2	Generation of an Optimized Requirements Management Executable Model.....	192
6.2.1	Executable Model Elements.....	192
6.2.2	Process 1: Data Centric Requirements Management Optimization Assessment	194
6.2.3	Process 2: Collaborative Tool Usage Optimization Assessment	201
6.2.4	Process 3: Consolidation and Minimizing of Requirements Optimization Assessment ...	206
6.2.5	Process 4: Requirement Stability and Enforcement Optimization Assessment	215
6.2.6	Requirements Management Process Model	229
6.2.7	Model Development Concluding Remarks	236
Chapter 7:	REQUIREMENTS MANAGEMENT MODEL OPTIMIZATION CASE STUDIES.....	237
7.1	Application of Requirements Management Model	237
7.1.1	Inputs For Simulation.....	237
7.1.2	Case Study Simulation Results	238
7.2	Discussion of Case Study Results	243
Chapter 8:	RECOMMENDATIONS AND CONCLUSIONS	246
8.1	Approach to Requirements Management Cost Optimization	246
8.2	Recommendations for a Requirements Management Approach.....	247
8.3	Suggestions for Additional Research.....	249
8.4	Conclusions.....	249
REFERENCES	250
Appendix A - SYSML MODEL DIAGRAMS.....		257
A1. Model Elements		257
A2. Block Definition Diagrams		257
A2. Parametric Diagrams.....		259
A3. Activity Diagrams		260
A4. Instance Blocks		270
A5. Simulation Diagrams.....		275
Appendix B - DATA TABLES.....		279
B1. Case Study Data Tables.....		280

B2. Variation of Change Per Month on Requirement Stability Enforcement..... 282

LIST OF TABLES

Table 1. Key Requirements Engineering Definitions. (IEEE 29148, 2018)	9
Table 2. Requirement Categories. (Pohl, 2010).....	17
Table 3. Characteristics of a Well-Formed Set of Requirements. (INCOSE, 2019).....	24
Table 4. Example Requirement Traceability Matrix.....	25
Table 5. Published Definitions for Requirements Management.	30
Table 6. Sample TBX Table.	52
Table 7. Characteristics of Quality Requirements. (INCOSE, 2019)	55
Table 8. Sample Attributes of Requirements Statements. (INCOSE, 2019).....	56
Table 9. Summary of Practitioner Recommendation Themes.	94
Table 10. Comparisons of Various Industries. (Wertz, Everett, & Puschell, 2011)	99
Table 11. MAVEN Project Outcome	112
Table 12. Maven Calculated Labor Costs from COSYSMO.....	113
Table 13. MSL Project Outcome	120
Table 14. MSL Calculated Labor Costs from COSYSMO.....	120
Table 15. GOES-R Project Outcome	129
Table 16. GOES-R Calculated Labor Costs from COSYSMO.....	130
Table 17. Assessments of the Constellation Program.....	139
Table 18. Constellation Project Outcome	140
Table 19. Total Constellation Level II Requirement Count.....	140
Table 20. Constellation Calculated Labor Costs from COSYSMO.....	141
Table 21. Total HLS Requirement Count	152
Table 22. Artemis HLS Calculated Labor Costs from COSYSMO.....	153
Table 23. Summary of Space Project Calculated Labor Costs from COSYSMO.	154
Table 24. Proposed Process Activity Updates for Requirements Management.....	160
Table 25. Document Centric Requirements Management Task Values and Rationale.	200
Table 26. Data Centric Requirements Management Task Values and Rationale.	200
Table 27. Non-collaborative Tool Usage Task Values and Rationale.	204
Table 28. Collaborative Tool Usage Task Values and Rationale.	205
Table 29. Consolidation of Requirements Simulation Results.	211
Table 30. Requirement Costs for Ten Products, Consolidated vs. Non-Consolidated Approaches.	212
Table 31. Return on Investment for Consolidating and Minimizing 200 Requirements.	212
Table 32. Non-consolidated Approach Task Values and Rationale.....	214
Table 33. Consolidated Approach Task Values and Rationale.....	214
Table 34. Unstable Requirement Enforcement Task Values and Rationale.	227
Table 35. Stability Before Enforcement Task Values and Rationale.....	228
Table 36. Project Inputs for Requirements Management Model.	237
Table 37. Case Study Inputs for Requirements Management Model.	238
Table 38. Case Study Project Cost Savings Results.	241
Table 39. Impact of Change Count in Post-Requirement Development Phase on Duration Time.	243
Table 40. Summary of Space Project Calculated Labor Costs from COSYSMO.	243

Table 41. Summary of Constellation Task Durations (Case Study 1).	245
Table 42. Recommended Requirements Management Approach to Achieve Optimization	248
Table 43. Case Study1 Results from Requirements Management Model (Instability Ratio = 0.25, Change Cost = \$75K).....	280
Table 44. Case Study2 Results from Requirements Management Model (Instability Ratio = 0.5, Change Cost = \$75K).....	280
Table 45. Case Study3 Results from Requirements Management Model (Instability Ratio = 0.25, Change Cost = \$150K).....	281

LIST OF FIGURES

Figure 1. Overview of the Optimization of the Requirements Management Process Model.	2
Figure 2. Dissertation Content Roadmap.	4
Figure 3. Overview of a Typical Project Life Cycle. (Forsberg, Mooz, & Cotterman, 2005).....	5
Figure 4. Various Applications of the Project Life Cycle. (Forsberg, Mooz, & Cotterman, 2005).....	6
Figure 5. The Systems Engineering Vee Model Showing the Path to System Validation. (Forsberg, Mooz, & Cotterman, 2005)	7
Figure 6. Recursive Nature of Transforming System Needs to Realized System. (IEEE 29148, 2018)	7
Figure 7. Overview of the Various Requirements Engineering Processes. (Pohl, 2010).....	9
Figure 8. Systems Engineering Sandwich showing Interrelation of Requirements and Models. (Dick, Hull, & Jackson, 2017)	13
Figure 9. Requirements Development Process Transitioning the Problem Domain to the Solution Domain. (Dick, Hull, & Jackson, 2017)	14
Figure 10. Requirements Engineering Process Showing Transformation of Input to Derived Requirements. (Dick, Hull, & Jackson, 2017)	15
Figure 11. Transformation of Concepts Into Needs and then to Requirements. (INCOSE, 2019).....	16
Figure 12. Business Operational Level Transformation of Concepts Into Needs and Requirements for multiple Products and Services. (INCOSE, 2021 draft)	16
Figure 13. Requirement Evolution over the Systems Engineering Vee Model. (INCOSE, 2019)	18
Figure 14. Requirements Development and the Change Feedback Loop. (Dick, Hull, & Jackson, 2017). ..	20
Figure 15. Example of Requirement Development Work with Multiple Development Organizations.	22
Figure 16. Example of Requirement Development Work with a Single Development Organization.	22
Figure 17. Various Examples of Product Breakdown Structures. (Forsberg, Mooz, & Cotterman, 2005) ..	23
Figure 18. Example of a Product Requirement Set and Specification Tree.....	24
Figure 19. Failure Types and Occurrence Based on Poor Requirements Management. (Engineering.com, 2018)	28
Figure 20. The IEEE 29148 Requirements Management Process Model.....	32
Figure 21. The NASA Requirements Management Process. (NASA, 2020)	33
Figure 22. The INCOSE Requirements Definition Process Model. (INCOSE, 2015)	34
Figure 23. The INCOSE Requirements Management Process Model.....	34
Figure 24. The INCOSE Product Realization Process. (INCOSE, 2021 draft)	35
Figure 25. INCOSE Product Realization Process with Concurrent Requirements Management Effort....	35
Figure 26. The INCOSE Information-Based Realization Process. (Wheatcraft, Ryan, Dick, & Llorens, 2019)	36
Figure 27. The PMI Requirements Process Model. (PMI, 2016)	37
Figure 28. Mapping of the Requirements Process to the Project Management Process. (PMI, 2016)	37
Figure 29. The PMI Requirements Monitoring and Controlling Process. (PMI, 2016)	38
Figure 30. The Dick Requirements Management Process Model.....	39
Figure 31. The Hooks Requirements Process Model. (Hooks & Farris, 2001)	40
Figure 32. The Hooks Requirements Development and Management Process Model.....	41
Figure 33. The Kumar Requirements Management Process Model. (Kumar, 2004).....	42

Figure 34. The Hood Requirements Management Interfaces. (Hood, Wiedemann, Fichtinger, & Pautz, 2008)	42
Figure 35. The Hood Requirements Processes. (Hood, Wiedemann, Fichtinger, & Pautz, 2008)	43
Figure 36. The Current State Requirements Management Process Model.	45
Figure 37. Evolution of Requirements Baselines Addressed by Requirements Management. (Forsberg, Mooz, & Cotterman, 2005)	46
Figure 38. Requirements Management Evolution Over the Project Life Cycle. (Forsberg, Mooz, & Cotterman, 2005)	48
Figure 39. Requirement Changes Over the Project Life Cycle. (Forsberg, Mooz, & Cotterman, 2005)...	48
Figure 40. Impact of Requirement Changes with Multiple Development Organizations.....	50
Figure 41. TBXs Represent Liens that Impact Projects if not Resolved. (Forsberg, Mooz, & Cotterman, 2005)	51
Figure 42. Example of a Product Line Specification Tree showing Inter-relationships among the Product Requirements.	53
Figure 43. Overview of Information-based Requirement Development and Management. (Wheatcraft, Ryan, Dick, & Llorens, 2019).....	58
Figure 44. Examples of Requirements in Microsoft Products. (Klariti, 2020)	66
Figure 45. Requirement Database in IBM Rational DOORS. (IBM, 2013)	67
Figure 46. Requirement Model in IBM Rational DOORS. (IBM, 2013)	67
Figure 47. DOORS Linking Between Requirement Modules. (IBM, 2013)	68
Figure 48. DOORS Requirement Attributes. (Makinen, 2013)	69
Figure 49. Requirement Database in IBM DOORS Next. (IBM, 2014).....	71
Figure 50. DOORS Next Traceability Between Requirement Artifacts. (IBM, 2014).....	72
Figure 51. DOORS Next Requirement Attributes. (IBM, 2014)	72
Figure 52. Requirement Database in Jama Connect. (Jama Software, 2020)	73
Figure 53. Jama Traceability Between Requirement Items. (Jama Software, 2020)	74
Figure 54. Jama Requirement Attributes. (Jama Software, 2020)	74
Figure 55. Requirement Diagram / Table in SysML (Enterprise Architect). (Baker, 2020).....	76
Figure 56. Requirement Diagram / Table in SysML (Cameo Systems Modeler). (No Magic, Inc./Dassault Systems, 2020).....	77
Figure 57. Seilevel Requirement Management Tool Rankings. (Seilevel, 2016).....	79
Figure 58. Engineered Systems Have Seen Exponential Growth in Complexity Over the Last 50 Years. (Guo, 2018).....	97
Figure 59. NASA Product Structure Example. (NASA, 2007).....	97
Figure 60. Wide Range of Space Mission Applications. (Wertz, Everett, & Puschell, 2011).....	100
Figure 61. 2008 Global Space Spending by Major Category. (Space Foundation, 2019)	100
Figure 62. Quadruple Constraints For Project Success. (Pinto, 2016).....	104
Figure 63. COSYSMO 2.0 Cost Model. (Valerdi, 2010).....	106
Figure 64. The MAVEN Spacecraft. (NASA, 2013)	108
Figure 65. The MAVEN Spacecraft Prior to Launch. (NASA, 2013)	109
Figure 66. The MAVEN System Product Structure.....	109
Figure 67. The MAVEN Requirement Tree. (NASA, 2011).....	110
Figure 68. Example MAVEN MRD Requirement. (NASA, 2012)	111
Figure 69. MAVEN Budget and Schedule Overview. (GAO, 2013).....	112

Figure 70. The Curiosity Rover in Operation (Self Portrait). (NASA, 2020).....	114
Figure 71. Curiosity Rover During Integration and Test Operations. (NASA, 2020)	115
Figure 72. The Curiosity Rover Science Instruments. (NASA, 2020).....	116
Figure 73. The Overall MSL System Product Structure	116
Figure 74. MSL Critical Task Delays. (GAO, 2011).....	119
Figure 75. MSL Budget and Schedule Overview. (GAO, 2011)	119
Figure 76. GOES-R Spacecraft. (eoPortal, 2020)	123
Figure 77. GOES-R Spacecraft Integration. (eoPortal, 2020)	124
Figure 78. The GOES-R Series System Product Structure.	125
Figure 79. The GOES-R Requirements Tree. (NOAA/NASA, 2020).....	126
Figure 80. Example GOES-R MRD Requirements. (NASA, 2012).....	127
Figure 81. Changes to the GOES-R Program Over Time. (GAO, 2014).....	128
Figure 82. The Constellation Program Elements. (Connolly, 2006).....	132
Figure 83. The Constellation System Product Structure.	133
Figure 84. Constellation Top Level Documentation. (NASA, 2010)	134
Figure 85. Constellation Requirements Development and Refinement Process. (NASA, 2010)	135
Figure 86. Constellation Architecture Requirement Examples. (NASA, 2008)	136
Figure 87. The Constellation High Level Requirement Tree. (NASA, 2010)	137
Figure 88. Constellation Budget Profiles. (NASA, 2011)	138
Figure 89. Schedule Delays on Constellation Over First Four Years. (NASA, 2011).....	138
Figure 90. Constellation Required use of a Design and Construction Standard. (NASA, 2008).....	144
Figure 91. Example Design and Construction Requirement. (NASA, 2016)	144
Figure 92. The Artemis Program Mission Overview. (NASA, 2019)	147
Figure 93. The Various HLS Concepts. (Leman, 2020)	150
Figure 94. The HLS Product Structure.	150
Figure 95. The HLS High Level Requirement Tree.	152
Figure 96. Project Complexity and Need for Optimized Processes.....	155
Figure 97. The High Costs of Schedule Delays. (Forsberg, Mooz, & Cotterman, 2005)	158
Figure 98. RM Model Activities Associated with "Data Focused Requirements Management Approach".	162
Figure 99. Document Centric Requirements Management Process Flow.	162
Figure 100. Document Centric Requirements Management Artifacts.....	163
Figure 101. Data Centric Requirements Management Process Flow.....	165
Figure 102. Data Centric Requirements Management Artifacts.....	166
Figure 103. Example of Requirement Reuse.	167
Figure 104. Example of a Reused Requirement in Jama Connect. (Jama Software, 2020).....	168
Figure 105. Document Focused Requirements Management Design Pattern (1a).	170
Figure 106. Data Focused Requirements Management Design Pattern (1b).	170
Figure 107. RM Model Activities Associated with "Use of a Collaboration Tool".	171
Figure 108. Collaboration Activities Outside of Requirements Management Tool.	172
Figure 109. Collaboration Options in Jama Connect. (Jama Software, 2020).....	173
Figure 110. Collaboration Options in IBM DOORS Next. (IBM, 2014).....	173
Figure 111. Non-Collaborative Requirements Management Tool Design Pattern (2a).....	175
Figure 112. Collaborative Requirements Management Tool Design Pattern (2b).....	175

Figure 113. RM Model Activities Associated with "Minimization and Consolidation of Requirements".	176
Figure 114. Supplier Effort Associated with Multiple Specification and Standards.	178
Figure 115. Project Team Effort to Compile Comprehensive Supplier Specification.	179
Figure 116. Non-Consolidated Requirements Design Pattern (3a).	181
Figure 117. Minimized and Consolidated Set of Requirements Design Pattern (3b).	181
Figure 118. RM Model Activities Associated with "Stability and Enforcement".	182
Figure 119. Impact of Requirements Volatility on Project Cost (Pena & Valerdi, 2014).	182
Figure 120. Decision Point of When to Levy Requirements for Suppliers.	186
Figure 121. Levy Unstable Requirements Design Pattern (4a).	187
Figure 122. Optimization Assessment of Levying Unstable Requirements Design Pattern (4b).	187
Figure 123. Requirements Management Process Model With Optimized Process Areas Highlighted.	188
Figure 124. Example of a Cameo Model. (Borky, 2019).	191
Figure 125. Example of a SysML Activity Diagram. (Borky, 2019).	191
Figure 126. Overview of the Requirements Management Components in the Cameo Model.	193
Figure 127. Requirement Management Process Activities (Current State and Optimized).	194
Figure 128. Document and Data Centric Requirements Management SysML Models.	195
Figure 129. Document and Data Centric Cameo Simulation Toolkit Simulation Configurations.	196
Figure 130. Process 1a Simulation Screen Capture.	197
Figure 131. Process 1a Simulation Duration Timeline Data (Minimum Process Duration).	197
Figure 132. Process 1a and 1b Simulation Duration Data Tables (Minimum Process Durations).	198
Figure 133. Data Centric Process Optimization Assessment Results.	198
Figure 134. Non-Collaborative and Collaborative RM Tool Usage SysML Models.	202
Figure 135. Collaborative RM Tool Usage Optimization Assessment Results.	203
Figure 136. Configuration of COSYSMO for Input Values.	207
Figure 137. Impacts of Changing the Consolidation of Requirements in COSYSMO.	207
Figure 138. Variation of Requirement Quantity from COSYSMO (low number of requirements).	208
Figure 139. Variation of Requirement Quantity from COSYSMO (large number of requirements).	208
Figure 140. Non-Consolidation and Consolidation of Requirements SysML Models.	209
Figure 141. Consolidation of Requirements Optimization Assessment Results.	211
Figure 142. Instability and Change Count Calculation Parametric SysML Model.	215
Figure 143. Total Change Cost Calculation Parametric SysML Model.	217
Figure 144. Change Cost Optimization Excel Model.	218
Figure 145. Instability and Stability before Enforcement of Requirements SysML Models.	219
Figure 146. Labor Comparisons for Levying Unstable vs. Stable Requirements for Varying Change and Delay Costs.	221
Figure 147. Cost Comparisons for Levying Unstable vs. Stable Requirements for Varying Change and Delay Costs.	222
Figure 148. Levy Wait Time with Variation of Instability Ratio.	223
Figure 149. Impact of Instability Ratio at \$10K/change.	225
Figure 150. Impact of Instability Ratio at \$100K/change.	226
Figure 151. Overall Requirements Management Process Model Converted to a SysML Model.	230
Figure 152. Activity Diagram Showing the Optimized Path Option.	231
Figure 153. Activity Diagram Showing Opaque Action Calculations.	232

Figure 154. Monitor and Change Requirements Process Options.	233
Figure 155. Sample Process Not Contributing to Optimization Assessment.....	234
Figure 156. COSYSMO Integrated in an Activity Diagram Simulation.	235
Figure 157. Simulation Results of COSYSMO Integrated in an Activity Diagram Simulation.	235
Figure 158. Example of the Requirements Management Process Activity Diagram Simulation.	238
Figure 159. MAVEN Case Study 1 Simulation Timeline Results Example.....	239
Figure 160. Example of the Requirements Management Process Simulation Data Table.	240
Figure 161. Case Study 1 Simulation Results Example.....	240
Figure 162. Space Project Optimized Requirements Management Process Benefits.	241
Figure 163. Project Cost Savings using Optimized Processes (Case Study 2).	242
Figure 164. Requirement Model Contents.....	257
Figure 165. Requirement Model Overall Block Diagram.....	258
Figure 166. Requirement Processes Block Diagram.....	259
Figure 167. Instability Calculation Parametric Diagram.	259
Figure 168. Total Change Cost Calculation Parametric Diagram.....	260
Figure 169. Process 1a Document Centric Activity Diagram.....	260
Figure 170. Process 1b Data Centric Activity Diagram.....	260
Figure 171. Process 2a Non-Collaborative Tool Usage Activity Diagram.....	261
Figure 172. Process 2b Collaborative Tool Usage Activity Diagram.....	261
Figure 173. Process 3a Non-Consolidated Requirements Activity Diagram.....	261
Figure 174. Process 3b Minimize and Consolidate Requirements Activity Diagram.....	261
Figure 175. Process 4a Requirement Instability Before Levy Activity Diagram.	262
Figure 176. Process 4b Requirement Stability Before Levy Activity Diagram.....	262
Figure 177. Overall Requirement Management Activity Diagram.....	263
Figure 178. Obtain Inputs Activity Diagram.	263
Figure 179. Generate Requirements Activity Diagram.....	264
Figure 180. Assess Requirement Quality Activity Diagram.....	265
Figure 181. Distribute Requirements Activity Diagram.	265
Figure 182. COSYSMO Reader Activity Diagram.	266
Figure 183. Plan Verification Activity Diagram.....	266
Figure 184. Monitor and Change Requirements Activity Diagram.....	267
Figure 185. Evaluate Changes Activity Diagram.	268
Figure 186. Change Requirements Activity Diagram.....	268
Figure 187. Review and Baseline Requirements Activity Diagram.	269
Figure 188. Measure and Report Requirement Metrics Activity Diagram.	269
Figure 189. Capture Verification and Validation Activity Diagram.....	270
Figure 190. Process 1 Instance Blocks.....	270
Figure 191. Process 2 Instance Blocks.....	271
Figure 192. Process 3 Instance Blocks.....	271
Figure 193. Process 4 Instance Blocks.....	271
Figure 194. Space Project MSL / Maven Instance Blocks.....	272
Figure 195. Space Project Constellation / HLS Instance Blocks.	273
Figure 196. Space Project GOES Instance Block.	273
Figure 197. Overall Requirement Management General Simulation Configuration.	274

Figure 198. MAVEN and MSL Simulation Configurations.	275
Figure 199. GOES, Constellation and HLS Simulation Configurations.....	276
Figure 200. Processes 1 and 2 Simulation Configurations.	277
Figure 201. Processes 3 and 4 Simulation Configurations.	278
Figure 202. Change Cost Sensitivity Evaluations.....	282

CHAPTER 1: INTRODUCTION

1.1 The Challenge with Requirements

Companies continue to evolve their products to remain competitive in the market place. As technology is improved to add more capabilities, product development becomes more complex, leading to a needed evolution of the development processes that ensure the products meet an ultimate set of defined needs. The processes of how a product will be designed, produced, and considered ready for usage is often done through a transformation process, where the product's objectives and needs are transformed to a set of requirements that define what the product will be able to accomplish (including the set of capabilities that it must perform). For complex systems, particularly those with software intensive features, requirements can take a significant amount of effort to define, communicate, and manage; leading to a significant effort by the product development team. It is observed there are minimally documented best practices for how to address management of a large set of requirements (this is shown in Chapter 3). Multiple sources can be found describing how to develop high quality requirements, but how can these requirements be managed to ensure they are organized, communicated to sub-tier product developers, evolved as new data is learned, and captured in a manner that they are easy to find?

When organizations have opted to spend **minimal** amount of effort in requirements development and management, studies have shown that they ultimately pay for this by cost overruns and resulting products which do not perform as expected (or worse, have unintended consequences resulting in accidents); this is highlighted later in Section 2.2.5. When organizations have opted to spend a **large** amount of effort in requirements development and management, they have also seen cost overruns, delayed schedule, and an over-constrained design. This then leads to the question, what is the **optimal** amount of effort to be spent in defining a product's requirements and addressing the processes of requirements communication, change control, and overall management? Some studies have found that spending 8% to 14% of total program costs on the requirements processes early on the project could result in significant reduction in project cost overruns, showing improvement from 140% overrun to 40%

(Gruhl, 1992). However, what does this effort actually involve? Are there processes that can be applied to ensure a requirements-driven engineering effort can be implemented which meets the objectives of the product needs, the constraints of the developer's budget and schedule, and ensure it is tailored to an optimized approach for the product being developed? Optimization recognizes that there are relationships and trade-offs between characteristics. For project success, an optimized balance of cost, performance, schedule and customer satisfaction is achieved (discussed in Section 4.1.2). To optimize for cost while still achieving project success, the other parameters are still obtained, but to a lesser degree based on how they are traded against associated expenses.

This dissertation looks into the specifics related to the development of a space system, a very complex type of product, and how requirements management processes have traditionally been applied. Current approaches are captured through literature searches, interviews with experienced practitioners, reviews of newer trends in the practice, and through research of space systems developed by NASA. A recommended process model for requirements management is proposed that would enable cost optimization of requirements management on a project when compared to a current state approach (Figure 1 highlights the two models, which are described more fully later within this dissertation).

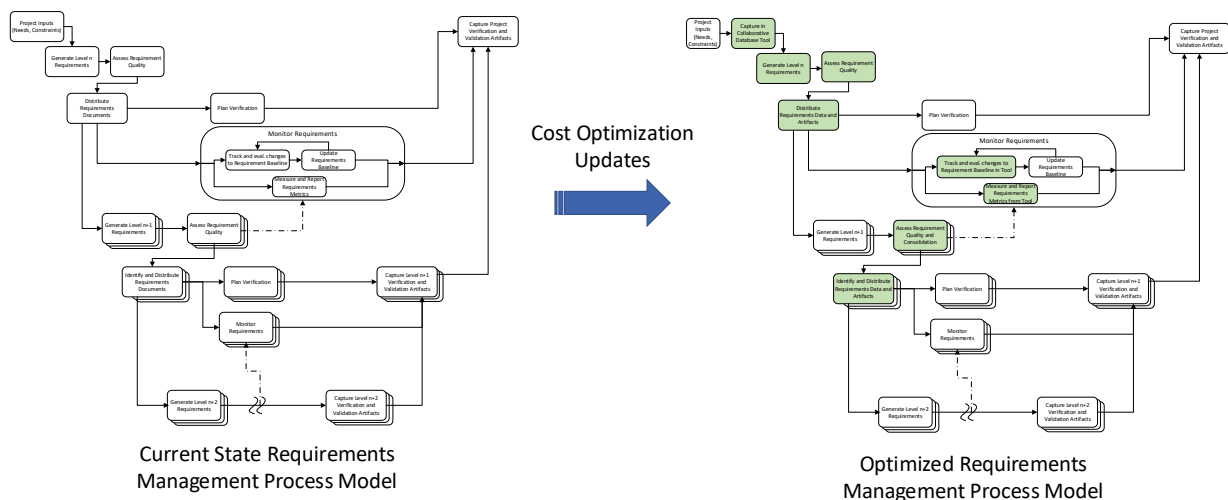


Figure 1. Overview of the Optimization of the Requirements Management Process Model.

Requirements management requires a well-thought out approach by a product team to ensure it spends just enough effort (not too little, not too much), and recommendations for how to achieve this are

provided through an optimized process model that developers could use fully, or tailor based on their project complexity, to enable affordable and timely realized complex systems. The focus of this research specifically looks at the application of the process on space systems, which are shown in Section 4 to contain a high degree of complexity in their development, however the concepts could be applied to other systems as shown in Section 8.

1.2 Dissertation Organization

This dissertation presents a proposed optimized method to manage requirements on space systems. To address why this is a challenge and give context to the topic, the content of this paper is organized as follows:

- Chapter 1 presents the objectives of this dissertation.
- Chapter 2 presents the systems engineering process over a project life cycle and introduces concepts related to requirements engineering.
- Chapter 3 defines requirements management and presents historical approaches, overviews of management tools, newer trends for requirements management, interviews with practitioners of requirements management across a broad range of industries, and presents the **current state requirements management process model**.
- Chapter 4 presents why the space industry has specific challenges in comparison to other industries, how requirements management contributes towards project success and costs, and gives examples of recent space projects and their approaches towards requirements management.
- Chapter 5 uses the challenges seen by the space projects to present four requirements management process areas with proposed updates to enable cost optimization, and describes how these contribute to an overall **optimized requirements management process model**.
- Chapter 6 presents the efforts to generate an executable model of the four processes using model based systems engineering (MBSE) system modeling language (SysML), presenting quantitative simulation results of these processes before and after applying the proposed updates. This chapter also presents an executable model for the overall optimized requirements management model, applying the four processes with options to use the current state or optimized approaches.
- Chapter 7 demonstrates application of the executable requirements management model using the space project data from Chapter 4 to assess if the proposed cost optimization effort is realized; this effort additionally provides application examples of using the requirements management model.

- Chapter 8 presents a summary of the overall effort, recommendations on using the findings from this research, and proposed research into additional topics.

The entire flow of the dissertation is shown graphically in Figure 2, highlighting how the research into current trends and case studies leads into a set of proposed improvements to the requirements management process.

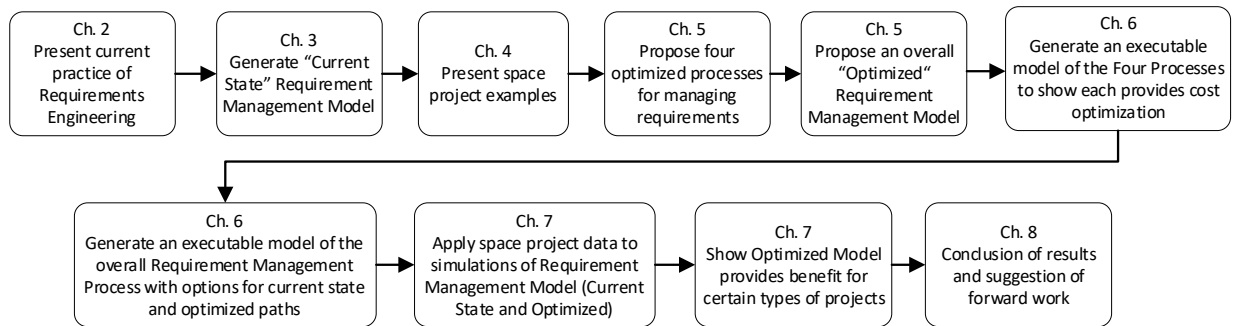


Figure 2. Dissertation Content Roadmap.

This dissertation presents considerations for adopting the process improvements to assess if an optimization is maintained for the project (investment in making process changes compared to anticipated benefits). To provide validation of the proposed approach, Chapters 6 and 7 present executable modeling techniques with simulation results for both specific process activities and for the entire requirements management process with data from the space industry case studies described in Chapter 4. The case study data is taken from real world examples; however, the requirements management model and process is able to be tailored with customized inputs to allow projects to use the model and see if the proposed updates would result in benefits to their efforts (described further in Section 8).

2.1 The Systems Engineering Process and the Project Life Cycle

Often an organization will follow an orderly sequence of activities, performed in phases, to implement their project plans. Each series of steps is followed by a review of the accomplishments and outstanding liens, and a decision point on whether to proceed to the next phase of development. This is referred to as the project life cycle, one of the fundamental principles in performing project management (Figure 3).

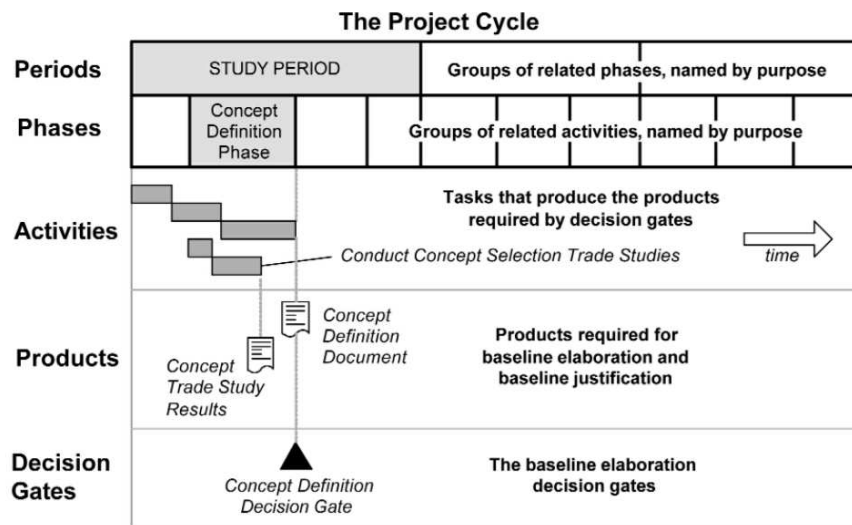


Figure 3. Overview of a Typical Project Life Cycle. (Forsberg, Mooz, & Cotterman, 2005)

An example of how this is implemented is shown in Figure 4, which takes the project cycle and expands to approaches by NASA, Department of Defense, ISO 15288, and other types of projects. This project life cycle model forms the foundation of how projects are implemented, which then addresses the various stages of product development and associated technical processes.

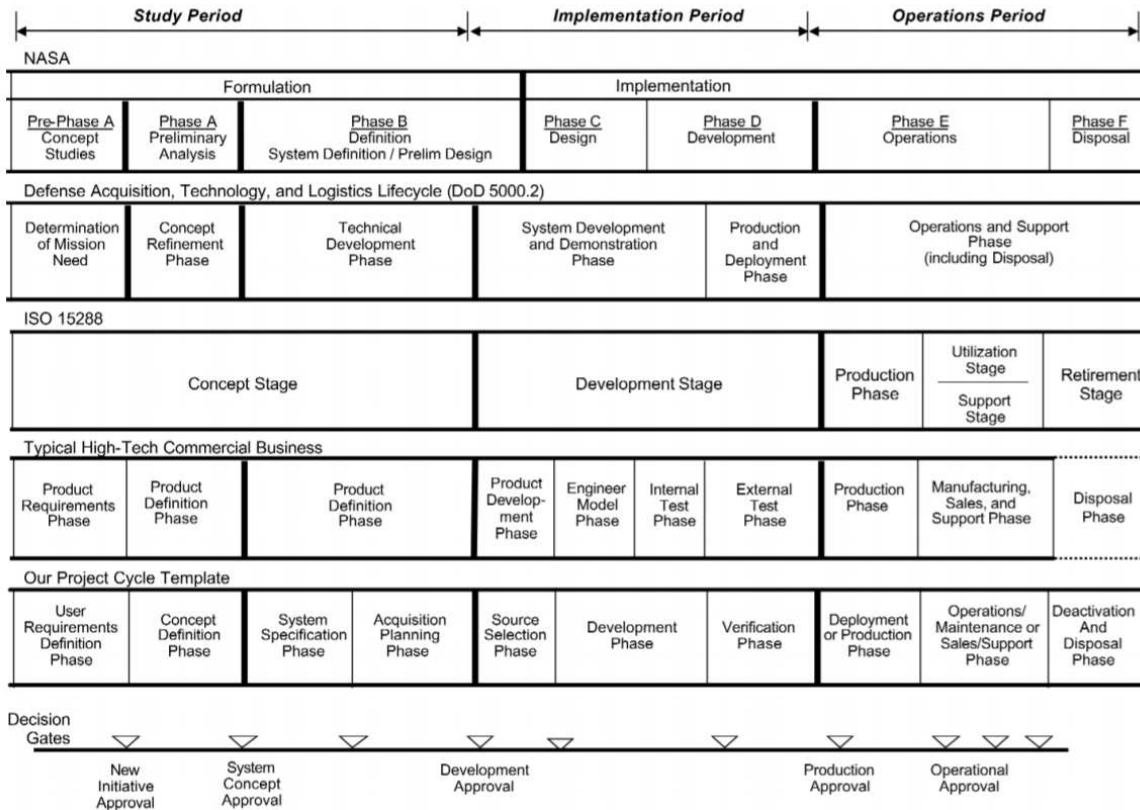


Figure 4. Various Applications of the Project Life Cycle. (Forsberg, Mooz, & Cotterman, 2005)

The various technical processes performed during these project life cycle phases is referred to as the "Systems Engineering" processes. These systems engineering processes include specific technical processes such as development of project requirements and certification of the final product, technical management processes, and the project adjudication and approval processes (IEEE 15288, 2015).

Systems engineering is a methodical, disciplined approach for the design, realization, technical management, operations, and retirement of a system (NASA, 2007). Each stage of a project life cycle addresses unique challenges, many of which are addressed in the processes defined in IEEE 15288, *Systems and Software Engineering - System Life Cycle Processes*, and some are even further defined in the *International Council of Systems Engineering (INCOSE) Systems Engineering Handbook*.

A few different graphical models have been created to show the various system engineering efforts conducted over a project life cycle. One is the Vee model (Figure 5), which presents the product development maturation from left to right, and a view of the different levels of abstraction within the

product (system through component) along the vertical. In this model, the stakeholder needs for the product are established at the system level, decomposed to lower levels of abstraction, and ultimately the system elements are integrated and verified until the entire system is shown to meet the needs.

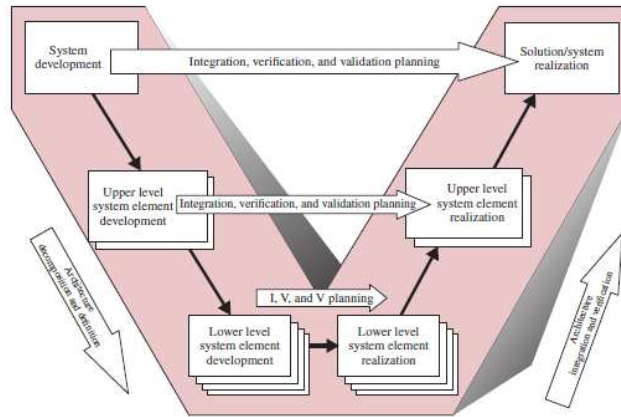


Figure 5. The Systems Engineering Vee Model Showing the Path to System Validation. (Forsberg, Mooz, & Cotterman, 2005)

In actuality, the nature of development over a project life cycle is frequently iterative, where the system is developed in progressive waves of maturity which applies feedback from prior activities (build a little, test a little, capture data, apply updates, and so on). This can happen at the system level and recursively through the lower levels (going down the left side of the Vee), showing the technical processes are occurring frequently (and concurrently) during the life cycle among the many levels of the product. This is highlighted in the model shown in Figure 6, taken from IEEE 29147.

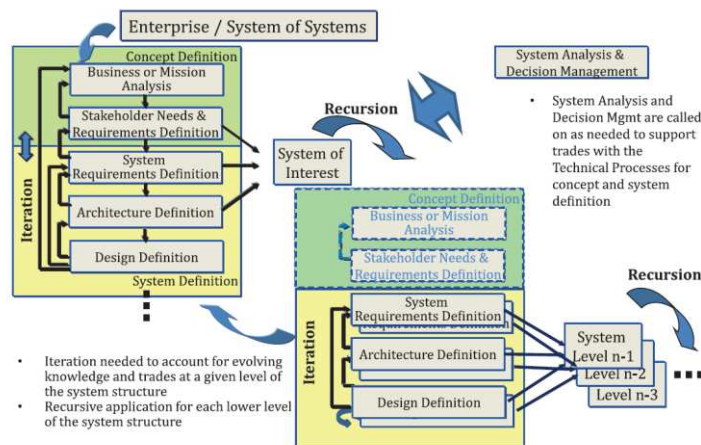


Figure 6. Recursive Nature of Transforming System Needs to Realized System. (IEEE 29148, 2018)

With a trend towards rapid technology growth, traditional systems engineering approaches appear less able to support rapid product development, particularly in the space industry. The rate of change of technology, the innovation of new contracting approaches and use of commercial companies, and the rush to market for new space systems have all challenged the established systems engineering approaches, resulting in traditional space companies changing their processes and infrastructure. Current industry trends have led to the following observations:

- Transforming customer *needs* to product *requirements* is an iterative process, requiring some knowledge of the design and analysis of options.
- Suppliers often need to be put on contract early to begin their development efforts, bringing a need for defining their requirements early in the program life cycle before the needs have been fully transformed to requirements.
- The resources required to address thousands of requirements can be substantial, and not always affordable for a system provider.
- The need to be affordable and fast are a reality with changing technology and competitive market for modern space systems.

To address rapid technology development on complex systems, there is need for a set of systems engineering processes which enable development of an integrated system, allowing for both innovation and optimized costs. A further look at one of the systems engineering technical processes, *requirements management*, a subset of an overall process call *requirements engineering*, has been done to see if there is opportunity for optimization. A review of the requirements engineering processes are provided beforehand to enable context for the subsequent discussion of requirements management.

2.2 Overview of Requirements Engineering

2.2.1 Terminology and Authoritative Publications

Requirements engineering is a discipline that combines the approach to develop, validate, and manage requirements on a project. To understand the application of requirements engineering, associated terminology is defined in Table 1.

Table 1. Key Requirements Engineering Definitions. (IEEE 29148, 2018)

Term	Definition
Requirement	Statement which translates or expresses a need and its associated constraints and conditions; Requirements exist at different levels in the system structure. A requirement is an expression of one or more particular needs in a very specific, precise and unambiguous manner. Example of a requirement: The bookshelf shall store a minimum of five books.
Requirements Elicitation	Use of systematic techniques, such as prototyping and structured surveys, to proactively identify and document customer and end user needs.
Requirements Engineering	Interdisciplinary function that mediates between the domains of the acquirer and supplier to establish and maintain the requirements to be met by the system, software or service of interest. Requirements engineering is concerned with discovering, eliciting, developing, analyzing, verifying, validating, communicating, documenting and managing requirements.
Requirements Management	Activities that identify, document, maintain, communicate, trace and track requirements throughout the life cycle of a system, product or service.
Stakeholder	Individual or organization having a right, share, claim or interest in a system or in its possession of characteristics that meet their needs and expectations.
Stakeholder Needs	Using the Concept of Operations to aid the understanding of the stakeholder concerns at the organizational level and the System Operational Concept from the system perspective, requirements engineering develops a set of objectively adequate, structured and more formal statements of stakeholder requirements and goals, referred to as stakeholder needs.

Requirements engineering encompasses two distinct sets of processes associated with requirements: **Requirements Development** and **Requirements Management** (Pohl, 2010). Figure 7 highlights the various activities that occur for each of these activities.

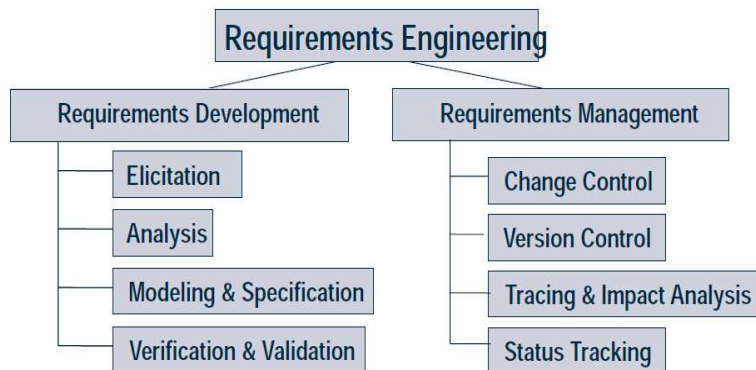


Figure 7. Overview of the Various Requirements Engineering Processes. (Pohl, 2010)

The requirements development process is often highly visible during a project development effort as it involves the generation of the requirements that form the basis of the product being developed. Multiple publications on requirements engineering devote a large percentage of content to the processes

of elicitation, analysis, specification, verification and validation (shown in Figure 7 as the fundamental processes of requirements development). A few of the publications used in the literature research, and throughout this dissertation, are highlighted below. Requirements management, on the other hand, is often addressed at a high level in these publications; demonstrated below by showing the amount of information on requirements management within the publication as compared to the overall content being addressed (percentages are based on quantity of text devoted to requirements management).

ISO/IEC/IEEE 29148, Systems and Software Engineering Life Cycle Processes - Requirements Engineering, also referred to as IEEE 29148, was prepared by the Joint Technical Committee with the IEEE Computer Society, and published in 2018. This standard describes the processes and products involved in engineering requirements throughout the life cycle of systems and software. This is more of an overview of what should be done, with less information on how to implement the processes. About 9% of the standard addresses requirements management processes.

The **INCOSE Guide for Writing Requirements (GfWR)**, was created by the INCOSE Requirements Working Group and published in 2019, provides guidance on how to express concepts, needs and requirements in a textual form in alignment with the standards of IEEE 29148. It defines the characteristics and rules for needs and requirements statements as well as for an entire set. It also provides guidance on attributes that can be used to expand on the needs and requirements (discussed further in Section 3.1.6). Many requirement engineering practitioners utilize the GfWR during the requirements development process as it provides specific application guidance and examples. GfWR is not a process document as much as a set of guidance to use when implementing the requirements engineering process, however approximately 50% of the content is useful information regarding requirements management implementation.

Requirements Engineering Fundamentals, Principles, and Techniques, authored by Klaus Pohl and published in 2010, is a comprehensive requirements engineering textbook with focus on elicitation practices, usage of scenarios and modeling to capture the system needs, addressing various formats for capturing formal requirements, performing analysis on the requirements to ensure they

address the needs appropriately, ensuring the resultant requirements support system test and verification, and providing templates and checklists for users to follow in their requirements development activities. A shortened version of this book is used as a study guide for the Certified Professional for Requirements Engineering Exam, to obtain certification with the International Requirements Engineering Board (IREB). About 9% of the textbook addresses the specifics of performing the requirements management processes.

Requirements Engineering, authored by Jeremy Dick, Elizabeth Hull and Ken Jackson and published in 2017, addresses many of the same themes as Pohl's book with a more condensed approach and specific examples of the various process steps within requirements engineering. It also provides examples of using a requirements management database for managing the product requirements, highlighting the features of traceability and change management. About 40% of the book addresses requirements management processes.

Requirements Engineering for Software and Systems, authored by Phillip A. Laplante and published in 2017, and provides "comprehensive treatment of the theoretical and practical aspects of discovering, analyzing, modeling, validating, testing, and writing requirements" with a focus on software-intensive systems. About 8% of the book addresses requirements management processes.

Requirements Engineering and Management - A Systems Approach, authored by Alberto Sols and published in 2016, addresses requirements development from the technical, financial and legal aspects of a system. A comprehensive case study is provided for users to have an example of the various processes. About 7% of the book addresses requirements management processes.

Requirements Management, A Practice Guide, authored by the Project Management Institute (PMI) and published in 2016, provides discussion and definition of requirements-related activities, with a target audience of project managers, team members and stakeholders. Despite the title, only 6% of the book actually addresses requirements management processes. Upon review of the material, it appears that the authors are establishing approaches towards general requirements engineering at a high level, leaving the aspects of requirement management logistics to the reader to define and implement.

Requirements Management - The Interface Between Requirements Development and All Other Systems Engineering Processes, authored by Colin Hood, Simon Wiedemann, Stefan Fictinger, and Urte Pautz, published in 2008, focuses on promoting understanding of the central role played by requirements in systems engineering projects. This book provides the viewpoints of other engineering disciplines, describing how the information from their efforts on the project closely relate to the project requirements, and how their association in the management of the requirements can enable all disciplines to succeed. Disciplines addressed include risk management, change management, configuration and version management, test management, quality management, and project management. The key message in this book is alignment of all of the project data with respect to the requirements (a data centric approach to requirements management). The book also provides a capability model to show the ranging aspects of maturity an organization can display with respect to requirements management from these different perspectives. About 90% of this book describes approaches to requirements management from the perspective of the different disciplines; however, the complete focus is on the information management discussion, there is very little methodology of how to manage requirements from a requirements engineer-role perspective.

Customer-Centered Products - Creating Successful Products through Smart Requirements Management, authored by Ivy Hooks and Kristin Farry, published in 2001, aims to provide practical requirement definition process and requirement management techniques with examples and checklists. Particular focus is provided for aspects that need to be considered during requirements development (interfaces, operation concepts, human interactions, etc.), along with the methods to generate clear requirement statements. For requirements management processes, content exists for addressing requirement allocation, traceability, baseline creation and change control; about 25% of the book addresses requirements management processes.

2.2.2 The Requirements Development Processes

The requirements development process starts with the collection of the stakeholder needs and project constraints, then goes through the activities of elicitation, negotiation, and documentation (Pohl,

2010). Requirements elicitation is more than a discussion with the stakeholders; it often involves an analysis of what is desired (outcome) compared to what exists currently; this analysis results in a feasible set of solutions of how the stakeholder needs can be achieved by various functions implemented in the system design. This analysis can often be achieved through the creation of a system model which captures the use cases, activities, and desired behaviors of the system. This analysis will often utilize the concept of concurrent engineering, which is the assessment of required design features to address multiple engineering concepts such as human factors, safety, producibility, inspectability, reliability, maintainability, logistics, manufacturability, etc., to support an assessment of completeness of the capabilities and needs definition for the product.

The development of requirements starts with the needs and a set of information about the product being developed and progresses through stages where the product design is evolved. Outputs from the design effort are applied to the requirements. This progresses iteratively (as described by Figure 6) as continued analysis provides further definition of required capabilities and functions. Dick, et al, describe the systems engineering sandwich, where the requirements are intertwined with the system model and design effort as shown in Figure 8.

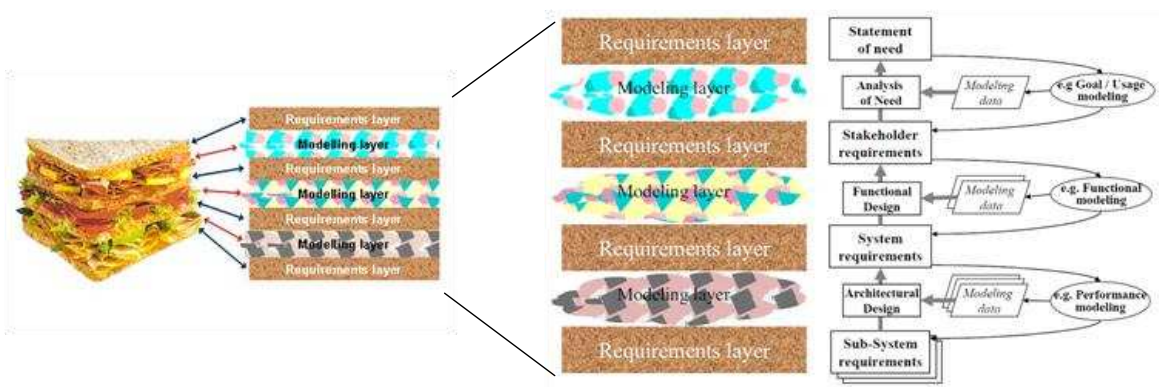


Figure 8. Systems Engineering Sandwich showing Interrelation of Requirements and Models. (Dick, Hull, & Jackson, 2017)

As the initial design concepts mature, the requirements generated provide input into further design efforts and modeling, which then take the system from a **problem-domain focus** to a **solution-**

domain focus as the set of needs go to a set of system requirements and then to a set of subsystem and component requirements (down the "left side of the Vee" in Figure 5).

The requirement development process model is shown in Figure 9, which depicts how requirements development iterates between requirements determination and design efforts.

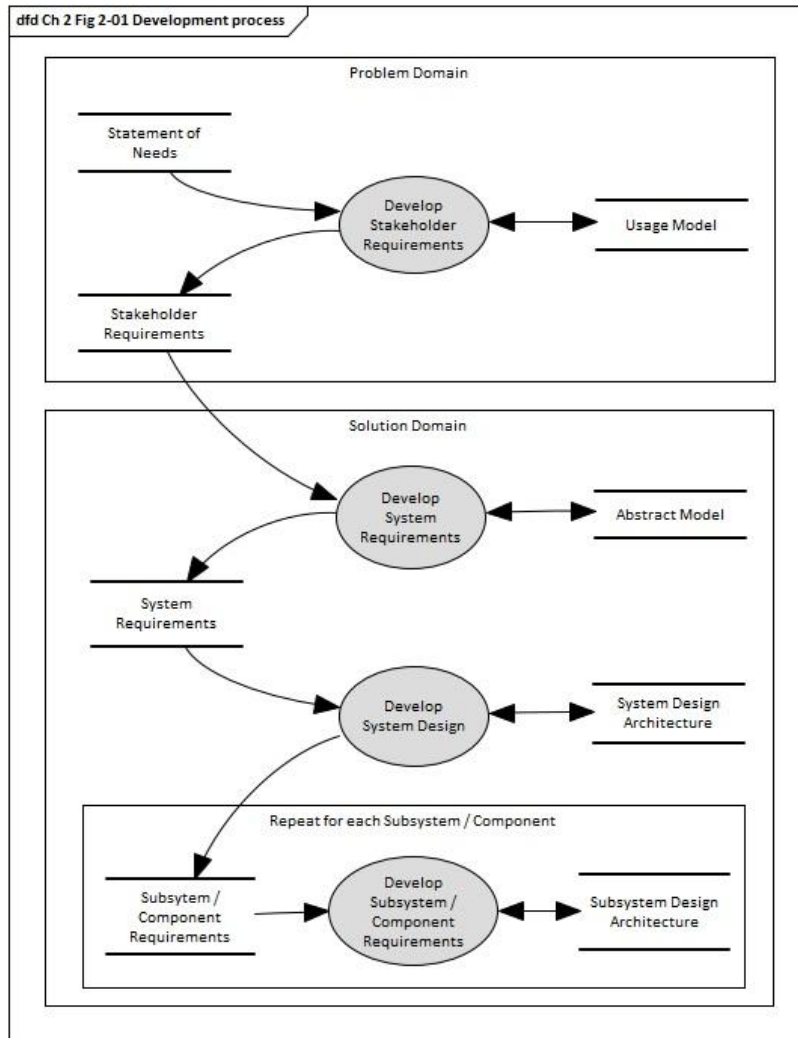


Figure 9. Requirements Development Process Transitioning the Problem Domain to the Solution Domain. (Dick, Hull, & Jackson, 2017)

Another view of this activity is shown in Figure 10, which highlights that the requirements engineering process consists of a continual transformation of higher level needs (or requirements) to lower level derived requirements over the multiple levels in a system.

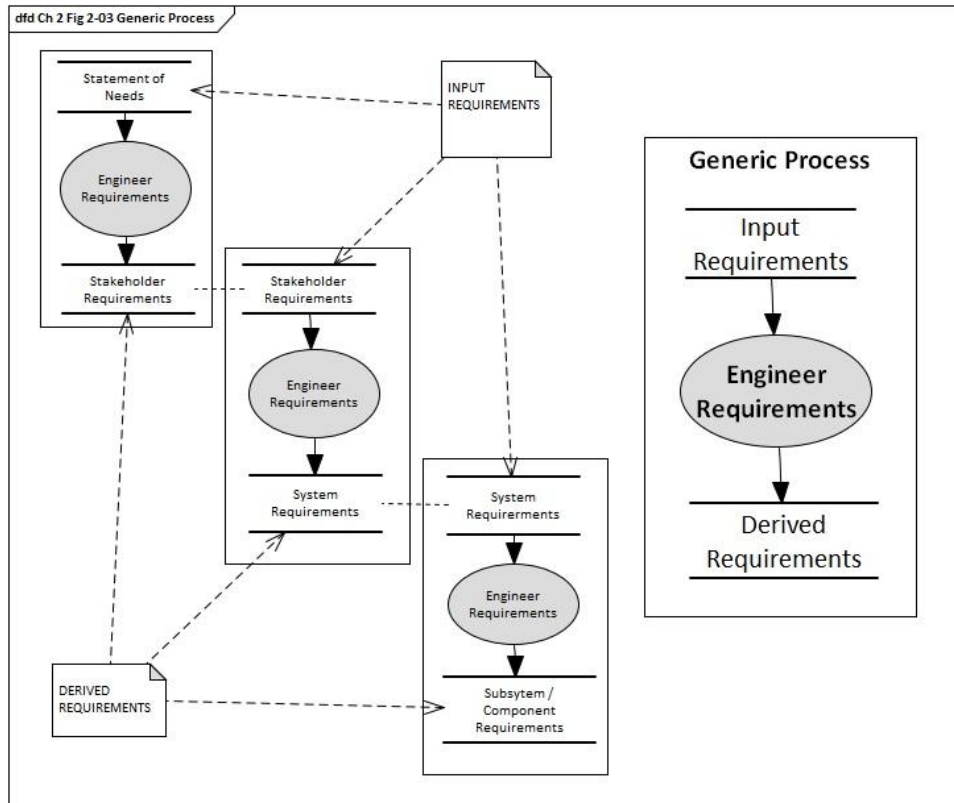


Figure 10. Requirements Engineering Process Showing Transformation of Input to Derived Requirements. (Dick, Hull, & Jackson, 2017)

Transformation of stakeholder needs to product requirements also takes into account the requirements for business processes and organization strategies, which feed into the system of interest and its resultant technical requirements. This set of transformation activities is depicted graphically in Figure 11, and often results in a set of inputs from the organization enterprise related to strategies such as a pursuit of certain markets or a desire to generate certain organizational capabilities and competencies; these are inputs to ensure the products being developed are in alignment with the developing organization's goals.

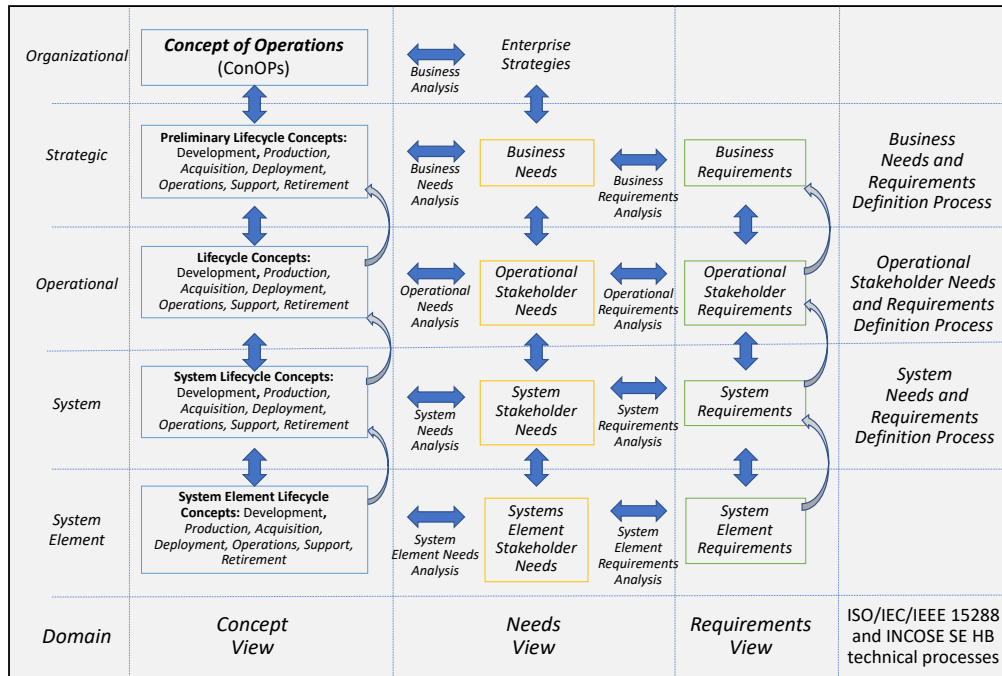


Figure 11. Transformation of Concepts Into Needs and then to Requirements. (INCOSE, 2019)

The organization often manages the set of products or services as part of a portfolio, and for each type of product produced by the business operational level there are different types of life cycle concepts, needs, and requirements. The strategic and operational level stakeholders will identify potential solutions in the form of products and services to address potential problems, opportunities, and threats. Given there can be multiple products and services, there will be individual operational level life cycle concepts, needs, and requirements for each product and service developed by the organization as shown in Figure 12 (INCOSE, 2021 draft).

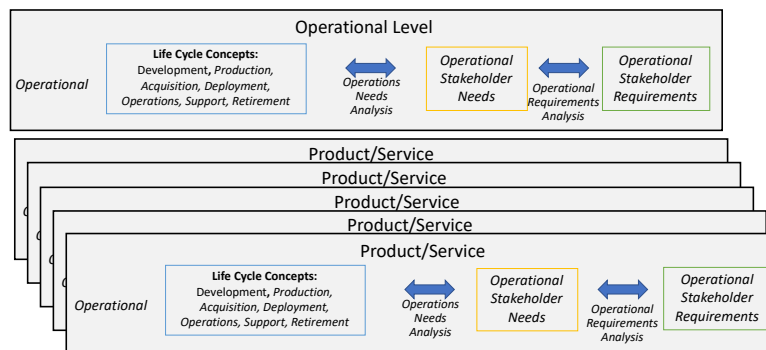


Figure 12. Business Operational Level Transformation of Concepts Into Needs and Requirements for multiple Products and Services. (INCOSE, 2021 draft)

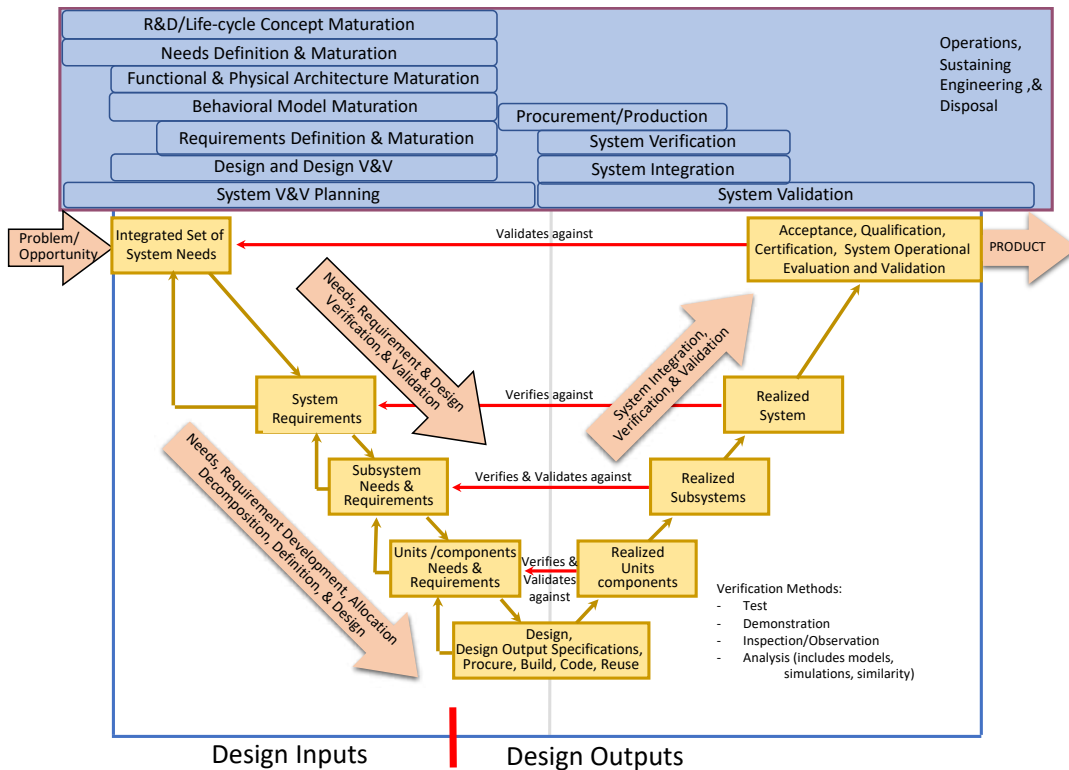
During product requirements definition, the resultant requirements will typically fit into three types of categories shown in Table 2. These categories cover what is to be done, how it might be achieved, and what boundaries exist that it must work within.

Table 2. Requirement Categories. (Pohl, 2010)

Category	Definition
Functional	Defines what the system should provide, its behavior, and in some cases, what it should not do.
Quality	Defines properties that the system should have in order to do what must be done; properties of the system, a component, a service or function; includes features such as availability, supportability, security, training, environmental.
Constraint	An organizational or technical requirement that restricts the way in which the system shall be developed; requirements that on the surface, resemble design constraints or project constraints.

Part of the requirements development process is an assessment of quality of the resultant requirements. This will often consist of assessing the requirement against a higher level need or requirement to ensure it captures the intent (validation), and that the requirement conforms to standards for requirements statements (verification). Analysis may be involved to ensure the requirement addresses the system stakeholder needs, which can also be achieved by modeling the requirement expected outcome in an overall system model. An assessment of completeness is also part of the quality check to ensure there are no missing requirements associated with the higher level needs and requirements. This can also be done through an analysis, showing how the requirement traces to a function or capability in the system model, or traces to a higher level requirement or need.

As described in Section 2.1, the systems engineering process is often iterative and recursive, and this is additionally true of requirements engineering over a project life cycle. The *INCOSE Guide for Writing Requirements* provides an updated view of the Vee model (Figure 13), showing that the requirement development activity starts with the stakeholder needs; allocation and development are performed to produce requirements at the system level; and this process is repeated to form the requirements at each of the lower levels in the product structure. The feedback loop to the higher level demonstrates that there is an iterative process in the development of the lower level requirements.



Adapted from Ryan, M. J.; Wheatcraft, L.S., "On the Use of the Terms Verification and Validation", February 2017 and INCOSE SE HB, Version 4, Figures 4.15 & 4.19

Figure 13. Requirement Evolution over the Systems Engineering Vee Model. (INCOSE, 2019)

In Figure 13, the product's requirements along the different levels of abstraction are generated during the design and development activities (left side of the Vee); each level ultimately shows it satisfies its allocated requirements and needs through system verification and validation activities (right side of the Vee).

A significant portion of the requirements engineering effort is often done in the early part of the project life cycle, when requirements engineering typically occurs concurrently with concept and preliminary design activities. A factor in this activity is the concept of concurrent development for multiple levels of the product. While it appears the development work from system to lower levels occurs sequentially (per the Vee model and other life cycle models), in actuality many aspects of a product development occur simultaneously. Lower tiers of the product development will often begin their early design work and modeling efforts to assess feasible concepts while the higher level is defining their requirements; the results are then utilized in the overall requirement development effort.

Figure 9 previously depicted the framework for developing the set of requirements throughout the life cycle of a system. What this figure did not address specifically is the concurrent and iterative nature of the development effort. Per IEEE 29148, the main reasons for iteration during requirements development include (IEEE 29148, 2018):

- Purposeful iteration within requirements analysis to apply results;
- Planned iteration from downstream activities back to requirements analysis because of a predicted, significant, genuine rate of change of requirements that reflect change of need;
- Planned or unplanned iteration from downstream activities back to requirements because of feasibility and balance issues arising from risk due to technology or implementation issues, or risk due to limited knowledge of them;
- Unplanned iteration from downstream activities back to requirements because of other solution issues, such as changes to or defects in non-developmental system elements, or obsolescence of system elements;
- Reverse engineering of requirements for reasons of regulatory compliance; and
- Limited iteration from downstream activities back to requirements analysis because of the reality that requirements can never be perfect, nor is it cost-effective to try to make them so.

Figure 14 expands upon Figure 9 by showing how this iteration loops back as a change to the original set of requirements; note that the concept of requirements changes are discussed in further detail in Section 3.1.3.

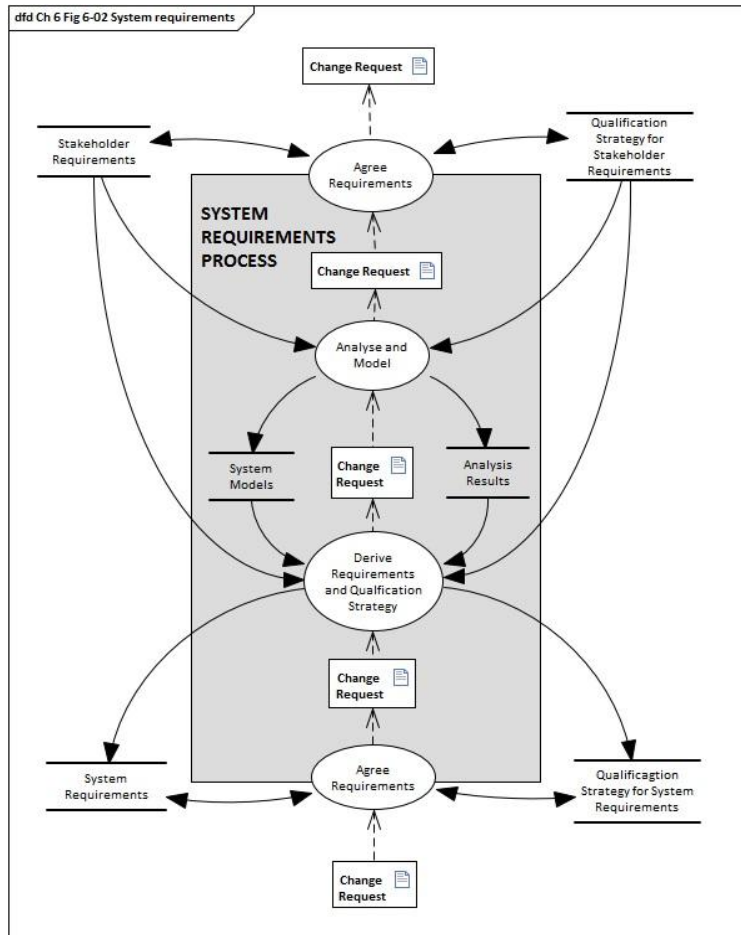


Figure 14. Requirements Development and the Change Feedback Loop. (Dick, Hull, & Jackson, 2017)

2.2.3 Requirements Development Among Different Organizations

Up to now the process of requirement development has been expressed in terms of *what* is done with lack of consideration related to *who* is performing the activity. An added complexity in this process can exist if the development effort is done by multiple organizations at various system levels, generating hand off points within the process.

A system owner often generates the needs, develops the system requirements, and then provides specific requirements to product developers to further design and produce, leading to further requirements engineering at lower levels. In some instances, the system owner may be responsible for the entire development effort, where the product development teams are within their organization and all development work is done by the same organization or company. In many complex systems, however,

there is often a mix of organizations involved, where the system owner will hand off development of lower level elements to other organizations or companies (often through a contractual relationship).

An example of development being implemented over multiple companies is with the National Aeronautics and Space Administration (NASA), a U.S. Government agency responsible for science and technology related to air and space. With the NASA Human Landing System (HLS) Program (described in Section 4.6), NASA provides the overall system requirements, serving as the overall system integrator, and then establishes contracts with other companies to provide the finished system elements. Those companies will often further subcontract development work to other companies to produce subsystems and components. Ultimately, the lowest level components will need to satisfy the overall NASA HLS requirements, bringing a need to align the requirements engineering outcomes across multiple companies to a reconciliation at the NASA level.

A different example which shows the system developer owning the overall design solution is with Virgin Galactic's space tourism effort (<https://www.virgingalactic.com/>). With this approach, Virgin Galactic's Spaceship Company is both the system owner and developer, addressing the requirements, designs, and fabrication. In this paradigm the system developer oversees the entire set of requirement and product development from system to lower levels, subcontracting out lower level component fabrication as necessary but maintaining control of the entire system design development. Figure 15 and Figure 16 highlight the different paradigms of a multi-organization development effort compared with a single developer example (each organization is portrayed in a unique color), showing the various handoff points that exist when there are multiple developing organizations.

Note: Sometimes multiple organizations within a single company may be involved in the product development, still necessitating the need for handoff points but less likely requiring a formal contractual arrangement.

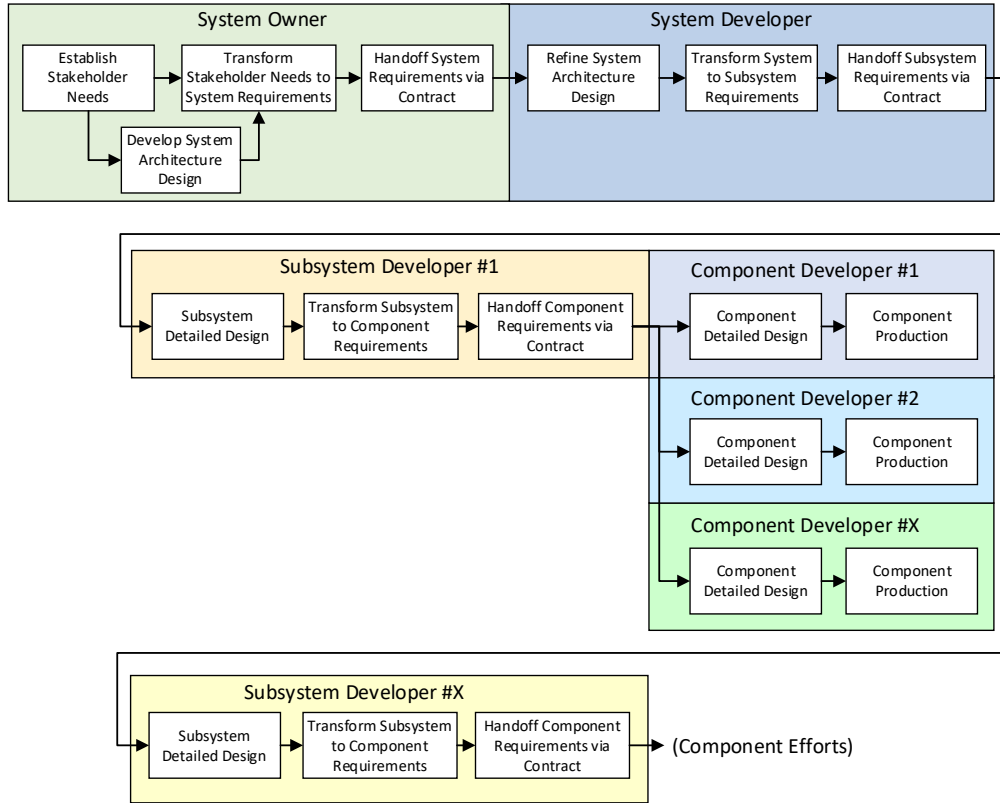


Figure 15. Example of Requirement Development Work with Multiple Development Organizations.

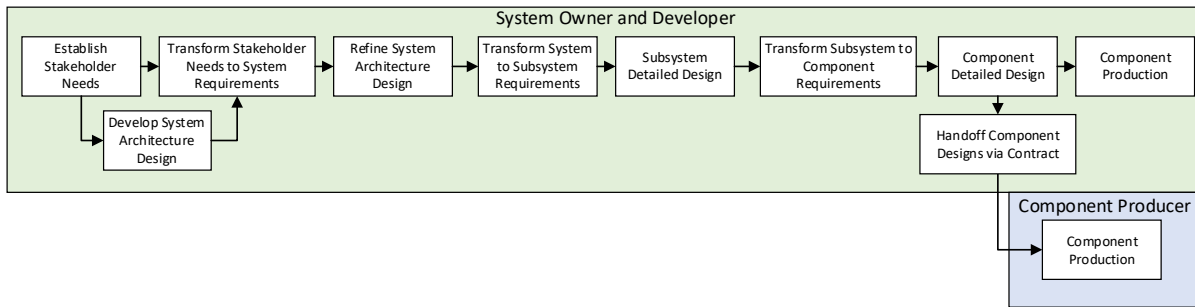


Figure 16. Example of Requirement Development Work with a Single Development Organization.

2.2.4 Concept of Requirements Traceability

For many projects the development of the system involves multiple levels of product development, where each level conforms to requirements derived to satisfy the overall stakeholder needs. This generates a need to show the different products and sub-products being created, referred to as a

product breakdown structure (PBS). Examples of various product breakdown structures are shown in Figure 17.

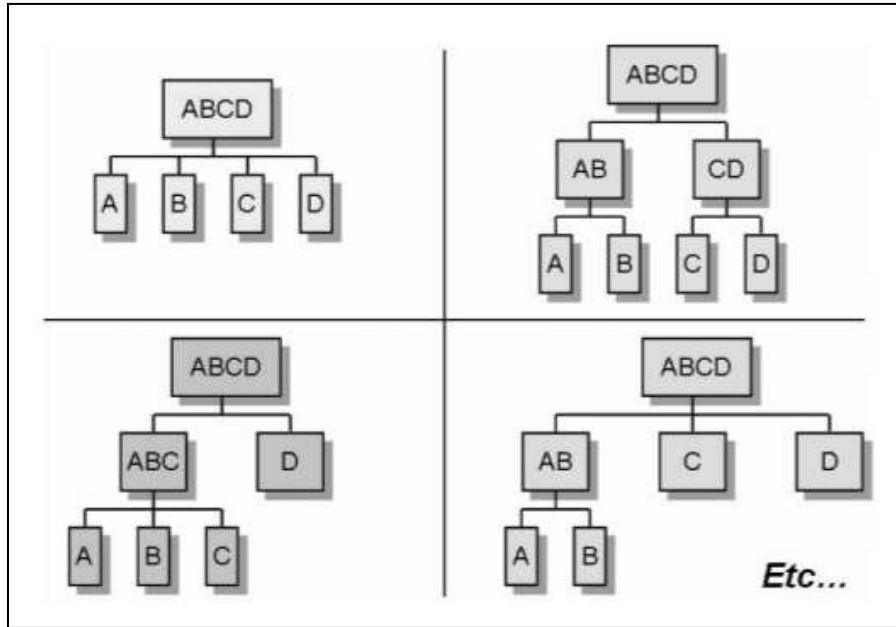


Figure 17. Various Examples of Product Breakdown Structures. (Forsberg, Mooz, & Cotterman, 2005)

The exact type of product structure used is typically determined by the strategy for system development, integration, and procurement. As described earlier, the system developer can either produce the entire system, or purchase some of it from other companies. Requirements are typically created for each level in the product structure, leading to a need to utilize a process to ensure traceability, change control, and a consistent approach to requirement maturation to ensure that products being produced at all levels align to the same set of requirements and support the development of an integrated system that addresses the stakeholder needs.

Requirements that are applied throughout the entire system is referred to as a "requirement set", and a quality assessment of the entire set is checked against characteristics like those in Table 3.

Table 3. Characteristics of a Well-Formed Set of Requirements. (INCOSE, 2019)

Characteristics of a Well-Formed Requirement Set	
<p>Formal Transformation Quality Characteristics:</p> <ul style="list-style-type: none"> • <i>C10 - Complete:</i> The need or requirement set for a given SOI stands alone such that it sufficiently describes the necessary capabilities, characteristics, constraints, interfaces, standards, regulations, and/or quality factors to meet the needs without requiring other sets of needs or requirements at the appropriate level of abstraction. • <i>C11 - Consistent:</i> The set of needs contains individual needs that are unique, do not conflict with or overlap with other needs in the set, and the units and measurement systems they use are homogeneous. The language used within the set of needs is consistent (i.e., the same words are used throughout the set to mean the same thing). 	<p>Agreed-To-Obligation Characteristics:</p> <ul style="list-style-type: none"> • <i>C13 - Comprehensible:</i> The set of need statements and resulting requirement statements must be written such that it is clear as to what is expected of the entity and its relation to the system of which it is a part. • <i>C14 - Able to be validated:</i> It must be able to be proven that the set of needs will lead to the achievement of the product goals and objectives, stakeholder expectations, risks, and concepts within the constraints (such as cost, schedule, technical, legal and regulatory compliance) with acceptable risk. It must be able to be proven that the set of requirements will lead to the achievement of the needs and higher-level allocate requirements within the constraints (such as cost, schedule, technical, and regulatory compliance) with acceptable risk.

Requirements for each product are often captured in individual specification documents, which is then visually represented in a system's specification tree showing the entire product requirement set (example shown in Figure 18).

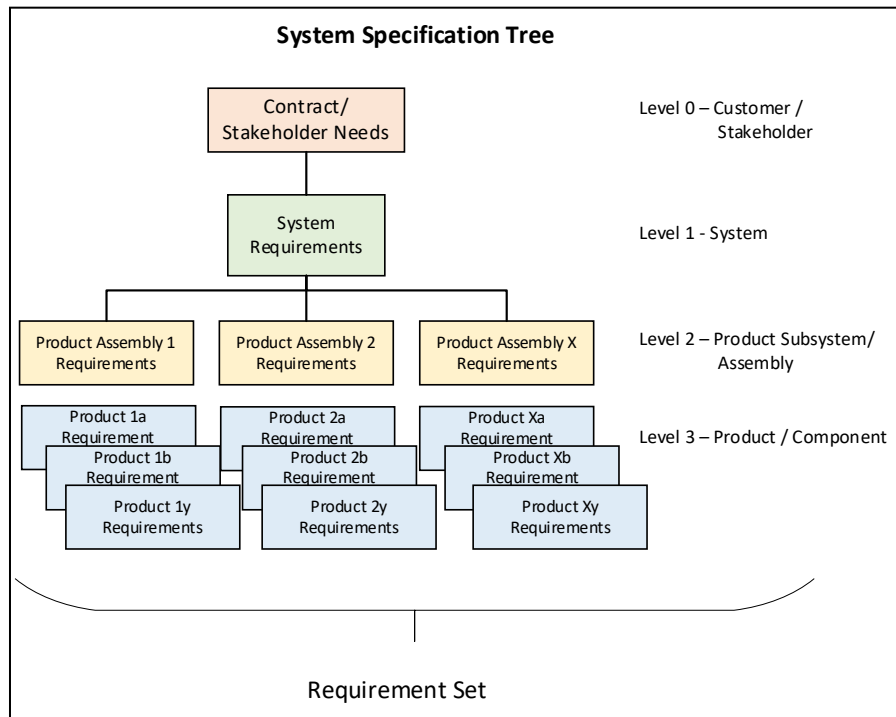


Figure 18. Example of a Product Requirement Set and Specification Tree.

Requirements traceability is the identification and management of the requirements starting from the stakeholder needs and going through the complete requirement set, it includes assessment of requirement “parentage” (parent-child relationships) from the highest-level system requirements to the lowest-level product requirements. The primary rationale that this information is tracked includes:

- Traceability shows that the system level requirements **are properly allocated** to the lower levels;
- Traceability shows how the lower requirements **are sufficient and necessary**, and how they **respond to a higher level need**;
- Traceability shows how the verification data for individual requirements **relate to the overall system verification**; and
- Traceability allows for **requirement changes to be assessed for impact** at different levels of the product structure.

Per the *INCOSE Guide for Writing Requirements*, the requirement attributes for "trace to parent" and "trace to source" ensure that the requirement supports a driving need (a complete list of the recommended requirement attributes is provided in Section 3.1.7.) Without a trace, there is potential a requirement will not support the stakeholder need, may conflict with a stakeholder need, or may be an over-specification resulting in design constraints and cost impacts.

The requirement trace between levels in a product structure can be shown in a matrix format, such as with a Requirements Traceability Matrix (RTM). An example RTM for a coffee maker is shown in Table 4.

Table 4. Example Requirement Traceability Matrix.

Stakeholder Need	System Requirement	Subsystem Requirement	Component Requirement
CN-1 Customers need to have enjoyable moments with a great coffee.	CM-1 After the operator has positioned the cup correctly, the system shall allow the operator to get a filled cup in equal to or less than 60 seconds.	PWR-1 The power subsystem shall enable 28 Volts + 5 Volts/-5 Volts to heating element within 2 seconds of receiving the power enable signal. [Other Subsystem Requirements continued]	PS-1 The power supply shall produce 28 Volts +/- 0.5 Volts of power. PS-2 The power supply shall enable power output upon receipt of the power enable signal.

2.2.5 Results of Poor Requirements Engineering

There have been multiple studies about the impact of poor requirements on a project. Most of these studies address the results of poor requirements development; however, there is also an impact of a poor management process with respect to lack of traceability, change control, and monitoring of requirements quality. Based on these studies, and others like them, poor requirements engineering (both requirements development and requirements management) has been shown to have a negative impact on project success.

Cost Impacts of Poor Requirements Development and Management

Hooks et al. describe the various types of requirement errors that could exist, including errors in incorrect facts, omissions, inconsistency, ambiguity, and misplacement (Hooks & Farris, 2001). The cost of having requirements errors is a function of the project life cycle, where the most costly errors are discovered in the later portion of the project; it can cost over 50 times to correct a system due to a requirements error when the error is found during the test phase than if discovered during the requirement development activity (Stecklein, 2004).

Per the Project Management Institute (PMI), "the real-world practice of requirements management continues to vex organizations, both small and large, for many reasons. And doing it poorly leads to project failure. 47% of unsuccessful projects fail to meet goals due to poor requirements management. Our study shows that far too many organizations still lack maturity in requirements management. They lack the necessary resources to do it properly. They are failing to develop the relevant skills in the people they do have. And, not surprisingly, executive management and sponsors are not fully valuing the importance of excellence in requirements management. It's costing them. For every dollar spent on projects and programs, 5.1 percent is wasted due to poor requirements management" (PMI, 2014).

The PMI report additionally notes that only 20% of organizations they surveyed reported high requirements process maturity. What does maturity look like? Per the report, "Maturity is about the levels

of capability and efficiency an organization demonstrates across its people, processes, and tools when performing requirements management activities on a project or program. It is achieved by continuously monitoring capabilities, identifying areas for improvement in the requirements process and implementing improvements to ensure optimal performance of the requirements-related activities. It is enabled by organizational and leadership recognition of the importance associated with the practice of requirements management on projects and programs.” (PMI, 2014).

A study implemented by NASA showed that projects which spent less than 5% of total project costs on the requirements engineering process experienced an 80% to 200% cost overrun, whereas those that invested 8% to 14% experienced less than a 60% overrun. The NASA study concluded that an investment of 8% to 14% of total program costs on the requirements processes will result in less overruns on the project (Gruhl, 1992).

In late 2018, engineering.com surveyed 246 design and engineering professionals about the growing complexity of their company’s products and how product requirements are being managed. The survey results noted that more than "4 out of 5 design team have experienced product design failures due to poor requirements management" (Engineering.com, 2018). The survey report noted that 49% of the respondents said they worked in a regulated industry, and only 15% worked in organizations that invested in a formal, dedicated requirements management solution. A breakout of the types of program impacts based on the survey is shown in Figure 19.

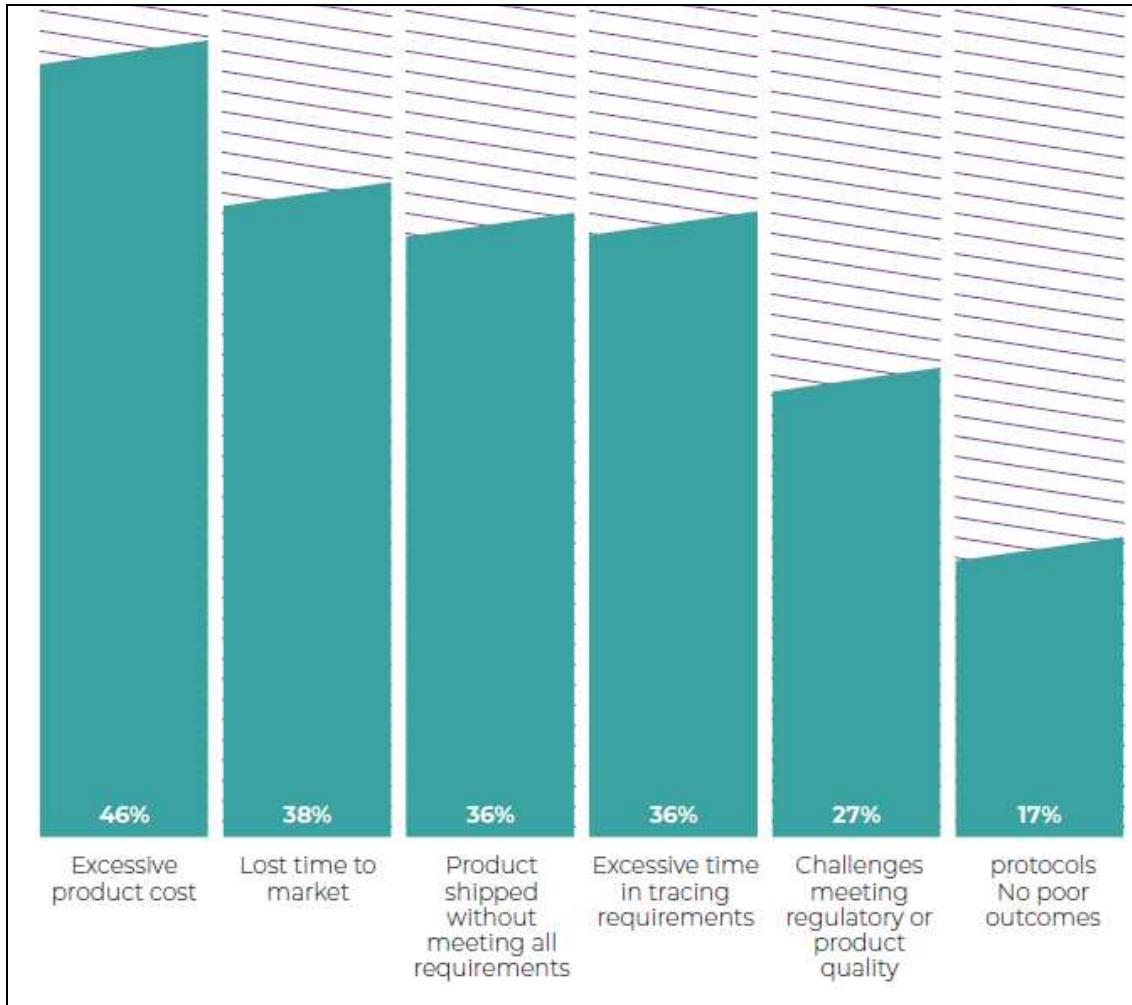


Figure 19. Failure Types and Occurrence Based on Poor Requirements Management. (Engineering.com, 2018)

Safety Impacts of Poor Requirements Development and Management

Howard, et al, address how poor requirements engineering can impact a system's overall safety, particularly for real-time software systems. From their research, most accidents related to software in the aviation industry stem from requirements problems, particularly related to incompleteness of the requirements (Howard & Anderson, 2002). Requirements are defined as "incomplete" if the requirement set omits requirements and constraints for significant behavior or if the requirements are ambiguous (subject to more than one interpretation); this finding relates to the assessment of requirements quality for the entire requirements set. Per Howard's analysis, a requirements process that only includes consideration of the desired outcomes, and does not evaluate all possible modes of operation, can produce

unintended consequences that have been shown to result in accidents leading to injury or loss of life. These safety impacts can result in loss of company reputation (or disbarment from further contracts), which leads to an impact on future business.

2.3 Requirements Engineering Conclusions

Some key elements associated with the topic of requirements engineering include:

- Requirements engineering encompasses the entire project life cycle.
- Scope and organization of a project can impact the approach used in development and management of requirements.
- Multiple publications address requirements engineering, with only a small percentage addressing the subject of requirements management.
- When requirements engineering is done poorly there is an impact to the project in terms of cost, schedule, and technical ramifications.

The next chapter presents a more in-depth presentation of requirements management, addressing key areas of the process, variations of application, and how the practice has been evolving over the last several years.

CHAPTER 3: INVESTIGATION INTO REQUIREMENTS MANAGEMENT

3.1 Current Approaches to Requirements Management

3.1.1 Definition of Requirements Management

Expanding upon the earlier definition of requirements management from Table 1, the following definitions in Table 5 are extracted from various publications on requirements engineering.

Table 5. Published Definitions for Requirements Management.

Requirements management is....	Source
The discipline of gathering, expressing, organizing, tracing, analyzing, reviewing, agreeing, tracking, communicating, changing and validating requirement statements; Managing the documents.	Requirements Engineering (Pohl, 2010)
The set of procedures that support the development of requirements including planning, traceability, impact analysis, change management and so on. The sum of the interfaces between requirements development and all other systems engineering disciplines such as configuration management and project management. The purpose of requirements management is to manage the requirements of the project's products and product components and to identify inconsistencies between those requirements and the project's plans and work products.	Requirements Management – The Interface Between Requirements Development and All Other Systems Engineering Processes (Hood, Wiedemann, Fichtinger, & Pautz, 2008)
The tasks of establishing a requirements baseline and maintaining traceability, change control, and configuration management.	Requirements Management A Practice Guide (PMI, 2016)
Identifying, documenting, and tracking system requirements from inception through delivery. One of the most overlooked aspects of requirements engineering, requirements management involves managing the realities of changing requirements over time. It also involves fostering traceability through appropriate aggregation and subordination of requirements and communicating changes in requirements to those that need to know.	Requirements Engineering for Software and Systems (Laplante, 2017)
The process of dealing with proposed changes to a set of approved requirements.	Requirements Engineering and Management, A Systems Approach (Sols, 2016)
Management of the project business, budget, and technical baselines. The objective is to keep the three baselines congruent. The process includes baseline change management and authorization. Also included are requirement flowdown, traceability, and accountability.	Visualizing Project Management (Forsberg, Mooz, & Cotterman, 2005)
The process that captures, traces and manages stakeholder needs and the changes that occur throughout a project's life cycle.	Requirements Engineering (Dick, Hull, & Jackson, 2017)

Per Hood, et al., "while requirements development assures that what is to be developed is indeed what the customer wants, requirements management integrates the data created during requirements development into the overall project flow." (Hood, Wiedemann, Fichtinger, & Pautz, 2008).

Requirements management can be viewed as *information* management associated with managing the requirement data for a system. Based on Table 5 and Figure 7 (Overview of the Various Requirements Engineering), there are a few driving processes that are fundamental to the overall requirements management process:

- Requirements management addresses **requirement traceability** of the requirement set;
- Requirements management addresses **version capture and change control** for the requirement set; and
- Requirements management addresses **completion status and stability** of the requirement set.

The requirements management process spans the entire product development life cycle. Ineffective requirements management results in program technical, cost, and schedule challenges with results such as cost overruns, poor customer satisfaction, potential fines, and lack of user acceptance, as described in Section 2.2.5.

It is worth noting that, because requirements are generated from a set of stakeholder needs, the process of "requirements" management is actually "needs and requirements" management. However, as this is not yet a term used in current literature, the nomenclature "requirements management" will be used in this dissertation with the understanding that its scope is inclusive of managing both the needs and the requirements for a product.

3.1.2 Requirements Management Process Models

Several organizations and authors have defined process models for requirements management, several of which are shown in this chapter. With each of these, it was noticed that only a few provided a graphical process flow depicting specific activities involved within requirements management. In cases where only a written description of process steps are provided, a graphical representation of the process

has been created for this dissertation based on those descriptions. After description of the various models a comprehensive model is proposed that contains the common elements from each published model.

IEEE 29148 Requirements Management Process

IEEE 29148 provides definitions, activities and artifacts involved in requirements engineering throughout the life cycle of systems and software. This standard does not provide a graphical representation of the process steps within requirements engineering, or the processes of managing the requirements, so a high level model is shown in Figure 20 that is reflective of the content within the standard.

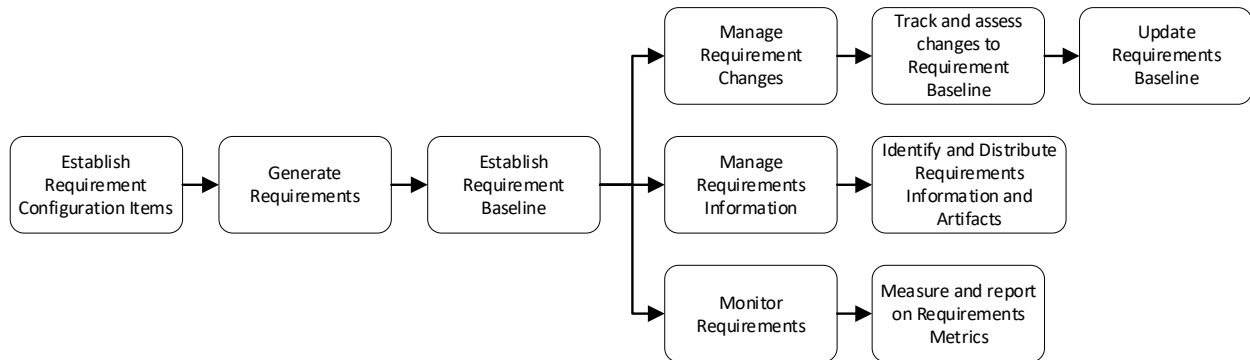


Figure 20. The IEEE 29148 Requirements Management Process Model.

Listed within the standard are some examples the themes of managing requirement changes, managing requirements information, and measuring requirements evolution; these examples contain high level statements about requirements traceability, requirements quality, and requirements stability. IEEE 29148 does not address the actual application of requirements engineering; rather, it is by design a standard that provides concepts and definitions, and leaves to the reader on how to apply the various concepts. It is for this reason that other documents, such as systems engineering handbooks and various published books, were generated to provide guidance on how the processes of requirements management can be performed.

NASA Requirements Management Process

NASA has developed processes to support the systems engineering activities, and in particular provided a set of process steps used within the requirements management process (Figure 21).

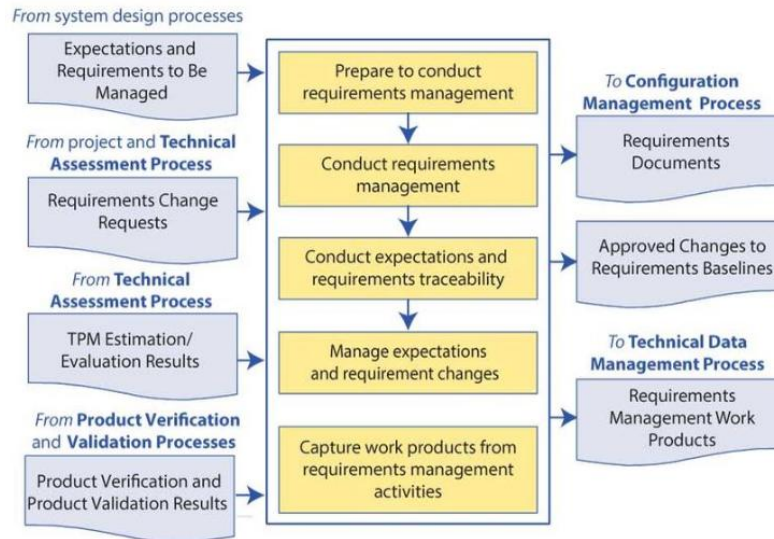


Figure 21. The NASA Requirements Management Process. (NASA, 2020)

The NASA requirements management process addresses many of the same themes as the IEEE process, however it expands on the various inputs from related technical processes, and explicitly shows traceability as one of the process steps. Additionally, the *NASA Systems Engineering Handbook* provides written guidance as to how to perform key activities, such as establishing a plan for executing requirements management, organizing the requirements in a hierarchical tree structure, ensuring bidirectional trace of the requirements, evaluating change requests, and maintaining consistency between the requirements and other product data (architecture, design, concept of operations). Success of the application of the NASA process has varied, however, as shown later in Chapter 4.

INCOSE Requirements Management Process

INCOSE has provided several process steps associated with the requirements development activity in the *INCOSE Systems Engineering Handbook*, as shown in Figure 22.

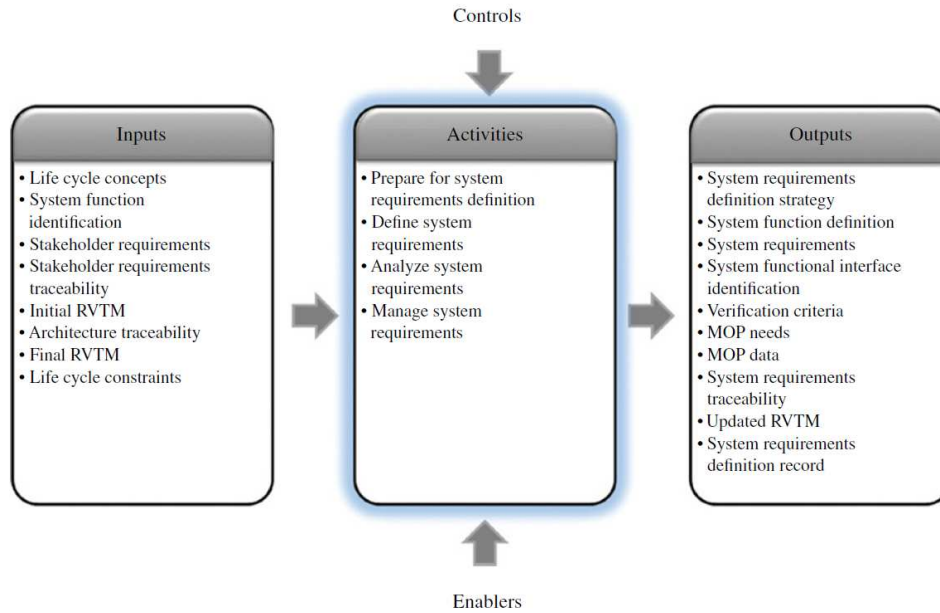


Figure 22. The INCOSE Requirements Definition Process Model. (INCOSE, 2015)

The requirements management process is referenced throughout the *INCOSE Systems Engineering Handbook*, containing a series of process steps that are recreated graphically in Figure 23. The requirements management activity descriptions are distributed among several other technical processes within the handbook, such as the configuration management process, information management process, and measurement process.

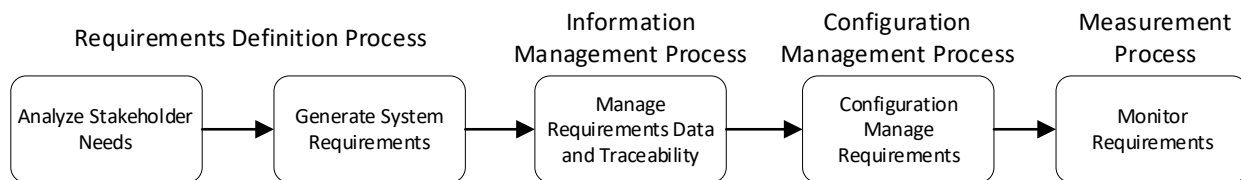


Figure 23. The INCOSE Requirements Management Process Model.

INCOSE has also published guidelines to support the application of requirements engineering, such as the *INCOSE Guide for Writing Requirements*, describing how stakeholder needs are transformed to requirements, which are then transformed to design and ultimately to a realized system (Figure 24).

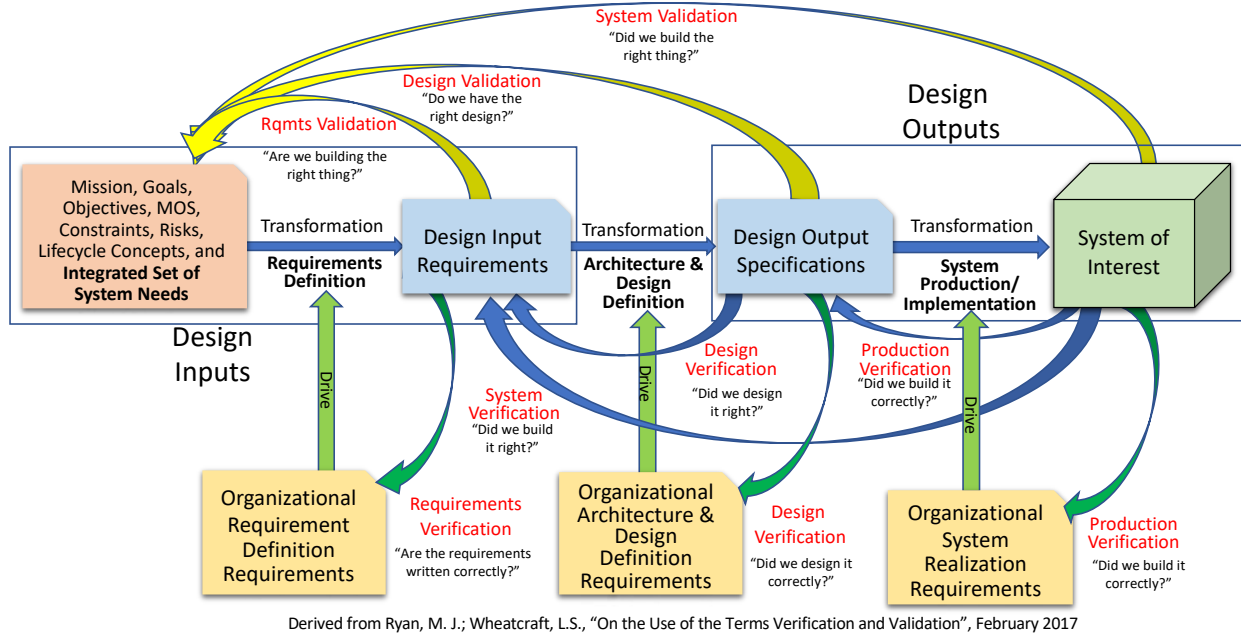


Figure 24. The INCOSE Product Realization Process. (INCOSE, 2021 draft)

Figure 24 is expanded upon in Figure 25 to show a process model of the various steps for product realization, with the underlying, concurrent process for requirements management continuously performed throughout the entire effort.

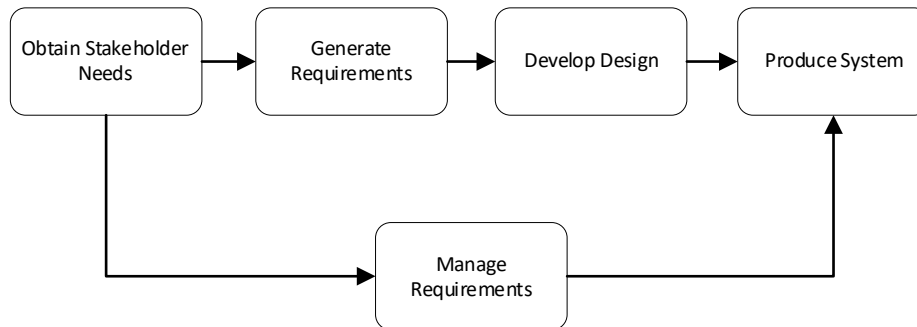


Figure 25. INCOSE Product Realization Process with Concurrent Requirements Management Effort.

For the generation of requirements (transforming the needs to requirements statements), the guide notes that "defining and documenting concepts, needs and requirements for an entity is more than just an exercise in writing; it is a systems engineering activity where the requirements engineer (RE) or business analyst (BA), through formal analysis, determines specifically what the customers need the entity to do to satisfy a specific problem or opportunity. This formal analysis starts with the RE or BA engaging the

customers and other stakeholders in order to formalize a number of concepts which provide an implementation-free understanding of what is expected of the entity (design inputs) without addressing how (design outputs) to satisfy the mission, goals, and objectives to satisfy a problem or opportunity within defined constraints with acceptable risk." (INCOSE, 2019).

The idea of generating the requirements for what the system needs to accomplish compared to specifying an implementation of how the system should be created is expanded upon in a paper by Wheatcraft, et al, which describes how the requirements process is impacted by the development of the design, going from the problem space to the solution space as the requirements change from "design input" requirements to "design output" requirements (shown in Figure 26). This paper on information based requirements development and management (I-RDM) and usage of model based design (MBD) is discussed further in Section 3.2.1.

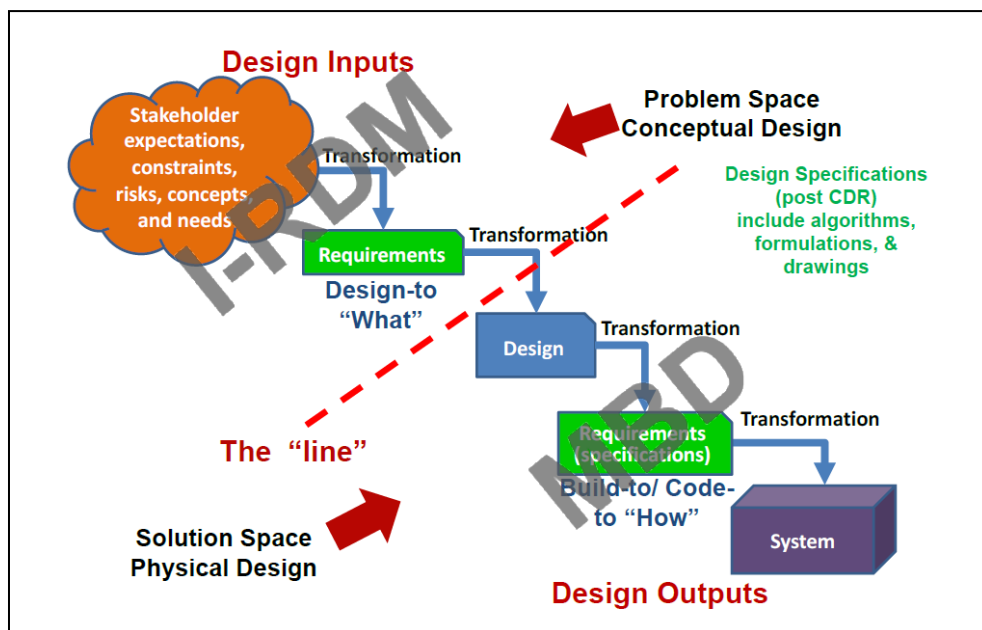


Figure 26. The INCOSE Information-Based Realization Process. (Wheatcraft, Ryan, Dick, & Llorens, 2019)

PMI Requirements Management Process

Project Management Institute (PMI) provides a requirements process model which encompasses both requirements development and management activities (reference Figure 27).

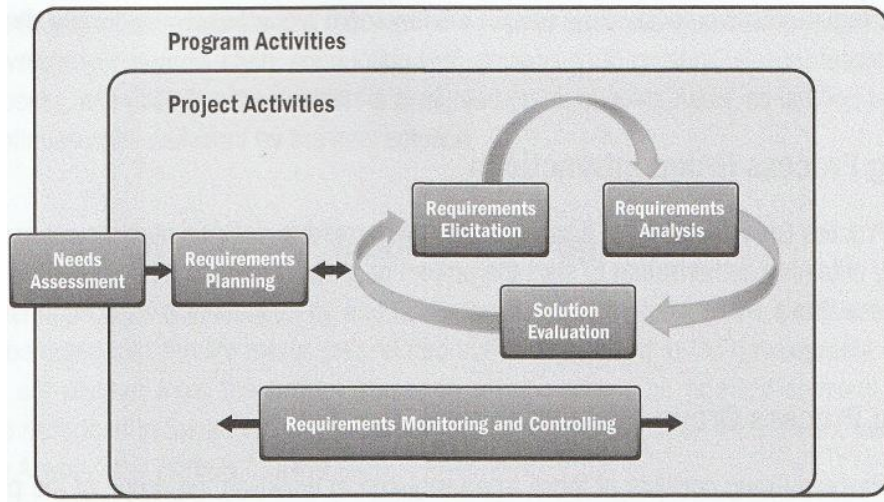


Figure 27. The PMI Requirements Process Model. (PMI, 2016)

This process model is overlaid onto a model for project management, showing that the two processes have a strong relationship (reference Figure 28).

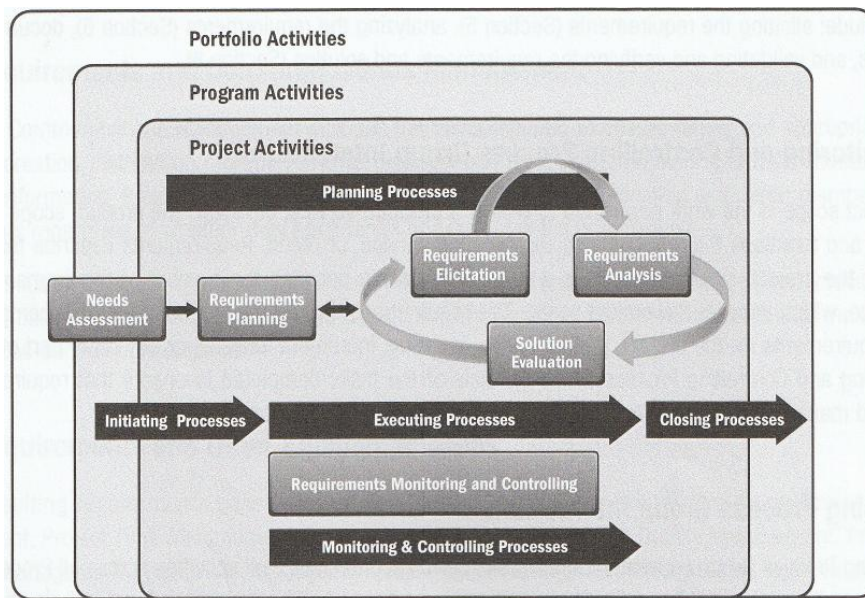


Figure 28. Mapping of the Requirements Process to the Project Management Process. (PMI, 2016)

The primary scope of requirements management process is captured in the "requirements monitoring and controlling activity" in the overall PMI process, which is shown in Figure 29.

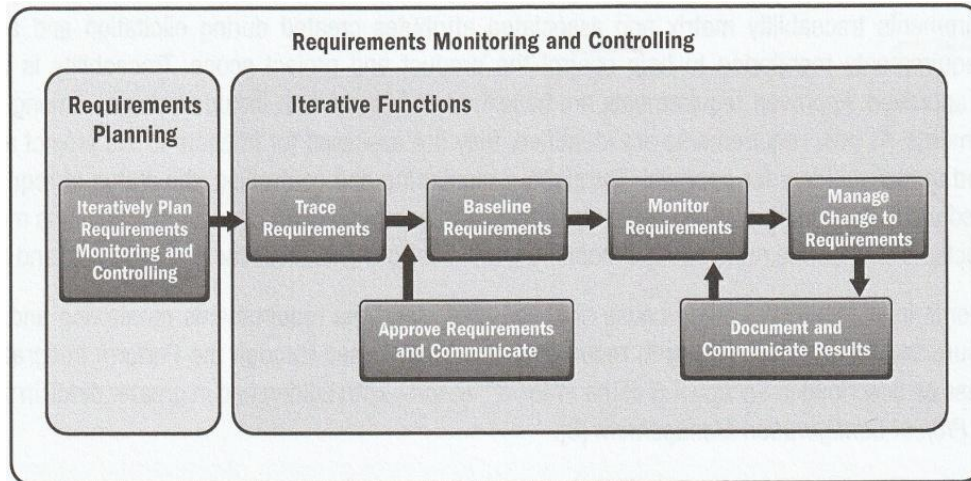


Figure 29. The PMI Requirements Monitoring and Controlling Process. (PMI, 2016)

PMI provides recommended practices for performing each step of the process, such as this recommendation related to trace requirements: "Organizations often trace their requirements using a structure called a traceability matrix. A traceability matrix is a grid that links product requirements from their origin to the deliverables that satisfy them. The implementation of a requirements traceability matrix supports the goal that each requirement adds business value by linking it to the business and project objectives. It provides a means to track requirements throughout the project life cycle, helping to ensure that approved requirements are delivered at the end of the project. The matrix also provides a structure for managing changes, thereby helping to manage the product scope." (PMI, 2016).

While the PMI provides a "practice guide", in actuality the information contained within is high level (such as that traceability example), omitting specific guidance and examples on how to generate the recommended artifacts.

Dick Requirements Management Process

Per Dick, et al, "products are becoming more complex to the point where no individual has the ability to comprehend the whole...structuring is by far the best way of organizing requirements, making them more manageable in terms of omissions or duplicate of information. Requirements management is about communication."

The process for requirements management identified by Dick includes the states of planning, monitoring and control of changes. A high level model of this is shown in Figure 30, which is reflective of the content within the publication.

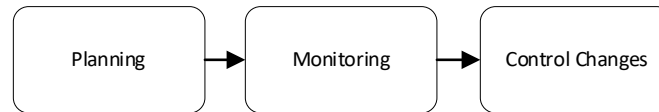


Figure 30. The Dick Requirements Management Process Model

The main steps within this process include the following:

- The *planning* stage includes the efforts that occur during the development of the requirements, such as addressing the stakeholders, identifying information to use as requirements attributes, establishing the trace information between the requirements and other sources of data, and development of criteria to address requirement information quality and completeness.
- The *monitoring* stage includes measurement of the requirements during the project life cycle against the plan, evaluating progression and completeness.
- The *control of changes* stage involves assessment of any requirement change on the project, utilizing data such as the requirements traceability and impacts to the product and any variant.

Several examples of performing these steps for different project types are provided, as well as examples of how to implement them in requirements management tools. The information notes that requirements management may vary depending on types of organization and projects, and specifically addresses the following examples:

- Acquisition organizations that purchase systems and then use them to provide an operational capability; requirements are developed by these organizations
- Supplier organizations that respond to acquisition requests for system development; requirements are received from a purchasing organization
- Product companies that develop and sell products to address a market need; requirements are developed by a marketing organization in the company

The approach used towards the requirements management activities is influenced based on the acquisition approach used by the product developer.

Hooks Requirements Management Process

Ivy Hooks is a renowned expert in requirements development and management, having worked across multiple industries and observed the costs of poor management and the benefits of effective management. Along with Kristin Farry, Ms. Hooks generated a book on requirements management in 2001, which describes effective techniques in development and management of requirements for projects to apply. Their recommended process model for the requirements development and management process is shown in Figure 31.

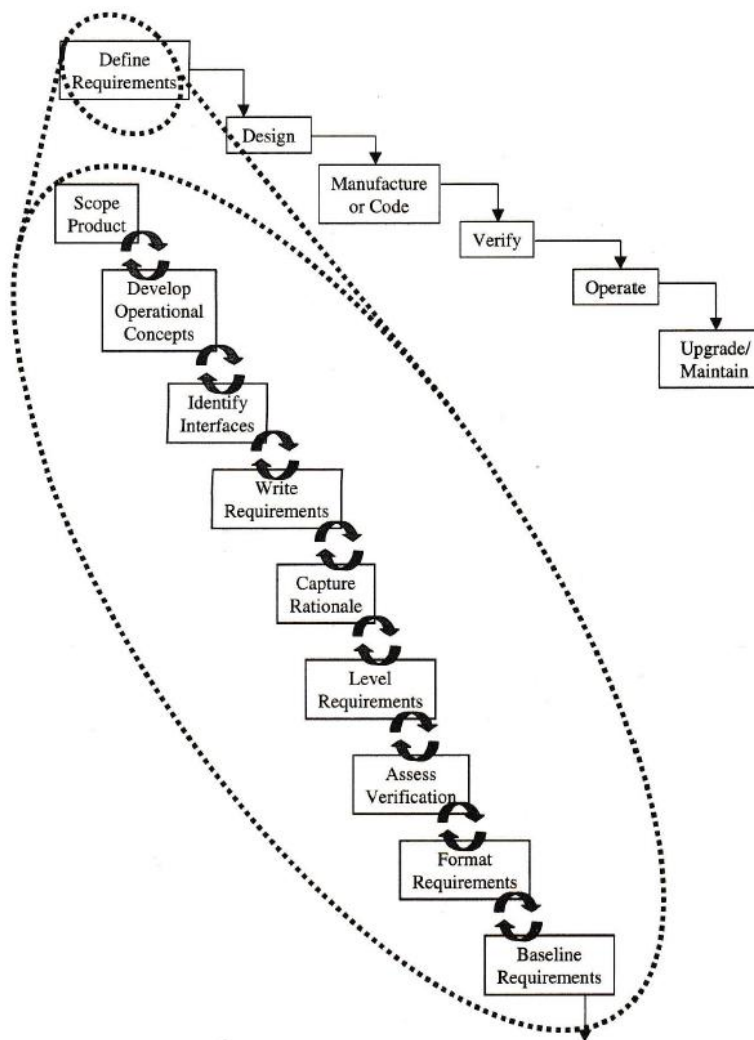


Figure 31. The Hooks Requirements Process Model. (Hooks & Farris, 2001)

The process steps in Figure 31 appear closer to a requirements development process as it ends at the baseline process (with an arrow showing there is further process steps afterwards). The management aspects are embedded within this and the book, and therefore the figure has been redrawn in

Figure 32 to display all of the process steps in the book for requirements management.

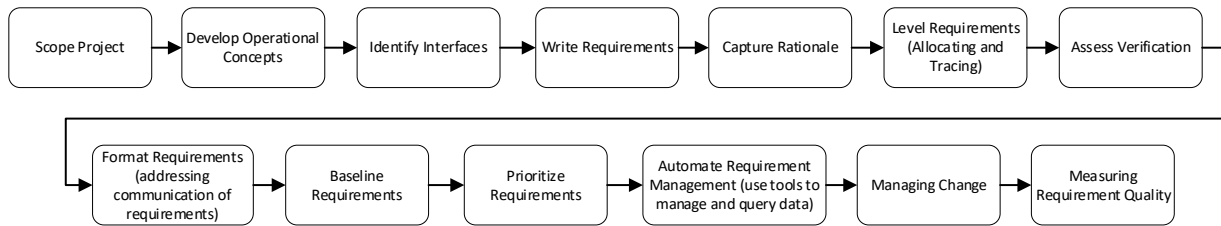


Figure 32. The Hooks Requirements Development and Management Process Model

Hooks additionally addresses the role of project and organization management responsibilities during the requirements engineering processes, stating how these influence the outcomes of the effort. While almost twenty years old, many of the recommendations within this book (Hooks and Farris, 2001) align with current best practice recommendations and process models from the other publications.

Kumar Requirements Management Process

Victoria Kumar presented a paper to the PMI Global Congress conference in 2006 to propose an “effective requirements management process”. Kumar highlighted the various causes of project problems associated with poor requirements development and management, and proposed a set of activities necessary to contribute to project success. Figure 33 provides the set of steps proposed, and the author notes that "An effective Requirements Management process must involve all four Requirements Processes defined above: Requirements Planning, Requirements Development, Requirements Verification, and Requirements Change Management." (Kumar, 2004). The paper provides high level guidance for considerations and inputs related to the activities required in the four noted processes.

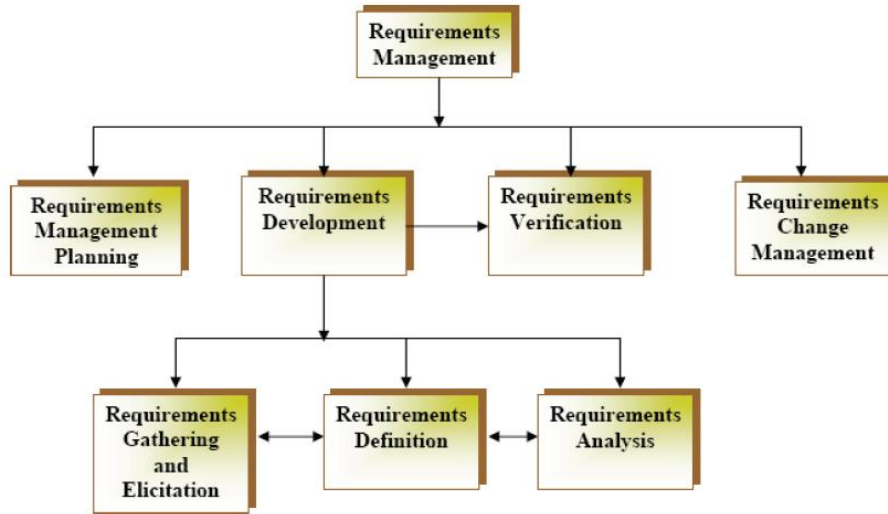


Figure 33. The Kumar Requirements Management Process Model. (Kumar, 2004)

Hood Requirements Management Process

Per Hood, et al, requirements management integrates overall project data with requirements data, creating a flow of information used by various project members. This includes overlaps of several areas of project efforts, including project management, quality management, risk management, as shown in Figure 34.

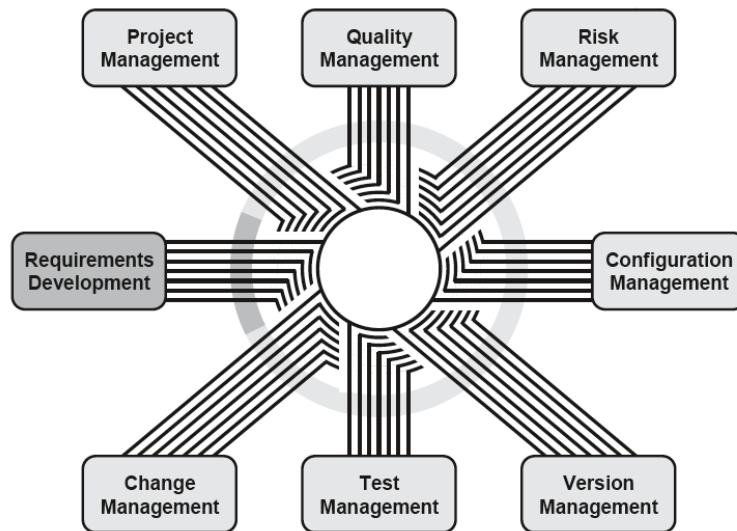


Figure 34. The Hood Requirements Management Interfaces. (Hood, Wiedemann, Fichtinger, & Pautz, 2008)

Similar to the *INCOSE Systems Engineering Handbook*, the Hood publication addresses requirements management as a theme that is interwoven into many different project processes, and describes how the results of requirements management enable the processes for project management, quality management, etc. Hood states that requirements development and communication consist of a series of exchanges occurring at different levels of the project, as represented in Figure 35.

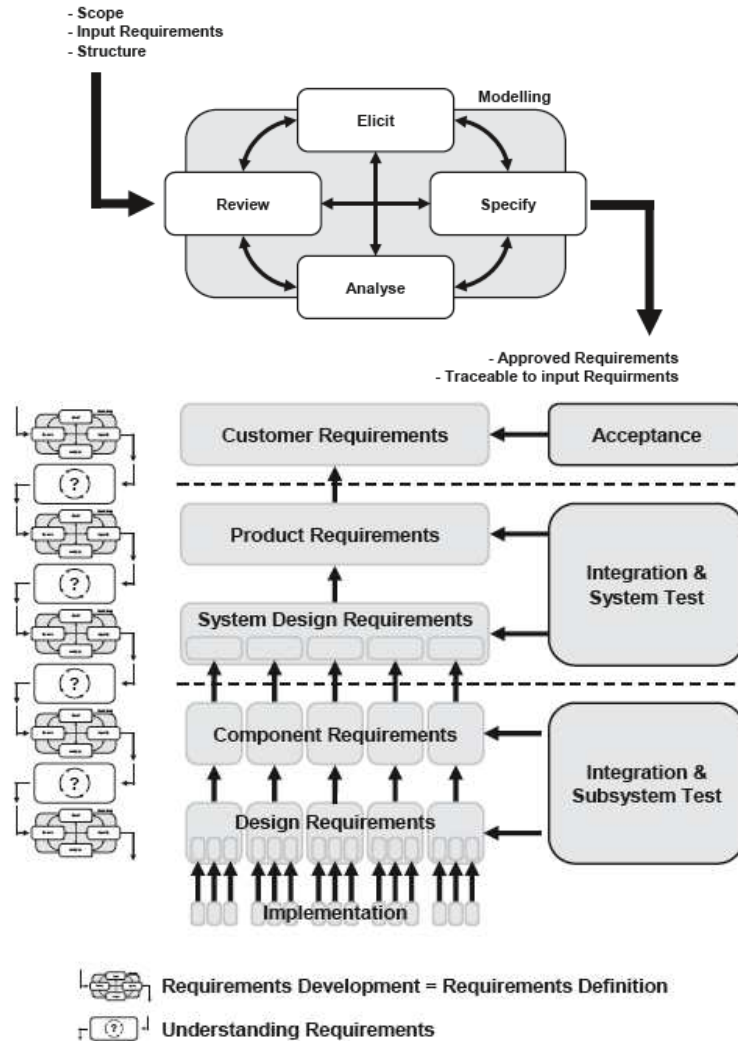


Figure 35. The Hood Requirements Processes. (Hood, Wiedemann, Fichtinger, & Pautz, 2008)

Per Hood et al., "Managing requirements consists of managing changes to requirements, managing various versions of requirements, managing multiple configurations of requirements, managing deliveries of requirements on time, in budget and to the correct quality without taking undue risks. A perfectly optimized system is a set of suboptimal subsystems. If teams try to optimize each subsystem

there will be conflict. Following the advice in this book teams will be inspired to see the big picture and be able to concentrate on getting the system built as required." (Hood, Wiedemann, Fichtinger, & Pautz, 2008).

While many recommendations for performing requirements management are provided, Hood does not provide an overall process model for the requirements management activities. After assessing the recommended steps, it is asserted that the Hood requirements management processes are addressed in the process models shown earlier. What is unique about Hood's approach is the concept of how requirements management is interwoven into project data and disciplines, and that these interfaces must be continuously addressed.

Current State Requirements Management Process Model

Going through an assessment of the various requirements management process models provided in this chapter, an overall **current state requirements management process model** is provided in Figure 36 that synthesizes the primary process steps from the prior models. This figure shows the various steps of the requirements management process for up to three levels of product structure (level n, n+1, and n+2), where the subsequent levels are a repeat of the activities, potentially across multiple organizations (reference Figure 15). This model is also overlaid onto a "V", reflecting how these steps correspond to the systems engineering Vee model shown earlier in Figure 5 and Figure 13.

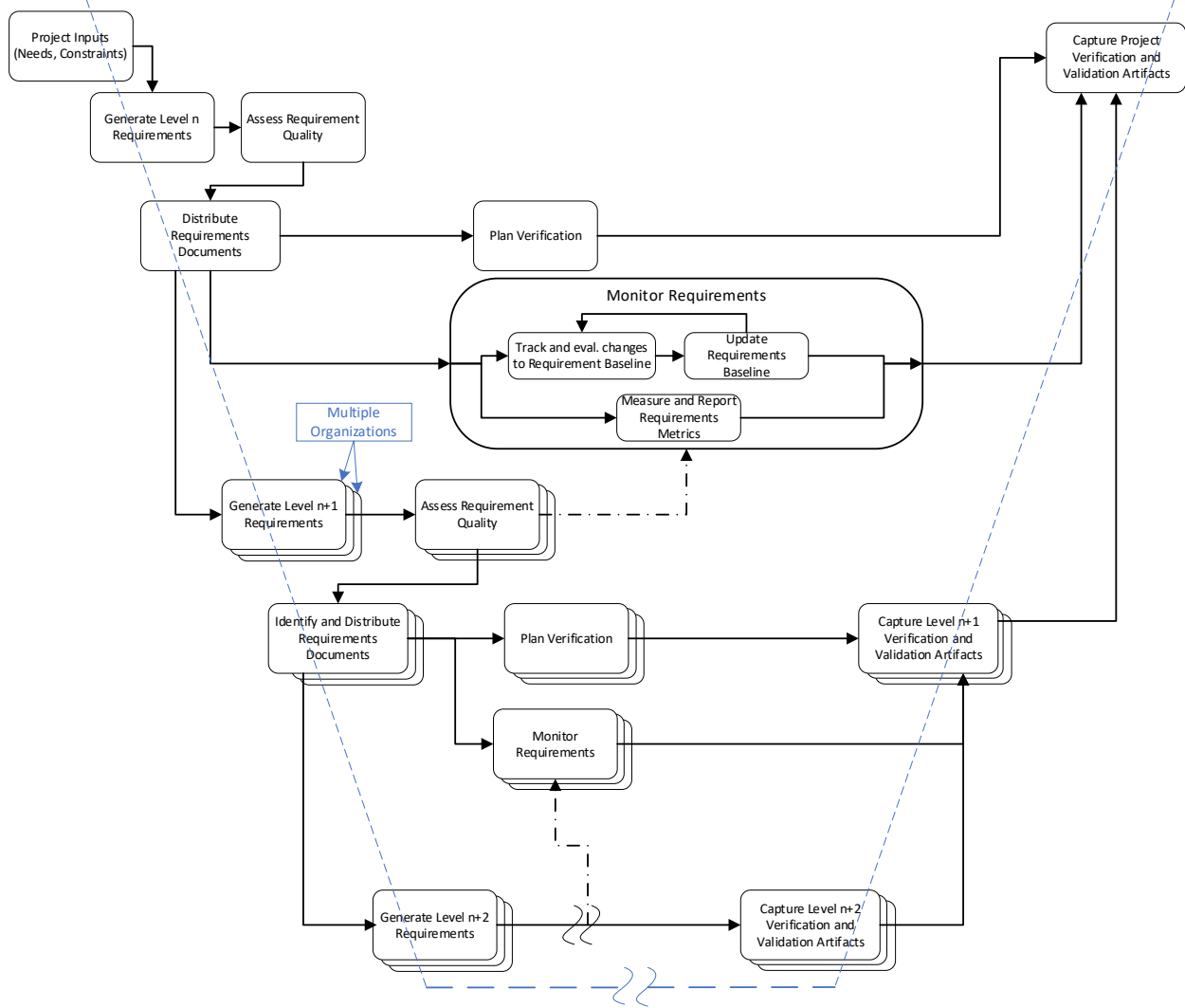


Figure 36. The Current State Requirements Management Process Model.

This model in Figure 36 reflects the “current” process for requirements management as described by numerous published sources on requirements engineering. Based on the product structure, this process is performed at multiple levels of the product assembly (as described in Section 2.2.4), where each level provides inputs impacting the levels above and below (per the levels of abstraction described previously in Figure 13).

3.1.3 Why Requirements Change on a Project

As noted in Section 2.1, a system is developed in progressive waves of maturity, resulting in iterative and recursive requirements development over a project life cycle. The concept of iteration among requirements resulting in the need for change was also described in Section 2.2.2.

Per Forsberg, et al., "The requirements management element is situational since new requirements can be introduced into the project at almost any time and decomposition level, to be managed concurrently with the maturing baseline." (Forsberg, Mooz, & Cotterman, 2005). This is further highlighted in Figure 37, which depicts that requirements change throughout various points in a project development as new information is obtained. This new information can take the form of analysis results from the design configuration and addressing inadequate system performance, as well as external factors such as changing operational environments, new technology, innovation, obsolescence, regulation updates, changes in the market, and emerging threats and risks.

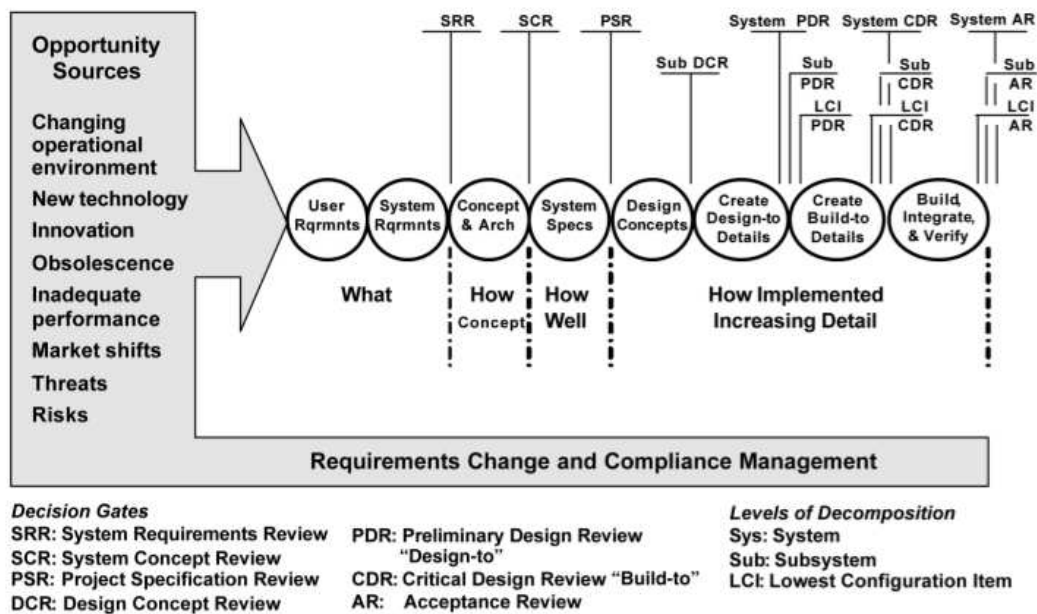


Figure 37. Evolution of Requirements Baselines Addressed by Requirements Management. (Forsberg, Mooz, & Cotterman, 2005)

At an early phase of a product's requirements development effort, the requirement set is controlled into a configuration managed "baseline". A baseline is an agreed-upon description of the

attributes of a product at a point in time, which serves as a basis for change. Upon approval, the technical baseline documentation is placed under formal configuration control, this often occurs at a designated time or project review early in the project life cycle. On many projects this review is called the Systems Requirements Review (SRR), which is shown in Figure 37 as the transition of establishing *what* will be done to the effort of *how* it will be done.

This initial baseline is not the finished and complete set of product requirements, it is an initial controlled configuration which allows product development to be based on a common framework of requirements. The development and refinement of requirements progresses as described earlier in Figure 13 (Requirement Evolution over the Systems Engineering Vee Model) and Figure 14 (Requirements Development Process with Change Feedback Loop), resulting in an updated set of requirements which are captured in revised baselines throughout the product development. This idea of evolving baselines is particularly consistent with Agile requirement development approaches (Agile Manifesto, 2001), where initial requirements are established and implemented in a prototype that is tested and analyzed, leading to information which is applied as updates to the requirements as the product development efforts continue.

As requirements change during the project, the concept of **change management** in requirements engineering takes on a high level of importance. Change management is the process by which the proposed requirement changes go through a defined impact assessment, review and approval process, using requirements tracing and version management (IEEE 29148, 2018).

If the change is not understood, if impacts to all areas that respond to the changing requirements are not captured, and if the change is not communicated, then the requirements will likely be misaligned at multiple portions of the project. Figure 38 highlights many of the areas that requirements affect during a project life cycle, and each of these need to be considered based on when a requirement changes, and what is actually changed.

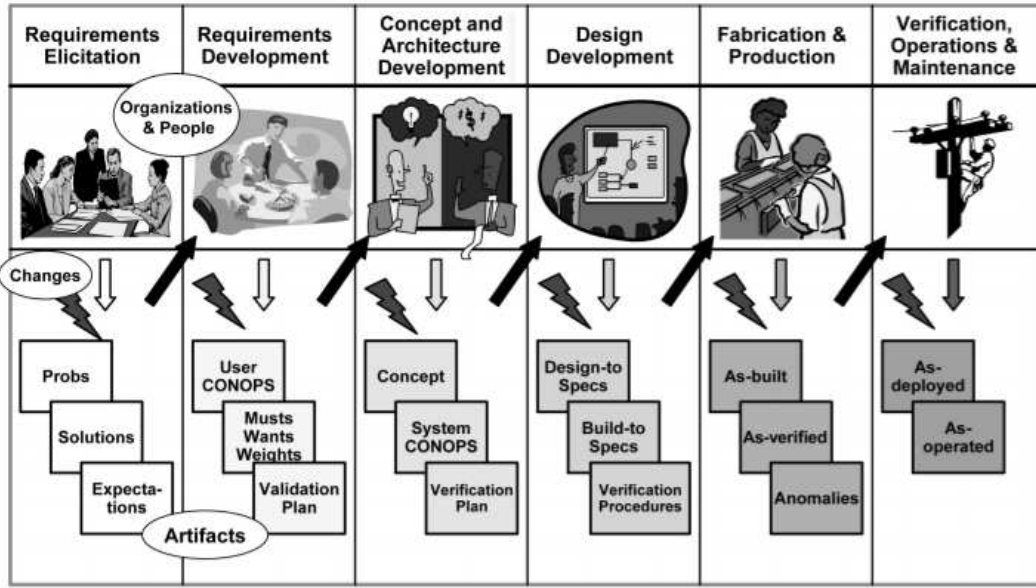


Figure 38. Requirements Management Evolution Over the Project Life Cycle. (Forsberg, Mooz, & Cotterman, 2005)

Per Forsberg, et al, "Requirements management and requirements management artifacts must be configured to ensure undistorted communication." (Forsberg, Mooz, & Cotterman, 2005), highlighting the need to establish a strong process for clear communication of requirements changes across all of the impacted artifacts and organizations. Figure 39 provides a model of how the product's original requirements evolve over its life cycle, showing proportions of new requirements, changed requirements, and deleted requirements at final product completion compared to the original requirements.

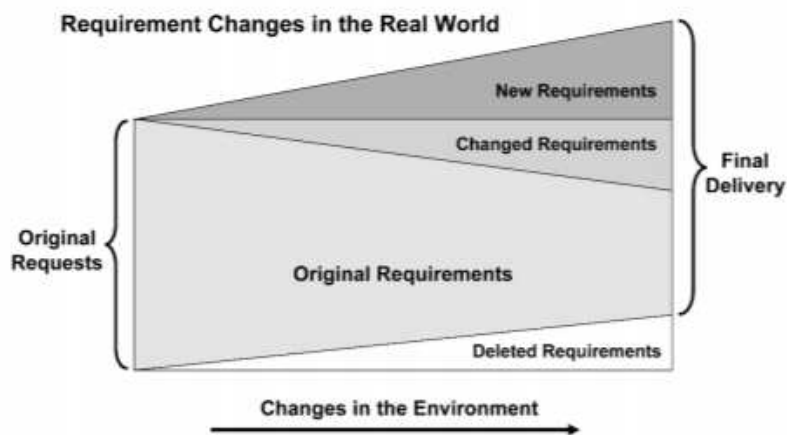


Figure 39. Requirement Changes Over the Project Life Cycle. (Forsberg, Mooz, & Cotterman, 2005)

It is interesting to observe that the requirement set at the final delivery in Figure 39 is larger than the original set. This growth of requirements over the course of a project is often referred to as **requirements creep**.

Per the *NASA Systems Engineering Handbook*, "Requirements creep is the term used to describe the subtle way that requirements grow imperceptibly during the course of a project. The tendency for the set of requirements is to relentlessly increase in size during the course of development, resulting in a system that is more expensive and complex than originally intended. Often the changes are quite innocent and what appear to be changes to a system are really enhancements in disguise. These new requirements are the result of evolution, and if we are to build a relevant system, we cannot ignore them." (NASA, 2020).

A complexity when addressing requirements change is the number of organizations involved in the system development (reference Section 2.2.3). When a requirement change occurs at the system level, it permeates down the supply chain through change orders. Based on allocation and impact of the changed requirement it may go all the way to component level, as shown in Figure 40. This set of "change orders" may occur with every change, and at different levels based on where a change can occur. This is one of the impacts that is assessed for acceptance of a change, as the **costs associated with these change orders can be a significant cost to the overall project**.

Figure 40 highlights areas of hidden costs based on how requirements are developed and implemented on a project, this will be expanded upon later in Chapter 5 as an opportunity for cost optimization associated with requirements management.

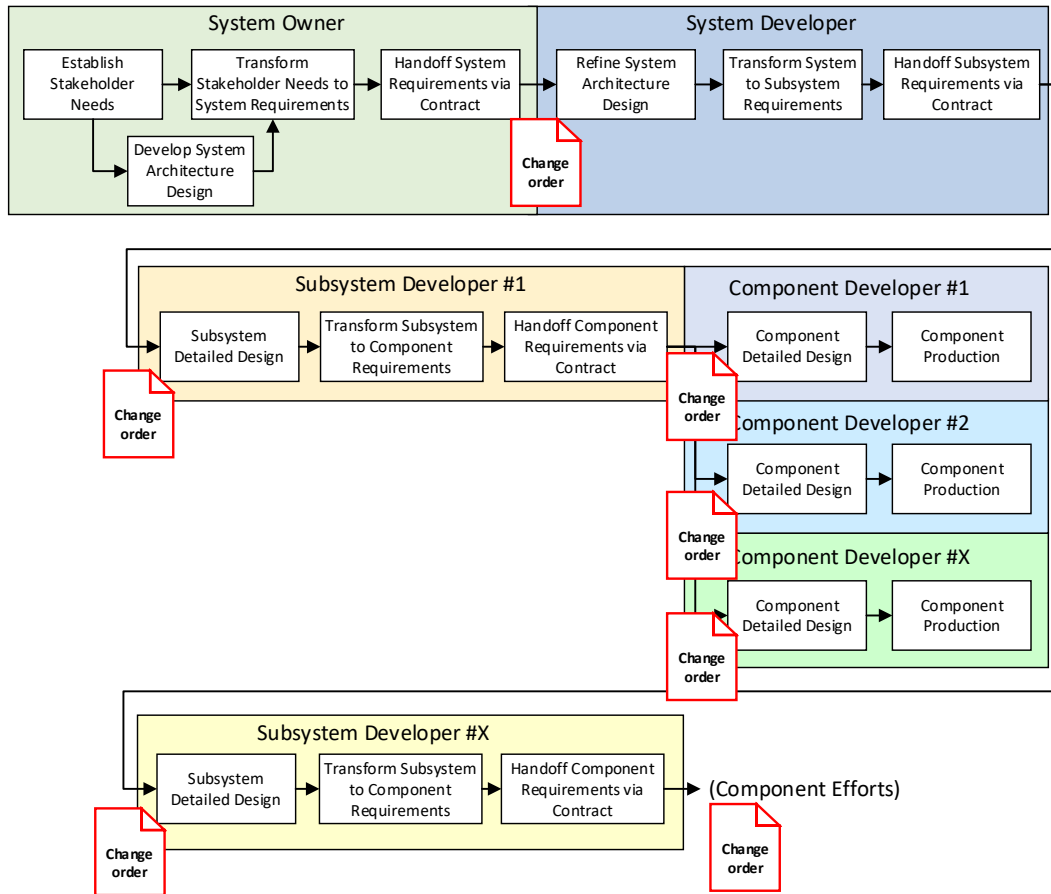


Figure 40. Impact of Requirement Changes with Multiple Development Organizations.

3.1.4 Requirement Stability Using TBX Management

As highlighted in the previous chapter, requirements evolve over the project life cycle. An initial set of requirements may reflect an accurate decomposition from the needs and can be refined for lower level products directly. Often, however, the initial information is a first estimate, which requires further work to refine. This is typical for many performance requirements where analysis is used to establish the final values and tolerance.

Undefined values in requirements are referred to as **To Be Determined (TBD)**. Requirements whose definition is approximate but not confirmed are referred to as **To Be Resolved (TBR)**. Requirements which contain TBD or TBR indications are often referred to as containing "TBX".

The need to address TBX early on a project is shown in Figure 41. The impacts of not resolving these unknowns can lead to a developed product that is not aligned with the stakeholder needs, or that is produced later than scheduled (often leading to an increase in cost).

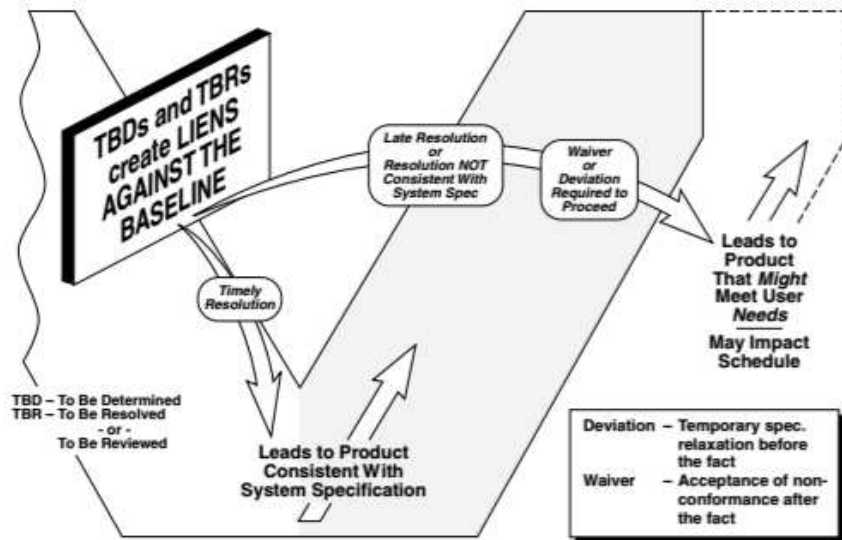


Figure 41. TBXs Represent Liens that Impact Projects if not Resolved. (Forsberg, Mooz, & Cotterman, 2005)

Per Forsberg, "Unresolved requirements should be viewed as liens against the baseline and must be resolved as early as possible to reduce programmatic risk. TBDs are a risk to the project since their impacts cannot be priced or scheduled. When the TBD is defined, it may have an impact that leads to contractual actions, such as an engineering change proposal, to adjust the contract baseline or a request for equitable adjustment. Usually, rough estimates of a TBR's impact can be made and accommodated in the contract baseline. There is always a risk that the resolution of a TBR may be beyond the schedule or cost baseline, resulting in a contract action to adjust the baseline. Formal work-off plans must be developed for both TBDs and TBRs, including "must have" delivery dates. Failure of the customer to deliver on these negotiated delivery dates may be grounds for contract-based constructive change claims, including compensation." (Forsberg, Mooz, & Cotterman, 2005).

Capturing and resolving the unknowns and approximations in requirements is referred to as **TBX management**, which is the process within requirements management that tracks all TBDs and TBRs,

assigns them to responsible individuals to resolve the unknowns, determine applicable updated requirement wording, and then apply the change management process for this update as described in Section 3.1.3. The purpose of using TBX indications is to allow for some effort of development to proceed at risk while the activities to resolve the full requirement is being performed; this is a risk based decision by the project to allow concurrent efforts.

An example of TBX usage within a requirement statement is shown with this sample requirement below:

PWR-1 Power Activation
 The power subsystem shall enable 28 Volts +/- 0.5 Volts (TBR) to heating element within TBD seconds of receiving the power enable signal.

Within a requirements specification, the TBX are typically captured in a summary table, resembling the sample in Table 6.

Table 6. Sample TBX Table.

TBX	Requirement	Topic	Assigned	Closure Plan
TBR-1	PWR-1 Power Activation	Voltage range needs confirmation	Person X	Power analysis in work, to provide final voltage range at end of analysis cycle 1.
TBD-2	PWR-1 Power Activation	Time required for power activation after command is still being determined	Person Y	Use case analysis is in work, expect resolution after timing analysis at end of analysis cycle 1.

Typically the TBX for a project will be summarized into an overall list for the project, assigned to personnel to resolve, captured in a project schedule for a resolution date, and managed as a set of reportable project metrics (number of TBXs, burn down graph of resolution, etc.).

The number of TBD and TBRs in a requirement set is an indication of a product's requirement stability, where requirements with no TBXs are considered more stable. This does not mean that the requirements will not change after TBX resolution, however, as changes could still be driven by external sources (as shown in Section 3.1.3). The requirements stability during the existence of TBXs is a "known

unknown" situation, where any changes after resolution is considered an "unknown unknown" situation. Impacts of requirements instability and methods to address is further described in Section 5.6.

3.1.5 Requirements Management for Product Lines

A consideration in determining an approach to requirements management is whether the product being developed is part of an overall product line. When managing singular products an organization can use project-customized approaches and techniques, as all requirements trace to a set of stakeholder requirements for that project only.

When developing a product line, or an element within a product line, the requirements management process will need to introduce an ability to manage project variant requirements into the process, and consider interfaces between projects in support of an overall product line requirements management process. Individual variants of a product may have common requirements that exist for all products in the product line, as well as unique requirements that exist only for that variant (as shown in Figure 42).

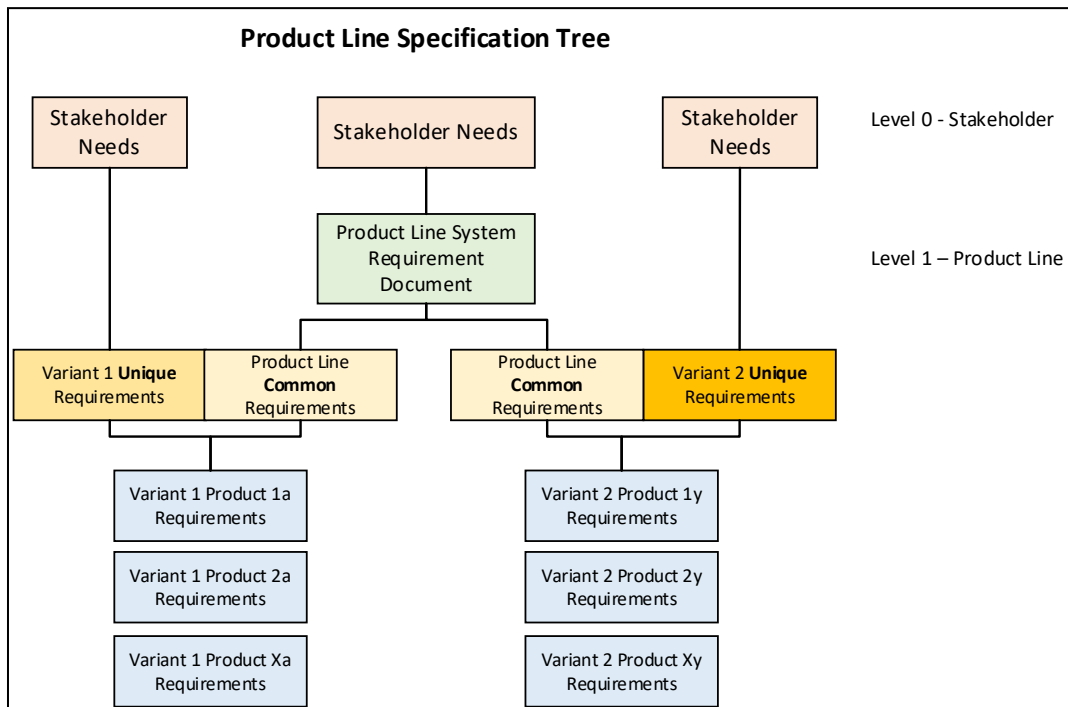


Figure 42. Example of a Product Line Specification Tree showing Inter-relationships among the Product Requirements.

For development of requirements within a product line the source of needs includes both the overall product line needs from the organization as well as needs associated with each variant, essentially linking all product variant requirements development. An example would be for development of an automobile product where one variant is a front wheel drive two door vehicle, and the other variant is an all wheel drive four-door vehicle; the company will ensure the common product line features are maintained in each variant as well as address the unique aspects of the user community that is reflected in the individual variants of the automobile versions.

Requirements management in this type of effort will need to address traceability of each type of requirement, and ensure that changes are assessed against impact to the variant as well as the overall product line. Additionally, any changes of the product line requirements will need to consider impacts to all of the variants, or allow for tailored requirements to be created for the variant products that will keep the original requirements.

3.1.6 Ensuring Requirements Quality through Requirements Management

How much of a contributing factor is the process of requirements management in ensuring quality of the product's set of requirements? As shown in Figure 7 (Overview of the Various Requirements Engineering), the requirements development process addresses the elicitation, analysis, and verification and validation of the product requirements. The processes behind these include the assessment of requirement alignment with the stakeholder needs and parent requirements, as well as conformance to requirement quality standards. However, the best requirements management process will still yield poor results if the requirements are of poor quality (i.e. garbage in, garbage out). The requirements management process will often include a forcing function to ensure that it enables the product development success by providing a process to perform a check on the quality of the data it is addressing.

Per Pohl, the requirements management process includes an evaluation of the requirement artifacts, which is a check of the requirement quality to ensure it meets the intent of the entire requirements engineering effort (Pohl, 2010). Just as the configuration management process will include a review of the products being configuration controlled, the requirements management process includes a

review of the requirement quality before being accepted by the project, and is highlighted in the requirements management model shown earlier (Figure 36). While the specific activities to produce requirement quality are part of the requirements development process, the requirements management process ensures these activities are done and tracked by the project.

When addressing requirements quality a question often asked is related to completeness; when is the requirements development work complete? Some projects will utilize a check against the stakeholder needs and example industry requirements to ensure the requirement set being developed addresses key topic areas, and that the development process has considered everything needed to ensure the realized system meets its needs and will have the intended capabilities and functions.

Per the *INCOSE Guide for Writing Requirements*, there are characteristics that can be assessed to determine if a requirement statement, conforms to quality standards. Several characteristics for requirement statement quality recommended in the INCOSE guide are shown in Table 7, and characteristics of a quality set of requirements were shown previously in Table 3.

Table 7. Characteristics of Quality Requirements. (INCOSE, 2019)

Characteristics of Well-Formed Requirement Statements	
<p><i>Formal Transformation Quality Characteristics:</i></p> <ul style="list-style-type: none"> • <i>C1 - Necessary:</i> The need or requirement defines an essential capability, characteristic, constraint, or quality factor needed to satisfy a concept, need or parent requirement. • <i>C2 - Appropriate:</i> The specific intent and amount of detail of the need or requirement is appropriate to the level of the entity to which it refers. • <i>C5 - Singular:</i> The stakeholder need or requirement statement should state a single capability, characteristic, constraint, or quality factor. • <i>C8 - Correct:</i> The need must be an accurate representation of the concept from which it was transformed. A requirement must be an accurate representation of the need from which it was transformed. • <i>C9 - Conforming:</i> The individual needs and requirements should conform to an approved standard pattern and style guide or standard for writing and managing needs and requirements. 	<p><i>Agreed-To-Obligation Characteristics:</i></p> <ul style="list-style-type: none"> • <i>C3 - Unambiguous:</i> Need statements must be written such that the stakeholder intent is clear. A requirement is stated in such a way that it can be interpreted in only one way by all the intended readers. • <i>C4 - Complete:</i> The requirement sufficiently describes the necessary capability, characteristic, constraint, or quality factor to meet the entity need without needing other information to understand the requirement. • <i>C6 - Feasible:</i> The need or requirement can be realized within entity constraints (for example: cost, schedule, technical, legal, ethical, safety) with acceptable risk. • <i>C7 - Verifiable:</i> The requirement is structured and worded such that its realization can be proven (verified) to the customer's satisfaction at the level the requirement exists.

These characteristics are used during the review of the requirements to assess for completeness and quality.

3.1.7 Data Attributes for Requirements Management

The requirements management process utilizes data as a way to assess requirements progress, stability, quality, and source, this data is referred to as **requirement attributes**.

Per the *INCOSE Guide for Writing Requirements*, an attribute is additional information included with a need or requirement statement which is used to aid in the management of that need or requirement. There are several requirement attributes recommended in the INCOSE guide that are key to help maintain a high quality set of requirements, a sample from GfWR is shown in Table 8.

Table 8. Sample Attributes of Requirements Statements. (INCOSE, 2019)

Attributes to Help Define the Requirement and its Intent	
A1 - Rationale*	A22 - Approval Date
A2 - System of Interest (SOI) Primary Verification or Validation Method*	A23 - Date of Last Change
A3 - SOI Verification or Validation Approach	A24 - Stability
A4 - Trace to Parent*	A25 - Responsible Person
A5 - Trace to Source*	A26 - Need or Requirement Verification Status*
A6 - Condition of Use	A27 - Need or Requirement Validation Status*
A7 - States and Modes	A28 - Status (of the Need or Requirement)
A8 - Allocation*	A29 - Status (of Implementation)
	A30 - Trace To Interface Definition
	A31 - Trace To Peer Requirements

* recommended minimum attributes

While not all of the attributes are needed for every project, the INCOSE guide provides recommendations on a minimum set, as well as how to apply them and considerations of use. Ultimately, these attributes provide enabling capabilities for managing requirements on a project and allow for situational awareness of the requirements completeness and stability.

3.1.8 Conclusion on Current Approaches to Requirements Management

This section addressed the definition of requirements management, the various process models that exist, and the impact of poor implementation. Some observations from this research include:

- Requirements management addresses management of the information and data associated with the project requirements;
- There exists extensively more documented processes for requirements development than requirements management;
- Requirements management includes the ability to configuration control a set of requirements and address impacts of change or stability, which can affect a project's cost and schedule;

- Requirements can be used across multiple products in a product line, requiring approaches that consider data management on more than one project; and
- In assessing various requirements management process models there does not seem to exist a single model that captures the entire approach for a project in how to perform the various requirements management activities.

To address the evolving work being done in this field, the next section will address the current work being done, including process updates, trends in the various requirements management tools, and observations from various subject matter experts.

3.2 Newer Requirements Management Trends

3.2.1 Trends Based on the INCOSE Requirements Working Group Efforts

The INCOSE Requirements Working Group (RWG) is a volunteer organization within INCOSE dedicated to advance the practices, education and theory of needs and requirements definition and management and their relationship to other systems engineering functions. This group is comprised of INCOSE members with backgrounds in requirements engineering in various industries and academia. In addition to discussions on various topics within requirements engineering, this group is responsible for development of guidance and methodology that various practitioners can reference when working to apply requirements engineering effort in their industry.

This organization has created and published the *INCOSE Guide for Writing Requirements*, and recently published a white paper on *Integrated-data as a Foundation of Systems Engineering*. Currently, the RWG is developing three new products for INCOSE: *Needs and Requirements Lifecycle Manual*, *Guide to Needs and Requirements Development and Management*, and the *Guide to Verification and Validation*. All of these products are generated to reflect best practices for systems engineering and product development throughout the product life cycle; this author is a contributor to the new RWG products, leveraging some of the research being done for this dissertation.

Two of the former chairs of the RWG have also generated papers proposing an Information-based Requirement Development and Management (I-RDM) approach to developing and managing requirements, asserting that requirements should not be developed and managed separate from other

system data and information model development and management activities (Wheatcraft, Ryan, Dick, & Llorens, 2019).

A theme found in these RWG efforts is the idea that requirements are a form of program data, much like design data, analysis data, and behavior data; and these sets of data will need to be managed from a system level perspective. The paper's recommendation is to move beyond requirements management processes and tools that focus on requirements only, and use an approach that allows "the organization to develop and manage needs and requirements in relation to all the SE artifacts developed across all system development life cycle activities." (Wheatcraft, Ryan, Dick, & Llorens, 2019).

Figure 43 provides a view of the I-RDM approach, showing that this allows an ability to create a data and information model that define, establish, and document the artifacts generated during the conceptual design phase as well as relationships between these artifacts.

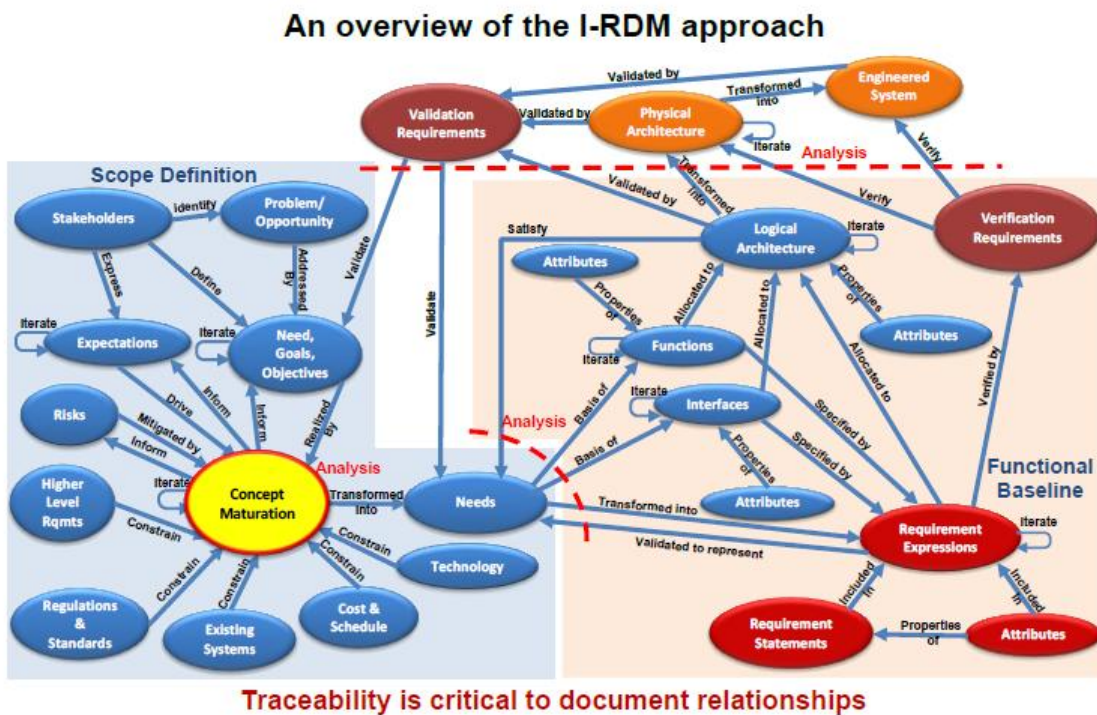


Figure 43. Overview of Information-based Requirement Development and Management.
(Wheatcraft, Ryan, Dick, & Llorens, 2019)

Figure 26, shown in an earlier chapter, highlights how the combination of an I-RDM approach (showing the system design inputs) with a model based design (MBD) approach (showing the system

design output), contributes to an overall view of the system data from a model-based perspective. This is a fundamental aspect of the Model Based Systems Engineering (MBSE) concept. MBSE is an evolving practice in systems engineering that moves away from a document and compartmentalized approach in generation of product development towards an approach using connected data via a set of modeling tools that enable product development. MBSE is discussed a bit further in Section 6.1, and while considered a newer trend in systems engineering, it is quickly becoming adopted as a standard of practice.

Among the new products being developed by the RWG, it is recognized that a project's focus of *requirements* compared to understanding and addressing the overall set of *needs* has lessened the ability to ensure the system will meet its intended purpose; two focus areas are being developed with respect to this concern. The first process update is a larger focus on the overall set of needs, ensuring the needs are adequately defined and transformed to a set of requirements. Another update is related to the processes of verification and validation (often called V&V), bringing larger focus to validation to show the product ultimately addresses the stakeholder needs.

Validation is defined as the evidence of addressing a set of needs, compared to **verification**, which is defined as evidence of conformance to requirements or standards. These terms are currently described in the *INCOSE Guide for Writing Requirements* relative to verification and validation of the needs, requirements, design and system.

As of the time of this writing the RWG is generating a *Needs and Requirements Lifecycle Manual*, expected to be released in the summer of 2021, addressing methodology, processes, and ontology to establish the set of system needs, transform them to requirements, and perform system verification and validation.

The RWG is also concurrently developing two associated guidelines intended to aid practitioners in applying the techniques and approaches. The generation of new guides by the RWG is a recognition that there exists documented information on *what* needs to be done, but limited information as to *how*. This idea formed the objective of this dissertation, and is also the basis of the dissertation of Dr. Karla Gomez Sotelo on *Quality Assurance Methodology for System Requirement Definition*, which addresses

how to perform requirements development from a set of needs for general product development (Gomez Sotelo, 2019). Much like these dissertations and the INCOSE efforts, an emerging trend is to show *how* to apply the concepts of requirements engineering in a way that practitioners can understand and use.

A summary of the requirements engineering trends discussed in this section are as follows:

- There is now a more "data focused" view of requirements as part of a larger set of system data;
- There is now also a larger focus on the overall needs driving the system development, compared to only attending to the product design requirements; and
- There is a stronger need to provide examples of approaches showing how organizations could apply the recommended processes for requirements development and management.

3.2.2 Trends Based on Author Published Research Efforts

In the spirit of addressing current practices in the space industry for methods of approach, this author has generated two INCOSE published papers addressing challenges in requirements engineering, offering some practical solutions to existing challenges. These publications and research align with the trend of addressing practical approaches on how organizations could apply the recommended processes for requirements development and management.

In a paper on evaluation of commercial-off-the-shelf (COTS) hardware assemblies, the processes for evaluation of existing products against specific project needs is provided, with considerations of the risk and budget posture of the system being developed (Katz, 2019). The paper addresses how to apply an assessment in situations where a lower level assembly is being used that had not been developed from requirements specific to a project; the result is an assessment for COTS assembly usage to show that it meets the intent, or is equivalent to, the requirements for the project, enabling the assessment to be used for proof of verification against the project requirements. The challenge being addressed in this paper is how a project could find an alternate approach towards implementing project requirements and still ensure the stakeholder's needs are met when using an already developed component.

In a paper related to design and construction standards, the processes of mandating standards was evaluated, with a recommendation that "usage of standards is not a one size fits all approach, and alternate

strategies exist for industries in cases where limiting the design solution could impact the ability to realize cost effective, innovative designs." (Katz, 2020). This publication addresses the hidden costs associated with usage of design and construction standards (a form of quality requirement), and considerations of whether to levy standards versus crafting specific requirements for the component developers. The challenge being addressed in this paper relates to communication of requirements to product developers in a manner that enables desired innovation and also ensures the project can stay within the cost expectations of the product being developed.

3.2.3 Trends in Various Industries

Many product development organizations across various industries utilize requirements management, the current practices of several of these have been reviewed to see if there are trends in the different industry approaches. This dissertation is specifically addressing optimization of the space industry requirements management based on complexity factors provided later in Chapter 4. Prior to addressing this specific industry, a look at multiple industries is provided below to enable comparison of approach.

Space Industry

This dissertation provides recommended approaches for management of requirements in the space industry, and the unique aspects of this industry are discussed in Chapter 4. In general, the space industry addresses complex systems with numerous requirements designated for specific operational use cases. Not all space products are classified as safety-critical (with respect to safety of people, only a small percentage of space products address human spaceflight). The need to ensure quality of product in this industry is high due to limitations of maintenance once in operation. Typical customers are risk averse and expect requirements quality with associated metrics such as traceability, stability, and verification.

This industry typically utilizes requirements management tools (reference Section 3.2.4) as well as dedicated staff to manage the requirements. Trends go towards more collaborative requirements

management with a view of requirements as a source of system data, as well as an increased use of model based systems engineering (MBSE) approaches in product development, aligning requirements with architecture, behavioral and analytical models.

Aviation Industry

Much like the space industry, the aviation industry develops complex products. These are often developed by companies that generate product lines to sell to transportation companies, government organizations, or private parties. While custom orders may occur, many of these are established products developed to meet aircraft regulations set by the Federal Aviation Administration (FAA). These products are almost always considered safety-critical, and they are capable of being maintained once in operation.

Many of the requirements management practices are comparable between the space industry and aviation industries. Of the trends, primarily the largest may be the focus on product lines and software-intensive systems. The FAA recently produced a requirements development and management handbook to guide aircraft developers in the complexities associated with software (FAA, 2009), addressing the considerations of use-cases and unanticipated usage of the aircraft.

Several of the larger aviation product developers have also started to utilize MBSE as part of their development efforts, such as Boeing Corporation development of the Air Data Reference Function for the 777X, where they are able to demonstrate faster development of complex systems with more accurate defined interfaces (Boeing, 2017).

Oil and Gas Industry

The oil and gas industry addresses upstream production (identify, extract, or produce raw material) and downstream production (refinement and delivery to the consumer). Therefore the type of equipment, product, and operations vary depending where in the production supply chain the product usage will occur. Regulations on these products do not exist at the national level, and vary by location of product development and usage.

Traditionally, oil and gas product development has been handled as a transactional effort, with development as part of a contract. However, many companies are starting to adopt more formal methods of product development and validation to ensure they meet safety regulations and contractual compliance, as well as control project costs and scope.

As noted at the INCOSE International Symposium 2020, "an international survey confirms that the oil and gas industry is lagging behind other industries when it comes to implementation of systematic requirement handling. There is a clear positive effect when implementing a requirement management system at a single supplier. By implementing such a system, the industry will limit the number of requirements, save cost, and reduce the need for testing and validation" (Helle, Engen, & Falk, 2020).

The trend in the oil and gas industry is to start adopting a formal requirements management approach and usage of management tools, and is still early in its adoption towards robust requirements management efforts.

Medical Device Industry

In the medical device industry, companies will develop a product for a consumer market that can meet government regulations such as those provided by the Food and Drug Administration (FDA). The regulations address the product development processes, requiring various documentation showing the strong control of requirements, verification, and configuration management. Medical devices are considered safety-critical, and the ability to show off safety, reliability, security, and quality control in the product development impacts success (marketing, market capitalization, positive minus negative review gap, etc.) for these companies.

Trends in this industry include optimizing the processes of the hardware and software development for a collaborative approach, ensuring the safety features and regulations are addressed by the requirements with traceability, providing the required artifacts associated with the requirements, and utilizing requirement management tools to enable these efforts (Modern Requirements, 2020).

Automotive Industry

Similar to the medical device and aviation industries, the automotive industry produces safety-critical products as part of an overall product line for a consumer market. This industry is also regulated by the National Highway Traffic Safety Administration (NHTSA).

The trends for requirements management with automotive original equipment manufacturers (OEM) is the focus on product lines and software-intensive systems, as well as an inclusion of costs associated with the requirements for impact assessments and trades. The industry makes use of process-engineering tools to support the decision process and potential variants are modeled to see how decisions will affect the product line and the company return on investment (Gulke, Rumpe, Jasen, & Axmann, 2012).

Summary of Industry Requirements Management Observations

A common theme among the described industries includes higher rigor in requirements management for more regulated industries, as well as for those considered safety-critical. With the exception of the space industry, each of the industries include considerations for product maintainability within their development; this poses a challenge for the space industry which must compensate for this by increasing the reliability and confidence level of their products. This, along with several other factors which make requirements engineering more challenging for the space industry, will be described further in Section 4.1.1.

3.2.4 Requirements Management (RM) Tools

Over the years multiple software applications have been developed to address management of requirements; a requirements management tool can enable a project's success with its execution and validation of end product. This chapter will provide information on the most commonly used applications and discuss the trends in the requirements management tool industry. Note that, like many software applications, these tools evolve rapidly; the intent of this chapter is not to promote any particular tool set,

it is to highlight the various capabilities and features that impact a project's implementation of their requirements management processes.

As the number of software applications has been growing over the past few years, the INCOSE organization has partnered with Project Performance International (PPI) to develop an online, searchable systems engineering tools database, <https://www.incose.org/setdbtest/system-engineering-tools-database>. The tools discussed below are the most commonly used, but not an exhaustive list by any means. If the reader is looking to obtain improvements in their requirements management efforts, they are encouraged to assess additional options through the INCOSE database, or other searches, to ensure they find the latest applications available that may suit their particular needs; recent evaluations and ranking of various requirements management tools is provided at the end of the section.

Microsoft Products (MS Word, MS Excel)

One method to manage a set of requirements is through a Microsoft Office product such as MS Word or MS Excel (<https://www.office.com/>). Many projects use these to capture their requirements into tabular or specification form, with no other application utilized.

For simple projects, this tool may be adequate to address the requirements management needs. It allows for an electronic capture, can be shared with others, and by using a product data management system can be released as a controlled document. Both applications allow for some form of change tracking (MS Word is better at this than MS Excel), and both allow for attributes of requirement information to be displayed with the requirement (Figure 44). This type of tool is considered a "document centric" approach towards requirements management as it addresses the requirement data within a set of individual document files.

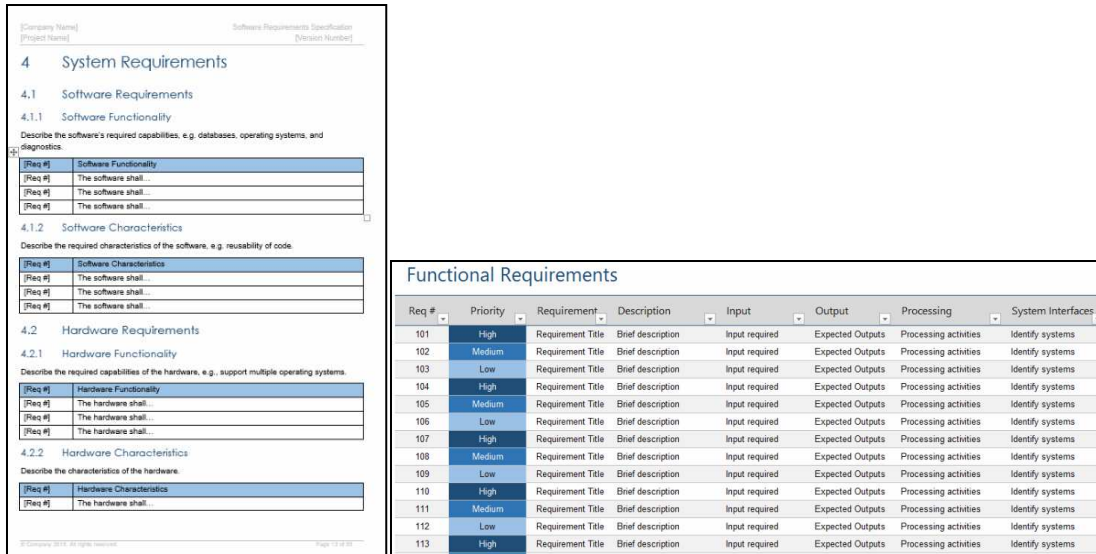


Figure 44. Examples of Requirements in Microsoft Products. (Klariti, 2020)

Why is this tool not sufficient for some projects? Primarily due to its lack of ability to capture traceability to other artifacts and requirements. Any trace is captured manually (by reference or hyperlink), and the reader is required to navigate multiple documents to understand how the requirements relate to other data sets.

This software application also lends itself to data integrity concerns as the requirements are disconnected from each other, raising the opportunity for errors as they are disseminated across multiple locations. A parent requirement may be missing properly allocated children requirements in lower specifications, values may be misaligned, or duplication of requirements may occur. Many projects start to see a break down in requirements management efforts using Microsoft products as their primary tool when they reach a large number of requirements, and then opt to use a relational database. The next few sections describe various relational database tools commonly used in requirements management.

IBM Rational DOORS

One of the prominent software tools for requirements management for many years has been IBM Rational Data Object-Oriented Requirements System®, or DOORS. DOORS was created in the 1990s as a software client relational database. Reference Figure 45 and Figure 46 for a sample view of a DOORS requirement database and module.

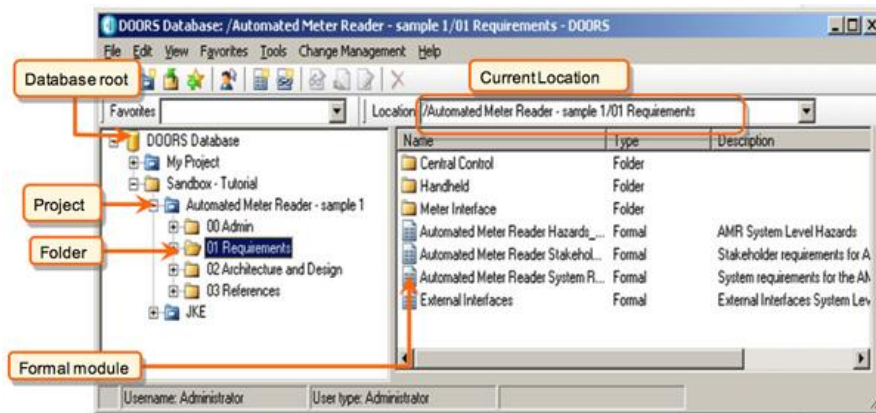


Figure 45. Requirement Database in IBM Rational DOORS. (IBM, 2013)

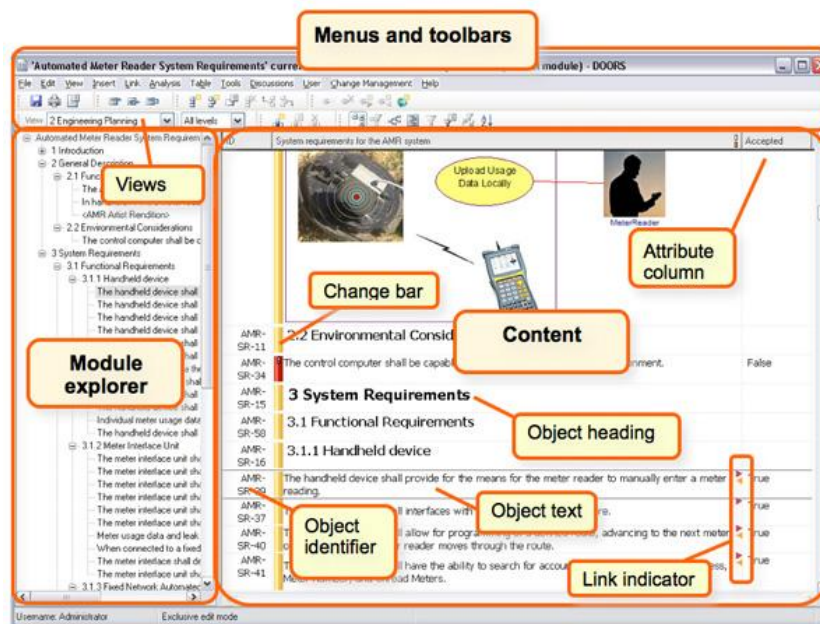


Figure 46. Requirement Model in IBM Rational DOORS. (IBM, 2013)

DOORS provides access to editing, configuration, analysis and reporting capabilities through a desktop client (web option available with limited functionality). Additionally, it supports the Requirements Interchange Format (REQIF), enabling suppliers and development partners to exchange DOORS data files and import into their software application. DOORS captures requirements text, graphics, tables and supports creation of attributes for requirements. It also relates requirements (or other objects) through linking and offers full traceability reports. Additionally, it supports external links that enable requirements to be directly associated with information outside of the DOORS application.

DOORS supports change management with either a simple pre-defined change proposal system or a more thorough, customizable change control workflow. Each object in a DOORS module is associated to that module file within a project data set, and each has a unique identifier (ID) for that module (requirements in a project in different modules could have the same ID, for this reason a prefix is often added to differentiate the different requirements within a product requirement set).

DOORS has advantages over document centric requirements management in that it allows for the requirements to trace to other requirements, supporting assessment of requirement allocation and completeness (reference Figure 47 for a linking example). DOORS modules can be published into multiple formats (Microsoft products, PDF, html) for configuration control in a product data management system, and various views of the data (such as traceability views) can be exported to show artifacts such as a requirement trace matrix (RTM), or a verification cross-reference matrix (VCRM). Additionally, DOORS attributes can be highly customized to reflect controlled data from a list, queried data from links, or typed data from a user (reference Figure 48 for an example of attributes shown with requirements).

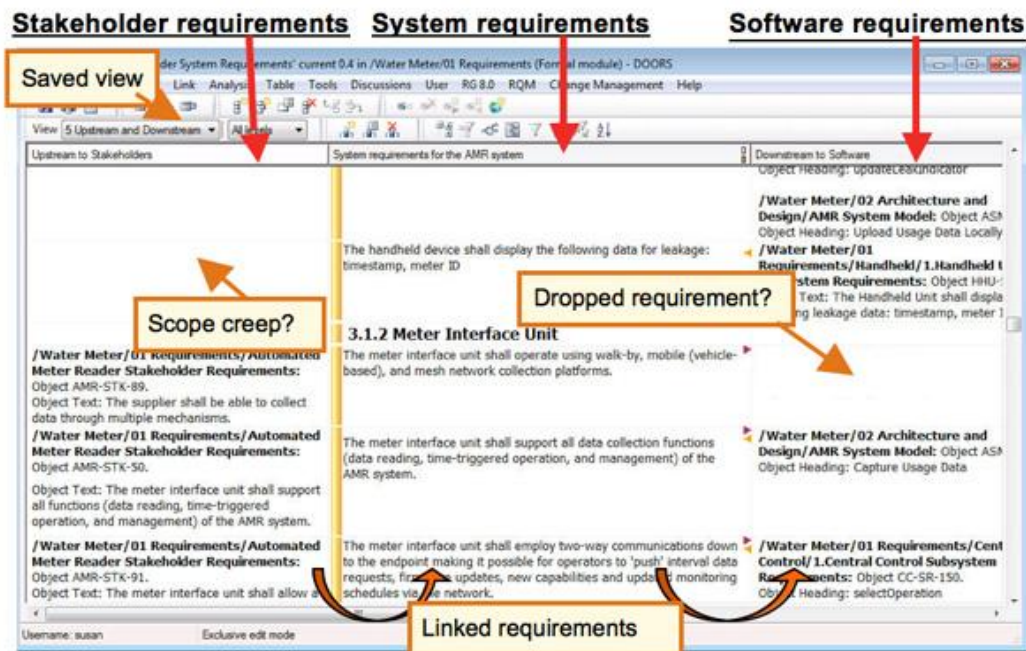


Figure 47. DOORS Linking Between Requirement Modules. (IBM, 2013)

Object Identifier	Object Number	Object Heading	Object Text	Requirement	Verification Method	Criticality	Owner
UR13	3.1	Capability Requirements		No		Medium	
UR14	3.1.1	Carrying Capacity		No		Medium	
UR15	3.1.1.1	Number of People		No	Test	Medium	Mel Gibson
UR17	3.1.1.1.0-1		<p>This object is a requirement and it is linked to a requirement in the next document (In reality, the System requirement is not a good one because it is not easily testable.) The wizard has built an impact trace column to the right. It is a column like any other you have used and can be dragged to another position, resized or saved as part of a new view. Here is the actual user requirement:</p> <p>Four people should be able to travel in comfort for a medium distance.</p> <p>Notice how the link tip to the right and the data in the column on the right both match - as they should. This new view, like all the others, can be printed to give you a traceability matrix. When you create multiple levels of traceability in your project, the Wizard will create a multiple level traceability matrix for you using multiple columns.</p>	Yes	Analysis	Medium	Sandra Bollock
UR383	3.1.1.1.0-2		<p>Now if you could only create links you could use DOORS for real requirements management and traceability. Follow the</p>	No			Mel Gibson

Figure 48. DOORS Requirement Attributes. (Makinen, 2013)

DOORS has been used by different companies for many years, and is still a primary requirements management tool for several organizations. However, it does have some limitations. It is not well embraced by product engineers as it has a perceived "unfriendly" user interface and a reputation for being difficult to learn. This often leads to the need for engineers responsible for the requirements to export the DOORS data into Microsoft products for reviews and then re-entering any updates into the DOORS application. To achieve metrics and views of traceability, the tool often requires extensive customization using DOORS eXtension Language (DXL) scripts to build views, reports, publishing, and queries for attributes. While requirements can be cloned for reuse across product lines or within a project for similar components, relationships between these requirements need to be manually created via links and carefully monitored to ensure the link integrity is maintained.

And while DOORS does interface to Model Based Systems Engineering (MBSE) applications, it does not interface with third part collaboration tools such as Atlassian Jira Software® (<https://www.atlassian.com/software/jira>), so working on requirement development and change is often done external to the DOORS application, with results entered in manually.

As noted in a review conducted by Seilevel, "This tool is one of the most well recognized requirements management tools. The tool is a proverbial 'heavy duty' tool as it has an extensive variety of

traceability, querying and reporting features. Virtually any possible view of information can be set up in the tool, providing flexibility for many types of projects. These strong capabilities arguably make DOORS a standard requirements management tool for industries with significant regulatory demands (e.g. space, defense, medical devices) given their need for regulatory documentation for governance. While DOORS is an excellent all-around tool, there are some limitations, including the dated appearance of the user interface; the tool would look out of place alongside current desktop software and could make for adoption issues with users. Some of the functionality is 'buried', as users have to go through several menus to access the specific capability they were looking for. Similarly, the tool is targeted at systems engineering projects, with an emphasis on features such as requirements traceability and versioning rather than requirements modeling, making it more cumbersome than other tools for visually creating requirements. This is important to keep in mind as the priorities of systems engineering projects do not always align with those of software projects." (Seilevel, 2011).

While Rational DOORS ranked eleventh out of seventeen in Seilevel's 2011 requirements management tool evaluation (Seilevel, 2011), it did not make the top fifty in the most recent Seilevel evaluation from 2016, nor is it listed in the user reviews on the G2 software application purchasing platform (<https://www.g2.com/categories/requirements-management>).

IBM DOORS Next

In the mid-2010s IBM created DOORS® Next as a way to optimize communication and collaboration during requirements engineering efforts (<https://www.ibm.com/products/ibm-engineering-requirements-management-doors-next/faq>). IBM offers DOORS Next as part of a suite of tools on their Jazz platform, which connects multiple products such as systems modeling, test management, and workflow management to the requirements management application. While DOORS Next has many of the same features as DOORS, it has an entirely different look and feel to the user (Figure 49).

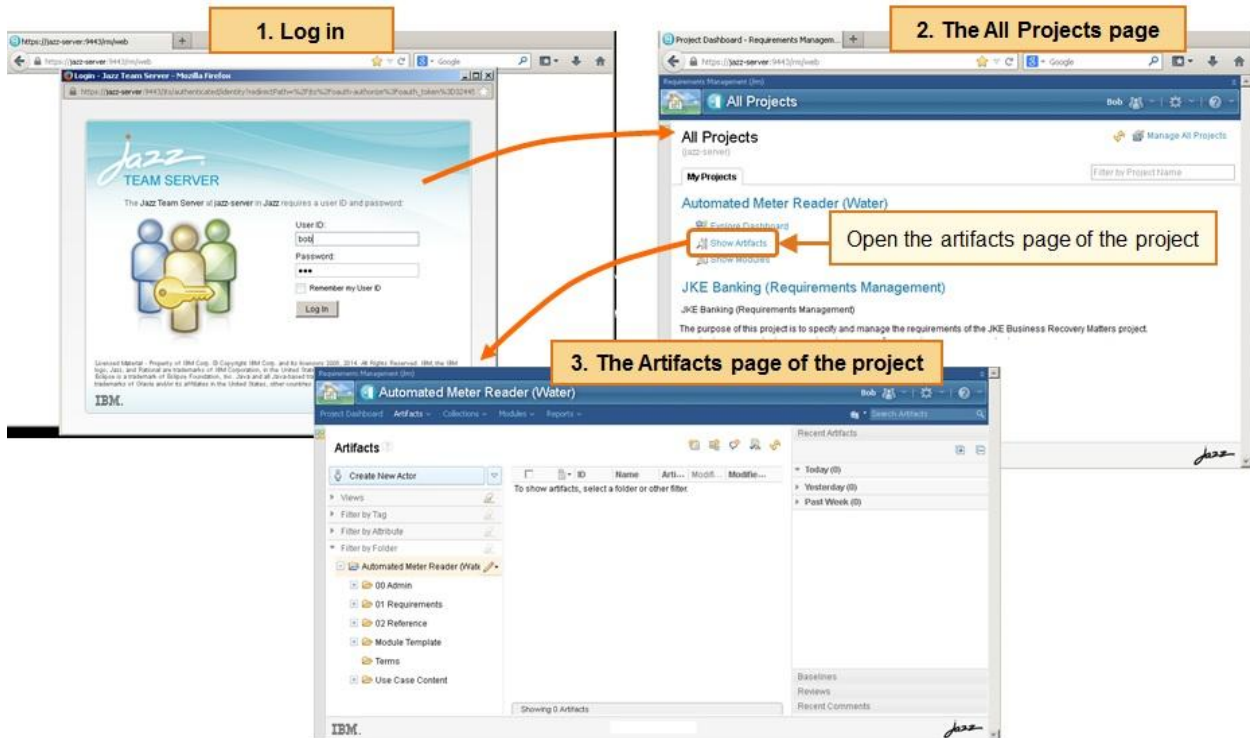
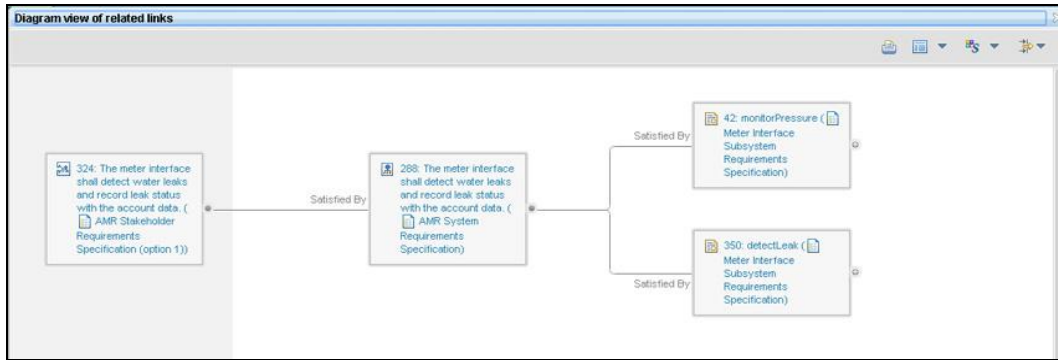


Figure 49. Requirement Database in IBM DOORS Next. (IBM, 2014)

DOORS Next is a web-based interface, and each object entry (such as a single requirement) can stand alone as an artifact, which means they can be used in multiple requirement specifications (known as "reuse"). Because of this capability all requirements within a project have a unique ID, regardless of location within the project's level of abstraction. Additionally, other types of artifacts can be created, such as use cases, test cases, verification events; being able to address requirements and supporting data allows management of a variety of system engineering products within a single application (not just the requirements). DOORS Next includes a built in change control capability, and various metrics are able to be automatically created and displayed (such as showing requirement development status). Reviews of a project's requirements are able to be done collaboratively (and virtually) within the tool using its various collaboration features.

Like Rational DOORS, DOORS Next has capability for artifact to artifact traceability, and allows for requirement attributes; however these come with an updated user interface compared to Rational DOORS (Figure 50 and Figure 51).



ID	Satisfies	Contents	Satisfied By	Dropped requirement?
206	22: The meter interface unit shall be able to collect meter data at regular intervals and store that data for up to one year. (AMR Stakeholder Requirements Specification (option 1))	When connected to a fixed network, the meter interface unit shall wake up and communicate for 4 seconds every 30 minutes, synchronizing all clocks and configuration information. Between these transmissions, the unit remains in a low power state, conserving battery life.		
288	324: The meter interface shall detect water leaks and record leak status with the account data. (AMR Stakeholder Requirements Specification (option 1))	The meter interface shall detect water leaks and record leak status with the account data.	350: detectLeak (Meter Interface Subsystem Requirements Specification) 42: monitorPressure (Meter Interface Subsystem Requirements Specification)	
267		The meter interface unit shall sample water flow every 15 minutes in a 24 hour period to determine leakage.	42: monitorPressure (Meter Interface Subsystem Requirements Specification)	

-3.1.3 Fixed Network Automated Meter Reading System

Figure 50. DOORS Next Traceability Between Requirement Artifacts. (IBM, 2014)

The screenshot shows a table of requirement artifacts for 'Automated Meter Reader (Water) (Requirements Specification)'. Annotations include: 'Module name' pointing to the title; 'Attribute Columns' pointing to the table headers; 'Search the artifacts that are in the module' pointing to the search icon; 'Configure page settings such as which columns to display' pointing to the column configuration icon; 'Collapsed section' pointing to a minus sign in the ID column; 'Artifact' pointing to a row; and 'Locked artifact' pointing to a lock icon in the ID column.

ID	Contents	Accepted	Artifact Type	Priority	Requirement...
13539	-2 General Description		Heading	Priority 2	Requirement...
13506	-2.1 Functions and Purpose		Heading		
13535	-2.2 Environmental Considerations		Heading		
13510	The control computer shall be capable of operating in a normal office environment.	False	System Requirement		Non-Functional
13532	-3 System Requirements		Heading		
13499	-3.1 Functional Requirements		Heading		
13529	-3.1.1 Handheld device		Heading		
13522	The handheld device shall provide for the means for the meter reader to manually enter a meter reading.	True	System Requirement	Medium	Functional
13534	The handheld device shall interfaces with the city's backoffice software.	True	System Requirement	Low	Functional
13508	The handheld device shall allow for programming of a defined route, advancing to the next meter on the route as the meter reader moves through the route.	True	System Requirement		Functional
13516	The handheld device shall have the ability to search for accounts by Last Name, Service Address, Meter Number, and Unread Meters.	True	System Requirement	Medium	Functional
13526	The handheld device shall have a screen capable of displaying	False	System Requirement	Medium	Functional

Figure 51. DOORS Next Requirement Attributes. (IBM, 2014)

At present time IBM recommends new customers start with the DOORS Next (compared to Rational DOORS) to obtain enhanced abilities for collaboration, metrics, and integration with other life cycle management tool suites.

DOORS Next did not qualify for the Seilevel Evaluation Report released in 2016, having been eliminated in an early phase of that study by not making the top 20 (it was within the top 50). User reviews on the G2 platform (<https://www.g2.com/categories/requirements-management>) ranked DOORS Next a 3.9 out of a possible 5 stars.

Jama Software's Jama Connect

Started in 2007, Jama Connect™ enables requirement collaboration and traceability across the product development life cycle (<https://www.jamasoftware.com/>). Jama Connect provides a simple user interface for entering and reviewing requirements (Figure 52), supports change management, and offers traceability to stakeholder needs, architecture, other requirements, and verification and test events. For change management, it provides a collaborative review process as well as an impact assessment analysis.

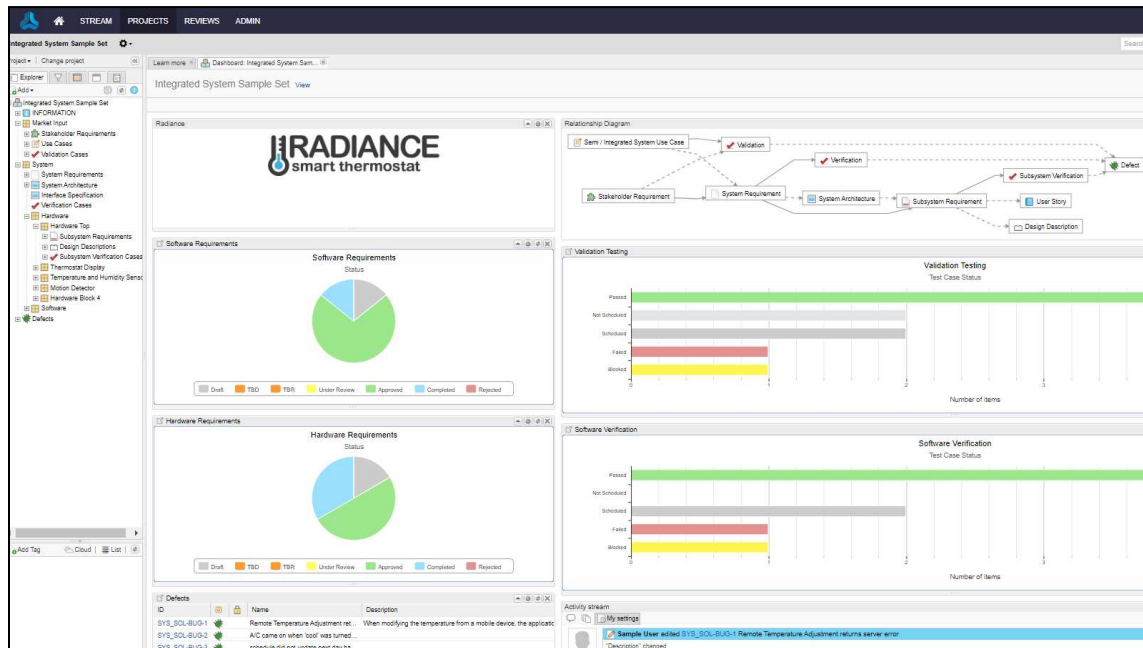


Figure 52. Requirement Database in Jama Connect. (Jama Software, 2020)

Similar to IBM DOORS Next, Jama Connect considers each requirement an individual item, and supports multiple item types, such as requirements, stakeholder needs, use cases, test cases, etc. Relationships between different items can be made quickly with a user friendly web based interface. Jama Connect also supports reuse of requirements among different product lines, or within a project among different components, with an option to keep the requirements related so that variations and changes can be assessed among all variants.

Project ID	Name	Description	ID	Name	Description
AIS-SET-82	System Architecture		AIS-D_DESC-13	Side Impact Sensors	
AIS-D_DESC-20	Architecture Diagram (Image)		AIS-D_DESC-14	Passenger Front Airbags	
			AIS-D_DESC-15	Front Impact Sensors	
			AIS-D_DESC-16	Driver Side Airbags	
			AIS-D_DESC-17	Side Airbag	
			AIS-D_DESC-18	ACU	
AIS-D_DESC-13	Side Impact Sensors		AIS-TSR-33	Impact Sensor Failure Date...	The two impact signals sha...
			AIS-TSR-35	Independent Impact Sensing	Two independent means of...

Figure 53. Jama Traceability Between Requirement Items. (Jama Software, 2020)

Name	Verification Method	Requirement Type	Tags
HLS-R-0070 Daylight Operations - Ini...	Test	Functional	Space Vehicle
HLS-R-0048 EVA Excursion Duration...	Test	Functional	Space Vehicle
HLS-R-0318 HLS Operations Mass ...	Test	Functional	Space Vehicle
HLS-R-0319 HLS Operations Mass ...	Test	Functional	Space Vehicle
HLS-R-0324 HLS Habitation Capabili...	Test	Functional	Space Vehicle
HLS-R-0308 Surface Access - Initial	Test	Functional	Space Vehicle
HLS-R-0001 HLS Reliability - Initial	Test	Functional	Space Vehicle

Figure 54. Jama Requirement Attributes. (Jama Software, 2020)

Jama Connect supports other systems engineering processes such as test and quality management, risk and hazard analysis within the same application to align multiple types of project data together.

When compared feature to feature, Jama and DOORS Next have very similar capabilities. The selection of one tool over the other will often come down to need for an interface with other software applications, preference of a single tool over a suite of tools, installation capability with an organization's IT establishment, and license costs.

As noted in a review conducted by Seilevel, "One of Jama's main benefits is the complete flexibility to customize the object data model and relationship rules, which allows you to really make the tool your own. The traceability feature, with coverage analysis and custom dashboard widgets, makes it easy for your team to achieve a complete set of requirements. The full review center is great; it has the ability to add approval, rejection or specific feedback to a set of selected items, and track review activity (including time tracker per reviewer) on collections of requirements objects." (Seilevel, 2016). Jama was rated #4 in the Seilevel 2016 assessment, and user reviews on the G2 platform ranked Jama Connect a 4 out of 5 stars.

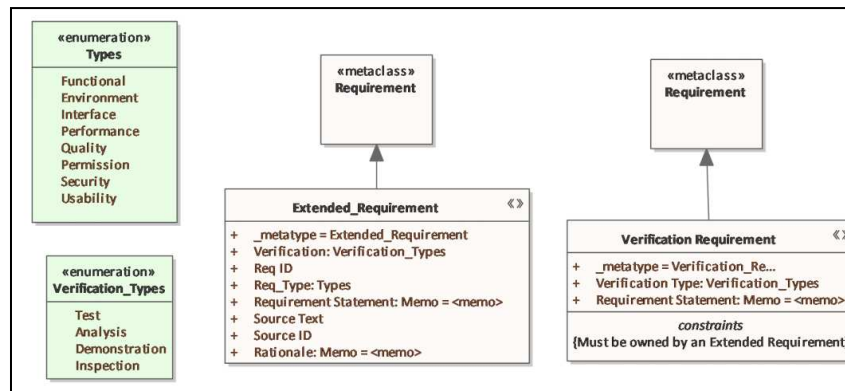
MBSE (SysML) Tool Sets

Multiple tool sets exist for model based systems engineering efforts, which capture the operations, architecture, interfaces, and requirements of various projects using model based systems engineering (MBSE) approaches (reference Chapter 6 for a description of MBSE and the systems modeling language SysML).

In some cases, the projects will use an MBSE tool and then interface (link) the requirements across from a dedicated requirements management tool (where they reside as the source of truth and are only mirrored in the MBSE tool). In other approaches, the project manages requirements directly within the MBSE tool itself. An approach towards complete requirements management in a MBSE model is discussed by (Bernard, 2011) using specific modeling techniques within the Unified Modeling Language (UML), or using the requirement modeling capacities inherent in Systems Modeling Language (SysML). The type of requirements modeling in Bernard's approach is very code intensive, however, and does not align with the approaches of a typical product developer looking to manage requirements generated in

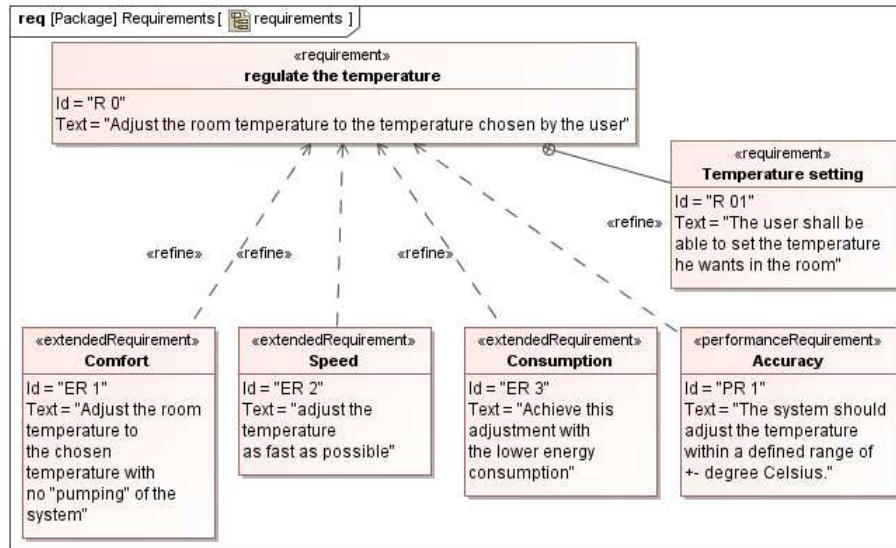
natural language, which is the recommended approach from the *INCOSE Guide for Writing Requirements*. Many product developers have started using SysML-created requirement models to generate and manage their product requirements; the INCOSE French Chapter AFIS has generated a white paper describing approaches to requirements development and management in MBSE models that aligns more with natural language requirements, and provided suggestions and considerations to projects (INCOSE AFIS, 2016).

Two examples are provided here of requirements management in SysML with different tools; the first is with No Magic/Dassault Systèmes Cameo Systems Modeler®, another is with Sparx Systems Enterprise Architect® (both are SysML modeling tools supporting MBSE). Figure 55 shows a requirement diagram and requirement table in Enterprise Architect, Figure 56 shows these in Cameo Systems Modeler.



Item	Req ID	Requirement Statement	Rationale	Status	Priority
Observation Metadata	F-701	The Clarus system shall accept only observations that includes the minimum set of metadata. The minimum set of metadata for an observation is location, timestamp, and source information.	Failure to provide the minimum set of metadata means the observations of multiple systems cannot be correlated.	Proposed	Medium
VF-701		The Clarus system shall be connected to a data input simulator and triggered with a sequence of inputs that includes all required metadata, in addition to inputs that do not include one and two of the required metadata.		Proposed	Medium
Need to determine the availability of the input simulator and ensure it has the desired capability					

Figure 55. Requirement Diagram / Table in SysML (Enterprise Architect). (Baker, 2020)



Criteria

Scope (optional): Source_Requirements () Filter: Q-

#	Id	Name	Text	Satisfied By
1	S0.0	<input type="checkbox"/> Original Statement	Describe a system for purifying dirty water. - Heat dirty water and condense steam are performed by a Counter Flow Heat Exchanger - Boil dirty water is performed by a Boiler. Drain residue is performed by a Drain. The water has properties: vol = 1 liter, density 1 gm/cm3, temp 20 deg C, specific heat 1cal/gm deg C, heat of vaporization 540 cal/gm.	
2	S0.1	<input type="checkbox"/> Elevation	The water distiller shall be able to operate at least 2 meters vertically above the source of dirty water.	
3	S1.0	<input type="checkbox"/> Purify Water	The system shall purify dirty water.	
4	S2.0	<input type="checkbox"/> Heat Exchanger	Heat dirty water and condense steam are performed by a Counter Flow Heat Exchanger	<input type="checkbox"/> Heat Exchanger <input type="checkbox"/> Heat Exchanger <input type="checkbox"/> Heat Exchanger <input type="checkbox"/> Heat Exchanger
5	S3.0	<input type="checkbox"/> Boiler	Boil dirty water is performed by a Boiler.	<input type="checkbox"/> Boiler <input type="checkbox"/> Boiler <input type="checkbox"/> Boiler <input type="checkbox"/> Boiler
6	S4.0	<input type="checkbox"/> Drain	Drain residue is performed by a Drain.	<input type="checkbox"/> drain : Valve <input type="checkbox"/> Valve <input type="checkbox"/> drain : Valve <input type="checkbox"/> Valve <input type="checkbox"/> drain : Valve <input type="checkbox"/> Valve <input type="checkbox"/> Valve <input type="checkbox"/> drain : Valve
7	S5.0	<input type="checkbox"/> Water Properties	Water has properties: density 1 gm/cm3, temp 20 deg C, specific heat 1cal/gm deg C, heat of vaporization 540 cal/gm.	
8	S5.1	<input type="checkbox"/> Water Initial Temp	Water has an initial temp 20 deg C	

Figure 56. Requirement Diagram / Table in SysML (Cameo Systems Modeler). (No Magic, Inc./Dassault Systems, 2020)

Both the requirements diagram and the table provide different views of the product requirements, which allows the traceability and configuration control essential for requirements management processes. These tools also allow for attribute definition, aligning to best practices in *INCOSE Guide for Writing Requirements*, with different capabilities in how to create and manage the attribute information. A

consideration of usage of these tools mirrors some of the concern relayed earlier with Rational DOORS, in that these modeling tools involve considerable training to learn, and tend to have limited amount of users interfacing with the tools directly as they are not considered "user friendly".

Ranking of Requirement Tools

This author has personally used each of the requirement management tools described above, however many more requirements management tools exist than those used by this author. In the mid-2010s, Project Performance International (PPI) has released a requirements management tools list containing over one hundred different tools, and is now currently working to build a database of tools with INCOSE that will be searchable by desired feature. The G2 software purchasing platform has 44 requirements management tools listed, many with user reviews about their features.

While this author has preferences in tool features and capabilities, for this dissertation an objective look has been done to show how various tools are evaluated and ranked across the broad industry of users. Seilevel, a professional services organization focused on IT product management, has released two reports that objectively evaluated various requirement management tools, one in 2011 and one in 2016. The most recent report originally assessed 46 requirement management tools, and then pared down to a list of 21, with rankings shown in Figure 57.

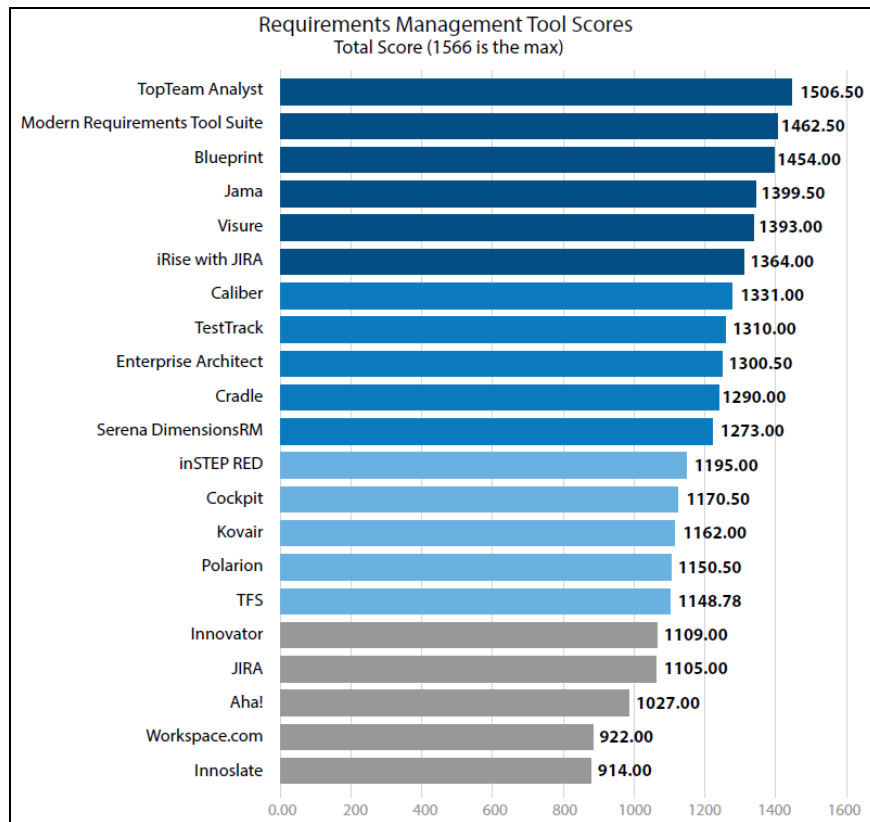


Figure 57. Seilevel Requirement Management Tool Rankings. (Seilevel, 2016)

Candase Hokanson, business architect for Seilevel, provided a list of top features of tools in her 2016 blog, noting the top ten capabilities and features that the users were looking for in management of their requirements, which includes the following (Hokanson, 2016):

1. Requirements specification and prioritization - ability to add, edit, delete and prioritize requirements easily.
2. Traceability and dependencies - ability to create relationships between requirements and change the data model to reflect the traceability needed in the organization.
3. Stakeholder management, review and collaboration - ability to give feedback on requirements or initiate workflows to approve requirements.
4. Change control - ability to baseline requirements, track changes after a baseline, or revert requirements set back to a baseline.
5. Visual Modeling - ability to create and edit models in the tool or link requirements to visual models.
6. Import/export and reporting - ability to import to / export from MS Word, Excel, Visio or other sources and report the requirements, models or subset of either group.

7. Requirements process support - ability to set up own templates and object types to support a methodology with things like checklists, issues, risks or constraints.
8. Task / Iteration management - ability to track development tasks on requirements, set release or iteration dates, or create burndown charts.
9. Licensing, support and tool administration - flexible licensing for the tool, adequate support materials and ease of maintenance.
10. Scalability, integrations and ease of use - how intuitive is the tool, ability to scale, ability to integrate with other applications.

In recent discussions with Ms. Hokanson, in addition to the top ten items from her 2016 blog post, current trends for desired features in requirements management tools now include flexible data hierarchy (multiple types of objects compared to requirements) and flexible traceability (many-to-many relationships) (Hokanson, 2020).

3.2.5 Observations from Requirements Engineering Practitioners

To understand lessons learned and recommended improvements to current requirements management practices, a Delphi Method approach was done to assess trends in the requirements management process by interviewing individuals who have actively developed and managed requirements across various industries. These interviews were held with experienced requirements management practitioners to capture their input on good practices and what they believe contributes to either effective or wasteful results. The following contains a summary of the interviews, including observed challenges with current approaches and recommendations for what should be included in an effective requirements management approach. Each practitioner has relevant take-away points summarized at the end of their sections, and some common themes are captured to build upon as improvement options for the requirements management process in Chapter 5. It should be stated that each viewpoint is the personal opinion of the requirement expert and does not represent the opinion of their respective institutions.

Lou Wheatcraft, Requirement Consultant (Wheatland Consulting)

Mr. Lou Wheatcraft has fifty years of experience in requirements development and management across multiple industry domains, and is currently working as a requirement consultant. Mr. Wheatcraft

is a co-chair of the INCOSE requirements working group, co-author of the *INCOSE Guide for Writing Requirements*, and lead author developing the new *INCOSE Needs and Requirements Lifecycle Manual*.

Mr. Wheatcraft has observed an evolution in requirements engineering over the last several years based on the need to develop increasingly complex systems. He noted that approaches which worked well previously are no longer bringing the same results, and that the role of model-based engineering is now leading to requirements management approaches that are more data-centric than before.

On some of his projects, approaches to requirements development often involved separate groups of teams with disconnected sets of efforts. As example, his work on one NASA project had a group responsible for interfaces, one responsible for development of requirements, one responsible for architecture development, and another responsible for verification development. These groups would be driven by management to synchronize their efforts periodically, but for the most part worked in separate "silos".

Another observation on the same project was the refinement of lower level requirements, to establish a contract with the supplier, well before the system level needs and requirements were fully developed. This had the result of putting a constraint at the system level and reducing various system capabilities.

Mr. Wheatcraft also noticed a disconnect in the requirements management processes among the NASA program team, as many were not aware of the project's documented requirement management plan and were implementing processes based on their own knowledge.

Besides the disconnected approach within management of the requirements, Mr. Wheatcraft noted that the requirements development was not based on an overall system concept or set of needs, but was driven by individual assessments and experiences (most particularly the organization responsible for operations of the space mission), and lacked perspectives of other viewpoints and as well as a system viewpoint.

When Mr. Wheatcraft started to work requirements engineering for the medical device industry, he noticed that this highly regulated field drove a large focus on robust product verification and

validation, influenced by regulatory agency insight into early development of needs and the design process. This drove a consistent process across the industry, from his perspective, towards robust requirements development, requirements management, and product verification and validation; all to ensure the resultant device would obtain regulatory certification.

Mr. Wheatcraft's experience in the diesel engine industry also led to the observation that feedback from the consumer and user community provides an opportunity for informal product validation, as any poorly developed product would be subject to recall or consumer rejection, impacting the product provider's profit margin. One provider that was having issues in this area sponsored an investigation into the current practices, which highlighted that the project had poorly documented requirements with several disconnects in component interfaces, and lacked an overall system view of the design. The results from this investigation allowed an opportunity to work improvement in those areas that were found deficient.

Mr. Wheatcraft's observation on various requirements engineering approaches across different industries is that too much focus is on the set of requirements, and not enough is placed on the process related to development and documentation of the system's needs (citing various industry standards from software engineering, INCOSE and NASA). Mr. Wheatcraft believes that the context of needs should have a larger focus in the requirements engineering process. The requirements management process is really a "needs and requirements and verification and validation" management process as all of this information is intrinsically related.

Mr. Wheatcraft also noted that requirements on a project have significant work associated with them (implementation, verification activities), and that he sees many companies using design output requirements (those that express how the design is to be realized) used as design input requirements (those that are implementation free and specify the needs of the system to be developed); the result of this can yield constraints upon the design solution. He also notes that this results in an increased number of requirements, ultimately resulting in increased costs to the project to manage and verify the system against all of the requirements.

Some overall requirements management recommendations from Mr. Wheatcraft include:

- Focus on the needs and system concept prior to working the requirements definition;
- Avoid locking in lower level requirements before the system concept and needs are established;
- Use a data centric approach to management of the needs, requirements, and project data;
- Ensure collaboration occurs during requirements development among project team members which reflects multiple points of view of the system being developed; and
- Avoid using design output requirements (implementation specific) as design input requirements (function specific).

Kathryn Trase, Senior Systems Engineer (Ball Aerospace)

Ms. Katherine Trase has nine years of experience in systems engineering, with a focus on requirements management for most of it. Ms. Trase started her career at NASA Glenn Research Center working requirements on several space-based projects. Her viewpoint provides a nice counter-balance to Mr. Wheatcraft as someone with less years of experience yet with several similar observations.

During one of her efforts working NASA projects, Ms. Trase observed that the association of international teams had an impact on where their product resided in the overall product structure, causing a confusing requirement allocation as this was a lower level element shown at a high level in the structure. This was due to the nature of organizational interface, as the international community works with higher organization levels of NASA for collaboration and communication compared to the U.S. based prime contractors. The net result was a requirements review process that varied based on component provider, and a mix of requirements at various levels causing inconsistencies of how the requirements decomposition process would be performed.

When she worked on this NASA project the requirements were in a stable phase, with most of the requirements development work completed and the providers under contract. Ms. Trase observed that a frequent source of requirements change came from the lower level, and would often be a suggestion from a contractor related to alignment with capabilities they were able to meet, or findings from their own implementation and verification efforts.

This product also had multiple missions, and Ms. Trase noted that they were beginning to address requirements variability for the different missions using modular requirement sets for the different variants, with a base set that they would all meet.

Ms. Trase has worked within a variety of different requirements management tools, including IBM Rational DOORS, 3SL Cradle, and No Magic Cameo Systems Modeler. This particular project used Cradle as a requirements management tool, and Ms. Trase noted that there were very few users trained or comfortable in directly interfacing with Cradle. Due to this, most of the reviews and discussions occurred in exported artifacts and then updates were manually entered into the tool by a specialist.

One challenge noted by Ms. Trase during her time at NASA was the lack of visibility into the entire set of system requirements data, from the headquarters customer and across three major programs which interfaced with each other. An effort was generated to align the data from the various project teams into a comprehensive set of data. This involved a task of setting up a tool that interfaced with the tools used by the various NASA project teams and some of the prime contractors to query the project data from their diverse requirements databases, as well as a number of other engineering sources, to provide a linked data environment; this effort required collaboration with IT development teams and was completed successfully.

While on a different NASA project, Ms. Trase had an opportunity to work requirements development utilizing a project architecture model in Cameo Systems Modeler (an MBSE SysML application). She found this approach enabled efficient requirements development and validation, as well as ensured the data had commonality across the requirements and the design.

Based on her projects and research, Ms. Trase noted that there could be a lot of benefit in aligning data on a project and use it to generate and validate the project requirements. There is currently development ongoing in the field of natural language processors that can extract requirement performance values and connect it with an architecture model, this can enable analysis of the requirement value to determine if it yields a desired outcome for the project. It is believed that this type of data centric approach shows promise for the future of requirements engineering.

Some overall requirements management recommendations from Ms. Trase include:

- Avoid mixing requirements for different levels at the system level;
- Use requirements management tools that easily enable the review of the requirements to be done within the tool;
- Find a tool that enables automatic generation of metrics associated with requirements development and management;
- Find a tool and process for management of requirements that frees up systems engineering time for engineering, instead of tool entry and administration work;
- Find an approach to extract requirements data from different organizations' management tool databases to allow for project data queries; and
- Develop a way to integrate the requirements data with other data and analyses functions to enable validation of the requirement set.

Ken Eastman, Senior Project Manager (Ball Aerospace)

Mr. Ken Eastman has over twenty five years of experience managing requirements in the space industry working for major corporations on products such as remote sensing instruments, spacecraft, and space systems.

Mr. Eastman has worked with the IBM Rational DOORS tool during his time in requirements management. His observation is that most project reviewers and customers prefer to review requirements from artifacts exported from DOORS (such as Microsoft Excel files), which involved marking up these files and then importing the data back into DOORS. This was a challenge in working with the test organization as well, as it became difficult to align the requirements to the correct test events (where the data was managed in other applications). It was also manually intensive to generate metrics using DOORS, such as verification completion burn-down graphs.

On one project, Mr. Eastman had a prime contractor who passed their customer's requirements directly to his project without doing the decomposition process. This resulted in a challenge to determine the exact requirements they were expected to meet for his project. This was compounded by hidden requirements in that specification, where requirement attributes such as "rationale" contained additional mandatory items that needed to be met, masked when this requirement set was entered into a

requirements management tool and the requirement statements were reviewed. This ultimately caused rework for that project as the customer expected a different result than was provided.

Mr. Eastman has observed that the better projects were ones where verification effort and selloff plans were worked early, during the requirements development process. This enabled the verification data to align with the requirements and resulted in a smoother verification closure process.

Many of the products Mr. Eastman has worked were similar to other projects at the company, where they were able to reuse many of the requirements of the past projects while developing approaches to meet the specific system or contractual needs.

On his current space system project, Mr. Eastman noted the requirements management process is a bit easier as much of the requirements development across the different product elements is occurring within the same company using co-located teams, allowing a singular control of development effort.

Some overall requirements management recommendations from Mr. Eastman include:

- Use a requirements management tool to enable requirements trace;
- Do up front verification planning and connect the verification data to the requirements;
- Use pre-defined templates to facilitate the requirements development and management effort;
- Use MBSE tools to define project architecture and needs early;
- Assess similar projects and apply their lessons learned to the requirements development and management effort; and
- Perform the requirement reviews in the requirements management tool, where users look at a common set of data, instead of making data exports and mark up external artifacts.

Raymond Wolfgang, Systems Engineer (Sandia National Labs)

Mr. Raymond Wolfgang has over eighteen years of experience managing requirements in the nuclear and naval industries. Mr. Wolfgang is also a member of the INCOSE requirements working group, and lead author developing the new *INCOSE Guide to Verification and Validation*.

Mr. Wolfgang observed that requirements management rigor has been variable over his career, with some projects using minimal efforts (such as development of small-scale physical security systems), and others more stringent (development of large national-security systems). In his more recent larger

projects he noted that the government customers often provided a set of top level requirements ranging from a six to eight page specification document, which would then be used to derive the total system requirements (often around three to four hundred system requirements), consisting of functional and nonfunctional requirements and associated constraints (such as environments). These would then be allocated to the system elements in the product structure to develop lower level requirements.

For requirements management tools, he has experienced managing document based requirements in a configuration management system, and more recently has been using IBM Rational DOORS to manage requirements, allowing his project to implement traceability and change control. Mr. Wolfgang also uses the requirements management database to support verification planning and associated reviews with the stakeholders.

Originally, he observed that only a few systems engineers were using the DOORS tool to manage the requirements, however efforts have been in work to train the project team members in how to update their requirements in DOORS directly in support of requirements development and verification planning. Mr. Wolfgang has observed a mixed response to this, some users are adapting to using DOORS while others are still requesting an export from DOORS to mark up so they can provide it to the systems engineers to enter updates into DOORS. One observation is that DOORS does not lend itself to an effective requirements review in meetings as the user interface cannot be scaled for viewing (font size or display cannot enlarge), and occasionally this necessitates the need to do an export for the meeting.

Mr. Wolfgang has also observed that the requirements change control process is starting earlier on some of his projects than on previous projects, and they are deliberately assessing the timeframe to ensure the development work can occur with flexibility during the time of most changes, yet the requirements are baselined early enough for the project to use a common and controlled set for their design and development efforts. Mr. Wolfgang advocates finding a time for this which is not too early, yet not too late (this may vary based on project parameters).

Some overall requirements management recommendations from Mr. Wolfgang include:

- Ensure an organizational mandate, with rudimentary guidance and training, to utilize the requirements management tools to avoid inconsistent usage and support;
- Spend time early to set up a strategy for the requirements management effort, and build in templates, organize the tool, and aim to have a common look and feel for the users across different projects;
- Align the functional requirements to the product environments early;
- Ensure requirements are in a requirements management tool to ensure the organization has comprehensive requirements in one place to support the design efforts; and
- Have the requirements engineering effort done by separate personnel than those doing the design of product; this allows for concurrent work as well as enables an objective focus on the functions needed (as opposed to how their design will fit the project solution).

Joel Knapp, Deputy Systems Engineering Lead (NASA Glenn Research Center)

Mr. Joel Knapp has over thirty years working in the space industry on NASA projects, with over twelve of them spent in requirements management. Mr. Knapp is also an INCOSE certified systems engineering professional (CSEP).

Mr. Knapp worked on microgravity experiment equipment early in his career, and noted that requirements management was not considered important on those projects. The requirements would often be documented and then put aside while the team develops the product. As he moved to projects that were high value and safety critical, Mr. Knapp found that requirements management became much more rigorous.

Mr. Knapp spent time on the same NASA project as Ms. Trase (mentioned previously), and worked requirements management in 3SL Cradle. He noted that some of the tool features were helpful, while others were not. He stated that having a common location of the system model and the requirements would further benefit the requirements engineering process (currently not an option with Cradle). Mr. Knapp noted that the requirements were fairly established on this project when he started it, and that most of the change management is addressing the contractor requested changes (which is a different type of impact assessment than a customer driven set of requirements changes).

Some overall requirements management recommendations from Mr. Knapp include:

- Find the best time to start change control on the requirements - early on it is expected they will change frequently, but these should stabilize as the design and subcontracts are starting to be established and change control will be needed at that time;
- Recognize that changes can occur for external factors outside of the project's control, and have a plan to address these changes;
- Develop the requirements from an analysis of what the system should do, and not based primarily on subject matter experts' opinions;
- Perform a rigorous functional decomposition, and an analysis to validate the requirement set; and
- Try to adhere to Eric Honour's finding on how much time to spend up front on systems engineering as an indication of project success - the optimum SE activity for programs is 14.4% of the total program cost (Honour, 2013)

Kevin Orr, Systems Engineering Specialist (Eaton Corporation)

Mr. Kevin Orr has over twenty years of experience managing requirements in the space/aviation industry and the oil and gas industry, working for major corporations. Mr. Orr is also a co-chair of the INCOSE requirements working group, and lead author developing the new *INCOSE Guide to Needs and Requirements Development and Management*.

During his time working space projects, Mr. Orr observed a highly rigorous application of requirements management, addressing requirements traceability and change control across the system. When he joined projects in the oil and gas industry, he observed less rigor and a more reactionary stance to start management of requirements after high profile malfunctions prompted the industry to address safety concerns. Requirements management was one of several systems engineering practices being developed to ensure the products being produced met regulations and conformed to safe practices (and avoiding making the news due to safety accidents). Major development of these systems were done by the oil companies, where the product developers were often responding to contracts to provide hardware without full understanding of the requirements or verification expectations. This was improving during Mr. Orr's time in that industry, where they started to use requirements management tools, however he noted this was variable in application and successful implementation.

Throughout his career, Mr. Orr observed a varied response of suppliers with respect to requirements development and changes. In some projects (such as high valued space product development), he observed the suppliers charging the prime contractor / customer significantly for any requirement changes. In other domains, such as an aviation project with long term maintenance opportunities, the suppliers would address the development and changes more collaboratively with their prime contractors as this was considered a mutually beneficial and long term business opportunity.

Mr. Orr has experience using multiple requirements management tools, including IBM DOORS Next, IBM Rational DOORS, 3SL Cradle, No Magic Cameo Systems Modeler. One observation from Mr. Orr was that the ability of Rational DOORS for customization allowed the tool's default requirements management features to be improved upon, but this was tempered by impact of any new version of the software supplied by IBM, which could work against the customization work done by the organization.

Mr. Orr also observed the 3SL Cradle tool had a difficult user interface, which led to only a few specialists that interfaced with the tool directly. Requirement reviews were often done outside of the tool and then review inputs manually imported into the tool afterwards. He also noted that it was a challenge when other vendors on the project used different requirements management tools as it was difficult to see requirement trace across all levels in a systems view. On one large NASA project, the prime contractors were contractually forced to change to the NASA default tool of Cradle. One company did switch over all of its requirements management on that project to Cradle, while the other used IBM Rational DOORS and only imported into Cradle for delivery to NASA.

For requirements management, a key area of focus is change management to ensure awareness of change impact. However, Mr. Orr cautions against change restrictions on all aspects of a requirement (such as attributes designed to enable planning of verification activities), as this can be a source of hidden costs associated with review of the change. Mr. Orr also advises a thoughtful approach to the timing of when to baseline the requirements to allow frequent changes during the early requirements development effort.

Some overall requirements management recommendations from Mr. Orr include:

- Find the right balance of rigor and flexibility in the requirements management approach for a project;
- Work in the requirements management tool as a project team, eliminate the data entry by specialists with the most team members only reviewing artifacts exported from the tool;
- Minimize requirements management tool customization and be prepared for the software tool maturation with new versions;
- Establish a rapport with the requirements management tool provider to influence the direction of the tool development to address any insufficient capabilities;
- When implementing change management, do not enforce it on all attributes associated with requirements, only apply to those that need impact assessments and version tracking; and
- Integrate the requirements with other project data, and use a data focused approach with supplier exchanges.

David Hill, NASA Gateway Digital Transformation Manager (Barrios Technology)

Mr. David Hill has over twenty five years of experience in the space and oil and gas industries, with two years devoted specifically to requirements management oversight. He is currently leading the NASA Gateway architecture effort and ensuring the requirements and architecture are captured in electronic tools, moving the effort away from its prior document centric approach. The NASA Gateway project is part of the Artemis Program, which is discussed in Section 4.6.

Mr. Hill noticed observable improvements in NASA artifact production after going to a more data focused approach. Prior requirements for NASA Johnson Space Center (JSC) projects were managed via a document approach with multiple approvers for every document change. This involved a significant amount of systems engineering staff to oversee requirements documentation creation, alignment of requirements within different documents to act as book managers, and continued discussions with requirement stakeholders to address every change within a document revision. The configuration management tool and revision process used at NASA JSC is a heritage effort that aligns with document centric change control.

On NASA Gateway they opted to use a requirements management tool (IBM DOORS Next) to have a more data centric approach to developing and managing the requirements, which includes

functional and nonfunctional requirements as well as interface requirements and environments. The Gateway architecture and mission was modeled in a MBSE SysML application (Cameo Systems Modeler) to support development of the system requirements. The requirements were then put into DOORS Next with a link to the architecture model to show its trace to source (the driving need) as well as show discussions via a collaboration record captured in the requirements management tool. Upon review, the individual requirement is baselined, and after that a change control process is done to make any updates. A document export is then able to be published after the review of the requirements is complete.

This approach was in effect when the artifacts for the Gateway Program System Requirement Review (SRR) were generated from the database, and Mr. Hill believes this new approach saved approximately 25% of the labor cost of prior project SRR artifact generation (being able to use less people for similar product generation). Having a data focused effort with a collaborative requirements management tool allowed a single source of truth and mitigated risk of requirement inconsistency at the system level, which is an additional benefit.

When asked why they chose this approach, Mr. Hill noted that the current generation of engineers entering the NASA workforce are expecting electronic, data focused approaches, having utilized these in prior educational efforts. As a counter, the engineers which preferred a more document focused approach had worked at NASA for many years, and Mr. Hill noted that the adoption of the electronic collaborative change process was more challenged by that demographic. Additionally, there existed a culture within the overall change control process of having documents reviewed by many stakeholders who were allowed to approve a change document. He stated that if the Gateway project was allowed to go entirely electronic in the change review process, they could realize a reduction in the amount of "non value" work and streamline the requirements management process even further, potentially realizing about 40% savings over prior efforts. Non-value work was defined by him as document generation, spending time on format of documents to align with prior program formats, spending time routing documents, and then manually inputting other reviewer information into an electronic tool.

An additional challenge noted by Mr. Hill has been the use of multiple companies and nations involved in the effort to develop Gateway. Because of concerns of violation of the International Traffic in Arms Regulations (ITAR) and exposure of company proprietary data, there is no plan for a system-wide tool used across the different partners and contractors, nor is there intent to mandate a common tool across all parties. However, the different organizations are expected to provide a common data set to the NASA Gateway program office (through a DOORS module or Microsoft file) that can be imported into the NASA requirements management tool. This enables traceability from the system level to ensure requirement allocation is complete at the lower levels. There is intent (forward work) to define a common data model and attributes for all organizations to utilize in their requirements management effort.

Mr. Hill's ideas for improvements in the requirements management process include utilization of a tool consultant to support configuration of the requirements management tool, and up front training to the program team and program leadership in the use of the data centric and collaborative requirements management approaches.

Some overall requirements management recommendations from Mr. Hill include:

- Establish the requirements from the system architecture model;
- Use a requirements management tool which allows artifact generation for project reviews, and have the tool vendor set up with a support contract to help establish the tool and address questions during the project life cycle;
- Find an approach to bring in requirements data from different organizations' management tools to see the entire system requirements data;
- Avoid non-value added work such as formatting, document centric reviews, and aim to collaborate as much as possible with the requirements management tool itself; and
- Obtain project management and organization management support for a data centric approach and team collaboration.

Observations from Practitioner Discussions

Several themes were noted with the individuals interviewed, including the need for a data centric approach to requirements management, usage of "user friendly" management tools, ensuring requirements development and change control timing are carefully balanced, and ensuring that the requirements are

developed to support the project needs and utilize analysis to validate the requirements against the needs. Table 9 provides a summary of the various themes and occurrence of recommendations from the requirement practitioners.

Table 9. Summary of Practitioner Recommendation Themes.

Theme	Number of Responses out of 7 Interviewees
Approach to Requirement Development	5
Data Centric Requirement Management Approach	5
Effective Tool Usage	5
Ensure Requirement Quality	4
Change Control Timing	3
Tailorable Approach to Requirements Engineering	2
Change Control Methodology	1
Early Verification Planning	1

When comparing the inputs from the practitioners against the current state requirements management model from Figure 36, each of these steps was noted as being performed and important. However, the inputs also highlighted that the method of *how* these are implemented can have major impact for project success; as example, the usage of tools and collaboration in the requirements development process will have a significant effect on the outcome.

3.2.6 Requirement Trends Conclusions

Looking at the various trends in requirements management processes and information from the experienced practitioners, some key trends include the following:

- There is a movement away from a document centric approach of managing requirements towards a data centric approach, treating requirements as a form of project data traceable to other project data;
- Requirement management tools are utilized for development, collaboration, change control, and trace to other project data; tools are more effective if they are setup with project configuration and templates beforehand;
- Project teams will utilize requirement management tools if they have easy to use features, otherwise only the requirements engineers tend to use the tools;

- Verification of the product showing it meets its requirements is enabled by usage of requirements management tools, traceability, and connection to events and artifacts from the verification events (such as tests);
- Careful planning is needed on when to start change control on requirements, too soon or too late can have impact to project execution, and controlling too many requirement attributes can drive schedule.

A requirements management process model was introduced in Figure 36. Taking this model, and considering the trends, themes are observed that prompt deeper investigation:

- What are the best approaches for addressing changes to requirements to the product development organizations?
- How can usage of a requirements tool help enable collaboration and requirements reviews?
- What is the optimal approach to flow requirements from the system to the component level to ensure efficient execution of a new development program?

The next chapter will look into the challenges associated with space system development, and look at examples of requirements management implemented on various space programs. An examination of what processes worked well, and which ones did not, will help define focus areas for optimizing the process of requirements management on space systems.

4.1 Overview of the Space Industry

Requirements management is a broad topic that encompasses about every system being developed. To ensure a manageable scope of effort in addressing optimization of requirements management, the focus will be reduced to one domain, the space industry. Development of space systems is a highly complex field with consistent challenges among its products. This chapter will highlight the specific challenges of space system development and factors that need to be considered when performing requirements management. Several space project examples will be provided and then summarized to compare the approach and complexity of requirements and the outcome of the project.

4.1.1 Complexity in Space Systems

Systems today are becoming more complex than at any time in the industrial world. As described in his keynote address at the INCOSE International Symposium in 2018, Dr. Zhang Xin Guo observed that the evolution of general product systems has grown from basic mechanical and analog products to increasingly "smart" systems with complex and real time mechanical, electronics, software and network (M/E/S/N) interactions. Modern products leverage more autonomous functions and artificial intelligence, using data and providing near instantaneous responses to address required functions (Guo, 2018).

Dr. Guo's example of this is shown in Figure 58, highlighting the exponential growth in the last fifty years of the index of complexity, measured as the number of elements, relationships between them, and the number of relationships between the system and its environment. While this increase in complexity is reflective of many types of industries, there are additional factors for the space industry that brings challenges towards system design.

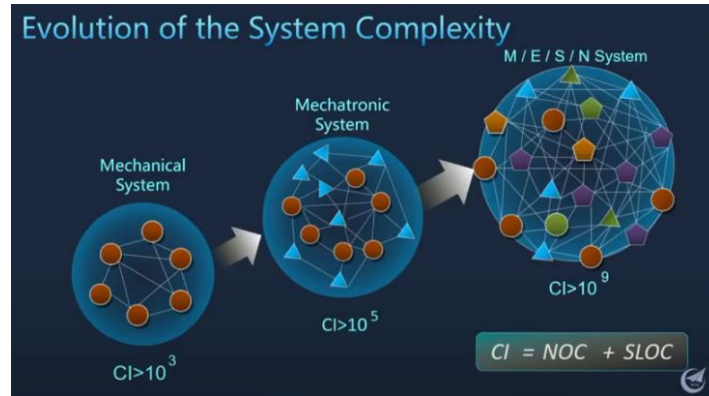


Figure 58. Engineered Systems Have Seen Exponential Growth in Complexity Over the Last 50 Years. (Guo, 2018)

The United States space industry includes space and launch vehicles, and associated infrastructure. Because of the great emphasis on research and development, about 25% of those who work in space are engineers, scientists, and technicians (Investopedia, 2018). The space industry's primary products require up to millions of individual parts, not including the support systems needed to operate the space vehicles and implement the space missions. An example product structure is shown in Figure 59, taken from the *NASA Systems Engineering Handbook*, highlighting the numerous parts that made up the space shuttle system.

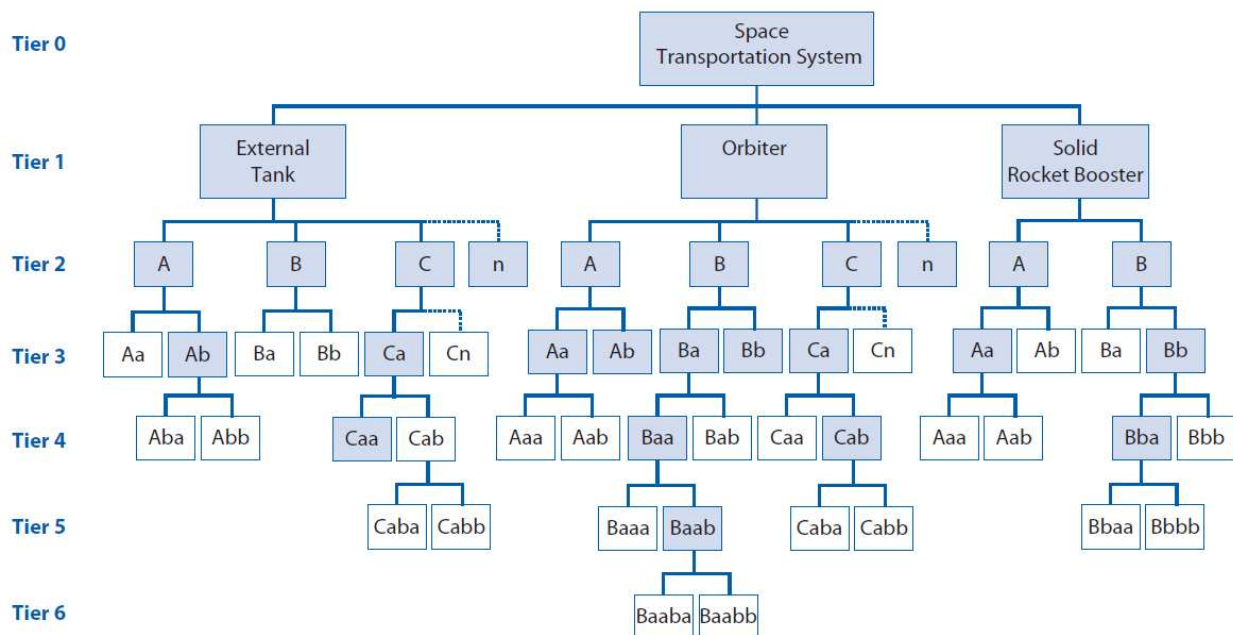


Figure 59. NASA Product Structure Example. (NASA, 2007)

Space system design and development efforts are most often accomplished by an experienced prime contractor, overseen by a customer organization which develops an overall space mission. Mission development and acquisition is often by a federal agency (NASA, Department of Defense, one of the military branches, etc.), but can include private customers looking to procure space assets such as a telecommunications or mapping imagery satellite.

Unlike the aviation industry, there are very few United States federal regulations for the development of space products as most operate in an environment away from people and are not considered human safety-critical. The exceptions are human crewed spaceflight systems, where NASA provides standards for human safety. While these are considered premiere missions, they only make up a small fraction of the overall space industry. For non-crewed missions, regulations include the Air Force launch requirements (for launches out of Cape Canaveral and Vandenberg) to ensure public safety during rocket launches; standards for space communication frequencies, controlled by the Federal Communications Commission (FCC) to ensure compatibility with allowed frequencies; and control of orbital debris, monitored by the Federal Aviation Administration (FAA) and Air Force (likely now the Space Force), to ensure debris does not pose a hazard for re-entry and public safety. To support regulatory compliance, organizations developing the space missions and supporting products supply evidence of meeting the specific regulation parameters as a function of obtaining licenses to launch or operate.

This dissertation proposes an optimized approach for requirements management for the space industry for a range of product types. While it may be applicable for other industries, this focus is chosen to ensure the process can address the unique considerations of developing space-based products. These unique challenges are shown in Table 10, which highlights challenges such as the product operating environment, need for high reliability parts with long lead times, small numbers of product suppliers, and a narrow market with only a few customers.

Table 10. Comparisons of Various Industries. (Wertz, Everett, & Puschell, 2011)

Comparisons of Various Industries	
<p>SPACE INDUSTRY:</p> <ul style="list-style-type: none"> • High technical risks-complex systems • Routine requirements for clean room assembly and testing • Hostile operating environment • Labor intensive production • High entry barriers -expensive R&D and tooling • Complex and costly integration technology and methodology • Launchers are finicky-need lots of attention • Narrow market-few customers • Few suppliers-limited competition • Long lead times • Transportation infrastructure critical • Extensive testing during and following manufacture and during prelaunch phase • Heavy regulatory burdens • Not location centric <p>MINING:</p> <ul style="list-style-type: none"> • Hostile operating environment for men and machines • Dirty, potentially hostile environment • Labor intensive • High entry barriers-expensive essentially handmade equipment • Equipment needs lots of maintenance • Limited markets-broader than for space industry • Limited number of mines-geography limits competition • Long lead times for mines to become productive • Transportation infrastructure critical • Heavy regulatory burdens • Location centric <p>BIG PHARMA:</p> <ul style="list-style-type: none"> • High technical risks-complex equipment and methodologies • High entry barriers • Wide markets • Lon lead times • Transportation infrastructure moderately critical • Heavy regulatory burdens • Not location centric <p>BIOTECH (STARTUPS):</p> <ul style="list-style-type: none"> • High technical risks • Labor intensive • Low entry barriers-usually evolve from academic laboratories 	<p>BIOTEC (STARTUPS) Continued:</p> <ul style="list-style-type: none"> • Narrow markets-goal is acquisition by Big Pharma • Long lead times • Transportation infrastructure not critical • Regulatory burdens low until human trials begin • Usually location centric <p>AMBULATORY SURGICAL CENTERS:</p> <ul style="list-style-type: none"> • Low technical risks • Labor intensive • Moderate entry barriers-no more than several million dollars • Broad markets with favorable demographic trends • Short lead times-roughly 1 yr from funding to operations with 1-2 months float on receivables • Transportation infrastructure not critical • Heavy regulatory burdens • Extremely high returns on initial investment • Location centric <p>EARLY MICROCOMPUTER INDUSTRY (HARDWARE):</p> <ul style="list-style-type: none"> • Variable technical risk-ICs development for other purposes • Low entry barriers • Wide market • Highly competitive market over time • Variable lead times • Transportation infrastructure not critical • Low regulatory burden • High returns on initial investment • Not location centric <p>EARLY MIRCOCOMPUTER INDUSTRY (SOFTWARE):</p> <ul style="list-style-type: none"> • Low technical risk • Low entry barriers • Potentially broad market • Variable competition increasing over time • Transportation infrastructure not critical • Low regulatory burden • Very high returns on initial investment • Not location centric <p>SANDWICH SHOP:</p> <ul style="list-style-type: none"> • No technical risk • Low to moderate entry barriers • Potentially broad market • Variable competition increasing over time • Transportation infrastructure not critical • Low to moderate regulatory burden • Variable return on investment • Location centric

The diversity of products in the space industry is highlighted in Figure 60, and typical budgets are highlighted in Figure 61, demonstrating that the development of products for space missions is a multi-billion dollar global industry.

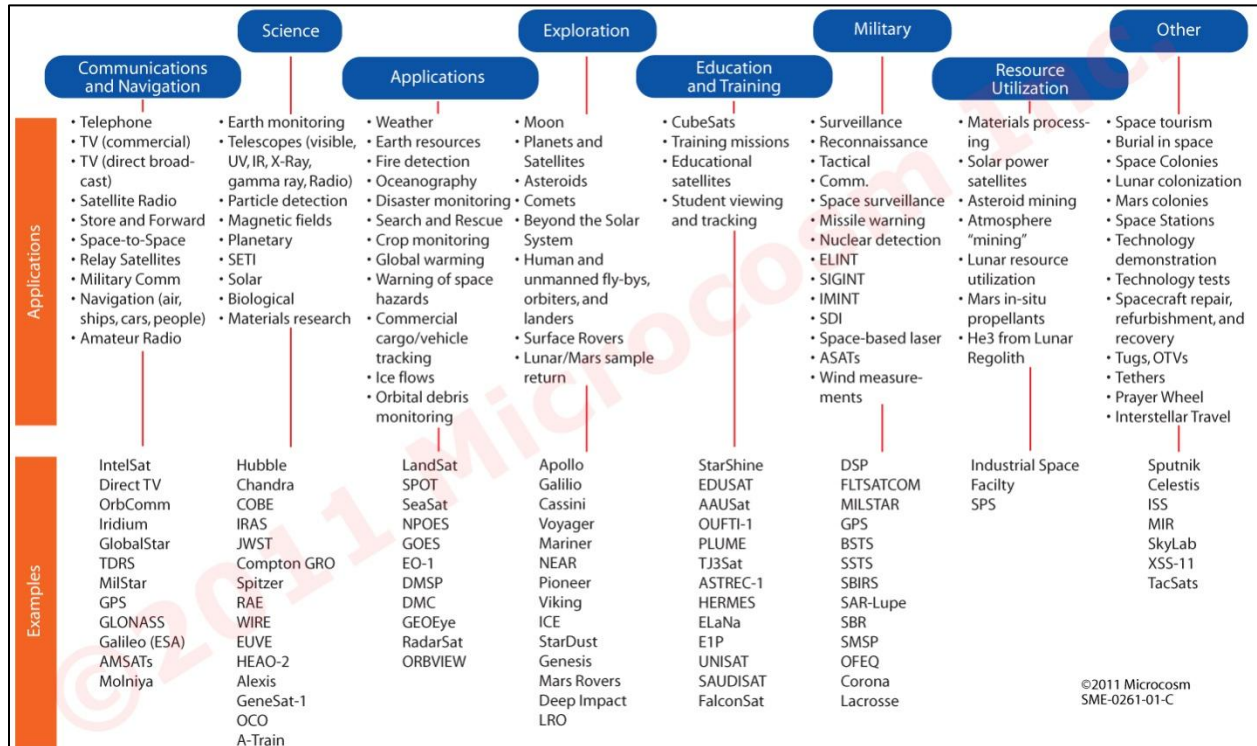


Figure 60. Wide Range of Space Mission Applications. (Wertz, Everett, & Puschell, 2011)

Number of Nations Operating in Space	81
Number of People Employed in the Space Economy (2017)	1 million
Size of the Space Economy (Q2 2018)	\$414.75B
Value of Commercial Space Products and Services (2018)	\$229.17B
Number of Spaceports (2018)	40 +10 in development
Economic Impact of GPS (since 1983)	\$1.35T
Increase in Number of Spacecraft Launched (2018-2019)	+7.2%
Space-Related Patents (2018)	Quadrupled over 20 years
Global Government Spending on Space Programs (2018)	\$85.55B
STEM Employment in 2028	10.56M

Figure 61. 2008 Global Space Spending by Major Category. (Space Foundation, 2019)

Traditional approaches to project management and systems engineering still apply to the development of space products; however, the project execution tends towards a more rigorous approach to development, review, and verification activities than most other industries. Both project management and systems engineering disciplines ensure success by focusing on technical performance, cost, and schedule, and on parameters such as return on investment, market acceptance, and sustainability (Forsberg, et al., 2005).

Space systems are highly complex and frequently consist of system of systems. Often a mission will be developed to meet a need (exploration, communication services, scientific observations), and then decomposed to a series of top-level requirements by the mission developers that are levied to various segments representing a launch vehicle, a space vehicle, and a ground system that provides communication, command and control services. Space products may still be generated from product lines, however they tend towards customization based upon the specific mission objectives they are expected to achieve. Requirements management has been in place since the early days of the U.S. space program; requirements from early space missions look similar to requirements today. To demonstrate this, below is a requirement statement from the Apollo program generated in 1963, compared to a current requirement in the NASA Gateway Human Landing System (HLS) requirement created in 2019:

Apollo System Specification, M-D M 8000.001, 1963:

4.3.1.1.5 The CM shall be designed such that a single crew member while in lunar orbit can perform all essential CM operations for at least seven days.

Human Landing System Requirements Document, HLS-REQMT-001, 2019:

HLS-HMTA-0106 Crew Task Volume
The system shall provide a habitable volume that accommodates crew living and work tasks.

For many years the United States space industry product development was overseen by a handful of prime contractors, notably Boeing Corporation, Lockheed Martin Corporation, Northrop Grumman, United Launch Alliance and the Raytheon Company. Many space companies have merged throughout the years, forming consolidated engineering and manufacturing organizations that compete with each other for major government and commercial contracts. These companies tend to have established processes and methods to address product development, manufacturing and test activities. Their various products are produced in similar manner, and trend towards high cost, lengthy schedule efforts.

In recent years several new companies have been established which use alternate approaches towards product development, decreasing their time to market and serving as a disruptor for introducing new technology (Bold Business, 2018). Examples of these companies are Space Exploration Technologies Corporation (SpaceX), Sierra Nevada Corporation and Blue Origin, which are pursuing opportunities for major space contracts as well as developing their own privately owned space systems. Much of the research into optimal requirements approaches come from looking at the methods of these companies in addressing newer processes, tools, and risk postures, and is discussed further in Chapter 5.

While NASA has directly developed many space systems, currently they tend to oversee the development of the space systems being produced by aerospace contractors. Some exceptions to this exist, however; NASA consists of several different centers, with some of them more active in the engineering and manufacturing efforts of space products than others. Because NASA information is publicly available, the research into various space systems in this dissertation will contain a variety of NASA managed programs, ranging from small probes to human exploration systems.

One interesting observation is how newer companies such as SpaceX respond to traditional approaches from traditional U.S. Government customers such as NASA and the Department of Defense; "It's more expensive to do these missions," SpaceX President Gwynne Shotwell said of U.S. government launches compared to commercial missions (Westcott, 2014). Shotwell noted that the company's Falcon 9 launch prices will add between \$10 million and \$30 million per launch to account for these additional processes. "A launch of the Dragon space station cargo capsule aboard a Falcon 9 about doubles the price

of the SpaceX mission..They have us provide more data to them. They have folks that basically reside here in SpaceX, and we need to provide engineering resources to them and respond to their questions. So by definition, the way the government buys missions is more expensive." (Westcott, 2014).

The differences in approach between SpaceX and more traditional product developers is based on a philosophy within SpaceX to achieve a "good balance of creativity and systems engineering for agility and affordability" (Muratore, 2012). The team at SpaceX works to have a balance between heavy up front systems engineering compared to rapid prototyping to reduce systems risk, addressing the dependencies of organizational agility, cost of iteration, and the ability to trade lower level requirements. This is in contrast to a more traditional approach of the U.S. Government customers, who rely on building a set of requirements for their contractors heavy with design standards and quality specifications that were based on past lessons learned, prompting the need to respond to all of these "shall" statements.

To demonstrate various complexities of requirements management on space systems, examples are provided of past and current projects that highlight the types of requirements applied, the variation of approach based on mission and customer types, and the different techniques utilized to meet the objective of product realization. Due to competition sensitivity most commercial space companies do not publish their system requirements, therefore the research is limited to published articles, and data available from open sources such as NASA; it is worth noting that these examples from NASA are reflective of this author's own experiences working on other space projects. For each example, the requirements approach is provided, an estimate of overall project cost is shown, outcome of the project is discussed (successful or not achieved), and a discussion of factors in requirements approach for further is provided. Prior to presenting this content a discussion on how project success is measured is presented in the next sections.

4.1.2 Definition of a Successful Space System Project

What is Project Success?

Project success takes into consideration the multiple factors that are weighted based on the parameters considered important to the project and its stakeholders. Often this is discussed as a triangle

associated with schedule, budget, and technical adherence, however organizations that only evaluate a project's success according to the original “triple constraint” may fail to apply the most important test of all: the client’s satisfaction with the completed project (Pinto, 2016). A "quadruple" constraint concept considering the client acceptance is shown in Figure 62.

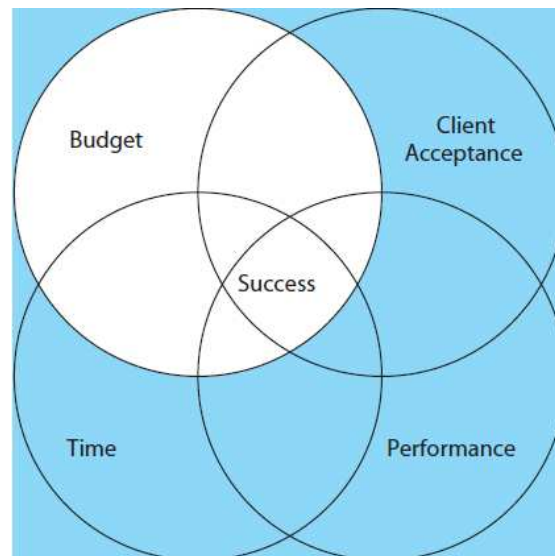


Figure 62. Quadruple Constraints For Project Success. (Pinto, 2016)

Each space project example that follows includes an assessment with respect to each of these parameters to address overall project success, with a discussion on how the approach towards requirements could have enabled, or caused issues with, the outcome. As the actual customer satisfaction value may not be available, the customer will be identified and a subjective evaluation based on their interests will be provided.

Cost Optimization for Project Success

Optimization addresses the relationships and trade-offs between characteristics. For project success, an optimized balance of cost, performance, schedule and customer satisfaction is achieved, as shown in the center of Figure 62. To optimize for cost while still achieving project success, costs of implementing changes in approach is weighed against the benefits of the approach. A classical trade off example in project management is regarding expediting a project's schedule. A low cost project could have minimal staff, and take longer to produce. To reduce the time to completion, more resources could

be added, increasing the cost. Achieving a balance of affordable resources towards an acceptable schedule duration becomes an optimization activity. To further optimize for cost, the schedule may be expedited by removing tasks or functionality in the product while still achieving an acceptable product for the customer. Cost optimization is not simply minimal cost, it is the **lowest cost while still achieving all objectives for project success**.

4.1.3 Space System Cost Calculations as a Function of System Requirements

To provide a basis for comparison, a cost model will be applied to calculate approximate systems engineering labor cost associated with each project. The Constructive Systems Engineering Cost Model (COSYSMO), developed at the University of Southern California with the support of a consortium of academic, industry, and government organizations, is a parametric model for estimating the systems engineering and integration effort required for the conceptualization, design, test, and deployment of space systems. COSYSMO defines a relationship that estimates systems engineering labor effort in person months, based on size drivers (including the number of requirements), factored by reuse of prior designs, and adjusted by fourteen effort multipliers that capture the product and project environment and complexity factors (including the quality of requirements). The COSYSMO parametric relationship is shown in Equation 1 (Valerdi, Wang, Ankrum, Millar, & Roedler, 2008).

$$PM_{NS} = A \cdot \left[\sum_k \left(\sum_r w_r (w_{e,k} \Phi_{e,k} + w_{n,k} \Phi_{n,k} + w_{d,k} \Phi_{d,k}) \right) \right]^E \cdot \prod_{j=1}^{14} EM_j \quad \text{Equation 1}$$

Where:

PM_{NS} = effort in Person Months (Nominal Schedule)

A = calibration constant derived from historical project data

k = {REQ,IF,ALG,SCN}

r = {New, Modified, Adopted, Deleted, Managed}

W_r = weight for defined levels of reuse

W_x = weight for "easy", "nominal", or "difficult" size driver

Φ_x = quantity of "k" size driver

E = represents diseconomies of scale

EM = effort multiplier for the *j*th cost driver. The geometric product results in an overall effort adjustment factor to the nominal effort.

The drivers are shown in the equation as k, and consist of inputs for the number of system-level requirements, the number of system interfaces, the number of system-specific mathematical algorithms used to achieve the system functional and performance requirements, and the number of operational scenarios that the system must satisfy to accomplish its intended mission. The version used in this dissertation is COSYSMO 2.0 (Valerdi, 2010). Figure 63 shows the COSYSMO model with the entry fields for each of the size drivers and complexity factors.

COSYSMO 2.0
CONSTRUCTIVE SYSTEMS ENGINEERING COST MODEL
 © 2009 Jared Fortune
 8-Jul-10

ENTER SIZE PARAMETERS FOR SYSTEM OF INTEREST

Reuse?

	Easy	Nominal	Difficult	
# of System Requirements				0.0
# of System Interfaces				0.0
# of Algorithms				0.0
# of Operational Scenarios				0.0

equivalent size
0.0

SELECT COST PARAMETERS FOR SYSTEM OF INTEREST

	N	1.00
Requirements Understanding	N	1.00
Architecture Understanding	N	1.00
Level of Service Requirements	N	1.00
Migration Complexity	N	1.00
Technology Risk	N	1.00
Documentation	N	1.00
# and diversity of installations/platforms	N	1.00
# of recursive levels in the design	N	1.00
Stakeholder team cohesion	N	1.00
Personnel/team capability	N	1.00
Personnel experience/continuity	N	1.00
Process capability	N	1.00
Multisite coordination	N	1.00
Tool support	N	1.00
1.00		composite effort multiplier

SYSTEMS ENGINEERING PERSON MONTHS

Figure 63. COSYSMO 2.0 Cost Model. (Valerdi, 2010)

The sources of data for COSYSMO are thirty-four projects from six companies in the aerospace and defense sector. Raytheon, BAE, General Dynamics, the Aerospace Corporation, Northrop Grumman, and Lockheed Martin provided data, and three of these companies were responsible for twenty-seven of the thirty-four data sets upon which COSYSMO is based (Valerdi, Wang, Ankrum, Millar, & Roedler, 2008).

The quantity of requirements used in COSYSMO is based upon the specific level of design being assessed and may be functional, performance, feature, or service-oriented in nature. Each requirement on a project has effort associated with it, such as verification activities, functional decomposition, functional allocation, etc. System requirements can typically be quantified by counting the number of applicable

requirements in the system specification and any derived requirements in support of achieving the mission objectives. The requirements may also spread across multiple specifications (such as a performance specification, and interface requirements specification, an environment specification, etc.).

For entry into COSYSMO the option of easy, nominal and difficult for number of requirements is provided, which has a weighting function for the input (Madachy, 2015). These are a measure of requirements quality, and the definitions are as follows:

- **Easy** is defined as simple to implement, traceable to source and singular;
- **Nominal** is defined as familiar, able to trace to source with some effort, and containing some overlaps; and
- **Difficult** is defined as complex to implement, difficult to trace to source, and containing a high degree of overlap.

As cost is a function of number of system level requirements, it can be surmised that the fewer the system requirements the less costly the system (all other parameters being held the same value); in actuality this may not be true as a very poorly specified system may have latent associated costs. For the sake of this study the COSYSMO parametric model will be utilized as a comparative cost analysis tool showing predicted labor costs on the basis of nominal system requirements that align to the needs of the system. For usage in this costing effort, the system will be defined as the system of interest - which starts at a defined entry point and encompasses a discrete boundary.

The next several sections will provide several examples of past space projects. The estimate of systems engineering costs based on COSYSMO will be provided based on how these various projects implemented their system level requirements. To normalize the calculations as a function of requirement quantity and quality, the parameters of reuse, number of system interfaces, software algorithms, and operational scenarios will be set to the same values (5 nominal system interfaces, 10 nominal algorithms, 5 operational scenarios, no reuse). The results from this assessment will provide a foundation for later comparisons from a proposed revised approach.

4.2 Space Project Example 1: MAVEN

4.2.1 MAVEN Overview

NASA's Mars Atmosphere and Volatile Evolution (MAVEN) Program is dedicated to the scientific exploration of the Martian atmosphere with the objective of understanding the reason for water loss on Mars. The MAVEN spacecraft, shown in Figure 64, was launched on an Atlas V-401 launch vehicle on November 18, 2013, and entered Mars' orbit on September 21, 2014. Its primary mission duration was one year, which concluded in October of 2015 (NASA, 2015). After that its extended mission has been to observe all of Mars's seasons for a full Martian year (98 weeks), and then to continue performing additional science collection as NASA budget approval is granted. By adjusting orbital parameters to minimize fuel consumption MAVEN has extended its operation several years after its primary mission.

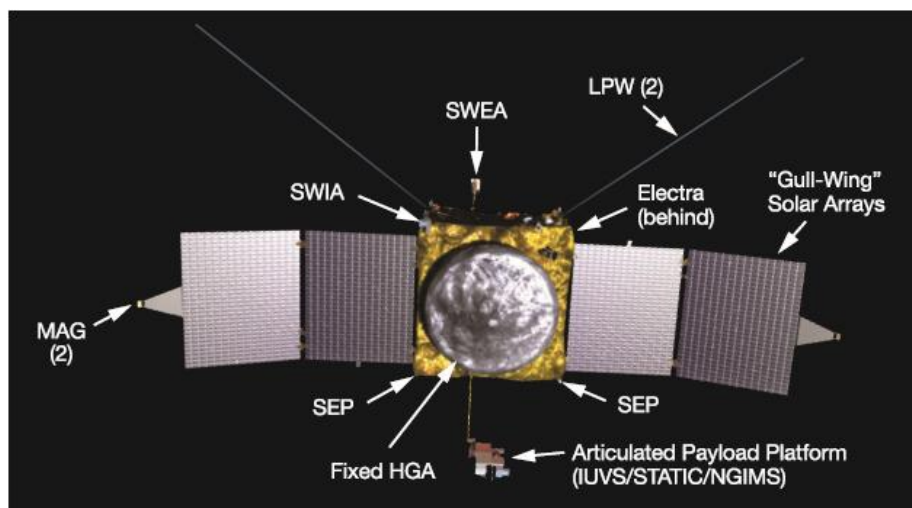


Figure 64. The MAVEN Spacecraft. (NASA, 2013)

The MAVEN Program is managed by the NASA Goddard Space Flight Center (GSFC) in Greenbelt, Maryland. The program was awarded on September 15, 2008, with a budget of \$671 million (NASA, 2013). Design efforts were overseen by GSFC, with the integration and test of the spacecraft overseen by Lockheed Martin (Figure 65).



Figure 65. The MAVEN Spacecraft Prior to Launch. (NASA, 2013)

Figure 66 provides the graphical depiction of its product structure, along with the responsible organizations for each element.

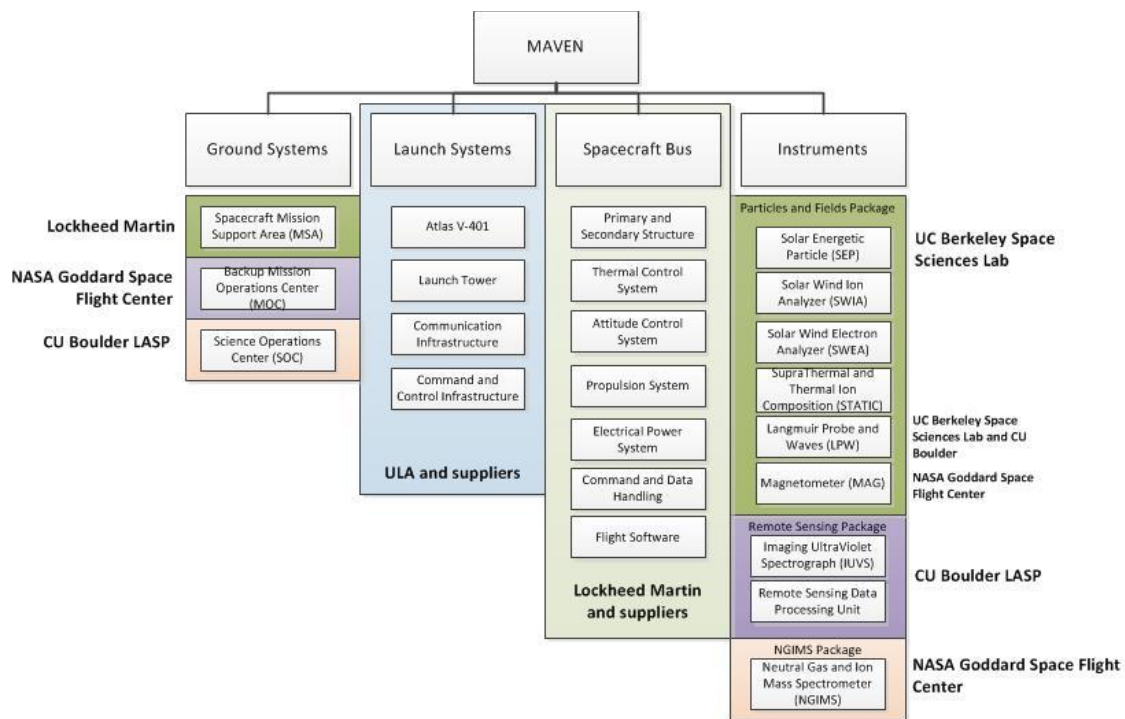


Figure 66. The MAVEN System Product Structure.

4.2.2 MAVEN Requirements Approach

The requirements imposed on MAVEN were derived from the Level 1 Science Requirements captured in MAVEN document MAVEN-PM-RQMT-0007. These requirements reflected the stakeholder needs, and were subsequently analyzed and decomposed into increasingly more specific and detailed

requirements in the level 2 specifications, such as the MAVEN Mission Requirements Document (MRD), MAVEN-PM-RQMT-0005. The MAVEN document tree, shown in Figure 67, provides the various requirements documents that made up the MAVEN requirement set from level 1 to level 4 (not shown is the component assembly requirements at level 5).

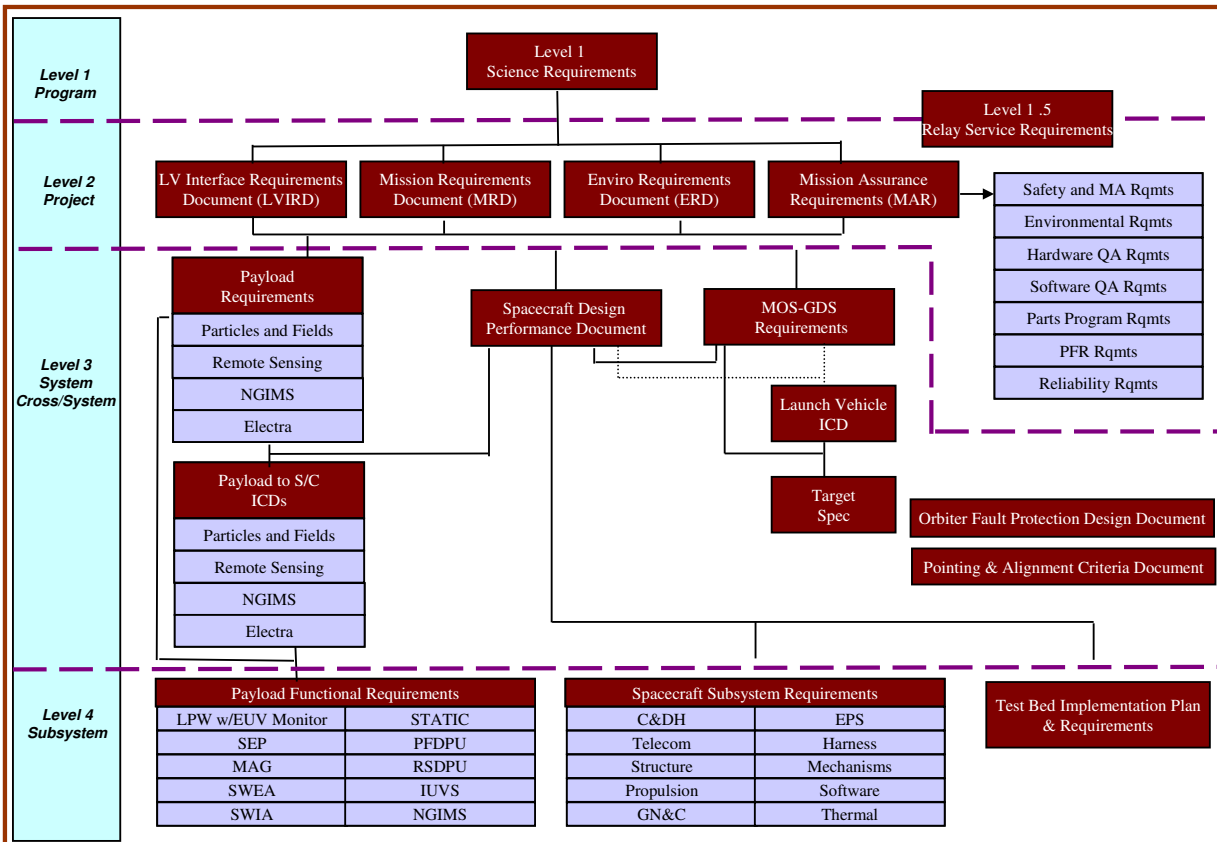


Figure 67. The MAVEN Requirement Tree. (NASA, 2011)

MAVEN level 2 also included the applicable NASA procedures and standards containing requirements on project management, systems safety, systems engineering, software development, quality management, financial management, and independent reviews. An example of a requirement from the Mission Requirements Document (MRD) is provided in Figure 68.

<p>ID : MRD153</p> <p>Section : 2.4.4</p> <p>Title : Electra Data Volume</p> <p>Relay Return Link Data</p> <p>The spacecraft shall be capable of storing 1 GB of Electra return link relay data and delivering that data to the Mars Exploration Program.</p>

Figure 68. Example MAVEN MRD Requirement. (NASA, 2012)

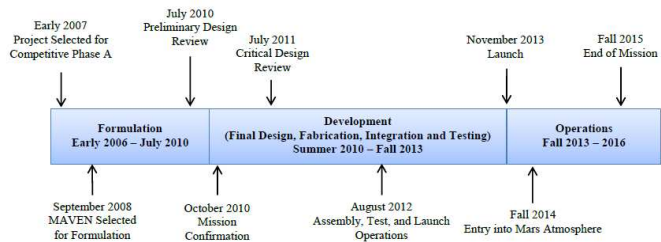
MAVEN requirements were managed using IBM Rational DOORS to support linking to higher/lower level requirements, requirements management, as well as requirement validation and verification planning and tracking.

Multiple organizations were involved in the development effort, enacting an approach similar to Figure 15, where the requirement handoffs were occurring with the system to instrument providers, and from system to spacecraft to spacecraft component providers.

4.2.3 MAVEN Project Outcome

As seen by the continued performance of the spacecraft on an extended mission, the MAVEN project is considered a success. MAVEN continues to perform science observations and provide data to support further understanding of Mars.

Upon review of project parameters reviewed by the U.S. Government Accounting Office (GAO), the MAVEN project's cost, schedule and technical performance aligned with the original planning estimates (GAO, 2013). The schedule was particularly critical as the launch window needed to be met to achieve parameters for reaching Mars. Figure 69 shows an overview of the MAVEN schedule and adherence to planned budget.



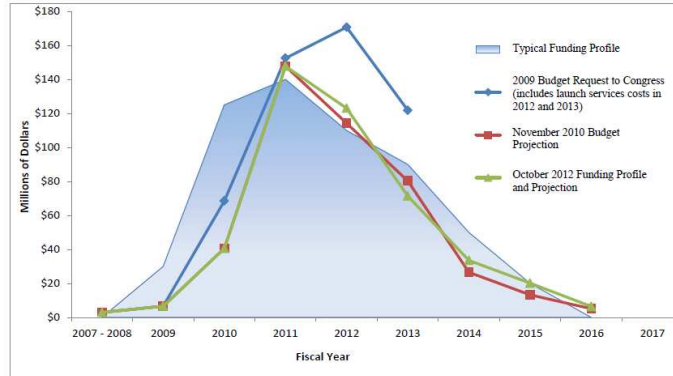


Figure 69. MAVEN Budget and Schedule Overview. (GAO, 2013)

Based on the above information, the following assessment in Table 11 reflects the total outcome of the MAVEN project.

Table 11. MAVEN Project Outcome

Parameter	Assessment	Project Success?
Cost	Within budget.	Success
Schedule	Met schedule.	Success
Technical	Met science objectives.	Success
Customer Satisfaction	Customer is the science community and the taxpayers who funded the mission; both communities viewed this project as a success.	Success

With respect to the costs associated with systems engineering due to requirements, the systems engineering labor estimate is obtained from COSYSMO. The COSYSMO calculation is based on requirements associated with the new development work, which are obtained from the mission requirements document, the environment requirements, and the mission assurance requirements (this system used an existing launch vehicle, therefore the launch vehicle interface control requirements were not included in this calculation). The total number of system requirements is 660, and it is observed that these are primarily complete and singular (contained minimal overlap with other requirements and very little inclusion of additional invoked standards). The COSYSMO weighting factor of "nominal" is used as the requirements are considered somewhat similar to prior Mars missions, able to trace to source with

some effort, and consisted of a few overlaps. The estimated systems engineering labor for MAVEN is shown in Table 12.

Table 12. Maven Calculated Labor Costs from COSYSMO.

Project	# of System Requirements	Estimated Systems Engineering Labor	Project Notes
MAVEN	660	298 person months	Successfully executed project objectives

4.2.4 Assessment of MAVEN Requirements Approach

The product structure in Figure 66 depicts established ground systems and launch system usage, showing that MAVEN's new development efforts were focused on the spacecraft bus and instruments. The level 2 system specifications consisted of a performance specification, a design and environment specification, and a mission assurance specification.

The spacecraft bus was developed by an experienced prime contractor using a vehicle design based on heritage spacecraft platforms. The instrument development efforts were distributed to multiple NASA and university laboratories, with project oversight by the University of Colorado. The spacecraft and instrument requirements focused on performance, functions, and quality expectations and were aligned with the level 1 science needs.

Based on the GAO report, the MAVEN team heavily leveraged prior efforts for Mars missions to address mission requirements, project execution, and testing approaches.

From a supplier perspective, the spacecraft and its sub-tier suppliers leveraged existing processes and designs. The requirements management process utilized existing processes and requirements management tools. The instruments were developed based on interface control documents and performance specifications through a NASA and University arrangement, which would enable a more collaborative approach compared to a company with a subcontract arrangement.

The MAVEN approach appeared successful on many fronts, with minimal to little challenges with respect to implementing a requirements management process.

4.3 Space Project Example 2: Mars Science Laboratory, MSL (Mars Curiosity Rover)

4.3.1 MSL Overview

The Mars Science Laboratory (MSL) consisted of a space probe mission to Mars which landed a Mars Rover named "Curiosity" in the Martian Gale Crater in 2012 (Figure 70). Curiosity set out to answer the question "Did Mars ever have the right environmental conditions to support small life forms called microbes?". Early in its mission, Curiosity's scientific tools found chemical and mineral evidence of past habitable environments on Mars. It continues to explore the rock record from a time when Mars could have been home to microbial life (NASA, 2020).



Figure 70. The Curiosity Rover in Operation (Self Portrait). (NASA, 2020)

Design efforts were overseen by the NASA Jet Propulsion Laboratory (JPL), and starting in October 2003 with a mission concept review and concluding in 2009 with a re-baseline review. The Rover was designed, assembled, and tested at JPL in Pasadena, California (Figure 71).



Figure 71. Curiosity Rover During Integration and Test Operations. (NASA, 2020)

The Curiosity Rover was part of an overall spacecraft that contained the cruise stage and the entry, descent and landing (EDL) system; each spacecraft element had specific functionality for each phase of the overall mission (transport to Mars, land on Mars, science exploration). The entire MSL spacecraft launched on a Atlas V-451 launch vehicle on November 26, 2011, and the Rover landed on Mars on August 6, 2011 (NASA, 2020). The expected mission was one Mars year (23 Earth months), which Curiosity successfully completed as it is still operating today. Information about Curiosity status is available on the NASA website <https://mars.nasa.gov/msl/home/>.

There are several instruments on Curiosity to enable it to sample Martian terrestrial and atmospheric elements. These are highlighted in Figure 72.

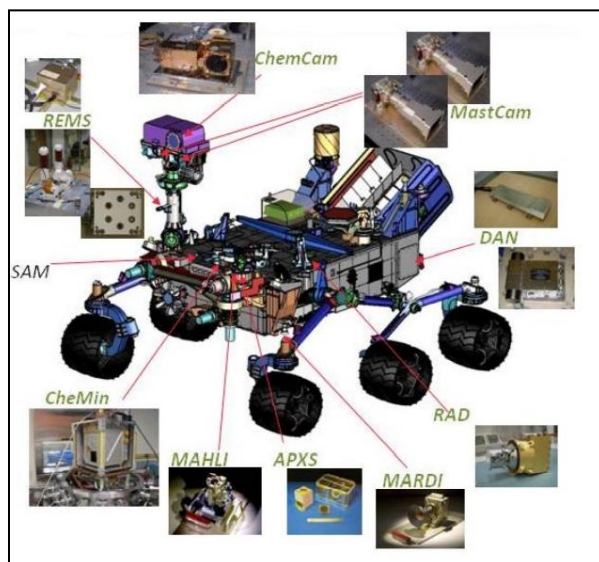


Figure 72. The Curiosity Rover Science Instruments. (NASA, 2020)

The MSL mission was overseen by NASA Associate Administrator for the Science Mission Directorate (SMD), NASA's Jet Propulsion Laboratory was responsible for performing overall system design and integration, and five NASA Centers support the MSL mission along with three foreign government space agencies from Russia, Spain, and Canada.

An overall view of the MSL mission elements (products) and organizations is shown in Figure 73, reflecting a highly complex mission with multiple interfaces, particularly within the Curiosity Rover element.

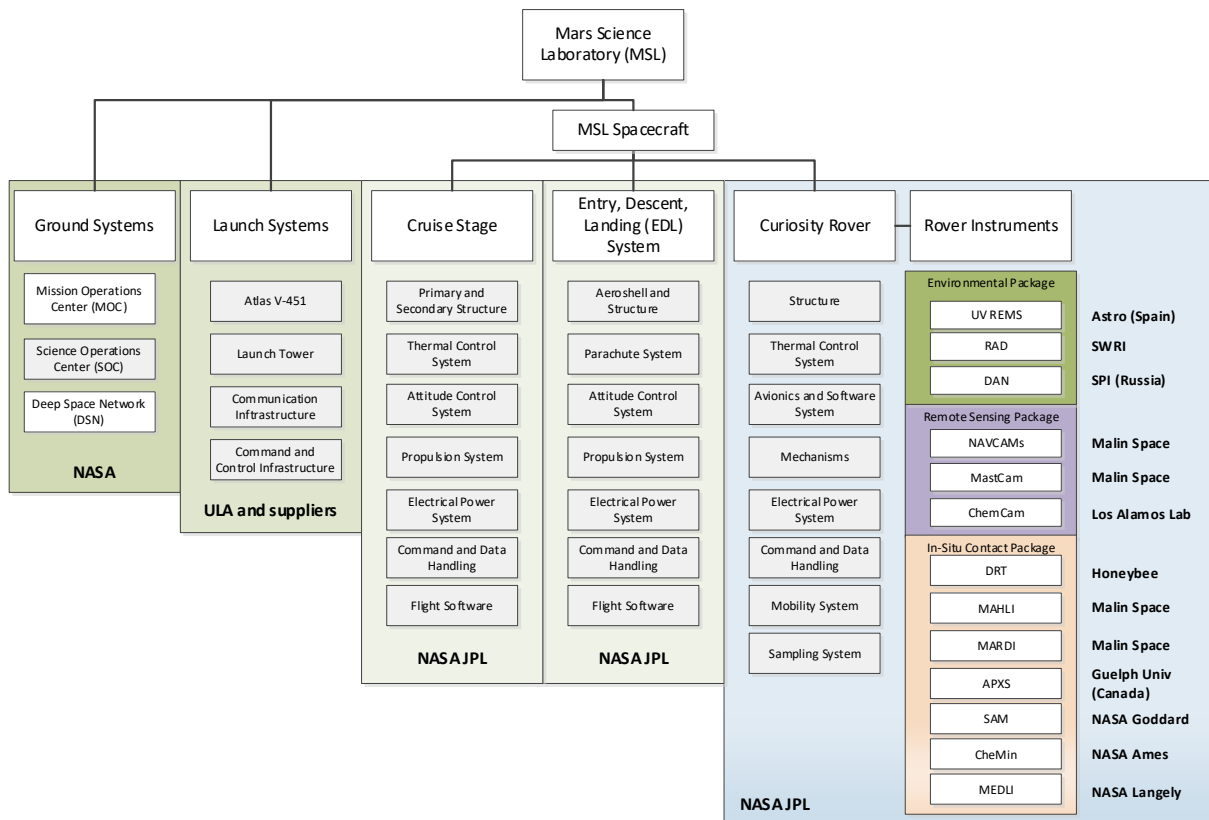


Figure 73. The Overall MSL System Product Structure

4.3.2 MSL Requirements Approach

The requirements imposed on MSL were derived from a dozen science requirements at level 1 (NASA JPL, 2012), and decomposed to levels 2 through 5. MSL requirements were managed using IBM Rational DOORS to support linking to higher/lower level requirements, requirements management, as

well as requirement validation and verification planning and tracking. The system development work was primarily done at JPL, with interface control documents provided for the various rover instruments to the instrument providers, enacting an approach similar to Figure 15, where the requirement handoffs were occurring with the system to instrument providers.

Also noted by Welch, et al, the requirements approach spanned "from the traditional functional requirements levied on each system and subsystem, the environmental requirements managed by the mission assurance office, interface control documents between different system elements, institutional policy documents, as well as stand-alone requirement documents covering topics, for example, planetary protection. While these were based on established institution practice, there was not an architecture that pulled them together in a cohesive manner for MSL. Functional Design Documents (FDD) for each functional area (e.g. thermal control, imaging, fault protection, etc.) provide a detailed description of desired behavior in nominal and off-nominal conditions, command and telemetry associated with function, as well as functional requirements on software. Over 40 FDDs that represent thousands of pages of SE design documentation were developed and maintained through the project life cycle." (Welch, Limonadi, & Manning, 2013).

4.3.3 MSL Outcome

As seen by the continued performance of the Curiosity Rover, the mission appears to be a success. Curiosity continues to perform science observations and provide data to support further understanding of Mars, as well as provide data in support of future Martian missions (such as Mars 2020, the Perseverance Rover, described in <https://mars.nasa.gov/mars2020/>). However, the overall MSL effort had challenges in project execution, as described in various news reports as well as the GAO report IG-11-019.

In early June 2007, the Mars Science Laboratory project completed its project-wide Critical Design Review (CDR), which was intended to establish completion of the project's design phase and transition into the manufacturing phase of the flight hardware. Per a news report, "A key component of the CDR process was a technical risk, programmatic, and cost review, from which multiple independent

cost assessments predicted that this technically challenging \$1.7B planetary science rover mission's current content would cause it to exceed its budgeted development costs to launch by approximately \$75M." (NASA, 2007).

By November 2008 most hardware and software development was complete, and initial testing started. At this point, cost overruns were approximately \$400 million (cost for MSL had increased from the initial \$1.5 billion to \$1.9 billion) (Atkinson, 2008). Per Atkinson, in the attempts to meet the launch date, several planned instruments and mission objectives were removed and other instruments and cameras were simplified to streamline testing and integration of the rover, resulting in a reduction of technical capabilities. The next month, NASA delayed the launch to late 2011 to extend its testing time (NASA, 2008).

Eventually the costs for developing the rover reached \$2.47 billion, for a rover that initially had been classified as a medium-cost mission with a maximum budget of \$650 million. However, NASA still requested an additional \$82 million to meet the planned November launch. (Space Policy Online, 2012). The GAO report IG-11-019, NASA's Management of the Mars Science Laboratory Project, highlighted the cost and schedule overruns, highlighted in Figure 74 and Figure 75.

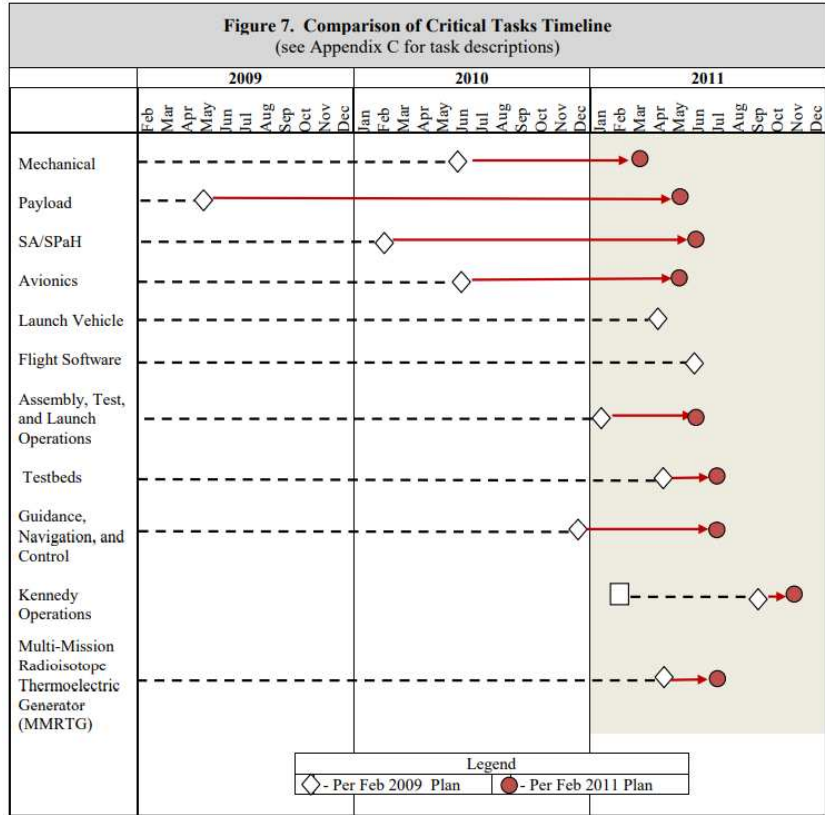


Figure 74. MSL Critical Task Delays. (GAO, 2011)

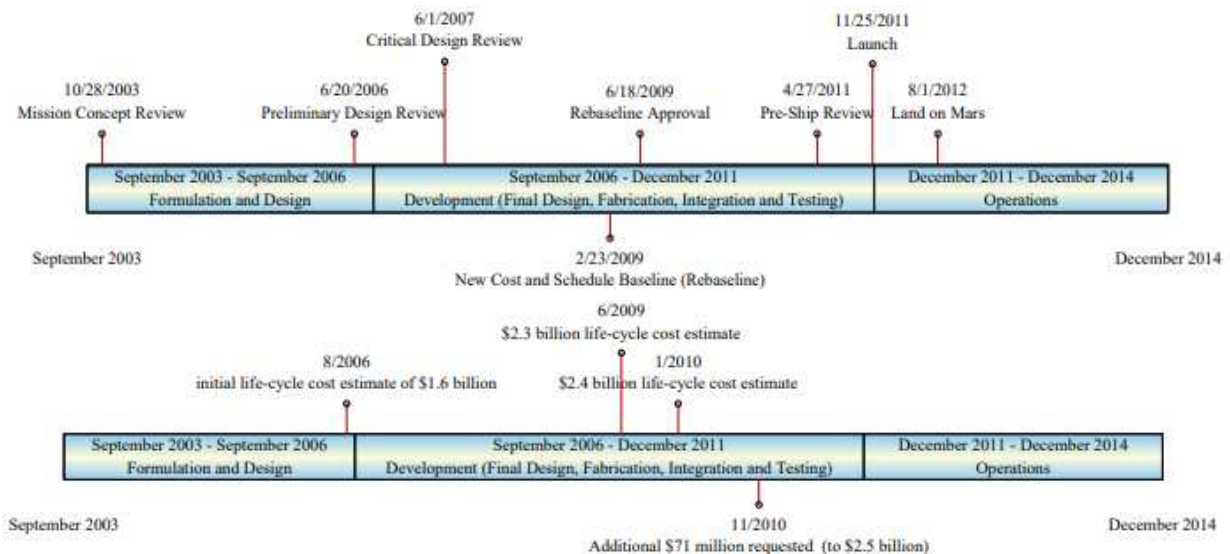


Figure 75. MSL Budget and Schedule Overview. (GAO, 2011)

Based on the above information, the following assessment in Table 13 reflects the total outcome of the MSL product, highlighting how the development challenges affected the project's final capabilities, and resultant cost and schedule.

Table 13. MSL Project Outcome

Parameter	Assessment	Project Success?
Cost	Exceeded budget by 84%	Not successful
Schedule	Exceeded schedule duration by 40%	Not successful
Technical	Mission objectives reduced to meet schedule, successfully met the re-baselined objectives	Success
Customer Satisfaction	Customer is the science community and the taxpayers who funded the mission Science community considered this successful, taxpayer community (reflective of GAO) has mixed response on value achieved	Moderately successful

With respect to the costs associated with the systems engineering requirements, the COSYSMO calculation is based on requirement count calculated in the NASA JPL report on right sizing requirements (NASA JPL, 2012). The new development aspect of the MSL system was the MSL spacecraft. The total number of MSL spacecraft requirements is 511, with an observation from the report that 309 were viewed as low quality requirements (described further in the next section); a COSYSMO factor of "difficult" is used for those requirements, and the rest are classified as nominal. The estimated systems engineering labor for MSL is shown in Table 14.

Table 14. MSL Calculated Labor Costs from COSYSMO.

Project	# of System Requirements	Estimated Systems Engineering Labor	Project Notes
MSL	511 (309 of these rated difficult)	747 person months	Project was moderately successful, further optimization could have ensured it met cost and schedule objectives.

4.3.4 Assessment of MSL Requirements Approach

Per Welch, et al, "it became difficult to maintain cognizance over the full requirements set given the diversity of approaches and turnover of staff (which sometimes led to changes in requirements

management approaches). The consequences were often felt late in development when divergent requirements were discovered during test resulting in late design changes. Flight software requirements were a challenge given the complicated and diverse system behaviors that are embodied in software. Compared to past projects, this was an area MSL did very well in, with the software design well documented and controlled. Without this strong FDD requirements approach, keeping track of and verifying the complex software on MSL would have been a much more challenging task." (Welch, Limonadi, & Manning, 2013). The following items were captured by Welch as lessons learned from the requirements management approach implemented on Rover:

- More upfront effort is needed to understand and architect the breadth of requirements processes and products across the project and a clear flow of parent requirements to target subsystems.
- More formal training and detailed documentation on the project specific approach is needed. Although most systems engineers involved had previous experience in requirements development and management, inevitably their experiences were different. Better training material would have also helped over the entire project life cycle given staff turnover.
- More attention to verifiability of high-level functional requirements is needed. Systems engineers with limited verification and validation experience did not fully appreciate the implication that they would eventually be tasked with verification of requirements and therefore did not always consider verifiability during requirement definition.
- Better documentation of analyses that led to requirements is needed. For instance, mechanism life requirements were based on mission use cases but those analyses were not formally documented in all instances. When time came to verify these requirements, their basis was not understood and needed to be re-analyzed to ensure they were still consistent.

In a review of the 511 MSL spacecraft requirements, only 121 were evaluated as quality requirements in a study conducted by the NASA JPL Office of Chief Engineer (NASA JPL, 2012). The others were assessed as follows:

- Statement of work or process-based (108)
- Implementation specific (10)
- Redundant (39)
- Ambiguous (41)
- Unverifiable (12)
- Lower level of abstraction (217)
- Interface detail appropriate for ICD (49)
- Not actually a requirement, reference information (38)

If all MSL spacecraft requirements were of high quality, then the "nominal" COSYSMO weighting would have been used, yielding an estimated labor of 238 labor months (one third the amount calculated for MSL, which was 747 person months). This could be even lower as likely less requirements would exist after removal of duplicates. Even if 4-6 months of work was spent achieving requirement quality, the savings overall will still be over 50% of the cost of systems engineering labor. Based on input from the systems engineering team, and the assessment of the requirement quality, this project would have benefitted from an optimized requirements management process.

4.4 Space Project Example 3: GOES-R Weather Satellites

4.4.1 GOES-R Series Overview

The Geostationary Operational Environmental Satellites (GOES) program is a collaborative development and acquisition effort between the National Oceanic and Atmospheric Administration (NOAA), and NASA. GOES-R is the third generation GOES spacecraft series that provides critical atmospheric, hydrologic, oceanic, climatic, solar, and space data, as well as communication services such as search and rescue signals. The definition and requirements phase of the project started in 2000 to develop a series of four spacecraft, designated GOES-R, S, T and U, that would support a two satellite constellation over the United States with two on-orbit spares. The first spacecraft, GOES-R (shown in Figure 76), launched on November 19, 2016 on an Atlas V-541, and provided the first update in sensing technology since the GOES-I spacecraft launched in 1994 (NOAA/NASA, 2017).

The goals of the GOES-R mission are (eoPortal, 2020):

- Maintain continuous, reliable operational environmental, and storm warning systems to protect life and property
- Monitor the Earth's surface and space environmental and climate conditions
- Introduce improved atmospheric and oceanic observations and data dissemination capabilities (increased spatial, temporal and spectral resolution)
- Develop and provide new and improved applications and products for a wide range of federal agencies, state and local governments, and private users.

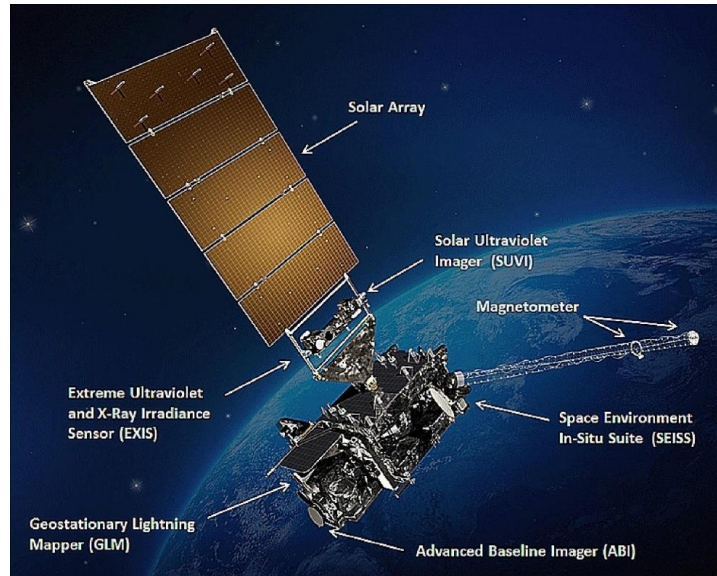


Figure 76. GOES-R Spacecraft. (eoPortal, 2020)

The GOES-S spacecraft launch on March 1, 2018. Upon launch, GOES-R and S were re-designated GOES-16 and GOES-17 (NOAA, 2020). In May 2018, NOAA announced that the GOES-17 spacecraft suffered a malfunction in its instrument cooling system, causing redesign of the subsequent spacecraft in development, GOES-T and GOES-U. The GOES-T launch date is now scheduled for December, 2021.

While NOAA is responsible for GOES-R program funding and overall mission success, it implemented an integrated program management structure with NASA for the GOES-R program since it relied on NASA's acquisition experience and technical expertise. The NOAA-NASA Program Management Council is the oversight body for the GOES-R program, with the program office located at NASA's Goddard Space Flight Center. Overall mission design efforts were overseen by NOAA-NASA team, with the integration and test of the spacecraft overseen by Lockheed Martin (Figure 77).



Figure 77. GOES-R Spacecraft Integration. (eoPortal, 2020)

Figure 77 provides the graphical depiction of its product structure, along with the responsible organizations for each element.

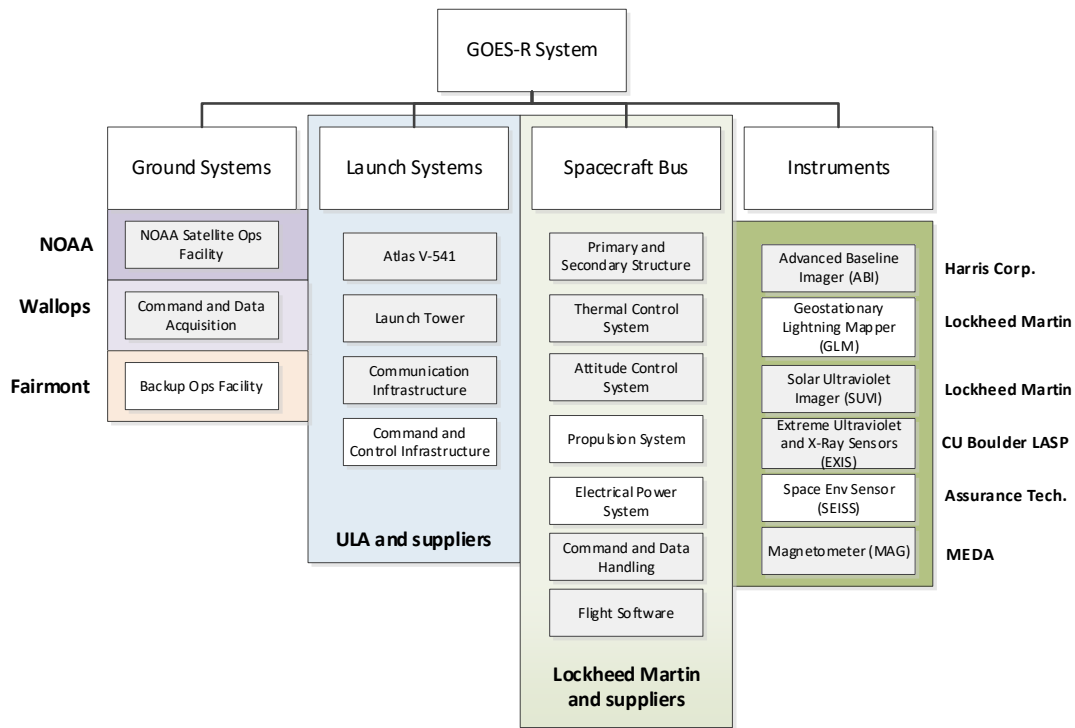


Figure 78. The GOES-R Series System Product Structure.

4.4.2 GOES-R Requirements Approach

All GOES-R Series requirements are derived from the NOAA Consolidated Observation Requirements List (CORL), which documents and prioritizes observational requirements across all NOAA Programs. The NOAA Observing Systems Council coordinates annual updates of the CORL and performs two functions:

- Allocation of user identified observing requirements to the appropriate NOAA observing system program office
- Verification that the observing systems are consistent with NOAA’s existing and planned Observing Systems Architecture

Requirements for the GOES-R Series Program have been assigned to two distinct groups: programmatic and technical. GOES-R Series Level I Requirements, documented in the GOES-R LIRD are the user/science requirements that the CORL allocated to the GOES-R Series Program which serve as the supervisory requirements document for the Program. All other requirements documents flow down from the Level I documents. Figure 79 shows the requirements tree for the GOES-R program.

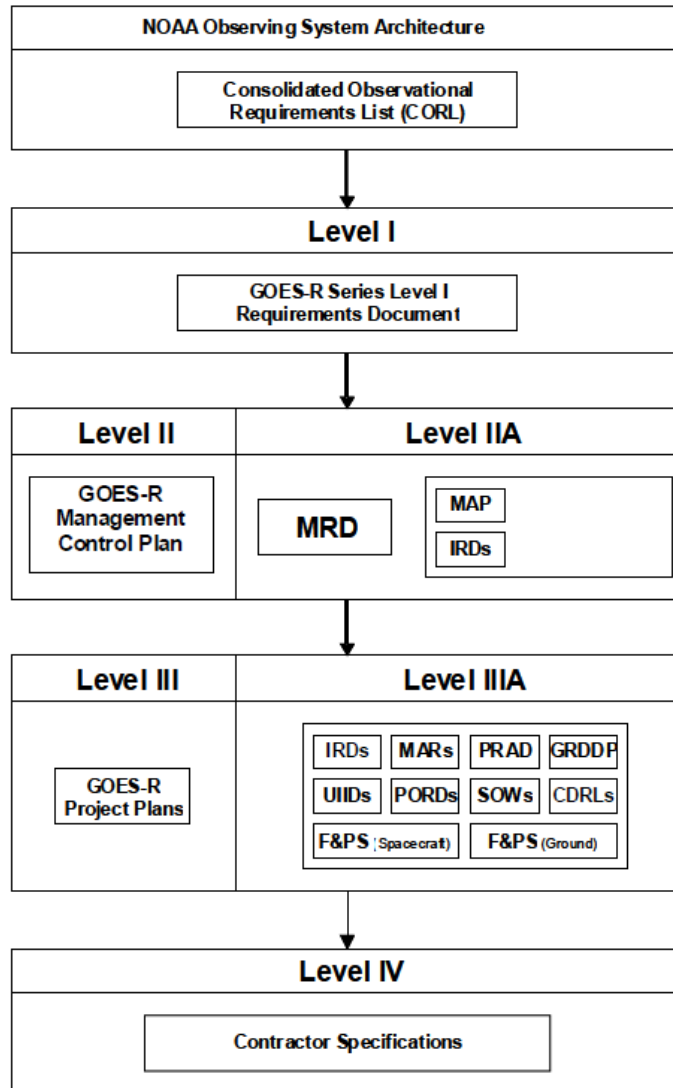


Figure 79. The GOES-R Requirements Tree. (NOAA/NASA, 2020)

The Program-level technical Mission Requirements Document (MRD 410-R-MRD-0070) derives Level I requirements into Level IIA mission engineering requirements used to acquire the GOES-R Series System. GOES-R Level IIA also included the mission assurance requirements and interface requirements documents for the different elements. Level III subsystem, element and interface requirements documents for the flight and ground segments are derived from the MRD. Some example Mission Requirements Document (MRD) requirements are shown in Figure 80.

MRD593	The GOES-R Space Segment shall employ a SUVI instrument with an 8.4 year Mean Mission Duration (MMD) at the end of 10 years, or equivalently a reliability of 0.6 after 10 years of on-orbit operations, preceded by up to 5 years of ground storage and up to 5 years of on-orbit storage. (CCR 02115)
MRD2143	The GOES-R Ground Segment shall implement a ground processing algorithm for the Lightning Detection: Hemispheric product. (CCR 02115)
MRD2144	The GOES-R Ground Segment shall implement a ground processing algorithm for the Solar Imagery: X-ray product. (CCR 02115)
MRD2145	The GOES-R Ground Segment shall implement a ground processing algorithm for the Energetic Heavy Ions product. (CCR 02115)
MRD775	The GOES-R System shall simulate operational activities with high fidelity. (CCR 02115)

Figure 80. Example GOES-R MRD Requirements. (NASA, 2012)

GOES-R requirements were managed using IBM Rational DOORS to support linking to higher/lower level requirements, requirements management, as well as requirement validation and verification planning and tracking. Multiple organizations were involved in the development effort, enacting an approach similar to Figure 15, where the requirement handoffs were occurring with the system to instrument providers, and from system to spacecraft to spacecraft component providers.

4.4.3 GOES-R Outcome

GOES-16 is operating nominally, and between both GOES-16 and 17 the NOAA weather objectives are being met. Based on this, so far the GOES-R program mission appears to be a success. However, the overall GOES-R effort had challenges in project execution, as described in various news reports as well as the GAO report GAO-15-60.

From the GAO assessment, "Since its inception, the GOES-R program has undergone several changes in cost and scope. As originally envisioned, GOES-R was to encompass four satellites hosting a variety of advanced technology instruments and providing 81 environmental products. The first two satellites in the series (called GOES-R and GOES-S) were expected to launch in September 2012 and April 2014. However, in September 2006, NOAA decided to reduce the scope and technical complexity of the GOES-R program because of expectations that total costs, which were originally estimated to be \$6.2 billion, could reach \$11.4 billion. Specifically, NOAA reduced the minimum number of satellites from four to two, cancelled plans for developing an advanced instrument (which reduced the number of

planned satellite products from 81 to 68), and divided another instrument into two separate acquisitions. The agency estimated that the revised program would cost \$7 billion and kept the planned launch dates unchanged." (GAO, 2014). Subsequently, NOAA would change scope of the program several more times, adding and removing capabilities, and adding two on-orbit spare spacecraft back into the project scope. Figure 81 provides a summary in the GAO 2014 report showing the variation of the program scope over time.

Table 2: Key Changes to the Geostationary Operational Environmental Satellite-R Program over Time

	August 2006 (baseline program)	September 2006	November 2007	February 2011	August 2013
Number of satellites	4	2	2	4	4
Instruments	<ul style="list-style-type: none"> Advanced Baseline Imager Geostationary Lightning Mapper Magnetometer Space Environmental In-Situ Suite Solar Imaging Suite (which included the Solar Ultraviolet Imager, and Extreme Ultraviolet/X-Ray Irradiance Sensor) Hyperspectral Environmental Suite 	<ul style="list-style-type: none"> Advanced Baseline Imager Geostationary Lightning Mapper Magnetometer Space Environmental In-Situ Suite Solar Ultraviolet Imager Extreme Ultraviolet/X-Ray Irradiance Sensor 	No change	No change	No change
Number of satellite products	81	68	34 baseline 34 optional	34 baseline 31 optional	34 baseline 31 optional
Life cycle cost estimate (in then-year dollars)	\$6.2 billion – \$11.4 billion (through 2034)	\$7 billion (through 2028)	\$7.67 billion (through 2028)	\$10.86 billion (through 2036) ^a	\$10.86 billion (through 2036)
Estimated launch dates for GOES-R and GOES-S	GOES-R: September 2012 GOES-S: April 2014	GOES-R: September 2012 GOES-S: April 2014	GOES-R: December 2014 GOES-S: April 2016	GOES-R: October 2015 GOES-S: February 2017	GOES-R: by March 2016 GOES-S: by June 2017 ^b

Source: GAO analysis of NOAA data. | GAO-15-60

^aBased on NOAA's fiscal year 2012 budget estimate, \$7.64 billion of this cost estimate was for the first two satellites in the series, GOES-R and GOES-S. The cost for the remaining two satellites—GOES-T and GOES-U—was estimated at \$3.22 billion.

^bProgram documentation shows that the launch commitment dates were changed to the first quarter of 2016 and the second quarter of 2017, respectively. The launch dates in this chart reflect the latest month in which launch can occur and still meet the launch commitment dates.

Figure 81. Changes to the GOES-R Program Over Time. (GAO, 2014)

The GAO report also highlighted the cost and schedule overruns, shown in the last two rows of Figure 81. The figure shows a comparable life-cycle cost value, so further review of the development costs are provided in the U.S. Department of Commerce report, OIG-13-024-A. The report noted that the overall life cycle budget was expected to be within, but the near term allocation budget in 2013 and 2014 increased its budget plan by \$186M in 2013 and \$150M in 2014, and adjustments were required to ensure

cost growth rate substantially reduced (development costs were exceeded by 18% in those two years, after already receiving budget adjustments of \$264 in 2012 due to re-baseline activities).

A finding of the Department of Commerce audit includes the following observation: "Program systems engineering has been strengthened; however, early in system development, it contributed to ground system schedule compression and increased costs. To ensure continued strength, NOAA must report on the adequacy of program systems engineering (including National Aeronautics and Space Administration support) for the integration and test phase of the program." (U.S. Dept of Commerce, 2013).

Based on the above information, the following assessment in Table 15 reflects the total outcome of the GOES-R project.

Table 15. GOES-R Project Outcome

Parameter	Assessment	Project Success?
Cost	Maintained life cycle budget, allocated development budget exceeded by 18%	Moderately successful
Schedule	Exceeded schedule duration by 33%	Not successful
Technical	Mission objectives reduced to meet schedule, successfully met the re-baselined objectives, one satellite has major anomaly.	Moderately successful
Customer Satisfaction	Customer is the weather community and the taxpayers who funded the mission Weather community considered this successful, taxpayer community (reflective of GAO) has mixed response	Moderately successful

With respect to the costs associated with systems engineering due to requirements, the COSYSMO calculation is based on requirements obtained from the mission requirements document (330 requirements) and ten interface requirements documents (~100 requirements in each). This system used an existing launch vehicle, so the launch vehicle interface control requirements are not included in this calculation. The total number of Level III system requirements is approximately 1330, with an observation that several dozen requirements in the MRD are at the wrong level, ambiguous, or

unverifiable. A COSYSMO factor of "difficult" is used for those requirements, and the rest are classified as nominal. The estimated systems engineering labor for GOES-R is shown in Table 16.

Table 16. GOES-R Calculated Labor Costs from COSYSMO.

Project	# of System Requirements	Estimated Systems Engineering Labor	Project Notes
GOES-R	~1300 (50 of these rated difficult)	643 person months	Project was moderately successful, further optimization could have ensured it met cost and schedule objectives.

4.4.4 Assessment of GOES-R Requirements Approach

Looking at the requirements applied on GOES-R, there is a mix at Level 1 between programmatic and technical requirements. Per the *INCOSE Guide for Writing Requirements*, "It is important to differentiate between requirements on the systems to be developed from the requirements on the organization, people, and program/project responsible for developing those systems. A common issue is mixing the two types of requirements together rather than documenting them separately."(INCOSE, 2019).

The technical requirements were managed in IBM Rational DOORS, and were allocated and decomposed to Levels II through IV (and decomposed further by contractors below that level). Programmatic requirements were addressed through program plans and management functions. Verification activities, such as test, are more applicable to the technical requirements, which is why mixing the two types of requirements at the same levels, as shown in Figure 79, is often avoided.

When assessing various MRD technical requirements, with examples shown in Figure 80, there is a mix of well-formed requirements (MRD593), redundant requirements (MRD243, MRD2144, and MRD2145), and vague/unverifiable requirements (MRD775). While the interface requirements documents (IRDs) were not available, there were almost a dozen of them, which increased the overall requirement count. These are assumed as low count, easy requirements based on comparable NASA IRDs; however, the spread of requirements across so many documents incurred work on the contractors to

pull out all relevant and applicable requirements into a comprehensive location (associated efforts for this activity is discussed further in Section 5.5).

Based on input from the systems engineering team, and the assessment of the requirement quality, this project may have benefitted from an optimized requirements management process to streamlined the Level II requirements and ensure a comprehensive set is applied to the flight and ground segments.

4.5 Space Project Example 4: NASA Constellation

4.5.1 Constellation Overview

In February 2004, under the Bush Administration and NASA Administrator Sean O'Keefe, NASA released the *Vision for Space Exploration (VSE)* to address the state of human spaceflight at NASA and invigorate the public enthusiasm for space exploration. In September 2005, new NASA Administrator Michael Griffin outlined an initial architecture for implementing the VSE in its *Exploration Systems Architecture Study (ESAS)*, presenting programmatic and technical details of the Constellation Program, and by late 2005 the *NASA Authorization Act of 2005* was released, directing NASA to establish a program to develop a sustained human presence on the Moon. This authorization and budget led to a several year, multi-billion dollar NASA effort to implement the Constellation Program.

The purpose of the Constellation Program was to develop an effective, safe, affordable, reliable, and sustainable architecture to conduct human exploration across the solar system. This effort spanned all of the NASA space centers and employed thousands of NASA and aerospace industry personnel. The best lessons from the shuttle were applied, rigorous system engineering approaches were utilized, and the workforce was committed to bring Americans back to the moon and beyond.

The Constellation Program elements, shown in Figure 82, included the crew and cargo launch vehicles (Ares I and Ares V), the Crew Exploration Vehicle (CEV) Orion, the Altair Lunar Lander, and the Earth Departure Stage secondary booster.

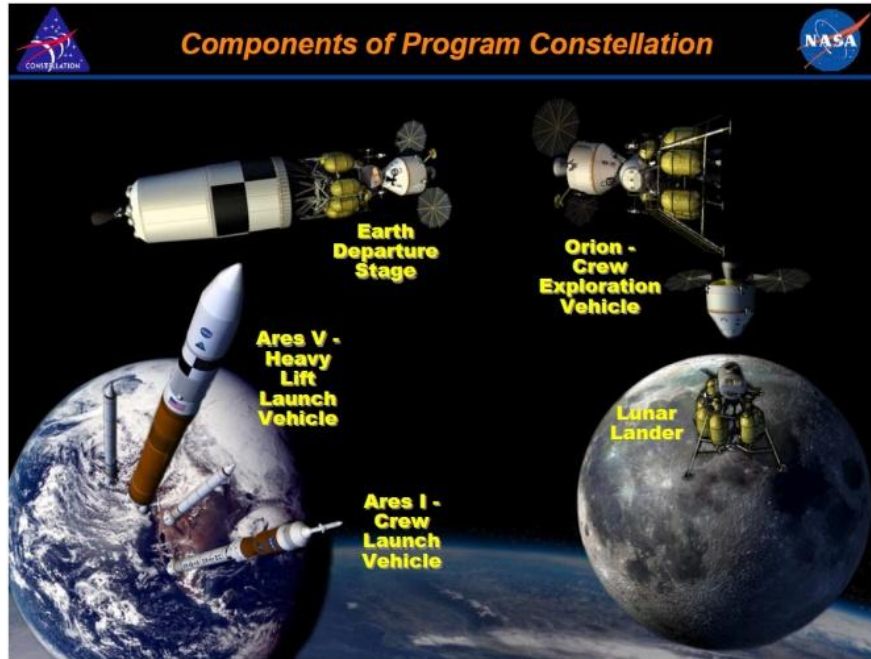


Figure 82. The Constellation Program Elements. (Connolly, 2006)

The customer of the program was the United States Government, which provided the charter and funding to NASA headquarters. An indirect stakeholder to the program was the United States taxpayer base, who provided funding for the program and had the ability to vote in new members of the government. A variability from the customer perspective was the administration of the current President, which impacted the requirements and objective of the program, as well as the funding profile. The program itself was made up of the following for enterprise, program, and project teams:

- NASA headquarters represented the overall mission owner, it provided direction to the architecture level (Level II), and would be considered the "enterprise" level.
- The program was executed from the Level II architecture effort and activities, this is comparable to the system level.
- The elements at Level III and IV were subordinate to the Level II, these were considered projects. These projects could be considered for optimization by their project leads, however the entire Constellation Program would ultimately need to balance across the different projects to be considered optimized.

Subject matter experts within NASA were considered stakeholders for the myriad of technical topics that they participated in within the program and project teams. In many cases, these experts were provided ability to approve, or disapprove, various program documentation and implementations.

Figure 83 provides the graphical depiction of its product structure, along with the responsible organizations for each element.

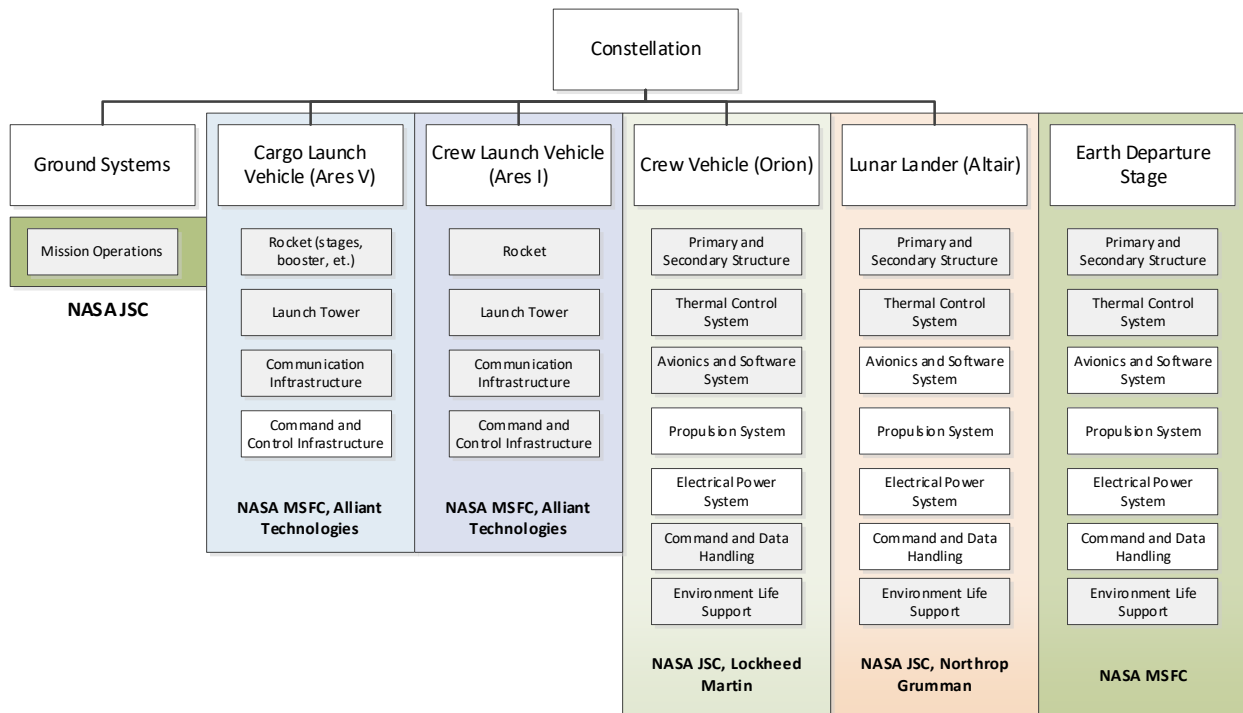


Figure 83. The Constellation System Product Structure.

The Constellation Program continued development efforts through 2009, when newly elected President Barack Obama and his administration formed the "Review of United States Human Space Flight Plans Committee" (also known as the Augustine Commission), with the charter to "conduct an independent review of ongoing U.S. human space flight plans and programs, as well as alternatives, to ensure the Nation is pursuing the best trajectory for the future of human space flight – one that is safe, innovative, affordable, and sustainable." (NASA, 2009). The Committee concluded that the Constellation Program was so far behind schedule, underfunded and over budget that meeting any of its goals would not be possible. The *NASA Authorization Act of 2010* ultimately removed funding from the Constellation Program, resulting in its cancellation.

4.5.2 Constellation Requirements Approach

The Constellation top level documentation hierarchy is shown in Figure 84, showing the foundational Level II documents that were levied to ensure requirements, processes and standards were consistently applied and followed at all levels of the program.

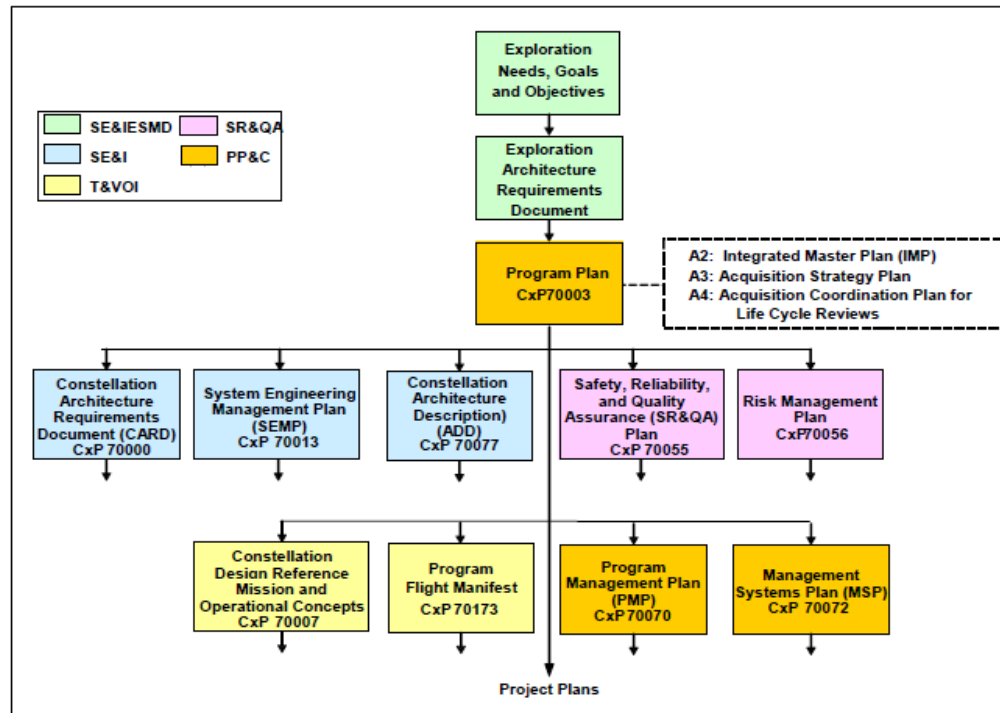


Figure 84. Constellation Top Level Documentation. (NASA, 2010)

An analysis of program needs was done to generate a set of requirements for Constellation, which were iterated through requirement analysis cycles (RACs) and Integrated Design Analysis Cycles (IDACs) to derive, refine and validate their feasibility. The Constellation system engineering management plan (SEMP) described this process using the graphic shown in Figure 85.

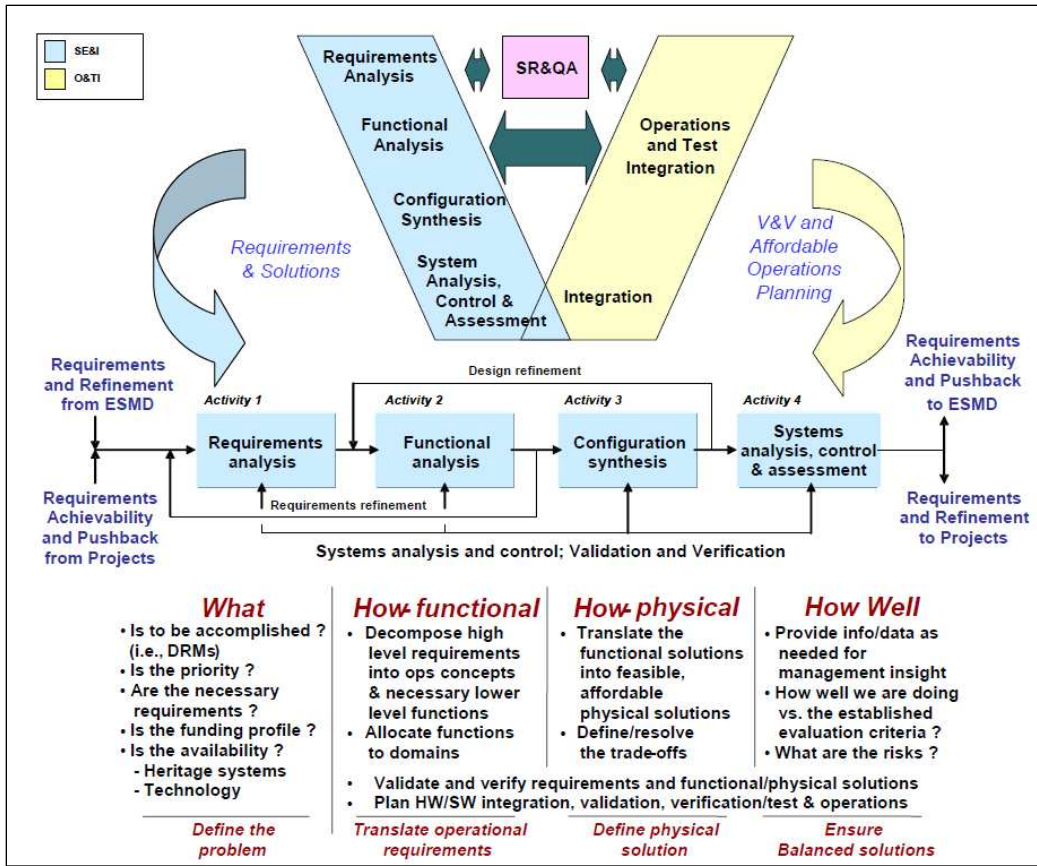


Figure 85. Constellation Requirements Development and Refinement Process. (NASA, 2010)

The requirement work culminated in release of the *Constellation Architecture Requirements Document (CARD)*, document CxP 70000, which defined requirements controlled by the Constellation Program for the hardware, software, facilities, personnel and services needed to perform the Design Reference Missions (DRMs). The CARD contained Level I needs and objectives as well as Level II program requirements. These requirements were managed in the requirements management tool 3SL Cradle (<https://www.threesl.com/cradle/>), which provided traceability from the top level requirements to the lower levels. Example requirements from the CARD are shown in Figure 86.

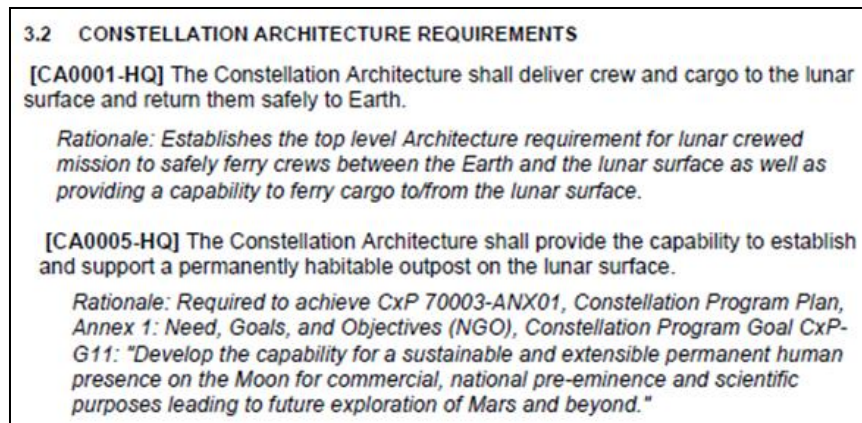


Figure 86. Constellation Architecture Requirement Examples. (NASA, 2008)

The primary emphasis of the requirement development was to derive the “right” requirements for the architecture and each system, which were necessary to safely accomplish the mission's top level functional objectives and that could be satisfied by a physical design solution within technology, budget and schedule constraints. In addition to the CARD, the Program created design and construction standards and specifications for subject-specific requirements for Human-Systems Integration and Safety, Reliability and Mission Assurance. The standards were defined based on best practices and lessons learned from past programs, created by subject matter experts for their topic areas, and were allowed to be tailored by the lower levels of the program. The Constellation Program high level requirements tree is summarized in Figure 9 and includes the following focus areas:

- Architecture and system functional and performance requirements (CARD)
- Design and construction (D&C) standards and specifications (multiple documents)
- Interface requirements (multiple documents)

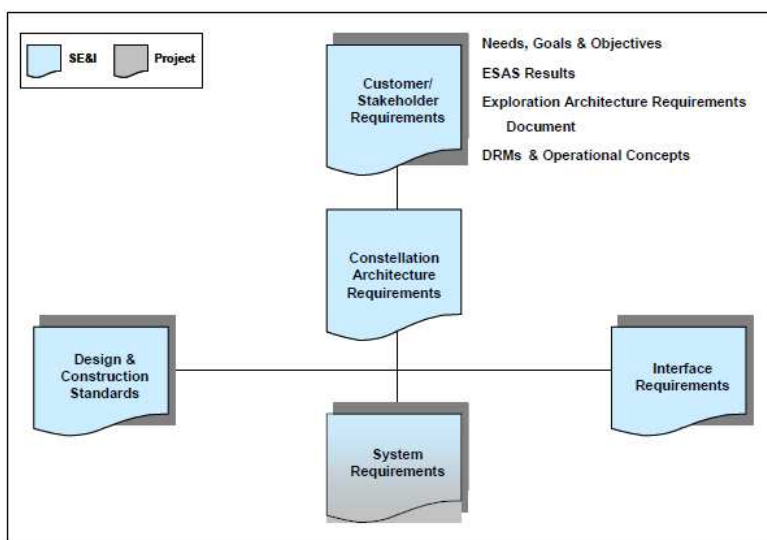


Figure 87. The Constellation High Level Requirement Tree. (NASA, 2010)

4.5.3 Constellation Project Outcome

The initial ESAS study in 2005 predicted an estimated total cost of \$124 Billion through the first lunar landing (FY06–FY18). NASA initiated the Constellation program in November 2005 and expected the program to enter implementation in 2009. By 2009 the Government Accountability Office (GAO) noted that "Over \$10 billion has already been obligated and NASA budget estimates indicate that over \$97 billion is to be spent on the Constellation program through 2020". Per the GAO report, "the Program has delayed its entry into implementation, however, and is still modifying its overall architecture and specific requirements." (GAO, 2009).

While the overall predicted cost did not change, the progress towards the end date was not being met, leading to uncertainty for the final completion date of the project. Additionally, government funding was continuously reduced by Congress as time progressed. As noted in a Constellation Program lessons learned report NASA/SP-2011-6127-VOL-1, "Funding for the Constellation Program was inconsistent and unreliable from its initial formulation through its cancellation...While Constellation was expected to transport crews to and from ISS soon after space shuttle retirement, the funding ramp-up needed for development was not available until after the space shuttle was retired." (NASA, 2011). Figure 88 shows the final funding profile for Constellation over the life of the program against the initial baseline

predictions (ESAS study) as well as against the Program Manager's Recommendation (PMR) and the President's Budget Submittal (PBS); it was asserted in the lessons learned report that the lack of available budget had significant impact in the development of the project. Both the slowdown in funding, and the approach used towards project management, led to the project delays shown in Figure 89.

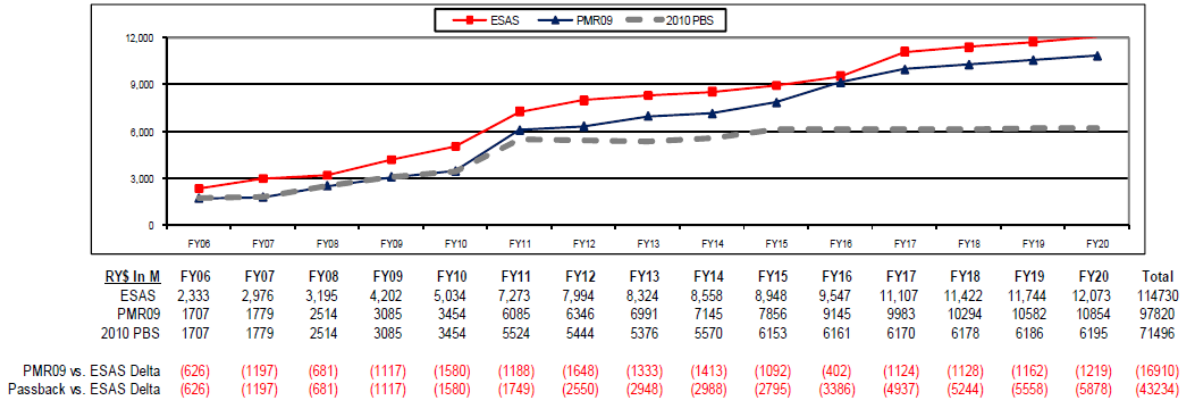


Figure 88. Constellation Budget Profiles. (NASA, 2011)

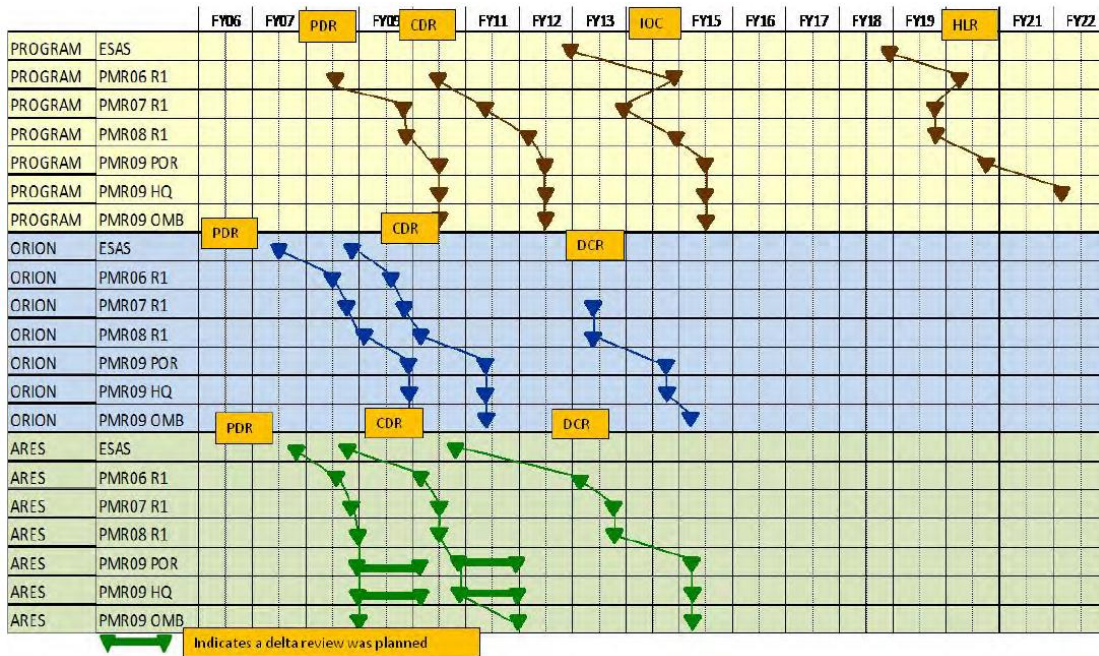


Figure 89. Schedule Delays on Constellation Over First Four Years. (NASA, 2011)

Over the course of the Program several reviews of Constellation were performed, and multiple external organizations implemented their own studies of the Program (reference Table 17).

Table 17. Assessments of the Constellation Program.

Agency	Event
GAO	GAO releases report GAO-06-218 recommending improvements to NASA's system engineering policy, specifically noting technology readiness should be demonstrated before going to the implementation phase, and design stability before transitioning to the fabrication stage (Dec. 2005).
GAO	GAO releases report GAO-06-817R highlighting that NASA's lack of sound business case for the CEV placed the project at risks for overruns (Jul. 2006).
GAO	GAO releases report GAO-06-218 recommending funding for CEV only until PDR based on high level of execution risk, noting Requirement Knowledge Gaps, Technology and hardware development knowledge gaps, aggressive schedule, projected funding shortfalls (Oct. 2007).
GAO	GAO releases report GAO-09-844 recommended a sound business case for Constellation, warning of budget and schedule problems, noting " NASA is still struggling to develop a solid business case—including firm requirements, mature technologies, a knowledge-based acquisition strategy, a realistic cost estimate, and sufficient funding and time—needed to justify moving the Constellation program forward into the implementation phase. " (Aug. 2009).
Constellation Program Acceleration Study	Constellation Program Acceleration Study Team releases the <i>Constellation Acceleration Study Report</i> to identify options to accelerate Constellation schedules to minimize gap in US human spaceflight post shuttle retirement, released as a response to concerns for Program cancellation (Dec. 2008).
Aerospace Technology Working Group	Aerospace Technology Working Group releases an independent assessment of the Constellation Program, recommending a Unified Strategic Vision (USV) as a replacement for the Bush Administration VSE, noting that space industrialization and promoting human commercialism in space was the best way to proceed in establishing a space program (Feb. 2009).
Augustine Commission	Obama Administration charts the Review of U.S. Human Spaceflight Committee (Augustine Commission) to do a review of the Constellation Program, where differences between approach to implementing Apollo to current times were highlighted, including the increase of the commercial options for developing space infrastructure (Oct. 2009).

After the Program cancellation several members of the NASA team generated a lessons learned document, NASA/SP-2011-6127-VOL-1. This document notes that the technical challenges were actually not insurmountable, but "the most difficult and most persistent challenges involved cost, schedule, and organization." This lesson learned document highlights various challenges, including (NASA, 2011):

- The top heavy organization structure, noting that "When everybody is responsible for everything, nobody is responsible for any one thing". It highlighted a need for a clear establishment of roles, responsibilities and accountability that could evolve over the project as it moves along the life cycle phases.
- The usage of the multiple NASA Centers, each with different cultures, approaches, differing standards of requirements for development and production, and lack of clear understanding that the tasks were going to the appropriate places and not duplicative.

- The need to allow a comprehensive approach to the design of the architecture by the developers, compared to a top down mandate on how the designs will be implemented. This was realized with Orion when it was too heavy based on the large number of design standard requirements imposed on it.
- The Program allowed for a large number of requirements to be levied without taking into account ability of many contractors to already meet these requirements through other means - forcing them to spend time responding to each of the "shall" statements.

Based on the above information, the following assessment in Table 18 reflects the total outcome of the Constellation project.

Table 18. Constellation Project Outcome

Parameter	Assessment	Project Success?
Cost	Budget was spent but progress not made (value was not earned)	Not successful
Schedule	Exceeded schedule duration by 50% to 100%	Not successful
Technical	Mission objectives reduced to meet schedule, did not go past the CDR phase.	Not successful
Customer Satisfaction	Customer is the Government and the taxpayers who funded the mission; neither had their objectives met.	Not successful

With respect to the costs associated with systems engineering due to requirements, the COSYSMO calculation is based on requirements at Level II, which involve the CARD as well as documents that it invokes, shown in Table 19.

Table 19. Total Constellation Level II Requirement Count

Source	# of Docs	Total # of Reqts	Nominal	Difficult
CARD	1	1800	1260	540
Human Integ. Standard	1	800	720	80
Safety Requirement Standards	5	1000	900	100
Env. and Test Standards	4	1000	900	100
Design and Const Standards	14	1400	1260	140
IRDs	26	2600	2340	260
Totals:		8600	7380	1220

The estimated systems engineering labor for Constellation is shown in Table 20.

Table 20. Constellation Calculated Labor Costs from COSYSMO.

Project	# of System Requirements	Estimated Systems Engineering Labor	Project Notes
Constellation	~8600 (1220 of these rated difficult)	6109 person months	Project Cancelled in Design Phase

While the Constellation Program was officially cancelled, several elements were revived and continue development to this day.

- NASA continues development of the Orion spacecraft for deep space travel, managed by NASA JSC and contracted to Lockheed Martin for design, development, test and evaluation efforts (DDT&E). In a move to reduce costs, NASA has contracted for private development of vehicles for use in low Earth orbit, leaving missions to the International Space Station as part of the separate Commercial Crew Development program.
- The Ares launch vehicle project was transformed to the Space Launch System (SLS), a Space Shuttle derived super heavy-lift expendable launch vehicle that will be used in NASA deep space exploration missions.

4.5.4 Assessment of Constellation Requirements Approach

The Constellation Program was vastly more complex than the prior space project examples in this section, and as it had thousands of requirements at the system level there is an expectation it would take significant engineering labor to implement. In addition to the NASA lessons learned document discussed previously, it is worth exploring the perspectives of those that worked on the project to obtain their insights. Below are some inputs from former NASA project team members with their viewpoint of how requirements management was applied.

Conflict in System Requirements

Mr. John Curry, former Level III Vehicle Integration Manager for Orion and Level II SE&I lead, noted many of the challenges with a multi-level NASA approach. When working on Orion, Mr. Curry had an opportunity to participate in the requirement development effort, and notes that a very thorough effort was done to ensure requirements met standards for quality, completeness, and ability to be verified. However, years were spent on these documents at the Level III effort alone, when time could have been spent developing prototypes or investigating technical options towards the requirements.

One challenge Mr. Curry noted was the difficulty in trying to meet competing sets of requirements related to the Design Reference Missions for lunar and ISS. Each mission design introduced factors that would cause the other design not to be optimized. As example, the vehicle was required to carry 4 people to the moon, yet the NASA Administrator had a goal that it would carry 6 people to ISS. This meant the design had to be bounded for 6, which impacted the lunar mission mass and other parameters. Mr. Curry noted that there were several challenges in pushing back on these requirements; however, once a new NASA Administrator was in place the requirement was able to be updated to remove the conflict.

Mr. Curry also led much of the effort related to creating the "zero based" approach. When Orion was too heavy by several thousand pounds, an assessment showed that applying all of the design standards led to a very mass intense vehicle. The team worked to scale back to the basics of mission success, safety and reliability and all of the other requirements had to "earn their way on", leading to various trades of true needs compared to desired features. This was a course correction that allowed the vehicle to achieve its performance metrics going into its design reviews.

Application of Safety Requirements

In an interview with Ms. Angie Wise, former Level II Program Integration Engineer and Level III Deputy Chief Safety and Mission Assurance (S&MA) Officer for Ares I-X, Ms. Wise noted that the safety, reliability and quality assurance requirements were levied by the *Constellation Program Environment Integrated Safety, Reliability and Quality Assurance (SR&QA) Requirements*, CxP 70059, which she co-authored. This document states that "the programmatic SR&QA requirements set forth in this document complement and support the technical CxP requirements defined in CxP 70000 (CARD) and subordinate documents.". This document was one of numerous that levied additional requirements in support of the CARD.

Early in the program there were thousands of requirements that Ms. Wise had to process to ensure a comprehensive set of safety requirements was levied on the program. Many of these requirements were

a combination of process control, approaches to conducting the safety, reliability and quality disciplines, and requirements on the products. Addressing these requirements was highly labor-intensive, there were twenty people per discipline working to flow the Level II documents down to the Level III effort and then down to Level IV. By the time they reached the contractor level, there was push back on requirements that were seen as highly costly by the contractor to implement.

When asked what prompted these series of requirements, Ms. Wise noted that teams with history of working on the Space Shuttle Program had documented various approaches to addressing specific topic areas based on lessons learned, which were then put into specific standards. NASA Level I opted to include these standards in addition to the mission requirements as a way to enforce best practices. The shuttle program was exempt from these standards as it was already operational, so the first real opportunity to apply these was with Constellation.

When Ms. Wise was developing CxP 70059 she noted that many of the subject matter experts were adamant that their content be included, and as they were given approval rights to the final document they could refuse to sign until they were satisfied their items were in place. These NASA members were highly risk averse and focused on singular topics, it was very difficult to ensure a proper and cohesive final product due to the intractable stance many of them held. Additionally, while these could appear to be good practices to a NASA engineer, the Level IV team responsible for implementation were far removed from the initial generation of the requirements, and therefore unable to give feedback on the feasibility of these requirements to implement. When Ms. Wise had a chance to do similar work on the Ares I-X test rocket, she was able to vastly scale back the requirements, and the technical manager took a stronger approach that every requirement needed to "buy itself into the project". This test rocket was launched in 2009, meeting all test objectives and providing valuable data for the Ares I project.

Over-prescription of Design and Construction Standards

NASA/SP-2011-6127-VOL-1 highlights various challenges for the Constellation program, including the observation that the program allowed for a large number of requirements to be levied

"without taking into account ability of many contractors to already meet these requirements through other means, and forcing them to spend time responding to each of the 'shall' statements levied by these documents" (NASA, 2011). An example of an invoked design and construction standard from the CARD is shown in Figure 90, invoking NASA-STD-6016, the NASA Materials and Processes Standard.

[CA3005-PO] The Constellation Architecture shall comply with NASA-STD-(I)-6016, Standard Materials and Processes Requirements for Spacecraft.

Rationale: This document defines the minimum requirements for manned spacecraft Materials and Processes (M&P) and provides a general control specification for incorporation into NASA program/project hardware procurements and technical programs.

Figure 90. Constellation Required use of a Design and Construction Standard. (NASA, 2008)

NASA-STD-6016 contains about 200 requirements. Noted from Table 19, the additional requirements in the design and construction standards at Level II adds about 1400 requirements to the total requirement count (a 20% increase).

An example requirement from NASA-STD-6016 is shown in Figure 91. Looking at these requirements in more detail, they are a mixture of process requirements (how to perform a design, test, production technique, or analysis), and lower level of abstraction design requirements (specific details on design features that are required, such as wiring). These are applied at the Level II, system level, which is traditionally where high level design-input requirements are levied.

a. [MPR 197] When used with tin-based solder joints in mission-critical hardware, gold shall be removed from at least 95 percent of the surface to be soldered of all component leads, component terminations, and solder terminals.

Figure 91. Example Design and Construction Requirement. (NASA, 2016)

The hidden costs (both financially and impact to innovation) are explored further in Katz's *When to Constrain the Design? Application of Design Standards on a New Development Program*, where the recommended option for ensuring quality of product when innovation is desired is to create quality assurance and performance-based requirements, and provide the different design and construction standards as supporting reference material (Katz, 2020).

Multiple Teams and Disconnect of Requirement Development

In the discussion with Mr. Wheatcraft (Section 3.2.5), he noted that NASA had several groups responsible for requirement development on Constellation, a project he worked previously. This resulted in parallel efforts of requirements development that would occasionally be aligned by management, but not always able to be aligned in a comprehensive and cohesive set of requirements that were maintained for the system in using a document centric requirements effort.

Mr. Wheatcraft also observed that the lower level requirements were generated to establish a contract for some of the products well before the system requirements were fully developed to address the system needs. This had the result of putting a constraint at the system level and reducing various system capabilities.

Mr. Wheatcraft noticed a disconnect in the requirements management processes among the NASA program team, as many were not aware of the project's documented requirement management plan and were implementing processes based on their own knowledge.

Besides the disconnected approach within management of the requirements, Mr. Wheatcraft also noted that the requirements development was not based on an overall system concept or set of needs, but was driven by individual assessments and experiences (most particularly the organization responsible for operations of the space mission), and lacked perspectives of other viewpoints and as well as a system viewpoint.

A take-away from this discussion is that the overall approach towards requirements development and management occurred by many disconnected teams, which drove both the amount of requirements as well as gaps or overlaps in the resulting requirements documents.

Conclusions on Constellation Requirements Management Approach

The assessment from the project outcome, and input from those that had experience on the project, led to the following observations regarding the requirements management approach used on Constellation:

- Too many requirements existed at the system level consisting of those at too low a level of abstraction, design-output requirements, multiple layers of requirements invoking documents; constraining the design and driving engineering cost.
- Requirements at lower elements were levied before higher system requirements were developed, constraining the overall architecture solution.
- Requirement development occurred by many teams working in disconnected efforts, resulting in a discordant requirement set that did not address the actual needs of the system.

The recommended approaches that will be shown in Chapter 5 are expected to address many of these findings and enable cost optimization in a project the scale of Constellation.

4.6 Space Project Example5: NASA Artemis Human Landing System (HLS)

4.6.1 Artemis HLS Overview

On December 11, 2017, President Trump signed Space Policy Directive 1, a change in national space policy that provides for a U.S. led, integrated program with private sector partners for a human return to the Moon, followed by missions to Mars and beyond. The directive instructs NASA to “Lead an innovative and sustainable program of exploration with commercial and international partners to enable human expansion across the solar system and to bring back to Earth new knowledge and opportunities. Beginning with missions beyond low-Earth orbit, the United States will lead the return of humans to the Moon for long-term exploration and utilization, followed by human missions to Mars and other destinations.” Additionally, on March 26, 2019, Vice President Mike Pence announced “It is the stated policy of this administration and the United States of America to return American astronauts to the Moon within the next five years.” (U.S. White House, 2017)

In response to the directive NASA has generated the Artemis program. "With the Artemis program, NASA will land the first woman and next man on the Moon by 2024, using innovative technologies to explore more of the lunar surface than ever before. We will collaborate with our commercial and international partners and establish sustainable exploration by the end of the decade. Then, we will use what we learn on and around the Moon to take the next giant leap – sending astronauts to Mars." (NASA, 2020).

Artemis is building upon the remnants of the Constellation program, utilizing the Space Launch System (SLS) launch vehicle and the Orion Crew Exploration Vehicle (CEV) as part of the architecture. Additional development includes the Gateway space station, the lunar Human Landing Systems (HLS), Spacesuits, and Exploration Ground Systems. An overview of the Artemis mission concept is shown in Figure 92.

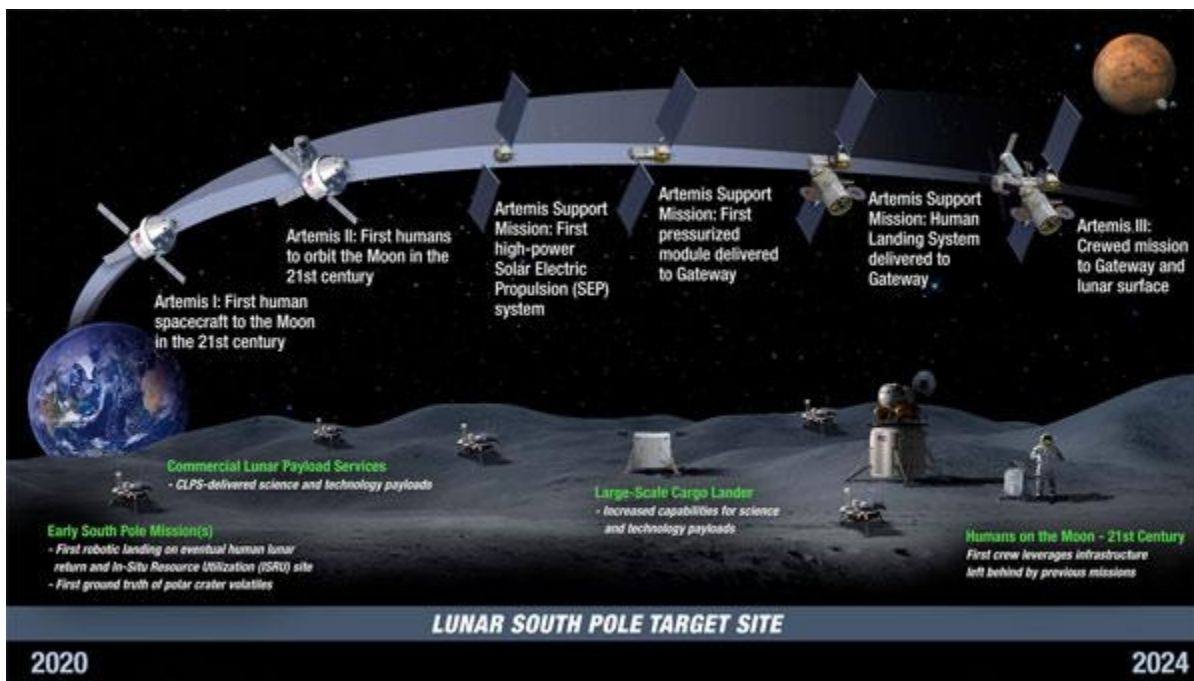


Figure 92. The Artemis Program Mission Overview. (NASA, 2019)

At first glance, this appears very similar to the Constellation program. Looking at the project elements, there are comparable elements in place. However, the Artemis approach towards program acquisition strategy is different, as it focuses on procuring commercially developed elements that meet mission needs.

Per NASA Administrator Jim Bridenstine, "We have to remember that we already have SLS, Orion, and the European Service Module well underway. Those are three of the biggest components to getting humans to the Moon, and we're on the brink of being ready with those programs. When we talk about what we need, we've got to get the Gateway developed, and we need to get the landing systems developed. That's really what we're focused on now. If you look at a normal development project, it

follows a bell curve. We're turning to commercial industry to provide us with their thoughts on how to go from the Gateway down to the surface of the Moon. We are in essence buying a service...We're looking for our industry partners to make their own investments into the lander. We expect them to invest in it with a purpose of having customers that are not NASA. They could have customers that would be international. They could have customers that would be commercial industry customers. So we're looking for our partners to invest their own money. We're doing this in a way that's never been done before." (Grush, 2019).

From this perspective, NASA is taking a different approach related to product development and integration, and instead of an "Artemis-wide requirements effort" NASA is focusing on developing requirements for the specific product elements such as Gateway and the Human Landing System, treating them like individual programs which are developed by commercial partners. This is a comparable approach that NASA has taken with the Commercial Crew Program, which recently saw a mission by Space X to transport two crew members to the International Space Station and back. In this paradigm, Space X developed the spacecraft and owns the assets and provides a service to NASA for transportation.

Artemis is also leveraging competition, similar to the Commercial Crew Program, where it has provided multiple contract awards to different companies for concurrent product development, and will award completed product contracts to some or all of the finalists in later program life cycle stages based upon their project performance.

This space project example provides a deeper look at a newly developed element of Artemis, the Human Landing System (HLS). The HLS Integrated Lander will deliver a crew from lunar orbit to the lunar surface, provide capabilities for surface extra-vehicular activities, and then return the crew to lunar orbit to enable their return to Earth. In order to meet these goals and directives, NASA "seeks to develop the HLS utilizing public-private engagements that will reduce the cost of developing the HLS, reduce the time required for the development cycle, and enhance U.S. competitiveness in the global space industry." (NASA, 2019).

The Human Landing System is the final mode of transportation that will take astronauts to the lunar surface in the Artemis lunar exploration program. On early missions, the astronauts will live inside the pressurized crew cabin portion of the lander for up to a week. In July 2019, NASA's Marshall Spaceflight Center was designated to lead NASA's Human Landing System (HLS) Program, performing oversight of a cross-center team responsible for the procurement of the rapid development and crewed demonstration of systems.

In September 2019, NASA asked the American space industry to propose innovative designs for a human lunar landing system through the NextSTEP-2 Broad Agency Announcement Appendix H procurement mechanism. NASA streamlined its partnering approach to empower industry to meet NASA's goal of achieving sustainability at the Moon while also expediting lander development to meet the 2024 timeline (NASA, 2020). According to the announcement, "NASA subject matter experts will work closely with these commercial partners to build their human landing systems, leveraging decades of human spaceflight experience and the speed of the commercial sector to achieve a Moon landing in 2024. The HLS program also will perform advanced development and risk reduction activities, working in parallel on activities to better inform the program and the contractor on the 2024 mission and the maturation of systems necessary for the future sustaining architecture."

On April 30, 2020, NASA announced selection of three companies to begin development on the Artemis Human Landing System: Blue Origin, Dynetics, and SpaceX. An overview of the various HLS concepts is shown in Figure 93.



Figure 93. The Various HLS Concepts. (Leman, 2020)

Figure 94 provides the graphical depiction of the HLS product structure. This program will be part of a larger system of systems, and the interfaces to the external elements will be overseen by the Artemis NASA organization. For the demonstration mission, which is crewless, the commercial provider will be responsible for launch vehicle procurement and a supporting mission control infrastructure.

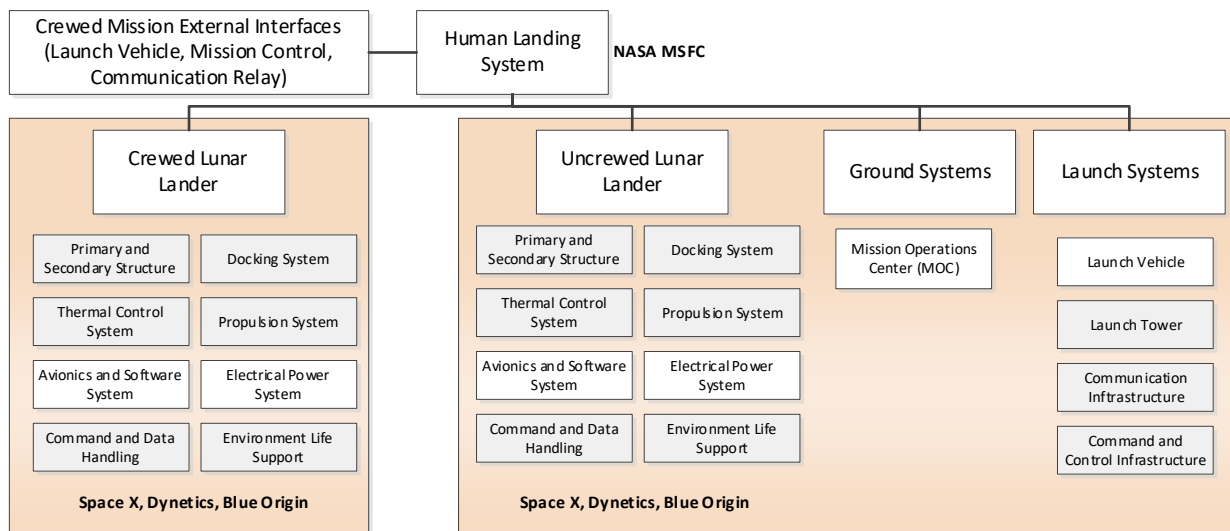


Figure 94. The HLS Product Structure.

4.6.2 Artemis HLS Requirements Approach

The overall requirement document is HLS-RQMT-001, which applies requirements as follows:

- The main functional performance requirements are contained within the body of the document.
- Appendix A contains the interfaces that HLS integrated lander must successfully interface with, calling out five interface requirements documents.
- Appendices B through D contain NASA standards that set forth a potential approach for how NASA would design, build, certify, and operate an HLS, calling out 37 design and construction standards.

For the appendices B-D, providers are expected to "demonstrate that the proposal meets or exceeds each NASA standard, employ an alternative approach to a specific standard which the provider asserts is equivalent in outcome, with a thorough explanation of such equivalency and a rationale in support of this approach in lieu of NASA's specification (i.e., a 'meets the intent of' approach); and/or (on a case-by-case basis) provide an approach that does not meet a particular NASA standard or its intent, but results in a demonstrably better approach that is more likely to enable the provider's ability to achieve one or more of NASA's overarching objectives and functional performance requirements." (NASA, 2019).

The HLS requirements were developed from the overall Artemis needs and requirements, and includes the program design and construction standards as well as interface requirement documents to establish interface control between the various Artemis program elements. The HLS document is then applied to the various contractor organizations to decompose further to their subsystems and components. This follows an approach Constellation used in their high-level requirement tree (shown previously in Figure 87), and for HLS is reflected as shown in Figure 95.

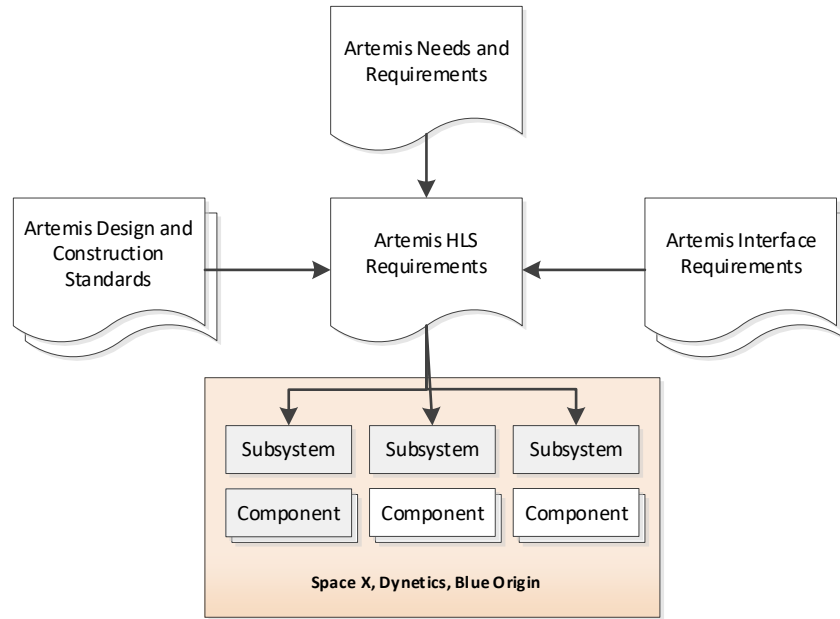


Figure 95. The HLS High Level Requirement Tree.

4.6.3 Artemis HLS Outcome

The development contract was awarded to Space X, Blue Origin and Dynetics in April 2020. At this point the projects are implementing their requirements approaches and it is too soon for a full assessment of project outcome. However, some initial estimates associated with systems engineering due to requirements can be calculated and compared with the prior projects.

The COSYSMO calculation is based on requirements obtained from the HLS Specification, which has an assortment of IRDs and standards invoked with add to the requirement count. The total HLS requirement count is shown in Table 21.

Table 21. Total HLS Requirement Count

Source from HLS-REQT-001	# of Docs	Total # of Reqts	Nominal	Difficult
Functional/Performance Requirements	1	35	30	5
Human Integ. Requirements	1	255	225	30
Safety Requirement Standards	1	11	11	0
Design and Const Standards	38	3800	3420	380
IRDs	5	450	405	45
Totals:		4551	4091	460

The resultant estimated systems engineering labor for HLS is shown in Table 22.

Table 22. Artemis HLS Calculated Labor Costs from COSYSMO.

Project	# of System Requirements	Estimated Systems Engineering Labor	Project Notes
Artemis HLS	~4551 (460 of these rated difficult)	2802 person months	Project Still in Development

4.6.4 Assessment of Artemis HLS Requirements Approach

A key challenge with the Constellation project was the magnitude of the scope of effort of a major system of systems. With HLS, this scope was reduced to oversee a specific element within Artemis, allowing the project to be developed in a focused effort for its ability to address the overall mission needs. Another difference with HLS is the change of approach to utilize commercial companies as partners, where the companies are providing a service and have more control over the products that they will ultimately own. This results in a more focused set of requirements for HLS that address the service aspects of the mission, as well as interfaces with the other Artemis elements.

One area that is still concerning, however, is the approach towards design and construction standards. These are still mandated in the HLS specification, and require verification, or effort associated with equivalent approaches, by the provider. The same challenges raised in Section 4.5.4 still apply to the HLS development; that is, potentially incurring hidden costs associated with the assessment of dozens of design standards.

4.7 Assessment of Requirements Management Approach of the Space Example Projects

Looking at these space projects, some information was provided related to overall budget, schedule, and success. Each project was assessed for estimated systems engineering labor costs associated with their requirements management approach, as well as how each applied the requirements management process model introduced in Figure 36 in the development and management of their requirements. A summary of the results presented is shown in Table 23. For the evaluation of the need to optimize requirements management, a scale of 1 to 3 is used, where 1 indicates a project has met all of its

objectives, 3 indicates a project that met none of their objectives, and 2 is used showing some objectives were able to be achieved and would benefit from further optimization.

Table 23. Summary of Space Project Calculated Labor Costs from COSYSMO.

Project	# of System Requirements	Estimated SE Labor (Person Months)	Project Notes	Need to Optimize Requirements Management (1-3)
MAVEN	660 (0 of these are rated difficult)	298	Successfully executed project objectives	1
MSL	511 (309 of these are rated difficult)	747	Project was moderately successful, further optimization could have ensured it met cost and schedule objectives.	2
GOES-R	~1300 (50 of these are rated difficult)	643	Project was moderately successful, further optimization could have ensured it met cost and schedule objectives.	2
Constellation	~8600 (1220 of these are rated difficult)	6109	Project Cancelled in Design Phase	3
Artemis HLS	~4551 (460 of these are rated difficult)	2802	Project Still in Development	Cannot rate

The project scale and complexity is indicated by the number and quality of requirements (resulting in an estimated systems engineering labor), and this can be plotted against project need for optimization, shown in Figure 96. An indication of projects that would benefit from optimized processes in requirements management is shown on the figure, proposing the type of project that should consider moving beyond the current state of the practice.

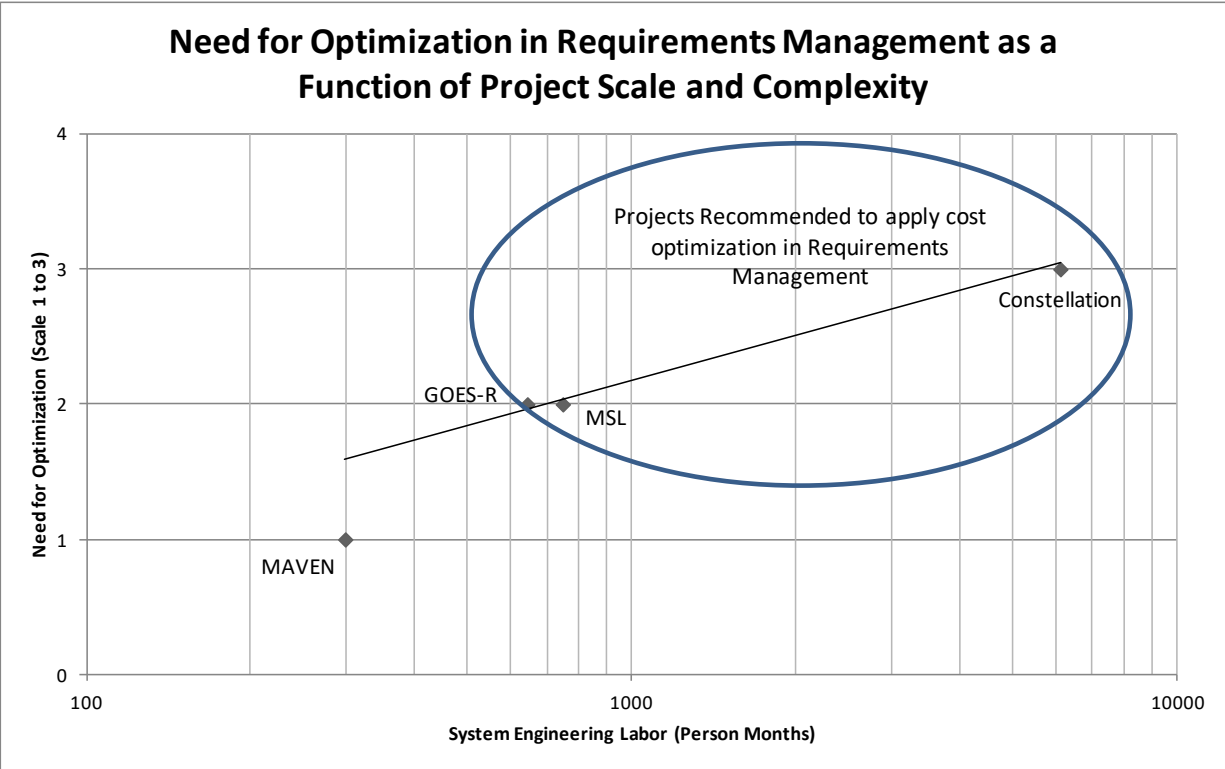


Figure 96. Project Complexity and Need for Optimized Processes.

The next chapter introduces the research done to establish an optimized approach to requirements management for the space systems, evaluating how the different space projects could have optimized their approaches to yield more desired outcomes of project success.

5.1 Factors Used to Evaluate Optimized Requirements Management Approaches

When considering the application of process activities, a few parameters impact whether the usage of the new approaches add value for the project. These considerations include:

- Cost of application (and maintenance) of the new process compared to the associated savings
- Cost associated with labor cost and direct cost of requirements management processes
- Cost associated with project schedules delays associated with requirements management processes
- Cost associated with number of requirements

It is not within the scope of this dissertation to present the full array of input values related to these costs, however a representative set is provided to allow for process comparisons; these are discussed further below.

5.1.1 Cost to Apply Process Updates

When providing updates to a set of organization or project processes, key contributing costs are the cost of any software applications (initial and maintenance), costs to document the new processes, costs to train the personnel in new processes / tools, and cost associated with initial inefficiencies as the team becomes familiar with the new methods.

Examples of these inputs are below, however these will vary at different organizations as they can be a function of number of projects, existence of a support organization to focus on the implementation of new approaches, and experience level of the personnel. The estimates below are from the author's experience in coordinating these types of activities on past efforts.

- Cost of software applications can range from \$1,000 to \$10,000 a user per year, with options ranging from floating to dedicated user licenses; training and support may be included with the tool provider.
- Cost to document new processes can be reflected in labor hours of personnel involved and can range from 20 hours to 80 hours based on complexity.

- Costs to train personnel include direct costs to instructors, cost to develop material, and labor costs of those being trained. Typical training courses could be 40 hours of instruction, and cost \$5,000 to \$10,000 to produce.
- Startup transients are likely to follow established patterns of inefficiencies, with an initial delay in execution for the initial effort, and then leading to a predicted outcome within a set learning curve. A method for calculation of a learning curve is provided by "Wright's Model With an 80% Learning Curve", showing the time to implement decreased by 50% from the initial execution to the eighth (MAAW, 2020).

As an example: For a project team implementing a new software application and requirement review approach, the investment cost for ten users could be \$50,000 (licenses) plus 400 hours of training time (estimate \$100/hr to result in \$40,000) with a \$10,000 instructor cost plus a reduction in efficiency in duration for the first several weeks of use estimated at a cost of 200 hours (\$20,000) resulting in a total application cost of \$110,000. Any new process would need to yield a savings over \$120,000 to be worth the effort of applying to the project for usage.

5.1.2 Duration and Labor Cost of Requirements Management Processes

When implementing processes on a project one method of calculating cost is by capturing labor hours associated with project activities. For labor costs, the personnel working for the project are paid a wage, and based on project accounting this is normalized to amount of cost per hour of labor. For the requirements management process, the effort is primarily conducted by engineers, where the cost/hour can range from \$100 to \$300, including cost to the company for management of the employees.

When comparing two processes there is often advantages to looking at how they compare in durations, especially when normalized for other factors such as startup costs. For many of the process steps shown in this dissertation a range of hours is provided showing a low and high end of hours that this step could take. A range of times allows for a sensitivity analysis compared to lower and higher durations and their impact on the overall outcome. The values used in the process analysis were obtained from this author's experiences, and can vary greatly based on experience level of the personnel involved; for this reason the hours are provided for comparison purposes only, to show how processes compare to one another, and not a basis for a project to calculate a total time effort related to the project's process implementation.

5.1.3 Direct Cost of Requirements Management Processes

The direct cost associated with requirements management is often associated with purchase of product, or payment for a supplier's effort during the process activity (such as their time to evaluate new requirements). For requirements management this direct cost can often show up as a value of cost per change for a supplier, which can range from \$10,000 to \$1M per change based on the supplier to address contract impacts associated with requirements changes.

Like the labor costs this estimate is based on the author's past experience, and will vary based on the complexity and type of the supplier contract. For purposes of evaluation, this is varied between \$10,000 to \$100,000 per change, where the change could indicate contract change for a set of requirement updates. This parameter could be adjusted as an input for other projects with more or less complex supplier efforts.

5.1.4 Cost Associated with Schedule Delay

Requirements management inefficiencies could result in delay in project completion. Project management processes provides techniques to evaluate a project schedule and address impacts of schedule delays (also known as schedule slips). Depending on the project a delay can be quite costly to the project, especially based on when the slip occurs (reference Figure 97).

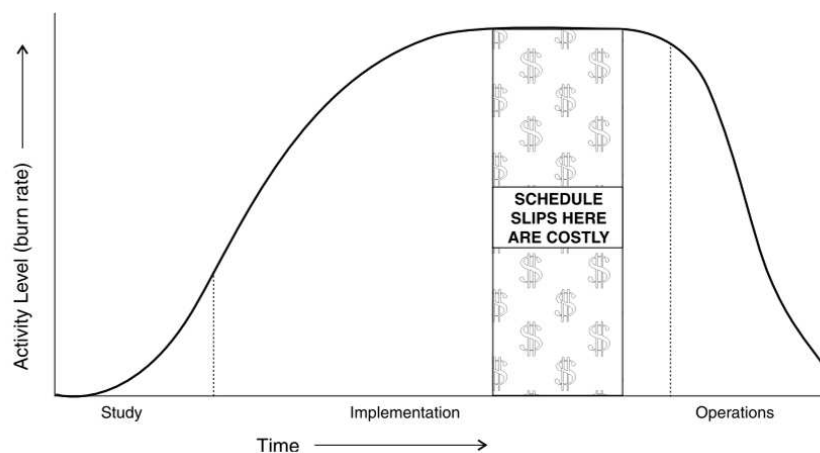


Figure 97. The High Costs of Schedule Delays. (Forsberg, Mooz, & Cotterman, 2005)

Techniques to bring in schedule and avoid costs due to schedule slips are sometimes referred to as schedule "crashing", and can incur their own set of costs to incorporate. It would make sense to spend money to ensure a schedule is met if there is a very high penalty associated with a delay, where for other projects it may be too costly to mitigate schedule slips compared to the impact of delay.

Relating schedule delay costs to requirements management, one option is to weigh the cost of refining the requirements to a mature state compared to impacts of schedule delay costs associated with waiting for mature requirements. In some cases the delay costs are too high, necessitating some concurrent work of product development in conjunction with requirements development, incurring some risk that the product will not meet the resultant, matured requirements (causing rework of either product or requirements). For purposes of evaluation, delay costs are varied in the analysis between \$50,000 to \$150,000 per month of delay to evaluate impacts of various options. This parameter could be adjusted as an input for other projects depending on their project parameters.

5.1.5 Cost Associated with Number of Requirements

As discussed in Section 4.1.3, systems engineering costs on a project is a function of number of requirements by utilizing COSYSMO, as well as the quality of the requirements. When assessing projects with comparable labor durations and direct costs, another factor for consideration is whether the process yields net fewer requirements or more nominal requirements. Establishing a comparison of systems engineering labor hours based on the resultant requirement quantity from a process is another method for assessing cost optimization

5.2 Proposed Process Updates for Requirements Management

Common challenges exist in many of the prior space system examples, including highly complex system of systems, multiple tiers of products, different suppliers performing development activities, and changes of requirements during the development efforts. An investigation into different options for requirements management was done to address challenges posed at the end of Chapter 3 regarding the optimal approach to flow requirement specifications from system to component level, and the best

approach for addressing changes to requirements among the product development organizations. As a refresher, the requirements management trends observed in Chapter 3 included:

- A movement away from a document centric approach towards a data centric approach of managing the project's requirements, minimizing the usage of documents or compartmentalization of the requirements and combining requirements to an overall project repository enabling traceability and connection to other project data;
- Use of requirements management tools for development, collaboration, change control, and trace to other project data; tools are more effective if project templates are established at the start and if they are easy to use;
- Use of tools to enable requirements to connect with other project data to support verification planning, leveraging traceability and connection between requirements and verification events and their artifacts; and
- Carefully planning on when to start change control on requirements, too soon or too late can have impact to project execution, and controlling too many requirement attributes can drive schedule.

These trends provide insight into opportunities for further process development in requirements management. Based on the literature review, observations from past and current space projects, and interviews with those working requirements management (Table 9), the process updates in Table 24 are proposed to achieve cost optimization when managing requirements for a space system development. All of these ideas will be expanded upon further in this section.

Table 24. Proposed Process Activity Updates for Requirements Management.

Identifier	Proposed Process Update
1	Implement a data focused requirements management approach
2	Utilize a management tool that supports electronic collaboration during requirement development and change activities throughout the project life cycle
3	Minimize and consolidate the requirements for the system of interest
4	Coordinate the timing between developing requirements and levying them officially

An updated requirements management process model is introduced later this chapter that includes an application of these different processes, allowing for comparison to the process model shown in Figure 36. This model will be implemented as an executable model, addressed in Chapters 6 and 7, to determine

if the proposed approaches yield optimized results when supplied project inputs. The proposed processes are described below in more detail, including rationale and associated design pattern for inclusion in an optimized process model.

5.3 Process 1: Data Focused Requirements Management Approach

The data focused requirements management process is proposed for further study based upon inputs from the subject matter experts and work done by the INCOSE requirements working group. The section below describes the benefits of this approach and generates a process model design pattern used for evaluation in a quantitative assessment in Chapter 6.

5.3.1 Document Centric Requirements Management Approach

Document-based systems engineering is described as an approach that relies upon paper or digital textual documentation to record system specifications and other development project related information (Carroll & Malins, 2016). For many years requirements were developed for various levels of a project and published in formal requirements documents for the associated products. Specific types of requirements, such as interface requirements, design requirements from standards, test requirements, etc., were created by subject matter experts and published in separate documents which were invoked by the main product specification. In this paradigm, requirements are viewed in the frame of their document, instead of how they fit into the entire system's set of requirements.

Figure 98 is an extraction and expansion from the requirements management (RM) model in Figure 36, showing the key areas where a document focused requirements management approach occurs during the development, assessment, review, update, and approval of the project requirements. The aspects of document vs. data management can occur in the later life cycle phases as well, however this section will focus on the initial efforts associated with requirements development and baseline activities.

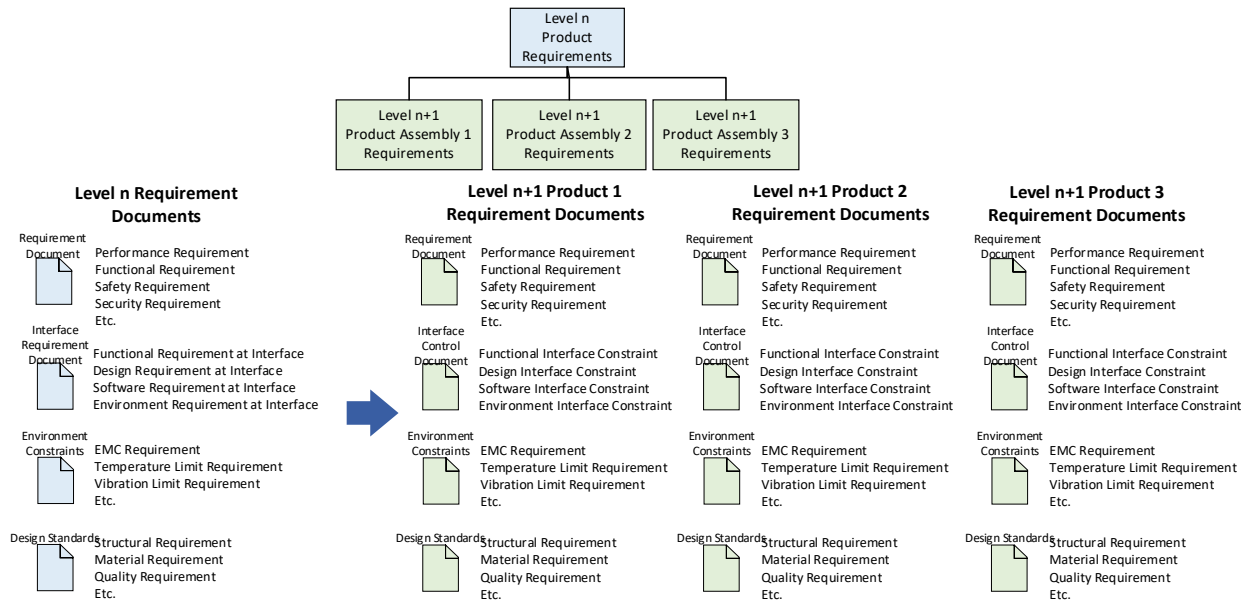


Figure 100. Document Centric Requirements Management Artifacts.

In both figures, the level n requirement development activities are shown in blue, while the development activities for level n+1 requirement development are shown in green. These may be addressed by the same company, but often represents a handoff between one developing organization to another (such as a developer for a subsystem receiving requirements from the systems engineering organization, and then decomposing them into requirements for their component suppliers). This concept is another view of Figure 15, which shows the handoff of requirements among multiple development organizations, where the requirements are contained in a series of documents that are handed off between organizations.

5.3.2 Current Trend Towards Model Based Systems Engineering

In Section 3.2.5 various requirements engineering practitioners observed that requirements development and management is becoming more "data focused". Overall, it is observed that the entire systems engineering field is moving away from document centric to a more data centric effort. This is described as Model Based Systems Engineering (MBSE), which is defined as the formalized application of modeling to support systems requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle

phases (INCOSE, 2015). An outcome of the MBSE process is the generation of a system model that describes the behavior and structure of a system with trace to the system's needs and requirements. As shown in Figure 9, the requirements development process is an iterative effort with the design process. MBSE allows the system model to capture the needs, address the behaviors and functions required to achieve the needs, generate the requirements associated with the behaviors and associate them with elements of the system that implement the functions, and then develop this further to more detailed behaviors and elements. As opposed to a document centric systems engineering process, the model based approach allows the requirements, structure, and behavior to exist in a comprehensive model that can be used to investigate architectures and trade options.

An investigation into this approach was conducted by Carroll and Malins at Sandia National Laboratories in 2016, where they compared projects that used a model based approach with a more document based approach, reaching a conclusion that there is a "significant advantage to project performance by applying an MBSE approach" (Carroll & Malins, 2016). They found that an MBSE approach made the engineering processes on a complex system development effort more efficient by improving requirements completeness, consistency, and communication. Examples of efficiencies included reduction in systems engineering labor, decrease in number of deficient requirements, and increases in project probability of success.

5.3.3 Current Trend Towards Information Based Requirements Management

As shown in Section 3.2.1, the concept of information-based requirements development and management enables of the requirements with the rest of the project information. "The advent of Requirements Management Tools (RMTs) and other SE tools has allowed systems engineers to **move away from printed, hardcopy document-centric** requirement development and management (RDM) processes, with no underlying dataset, **to a data-centric process** where the data and information associated with the requirements set is represented in an shareable data and information model. To ensure needs and requirements are developed and managed concurrently with the product design effort, it is recommended that projects use a systems engineering (SE) toolset that allows the organization to develop

and manage needs and requirements in relation to all the SE artifacts developed across all system development life cycle activities." (Wheatcraft, Ryan, Dick, & Llorens, 2019). As part of an information-based approach, the requirements are generated using an underlying data and information model, ensuring the resultant requirement set is consistent, correct and complete.

5.3.4 Usage of a Data Centric Requirements Management Approach

Applying a data-focused approach to the figure shown earlier on document centric requirements management, the processes associated with data centric requirements management are shown in Figure 101 (with the artifacts of the process highlighted in Figure 102). The concept of receiving documents from a higher level is still shown (reflective of how most prime contractors are receiving requirements currently); however, the developer's response is to input all of these documents into a database which allows the requirements to be associated with all of the other project data, such as models, verification plans, and other requirements.

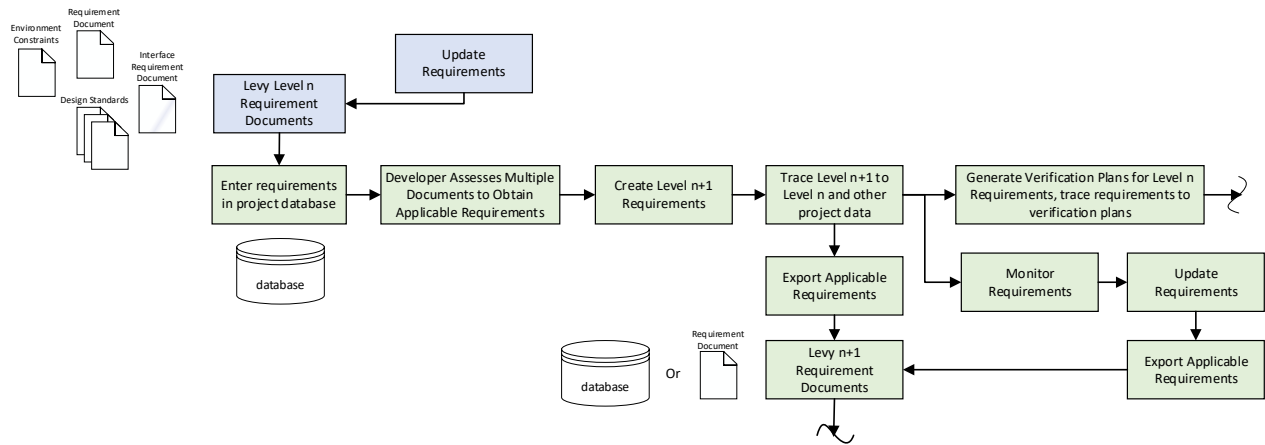


Figure 101. Data Centric Requirements Management Process Flow.

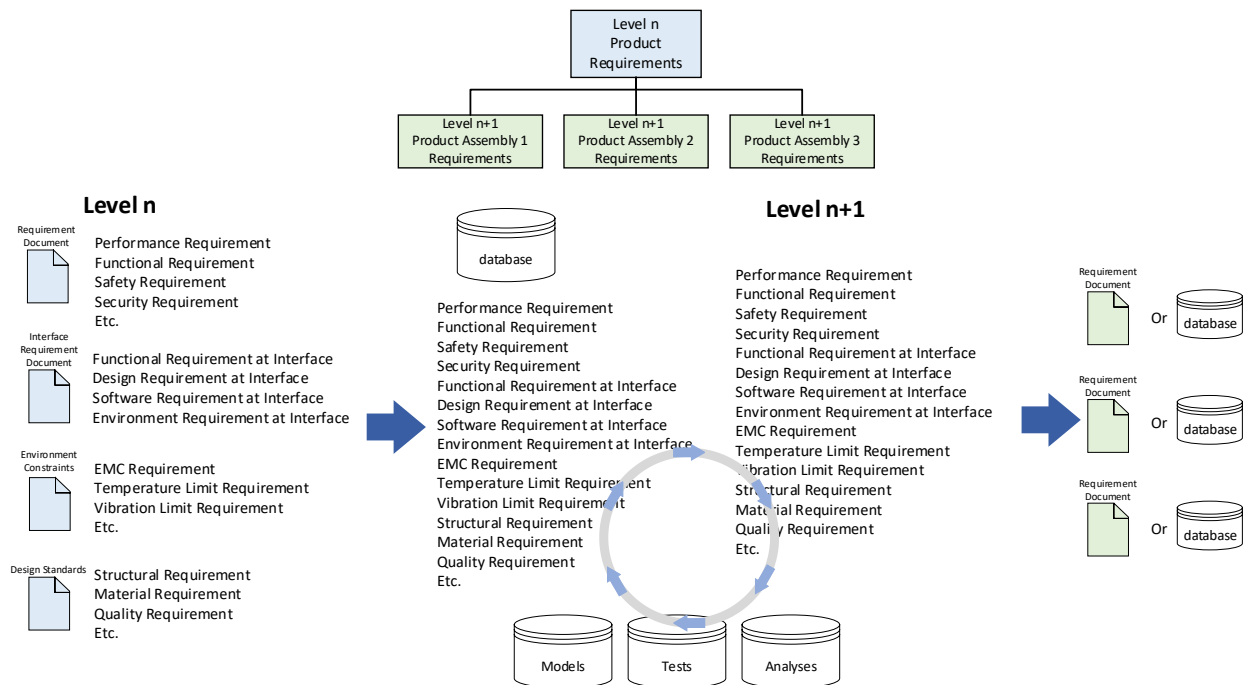


Figure 102. Data Centric Requirements Management Artifacts.

Compared to Figure 100, this data centric approach allows for lower level product specifications to be provided as exported documents (one per product) or as a set of requirements data that can be provided to the supplier electronically. This approach also allows for a singular location for all requirements being managed, with a connection to other project data, providing a comprehensive view that allows for an assessment of completeness, consistency, and correctness.

5.3.5 The Benefit of Requirement Reuse

One concept developed over the last several years is called "requirements reuse". Depending on the context, this can be a reuse of existing requirements from one project to another, or a reuse of requirements across a single project to similar lower level products. Requirements reuse application can take many forms, from a "copy and paste" concept to a data share concept, such as leveraging requirements attributes and trace (Visure Solutions, 2020).

The idea of reusing requirements from a document centric approach has been around for many years, where the starting point of generating a set of product requirements would involve copying a prior product specification and updating it. In a data centric approach, this starting point becomes a set of

common requirements that are then associated with specific products and updated as applicable. This is shown in Figure 103, demonstrating how a customer's requirement "REQT-X" is reused for various products in the product structure; this reuse means the requirement is cloned with trace to the original requirement, allowing a quick method to see everywhere this requirement is applied.

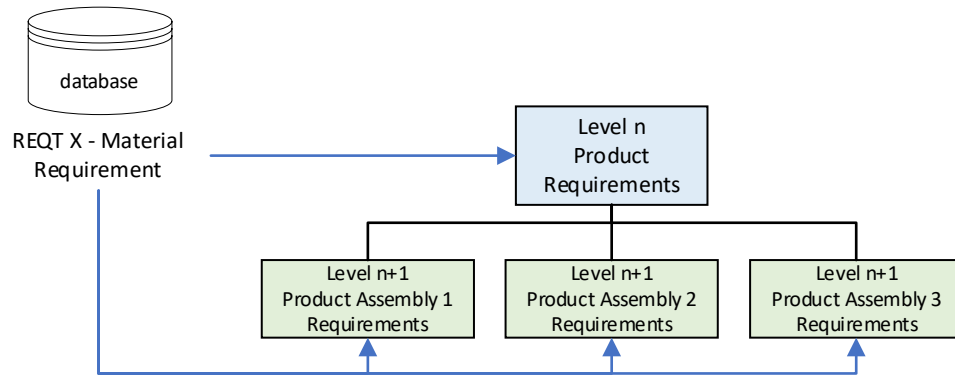


Figure 103. Example of Requirement Reuse.

As a starting point, projects may start with requirements from prior projects or from a common library (typical for a product line approach). These may be brought into a requirements management tool, and associated to all products and projects that use those requirements, allowing a comprehensive view of where these requirements are all used. Modern requirements management tools like Jama Connect and IBM DOORS Next allow this capability by assigning a global identifier to the requirement, which can be tracked and analyzed for all instances of use.

An example of this is shown in Figure 104, which shows a sample requirement on materials that is reused to a component level, displaying the Global ID of the requirement as well as a local ID, and showing the related (reused) requirements elsewhere in the project. In this example the ability to see a synchronization of the reused requirements is provided, where the tool is able to maintain trace and status of values of the requirements. Modern tools also provide the ability to tailor reused requirements, still maintaining a trace but allowing for the values to be adjusted to show where some components differ from the main set of requirements.

By treating each requirement as a data element the project has flexibility on how often it can be used within the project as well, such as common design requirements that apply to many components.

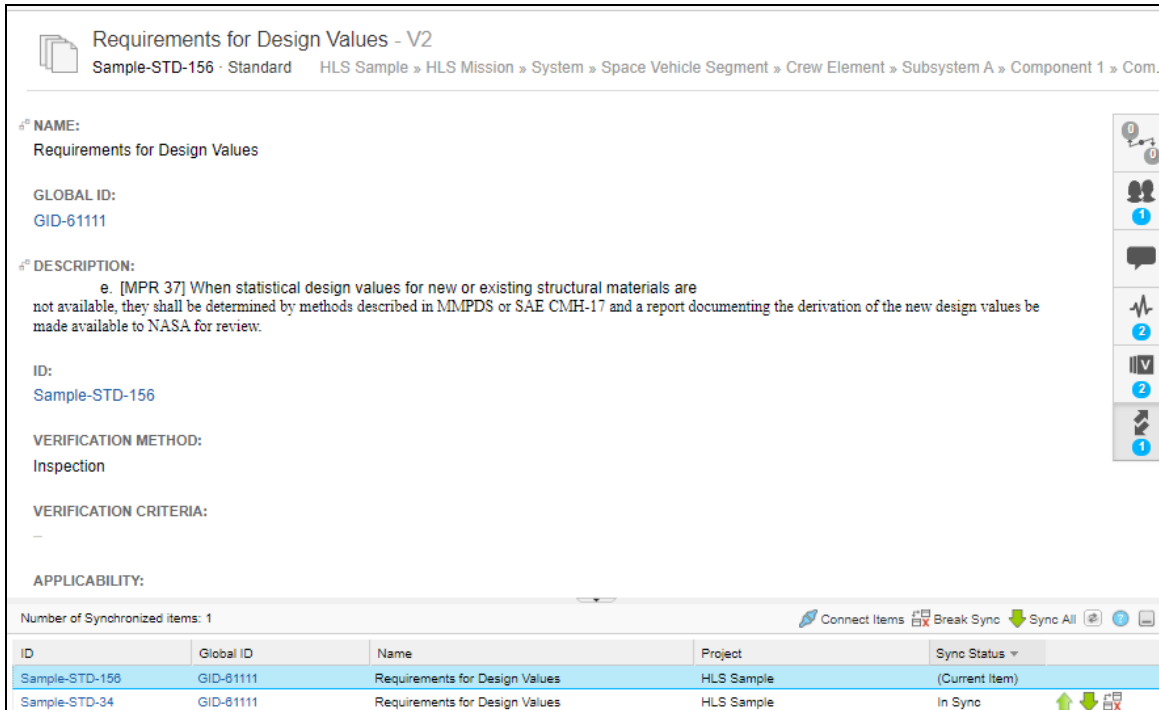


Figure 104. Example of a Reused Requirement in Jama Connect. (Jama Software, 2020)

Besides simplifying approach to application of requirements such as design standards, environments, and quality requirements, reusing requirements allows for requirement standardization and aids in creation of a repository of previously implemented and tested requirements (enabling a faster approach to obtain starting requirements for a project). However, if not done thoughtfully, there is a risk of blindly copying requirements that may be outdated or not applicable to the project, so maintaining the context data with the requirements is critical to ensure proper usage (QRA, 2020). QRA offers seven tips on performing requirements reuse at their website <https://qracorp.com/how-to-reuse-requirements>. Among them is the idea of creating templates and libraries as a starting point for common requirements. Toval, et al, explores the option of creating tools for enabling reuse of requirements to provide a practical approach for selecting and specifying the requirements of a software system based on requirements reuse and software engineering standards that utilizes a spiral process model, templates, repository organized by catalogs, and a requirements software tool (Toval, Nicolas, Moros Valle, & Lasheras, 2008); all of this is enabled by usage of a data centric approach to requirements management.

5.3.6 Cost Optimization using a Data Focused Requirements Management Approach

The industry trend towards data centric systems engineering has already yielded evidence that a comprehensive approach of the project data yields cost improvements for a project (Carroll & Malins, 2016). Applying this concept to requirements takes this concept to a narrower focus.

The trade-off between managing several documents with compartmentalized requirements compared to a comprehensive set of data involves assessing the costs associated with the requirements management tools (training, license costs, computing infrastructure, tool management). For the space industry, requirements management efforts are often performed by highly educated engineers, with an associated high labor rate. Considerations of cost optimization assesses the amount of time engineers will spend on requirements management compared to the costs of the tools and associated support. For simple projects, a data focused approach may not yield obvious benefits. For most space projects, the tool cost is trivial compared to the cost of the engineering labor spent to manage multiple documents and manually assess requirements trace to project data.

Based on findings from the Sandia study, it is asserted that these same findings apply to the benefits of a data focused approach to requirements management, where the initial work of establishing a data focused approach yields cost optimization for a project compared to a document centric approach.

However, to demonstrate the process differences between the document and data centric approaches, the activities for each have been incorporated into an executable model and the labor time associated with each is compared; the results of this effort is provided in Chapter 6.

5.3.7 Data Focused Requirements Management Design Pattern

For a document centric approach to requirements management, a design pattern was extracted from Figure 100 to provide the basic steps (Figure 105). Each of these steps uses associated labor hours based on the number of documents involved (sample document types are shown in Figure 100).

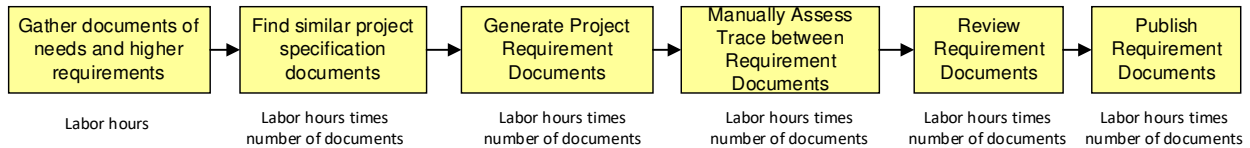


Figure 105. Document Focused Requirements Management Design Pattern (1a).

For an optimized approach using a data focused requirements management processes, the process flow in Figure 101 is represented as a design pattern to enable it to be added to the requirements management process from Figure 36 (the current, synthesized requirements management model). This design pattern is shown in Figure 106. Likewise, each step is associated with labor hours, however the usage of a data repository for the requirements removes the need to manage multiple documents and simplifying the labor costs.

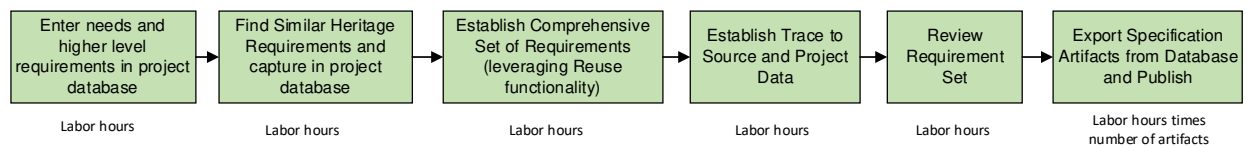


Figure 106. Data Focused Requirements Management Design Pattern (1b).

A note on convention: Throughout this dissertation the current state processes are noted as the process number followed by the letter a (1a, 2a, etc.), and represented by the color yellow; the proposed optimized processes are noted as the process number followed by the letter b (1b, 2b, etc.) and are represented using the color green.

5.4 Process 2: Use of a Collaborative Requirements Management Tool

The use of a collaborative requirements management tool is proposed for further study based upon inputs from the subject matter experts and results of requirements management tool investigations from Chapter 3. The section below describes the benefits of this approach and generates a process model design pattern used for evaluation in a quantitative assessment shown in Chapter 6.

5.4.1 Rationale for Requirement Collaboration

Requirements development and management occurs as a comprehensive effort by multiple members of a project team. A requirements engineer may be designated to lead the effort, but other project engineers and staff are stakeholders that either provide input to, or approve, the project requirements. Figure 107 is an extraction from the requirements management model in Figure 36, showing the key areas where collaboration will occur in the development, assessment, review, update, and approval of the project requirements.

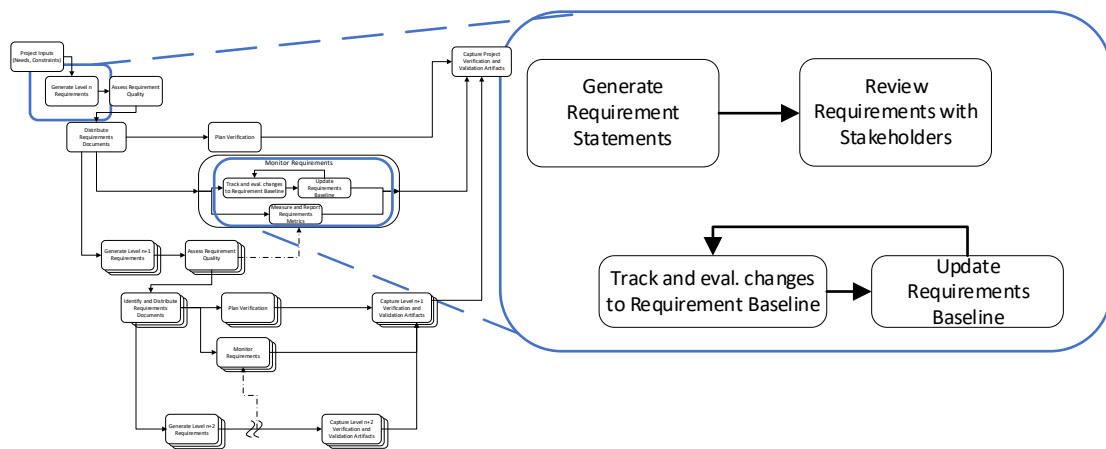


Figure 107. RM Model Activities Associated with "Use of a Collaboration Tool".

Information obtained from discussions with the requirements management practitioners highlighted that several historical requirements management tools are difficult to use, resulting in the need to export the information to project team members and hold reviews outside of the tool. Figure 108 shows this series of steps, where the reviewers will look at exports from the tool, perform markups, and provide them to the engineer for entry into the tool. Items in the outlined box are done outside of the requirements management tool, with the results entered in by the responsible engineer for the requirements. The steps for export and input is considered a non-value usage of the engineering labor, and the multiple review export files create a potential for data integrity errors with the manual process.

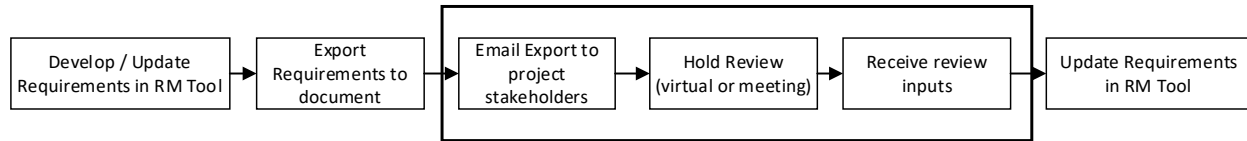


Figure 108. Collaboration Activities Outside of Requirements Management Tool.

Based on the concept of information-based requirements management shown in the prior section, the idea that the requirements are tied to other project data leverages a common source of "truth", where the data is part of a larger network of project information. Having the ability to see this data provides value to the various team members, as they can see the requirements in context with other project information (such as design models, behavior models, verification plans, etc.).

To enable more complete situational awareness of the project team, and reduce non-value labor by the requirements engineering team, the concept of using a tool that enables collaboration by the different project team members is gaining more usage in various organizations.

5.4.2 Options for Collaboration During Requirements Management

Document centric tools, and older requirements management tools, utilize a pass back and forth approach to collaboration, as noted above. Newer tools, such as those that rated highly on the Seilevel assessment (Section 3.2.4), have built the ability to perform communication and review processes within the tool. The concept of "easy to use" enables non-requirements team members and infrequent users to learn quickly how to access the tool, navigate to areas of interest, and enter in comments, questions, and actions. While some older tools do have a built in comment capability (such as IBM Rational DOORS), these have been reported as difficult to use, which causes non-requirements team members to not use the these features.

Several of the newer tools integrate with other collaboration tools, which provide the front-end interface to obtain requirements data and provide input (not available with IBM Rational DOORS). Examples of two tools are shown in Figure 109 and Figure 110, showing the various user interfaces that enable online collaboration (comments and review input) between team members.

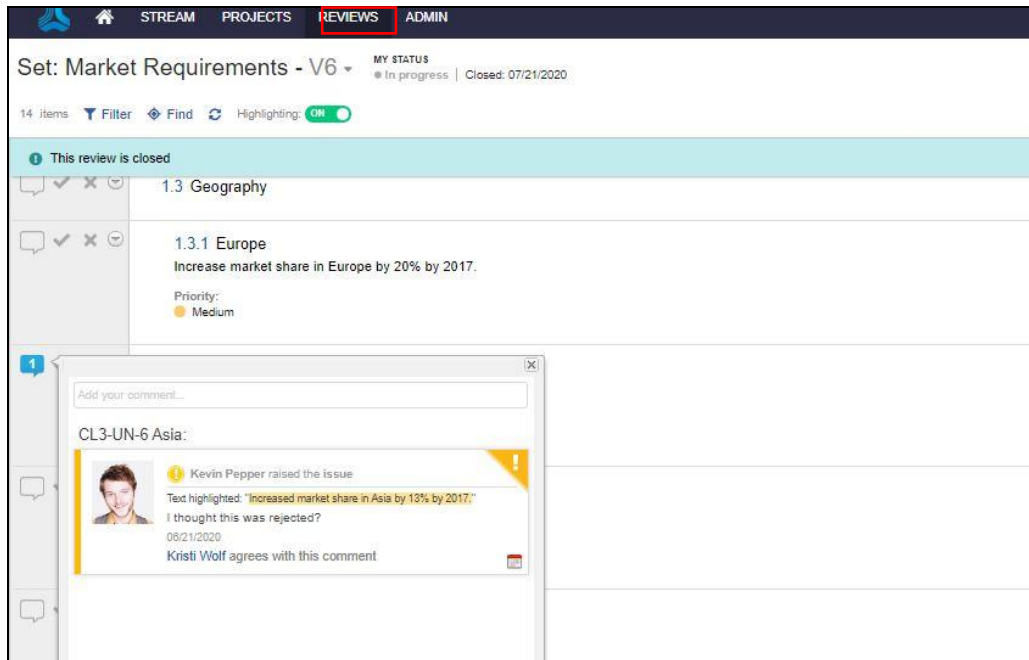


Figure 109. Collaboration Options in Jama Connect. (Jama Software, 2020)

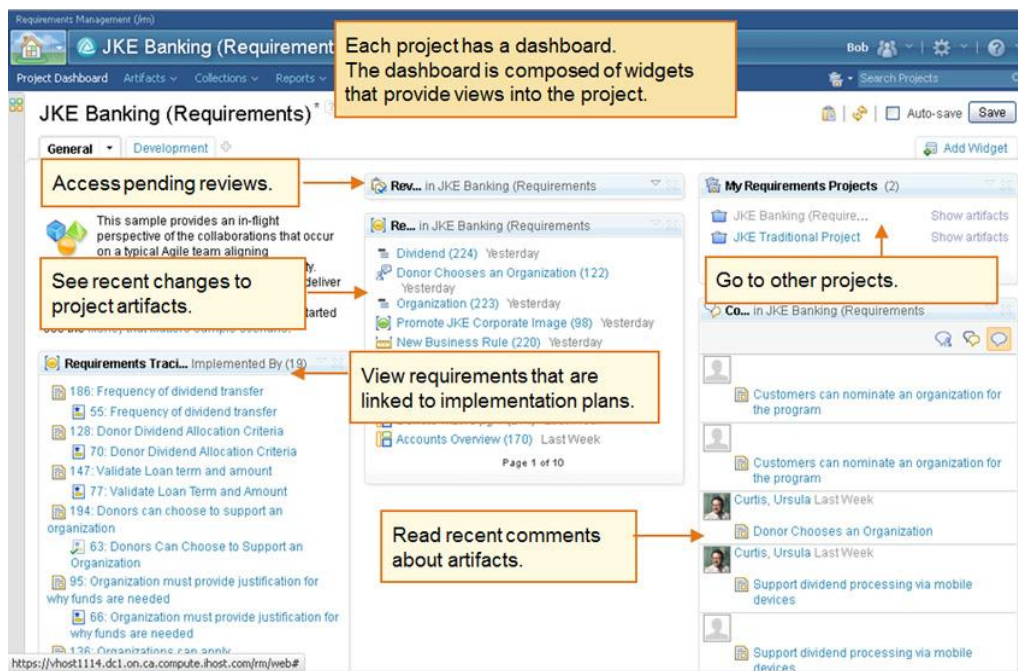


Figure 110. Collaboration Options in IBM DOORS Next. (IBM, 2014)

5.4.3 Cost Optimization with a Collaborative Requirements Management Tool

Research conducted by Forrester Consulting, on behalf of Jama Software, identified five obstacles to optimized product development (Jama Software, 2020):

- Unclear or changing requirements coupled with lack of timely feedback for solutions
- Lack of focus caused by conflicting stakeholder priorities, assumptions, and unclear objectives
- Difficulty collaborating across globally distributed teams
- Unnecessary handoffs and delayed decisions
- Increased collaboration across diverse roles, including executives, operations, marketing, and quality assurance

Many of those challenges were demonstrated with the space projects in Chapter 4, particularly the larger complex projects such as Constellation.

By streamlining collaboration, there are fewer meetings and more iterative reviews which equals faster completion times, clearer communication between stakeholders, and more time for teams to focus on improving quality. Case studies of Jama Software customers revealed that implementing a structured collaboration in the requirements management tool saved \$150,000 per project, and planning time for requirements took 20% of the time it used to in legacy approaches.

Many of the modern requirements management tools provide ability to address challenges of requirements management, such as collaboration and reviews; the goal of this process recommendation is to demonstrate why a project would invest in changing over to a newer tool compared to continued usage of a less-collaborative one. Much like the data centric study, the concept of the project team working from a common set of data yields cost savings, and improves data integrity, enabling cost optimization. To demonstrate the process differences between the collaboration approaches, the activities for each process has been modeled and the labor time associated with each has been compared; the results of this effort is provided in Chapter 6.

5.4.4 Collaborative Requirements Management Tool Usage Design Pattern

For a traditional approach of collaboration outside of the requirements management tool a design pattern was extracted from Figure 108 to provide the basic steps, reflected in Figure 112. Each of these steps uses associated labor hours to implement the activities.

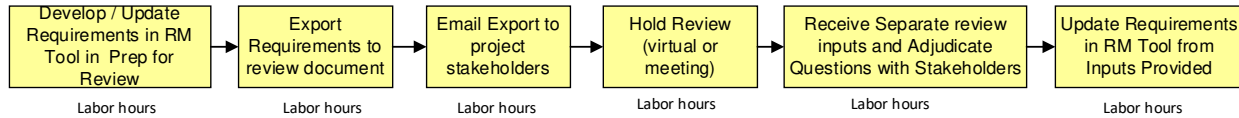


Figure 111. Non-Collaborative Requirements Management Tool Design Pattern (2a).

The process flow is updated to reflect application of collaboration **within** the requirements tool; this is represented as a design pattern to enable it to be added to the requirements management process from Figure 36, and is shown in Figure 112.

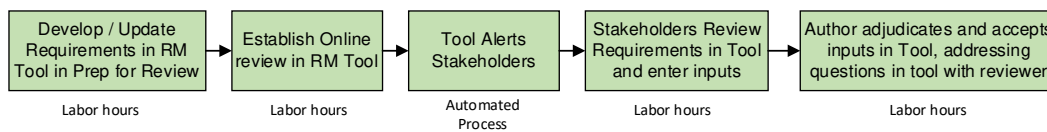


Figure 112. Collaborative Requirements Management Tool Design Pattern (2b).

5.5 Process 3: Minimize and Consolidate the Requirements

The concept of minimizing and consolidating requirements for the system is proposed for further study based upon review of the COSYSMO approach for cost estimation as a function of requirement quantity and quality, as well as observations generated from Katz's paper *When to Constrain the Design? Application of Design Standards on a New Development Program*. The section below describes the benefits of this approach and generates a process model design pattern used for evaluation in a quantitative assessment shown in Chapter 6.

A common trend during requirements development is to ensure a full and complete requirements set exists, which is inclusive of using industry standards, quality control standards, and legacy requirement documentation. Figure 113 is an extraction from the requirements management model in Figure 36, showing the key areas where requirements generation and synthesis activities occur.

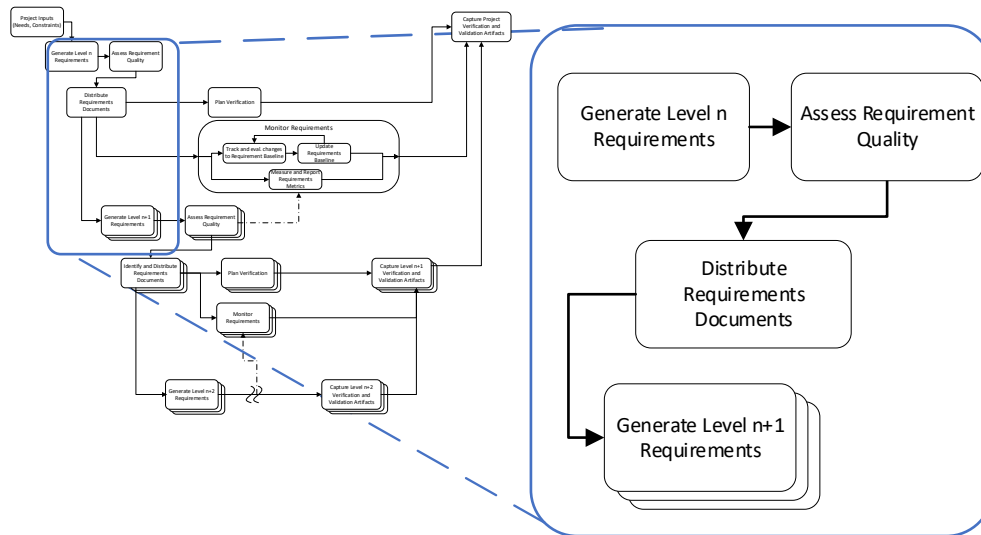


Figure 113. RM Model Activities Associated with "Minimization and Consolidation of Requirements".

5.5.1 Achieving Balance in Requirement Quantity

The *NASA Goddard Systems Engineering Training* package provides an observation that there is a balance of approach needed to ensure there are not too few or too many requirements; with too few, the design may not realize the needs properly, and with too many, the design solution is over constrained (NASA, 2015). This is the general idea behind minimizing the requirements, where *minimum* is ensuring there are just enough requirements to specify the system of interest following requirement quality guidelines, and avoid adding any beyond these. Additionally, requirements on how to make the product, how to verify it, how to implement tests, etc., are referred to as "design-output requirements" in the *INCOSE Guide for Writing Requirements*. These type of requirements need to be considered carefully before imposing on a system of interest as they will add cost and need to be accounted for in the project schedule. Unless specifically addressed in regulations or mandated for control by a customer, it is recommended that quality assurance requirements be developed to address the desired outcome, as opposed to specifying how a product developer will implement their development processes, to avoid hidden costs associated with these types of requirements (Katz, 2020).

Per the *INCOSE Guide for Writing Requirements*, one characteristic of requirement quality is the concept of **complete** (reference Table 3), which is defined as a requirement set that "sufficiently describes

the necessary capabilities, characteristics, constraints, interfaces, standards, regulations, and/or quality factors to meet the needs without requiring other sets of requirements" (INCOSE, 2019). This assessment ensures that there are enough requirements to ensure the product meets the stakeholder needs.

Another characteristic from the guide is **consistent**, which ensures that set contains unique requirements which do not conflict with or overlap with others in the set, the units and measurement systems uses are homogeneous, and the nomenclature used is consistent. This is highlighted in the COSYSMO tool in the weighting function for number of requirements, where "difficult" is used when there are a high number of overlaps (Madachy, 2015). One potential downside to overlapping requirements is the potential of conflicting or contradictory requirements, another is the cost associated with managing and verifying similar requirements.

An additional type of problem is **over-specification** of a design solution using implementation specific requirements. This can exist when specifying requirements at too low of a level of abstraction, or those that specify a design solution. Considering the case where lower level component requirements exist at the higher system level, the requirements are assuming a design, essentially constraining the system to those features. This is explored in the usage of mandatory design standards for new development efforts, where the mandate of design standards impacts innovation of the design (Katz, 2020).

In assessing several projects at JPL, it was noticed that many requirements mixed business processes, interface details, and technical requirements (NASA JPL, 2012). The challenge with this situation is in how these mix of requirements are addressed; not all require the same level of verification and effort for implementation and closure, and several of these may be responded to by the business and management personnel instead of the product design personnel.

Based on these observations, the idea of assessing the set of requirements and ensuring they contain only appropriate and singular requirements balanced with the need to control the outcome through usage of quality assurance requirements and constraints is proposed as a means of improving the outcome of requirements management on a project.

5.5.2 Requirement Document Quantity as a Function of Cost

Figure 100 shows multiple requirement documents levied throughout the multiple levels of a product structure. Based on historical observations, a comprehensive set of requirements in a single location is typically associated with lower cost compared to the same number of requirements spread out across multiple documents. This is based on this author's experience, which showed that suppliers will typically charge for their efforts of reviewing multiple document sets, going through multiple tiers of requirements which invoke requirements in other documents (such as design and construction standards), assessing if the invoked requirements are all applicable, and ensuring they have comprehensively captured all applicable requirements for which they must show contractual compliance evidence against. The activities associated with this effort for a supplier is shown in Figure 114, highlighted in the "Supplier Effort" box. This reflects a single supplier, and would be multiplied by the number of suppliers if more than one is used.

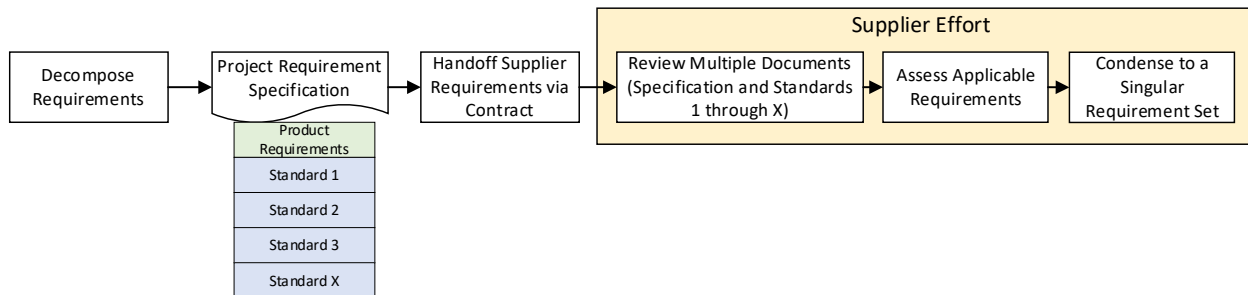


Figure 114. Supplier Effort Associated with Multiple Specification and Standards.

To avoid this cost, and minimize the supplier requirement set, the requirements can be condensed to a comprehensive, applicable requirement set prior to levying on a supplier. When all requirements are located in a single specification that is applicable to the supplier, the supplier responds directly to what is levied. Figure 115 depicts the effort to minimize and condense the requirements applicable for a supplier.

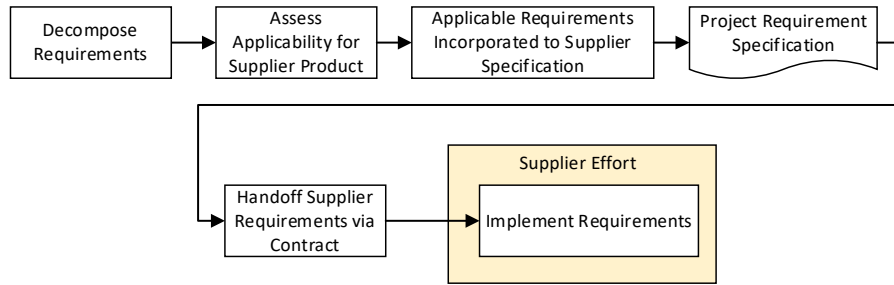


Figure 115. Project Team Effort to Compile Comprehensive Supplier Specification.

Depending on the requirements management tool and usage of requirements attributes, it can be straightforward for the project team to generate into a comprehensive requirements package for the supplier. Most often the costs of labor to generate this condensed document is less than the cost the supplier will charge to address a multiple set of documents levied upon them.

5.5.3 Assessment to Ensure Minimized Set of Good Quality Requirements

To expand on the *INCOSE Guide for Writing Requirements* characteristics, it is proposed that the requirement set for a system of interest be assessed against the following criteria to ensure there is a minimized and good quality set of requirements for the system of interest:

- Consists of design-input requirements appropriate for the level of abstraction;
- Consists of all applicable requirements in a single location;
- Addresses all system needs and includes necessary capabilities, characteristics, constraints, interfaces, standards, environment conditions, regulations, and quality factors;
- Contains unique requirements which do not conflict with or overlap with others in the set,
- Usage of specifying design-outputs (such as design solutions and design standards) is limited to conditions where design features are mandatory (driven by regulations or need for modular open systems);
- Usage of specifying design-output processes (such as a how to design, manufacture, analyze, test, etc.) is limited unless the specific intent is to mandate these activities; and
- Avoids mixing system of interest requirements with business processes (defining what is to be delivered, how the project reports progress, etc.); utilizes different methods of documentation and communication for business processes compared to design and development of a system of interest.

5.5.4 Cost Optimization with Minimization and Consolidation of Requirements Approach

As shown in Section 4.1.3, systems engineering labor cost is a function of the number of system level requirements; this in turn is based on data from numerous space mission development efforts showing the correlation of requirement quantity to development costs. Essentially, this is capturing the cost of efforts behind the development, management, implementation, verification and validation activities associated with every formal requirement. When transforming requirements from a set of needs, it is critical to realize that each requirement is responded to by systems engineers as well as other members of the project, especially for a product that is generated through a contractual effort. The more requirements specified, the greater the overall cost of the design and development efforts.

Looking at the space project examples, there is calculated reduction in costs when the requirement rating in COSYSMO was nominal, compared to difficult. Having overlapping, redundant requirements, and requirements not deriving from a need, are contributing factors for a "difficult" weighting. The effort to reduce the amount of requirements lessens the overall work later in responding to the requirements, and based on COSYSMO calculations for MSL and Constellation can yield 50-60% savings in overall systems engineering labor on a project. To demonstrate the difference when a minimized and consolidated process approach is applied, the activities for each method have been modeled and the labor time associated with each has been compared; the results of this effort is provided in Chapter 6.

5.5.5 Minimize and Consolidate Requirements Design Pattern

A design pattern extracted from Figure 114 is shown in Figure 116. Each of these steps uses associated labor hours based on the number of design teams and suppliers involved (and example of distributing requirements to multiple organizations is provided in Figure 15).

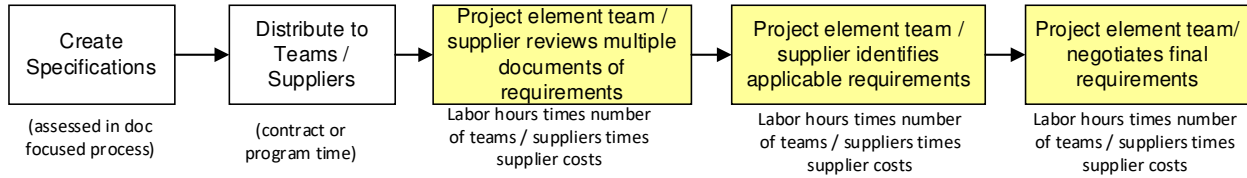


Figure 116. Non-Consolidated Requirements Design Pattern (3a).

The process to assess the requirements to a comprehensive set of minimized, yet high quality, requirements is represented as a design pattern, shown in Figure 117; this effort includes an evaluation of the requirements using the criteria provided in Section 5.5.3.

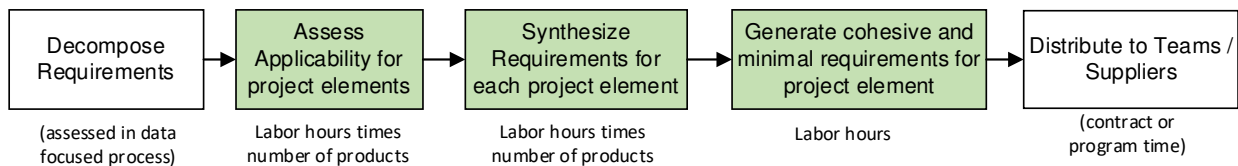


Figure 117. Minimized and Consolidated Set of Requirements Design Pattern (3b).

5.6 Process 4: Requirement Stability and Enforcement

The concept of evaluating when to levy requirements based on their measure of stability is proposed for further study based upon inputs from the subject matter experts and observations generated from the project management research associated with change control and resolving TBXs early (from Chapter 3). The section below describes the benefits of this approach and generates a process model design pattern used for evaluation in a quantitative assessment shown in Chapter 6.

When providing requirements to the teams on a project, the trend is to ensure the developing organizations have the requirements as soon as possible so that they can start their efforts and uncover design solutions sooner, allowing this design information to be used for further refinement of the requirements as described in the requirements "sandwich" (Figure 8). However, consideration should be made to address the stability of the requirements prior to levying them on the development organizations, particularly if these are contracted suppliers, as unstable requirements will drive changes that could impact the project significantly in rework time and associated cost.

Figure 118 is an extraction from the requirements management model in Figure 36, showing the key areas where requirements generation and levying on the next level of development occur.

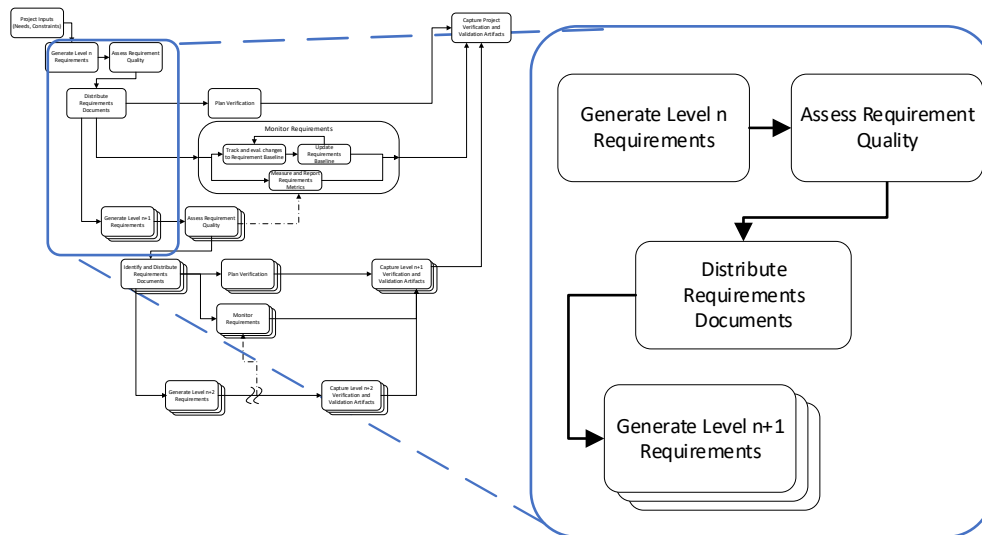


Figure 118. RM Model Activities Associated with "Stability and Enforcement".

5.6.1 Requirement Volatility Impact in Cost Calculations

Research into the change in requirements over a system life cycle, referred to as requirements volatility, was done to evaluate the impact in a systems engineering effort on a project. A set of impacts were reported, including a conclusion that there is a correlation between changes in requirements and increases in engineering effort, project cost, and schedule duration, with an increase in impact the later the change occurs in the life cycle (Pena & Valerdi, 2014). This relationship was placed into a process model, shown in Figure 119.

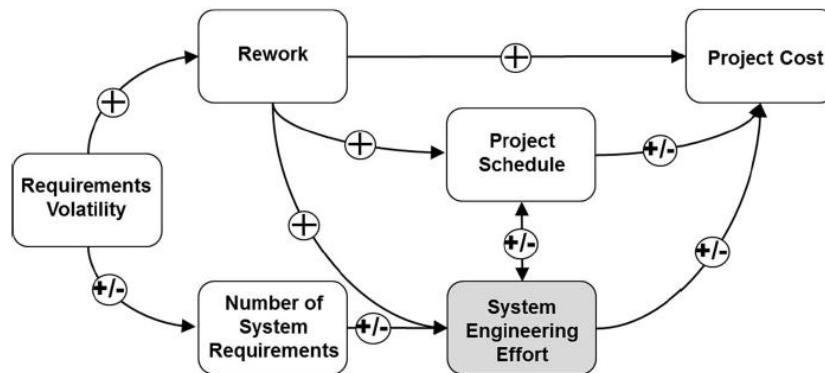


Figure 119. Impact of Requirements Volatility on Project Cost (Pena & Valerdi, 2014).

Per Figure 119, the relationship between requirements changes does have an impact on project cost as a function of added or removal of work, and as a function of occurrence in the project schedule. Per Pena, the process considers "requirements volatility as the independent variable and the number of system requirements and level of rework as dependent variables. In turn, these variables influence the amount of systems engineering effort, which is then linked to project cost and schedule. Prior studies on the value of systems engineering indicate that an increase in systems engineering effort may in fact reduce the total program cost and schedule. Conversely, unnecessary systems engineering on a program will yield diminishing returns; hence, the relationship between these variables is depicted as either positive or negative. Similarly, the relationship between requirements volatility and systems engineering effort is labeled with a +/- sign indicating that some requirements changes may reduce work scope and result in less effort." (Pena & Valerdi, 2014).

Requirements changes, brought on by any reason, will typically result in increased systems engineering effort and drive labor (this is described in Sections 3.1.3 and 3.1.4). Particularly impactful is the cost associated with the entire project associated with change assessments, rework of designs, and supplier costs associated with addressing a change (though assessments, rework, and delay of completion). When there is known work to mature the requirements the project team has the ability to put mitigations in place to reduce change costs, such as holding back on requirement enforcement while requirements are still being developed.

5.6.2 Requirement Instability Assessment

The concept of requirements instability is proposed as a version of requirement volatility where there is known work to achieve the finalized requirement content (compared to unknown changes driven by sources such as external factors, described further in Section 3.1.3). Requirement stability is an attribute associated with requirements management to classify if the requirement maturity is resolved; an indication of requirement instability is one where there is an indication of "to be resolved" or "to be determined", known as a "TBX" (refer to Section 3.1.4).

A measure of requirements instability can be viewed as a ratio of the TBX count over the total requirement count, where a high requirement instability value correlates to a high level of work to mature the requirements, and a low to zero value correlates to a set of mature requirements (shown in Equation 2).

$$\text{Requirement Instability Ratio} = \frac{\text{TBX Count}}{\text{Total Requirement Count}} \quad \text{Equation 2}$$

The impact of requirement instability on a project's cost is a function of when the requirements are officially enforced on the project team for implementation, this is described further below.

5.6.3 Impact of Enforcement of Requirements with High Instability

Because development of requirements is an iterative process, the initial set of requirements established often includes several TBXs; however, the entire set is used as a starting point for initial design work to refine requirement values through further analysis. The effort yields results which is used to provide updates to the requirements, removing the associated TBX. When development work is done by the same organization this iteration reflects a spiral design effort by a team, similar to an Agile cycle or sprint, where the updated requirements lead to maturation of a design, producing further analysis results, which may yield more refined requirements; this occurs until the TBXs are all removed and the requirements are classified as "stable". When performed by a single team this can be a coordinated effort, where design work is stalled in areas associated with the instability until further data is obtained.

However, when the development work is done by external organizations, as reflected in Figure 15, the continued update of requirements perturbs progress of the different organizations. These organizations may choose to stall work on the requirements associated with the TBXs, or they may make assumptions and proceed with their design efforts, leaning forward to a solution that may be disrupted when the TBX is ultimately resolved (causing the rework shown in Figure 119). In some cases, the developing organization can be a supplier, put on contract to design and produce a product, and motivated

to achieve a specific schedule. In many cases, providing a change to the contract will, on its own, drive a cost to the effort based on overhead costs associated with contract changes. The supplier will often be asked to assess the new requirements, update to the new requirements, and adjust design parameters (or perhaps verification plans) to adopt the new requirements. These change costs can be driven higher with lengthy resolution of TBXs, or with a large number of them. In some cases, it may be more cost effective to wait to put the supplier on contract until the requirement instability has been reduced or eliminated entirely.

5.6.4 Cost Optimization with Levying Requirements vs. Maturing Requirements Approach

The evaluation of when to enforce unstable requirements becomes an optimization calculation. The costs associated with delaying design may push the project schedule out, which can incur delay penalty cost (per Section 5.1.4). Starting a design team too soon with high requirements instability could invoke rework when the requirements stabilize, also incurring cost due to the changes (per Section 5.1.3). The decision to levy or resolve unstable requirements will vary based on expense of schedule delay associated with a later delivery of the requirements to the supplier. An optimization based on schedule delay costs and incurred direct costs for supplier changes provides a recommendation of the appropriate timing for the project, this is reflected with a first order graphic shown in Figure 120.

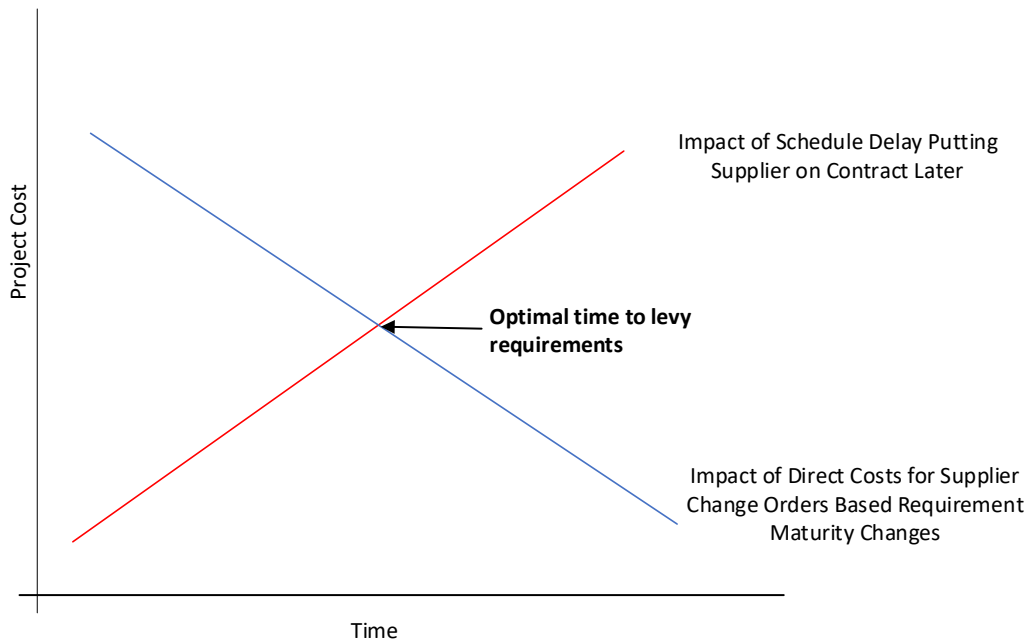


Figure 120. Decision Point of When to Levy Requirements for Suppliers.

Figure 120 provides a simple graph showing the change costs associated with levying immature requirements, which reduces as the requirement development activity improves the requirements, and the delay costs associated with waiting too long to levy the requirements on the supplier. A larger cost is associated with external suppliers put on contract than with an internal design team; therefore this effort is focused primarily on requirements levied on external suppliers. To achieve cost optimization, the timing of the requirements enforcement with the supplier should be assessed. This assessment includes looking at the project schedule, determining how late a supplier can be put on contract, evaluate the effort with resolving the requirements that are not stable and associated change costs that could be incurred, and finding an optimized time to enforce the project requirements.

For high heritage projects, the risk is lessened with the expectation of similarity. For new development projects the risk is increased when work is started too early with requirements that are not fully defined. To demonstrate the difference when this optimized wait time to levy the requirements is applied the effort the activities for each process have been modeled and the labor time and supplier costs associated with each has been compared; the results of this effort is provided in Chapter 6.

5.6.5 Requirement Stability and Enforcement Design Pattern

A design pattern associated with levying requirements with high instability is shown in Figure 121. Each of these steps uses associated labor hours or supplier based on the number of suppliers involved.

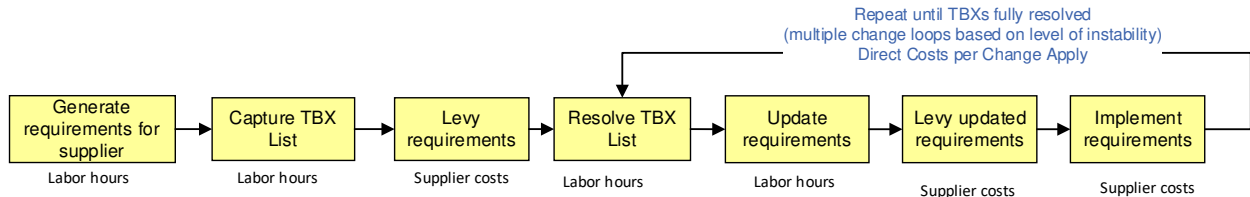


Figure 121. Levy Unstable Requirements Design Pattern (4a).

The process to assess the requirements stability and resolve some or all of the TBXs prior to levying on the supplier is also represented as a design pattern, shown in Figure 122. Parameters for this assessment will address how many TBXs should be resolved (such as limiting to key performance parameter resolution) prior to enforcing the requirements officially on a supplier. Another outcome could be to contract the supplier on a time and material effort to help resolve the TBXs, and then establish a development contract afterwards with officially enforces the requirements on the supplier's development and production effort.

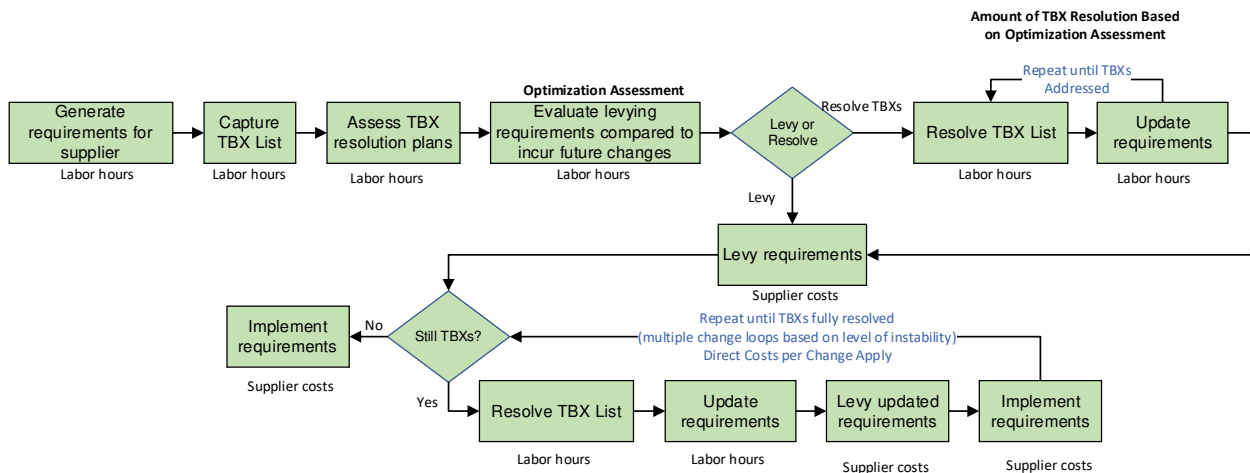


Figure 122. Optimization Assessment of Levying Unstable Requirements Design Pattern (4b).

5.7 Impact to the Requirements Management Model

Looking at how the various processes would impact the overall requirements management process model shown earlier in Figure 36, the new process steps are folded into that model for an overall revised optimized requirements management process model (Figure 123). In this figure, the impacted process steps are highlighted in green, which align with the various updated approaches shown earlier in this section.

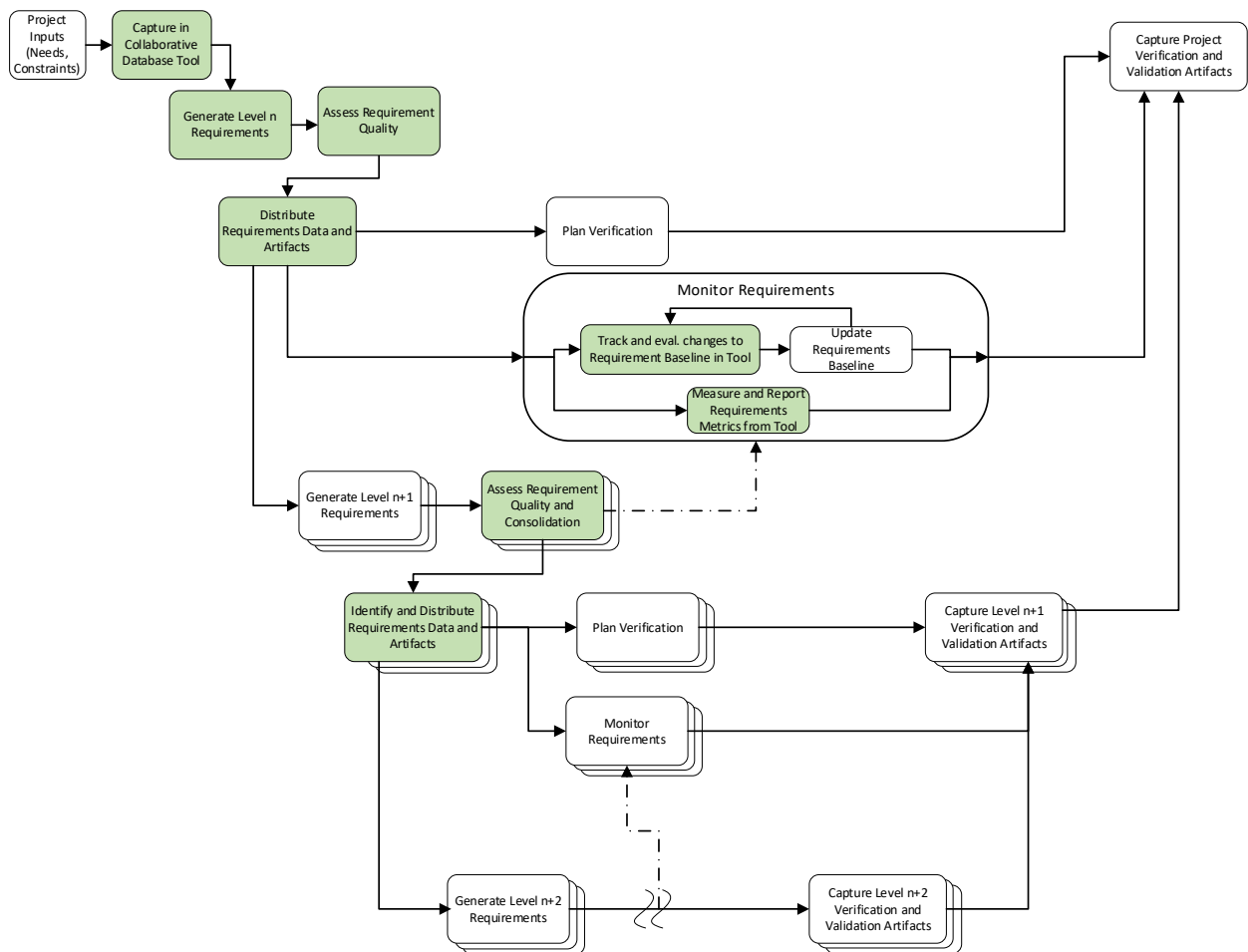


Figure 123. Requirements Management Process Model With Optimized Process Areas Highlighted.

Note that some of the verification steps could benefit from many of the methods shown in this section (such as a data centric approach to requirements management); however, to limit the scope of the validation of this model, these will be left as is. It is a potential future opportunity to show how the

recommended requirements management process updates would benefit system verification and validation cost efforts.

The next few chapters will address validation of this process model using an executable model to simulate how each of the four proposed updates would yield overall cost savings for a project compared to the expenditures required to implement and then providing an overall simulation of Figure 123 using the parameters of the space projects discussed in Chapter 4.

CHAPTER 6: DEVELOPMENT OF AN OPTIMIZED REQUIREMENTS MANAGEMENT EXECUTABLE MODEL

6.1 Generation of a Requirements Management Executable Model

Model Based Systems Engineering (MBSE) is an approach towards systems engineering which provides an alternative to the document-based systems engineering approach. With MBSE, the systems engineers perform the same life cycle activities and product similar artifacts, however with the MBSE approach the primary artifact of the engineering activities is an integrated, coherent, and consistent system model created by using a dedicated modeling tool (Delligatti, 2014). With MBSE, the other artifacts are secondary, typically automatically generated from the system model. Diagrams can be used to view the elements within the model, which have relationships and data exchanges at many levels; the diagrams are merely views of the modeled elements.

In an MBSE system model, a change made in one area will automatically propagate to all views where that element appears. An example is changing the name of a product element which propagates to views such as the definition of the product structure, behaviors of the product elements, or a requirements diagrams showing how the product satisfies its requirements.

Systems Modeling Language (SysML) is a graphical modeling language used to visualize and communicate designs of systems consisting of hardware, software, data, people and processes (Delligatti, 2014). It is an implementation approach for MBSE, and several software applications exist which enable model creation using the SysML language.

The software application utilized by the Colorado State University (CSU) Systems Engineering department for MBSE education is Cameo Systems Modeler, provided by Dassault Systèmes (formerly No Magic, Inc.). Further information on this product can be found at their website, <https://www.nomagic.com/products/cameo-systems-modeler>. An example of the Cameo Systems Modeler application is shown in Figure 124.

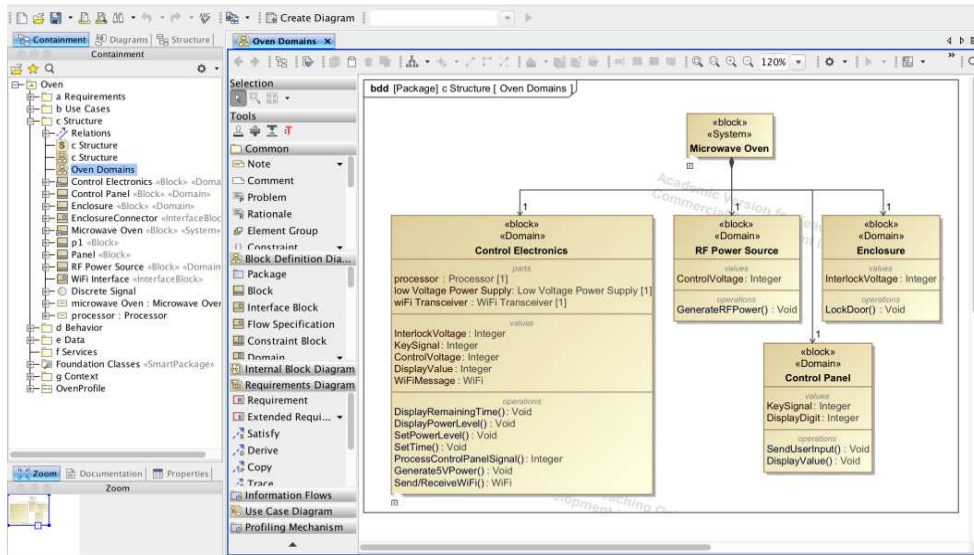


Figure 124. Example of a Cameo Model. (Borky, 2019)

Several types of model diagrams can be connected to show the interactions of the system elements. One used prominently in this dissertation is called the activity diagram, with an example shown in Figure 125.

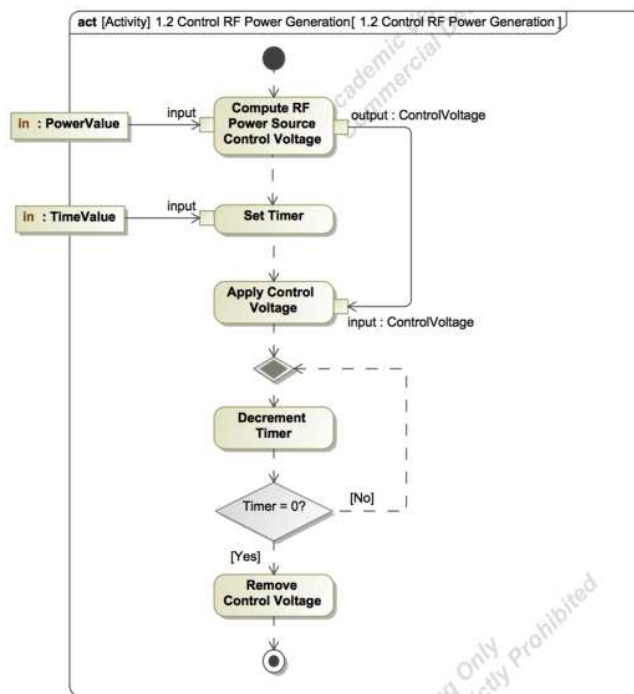


Figure 125. Example of a SysML Activity Diagram. (Borky, 2019)

Cameo has the option for several add on features, including a simulation application plug-in called Cameo Simulation Toolkit. With the Cameo Simulation Toolkit, users can test how the system reacts to user interaction or predefined testing data and execution scenarios (No Magic, Inc./Dassault Systems, 2020). In the example activity diagram from Figure 125, the simulation toolkit could be used to show the data of power and time passed throughout the activity steps, highlighting interfaces, as well as providing simulation times.

For this dissertation Cameo Systems Modeler with simulation toolkit was used to build a model of the requirements management process in SysML, enabling execution of simulations to assess the proposed requirements management processes compared to the current state activities.

6.2 Generation of an Optimized Requirements Management Executable Model

On its own the modeling of the requirements management process in SysML is informative for understanding of relationships, inter-dependencies, process actions, input and output activities, etc. However, by turning this model into an executable model more information can be obtained related to parameters of implementation, addressing questions such as:

- How long could the entire effort take if durations of each step are known?
- Is there a way to compare variations on a process step to understand impacts to the overall requirements management process duration?
- Can values based on outcomes of the effort, such as number of requirements, be fed into existing models to predict overall systems engineering labor costs of the effort?

6.2.1 Executable Model Elements

The fundamental start of building a system model is to craft its structural elements. The structure model is displayed in a block definition diagram (bdd), showing the requirements management process and its relations to other project elements as well as associated data used in the process (Figure 126). This diagram also shows an associated requirement specifying the need to meet or improve the project schedule when implementing the process; for this dissertation the requirement is shown as a reference,

future work could be done to use this requirement as part of the overall simulation to assess ability to meet this constraint.

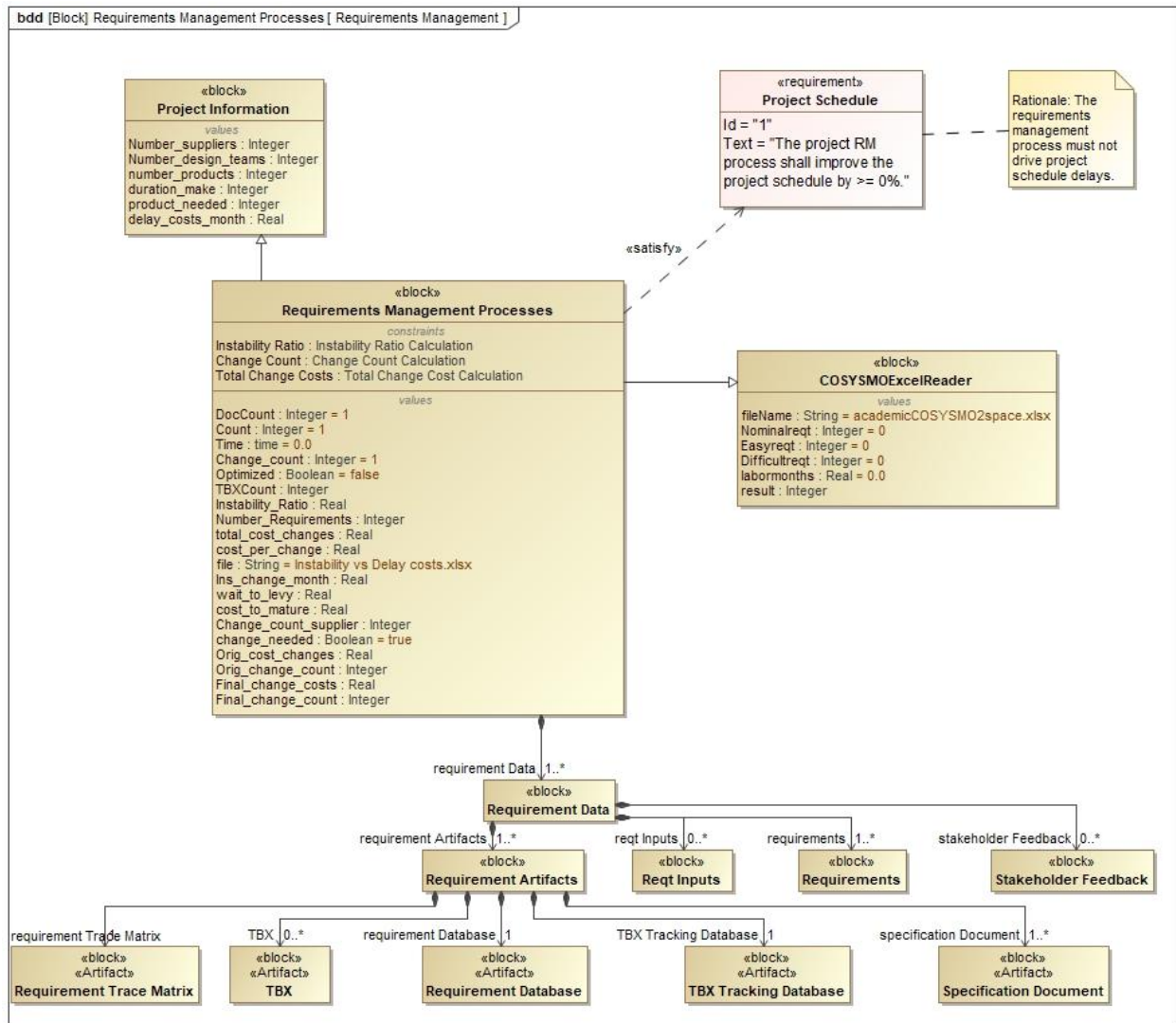


Figure 126. Overview of the Requirements Management Components in the Cameo Model.

The requirements management processes described in Chapter 5 are shown as associated process blocks in Figure 127, highlighting the current state and optimized design patterns of each process with the color coding shown in Chapter 5 (yellow for current state, green for optimized).

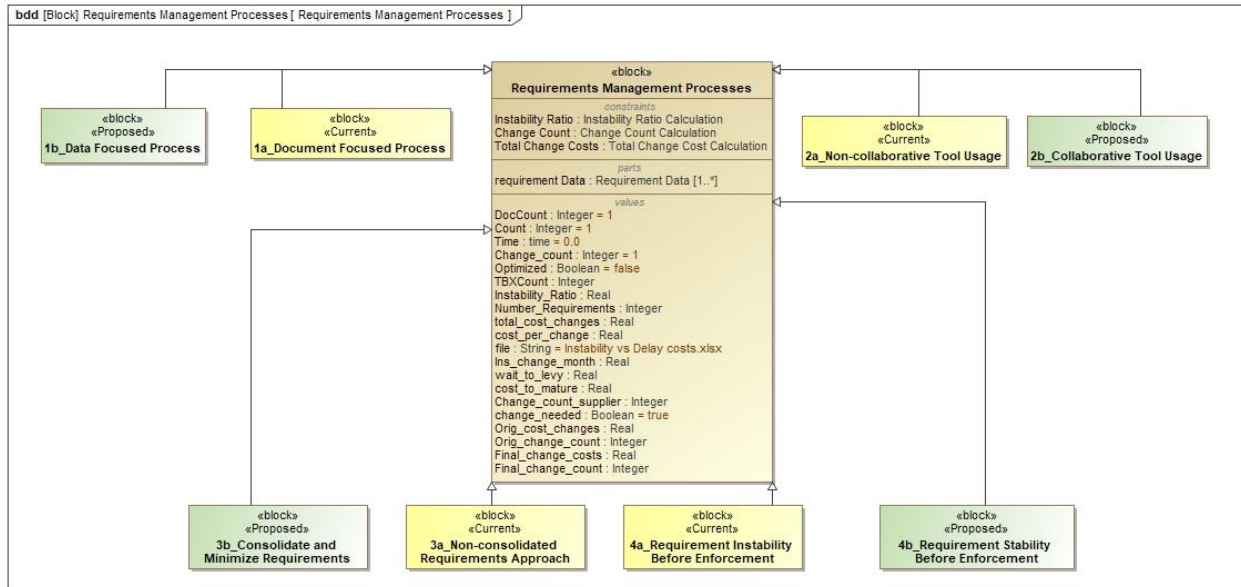


Figure 127. Requirement Management Process Activities (Current State and Optimized)

The following sections provide descriptions of the models created of the design patterns from Chapter 5, assessing each of the processes using simulations and evaluating the outcomes to determine if the proposed activities yield optimization for cost. These processes are then included in an overall process model that is further simulated in Chapter 7 using inputs from the space projects described earlier.

6.2.2 Process 1: Data Centric Requirements Management Optimization Assessment

Model Approach

The design patterns from Figure 105 (document centric process) and Figure 106 (data centric process) were created as activity diagrams in Cameo Systems Modeler as shown in Figure 128. The various process steps were shown as SysML actions with assigned times reflecting the estimate of systems engineering labor hours to implement. These hours were provided as a range of values as different experiences and circumstances may skew the activity duration slower or faster, ultimately many of these values were comparable in time with the main variation being the effort associated with managing a singular set of data compared to distributed data; the rationale for the durations used are provided later in this section.

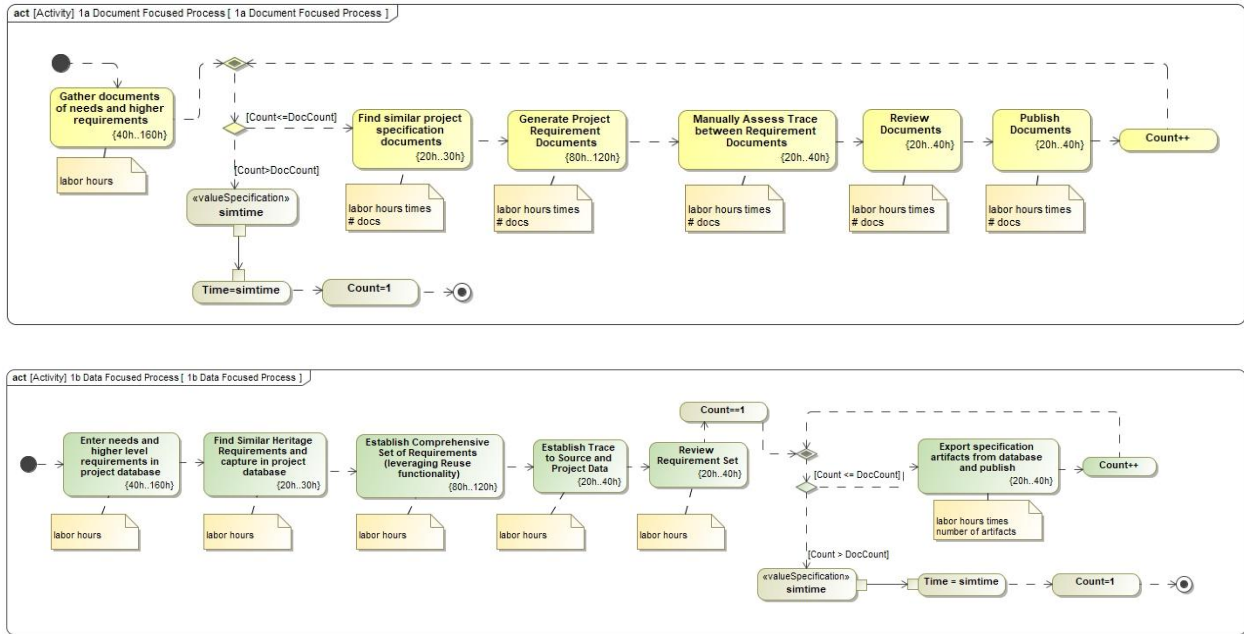


Figure 128. Document and Data Centric Requirements Management SysML Models.

Considering that the implementation of these processes vary based on the number of documents involved in the requirements management process, an adjustable parameter called "DocCount" is utilized to repeat aspects of the process based on number of documents. In addition to reporting the time to the console window, this activity also captures it to a value parameter called "Time", allowing this value to become part of the model data set.

Using Cameo Simulation Toolkit the activity diagrams can be executed as a simulation; a simulation configuration is created for each of the activity diagrams that establishes various parameters for the simulation (Figure 129). The parameter "clock ratio" sets the rate of the simulation execution (real time or faster); Figure 129 shows a clock ratio of 0.000003 which enables a much faster than real time execution of the activity steps. The figure also shows a parameter for "durationSimulationMode" which enables the simulation to occur at the minimum times, the maximum times, the average times, or at random times provided for the process activities.

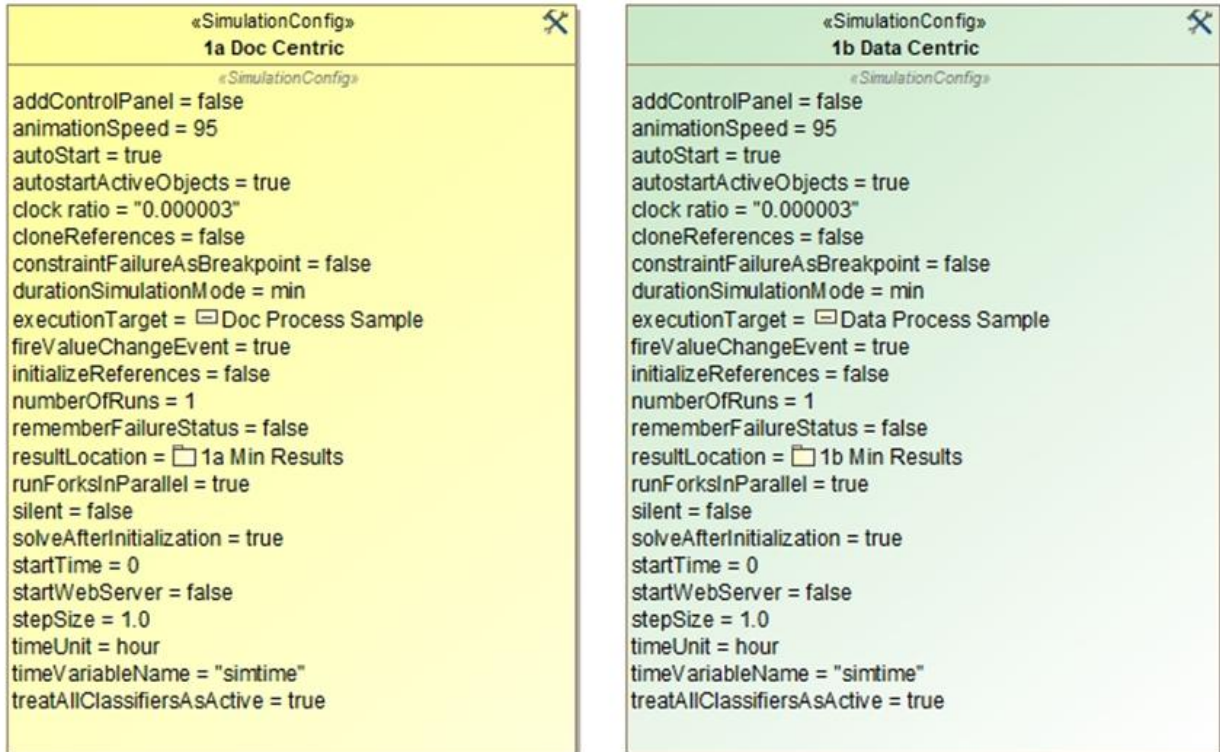


Figure 129. Document and Data Centric Cameo Simulation Toolkit Simulation Configurations.

Simulation Results

Using these activity diagrams and varying the inputs for number of documents, the simulation for each process is executed based on a range of durations (minimum and maximum) as well as document count. Figure 130 shows a screen capture of the process 1a simulation, showing the visual representation of the activities being executed as well as a display of various value parameters (lower right of the screen) during the simulation. Figure 131 provides a capture of the visual timeline created during the process simulation, showing the durations of each step and the overall activity.

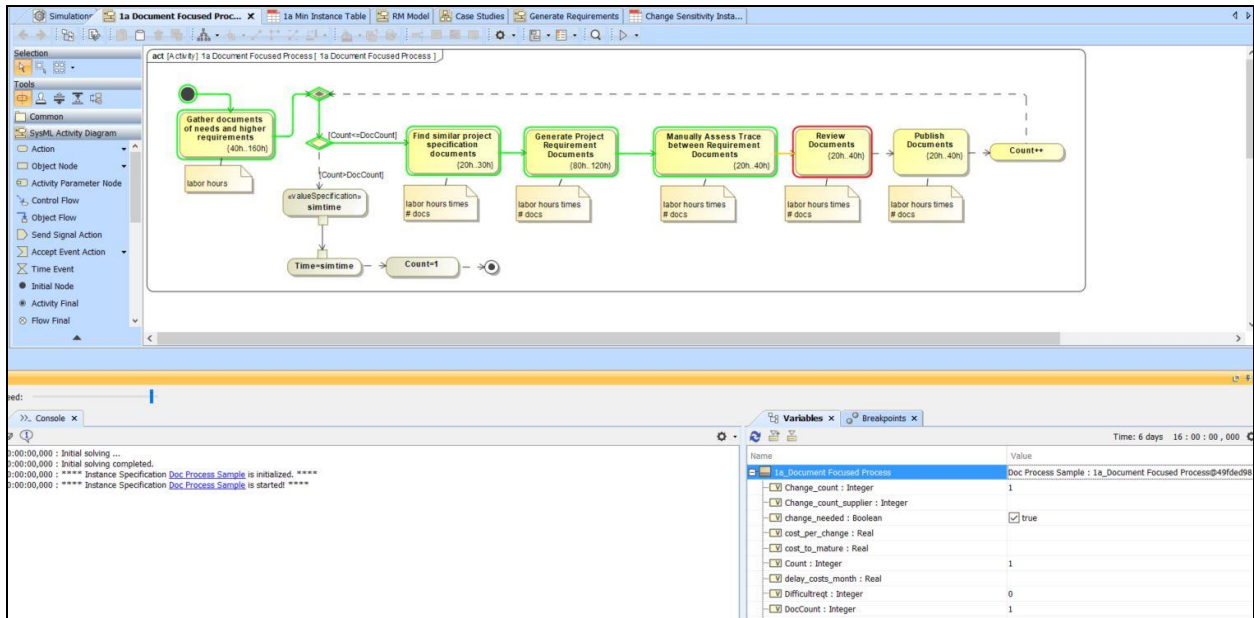


Figure 130. Process 1a Simulation Screen Capture.

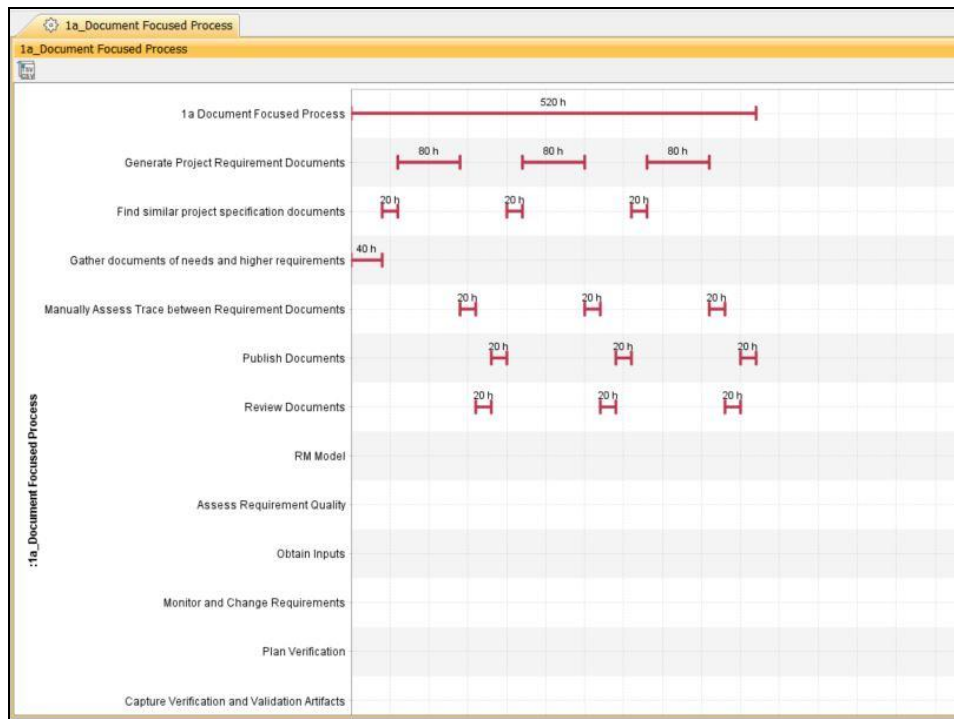


Figure 131. Process 1a Simulation Duration Timeline Data (Minimum Process Duration).

Figure 132 provides the resultant data tables produced from each simulation run for the processes at the minimum task durations, showing the resultant duration in hours based on variation of the document quantity; similar data was gathered for the runs at the maximum task durations.

#	Name	DocCount : Integer	Time : time
1	Doc Process Sample	1	0
2	1a_Document Focused Process at 2021.01.17 14.04	1	200
3	1a_Document Focused Process at 2021.01.17 14.05	2	360
4	1a_Document Focused Process at 2021.01.17 14.06	10	1640
5	1a_Document Focused Process at 2021.01.17 14.06	20	3240
6	1a_Document Focused Process at 2021.01.17 14.07	30	4840

#	Name	DocCount : Integer	Time : time
1	Data Process Sample	1	0
2	1b_Data Focused Process at 2021.01.17 13.57	1	200
3	1b_Data Focused Process at 2021.01.17 13.56	2	220
4	1b_Data Focused Process at 2021.01.17 13.51	10	380
5	1b_Data Focused Process at 2021.01.17 13.52	20	580
6	1b_Data Focused Process at 2021.01.17 13.56	30	780

Figure 132. Process 1a and 1b Simulation Duration Data Tables (Minimum Process Durations).

After conducting several simulations varying the document number and simulation duration mode (minimum and maximum), the data was analyzed to assess overall trends. The results of the process 1a and 1b simulations are represented graphically in Figure 133.

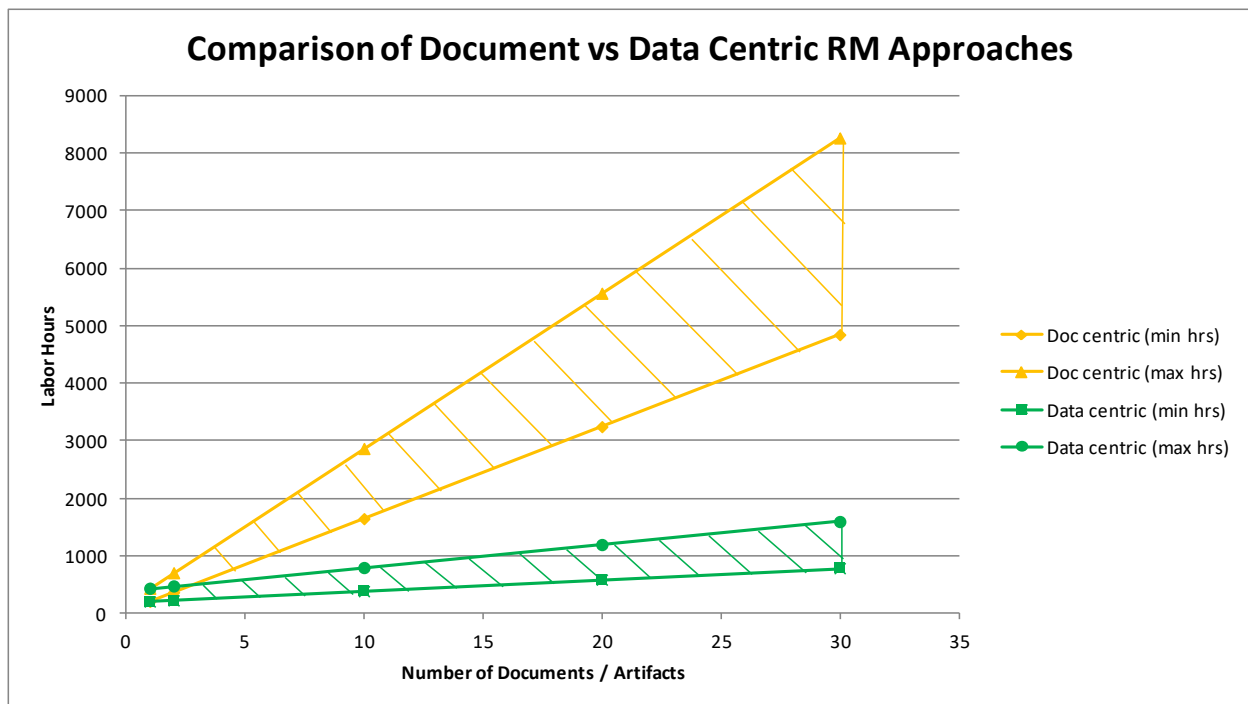


Figure 133. Data Centric Process Optimization Assessment Results.

The results of this simulation provides the project with data to assess whether the project should invest in pursuing a particular process method (discussed in Section 5.1). As the project considers the cost of implementing a data centric approach, there can be a clear benefit to implementing it for a product containing a high number of requirement documents, there may be less benefit of implementing the approach for projects that have very few requirement documents (small products with fewer requirements, as example).

Sensitivity and Data Validity Discussion

Assessing the variables for the data centric and document centric requirements management simulations, the following parameters are subject to scrutiny for sensitivity and data validity checks:

- Hours for each task
- Number of documents

With respect to sensitivity, the analysis utilizes a range of values to show impacts of applying different task durations or document count for the activities. The resulting data in Figure 133 shows a duration range, instead of a single data line, to account for the variation in process times and to establish how much this can impact the outcome. A sensitivity analysis of these parameters yields the following observations:

- For the data centric requirements management process, the task hour range appeared fairly consistent with an increase in number of exported document artifacts.
- For the document centric requirements management process, the task hour range varied significantly with the increase in number of documents.

In assessing why the document centric approach has a larger variation of time results, the tasks shown in Figure 128 reflect a significant number of actions which are performed in a document centric process based upon each requirements document generated, whereas the data centric process performs the comparable requirement management activities one time using a data management tool, publishing each document artifact at the end (representing only a small portion of the overall work). Additionally, some

of the tasks have a large difference between the minimum and maximum duration values, which is compounded during the document centric simulation over a large number of documents applied.

For task duration inputs, the hours per task can be adjusted to reflect an experienced project team by reducing the values, or a less experienced team that could take longer, this is discussed in Section 5.1.2. The values assigned in Figure 128 are shown in further detail in Table 25 and Table 26 along with rationale for the durations chosen.

Table 25. Document Centric Requirements Management Task Values and Rationale.

Task Name	Duration	Rationale
Gather documents of needs and higher requirements	40h..160h	Observation associated with collecting needs for an effort, going through assessment of use cases, contracts, higher documents, applicable standards; effort can take 1-4 weeks to obtain the inputs for requirement development.
Find similar project specification documents	20h..30h	Observation associated with generation of requirements on past projects with respect to researching similar projects and obtaining similar and applicable specifications to use as inputs; effort can take 2-3 days to find and obtain the data.
Generate Project Requirement Documents	80h..120h	Observation associated with the requirements development process on past projects in transforming needs to requirements for the system or product; effort can take 2-3 weeks to generate requirements.
Manually Assess Trace between Requirement Documents	20h..40h	Observation associated with prior analysis of looking at requirements, comparing to other documents and sources of data, discussion among team members; effort can take a half to a full week of effort among one or two personnel
Review Documents	20h..40h	Observation associated with performing reviews of several documents, including table top and email correspondence; effort can take a half to a full week of effort among multiple personnel
Publish Documents	20h..40h	Observation associated with personnel creating a finished document, applying appropriate markings, working with configuration management and obtaining all approvals; effort can take a half to a full week of effort among multiple personnel

Table 26. Data Centric Requirements Management Task Values and Rationale.

Task Name	Duration	Rationale
Enter needs and higher level requirements in project database	40h..160h	Observation associated with collecting needs for an effort, going through assessment of use cases, contracts, higher documents, applicable standards; effort can take 1-4 weeks to obtain the inputs for requirement development and put them into a project requirement database.

Task Name	Duration	Rationale
Find Similar Heritage Requirements and capture in project database	20h..30h	Observation associated with generation of requirements on past projects with respect to researching similar projects and obtaining similar and applicable specifications to use as inputs; effort can take 2-3 days to find and obtain the data.
Establish Comprehensive Set of Requirements (leveraging Reuse functionality)	80h..120h	Observation associated with the requirements development process on past projects in transforming needs to requirements for the system or product; effort can take 2-3 weeks to generate requirements.
Establish Trace to Source and Project Data	20h..40h	Observation associated with prior analysis of looking at requirements, comparing to other requirements and sources of data, discussion among team members; effort can take a half to a full week of effort among one or two personnel
Review Requirement Set	20h..40h	Observation associated with performing reviews of several requirements including table top and email correspondence; effort can take a half to a full week of effort among multiple personnel
Export specification artifacts from database and publish	20h..40h	Observation associated with personnel creating a finished document, applying appropriate markings, working with configuration management and obtaining all approvals; effort can take a half to a full week of effort among multiple personnel

The comparable tasks in processes 1a and 1b were given similar ranges of durations (such as tasks for establishing trace of requirements, or publishing artifacts). As noted in the tables, the values used in the simulation reflect this author's experiences in performing the efforts and observations of similar tasks performed by systems engineering teams in multiple projects worked by this author.

6.2.3 Process 2: Collaborative Tool Usage Optimization Assessment

Model Approach

The design patterns from Figure 111 (non-collaborative tool usage) and Figure 112 (collaborative tool usage) were implemented as activity diagrams in Cameo Systems Modeler, and the times for each activity were provided as an estimate of systems engineering labor similar to the prior section. The resultant activity diagrams are shown in Figure 134.

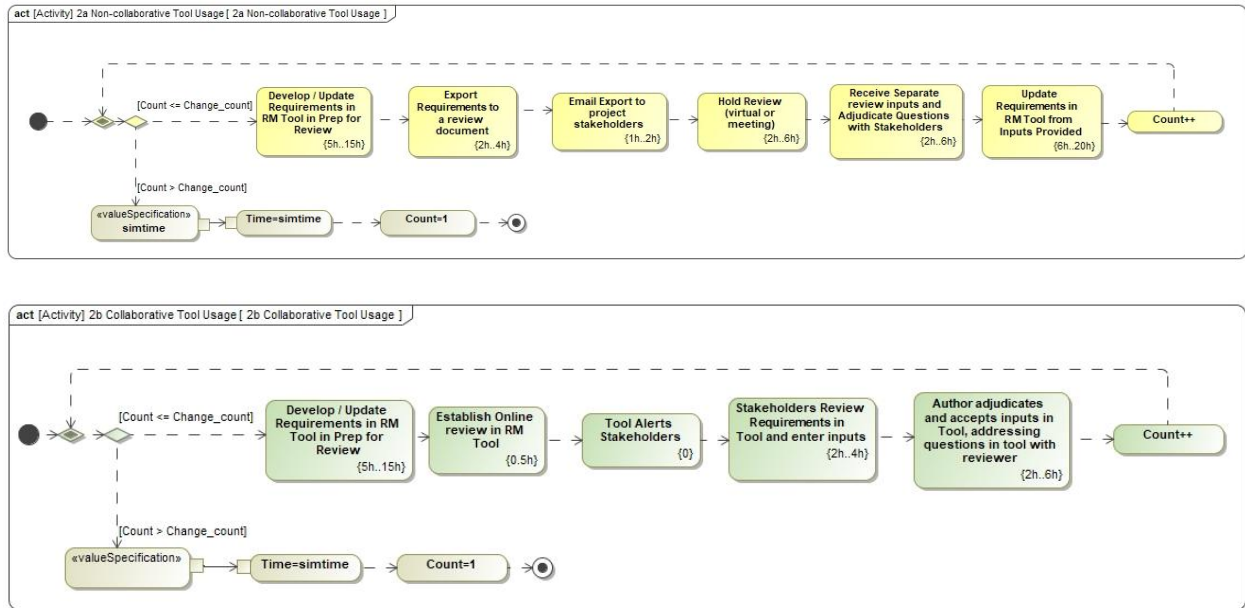


Figure 134. Non-Collaborative and Collaborative RM Tool Usage SysML Models.

Considering these processes will likely be varied by the number of changes involved in the requirements management process (necessitating reviews by the stakeholders), an adjustable parameter called "Change_count" is utilized to assess the impact based on number of requirement changes. This parameter considers a set of changes, compared to individual requirement updates; for efficiency it is common to have product change cycles address several requirements being updated and reviewed at a time.

Results

Similar to the process 1 simulations, simulation configurations were created for the process 2a and 2b activities to vary the durations. The process simulations were executed for a range of times and change counts, and results of the simulations are provided graphically in Figure 135.

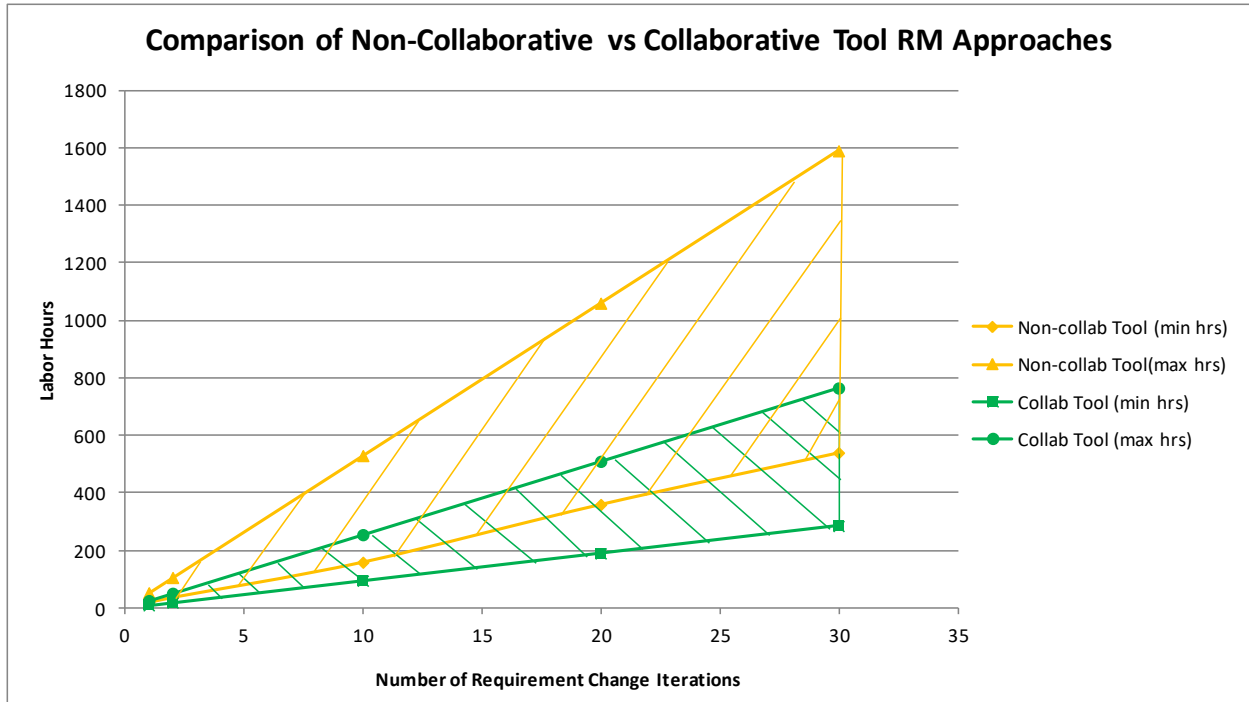


Figure 135. Collaborative RM Tool Usage Optimization Assessment Results.

The results of this assessment provides the project with data regarding whether the project should invest in a collaborative requirements management tool. As the project considers the cost of implementing a collaborative tool, some benefit of using the optimized process exists for a product containing a high number of requirement iterations; in cases where the project has many requirements with a high level of development, cost savings could be realized compared to a comparable project using a non-collaborative requirements management tool.

The aspect of this process that is more difficult to model is the requirement quality associated with each process. While the labor savings may be less compared to the prior section in using the new approach, the input from the requirements management experts noted a clear benefit to the requirement quality when multiple users are in the tool developing the requirements together. This is also demonstrated by the research conducted by Forrester Consulting described in Section 5.4.3.

Sensitivity and Data Validity Discussion

Assessing the variables for the non-collaborative and collaborative tool usage simulations, the following parameters are subject to scrutiny for sensitivity and data validity checks:

- Hours for each task
- Number of changes

With respect to sensitivity, the analysis utilizes a range of durations to show impacts of applying a minimum and a maximum value for the simulation. The resulting data in Figure 135 shows this range to account for the variation in process times and to establish how much that can impact the outcome. A sensitivity analysis of these parameters yields the following observations:

- For the collaborative tool usage process, the effort showed little sensitivity to the range of duration with an increase in number of changes.
- For the non-collaborative tool usage process, the results varied significantly with the increase in the number of changes.

The process steps were studied to assess why results would vary with the non-collaborative tool usage approach; the tasks shown in Figure 134 reflect a large duration variation from minimum to maximum for applying review inputs into the tool - this range is explained further in Table 27 and reflects the variability associated with manual entry of review data, leading the minimum and maximum times to vary greatly over a large number of changes. As the change count increases, this task is repeated and the variation is accumulated.

For data inputs, the hours per task can be adjusted to reflect an experienced project team, or a less experienced team that could take longer, as discussed in Section 5.1.2. The values assigned in Figure 134 are shown in further detail in Table 27 and Table 28 along with rationale for the durations chosen.

Table 27. Non-collaborative Tool Usage Task Values and Rationale.

Task Name	Duration	Rationale
Develop / Update Requirements in RM Tool in Prep for Review	5h..15h	Observation associated with the requirements development process on past projects in preparing requirement review content; effort can take 1-2 days.

Task Name	Duration	Rationale
Export Requirements to a review document	2h..4h	Observation associated with the requirements development process on past projects in exporting review artifacts; effort can take a few hours.
Email Export to project stakeholders	1h..2h	Observation associated with the requirements development process on past projects in sending review artifacts; effort can take a couple of hours.
Hold Review (virtual or meeting)	2h..6h	Observation associated with the requirements development process on past projects in holding a table top review with stakeholders; effort can take a few hours.
Receive Separate review inputs and Adjudicate Questions with Stakeholders	2h..6h	Observation associated with the requirements development process on past projects in assessing and addressing all review inputs from stakeholders; effort can take a few hours.
Update Requirements in RM Tool from Inputs Provided	6h..20h	Observation associated with the requirements development process on past projects in updating requirements based on review inputs from all stakeholders; effort can take 1-3 days based on amount of requirements and changes requested.

Table 28. Collaborative Tool Usage Task Values and Rationale.

Task Name	Duration	Rationale
Develop / Update Requirements in RM Tool in Prep for Review	5h..15h	Observation associated with the requirements development process on past projects in preparing requirement review content; effort can take 1-2 days.
Establish Online review in RM Tool	0.5h	Observation associated with the requirements development process on past projects in setting up a review feature in an online tool; effort can take less than one hour.
Tool Alerts Stakeholders	0	Autonomous tool operation
Stakeholders Review Requirements in Tool and enter inputs	2h..4h	Observation associated with the requirements development process on past projects in holding a virtual table top review with stakeholders; effort can take a few hours but slightly less if the reviewers are working directly in the tool for the requirements applicable to their interests.
Author adjudicates and accepts inputs in Tool, addressing questions in tool with reviewer	2h..6h	Observation associated with the requirements development process on past projects in assessing and addressing all review inputs from stakeholders; effort can take a few hours; for this item the review adjudication is addressed real time in the tool by accepting the review inputs.

The comparable tasks in processes 2a and 2b were given similar ranges of durations (such as tasks for preparing requirements for review and adjudicating review input). As noted in the tables, the values used in the simulation reflect this author's experiences in performing the efforts and observations of similar tasks performed by systems engineering teams in multiple projects worked by this author.

6.2.4 Process 3: Consolidation and Minimizing of Requirements Optimization Assessment

The efforts for consolidating and minimizing requirements leverages the use of the COSYSMO cost model tool, discussed in Section 4.1.3, which estimates systems engineering labor as a function of requirement quantity and quality. The results from COSYSMO are used in combination with the activity durations of the process model from Chapter 5 to assess this process update.

Calculations from COSYSMO

From Section 4.1.3, the parameters of COSYSMO can provide indication of requirement quality:

- **Easy** is defined as simple to implement, traceable to source and singular;
- **Nominal** is defined as familiar, able to trace to source with some effort, and containing some overlaps; and
- **Difficult** is defined as complex to implement, difficult to trace to source, and containing a high degree of overlap.

Looking at the case studies in Table 23, many of the less successful projects had an average of 10% **difficult** requirements, meaning these were complex to implement, hard to trace, and had a high degree of overlap. From the perspective of minimizing and consolidating requirements the aim is to reduce this amount in such a way the difficult requirements are primarily those that could be a challenge to implement, but more singular in nature. The rating of **nominal** implies some overlaps, but also a more realistic and conservative scenario for a project compared to usage of the **easy** rating. For this analysis the easy rating will not be considered, allowing a comparison of high degree of overlap and some overlap with respect to project requirements.

For this input a constant set of values for system interfaces, algorithms and operational scenarios is applied (utilizing the same values from the space project assessments in Section 4.1.3). Figure 136 displays the starting configuration of COSYSMO for the values, with an initial input of zero requirements; note that the starting point of zero requirements yields a value of 43.1 months based on the usage of interfaces, algorithms and scenarios.

		Easy	Nominal	Difficult
7	# of System Requirements			0
15	# of System Interfaces		5	
23	# of Algorithms		10	
31	# of Operational Scenarios		5	

		N	1.00
42	Requirements Understanding	N	1.00
43	Architecture Understanding	N	1.00
44	Level of Service Requirements	N	1.00
45	Migration Complexity	N	1.00
46	Technology Risk	N	1.00
47	Documentation	N	1.00
48	# and diversity of installations/platforms	N	1.00
49	# of recursive levels in the design	N	1.00
50	Stakeholder team cohesion	N	1.00
51	Personnel/learn capability	N	1.00
52	Personnel experience/continuity	N	1.00
53	Process capability	N	1.00
54	Multisite coordination	N	1.00
55	Tool support	N	1.00
			1.00 composite

SYSTEMS ENGINEERING PERSON MONTHS

Figure 136. Configuration of COSYSMO for Input Values.

Using a constant requirement count while changing the ratio of difficult and nominal requirements, a range of values for SE labor can be obtained from COSYSMO. Figure 137 shows how addressing consolidation into a more singular set of requirements by reducing the overlaps can result in systems engineering cost savings based on the COSYSMO cost model. From this cost model assessment, going from fully nominal to a fully difficult set of requirements can increase the labor cost by almost 300%.

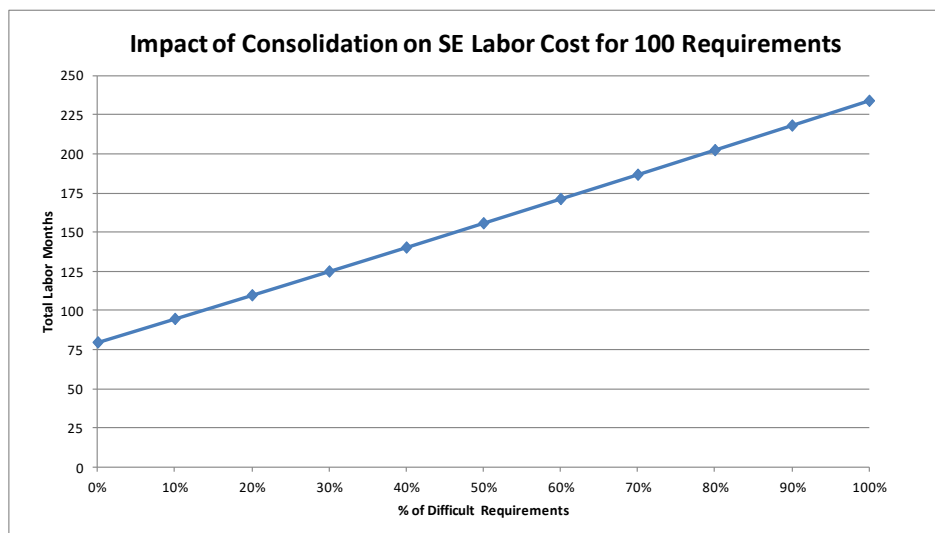


Figure 137. Impacts of Changing the Consolidation of Requirements in COSYSMO.

To assess minimization, COSYSMO is used to show a comparison of requirement quantity. The initial SE labor for zero requirements is used to normalize the other factors. The impact of adding requirements and the resultant increase in SE labor hours is shown in Figure 138.

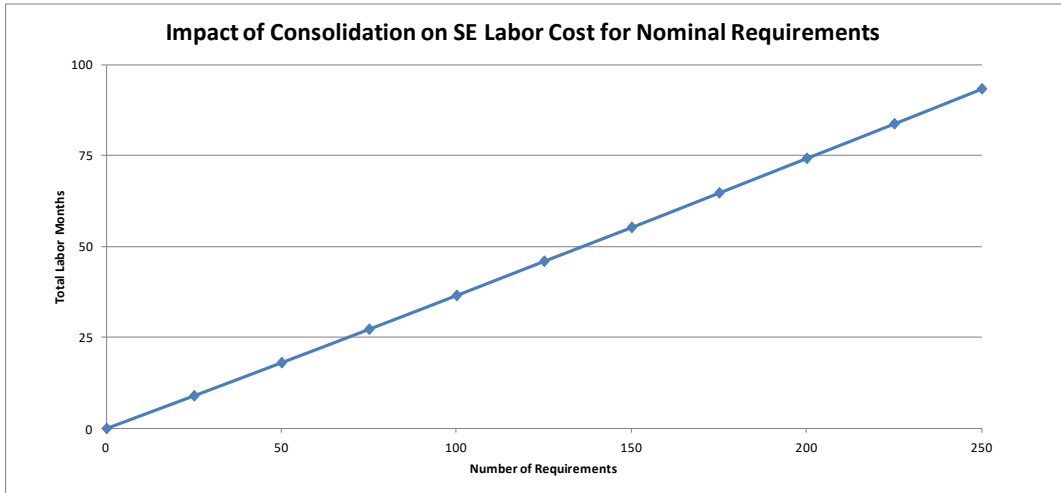


Figure 138. Variation of Requirement Quantity from COSYSMO (low number of requirements).

For each increase in 25 nominal requirements, the SE labor increases by approximately 9 months. This is not perfectly linear, as applying higher values of requirements (500, 1000, etc) yields a greater difference for each increase of 25, however the variation is minor and a linear estimation is a good first order approximation, as shown in Figure 139.

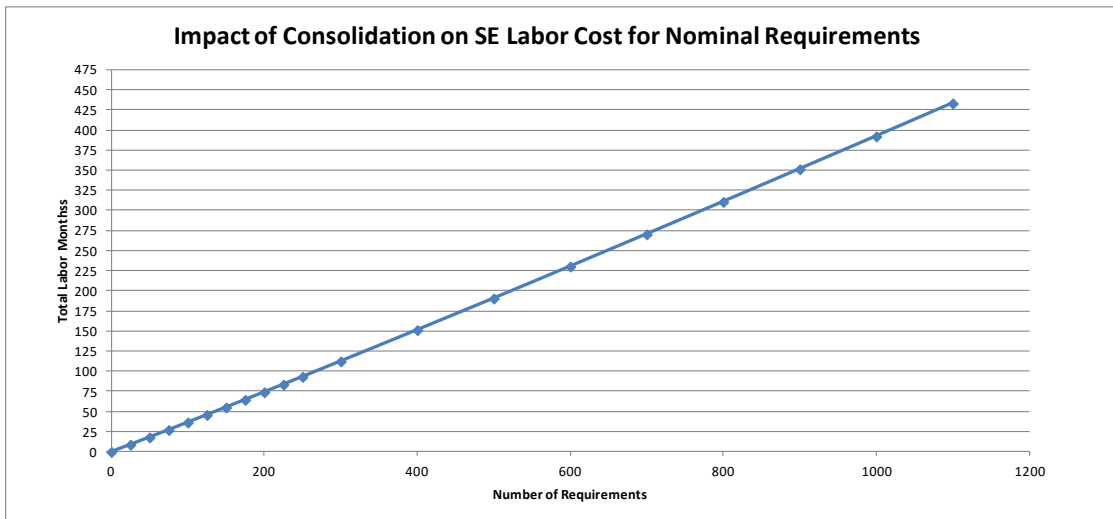


Figure 139. Variation of Requirement Quantity from COSYSMO (large number of requirements).

While consolidation and minimization looks like an obvious approach to yield cost savings, there is an impact associated with the effort to consolidate and minimize requirements that can be shown in terms of process steps and time to implement.

Model Approach

The design patterns from Figure 116 (non-consolidation of requirements) and Figure 117 (consolidation of requirements) were implemented as activity diagrams in Cameo Systems Modeler, The resultant activity diagrams are shown in Figure 140.

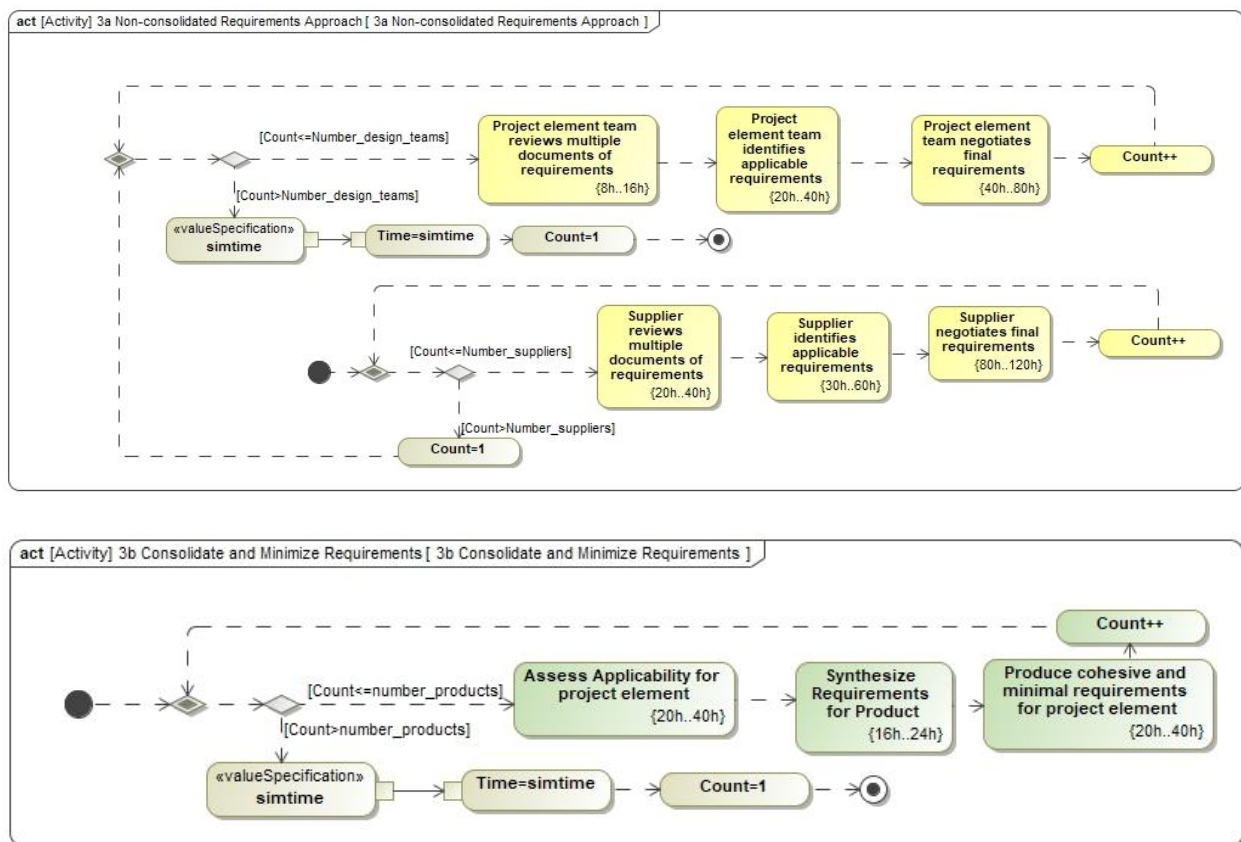


Figure 140. Non-Consolidation and Consolidation of Requirements SysML Models.

For the non-consolidated activity, most of the work to implement the effort is by the responding design teams and suppliers. The hours of the suppliers doing the work is skewed higher based on their unfamiliarity with the project documentation as well as the impact of negotiation boundaries across different companies. The direct costs of the suppliers doing this extra labor would also be realized in

their contract costs, these are not considered in the simulation (these are a potential area for future study). For the consolidated activity, most of the effort is through the systems engineers generating the final requirement specifications, therefore the effort is a function of the number of products being specified.

The hours used in the activity diagrams assumed a starting value of 200 requirements per product, an average value for a typical aerospace component. For the non-consolidated process this can be spread across multiple documents. For future simulations where the number of requirements is substantially more or less the scope of hours in the tasks are able to be adjusted.

The variation for this activity includes the number of design teams, the number of suppliers, and the number of overall products. Another variation is *when* the effort is implemented; the non-consolidated requirements approach would occur after the requirements are levied on the various design teams and suppliers, incurring the work at their level to assess and understand the multiple requirement documentation, whereas the consolidated approach occurs earlier in the requirements development phase by the project systems engineers. The effect of this parameter is shown in Chapter 7 during the execution of the overall requirements management model.

Results

Simulation configurations were created for each of these activities to vary the durations, and simulations were executed for a range of times and variation of number of products, suppliers, and design teams. For this simulation it was established that there were unique suppliers and design teams for every product (a one for one relationship), so the combination of suppliers and design teams always equaled the number of products. The results of the simulations are shown graphically in Figure 141.

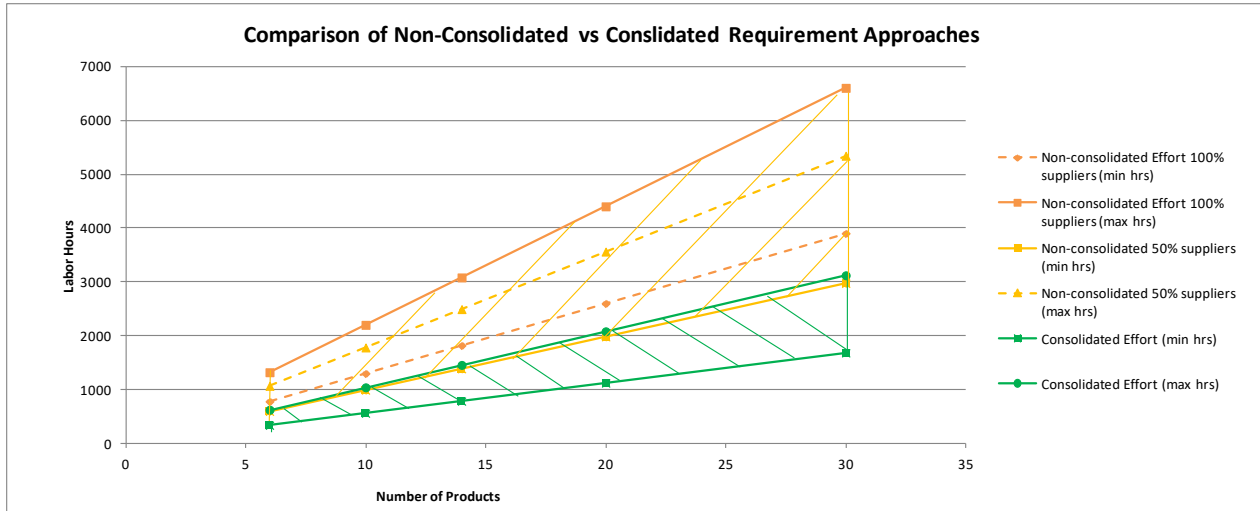


Figure 141. Consolidation of Requirements Optimization Assessment Results.

The full range of simulation results is shown in Table 29 where the variation of non-consolidated requirement impacts increases as a function of supplier developed products.

Table 29. Consolidation of Requirements Simulation Results.

# Products	Non-consolidated								Consolidated	
	100% suppliers (min hrs)	100% suppliers (max hrs)	80% suppliers (min hrs)	80% suppliers (max hrs)	50% suppliers (min hrs)	50% suppliers (max hrs)	100% design teams (min hrs)	100% design teams (max hrs)	SE Labor (min hrs)	SE Labor (max hrs)
6	780	1320	718	1236	594	1068	408	816	336	624
10	1300	2200	1176	2032	990	1780	680	1360	560	1040
14	1820	3080	1634	2828	1386	2492	952	1904	784	1456
20	2600	4400	2352	4064	1980	3560	1360	2720	1120	2080
30	3900	6600	3528	6096	2970	5340	2040	4080	1680	3120

From the table it can be observed that there is a substantial labor cost associated with efforts to consolidate the requirements with a larger number of products (right side of the table). There is a larger cost associated with having the suppliers do this effort (left side of the table). It is worth noting that if the entire development effort was done in-house, there does not seem to be an obvious benefit to having the systems engineering organization consolidate requirements at the system or product level.

Looking at this in terms of return on investment to address if it is worth doing the activity of consolidation earlier, a sample project of ten products in the product tree is assessed (similar to the

GOES-R project). Assuming one labor month contains 160 hours (an upper bound with four entire weeks in a month), the number of labor months to address ten products is provided in Table 30, showing the cost of systems engineering labor to address applicability and consolidate and minimize requirements for the products, compared to whether the suppliers or in-house design teams perform this effort.

Table 30. Requirement Costs for Ten Products, Consolidated vs. Non-Consolidated Approaches.

Activity	Duration (months)	Comment
Maximum labor time of SE consolidating and minimizing at system level for ten products	7	Labor cost to consolidate requirements
Versus...		
Maximum labor time for ten suppliers to address non-consolidated requirements	14	Supplier cost resulting from non-consolidated requirements
Maximum labor time for ten Design teams to address non-consolidated requirements	9	Design team cost resulting from non-consolidated requirements

On its own Table 30 presents the cost savings with having the systems engineering team at the system level perform an assessment and refinement of the lower level requirements, where the system will net an overall savings. With respect to outcome of the effort, Table 31 provides the systems engineering costs of the requirement activity (in labor months) along with the associated savings from COSYSMO for reducing the product requirements quantity and overlaps.

Table 31. Return on Investment for Consolidating and Minimizing 200 Requirements.

Activity	Duration (months)	Comment
Maximum labor time of SE consolidating and minimizing at system level for ten products	7	Labor cost to consolidate requirements
Yields....		
<i>Minimization only option:</i> Resulting effort reduces requirements by 10%	-8	Project labor savings if 200 requirements reduced by 10%
<i>Minimization and Consolidation:</i> Resulting consolidation goes from 20 difficult, 180 nominal requirements to 180 nominal only.	-30	Project labor savings if 200 requirements reduced by 10% and no difficult requirements remain.

Sensitivity and Data Validity Discussion

Assessing the variables for the non-consolidated and consolidated requirements approach simulations, the following parameters are subject to scrutiny for sensitivity and data validity checks:

- Hours for each task
- Number of products
- Percentage of suppliers compared to in-house design teams
- Number of requirements

With respect to sensitivity, the COSYSMO assessment already described the variation associated with requirements quantity. The activity simulation used a range of values to show impacts of applying a minimum and a maximum duration value for the simulation. The resulting data in Figure 141 shows the variation in process times while Table 29 shows a variation in who does the effort over the amount of products being developed. A sensitivity analysis of these parameters yields the following observations:

- For the consolidated requirement process, the effort showed little sensitivity to the range of duration inputs with an increase in number of products.
- For the non-consolidated tool usage process, the results varied significantly for the duration inputs over the number of products and percentage of suppliers.

In looking as to why this would vary more with the non-consolidated tool usage approach, the tasks shown in Figure 140 reflect a negotiation task where the design team and suppliers are assessing their interpretation of applicability with the requirement providers and resolving any disconnects - this negotiation can take a significant amount of time depending on the subjectivity of the teams involved. Additionally the suppliers have a contractual interface to negotiate through which additionally adds uncertainty for predicting a tighter duration. As suppliers are external to the project team they will have less familiarity with the requirements, which also accounts for a larger variation of duration associated with a higher percentage of suppliers.

For data inputs, the hours per task can be adjusted to reflect an experienced project team, or one that would take longer. The values assigned in Figure 140 are shown in further detail in Table 32 and Table 33 along with rationale for the durations chosen.

Table 32. Non-consolidated Approach Task Values and Rationale.

Task Name	Duration	Rationale
Project element team reviews multiple documents of requirements	8h..16h	Observation associated with the requirements development process on past projects in reading requirements spread over different documents; effort can take 1-2 days.
Project element team identifies applicable requirements	20h..40h	Observation associated with the requirements development process on past projects in assessing requirement application to elements and components; effort can take 3-5 days.
Project element team negotiates final requirements	40h..80h	Observation associated with the requirements development process on past projects in collaborating the final applicability of requirements with system team; effort can take 1-2 weeks.
Supplier reviews multiple documents of requirements	20h..40h	Observation associated with the requirements development process on past projects in reading requirements spread over different documents; effort can take 3-5 days labor time of supplier staff.
Supplier identifies applicable requirements	30h..60h	Observation associated with the requirements development process on past projects in assessing requirement application to elements and components; as suppliers are less familiar with requirements this effort can take 4-7 days labor time of supplier staff (longer durations due to less familiarity with requirements)
Supplier negotiates final requirements	80h..120h	Observation associated with the requirements development process on past projects in collaborating the final applicability of requirements with purchasing team; effort can take 2-3 weeks (longer durations due to inclusion of contract personnel and associated contractual agreements for scope).

Table 33. Consolidated Approach Task Values and Rationale.

Task Name	Duration	Rationale
Assess Applicability for project element	20h..40h	Observation associated with the requirements development process on past projects in assessing requirement application to elements and components; effort can take 3-5 days.
Synthesize Requirements for Product	16h..24h	Observation associated with the requirements development process on past projects in flowing down requirements for a component (does not account for derivation analysis, requirements are assumed to be a direct flow-down); effort can take 2-3 days.
Produce cohesive and minimal requirements for project element	20h..40h	Observation associated with personnel creating a finished specification, applying appropriate markings, working with configuration management and obtaining all approvals; effort can take a half to a full week of effort among multiple personnel

The comparable tasks in processes 3a and 3b were given similar ranges of durations (such as tasks for assessing applicability of requirements for the project teams). As noted in the tables, the values

used in the simulation reflect this author's experiences in performing the efforts and observations of similar tasks performed by systems engineering teams in multiple projects worked by this author.

6.2.5 Process 4: Requirement Stability and Enforcement Optimization Assessment

Model Approach

Multiple model diagrams were used to assess the requirement stability and enforcement. This includes some parametric diagrams for calculation of various parameters that are used in the overall activity diagrams.

Instability Ratio and Change Count Calculations

The executable model for the requirement stability process uses parametric SysML models to calculate parameters based on user entered values for number of requirements, number of TBX, and cost per change. The requirement instability ratio is the ratio of the number of TBX and the number of requirements (Equation 2), this is calculated using a SysML parametric diagram where Equation 2 is entered as a constraint function (shown in Figure 142 on the right side of the diagram).

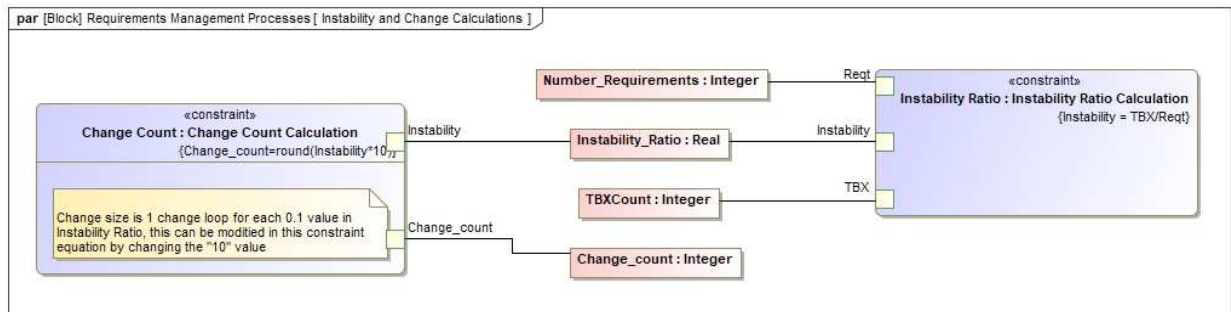


Figure 142. Instability and Change Count Calculation Parametric SysML Model.

An additional parameter of change count is also calculated in the parametric model; change count is defined as the number of changes needed to bring the instability ratio to zero, realizing a fully matured requirement set. For this effort it is assumed to be a linear function, where each change reduces the instability ratio by a set amount. change count is calculated using Equation 3:

$$\text{Change Count} = \text{Round} \left[\text{Instability Ratio} * \left(\frac{1}{\text{Change Size}} \right) \right] \quad \text{Equation 3}$$

An example of calculating the instability ratio and change count values based on given inputs are shown below.

Example Calculation:

Inputs

Number of Requirements = 100

Number of TBX = 33

Instability Ratio Change Size = 0.1

Results

Instability Ratio = $33/100 = 0.33$

Change Count = $\text{Round} (0.33 * 1/0.1) = \text{Round} (3.3) = 3$

In this example, adjusting the instability ratio by 0.1 for each change equates to 11 requirements matured for each change cycle. Several projects are capable of maturing more TBXs in a change cycle, where the adjustment could be more like a 0.2 to 0.4 change in instability ratio, therefore this is an adjustable parameter in the executable model; the change count calculation is shown in Figure 142 on the left side of the diagram. The sensitivity of adjusting the instability ratio change amount is described further in the *Sensitivity and Data Validity Discussion* section provided later.

Total Change Cost Calculations

The total costs associated with requirement changes is the cost incurred by imposing requirements changes on a supplier, which is a direct cost as described in Section 5.1.3. This value is calculated by the number of changes (change count) multiplied by the cost per change (shown in Equation 4).

$$\text{Total Change Costs} = \text{Change Count} * \text{Cost per Change} \quad \text{Equation 4}$$

The total cost due to changes is reflected in the executable model using the parametric diagram shown in Figure 143. The change count is calculated based on instability ratio as described earlier, and the cost per change is an input value that can be adjusted for a project.

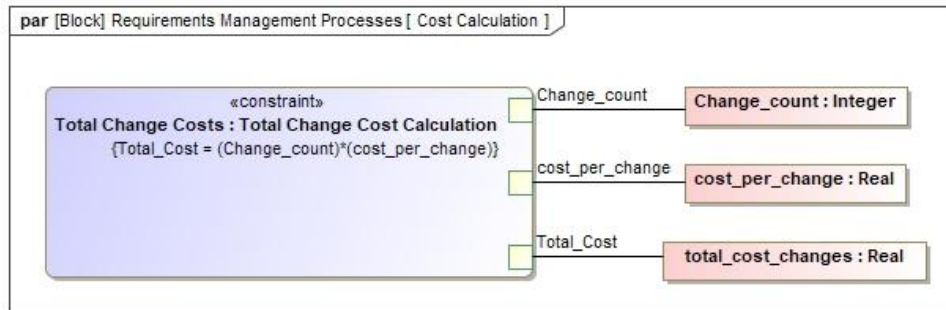


Figure 143. Total Change Cost Calculation Parametric SysML Model.

An example of calculating the total cost of changes based on given inputs is shown below.

Example Calculation:

Inputs

Cost/Change = \$10,000

Change Count = 3

Results

Total Cost of Changes = 3*\$10,000 = \$30,000

Change Costs Compared to Delay Penalty Costs

As described in Section 5.6.2, there is an optimized time to levy requirements for a supplier or design organization based upon requirement maturity and time the product is needed. From a cost perspective, Figure 120 provides a simple graph showing a high cost for levying immature requirements too soon based upon cost of changes, and a high cost of levying perfect requirements too late based upon delay costs such as penalties for late product (delay costs are based on schedule delay, and are described in further detail in Section 5.1.4). The optimal point between incurring a high cost due to changes compared to a high cost due to delay is expanded further using a Microsoft Excel model, shown in Figure 144.

	A	B	C	D	E
1	Parameters				
2	Cost/change	\$ 100,000.00			
3	Starting Instability_ratio	0.6	change per month	0.1	
4	Duration to make	10	Change Count	6	
5	Product Needed	13			
6	Must Start	3			
7	Project Delay Costs/month	\$ 150,000.00			
8					
9	Month	Instability Ratio	Costs Due to Change if Levied	Delay	Costs Due to late Reqts
10	0	0.6	\$ 600,000.00	0	\$ -
11	1	0.5	\$ 500,000.00	0	\$ -
12	2	0.4	\$ 400,000.00	0	\$ -
13	3	0.3	\$ 300,000.00	0	\$ -
14	4	0.2	\$ 200,000.00	1	\$ 150,000.00
15	5	0.1	\$ 100,000.00	2	\$ 300,000.00
16	6	0.0	\$ 0.00	3	\$ 450,000.00
17	7	0.0	\$ -	4	\$ 600,000.00
18	8	0.0	\$ -	5	\$ 750,000.00
19	9	0.0	\$ -	6	\$ 900,000.00
20	10	0.0	\$ -	7	\$ 1,050,000.00
21	11	0.0	\$ -	8	\$ 1,200,000.00
22	12	0.0	\$ -	9	\$ 1,350,000.00
23	13	0.0	\$ -	10	\$ 1,500,000.00
24	14	0.0	\$ -	11	\$ 1,650,000.00
25					
26	$y=mx+c, x=(y-c)/m$	m	c		
27	slope/intercept change cost	-100000	600000	1	
28	slope/intercept late costs	150000	-450000	2	
29					
30	Optimal Time to Levy	Supplier Change Costs	Instability Ratio	Supplier Chg Count	Internal Change Count
31	4.2	\$ 200,000.00	0.18	2	4

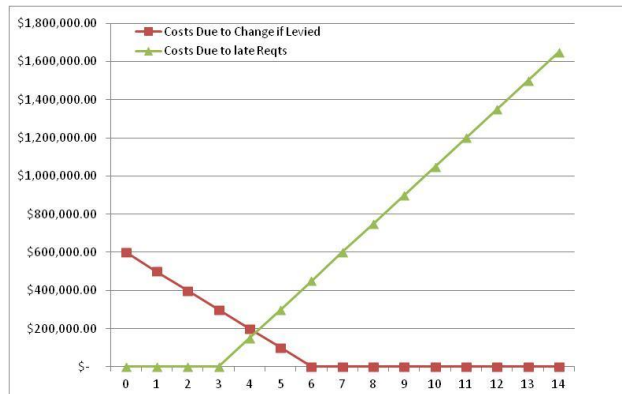


Figure 144. Change Cost Optimization Excel Model.

The change cost optimization MS Excel model, also referred to as "Instability vs Delay Costs", allows for user input for cost per change, starting instability ratio, the amount of instability ratio improvement each month, the duration to make the product, the duration until the product is needed, and delay costs for each month the product is late.

Based upon inputs provided the change cost optimization model calculates the number of months until requirement stability (considered to occur when the instability ratio is less than 0.1), the number of months until delay costs start to accumulate, generates a graph and linear equation for stability and delay costs, and then calculates the intersection of the lines to present the optimal month to levy the requirements to minimize costs. The remaining supplier change costs are also calculated, reflecting a situation where the project cannot wait for fully mature requirements and must incur some changes to avoid penalty costs.

This Excel calculation reflects the situation of officially levying the requirements on a supplier and choosing to delay enforcement; in some cases a project could provide the supplier a preview of the early requirements without formally levying as a contractual document to enable early development effort

of the product while waiting for the formal requirements, this would be a mitigation opportunity enabling earlier supplier efforts without absorbing change costs.

Requirement Process Executable Models

The design patterns from Figure 121 (enforcement of requirements with high instability) and Figure 122 (enforcement of stable requirements) were implemented as activity diagrams in Cameo Systems Modeler, the resultant activity diagrams are shown in Figure 145.

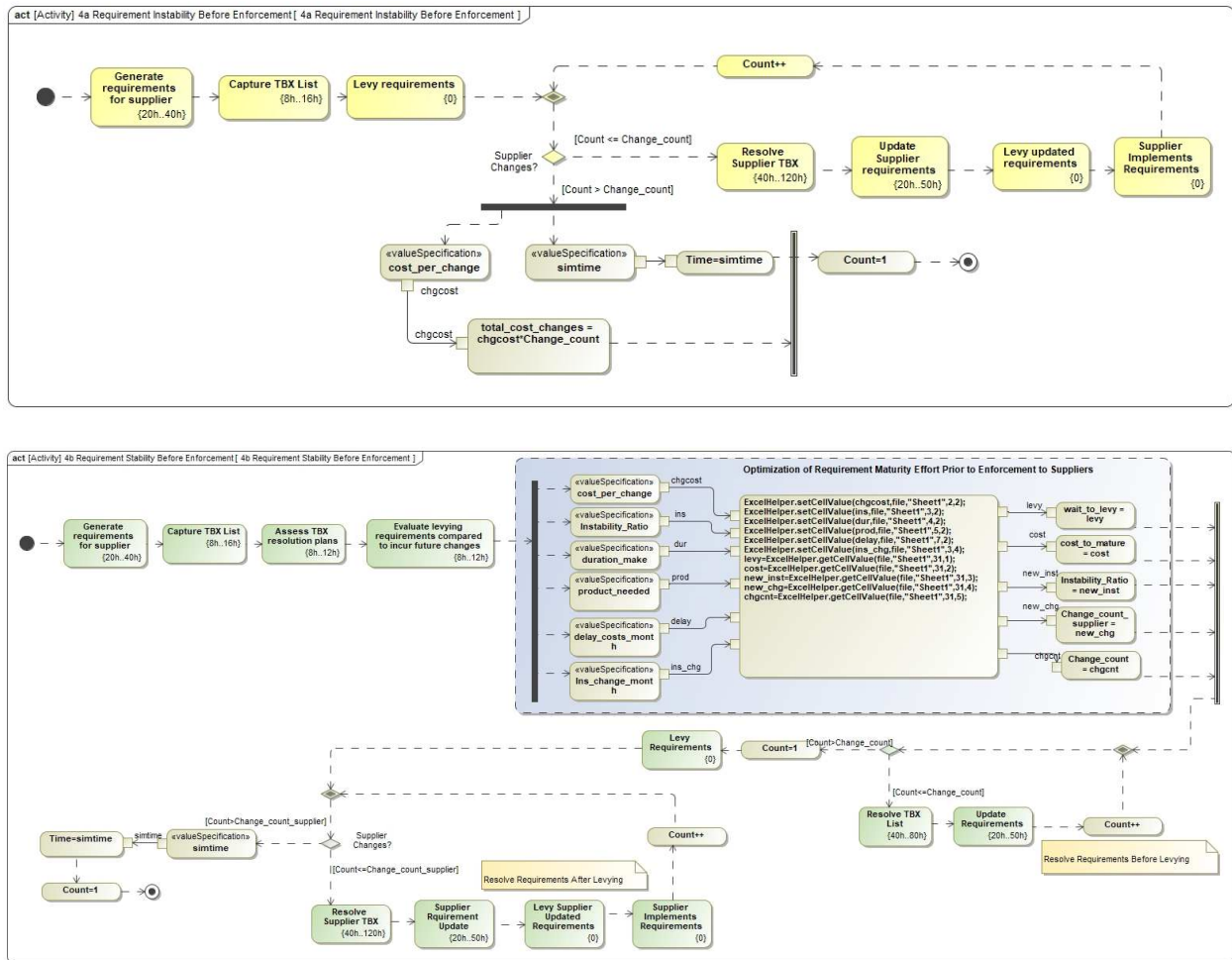


Figure 145. Instability and Stability before Enforcement of Requirements SysML Models.

For the activity of *Requirement Stability before Enforcement*, the "Instability vs Delay Costs" Excel file is used by the Cameo model to calculate an optimal time to levy the requirements. If further change cycles are needed the file also calculates the associated effort to work through the remaining TBX

requirements with the supplier; this is shown in the shaded area of the activity diagram, where inputs are supplied to the MS Excel file and resulting computations extracted and used as decision points for the activity simulation.

The labor hours being compared in Figure 145 are based on the work of assessing the TBX plans and updating requirements for a change cycle, as well as some effort for the optimization study shown in the stability process diagram. The primary input variation for these activities is the requirement instability ratio (based on TBX and requirements count). The simulation is implemented in different cases where the inputs are varied for cost per change and the delay costs, which can significantly impact the optimization outcome for time to levy the requirements. The supplier quantity is not considered for this analysis as the effort may be unique for each supplier (such as a variation of cost per change). One area for future study could be to assess impact of doing this analysis over a full set of different suppliers at different requirement maturity levels.

For simplicity a common value of duration to make the product (10 months) and time the product needed (13 months) was provided for all simulation runs, this provides a three month schedule slack for the supplier at the start of the evaluation.

Results

Simulations of the two processes were implemented for a range of times and a variation of number of TBX, cost per change, and delay costs. The results of the simulations are shown graphically in Figure 146 for labor costs of the two processes, and Figure 147 for direct costs associated with supplier change cycles of the two processes. Each figure provides a comparison of two cases varying the cost per change and delay cost, showing the impact of these variations using a low change cost with high delay cost, and a high change cost with a low penalty cost.

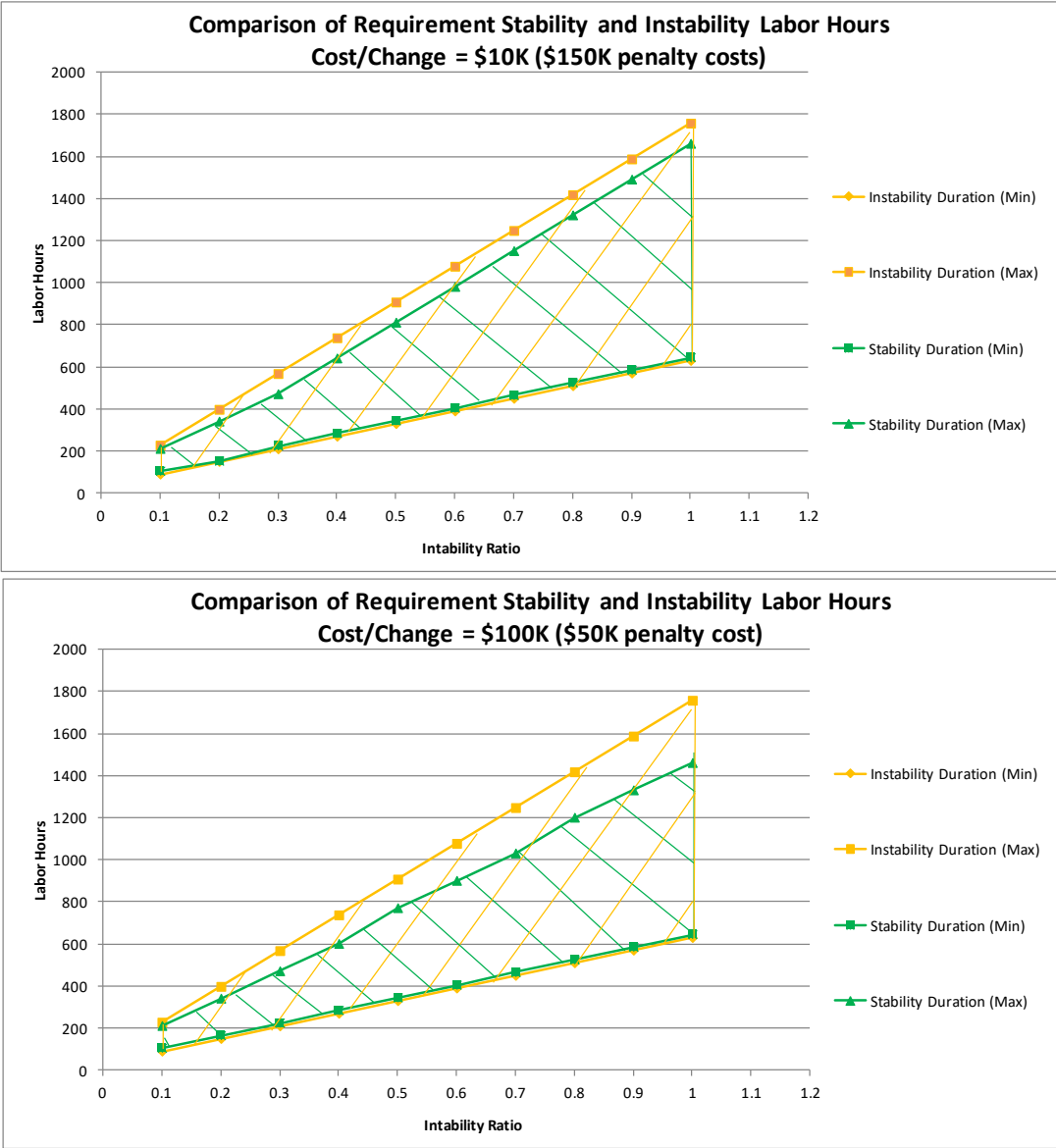


Figure 146. Labor Comparisons for Levying Unstable vs. Stable Requirements for Varying Change and Delay Costs.

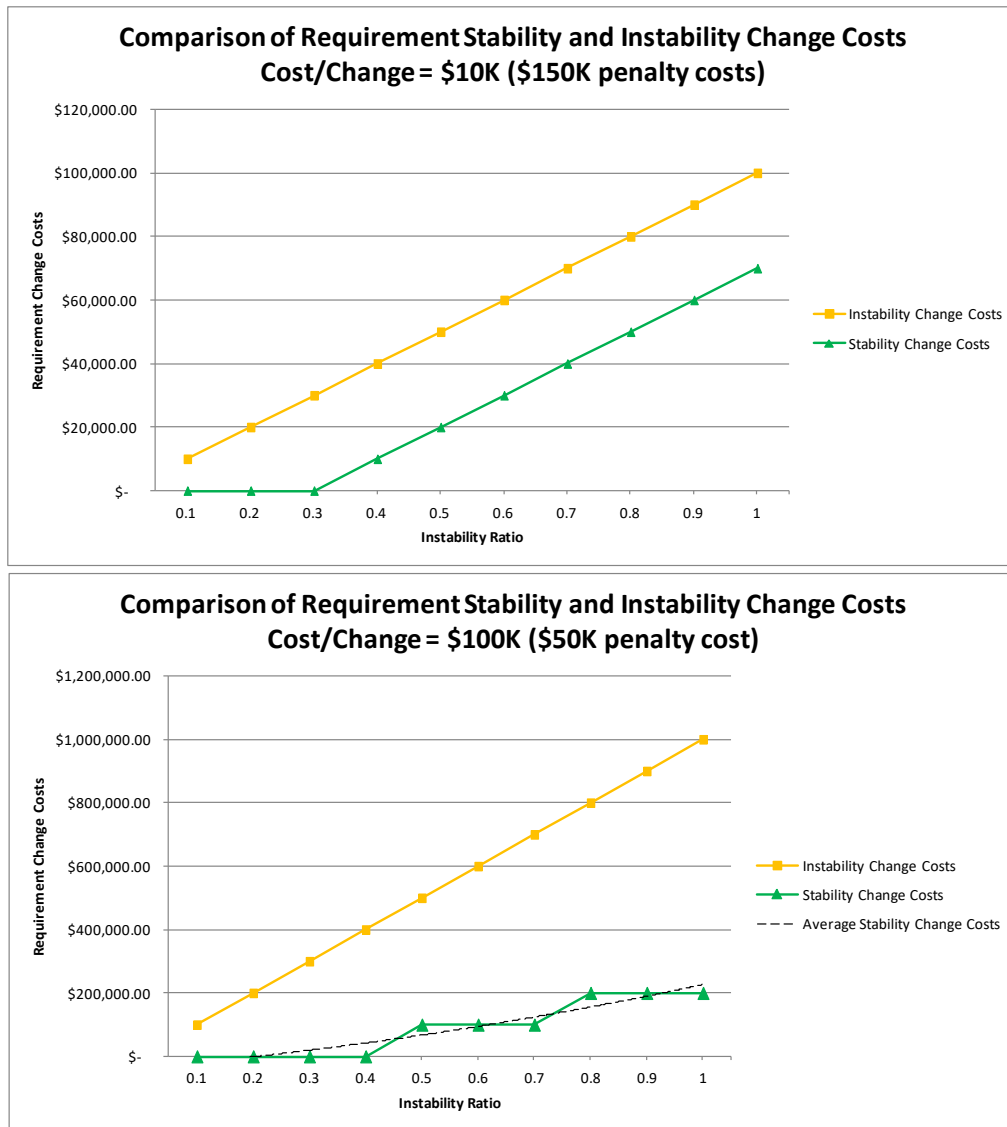


Figure 147. Cost Comparisons for Levying Unstable vs. Stable Requirements for Varying Change and Delay Costs.

It is observed from the first set of figures that there is very little difference in labor costs for the two different processes. The requirement maturity effort is fundamentally the same set of processes, and will have some impact due to involvement of suppliers in the process, but this is not a significant driver in time spent. However, as shown in the second set of figures the direct costs are improved for the optimized process. In the case of low cost per change and high delay costs this is a moderate improvement of about \$30K, but this is significantly improved to an average difference of \$600K when there is a high cost per change and low delay costs.

A comparison of optimal time to wait in Figure 148 shows that when there is a low cost per change and high penalty cost, there is limited benefit in waiting past the schedule slack; when there is a high cost per change and low penalty cost there is a large benefit in waiting for the requirements to mature before levying them on the supplier (even if there are delay costs incurred). In the simulation results shown in Figure 148 it is observed that a higher instability ratio yielded a proposed delay up to 7 months, well past the original schedule slack. This finding is based on the low penalty costs of a schedule slip of the product, allowing the time to mature the requirements. In this type of situation it is expected that the product can be late with very little consequences.

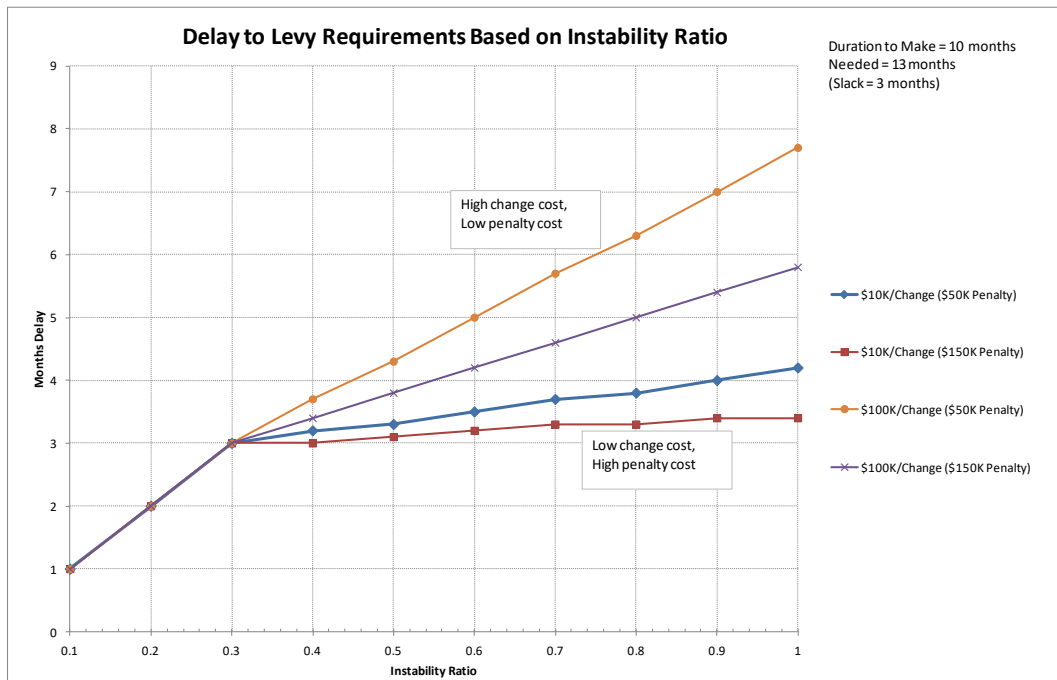


Figure 148. Levy Wait Time with Variation of Instability Ratio.

Sensitivity and Data Validity Discussion

Assessing the variables for the enforcement of unstable and stable requirements approach simulations, the following parameters are subject to scrutiny for sensitivity and data validity checks:

- Hours for each task
- Instability Ratio (number of TBX, number of requirements)

- Change Size (improvement of instability ratio each month)
- Cost per Change
- Delay Costs

To assess impact of changing the direct costs (cost per change and delay costs), four simulation cases were run varying the ratio of these costs as follows:

- Low cost per change, low delay cost (\$10K, \$50K)
- Low cost per change, high delay cost (\$10K, \$150K)
- High cost per change, low delay cost (\$100K, \$50K)
- High cost per change, high delay cost (\$100K, \$150K)

Shown in the results provided earlier, varying these parameters did not yield much difference in labor costs of the change activities (Figure 146), however it had a huge impact on the direct costs for the two different requirement management processes (Figure 147) as well as calculation of optimal time to levy immature requirements (Figure 148).

Looking at the sensitivity of the cost per change and delay costs further, Figure 149 and Figure 150 from the Excel model highlight the impact of the different ranges of cost per change and delay cost at various levels of requirements maturity. In Figure 149 the \$10K cost/change results in a small slope compared to the penalty cost over time, showing that the change cycles are not costly and that levying the requirements as soon as the schedule slack is met yields the best cost solution regardless of level of requirement maturity (instability ratio of 0.4 and 1 yields similar results).

In Figure 150, the \$100K cost/change slope varies more significantly with requirement maturity, resulting in a more dramatic change in the intersection with the line for delay cost over a comparable change in instability ratio. This leads to the observation that the optimization of requirement enforcement associated with requirements stability is most beneficial to projects with a higher cost per change associated with their suppliers.

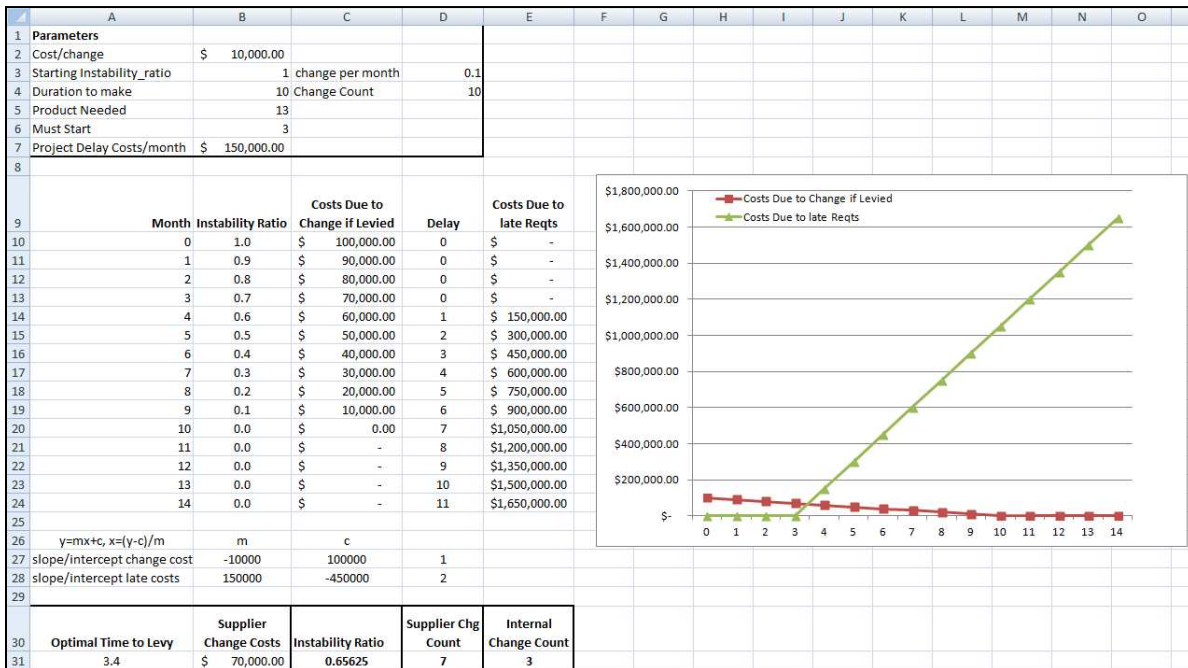
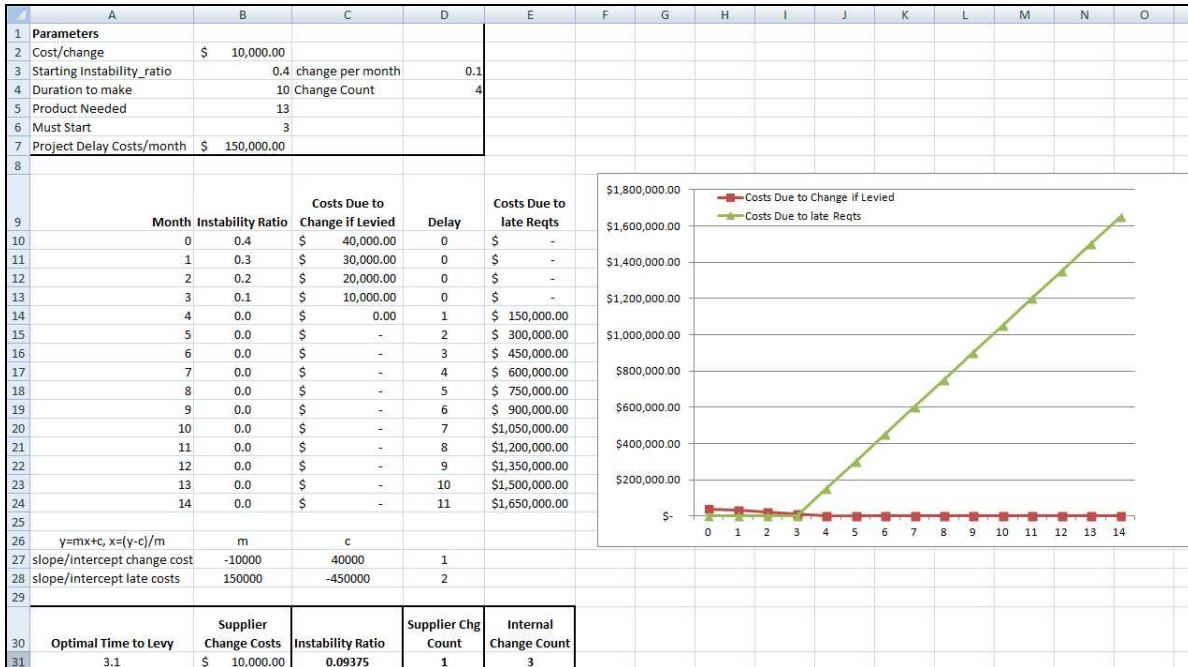


Figure 149. Impact of Instability Ratio at \$10K/change.

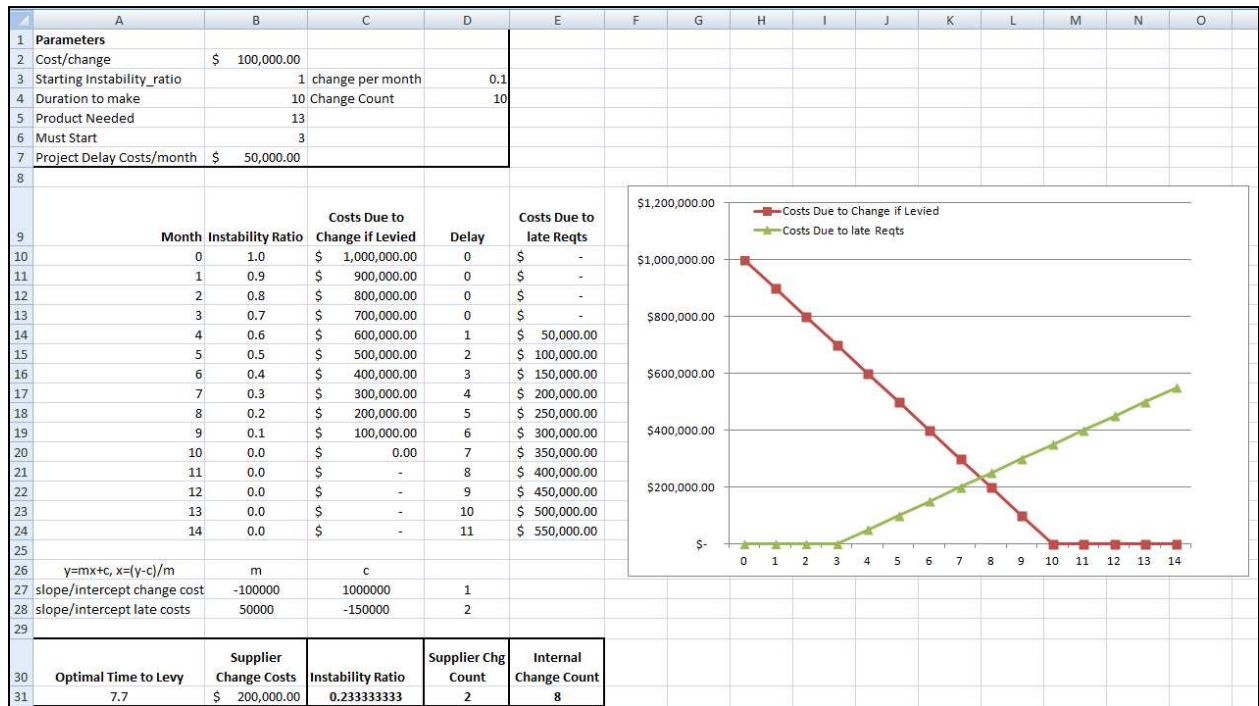
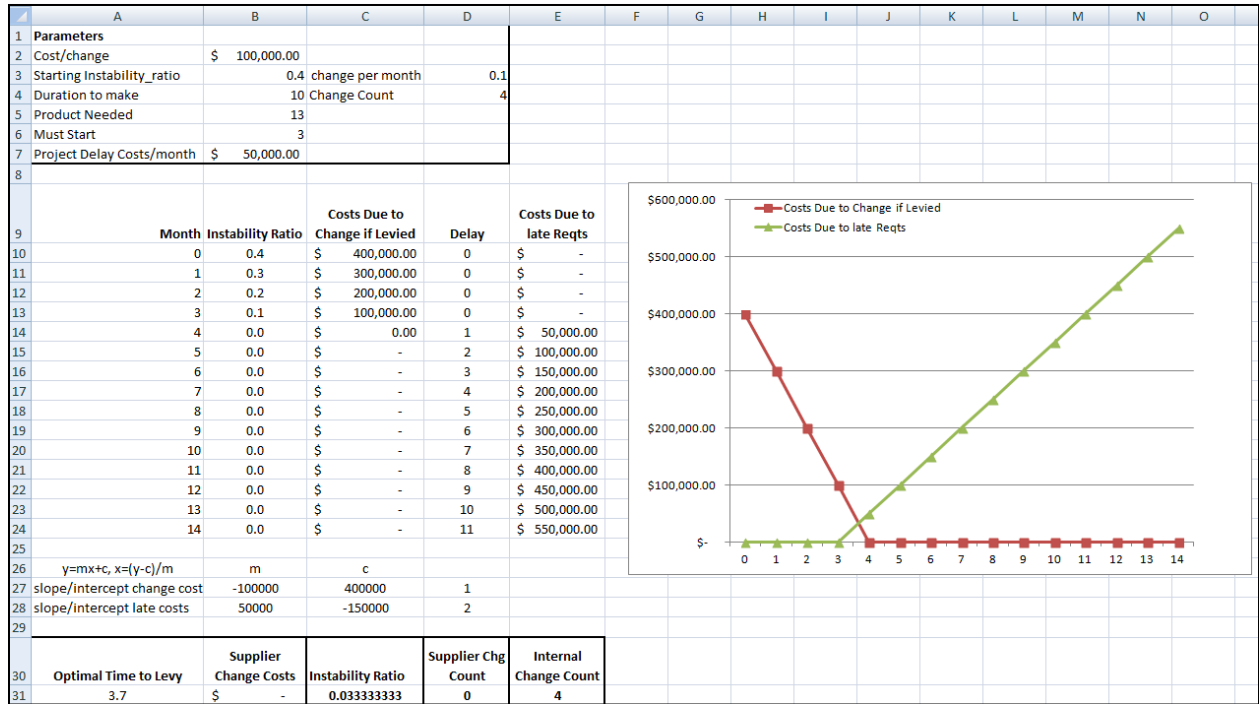


Figure 150. Impact of Instability Ratio at \$100K/change.

For task labor hours and instability ratio, sensitivity analysis utilizes a range of duration values to show impacts of applying a minimum and a maximum value for the simulation. The resulting data in Figure 146 shows a labor range, instead of a single data line, to account for the variation in process times and to establish how much this can impact the outcome. A assessment shows that both processes had similar impacts due to the level of requirement stability (instability ratio), where labor costs increased as a function of need for more changes comparably for both processes.

To assess impact of change size, the other values were held constant while the change size was varied from 0.05, 0.1, 0.2 and 0.4 per month, with the results shown in Appendix B2. Increasing the change size yields faster maturity steps, resulting in earlier optimal time to levy the requirements. Results of changing this parameter provides expected outcome based on amount of TBX resolved in a single change cycle.

For data inputs, the hours per task can be adjusted to reflect an experienced project team, or one that would take longer. The values assigned in Figure 145 are shown in further detail in Table 34 and Table 35 along with rationale for the durations chosen.

Table 34. Unstable Requirement Enforcement Task Values and Rationale.

Task Name	Duration	Rationale
Generate requirements for supplier	20h..40h	Observation associated with the requirements development process on past projects in flowing down requirements for a component (does not account for derivation analysis, requirements are assumed to be a direct flow-down); effort can take 2-3 days.
Capture TBX List	8h..16h	Observation associated with the requirements development process on past projects in identifying the requirements with unknown or unresolved parameters that need further work; effort can take 1-2 days.
Levy requirements	0	Costs are associated with contract efforts and durations are not impacted by requirement management process, normalizing to zero for simulation efforts.
Resolve Supplier TBX List	40h..120h	Observation associated with the requirements development process on past projects in analysis and investigation efforts to address unknowns in requirements, and discussing changes with the Supplier; using a one month change cycle for performing updates, this effort will take 1-3 weeks of that time period.

Task Name	Duration	Rationale
Update Supplier requirements	20h..50h	Observation associated with the requirements development process on past projects in efforts to update requirements; using a one month change cycle for performing updates, this effort will take 0.5 to 1 week of that time period.
Levy updated requirements	0	Costs are associated with contract efforts and durations are not impacted by requirement management process, normalizing to zero for simulation efforts.
Supplier Implements requirements	0	Costs are associated with supplier efforts and durations are not impacted by requirement management process, normalizing to zero for simulation efforts.

Table 35. Stability Before Enforcement Task Values and Rationale.

Task Name	Duration	Rationale
Generate requirements for supplier	20h..40h	Observation associated with the requirements development process on past projects in flowing down requirements for a component (does not account for derivation analysis, requirements are assumed to be a direct flow-down); effort can take 2-3 days.
Capture TBX List	8h..16h	Observation associated with the requirements development process on past projects in identifying the requirements with unknown or unresolved parameters that need further work; effort can take 1-2 days.
Assess TBX resolution plans	8h..12h	Observation associated with the requirements development process on past projects in identifying the resolution plan to address the unknown or unresolved parameters that need further work; effort can take 1-2 days.
Evaluate levying requirements compared to incur future changes	8h..12h	Estimate for assessment on whether the project should levy requirements to a supplier as soon as possible or wait until further maturity, requires research into cost per change and delay costs and implementation of the optimization model; expect effort to take about a day.
Resolve TBX List	40h..80h	Observation associated with the requirements development process on past projects in analysis and investigation efforts to address unknowns in requirements (internal project effort only); using a one month change cycle for performing updates, this effort will take 1-2 weeks of that time period (no interaction with supplier reduces the maximum duration)
Update requirements	20h..50h	Observation associated with the requirements development process on past projects in efforts to update requirements; effort will take 0.5 to 1 week
Levy requirements	0	Costs are associated with contract efforts and durations are not impacted by requirement management process, normalizing to zero for simulation efforts.

Task Name	Duration	Rationale
Resolve Supplier TBX List	40h..120h	Observation associated with the requirements development process on past projects in analysis and investigation efforts to address unknowns in requirements, and discussing changes with the Supplier; using a one month change cycle for performing updates, this effort will take 1-3 weeks of that time period.
Update Supplier requirements	20h..50h	Observation associated with the requirements development process on past projects in efforts to update requirements; using a one month change cycle for performing updates, this effort will take 0.5 to 1 week of that time period.
Levy updated requirements	0	Costs are associated with contract efforts and durations are not impacted by requirement management process, normalizing to zero for simulation efforts.
Supplier Implements requirements	0	Costs are associated with supplier efforts and durations are not impacted by requirement management process, normalizing to zero for simulation efforts.

The comparable tasks in processes 4a and 4b were given similar ranges of durations (such as tasks for generating requirements, capturing TBX list, etc.). As noted in the tables, the values used in the simulation reflect this author's experiences in performing the efforts and observations of similar tasks performed by systems engineering teams in multiple projects worked by this author.

The costs per change and delay costs can also be adjusted to reflect values associated with a project, or extrapolated from experience on similar projects; this is discussed in Sections 5.1.3 and 5.1.4. For the simulations basic values were supplied varying in order of magnitude for each of these inputs; the lower bounds were reflective of a smaller project where the upper numbers were reflective of a typical space system effort (it is realistic to see even larger cost per change and delay cost values on very large and complex space system subcontract efforts).

6.2.6 Requirements Management Process Model

Assessing the individual approaches provides insight into cost optimization opportunities with the four specific process areas. To look at the entire requirements management effort the individual processes models are integrated together into an overall SysML model of the requirements management process.

Top Level Model

Combining approaches into an overall process model representing Figure 123 (Requirements Management Process Model With Optimized Process Areas Highlighted) can be reflected in a SysML activity diagram as shown in Figure 151.

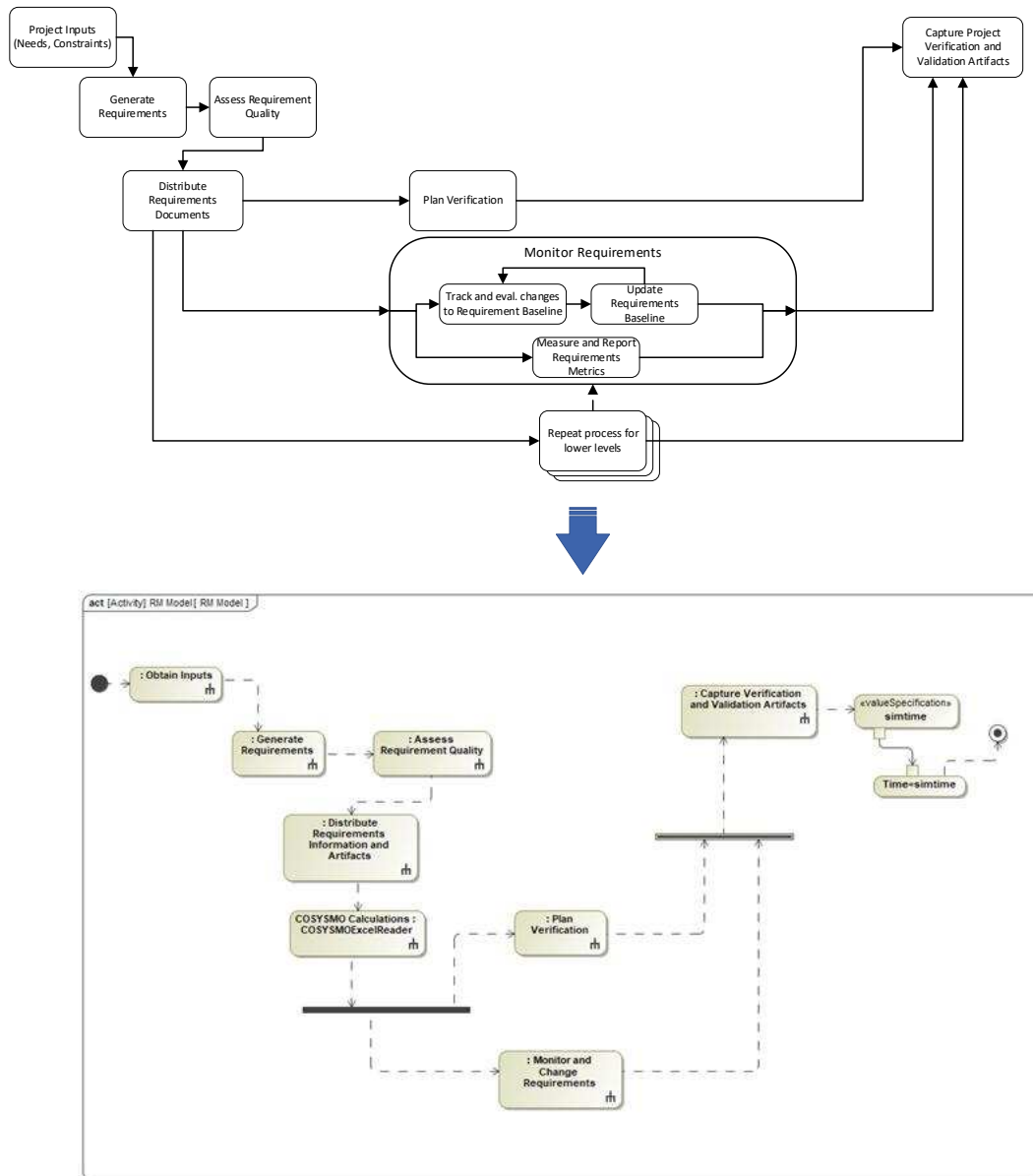


Figure 151. Overall Requirements Management Process Model Converted to a SysML Model.

Instead of developing two distinct models for the current state and optimized approaches, the single model provides a path for using the current state or optimized approach in the same model. The

following sections describe the approach of generating the model, the complete set of diagrams for this SysML model are shown in Appendix A.

The overall model contains activities which reflect the different process steps. For the activities that contain the process steps for processes 1-4, a decision point exists to use the current state or optimized processes, which invokes the SysML models for those process (reference Figure 152 for an example). The decision to use the optimized approach is a user entered value provided prior to the activity simulation.

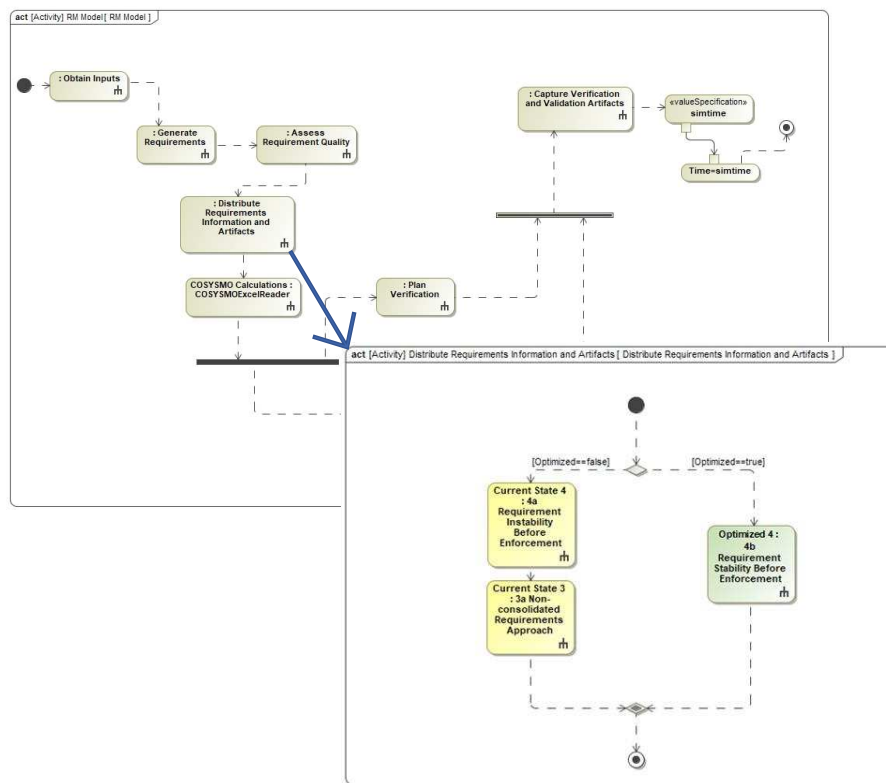


Figure 152. Activity Diagram Showing the Optimized Path Option.

Discrete Process Models

All of the requirements management process activity models are shown in Appendix A. This section provides information related to building those activities and performing data calculations with the different options.

As described previously in Section 6.2.5, calculations can be performed with parametric diagrams, generating equations using constraint blocks and providing input from the various value parameters. Another method of performing calculations within an activity diagram is through an opaque action, where the equations are applied at a certain point in the activity process. An example of this is shown in Figure 153. In the optimized processes, the consolidation effort will be shown to yield a 10% reduction in requirements and a reduction in overlaps (reducing the "difficult" requirement count to zero). The activity diagram contains an opaque action after process 3b to reduce the requirement quantity from the original amount, followed by activity steps that reduce the number of difficult requirements to zero and set the nominal requirement count to be equal to the requirement quantity. Having the calculations performed during the activity diagram simulation enables the calculations to be adjusted based upon which option is selected for the simulation (optimized or current state).

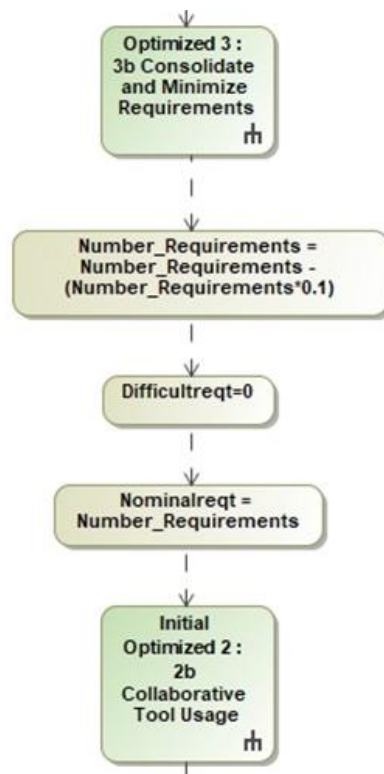


Figure 153. Activity Diagram Showing Opaque Action Calculations.

It should be noted that the values in the opaque actions can be changed to values reflecting an organization's capabilities, or as an area for future study.

For the processes associated with the "monitor and change of requirements" activity, requirement change cycles are a function of several factors and can occur after TBX resolution (reference Section 3.1.3). To represent a realistic simulation a value of five change cycles is used Figure 154; this is an adjustable parameter and assessed for sensitivity later in this section. The change activity also utilizes the option of performing processes 2a or 2b related to using a collaborative tool during the requirements change effort.

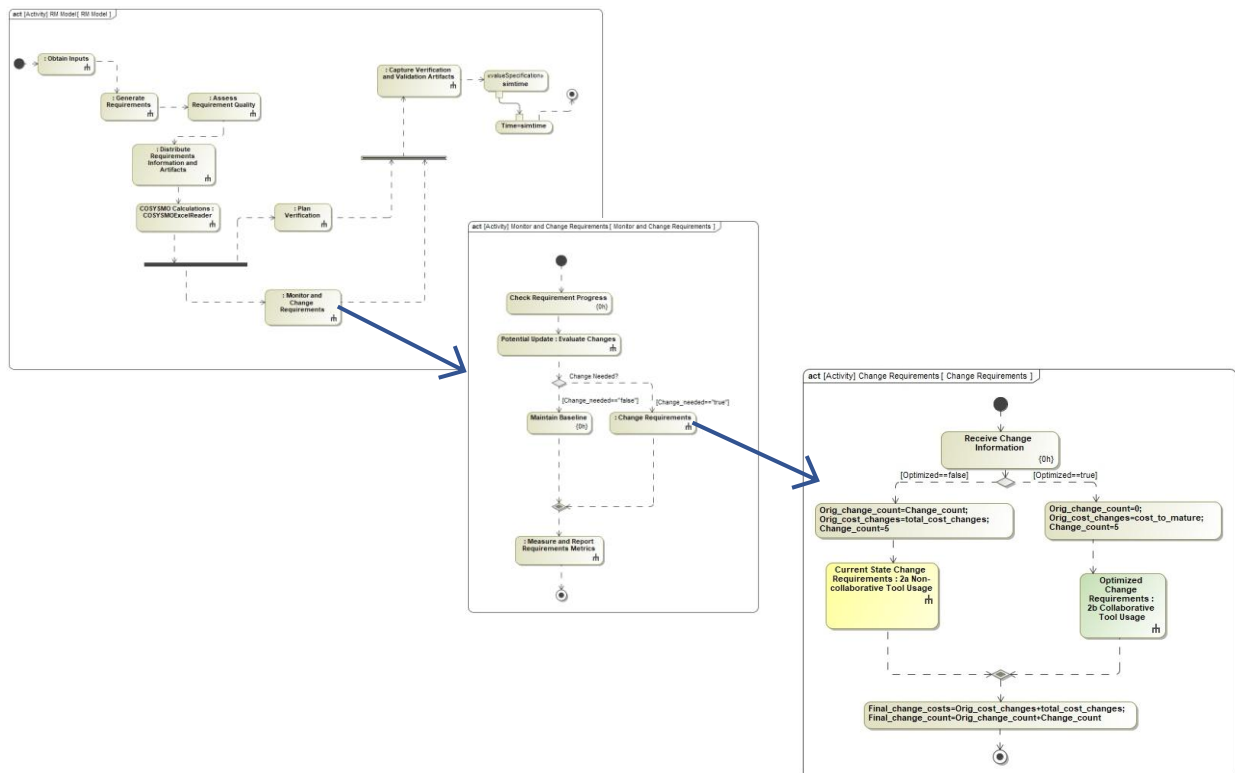


Figure 154. Monitor and Change Requirements Process Options.

To allow a direct comparison of the current state and optimized processes the other process activities are included but set to a duration of zero, removing their contribution to the total process time (reference Figure 155). In actuality these would be processes that consume systems engineering and project staff labor, this is addressed by usage of the COSYSMO calculated labor months to obtain expected costs associated with these type of processes based on requirement quantity and complexity. Normalizing the processes that are not being evaluated allows for a direct comparison of the current state

and optimized process flows, however the resultant durations from these only reflect a small fraction of the overall time of the requirements management process; this is a potential area for future study and evolution of the overall requirements management simulation model.

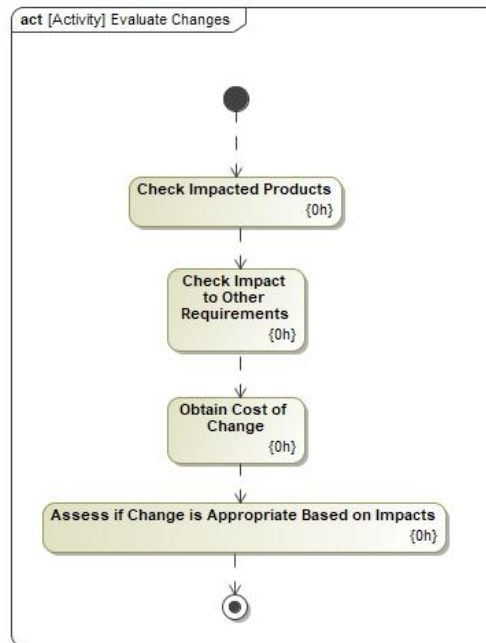


Figure 155. Sample Process Not Contributing to Optimization Assessment.

COSYSMO Integration

Combining the process steps of performing requirement management activities with the COSYSMO results in Section 5.5 appears to show overall benefit, however these simulations were done as separate data queries in MS Excel and Cameo. If the COSYSMO model is integrated into the Cameo activity diagrams it could be queried to show a value for project systems engineering labor along with the simulated time durations of performing activities in the SysML diagrams, providing a more comprehensive executable simulation and capture of data directly within the Cameo model.

In a previously published paper, Dalton looked at the logistics of integrating SysML with COSYSMO (Dalton, 2020) by using a graphical user interface (GUI) plug-in for Cameo Systems Modeler. A more basic integration method looked at for this dissertation uses the Cameo Systems Modeler ability to pass data to and from an attached MS Excel file (Figure 156), similar to the integration

of models with the "Requirement Stability and Enforcement Optimization" process effort. With this approach, an activity diagram is created to establish input values of requirement quantity for easy, nominal and hard values, and to extract the value for labor duration.

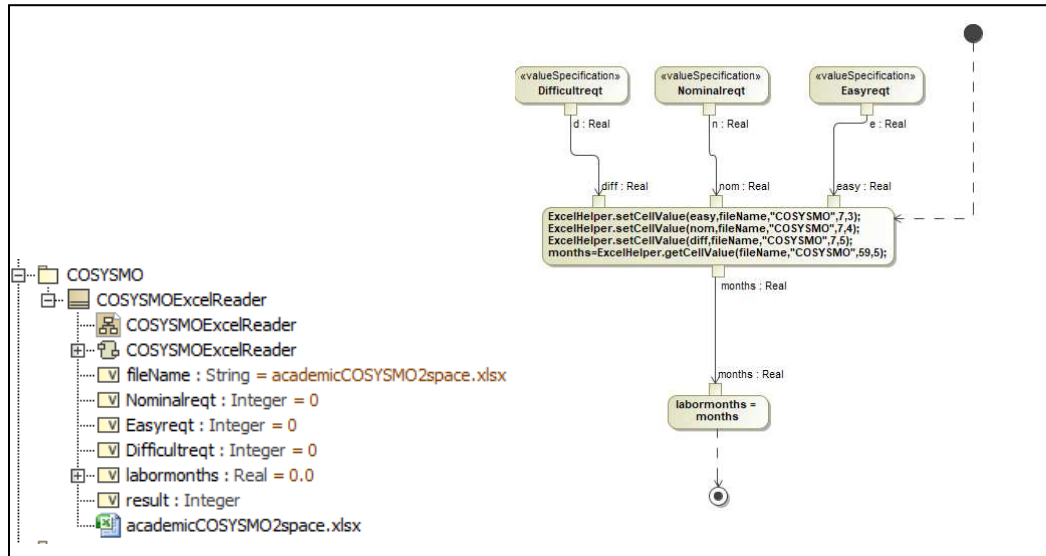


Figure 156. COSYSMO Integrated in an Activity Diagram Simulation.

Figure 157 shows the result of implementing the activity diagram in SysML; for a requirement quantity of zero the SysML simulation yields the same outcome for labor as the manually entered results shown in Figure 136.

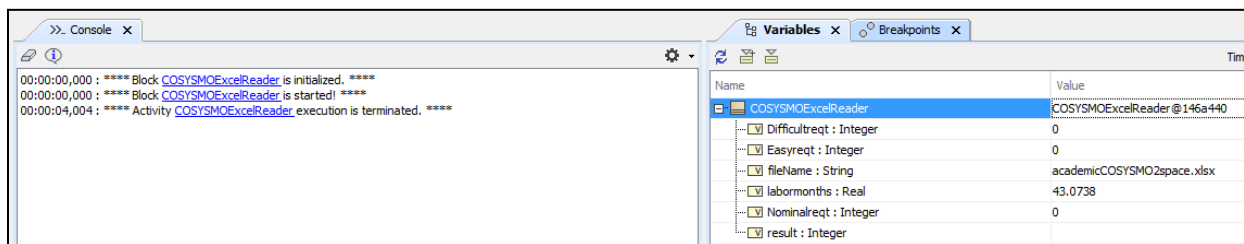


Figure 157. Simulation Results of COSYSMO Integrated in an Activity Diagram Simulation.

The COSYSMO Excel reader activity shown in Figure 156 is incorporated into the overall requirements management simulation model shown in Figure 151, allowing the data on systems engineering labor months to be captured in the simulation along with the time.

6.2.7 Model Development Concluding Remarks

This section modeled four process recommendations from Chapter 5 to show how the proposed approach would yield cost savings compared to the current approach. The variations of inputs and associated sensitivities were investigated, and the assumptions of labor durations and direct cost values were provided with rationale. Each of these values are also adjustable parameters in the model, either through simulating a range of occurrences (task durations) or providing direct input to the model (direct costs).

The overall requirements management executable model combines the current and proposed optimized processes together to generate comparative costs between the different approaches. Chapter 7 presents the execution of this model using the parameters from the space projects from Chapter 4, showing examples of the overall simulation as well as development of data to see if the space project requirements management processes can be optimized for cost using the proposed approaches.

CHAPTER 7: REQUIREMENTS MANAGEMENT MODEL OPTIMIZATION CASE STUDIES

Assessing the prior space projects, the requirements management model is simulated to compare how requirements management would change if these projects were in work today. A comparison of outcomes is then shown and discussed.

7.1 Application of Requirements Management Model

This section provides examples application the requirement management model (current state and optimized) to space system projects using the information obtained from the space projects in Chapter 4. Known information will be supplied where available, and assumed inputs will be used in a way that reflects reasonable values and to normalize inputs (such as cost per change and delay penalty cost).

7.1.1 Inputs For Simulation

Establishing information from the case studies, the parameters in Table 36 were obtained from Chapter 4 to reflect the various space project examples presented.

Table 36. Project Inputs for Requirements Management Model.

Project	# of System Requirements	Number of Documents	Number of Suppliers	Number of Internal Design Teams
MAVEN	660 (0 of these are rated difficult)	6	6	0
MSL	511 (309 of these are rated difficult)	1	12	22
GOES-R	~1300 (50 of these are rated difficult)	11	9	0
Constellation	~8600 (1220 of these are rated difficult)	51	4	1
Artemis HLS	~4551 (460 of these are rated difficult)	46	1	0

The different space projects have unknowns regarding their initial TBX count, or if they had a large value of cost per change with their suppliers. To assess the different situations multiple case study runs are used to vary these parameters. The values of Table 37 show the parameters applied, which

represent a change in TBX, a change in cost per change, and constant, representative values instability ratio change, product duration and delay costs.

Table 37. Case Study Inputs for Requirements Management Model.

Parameter	Case Study 1	Case Study 2	Case Study 3
TBX Count (% of requirements)	25%	50%	25%
Duration to Make	10 months	10 months	10 months
Product Needed	13 months	13 months	13 months
Delay Costs per Month	\$50,000	\$50,000	\$50,000
Costs per Change	\$75,000	\$75,000	\$150,000
Instability ratio change per month	0.1	0.1	0.1

7.1.2 Case Study Simulation Results

For the overall requirements management simulation, the three case studies were executed in the model using the space project parameters from Table 36. The simulation run time was set to "average", avoiding process time extremes in order to assess other parameters such as requirement and product count. The simulations were executed for each case using the current state and optimized options in the model; Figure 158 provides screen captures of the simulation execution during case study 1.

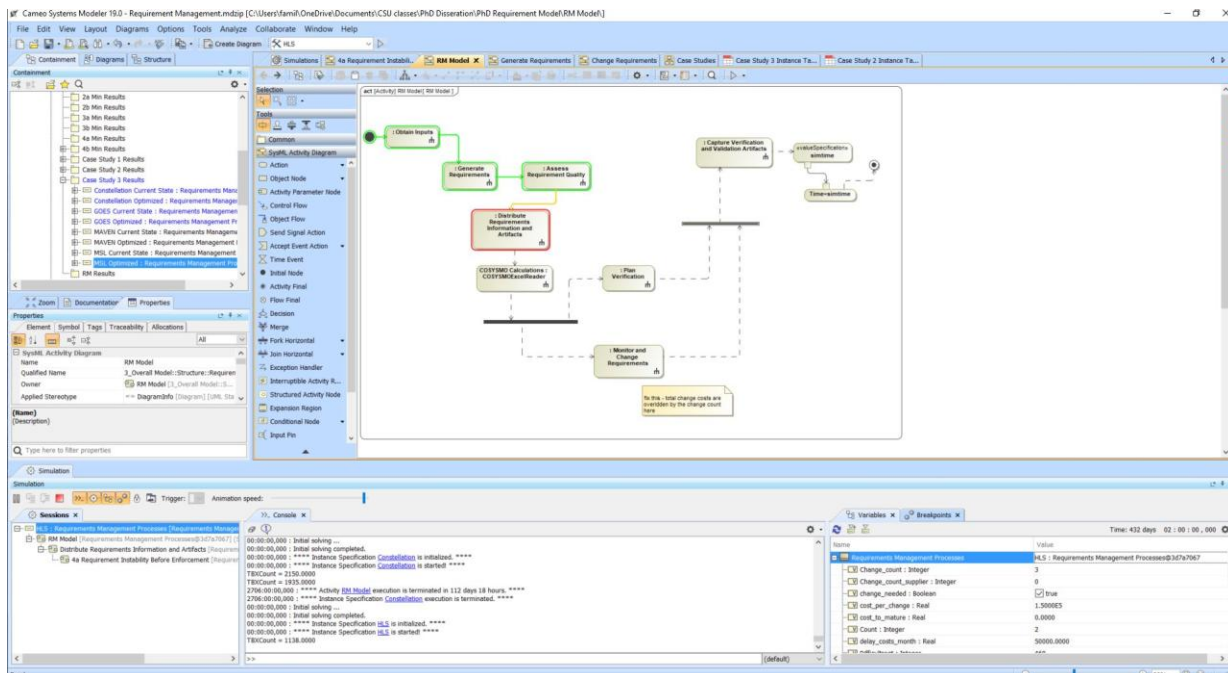


Figure 158. Example of the Requirements Management Process Activity Diagram Simulation.

During the simulation a timeline graph of the activity process durations was displayed (example shown in Figure 159) providing process durations for each of the activities; the current state simulation is shown in the top graph while the optimized simulation is shown below.

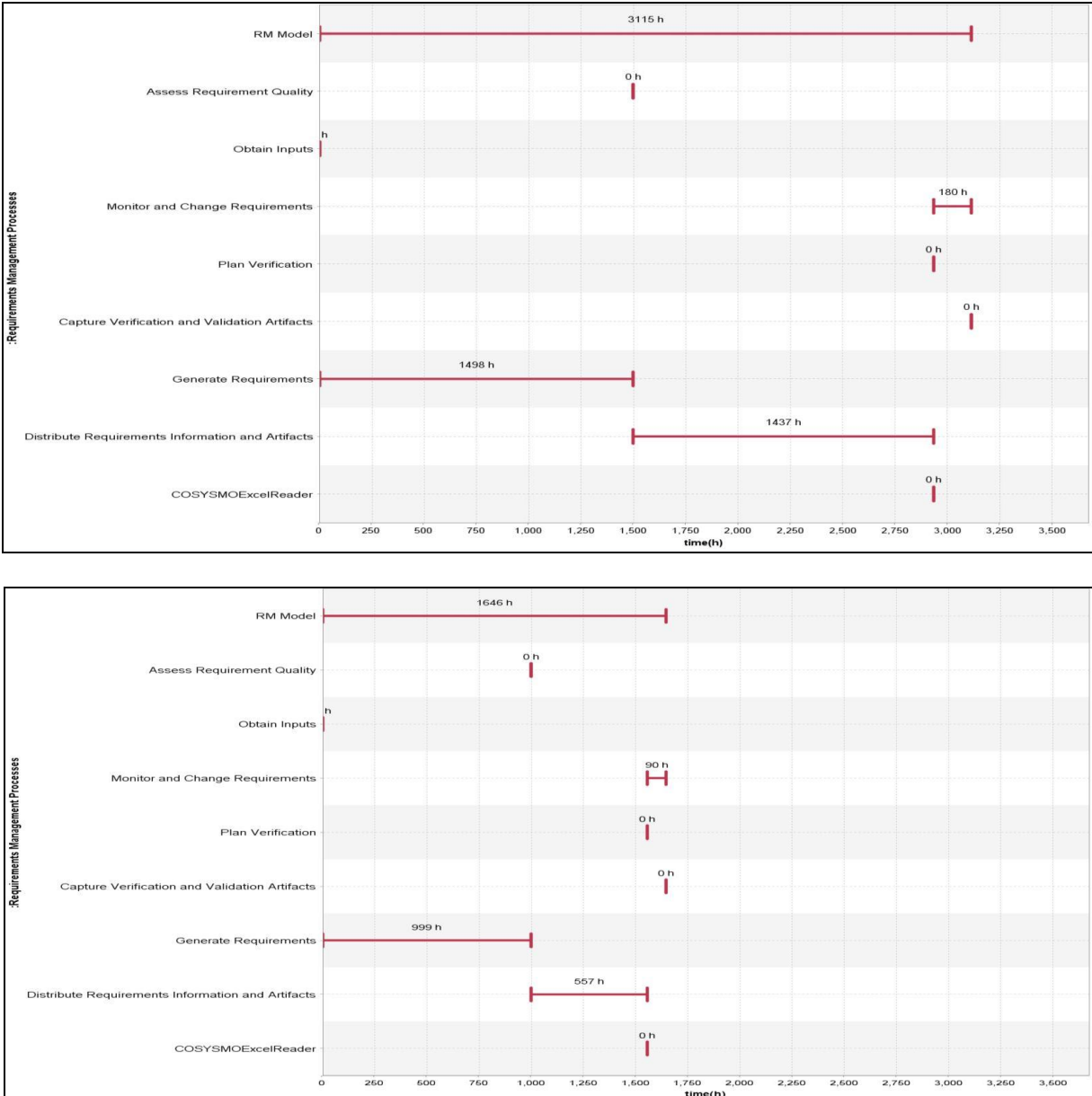


Figure 159. MAVEN Case Study 1 Simulation Timeline Results Example.

Each simulation resulted in outputs that included a calculated output of the requirements management activity duration, the systems engineering labor from COSYSMO, and the cost of changes

incurred through various change cycles. The results were captured in a Cameo instance table, highlighted in Figure 160.

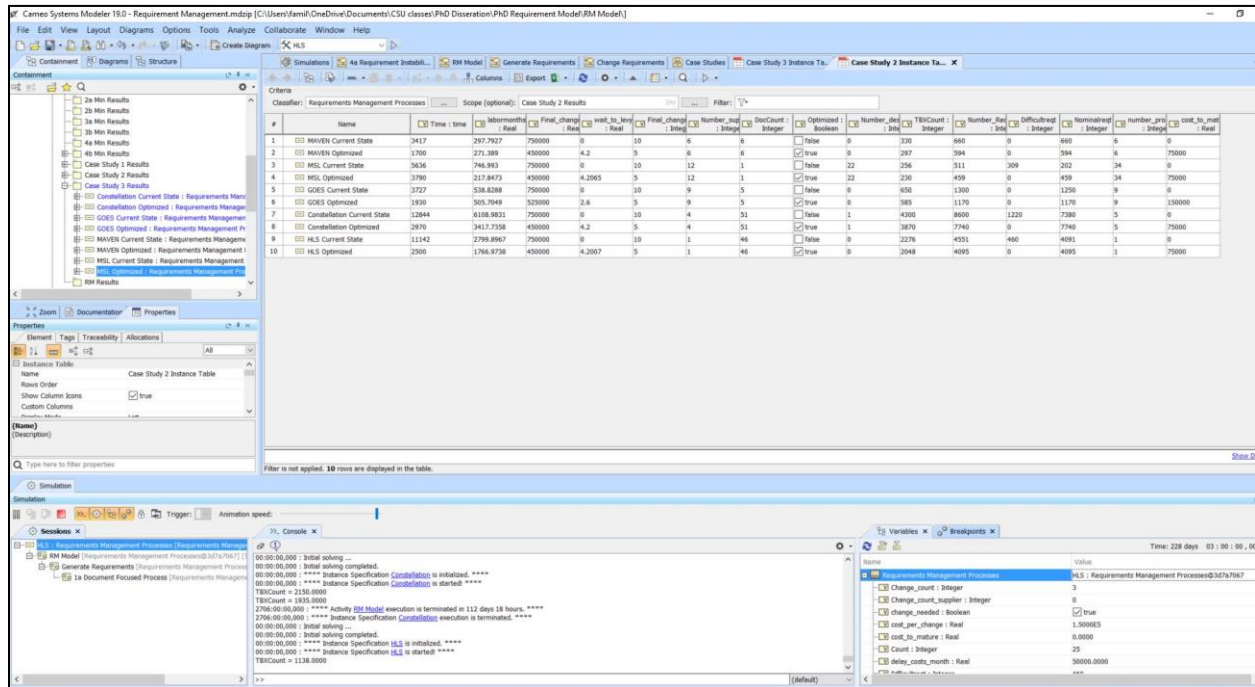


Figure 160. Example of the Requirements Management Process Simulation Data Table.

The data table from the model was extracted to MS Excel to allow an analysis of the results to calculate how much improvement the optimized option provided; an example data table is shown in Figure 161, the complete set of case study data tables are provided in Appendix B1.

Name	Current State Time (Hours)	Optimized Time (Hours)	Improvement	Current State SE Labor (Months)	Optimized SE Labor (Months)	Improvement	Current State Change Count	Optimized Change Count	Current State Change Costs	Optimized Change Costs	Optimized Wait Time to Levy (Months)
MAVEN	3115	1646	47%	297.8	271.4	9%	8	5	\$ 600,000	\$ 450,000	4.2
MSL	5334	3736	30%	747.0	217.8	71%	8	5	\$ 600,000	\$ 450,000	4.2
GOES	4715	1846	61%	643.1	505.7	21%	8	5	\$ 600,000	\$ 450,000	1.1
Constellation	12542	2916	77%	6109.0	3417.7	44%	8	5	\$ 600,000	\$ 450,000	4.2
HLS	10840	2446	77%	2799.9	1767.0	37%	8	5	\$ 600,000	\$ 450,000	4.2

Figure 161. Case Study 1 Simulation Results Example.

The resultant percent improvement in requirements management durations and systems engineering labor are shown for each project in Figure 162.

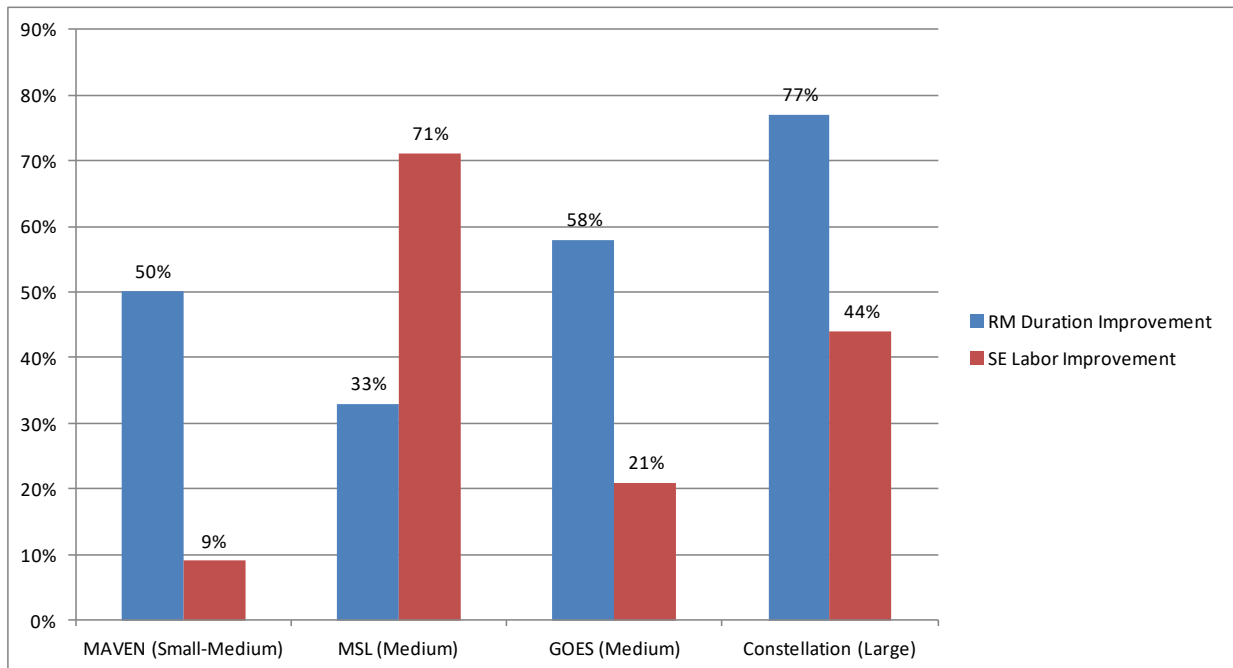


Figure 162. Space Project Optimized Requirements Management Process Benefits.

The various projects had different improvements based upon their initial parameters; some saw benefits in the requirements management labor durations due to number of requirements documents driving the requirements development effort, while others saw a savings in the overall systems engineering labor due to the improvement of requirements quantity and quality.

The results of the labor savings in the data tables are converted to dollar saving using the inputs of \$100/hr (reference Section 5.1) and 160 hr/month, these are added to the direct cost savings from the change costs to show total cost savings for each project using the optimized processes (Table 38).

Table 38. Case Study Project Cost Savings Results.

Name	Case Study 1 Total Improvement	Case Study 2 Total Improvement	Case Study 3 Total Improvement
MAVEN	\$ 719,359	\$ 894,159	\$ 1,040,359
MSL	\$ 8,776,889	\$ 8,950,931	\$ 9,097,889
GOES	\$ 2,632,300	\$ 2,714,021	\$ 2,785,300
Constellation	\$ 44,173,129	\$ 44,347,356	\$ 44,494,129
HLS	\$ 17,515,748	\$ 17,690,966	\$ 17,836,748

The results from Case Study 2 (instability ratio of 0.5 and moderately low cost per change) reflects a rough average of all of the three case study results, and is represented graphically for each space project in Figure 163.

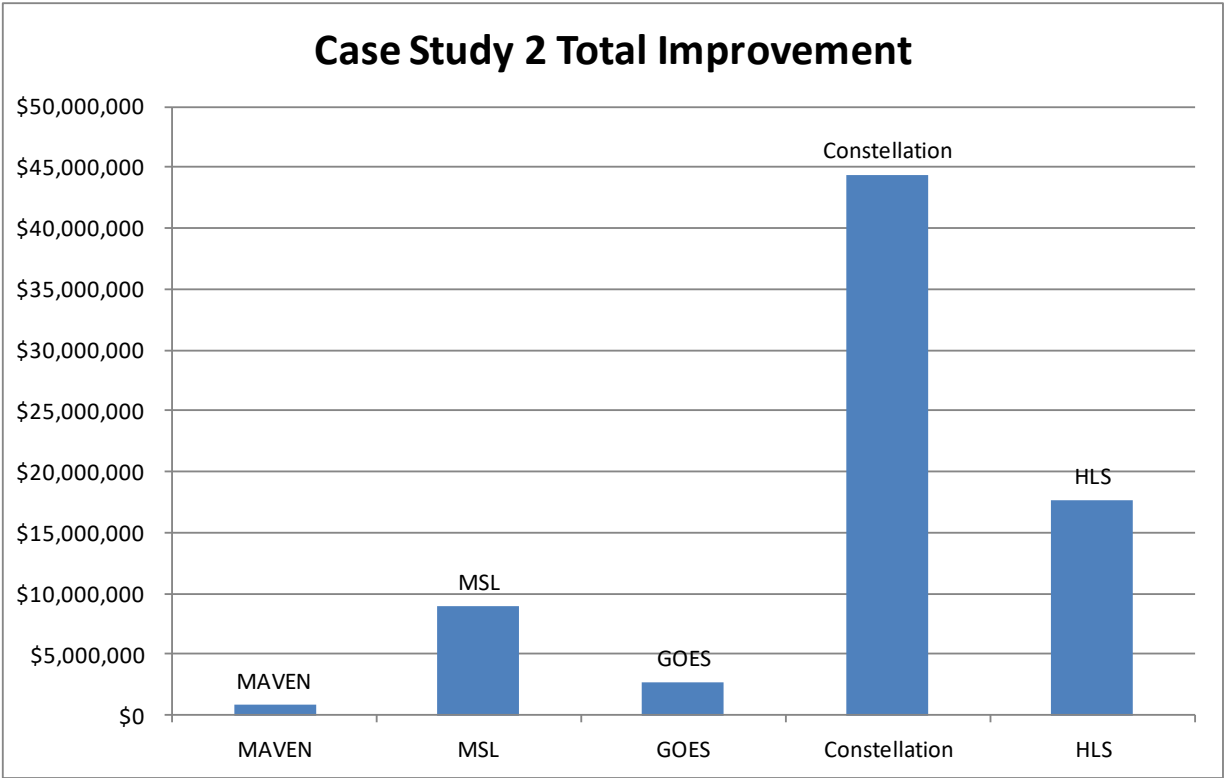


Figure 163. Project Cost Savings using Optimized Processes (Case Study 2).

A parameter from the simulation assessed for further sensitivity is *change count* (the number of requirement changes levied), which is used in the "monitor and change process" during the requirements management activities. The assessment evaluated the usage of the collaborative requirements management tool versus a less useful tool for a varying set of change cycles during the requirement change efforts later in a project. While cost per change did increase with each change cycle, the data in Table 39 shows that the total duration for performing the entire requirements management process did not vary significantly with change count value ranging from zero and five changes. Implementation of requirement changes later in a project can certainly incur significant costs on the project, as shown in (Stecklein, 2004), and usage of an optimized approach can be shown to improve the duration of the requirements management efforts compared to the current state practice; however, the duration of the

effort in either the current state or the optimized approach does not appear to be sensitive to the amount of change cycles that occur later in the project.

Table 39. Impact of Change Count in Post-Requirement Development Phase on Duration Time.

Number of post Development Changes	Current State Duration (hr)	Optimized Duration (hr)
0	2935	1346
1	2971	1364
2	3007	1382
3	3043	1400
4	3079	1418
5	3115	1436

7.2 Discussion of Case Study Results

Comparing the benefits found among the different projects, it is observed that the amount of benefit per project aligns with the project outcomes noted in Chapter 4 (and summarized in Table 23, shown previously). The data from Table 23 has been combined with the estimated cost savings shown in Table 38, reflecting the improvement of using an optimized approach along with the outcomes of the different projects. Table 40 presents each project along with a description of its complexity level, the outcome realized for project success, the prior assessment on need to optimize (1 was low need, 3 was high), and the results of the predicted cost savings through the optimized process simulation. The results shown in Table 40 also align with benefits of optimization based on project complexity shown earlier in Figure 96.

Table 40. Summary of Space Project Calculated Labor Costs from COSYSMO.

Project	Project Notes	Need to Optimize Requirements Management (1-3)	Case Study 2 Total Improvement
MAVEN (small-medium complexity)	Successfully executed project objectives	1	\$ 894,159
MSL (medium complexity)	Project was moderately successful, further optimization could have ensured it met cost and schedule objectives.	2	\$ 8,950,931

Project	Project Notes	Need to Optimize Requirements Management (1-3)	Case Study 2 Total Improvement
GOES-R (medium complexity)	Project was moderately successful, further optimization could have ensured it met cost and schedule objectives.	2	\$ 2,714,021
Constellation (large complexity)	Project Cancelled in Design Phase	3	\$ 44,347,356
Artemis HLS (large complexity)	Project Still in Development	Cannot rate	\$ 17,690,966

The results in Table 40 show that MSL yielded significant savings when the optimized processes were applied. Reviewing some of the feedback on MSL, it had substantial cost overruns of around \$400 million; the initial observations from Section 4.3 noted that an improvement in requirement quality would yield a reduction in labor months from 747 to 238 based on COSYSMO calculations. What was not evident in the earlier assessment was the potential for 30% improvement on the requirements management process duration calculated from the simulation results, enabling the production of *better* requirements *sooner*, along with a reduction in costs due to change cycles.

Another project of interest is Constellation, with a prediction of savings associated with requirements management durations greater than 70% reduction in labor time. Table 41 provides a breakout of calculated duration for the Constellation requirements management effort for the current state and optimized processes. In looking at the Constellation activities, most of the time was spent in the requirements generation activity; this is the impact of a large number of requirements documents (51) driving a significant amount of time spent in requirements development using a document centric requirements management approach (Process 1a); this labor time is significantly reduced when using a the data centric requirements management effort (Process 1b). This trend of labor duration for both document centric and data centric approaches as a function of document count is consistent with the results presented in the Section 6.2.2.

Table 41. Summary of Constellation Task Durations (Case Study 1).

Task Name	Current State Duration (hr)	Optimized Duration (hr)
Overall RM Model	12542	2916
Generate Requirements (51 specifications)	11173	2269
Distribute Requirements Information and Artifacts	1189	557
Monitor and Change Requirements	180	90

It can be observed that the Constellation savings could also be realized by the HLS project, which is still in the early stages of development. HLS has an equivalent amount of requirement documents as Constellation, and appears to have an opportunity to benefit from a more data centric approach to requirements generation and management.

Based on the simulations from Chapter 6 and the case studies shown in this chapter, it appears that the optimized requirements management approach has the potential to improve a project's schedule and cost. A legitimate question for any project is whether to invest in the purchase of new tools, process updates and training of personnel, which can be time consuming and expensive (reference Section 5.1) . Looking at the analysis in this section, it appears that for complex projects the savings in time and cost could warrant upfront investment to implement the new approaches.

8.1 Approach to Requirements Management Cost Optimization

When projects choose to spend too little on the requirements management effort, the results tend towards cost overruns and deficient products (Gruhl, 1992). When they spend too much they incur schedule delays and expenses as a result of the labor and direct costs expended. Investigation was done to discover an optimal approach towards management of requirements for a project, which includes development of an overall process model and specific process steps within it that allow a project to achieve a good balance of costs and results.

This dissertation proposed four distinct process improvements that projects could apply to help them realize cost optimization in requirements management, which includes:

1. Implement a data focused requirements management approach
2. Utilize a management tool that supports electronic collaboration during requirement development and change activities throughout the project life cycle
3. Minimize and consolidate the requirements for the system of interest
4. Coordinate the timing between developing requirements and levying them officially

After utilizing an executable system model created in SysML to simulate these processes, each proposed update showed some benefit associated with their usage individually. These processes were then combined into an overall requirements management SysML model using inputs from five actual NASA space projects. The results of the simulation showed that projects with medium to high complexity (many overlapping requirements, many requirement documents, and several products and suppliers) could benefit from an investment in updating their requirements management approaches to realize cost and schedule savings. These benefits were observed to outweigh the anticipated costs of investing in the process updates, leading to an overall cost optimized requirements management effort.

8.2 Recommendations for a Requirements Management Approach

When assessing the complexity associated with space system development and the need to realize innovative systems sooner, there exists a strong need to improve project management processes to yield results quicker. Considering the scale of space system development there is potential for companies to price themselves out of a competitive market with unaffordable products; there is a strong need to improve project management processes to yield minimized development cost while still meeting schedule, technical and customer acceptance. The requirements management process model provided in this dissertation addresses these parameters and provides options for companies to implement when developing space systems. Table 42 presents a checklist for an organization to evaluate their options in conducting requirements management, containing recommendations based on the research done within this dissertation. While this table was specifically generated for application in the development of space systems, the recommendations are applicable to the development of any complex system.

With some projects the need to "win the work" at low costs may preclude them from investing money and time to optimize their requirements management process. Hopefully it has been shown that some approaches may actually realize cost savings and should be considered an opportunity for projects to bid on work at a lower cost and faster schedule once the upfront investment in tools and process changes have been applied. Organizations may also realize benefits in implementing a phased approach in working some of the recommended methods while still looking into future application of processes for the others.

Table 42. Recommended Requirements Management Approach to Achieve Optimization

#	Process Recommendation	Life-cycle Phase	Considerations
1	<p>Select whether a "document centric" or "data centric" requirements management approach will be applied.</p> <p><i>For complex projects with significant amount of specifications and standards, the selection of a data centric approach is advised to realize cost and schedule savings compared to the investment of establishing this process for a project.</i></p>	Project Proposal/ Pre-Award	Project complexity (technical performance and product structure), document quantity, existing processes at the organization, required effort to develop updated processes.
2	<p>Select a Requirements Management Tool for the project.</p> <p><i>For new development projects with a significant number of requirements, the selection of a user-friendly and collaborative requirements management tool is advised to realize cost savings compared to the cost of a new tool and associated training required.</i></p>	Project Proposal/ Pre-Award	Amount of requirements, expectation on maturity and change evolution based on product being developed, existing tools, costs to purchase any new tools, associated training costs and learning curve schedule impacts.
3	<p>Ensure requirement quantity and quality is addressed during the requirements development effort.</p> <p><i>The amount of time spent ensuring the requirement set is minimized and consolidated for the project may vary, but for complex projects with a significant number of requirements it is advised that the time invested early to improve the requirement set (reduce overlaps, minimize the amount of requirements to a set of singular requirements that are necessary) will yield benefits in labor costs and schedule later in the project life cycle.</i></p>	Project Start - Preliminary Design Review	Amount of initial set of requirements, project complexity in product structure, amount of outside organizations receiving the requirements; usage of standards such as INCOSE Guide for Writing Requirements provides the methods to achieve requirement quality.
4	<p>Select the timing to levy requirements on any subcontractors that will be developing products.</p> <p><i>Based on requirements maturity, evaluate whether to 1) wait on establishing the contracts, 2) bring on the subcontractors with a contract to help advance the preliminary requirements while working early development activities, or 3) establish the contract with an official set of requirements.</i></p> <p><i>For highly unstable requirements, it is advised to either wait to establish the subcontract or to use an approach to have the subcontractor support the requirement development activities; this approach could realize cost savings associated with future change cycles and rework by the supplier.</i></p>	Project Start - Preliminary Design Review	Stability of the requirements, heritage of the subcontractor, anticipated cost of future change cycles, anticipated costs associated with any schedule delays of starting development efforts.

8.3 Suggestions for Additional Research

During the development of this dissertation there were several items observed as viable future areas of research. Some considerations for areas of further work include exploring additional cases in the executable requirements management model to assess other parameters, including:

- Refinement of the project durations with more discrete values obtained from prior programs.
- Assessment of impacts associated with different types of supplier contracts (fixed price compared with cost plus).
- More defined inputs for costs associated with incorporation of changing processes for a project or organization (addressing the benefits of change compared to the cost).
- Further assessment of the other processes within the SysML requirement management model that were normalized to a zero duration during this study.
- Explore how the recommended requirements management process updates could benefit the system verification and validation costs and schedule.

Additionally, opportunity for further research exists with the data from the case studies in Chapter 4, such as assessing a relationship between the number of requirements as a predictor of cost overrun when using traditional requirements management approaches.

8.4 Conclusions

When done poorly, requirements management can have significant impact against project success. However, when done well and efficiently, requirements management can appear effortless, moving focus towards the more technical aspects of product development, enabling innovation and product excellence. Appearing effortless, however, is a benefit of using an upfront assessment to ensure the right amount of time and rigor is applied to how the project will execute their processes. This dissertation provided multiple examples of how the processes of requirements management could be improved for a project during the development of space systems, enabling an organization to establish approaches to improve requirements quality while also realizing cost benefits.

REFERENCES

- Atkinson, N. (2008, October 10). *Mars Science Laboratory: Still Alive, For Now*. Retrieved August 8, 2020, from Universe Today: <https://www.universetoday.com/19429/mars-science-laboratory-still-alive-for-now/>
- Baker, J. (2020). *Sparx Systems - Extending the Requirement Type*. Retrieved from <https://www.eaglobalsummit.com/sessions/extending-the-requirement-type/>
- Bernard, Y. (2011). Requirements Management within a Full Model-Based Engineering Approach. *Systems Engineering (DOI 10.1002/sys.20198)* .
- Bernard, Y. (2011). Requirements Management within a Full Model-Based Engineering Approach. *Systems Engineering, Wiley Periodicals* .
- Boeing. (2017). *Developing Airplane Systems Faster and with Higher Quality*. Retrieved from Boeing Corporation: <https://www.boeing.com/features/innovation-quarterly/may2017/feature-technical-model-based-engineering.page>
- Bold Business. (2018). *SpaceX Named the Number One Disruptor*. Retrieved from Bold Business: <https://www.boldbusiness.com/transportation/spacex-disrupts-rocket-industry/>
- Borky, M. (2019). *SysML Cameo Tutorial*.
- Carroll, E., & Malins, R. (2016). *Systematic Literature Review: How is Model-based Systems Engineering Justified?* Sandia National Laboratories.
- Connolly, J. (2006). *Constellation Program Overview*. NASA.
- Dalton, J. (2020). Integrating the Constructive Systems Engineering Cost Model with the System Modeling Language. *18th Annual Conference on Systems Engineering Research (CSER)*. SERC.
- Delligatti, L. (2014). *SysML Distilled*. Pearson Education.
- Dick, J., Hull, E., & Jackson, K. (2017). *Requirements Engineering* (Fourth Edition ed.). Springer; Figures reprinted with permission.
- Engineering.com. (2018). *Engineering.com*. Retrieved 10 22, 2020, from Research Report: Design Teams: Requirements Management and Product Complexity: <https://www.engineering.com/ResourceMain?resid=873>
- eoPortal. (2020). *GOES-R 3rd Generation Series*. Retrieved August 18, 2020, from eoPortal Directory: <https://directory.eoportal.org/web/eoportal/satellite-missions/g/goes-r>

- FAA. (2009). *Requirements Engineering Management Handbook*.
- Forsberg, K., Mooz, H., & Cotterman, H. (2005). *Visualizing Project Management: Models and Frameworks for Mastering Complex Systems*. New Jersey: John Wiley & Sons, Inc.; Figures reprinted with permission.
- GAO. (2009). *Constellation Program Cost and Schedule Will Remain Uncertain Until a Sound Business Case is Established*. Report to the Chairman, Committee on Science and Technology, House of Representatives. .
- GAO. (2014). *Geostationary Weather Satellites - Launch Date Nears, but Remaining Schedule Risks Need to be Addressed*. Office of Inspector General.
- GAO. (2011). *IG-11-019, NASA's Management of the MARS Science Laboratory Project*. Office of Inspector General.
- GAO. (2013). *IG-13-009, Mars Atmosphere and Volatile Evolution (MAVEN) Project*. Office of Inspector General.
- Gomez Sotelo, K. (2019). *Quality Assurance Methodology for System Requirement Definition*. INSA de Toulouse.
- Gruhl, W. (1992). *Lessons Learned, Cost/Schedule Assessment*. NASA.
- Grush, L. (2019, May 17). *NASA administrator on new Moon plan: 'We're doing this in a way that's never been done before'*. Retrieved August 30, 2020, from The Verge: <https://www.theverge.com/2019/5/17/18627839/nasa-administrator-jim-bridenstine-artemis-moon-program-budget-amendment>
- Gulke, T., Rumpe, B., Jasen, M., & Axmann, J. (2012). High-Level Requirements Management and Complexity Costs in Automotive Development Projects: A Problem Statement. *Requirements Engineering: Foundation for Software Quality 18th International Working Conference*.
- Guo, D. Z. (2018). Co-Evolution of Complex Aeronautical Systems and Complex Systems Engineering, Keynote Address. *INCOSE IS2018 Proceedings* (pp. <https://www.youtube.com/watch?v=ozDFtx-Cb-s>). Washington DC: INCOSE.
- Helle, K., Engen, S., & Falk, K. (2020). Towards Systemic Handling of Requirements in the Oil and Gas Industry – a Case Study. *INCOSE IS2020*. INCOSE.
- Hokanson, C. (2020, 07 29). Business Architect, Seilevel. (T. Katz, Interviewer)
- Hokanson, C. (2016). *Community Blog for Business Analysts - What is Good Enough when it comes to a Requirements Management Tool?* Retrieved from Modern Analyst.com.
- Honour, E. (2013). *Systems Engineering Return on Investment*. University of South Australia.

Hood, C., Wiedemann, S., Fichtinger, S., & Pautz, U. (2008). *Requirements Management – The Interface Between Requirements Development and All Other Systems Engineering Processes*. Germany: Springer.

Hooks, I., & Farris, K. (2001). *Customer-Centered Products, Creating Successful Products Through Smart Requirements Management*. AMACOM.

Howard, J., & Anderson, P. (2002). The Safety Risk of Requirements Incompleteness. In S. Engineering (Ed.), *ISSC*.

IBM. (2010). *DOORS Training*. Retrieved from YouTube:
<https://www.youtube.com/watch?v=Reff4ELfwrM>

IBM. (2014). *Getting started with IBM Rational DOORS Next Generation*. Retrieved August 23, 2020, from IBM Developer: <https://www.ibm.com/developerworks/rational/library/rational-doors-next-generation-getting-started/tutorial/index.html>

IBM. (2020). *IBM DOORS*. Retrieved from <https://www.ibm.com/products/requirements-management/details>

IBM. (2020). *IBM DOORS Next*. Retrieved from <https://www.ibm.com/products/ibm-engineering-requirements-management-doors-next/faq>

IBM. (2020). *IBM DOORS Next Previews*. Retrieved from IBM:
https://jazz.net/previews/?&_ga=2.118951489.958070923.1594074492-548003569.1593641238#dng

IBM. (2013). *IBM Rational DOORS getting started*. Retrieved August 23, 2020, from IBM Developer: <https://www.ibm.com/developerworks/rational/library/getting-started-ibm-rational-doors/index.html>

IEEE. (2015). IEEE 15288. *Systems and Software Engineering - System Life Cycle Processes* .

IEEE. (2018). IEEE 29148. *Systems and Software Engineering Life Cycle Processes - Requirements Engineering* . IEEE.

INCOSE AFIS. (2016). *Requirements and Architecture within Modelling Context*.

INCOSE. (2019). *INCOSE Guide for Writing Requirements*. (L. Wheatcraft, & M. Ryan, Eds.) INCOSE.

INCOSE. (2021 draft). *INCOSE Needs and Requirements Lifecycle Manual*. INCOSE.

INCOSE. (2015). *INCOSE Systems Engineering Handbook (INCOSE-TP-2003-002-04)*. Wiley.

Investopedia. (2018, November 12). *Sector and Industries Analysis*. Retrieved May 6, 2020, from Investopedia: <https://www.investopedia.com/ask/answers/041415/what-aerospace-sector.asp>

Jama Software. (2020). *Figures obtained from Jama Connect Trial Usage, courtesy of Jama Software*. Retrieved from <https://www.jamasoftware.com/>

- Jama Software. (2020). *Optimize Engineering Team Collaboration to Streamline Your Product Development Process*.
- Katz, T. (2019). Evaluation of COTS Hardware Assemblies for use in Risk Averse, Cost Constrained Space-based Systems. *INCOSE IS2019*. INCOSE.
- Katz, T. (2020). When to Constrain the Design? Application of Design Standards on a New Development Program. *INCOSE IS2020*. INCOSE.
- Klariti. (2020). Retrieved from <https://klariti.com/software-development-lifecycle-templates/software-requirements-specification-template/>
- Kumar, V. (2004). Effective Requirements Management. *PMI Global Congress*. PMI.
- Laplante, P. A. (2017). *Requirements Engineering for Software and Systems*. (Third, Ed.) Boca Raton, Florida: CRC Press.
- Leman, J. (2020, April 30). *NASA's Artemis Lunar Lander Is Beginning To Take Shape*. Retrieved August 30, 2020, from Popular Mechanics: <https://www.popularmechanics.com/space/moon-mars/a32332363/artemis-lunar-lander/>
- MAAW. (2020). *What is a Learning Curve?* Retrieved November 21, 2020, from Management and Accounting Web (MAAW): <https://maaw.info/LearningCurveSummary.htm>
- Madachy, R. (2015). *Systems Engineering Cost Estimation Workbook*.
- Makinen, P. (2013). *DOORS and RE Blog*. Retrieved August 23, 2020, from <https://softqa.wordpress.com/tag/doors/>
- Modern Requirements. (2020). *Requirements Management for Medical Devices*. Retrieved from <https://www.modernrequirements.com/blogs/requirements-management-for-medical-devices/> 2/3
- multiple. (2001). *Agile Manifesto*. Retrieved from <https://agilemanifesto.org/principles.html>
- Muratore, J. (2012). *System Engineering: A Traditional Discipline in a Non-traditional Organization*. SpaceX.
- NASA. (2020). *Artemis: Humanity's Return to the Moon*. Retrieved August 30, 2020, from NASA: <https://www.nasa.gov/specials/artemis/>
- NASA. (2019, Oct 3). Broad Agency Announcement HLS Industry Forum. *Solicitation Number: NNH19ZCQ001K_APPENDIX-H-HLS* .
- NASA. (2009). *Charter of the Review of U.S. Human Space Flight Plans Committee*.
- NASA. (2008). Constellation Architecture Requirements Document (CARD) rev B.

NASA. (2011). *Constellation Program Lessons Learned Volume 1: Executive Summary*.

NASA. (2010). Constellation Program Systems Engineering Plan (SEMP) Rev C.

NASA. (2013). Exploring Mars' Climate History. *NASA MAVEN Press Kit* .

NASA. (2019). *HLS Requirements Specification*.

NASA. (2019). Human Landing System Statement of Work.

NASA JPL. (2012). *Rightsizing Requirements, A Quantitative Assessment of Six Recent JPL Flight Projects*.

NASA. (2020). *Mars Curiosity Rover*. Retrieved from <https://mars.nasa.gov/msl/spacecraft/rover/summary/#:~:text=Fast%20Facts&text=About%20the%20size%20of%20a,height%20of%20a%20basketball%20player>.

NASA. (2007, September 17). *Mars Science Laboratory Project Changes Respond to Cost Increases, Keep Mars Program On Track*. Retrieved from Curiosity News: <https://mars.nasa.gov/news/83/mars-science-laboratory-project-changes-respond-to-cost-increases-keep-mars-program-on-track/?site=msl>

NASA. (2020). *Mars Science Laboratory/Curiosity Fact Sheet*.

NASA. (2012). MAVEN Mission Requirements Document (MRD), MAVEN-PM-RQMT-0005 rev W.

NASA. (2011). MAVEN Project Plan. *MAVEN-PM-PLAN-0008* .

NASA. (2020). *More About the Human Landing System Program*. Retrieved August 30, 2020, from <https://www.nasa.gov/content/more-about-the-human-landing-system-program>

NASA. (2019, May 23). *NASA Chief Explains Artemis Phase I, Announces Commercial Partner*. Retrieved August 30, 2020, from YouTube: <https://www.youtube.com/watch?v=OaORpIVht0c>

NASA. (2015). NASA Mission Reveals Speed of Solar Wind Stripping Martian Atmosphere. *NASA Press Release 15-217* .

NASA. (2007). *NASA Systems Engineering Handbook*. NASA Headquarters.

NASA. (2020). *NASA Systems Engineering Handbook*. Retrieved 2020, from <https://www.nasa.gov/seh/6-2-requirements-management>

NASA. (2008, December 04). *Next NASA Mars Mission Rescheduled for 2011*. Retrieved from NASA (press release): https://www.nasa.gov/mission_pages/mars/news/msl-20081204.html

NASA. (2016). Standard Materials and Processes Requirements for Spacecraft.

NASA. (2015). Thermal and Fluids Analysis Workshop Systems Engineering Training.

No Magic, Inc./Dassault Systems. (2020). *Cameo Diagram Descriptions*. Retrieved July 14, 2020, from <https://docs.nomagic.com/display/CRMP190/Diagram+descriptions>

No Magic, Inc./Dassault Systems. (2020). *Cameo Simulation Toolkit*. Retrieved November 21, 2020, from <https://www.nomagic.com/product-addons/magicdraw-addons/cameo-simulation-toolkit>

NOAA. (2020, February 19). *GOES-16 Transition*. Retrieved August 19, 2020, from NOAA: <https://www.ospo.noaa.gov/Operations/GOES/16/transition.html>

NOAA/NASA. (2017). *An Overview of the Design and Development of the GOES R Series Space Segment*.

NOAA/NASA. (2020). GOES-R Mission Requirements Document (MRD).

Pena, M., & Valerdi, R. (2014). Characterizing the Impact of Requirements. *Systems Engineering (DOI 10.1111/sys.21288)* .

Pinto, J. (2016). *Project Management: Achieving Competitive Advantage* (Fourth Edition ed.). Pearson.

PMI. (2014). *PMI's Pulse of the Profession: Requirements Management - A Core Competency for Project and Program Success*. PMI.

PMI. (2016). *Requirements Management: A Practice Guide*. Newtown Square, Pennsylvania: Project Management Institute, Inc.

Pohl, K. (2010). *Requirements Engineering Fundamentals, Principles, and Techniques*. Springer.

QRA. (2020). *How to Reuse Requirements; 7 Tips for Reusability Best Practices*. Retrieved August 22, 2020, from qracorp.com

Seilevel. (2016). *Requirements Management Tool Evaluation Report*.

Seilevel. (2011). *Seilevel's Evaluations of Requirements Management Tools: Summaries and Scores*.

Sols, A. (2016). *Requirements Engineering and Management - A Systems Approach*. Madrid, Spain: Amazon CreateSpace.

Space Foundation. (2019). *Space Foundation Annual Report*.

Space Policy Online. (2012, March 1). *GAO Slams JWST, MSL Cost Overruns*. Retrieved from SpacePolicyOnline. Com: <https://spacepolicyonline.com/news/gao-slams-jwst-msl-cost-overruns/>

Stecklein, J. (2004). *Error Cost Escalation Through the Project Life Cycle, NASA Technical Report*. NASA.

Toval, A., Nicolas, J., Moros Valle, B., & Lasheras, J. (2008). *Eight Key Issues for an Effective Reuse-based Requirements Process*. Computer Systems Science and Engineering.

U.S. Dept of Commerce. (2013). *Audit of Geostationary Operational Environment Satellite-R Series: Comprehensive Mitigation Approaches, Strong Systems Engineering, and Cost Controls are Needed to Reduce Risks of Coverage Gaps*. Office of Inspector General.

U.S. White House. (2017). *Presidential Memorandum on Reinvigorating America's Human Space Exploration Program*. Retrieved August 30, 2020, from Office of Space Commerce: <https://www.whitehouse.gov/presidential-actions/presidential-memorandum-reinvigorating-americas-human-space-exploration-program/>

Valerdi, R. (2010, July 8). *MIT COSYSMO Downloads*. Retrieved August 8, 2020, from COSYSMO: <http://cosysmo.mit.edu/downloads/>

Valerdi, R., Wang, G., Ankrum, A., Millar, C., & Roedler, G. (2008). COSYSMO Reuse Extension. INCOSE.

Visure Solutions. (2020). *Requirements Reusability*. Retrieved August 22, 2020, from Visure Solutions: <https://visuresolutions.com/requirements-reusability>

Welch, R., Limonadi, D., & Manning, R. (2013). *Systems Engineering the Curiosity Rover: A Retrospective*.

Welch, R., Limonadi, D., & Manning, R. (2013). Systems Engineering the Curiosity Rover: A Retrospective. *8th International Conference on System of Systems Engineering (SoSE)*, DOI: 10.1109/SYSoSE.2013.6575245.

Wertz, J., Everett, D., & Puschell, J. (2011). *Space Mission Engineering: The New SMAD*. Hawthorne: Microcosm Press; Figures reprinted with permission.

Westcott, J. (2014). *SpaceX Says Requirements, Not Markup, Make Government Missions More Costly*. Space News.

Wheatcraft, L., Ryan, M., Dick, J., & Llorens, J. (2019). Information-based Requirement Development & Management. *INCOSE IS2019 Proceedings* .

Zimmerman, I. (. (2014). *Software Education*. Retrieved from YouTube: <https://www.youtube.com/watch?v=N4Z7h4qt1MU>

A1. Model Elements

The Requirement Management SysML model elements are located in the Cameo containment tree, which provides an organization of the various objects (blocks), data (value properties), behaviors (activities), calculations (constraints), and relationships between them. Having an element within the model allows it to be shown within various diagrams, which provide different viewpoints of the model (structure, behavior).

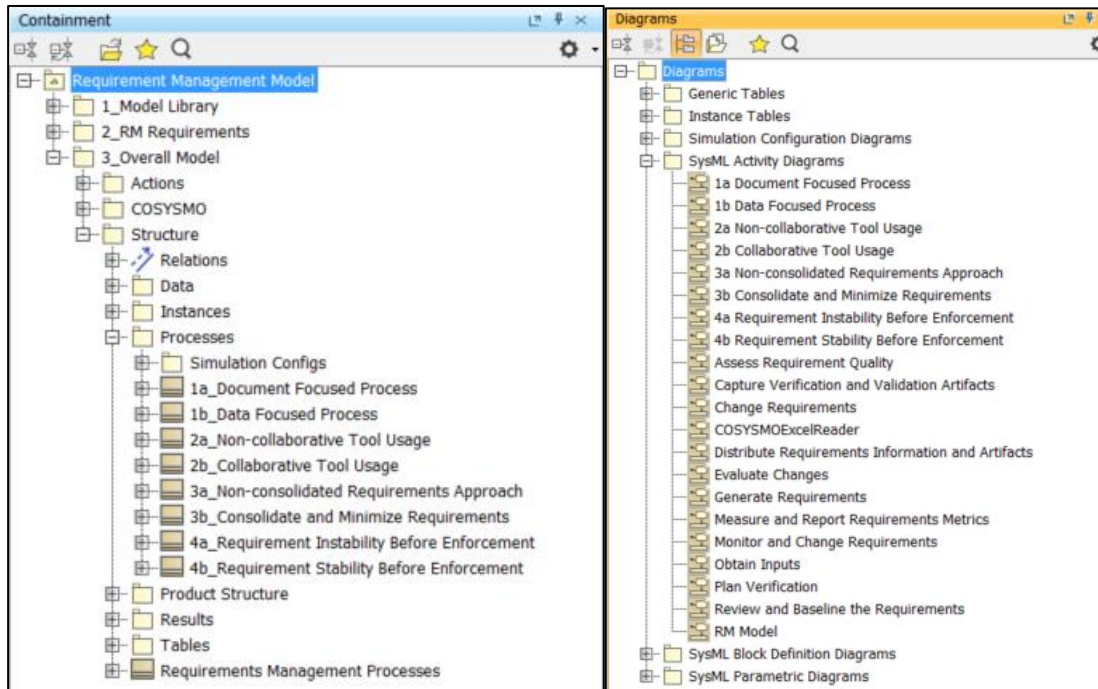


Figure 164. Requirement Model Contents.

A2. Block Definition Diagrams

The following block definition diagrams provide the hierarchy of containment and association relationships between the requirements management process, the project, the sub-processes, and the various data elements (value properties) used throughout the model.

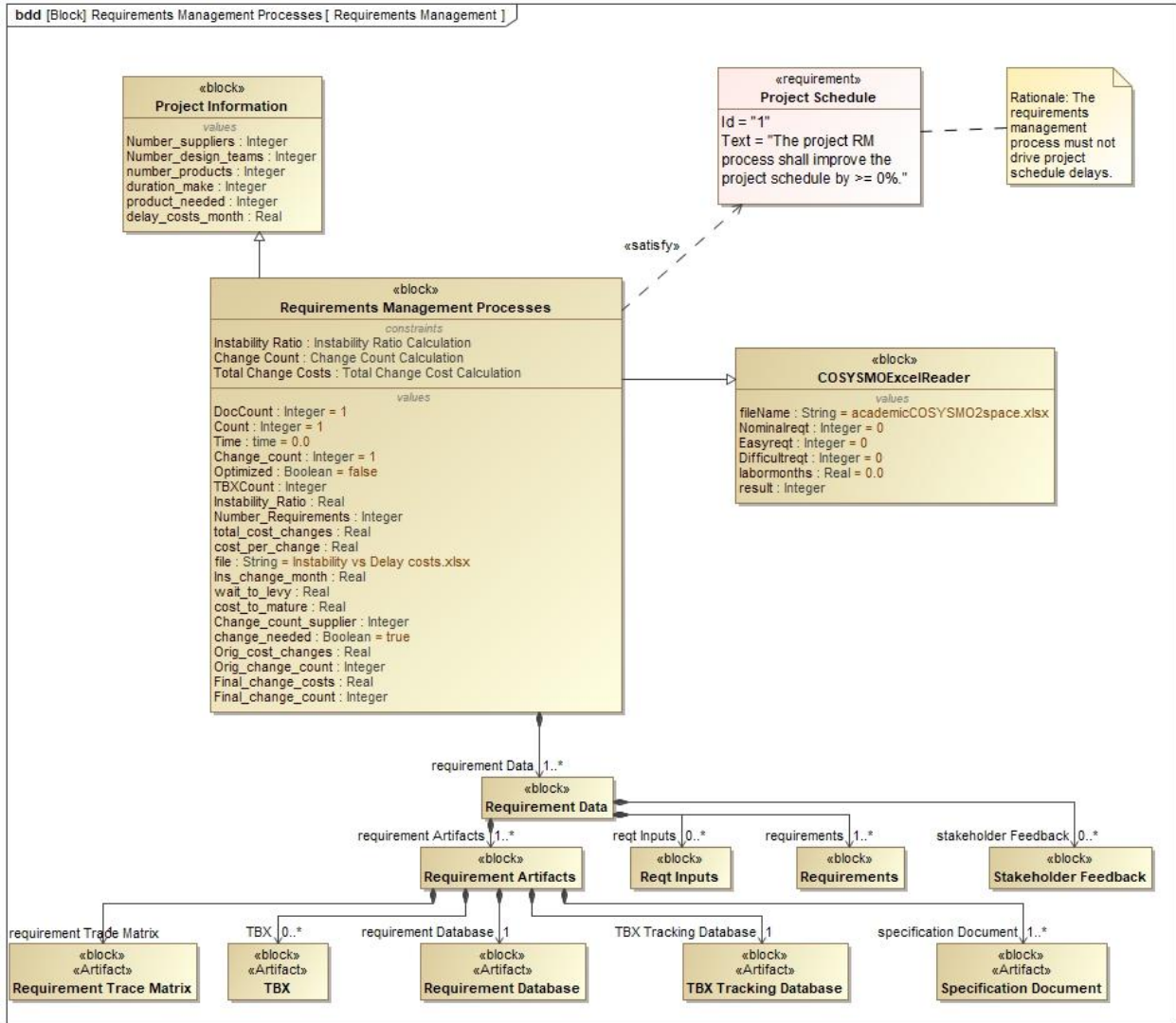


Figure 165. Requirement Model Overall Block Diagram.

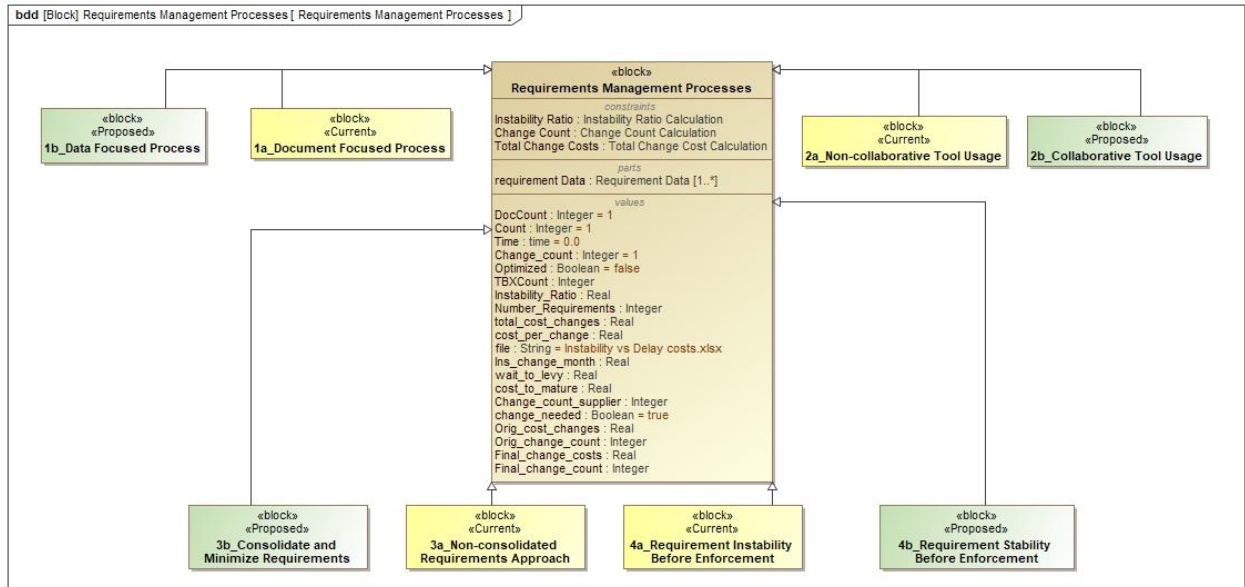


Figure 166. Requirement Processes Block Diagram.

A2. Parametric Diagrams

The following diagrams provide calculations within the model for parameters, these are fully described in Chapter 6.

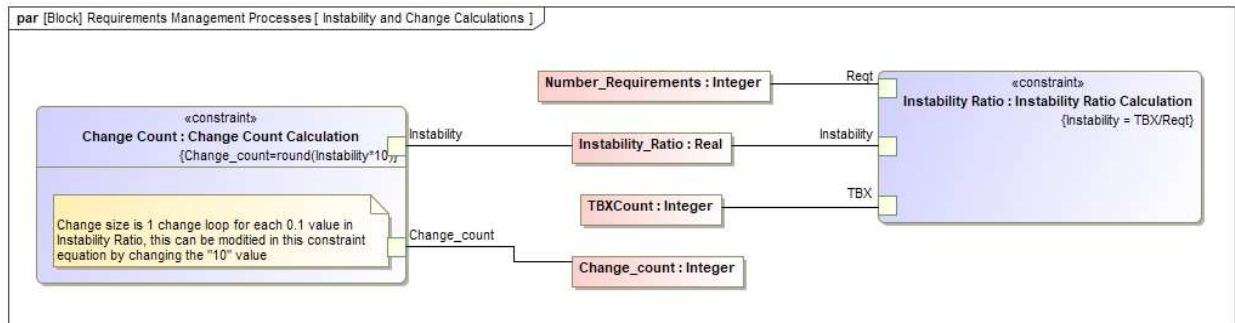


Figure 167. Instability Calculation Parametric Diagram.

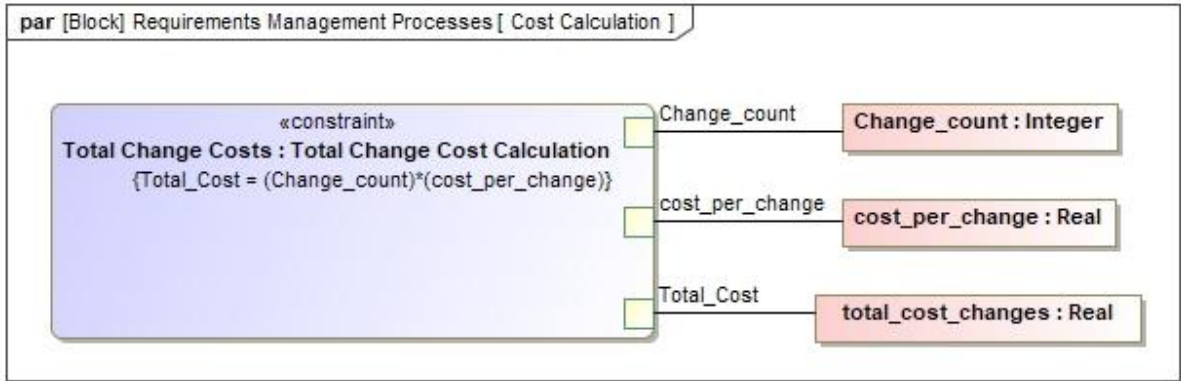


Figure 168. Total Change Cost Calculation Parametric Diagram.

A3. Activity Diagrams

The following provides the current state and optimized processes 1-4.

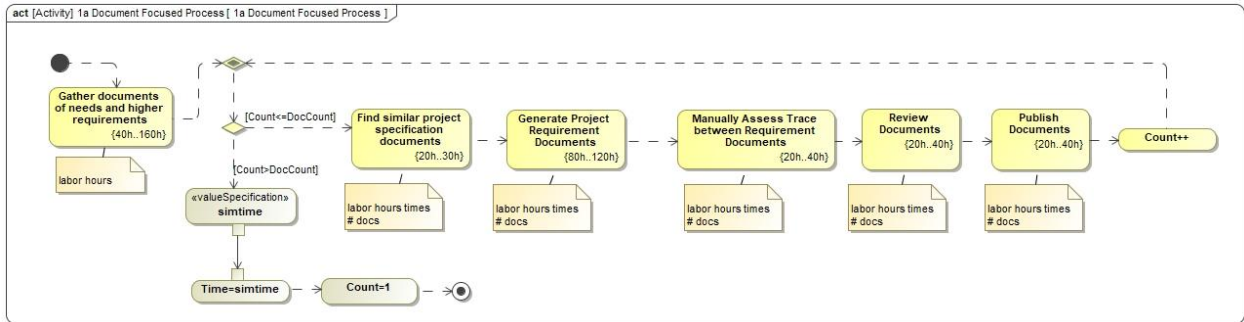


Figure 169. Process 1a Document Centric Activity Diagram.

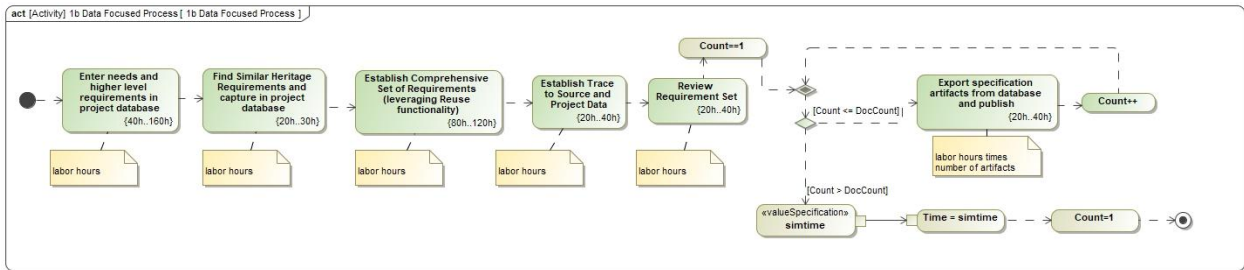


Figure 170. Process 1b Data Centric Activity Diagram.

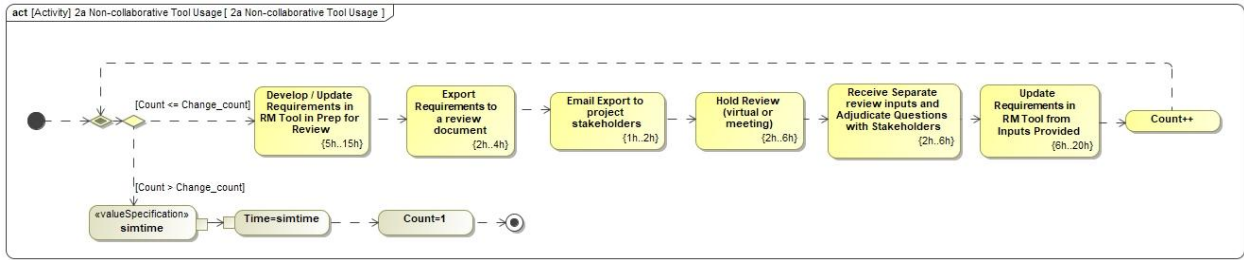


Figure 171. Process 2a Non-Collaborative Tool Usage Activity Diagram.

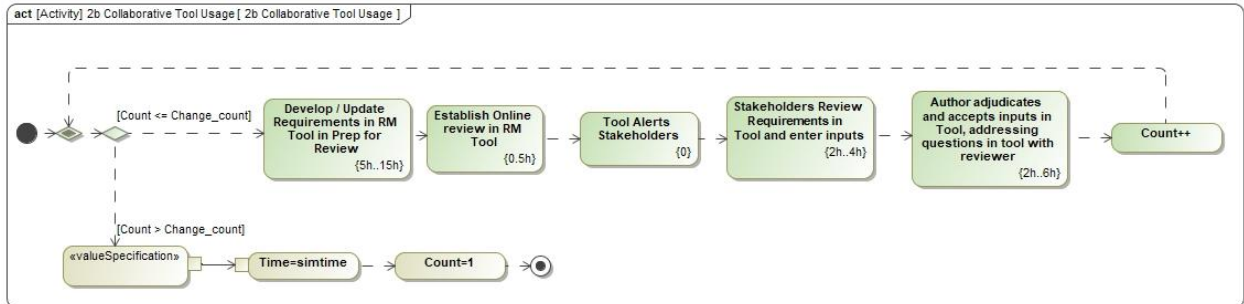


Figure 172. Process 2b Collaborative Tool Usage Activity Diagram.

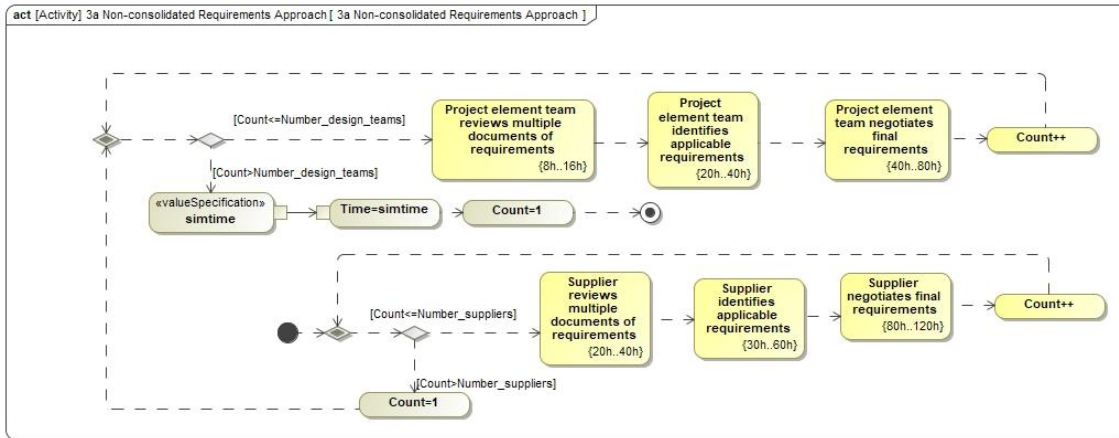


Figure 173. Process 3a Non-Consolidated Requirements Activity Diagram.

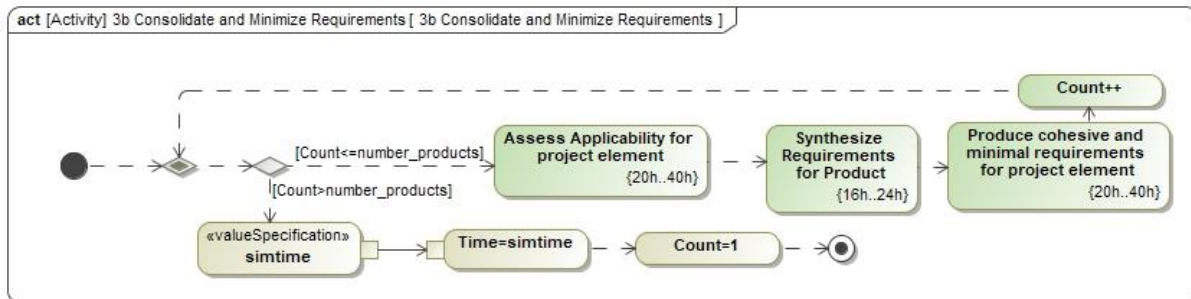


Figure 174. Process 3b Minimize and Consolidate Requirements Activity Diagram.

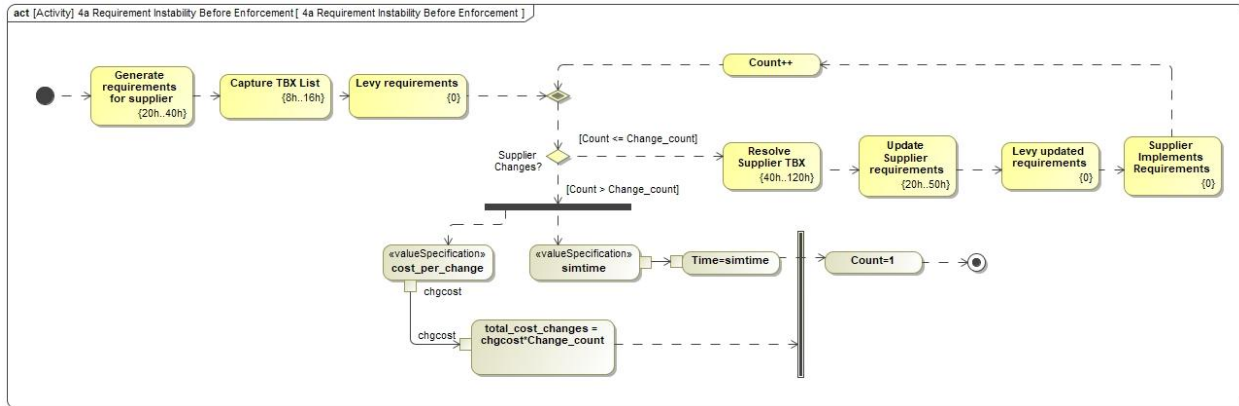


Figure 175. Process 4a Requirement Instability Before Levy Activity Diagram.

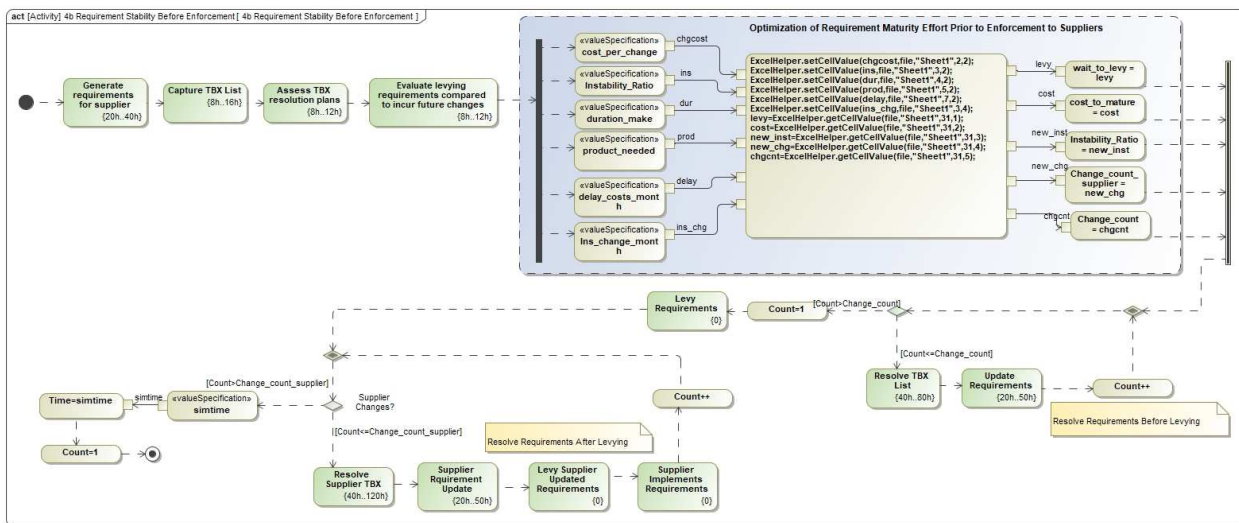


Figure 176. Process 4b Requirement Stability Before Levy Activity Diagram.

The following diagrams make up the overall Requirements Management process model. The different paths for Current and Optimized are shown as activities within the Generate Requirements and Distribute Requirements Information and Artifacts actions. The break-out of the various activities are provided following the overall RM model.

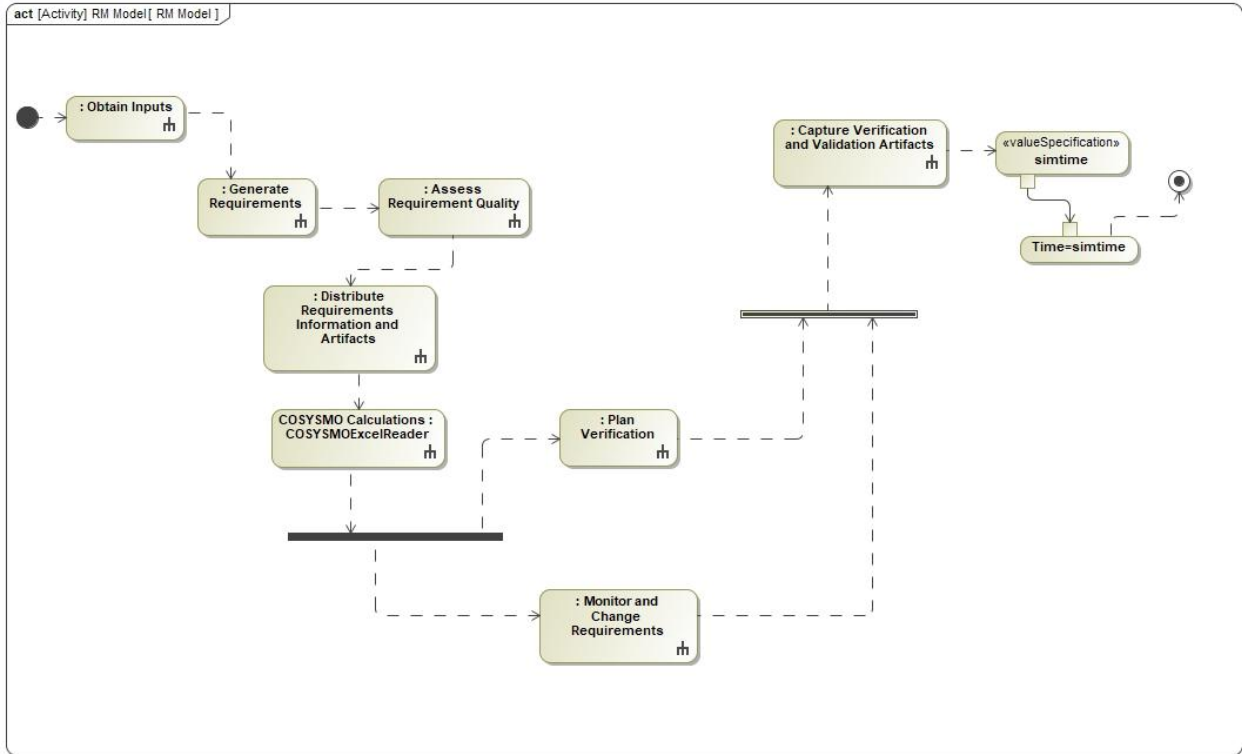


Figure 177. Overall Requirement Management Activity Diagram.

Obtain Inputs Activity

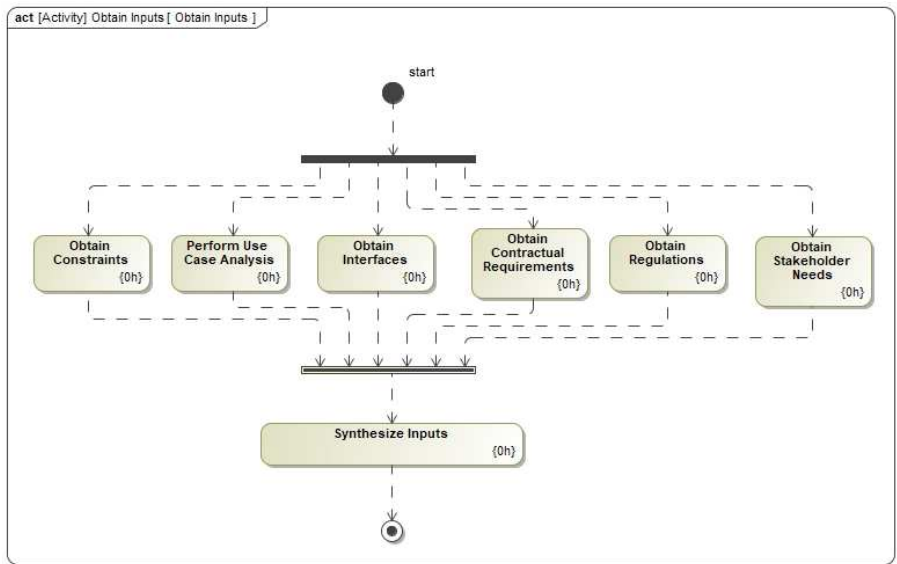


Figure 178. Obtain Inputs Activity Diagram.

Generate Requirements Activity (with optimization option)

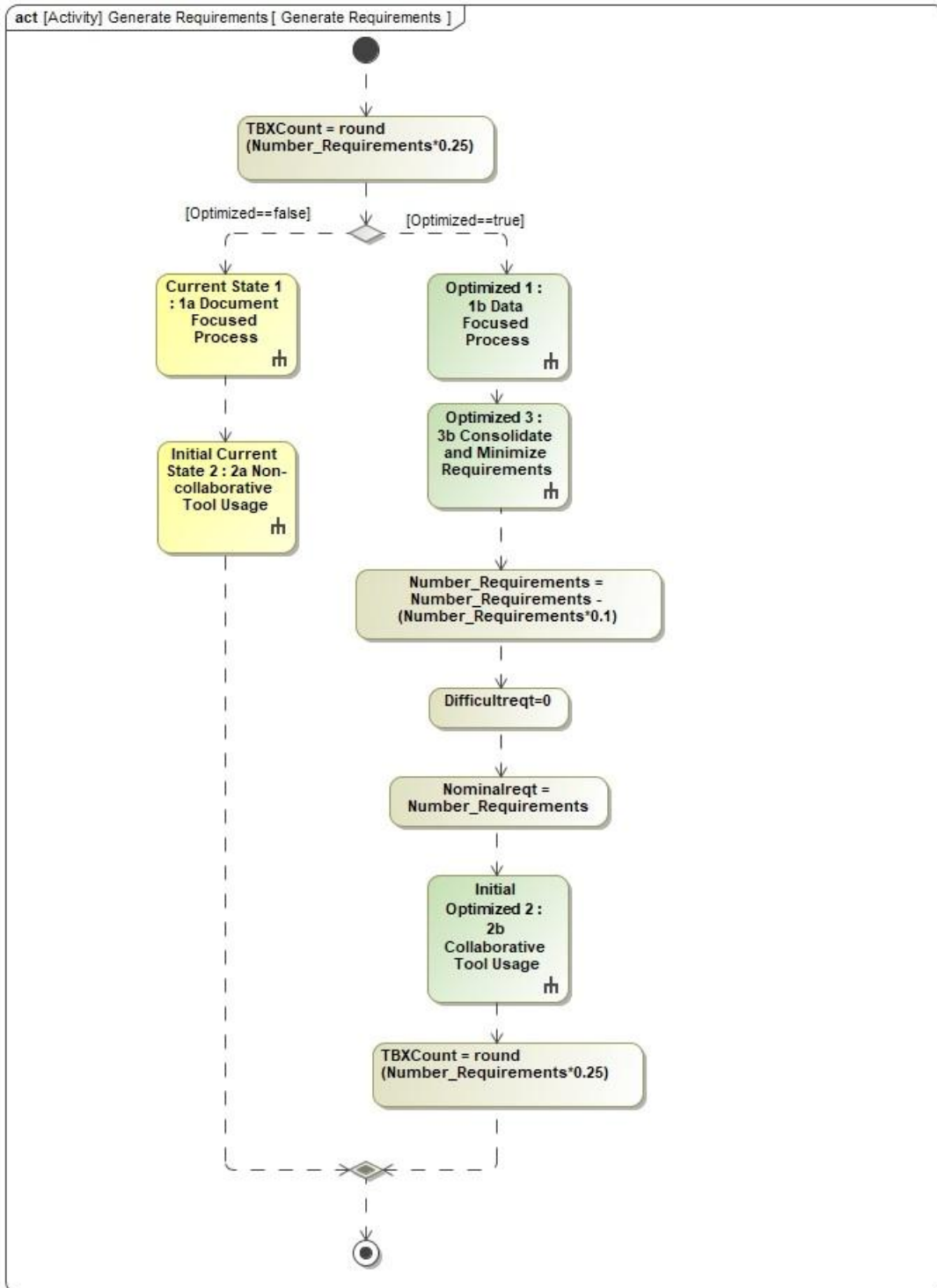


Figure 179. Generate Requirements Activity Diagram.

Assess Requirement Quality Activity

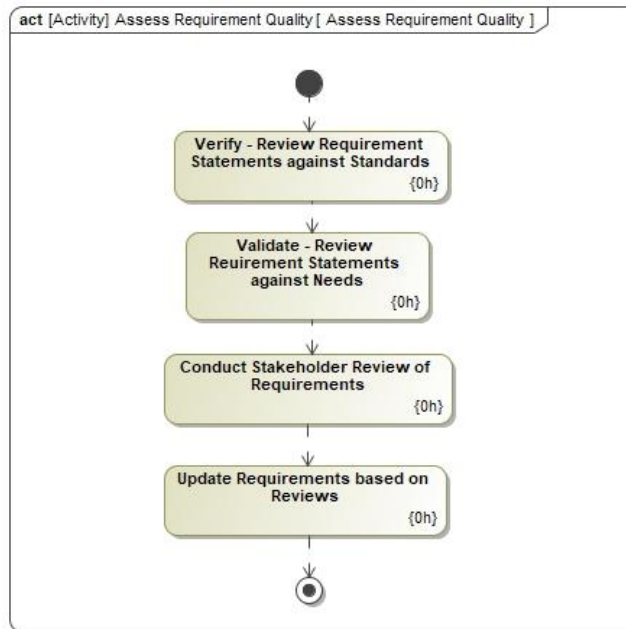


Figure 180. Assess Requirement Quality Activity Diagram.

Distribute Requirement Information and Artifacts (with optimization option)

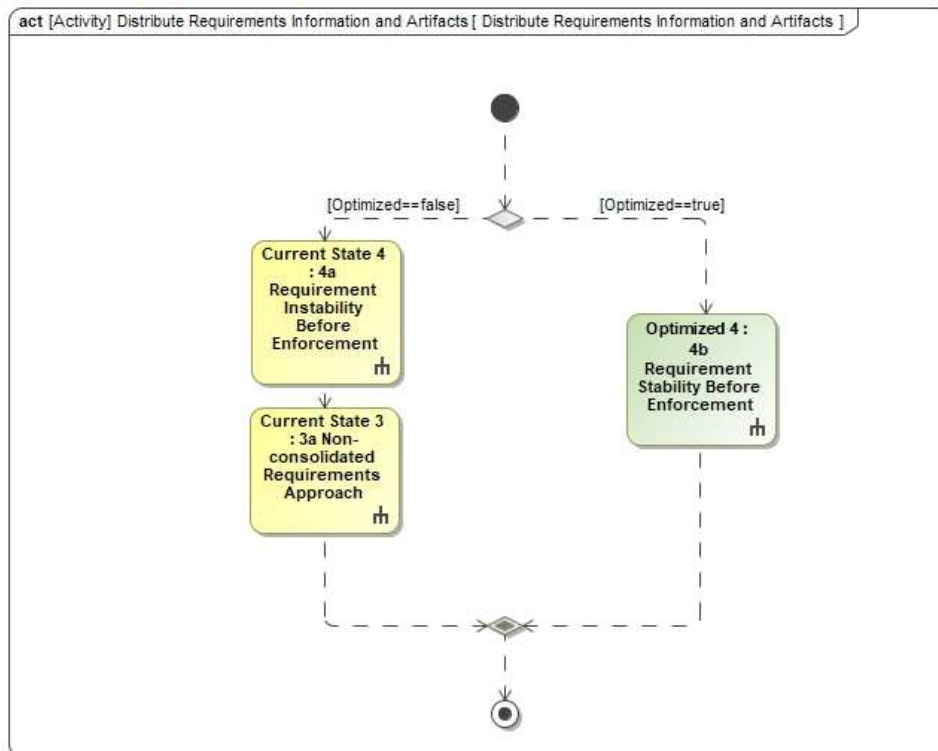


Figure 181. Distribute Requirements Activity Diagram.

Calculate SE labor hours from COSYSMO

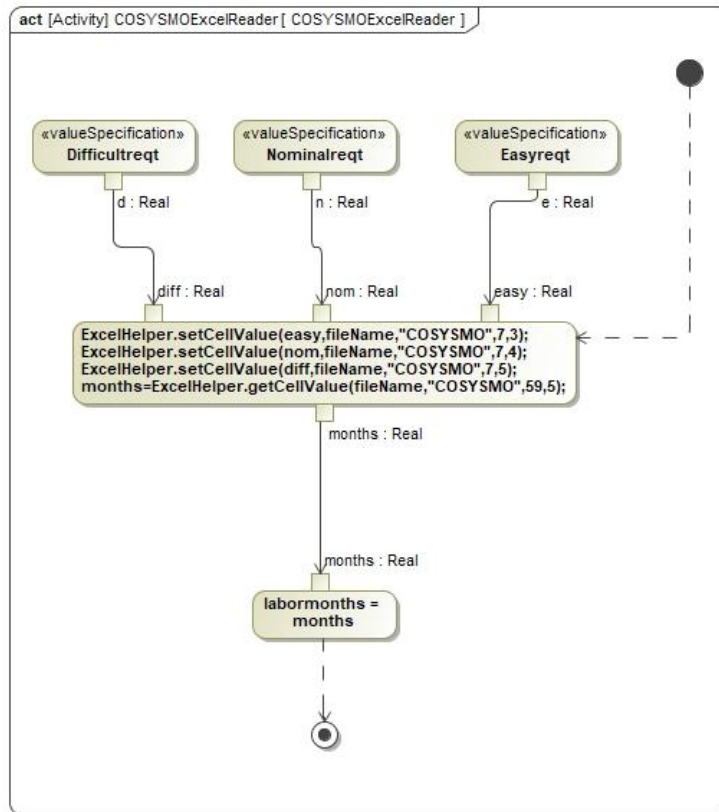


Figure 182. COSYSMO Reader Activity Diagram.

Plan Verification Activity

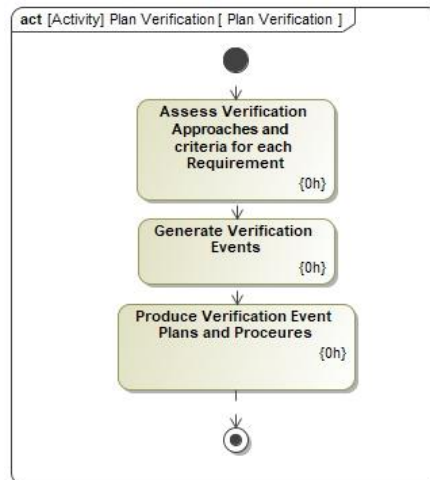


Figure 183. Plan Verification Activity Diagram.

Monitor and Change Requirements Activity and Sub-Activities

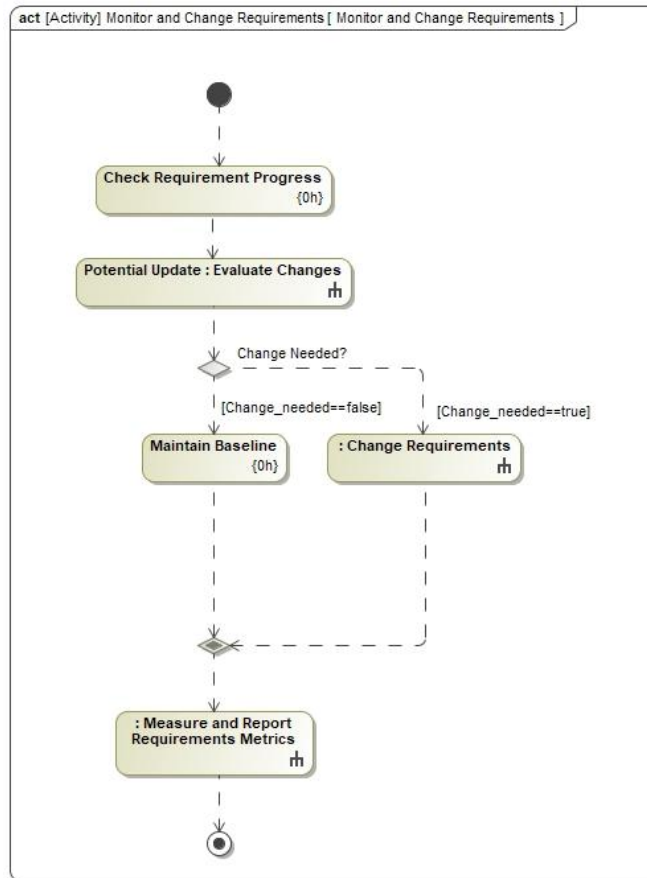


Figure 184. Monitor and Change Requirements Activity Diagram.

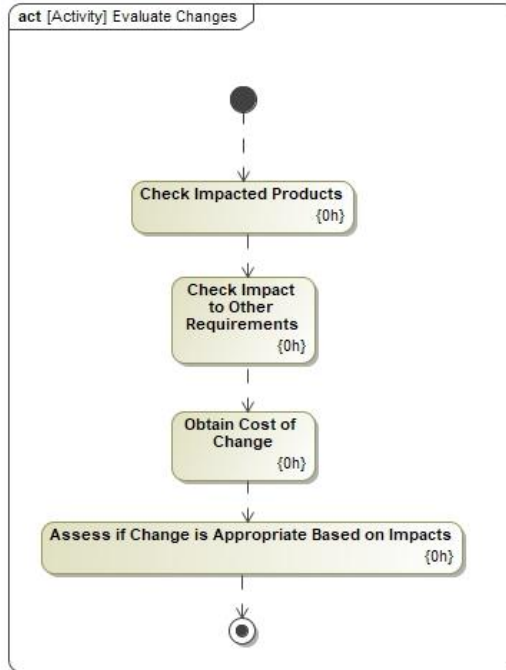


Figure 185. Evaluate Changes Activity Diagram.

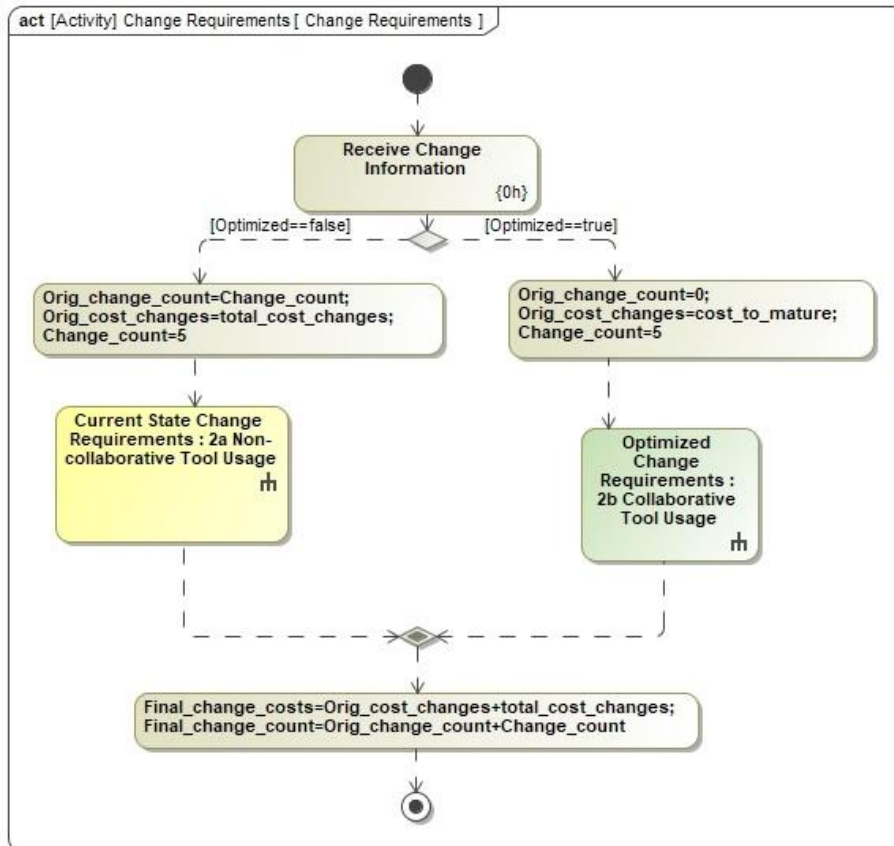


Figure 186. Change Requirements Activity Diagram.

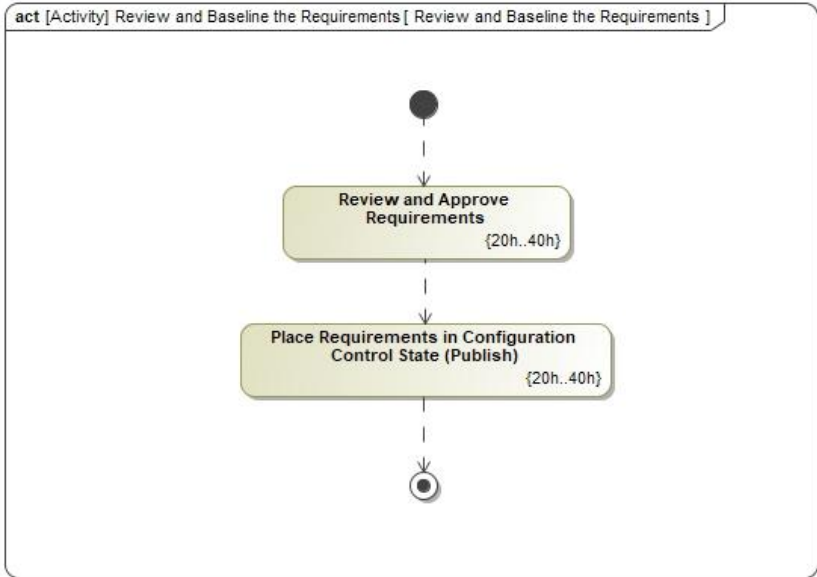


Figure 187. Review and Baseline Requirements Activity Diagram.

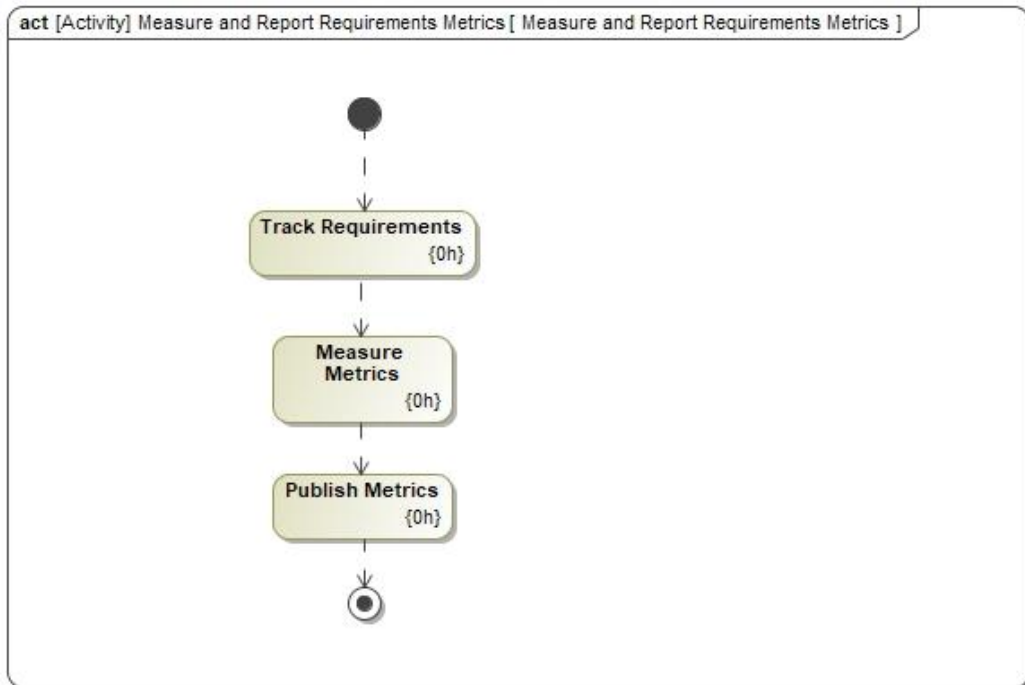


Figure 188. Measure and Report Requirement Metrics Activity Diagram.

Capture Verification and Validation Artifacts Activity

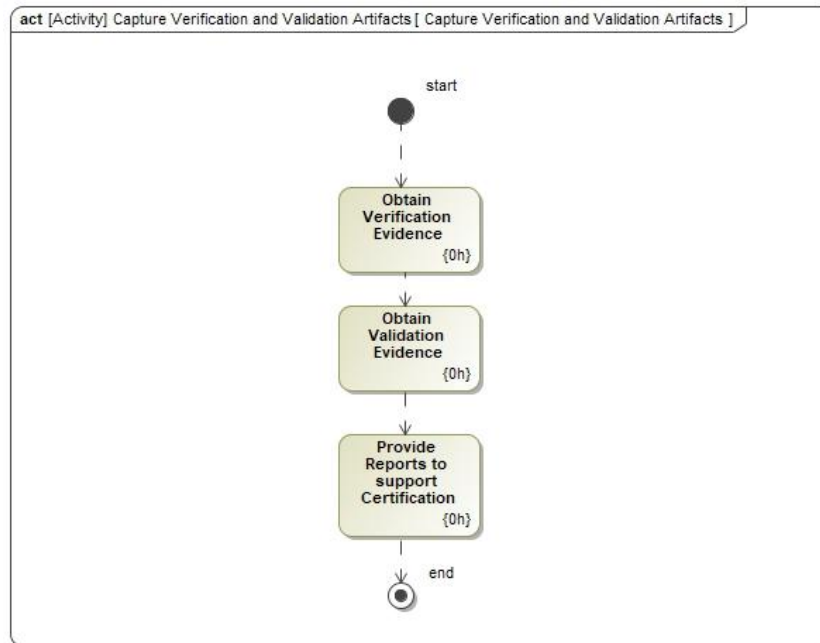


Figure 189. Capture Verification and Validation Activity Diagram.

A4. Instance Blocks

Process Instance Blocks

The following instance blocks establish starting value assignments for the various processes used in the simulations. Values of the parameters were changed prior to simulation.



Figure 190. Process 1 Instance Blocks.



Figure 191. Process 2 Instance Blocks.

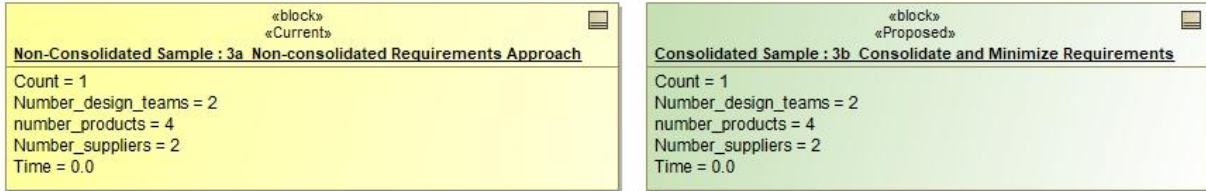


Figure 192. Process 3 Instance Blocks.

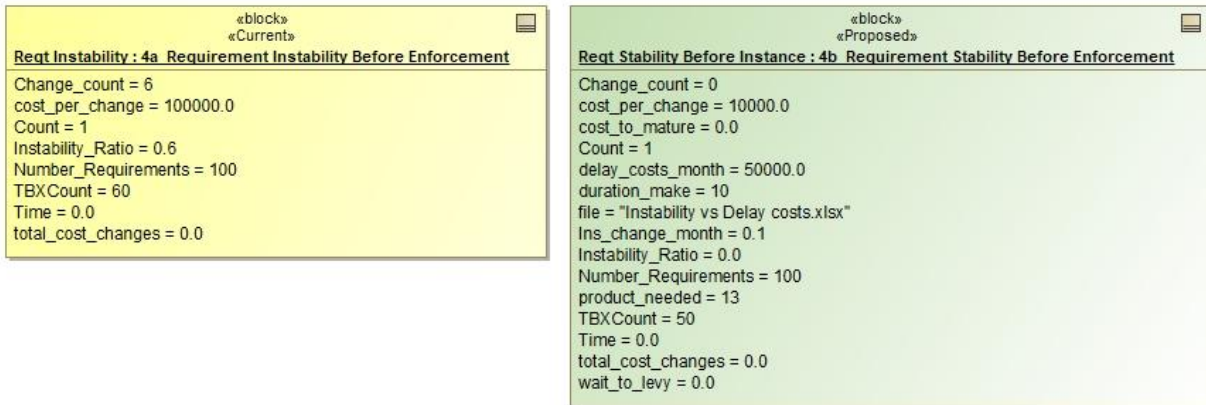


Figure 193. Process 4 Instance Blocks.

Space Project Instance Blocks

The following instance blocks establish starting value assignments for the various space projects used in the case study simulations. Values of the parameters were changed prior to simulation.

«block» MAVEN : Requirements Management Processes	«block» MSL : Requirements Management Processes
Change_count = 1 Change_count_supplier = 0 change_needed = true cost_per_change = 150000.0 cost_to_mature = 0.0 Count = 1 delay_costs_month = 50000.0 Difficultreqt = 0 DocCount = 6 duration_make = 10 Easyreqt = 0 file = "Instability vs Delay costs.xlsx" fileName = "academicCOSYSMO2space.xlsx" Final_change_costs = 0.0 Final_change_count = 0 Ins_change_month = 0.1 Instability_Ratio = 0.0 labormonths = 0.0 Nominalreqt = 660 Number_design_teams = 0 number_products = 6 Number_Requirements = 660 Number_suppliers = 6 Optimized = true Orig_change_count = 0 Orig_cost_changes = 0.0 product_needed = 13 result = 0 TBXCount = 0 Time = 0.0 total_cost_changes = 0.0 wait_to_levy = 0.0	Change_count = 1 Change_count_supplier = 0 change_needed = true cost_per_change = 150000.0 cost_to_mature = 0.0 Count = 1 delay_costs_month = 50000.0 Difficultreqt = 309 DocCount = 1 duration_make = 10 Easyreqt = 0 file = "Instability vs Delay costs.xlsx" fileName = "academicCOSYSMO2space.xlsx" Final_change_costs = 0.0 Final_change_count = 0 Ins_change_month = 0.1 Instability_Ratio = 0.0 labormonths = 0.0 Nominalreqt = 202 Number_design_teams = 22 number_products = 34 Number_Requirements = 511 Number_suppliers = 12 Optimized = false Orig_change_count = 0 Orig_cost_changes = 0.0 product_needed = 13 result = 0 TBXCount = 0 Time = 0.0 total_cost_changes = 0.0 wait_to_levy = 0.0

Figure 194. Space Project MSL / Maven Instance Blocks.

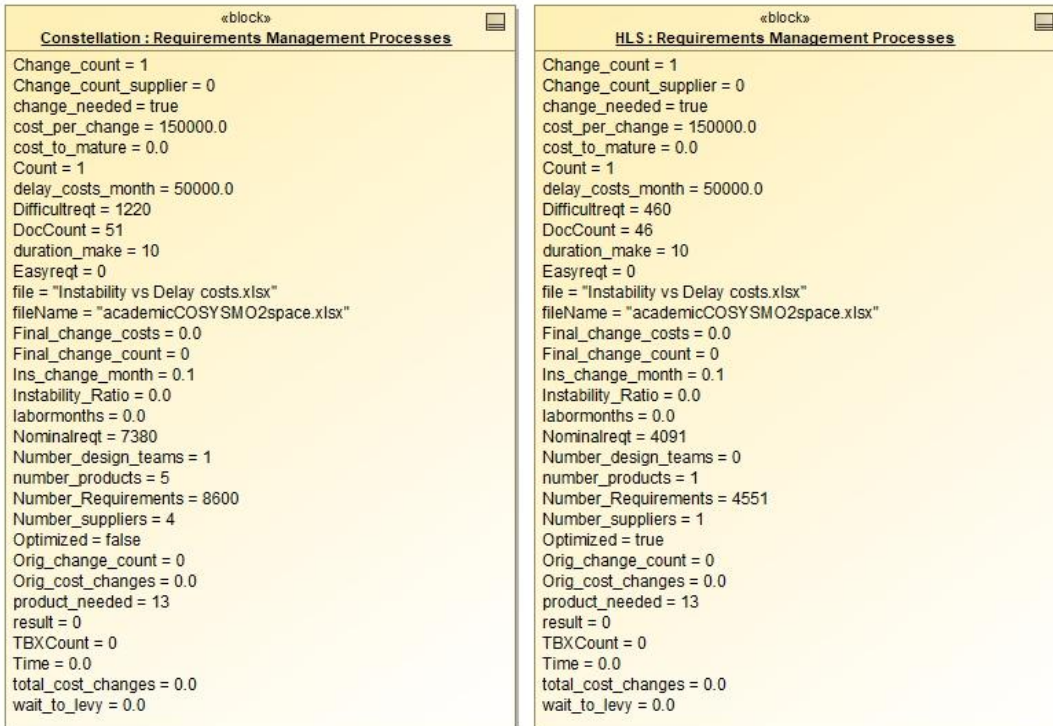


Figure 195. Space Project Constellation / HLS Instance Blocks.

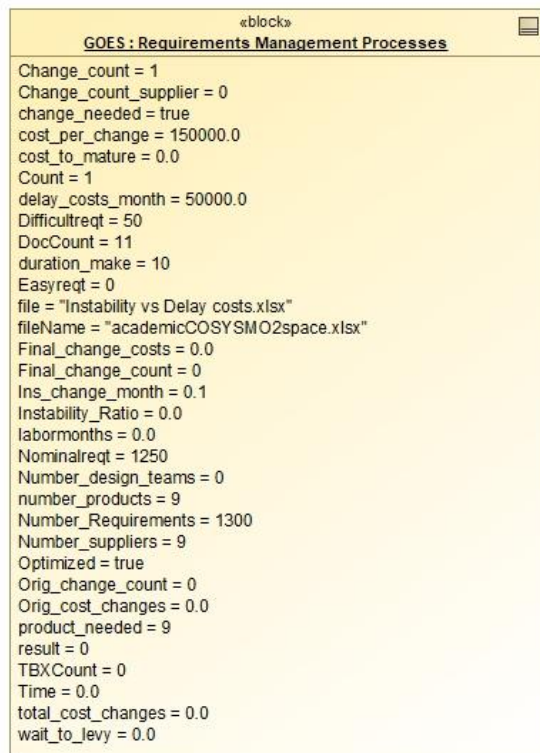
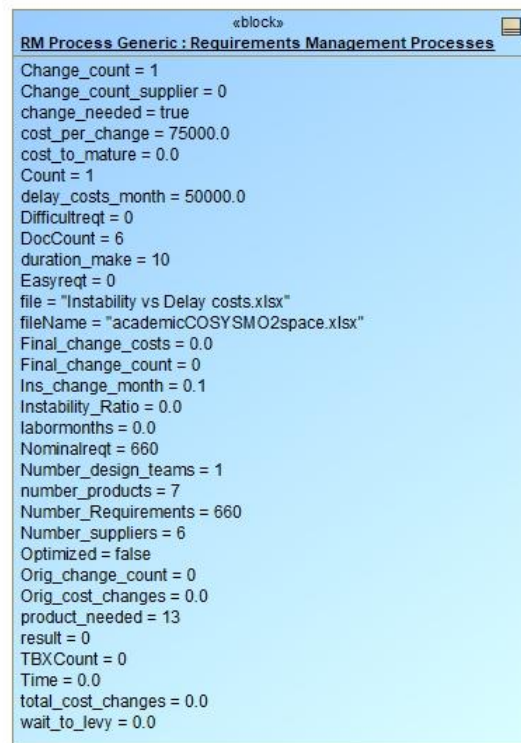


Figure 196. Space Project GOES Instance Block.

Generic RM Model Instance Block

A generic instance was generated to allow for general usage of RM Model for any set of parameters.



```
«block»
RM Process Generic : Requirements Management Processes
Change_count = 1
Change_count_supplier = 0
change_needed = true
cost_per_change = 75000.0
cost_to_mature = 0.0
Count = 1
delay_costs_month = 50000.0
Difficultreqt = 0
DocCount = 6
duration_make = 10
Easyreqt = 0
file = "Instability vs Delay costs.xlsx"
fileName = "academicCOSYSMO2space.xlsx"
Final_change_costs = 0.0
Final_change_count = 0
Ins_change_month = 0.1
Instability_Ratio = 0.0
labormonths = 0.0
Nominalreqt = 660
Number_design_teams = 1
number_products = 7
Number_Requirements = 660
Number_suppliers = 6
Optimized = false
Orig_change_count = 0
Orig_cost_changes = 0.0
product_needed = 13
result = 0
TBXCount = 0
Time = 0.0
total_cost_changes = 0.0
wait_to_levy = 0.0
```

Figure 197. Overall Requirement Management General Simulation Configuration.

A5. Simulation Diagrams

The following simulation diagrams establish simulation for the overall RM Model, as well as the individual process steps discussed in Chapter 6.

Case Study Space Project Simulation Configurations

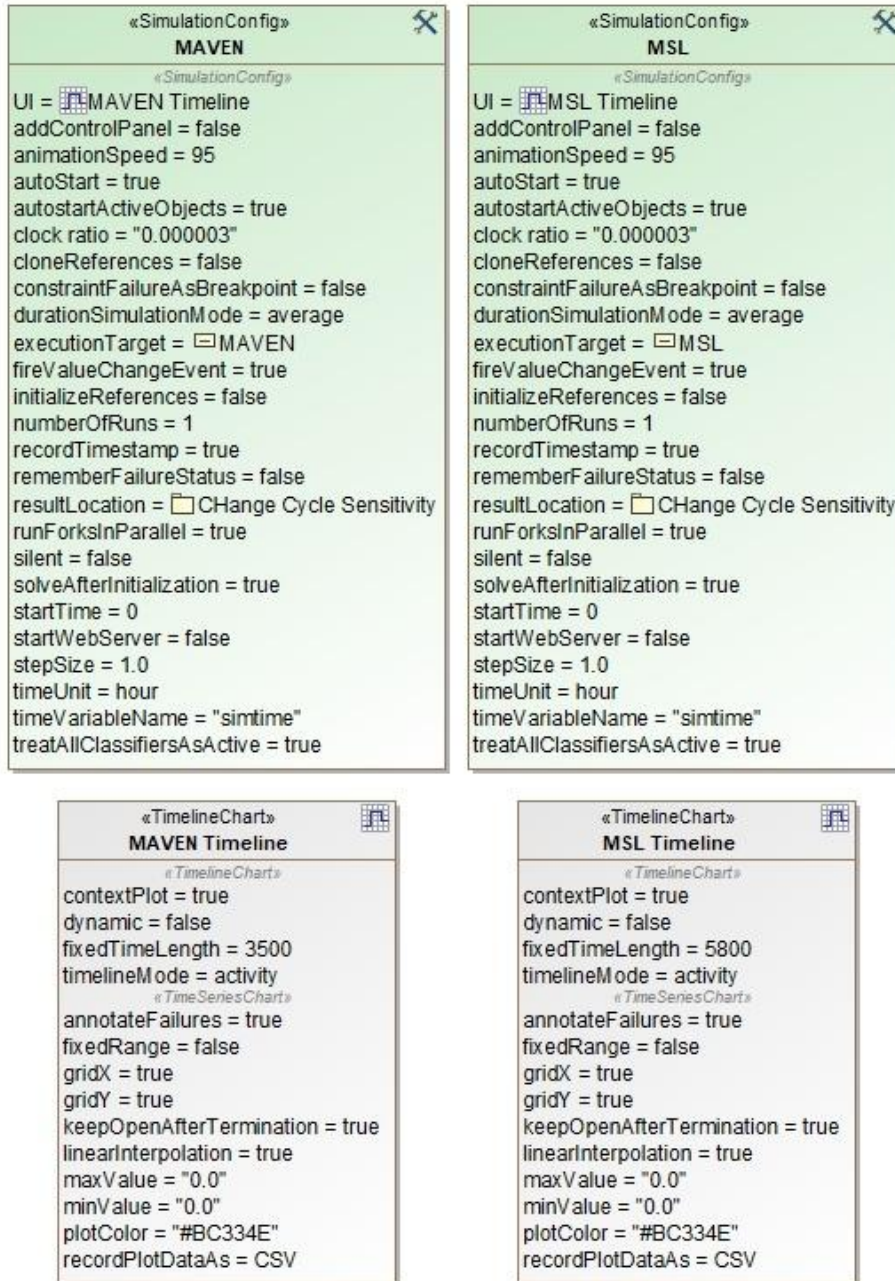


Figure 198. MAVEN and MSL Simulation Configurations.

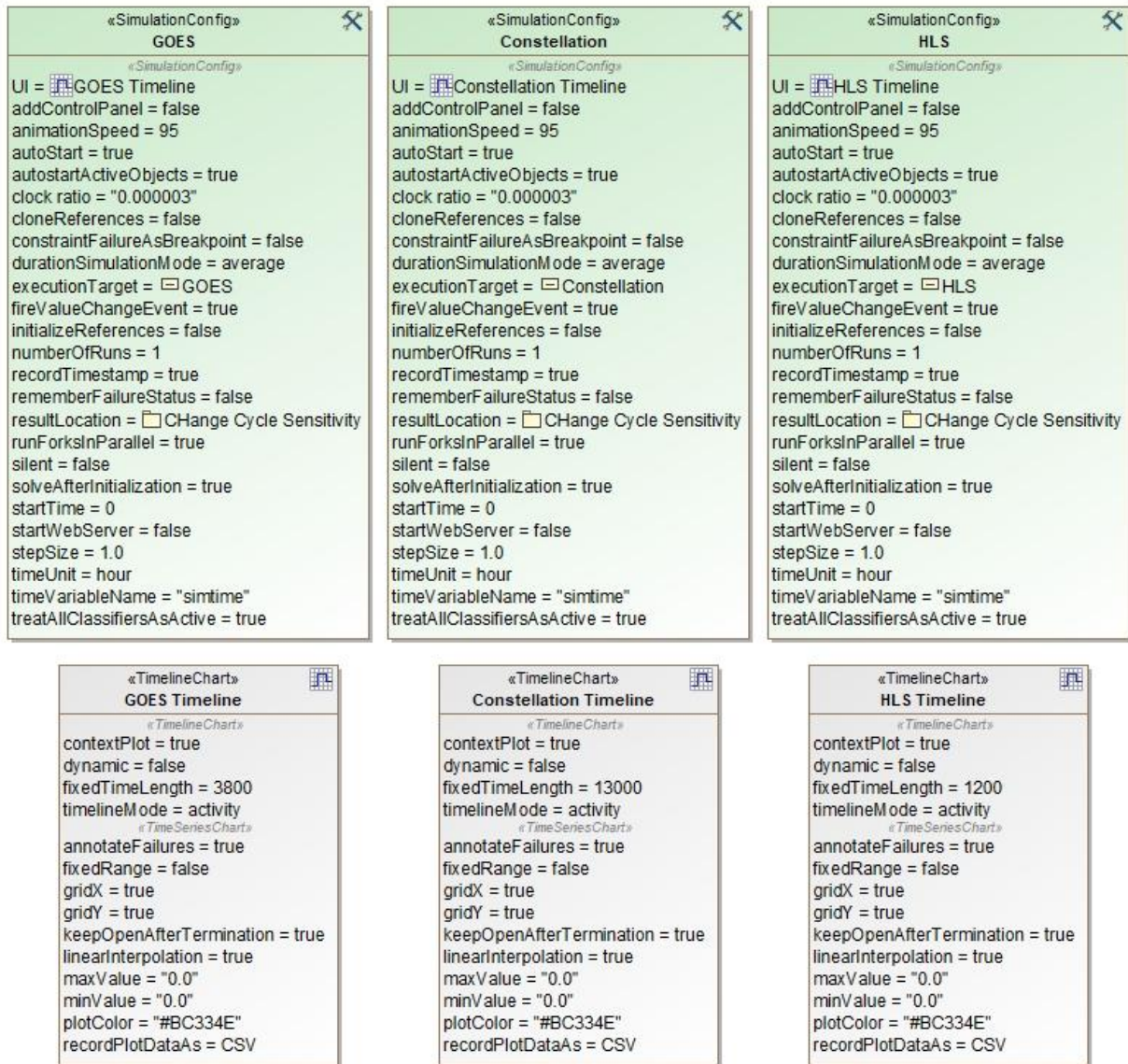


Figure 199. GOES, Constellation and HLS Simulation Configurations.

Individual Processes Simulation Configurations

<p>«SimulationConfig» 1a Doc Centric</p> <p>«SimulationConfig»</p> <pre> addControlPanel = false animationSpeed = 95 autoStart = true autostartActiveObjects = true clock ratio = "0.000003" cloneReferences = false constraintFailureAsBreakpoint = false durationSimulationMode = min executionTarget = <input type="checkbox"/> Doc Process Sample fireValueChangeEvent = true initializeReferences = false numberOfRuns = 1 rememberFailureStatus = false resultLocation = <input type="checkbox"/> 1a Min Results runForksInParallel = true silent = false solveAfterInitialization = true startTime = 0 startWebServer = false stepSize = 1.0 timeUnit = hour timeVariableName = "simtime" treatAllClassifiersAsActive = true </pre>	<p>«SimulationConfig» 1b Data Centric</p> <p>«SimulationConfig»</p> <pre> addControlPanel = false animationSpeed = 95 autoStart = true autostartActiveObjects = true clock ratio = "0.000003" cloneReferences = false constraintFailureAsBreakpoint = false durationSimulationMode = min executionTarget = <input type="checkbox"/> Data Process Sample fireValueChangeEvent = true initializeReferences = false numberOfRuns = 1 rememberFailureStatus = false resultLocation = <input type="checkbox"/> 1b Min Results runForksInParallel = true silent = false solveAfterInitialization = true startTime = 0 startWebServer = false stepSize = 1.0 timeUnit = hour timeVariableName = "simtime" treatAllClassifiersAsActive = true </pre>
<p>«SimulationConfig» 2a Non-Collab Tool</p> <p>«SimulationConfig»</p> <pre> addControlPanel = false animationSpeed = 95 autoStart = true autostartActiveObjects = true clock ratio = "0.000003" cloneReferences = false constraintFailureAsBreakpoint = false durationSimulationMode = min executionTarget = <input type="checkbox"/> Non-collab Tool Sample fireValueChangeEvent = true initializeReferences = false numberOfRuns = 1 rememberFailureStatus = false resultLocation = <input type="checkbox"/> 2a Min Results runForksInParallel = true silent = false solveAfterInitialization = true startTime = 0 startWebServer = false stepSize = 1.0 timeUnit = hour timeVariableName = "simtime" treatAllClassifiersAsActive = true </pre>	<p>«SimulationConfig» 2b Collab Tool</p> <p>«SimulationConfig»</p> <pre> addControlPanel = false animationSpeed = 95 autoStart = true autostartActiveObjects = true clock ratio = "0.000003" cloneReferences = false constraintFailureAsBreakpoint = false durationSimulationMode = min executionTarget = <input type="checkbox"/> Collab Tool Sample fireValueChangeEvent = true initializeReferences = false numberOfRuns = 1 rememberFailureStatus = false resultLocation = <input type="checkbox"/> 2b Min Results runForksInParallel = true silent = false solveAfterInitialization = true startTime = 0 startWebServer = false stepDelay = 1.0 timeUnit = hour timeVariableName = "simtime" treatAllClassifiersAsActive = true </pre>

Figure 200. Processes 1 and 2 Simulation Configurations.

```

«SimulationConfig»
3a Non-Consolidated
«SimulationConfig»
addControlPanel = false
animationSpeed = 95
autoStart = true
autostartActiveObjects = true
clock ratio = "0.000003"
cloneReferences = false
constraintFailureAsBreakpoint = false
durationSimulationMode = min
executionTarget =  Non-Consolidated Sample
fireValueChangeEvent = true
initializeReferences = false
numberOfRuns = 1
rememberFailureStatus = false
resultLocation =  3a Min Results
runForksInParallel = true
silent = false
solveAfterInitialization = true
startTime = 0
startWebServer = false
stepDelay = 1.0
timeUnit = hour
timeVariableName = "simtime"
treatAllClassifiersAsActive = true

```

```

«SimulationConfig»
3b Consolidated
«SimulationConfig»
addControlPanel = false
animationSpeed = 95
autoStart = true
autostartActiveObjects = true
clock ratio = "0.000003"
cloneReferences = false
constraintFailureAsBreakpoint = false
durationSimulationMode = min
executionTarget =  Consolidated Sample
fireValueChangeEvent = true
initializeReferences = false
numberOfRuns = 1
rememberFailureStatus = false
resultLocation =  3b Min Results
runForksInParallel = true
silent = false
solveAfterInitialization = true
startTime = 0
startWebServer = false
stepSize = 1.0
timeUnit = hour
timeVariableName = "simtime"
treatAllClassifiersAsActive = true

```

```

«SimulationConfig»
4a Reqt Instability
«SimulationConfig»
addControlPanel = false
animationSpeed = 95
autoStart = true
autostartActiveObjects = true
clock ratio = "0.000003"
cloneReferences = false
constraintFailureAsBreakpoint = false
durationSimulationMode = min
executionTarget =  Reqt Instability
fireValueChangeEvent = true
initializeReferences = false
numberOfRuns = 1
rememberFailureStatus = false
resultLocation =  4a Min Results
runForksInParallel = true
silent = false
solveAfterInitialization = true
startTime = 0
startWebServer = false
stepSize = 1.0
timeUnit = hour
timeVariableName = "simtime"
treatAllClassifiersAsActive = true

```

```

«SimulationConfig»
4b Reqt Stability
«SimulationConfig»
addControlPanel = false
animationSpeed = 95
autoStart = true
autostartActiveObjects = true
clock ratio = "0.000003"
cloneReferences = false
constraintFailureAsBreakpoint = false
durationSimulationMode = min
executionTarget =  Reqt Stability Before Instance
fireValueChangeEvent = true
initializeReferences = false
numberOfRuns = 1
rememberFailureStatus = false
resultLocation =  4b Min Results
runForksInParallel = true
silent = false
solveAfterInitialization = true
startTime = 0
startWebServer = false
stepSize = 1.0
timeUnit = hour
timeVariableName = "simtime"
treatAllClassifiersAsActive = true

```

Figure 201. Processes 3 and 4 Simulation Configurations.

APPENDIX B - DATA TABLES

The following provides more detailed data tables to support the main text, divided into Appendix B1 (case study results) and B2 (sensitivity data for change amount per month)

B1. Case Study Data Tables

Below are the space project requirements management simulation case study results. Note that the process for requirement monitoring included 5 change cycles to account for general changes.

Table 43. Case Study1 Results from Requirements Management Model (Instability Ratio = 0.25, Change Cost = \$75K).

Name	Current State Time (Hours)	Optimized Time (Hours)	Improvement	Current State SE Labor (Months)	Optimized SE Labor (Months)	Improvement	Current State Change Count	Optimized Change Count	Current State Change Costs	Optimized Change Costs	Optimized Wait Time to Levy (Months)
MAVEN	3115	1646	47%	297.8	271.4	9%	8	5	\$ 600,000	\$ 450,000	4.2
MSL	5334	3736	30%	747.0	217.8	71%	8	5	\$ 600,000	\$ 450,000	4.2
GOES	4715	1846	61%	643.1	505.7	21%	8	5	\$ 600,000	\$ 450,000	1.1
Constellation	12542	2916	77%	6109.0	3417.7	44%	8	5	\$ 600,000	\$ 450,000	4.2
HLS	10840	2446	77%	2799.9	1767.0	37%	8	5	\$ 600,000	\$ 450,000	4.2

Table 44. Case Study2 Results from Requirements Management Model (Instability Ratio = 0.5, Change Cost = \$75K).

Name	Current State Time (Hours)	Optimized Time (Hours)	Improvement	Current State SE Labor (Months)	Optimized SE Labor (Months)	Improvement	Current State Change Count	Optimized Change Count	Current State Change Costs	Optimized Change Costs	Optimized Wait Time to Levy (Months)
MAVEN	3417	1700	50%	297.8	271.4	9%	10	5	\$ 750,000	\$ 450,000	4.2
MSL	5636	3790	33%	747.0	217.8	71%	10	5	\$ 750,000	\$ 450,000	4.2

Name	Current State Time (Hours)	Optimized Time (Hours)	Improvement	Current State SE Labor (Months)	Optimized SE Labor (Months)	Improvement	Current State Change Count	Optimized Change Count	Current State Change Costs	Optimized Change Costs	Optimized Wait Time to Levy (Months)
GOES	5017	2110	58%	643.1	505.7	21%	10	5	\$ 750,000	\$ 525,000	2.6
Constellation	12844	2970	77%	6109.0	3417.7	44%	10	5	\$ 750,000	\$ 450,000	4.2
HLS	11142	2500	78%	2799.9	1767.0	37%	10	5	\$ 750,000	\$ 450,000	4.2

Table 45. Case Study3 Results from Requirements Management Model (Instability Ratio = 0.25, Change Cost = \$150K).

Name	Current State Time (Hours)	Optimized Time (Hours)	Improvement	Current State SE Labor (Months)	Optimized SE Labor (Months)	Improvement	Current State Change Count	Optimized Change Count	Current State Change Costs	Optimized Change Costs	Optimized Wait Time to Levy (Months)
MAVEN	3115	1436	54%	297.8	271.4	9%	8	5	\$ 1,200,000	\$ 750,000	2.5
MSL	5334	3526	34%	747.0	217.8	71%	8	5	\$ 1,200,000	\$ 750,000	2.5
GOES	4715	1846	61%	4715	505.7	21%	8	5	\$ 1,200,000	\$ 900,000	1.6
Constellation	12542	2706	78%	6109.0	3417.7	44%	8	5	\$ 1,200,000	\$ 750,000	2.5
HLS	10840	2236	79%	2799.9	1767.0	37%	8	5	\$ 1,200,000	\$ 750,000	2.5

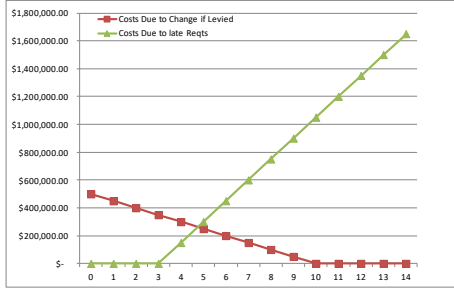
B2. Variation of Change Per Month on Requirement Stability Enforcement

Steady State inputs:

Parameters	
Cost/change	\$ 100,000.00
Starting Instability_ratio	0.5
Duration to make	10
Product Needed	13
Must Start	3
Project Delay Costs/month	\$ 150,000.00

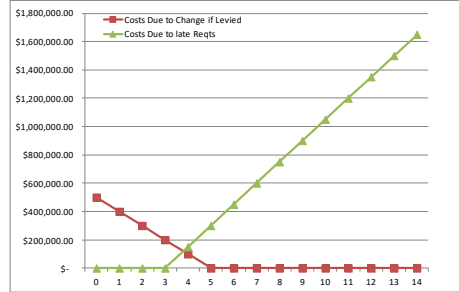
Instability Ratio change per month: 0.05

Optimal Time to Levy	Supplier Change Costs	Supplier Chg Count
4.8	\$ 262,500.00	3



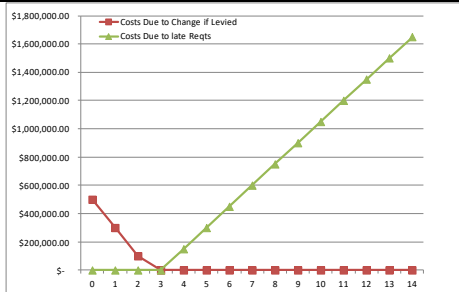
Instability Ratio change per month: 0.1

Optimal Time to Levy	Supplier Change Costs	Supplier Chg Count
3.8	\$ 120,000.00	1



Instability Ratio change per month: 0.2

Optimal Time to Levy	Supplier Change Costs	Chg Count
2.5	\$ -	0



Instability Ratio change per month: 0.4

Optimal Time to Levy	Supplier Change Costs	Chg Count
1.3	\$ -	0

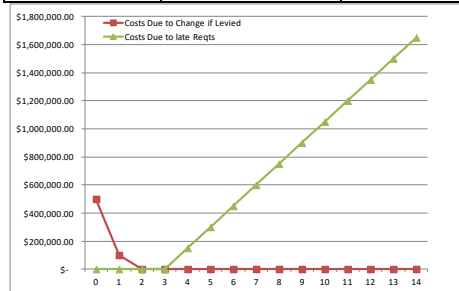


Figure 202. Change Cost Sensitivity Evaluations.