Applying Systems Engineering Modeling Language (SysML) to System Effort Estimation Utilizing Use Case Points

Mary Alice Bone
Stevens Institute of Technology
maryabone@gmail.com

Dr. Robert Cloutier
Stevens Institute of Technology
robert.cloutier@stevens.edu

Abstract. This paper proposes that as model based systems engineering begins to play a larger role in system development it could also be utilized in estimating systems effort, making model based systems engineering even more valuable to system engineering. This paper extends the Use Case Points (UCP) estimating approach used by some software engineers, to SysML for estimating the systems engineering effort. The paper reviews current practices in software engineering and systems engineering. It also reviews the literature that encourages the notion of a UCP for Systems Engineering (UCPSE) which uses UCP with SysML to estimate system effort as a viable and valuable method to the systems engineering community.

Introduction

The Object Management Group's Systems Modeling Language (OMG SysML™) is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems [OMG 2009]. In late 2009 the OMG issued a Request for Information (RFI), via an online survey. One significant finding from the survey was that a promising 91% of respondents were considering, planning or currently implementing SysML into their organization [Cloutier 2009]. It was observed though, from the survey results that many respondents and/or their organizations have not fully explored the benefits of adopting SysML into their organizations.

Since SysML is an extension (profile) of the Unified Modeling Language (UML), it is worth exploring many of the benefits the software engineering community has realized from UML [Dzidek 2008]. It is hoped that these benefits can be extended to systems engineering and the use of SysML. The purpose of this research is to focus on the use of SysML to improve the systems engineering estimating practice. The main research question "*Can systems modeling approaches be utilized to create an estimate of systems engineering effort for a given project?*" is proposed to possibly uncover one of the benefits of Model Based Systems Engineering (MBSE) (such as SysML). Specifically, the research will investigate the use of Use Case Points (UCP), similar to current software effort estimation approaches that are based on Karner's [1993] UCP approach using UML.

Topic Significance to System Engineering

The ability to design and build systems, especially complex systems, with a higher program success rate is a goal of both commercial and military industries [Ward 2010]. This research, if successful, would develop a method for using Use Cases developed with SysML to estimate the systems effort like the software industry utilizes UML effort estimation techniques, such as Use Case Points [Karner 1993]. The ability to accurately and precisely estimate the systems effort on a project is of extreme importance to the success of the overall project [Elm 2008]. Valerdi [2005] also noticed that accurate and precise effort estimation was of importance and sets out to answer how much systems effort is required for successful large-scale system projects in his dissertation. Valerdi [2005] went on to develop one validated system engineering effort estimation technique, Constructive Systems Engineering Cost Model (COSYSMO), to answer that question. COSYSMO has been further studied since 2005 [Valerdi 2007] and recently a revised model, COSYSMO 2 [Fortune 2009], was developed. It is worth noting that, COSYSMO is based on Constructive Cost Model (COCOMO), a functional point (FP) software effort estimation technique [Valerdi 2005]. The significance of the research proposed in this manuscript will be to identify a new approach to systems estimation that is based on the models developed during the early concept engineering and proposal phases of the project. This would represent a breakthrough in systems effort estimation that is directly derived from the systems engineering effort being performed. It is believed that this method could then be automated, and could be updated directly from the SysML models at any time. Ultimately, this would provide the systems engineering community with another cost estimation tool to better refine estimates. Another key benefit of this research would be the ability to provide systems engineering estimates of effort earlier in the project and with more accuracy than current methods if the results are similar to those found with software implementation of UCP [Carroll 2005].

Review of Current Practices

The review will be broken into three distinct areas: 1) Current System Effort Estimation Techniques; 2) UML Use Case Point Method; and 3) System Effort Estimation with SysML. These three areas combined cover the research topic at hand.

Current System Effort Estimation Techniques

The review of available system effort estimation techniques are few and thus emphasize the need for more research into the area. The majority of current system effort estimation literature is focused on COSYSMO, which began development in 2001 [Valerdi 2005]. Since that time Valerdi and others have published many articles on COSYSMO which summarizes the effort that has gone into the continuous refinement of COSYSMO. The research regarding COSYSMO is important to this research as it has uncovered many attributes needed for a successful system effort estimation model through lessons learned [Valerdi 2007]. The parallel between COSYSMO being developed from the software cost estimation tool COCOMO [Valerdi 2005] and the goal of this research to develop an effort estimation technique from the software estimation method Use Case Points (UCP) is notable. The success of COSYSMO only encourages this research topic and provides evidence that a useful effort estimate model for systems can transpire from a software cost estimation tool like COCOMO [Lane 2009].

UML Use Case Point Method

The Use Case Point (UCP) Method origins are credited to Karner [1993]. Since its conception in 1993 UCP has been studied and updates to the method have been published (see Figure 1). The most current UCP method, which does not even appear on Figure 1, and the one that will be used for reference in this research is UCP 3.0 [Frohnoff 2009]. The UCP 3.0 method has been selected for reference in this research as it has been shown, via calculation comparisons, to be a more accurate and precise method than UCP 2.0 [Heltewig 2008]. Much of the published research into the advancement of UCP has come from Capgemini sd&m AG, a consulting firm, in Germany [Capgemini 2009].

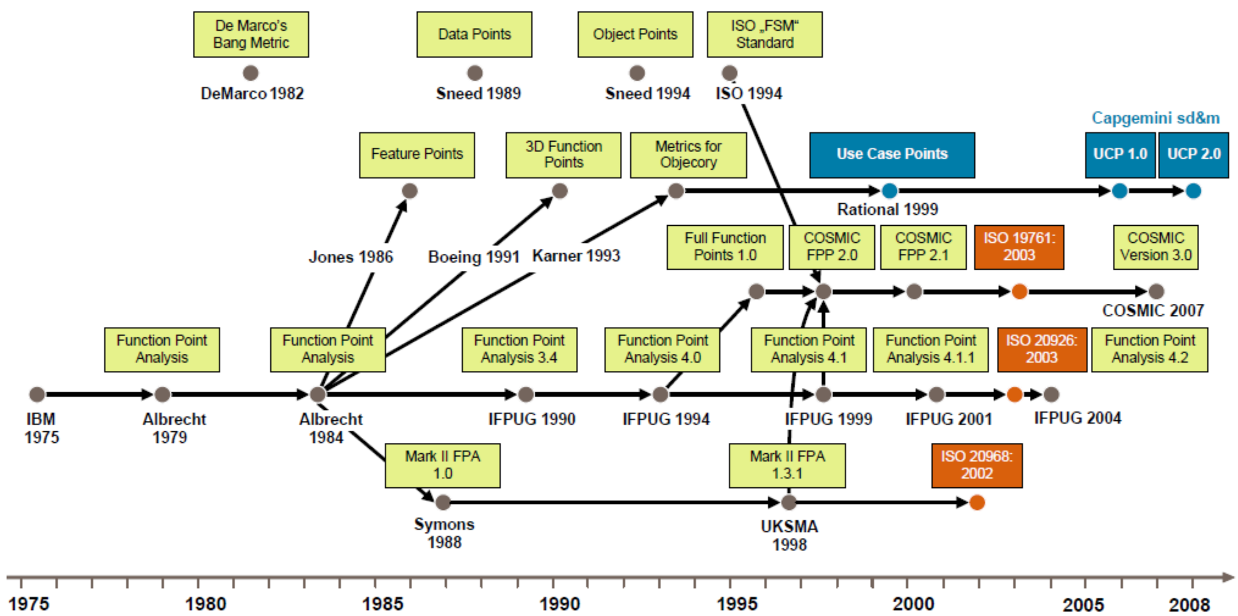


Figure 1 Evolution of Functional Size Measurement (FSM) Methods. [Heltewig 2008]

The evolution of UCP from its birth by Karner [1993] to its current state [Frohnoff 2009] will be of great importance to this research. The lessons that have been learned from the advancement of UCP will need to studied and understood for this research to be successful. The method developed in this research will be molded from the current UCP 3.0 [Frohnoff 2009] but past experiences and lessons noted from what the software community's development of UCP to UCP 3.0 will also play a significant role in developing the modified UCP method for systems engineering.

There are other variations of UCP. One such variation is the extended UCP (e-UCP) model [Periyasamy 2009]. This variation of Karners original model adds a detailed narrative of each use case and therefore uses a different set of calculations for the unadjusted UCP (UUCP) value. The current research into e-UCP has only been applied to work performed by graduate students and has not been applied to industry. To understand the pros and cons of this variation of UCP is also of interest to this research as it may shed light onto an improvement of the original UCP method.

Through the literature review UCP is well known but does not appear to be well utilized in industry. Vijay [2005] expresses that UCP has not became popular, although that was five years ago and some research has still gone into the effort of improving and studying UCP. Research has shown that UCP, if applied correctly, can be an acceptable method of estimating software effort [Barbosa da Silvia 2008][Heltewig 2008][Carroll 2005]. Further research should be conducted on why UCP has not been widely adapted in the software community. The reasons behind this lack of acceptance could hold key information needed by the research proposed in this paper.

Some of the reasons behind the lack of adoption of UCP have been hypothesized to be due to the fact that Use Cases, while widely utilized, are not consistently developed and therefore cause the UCP method to be less effective than other methods [Smith 1999]. Smith [1999] also goes on to describe other problems with the UCP method. Although the paper is dated 1999 the problems listed appear in other more current articles [Vinsen 2004]. It should be noted that many of the articles that discuss the issues with the UCP method appear to still be referencing Smith's 1999 paper and Karner's original UCP method. Most of the current UCP literature reviewed appears to be unaware of the newer versions of UCP such as UCP 3.0 [Frohnoff 2009]. Therefore this research needs to understand if the issues behind the lack of acceptance of UCP may be lessened by the recent updates made to the UCP methods (see Figure 1).

System Effort Estimation with SysML

SysML is fairly new to Systems Engineering, having version 0.9 released only in early 2005 and then version 1.0 released in late 2005. The 1.0 version was then updated in late 2008 to version 1.1, which is the current version to date (see Figure 2). The research regarding SysML has been limited in the five years since the release of SysML, with only 15 results found in the INCOSE i-Pub depository when "SysML" is used as the key word search. While a Google Scholar search finds many articles regarding SysML many of these are in regards to how SysML can be applied to systems and do not focus on why (benefits) SysML should be applied. Although one paper of interest to this research, by Soares [2008] describes a model driven approach to requirements engineering based on SysML Requirements and Use Case Diagrams. In his paper Soares [2008] discusses his perceived advantages with the method, using SysML, he puts forth but no data or references are provided to back up his opinions of those benefits. In his conclusion Soares [2008] only discusses future research regarding relationships between SysML Requirements diagram, Use Case Diagrams and other UML and SysML models the authors do not mention research to explicitly define the benefits of such diagrams. The research proposed here is complementary to the Soares' [2008] research. While Soares falls short of explicitly defining the benefits of SysML, Dzidek [2008] has proven the benefits of using UML through an empirical study. These findings by Dzidek [2008] are promising since SysML was birthed from UML; the hope is that the same or similar benefits to Systems Engineering can be achieved through SysML.

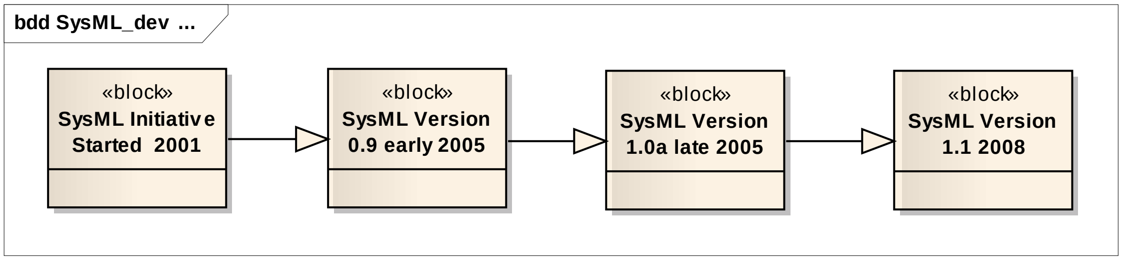


Figure 2 SysML Evolution

One doctoral dissertation [Adekile 2008] promised to have defined a SysML Point methodology to estimate effort the use of SysML in the methodology, but on further review, the work is UML centric. The reason for this discrepancy is unclear. Even though Adekile's [2008] research does not deliver on the promise set by the abstract it is still complementary to the research proposed in this paper.

The literature review has not uncovered published research that answers the proposed research question posed earlier in this paper. At least one SysML tool, Sparx Enterprise Architect (EA) has a function that allows one to perform effort estimation through Use Case Metrics [Enterprise 2009]. From the Enterprise Architect User Guide [2009] it is not clear what the effort estimation is based on or what the assumptions are so further investigation needs to be done. Grecki [2008] has a brief paragraph on how they used Sparx EA to estimate effort of both hardware and software. This brief paragraph along with the figure that displays all the factors one must define to make such estimates only encourages the proposed research topic, as Grecki [2008] explains the current method must be "tuned to obtain results close to reality", which is a future goal of this research. Additionally, Price Systems, a software cost estimating company, has incorporated UCP in their software.

Overview of the UCP Approach

The approach being taken is to first model the UCP in Microsoft Excel. Reed [2001] provided a good description for this model. The UCP approach requires a number of inputs, which include model information – Actor and Use Cases, System Technical Factors, and information regarding project team experience. Figures 3-5 represent potential input screens.

Actors		Weighting Factor	Number of Each	Extended Rate
Simple Actors: external systems		1	0	0
Average Actors: hardware devices or timers		2	0	0
Complex Actors: humans		3	0	0
		Actor Weight		0
Use Cases - all includes, extends, and generalizes		**Weighting Factor**	**Number of Each**	**Extended Rate**
Simple Use Case: 3 or less pathways		5	0	0
Average Use Case: 4 - 7 pathways		10	0	0
Complex Use Case: More than 7 pathways		15	0	0
		Use Case Weight		0
UUCP - Unadjusted use-case points				0

Figure 3: System Model Inputs

Technical Factors	Weight
Distributed System	0
Response Adjectives	0
End-User Efficiency	0
Complex processing	0
Reusable code	0
Easy to install	0
Easy to use	0
Portable	0
Easy to change	0
Concurrent	0
Security features	0
Access for third parties	0
Special training required	0
Technical Complexity Factor	0

Figure 4: System Technical Factors

Project Participants	Weighting	Rating	Extended Rate
Use of a formal process	1.5	0	0.0
Application experience	0.5	0	0.0
Object Oriented experience	1.0	0	0.0
Lead analyst capability	0.5	0	0.0
Motivation	1.0	0	0.0
Stability of requirements	2.0	0	0.0
Number of part-time workers	-1.0	0	0.0
Difficulty of programming	-1.0	0	0.0
Environmental Complexity Factor - ECF			0.0

Project Participants
For the first four factors, 0 means no experience in the subject, 3 means average, and 5 means expert.
For the fifth factor, 0 means no motivation for the project, 3 means average, and 5 means high motivation.
For the sixth factor, 0 means extremely unstable requirements, 3 means average, and 5 means unchanging requirements.
For the seventh factor, 0 means no part-time technical staff, 3 means average, and 5 means all part-time technical staff.

Figure 5: Project Participants Input

After a number of calculations, the output for the UCP approach might provide the following information (figure 6).

UCP (Use Case Points)	0.00
Person hours/UCP *	20
Estimated hours	0.0
Estimated hrs/week	38
Project Staff	15
Estimated Calendar Months	0.0

Figure 6: UCP Output

This is the starting point for this research. However, the approach needs to be expanded to take into account many of the tasks and responsibilities that fall into the realm of systems engineering to evolve this into a viable estimation tool. For instance, the effort required for the “ilities” must be added to the approach. Additionally, the assumptions for software effort – 20 hours/UCP for instance (figure 6) must be revisited. While that may be good for software, is it a valid number for systems engineering? Are there other technical factors that need to be taken into consideration?

Conclusion

The review of current practices has been very encouraging to the research proposed in this paper. The research topic has not currently been directly addressed and the key factors; system effort estimation and UCP have been dissertation topics within the last five [Valerdi 2005] and one year [Frohnoff 2009] respectively. The review of current practices in software engineering using UML has also shown encouragement that a Use Case Point Method utilizing SysML is possible and could be developed based on current UML UCP, SysML, and COSYSMO research and methods. As the research progresses, the intention is to work with the creators of COSYSMO to validate this approach, and to work toward an integration of the two approaches to provides a more tangible input to COSYSMO, using UCPSE.

References

1. Adekile, Olusegun. 2008. Object-Oriented Software Development Effort Prediction Using Design Patterns from Object Interaction Analysis. Dissertation Texas A&M University.

2. Aguiar, Mauricio. 2009. "Function Points or Use Case Points?" IFPUG MetricViews Summer 2009. http://www.scribd.com/doc/26509647/Function-Points-or-Use-Case-Points. Last accessed Feb. 22, 2010.

3. Anda, Bente. 2002. "Comparing Effort Estimates Based on Use Case Points with Expert Estimates". Empirical Assessment in Software Engineering (EASE 2002) April 8-10 2002. Keele, UK.

4. Anda, Bente, H. Dreiem, DIK Sjoberg, M. Jorgensen. 2001. "Estimating Software Development Effort based on Use Cases – Experiences from Industry", Proceedings of Fourth International Conference on the UML, pp. 487-504.

5. Barbosa da Silva, Caio, D. Loubach, and A. Marqus da Cunha. 2008. "Applying the Use Case Points Effort Estimation Technique to Avionics Systems". Proceedings 27^{th} Digital Avionics Systems Conference October 26-30, 2008.

6. Boehm, B.W.,C. Abts, A.W. Brown, C. Chulani, B. Clark, E. Horowizt, R. Madachy, D.J. Reifer, and B. Steece. 2000. Software Cost Estimation With COCOMO II, Prentice Hall.

7. Carroll, Edward. 2005. "Estimating Software Based on Use Case Points". 2005 Object-Orientated, Programming, Systems, Languages, and Applications (OOPSLA) Conference. San Diego, CA.

8. Cloutier, Robert and M. Bone. 2009. SysML RFI Analysis Result. http://www.omgsysml.org/. Last accessed Feb. 22, 2010.

9. Collaris, Remi-Armand, E. Dekker. 2009. "Software cost estimation using case points: Getting use case transactions straight". March 15, 2009. http://www.ibm.com/developerworks/rational/library/edge/09/mar09/collaris_dekker/

10. Diev, Sergey. 2006. "Use cases modeling and software estimation: applying use case points". ACM Software Engineering Notes, Vol. 31 Num 6, November 2006.

11. Dzidek, Wojciech, E. Arisholm, and L. Briand. 2008. "A Realistic Empirical Evaluation

of the Costs and Benefits of UML in Software Maintenance". IEEE Transactions on Software Engineering. Volume 34. No. 3. May/June 2008.

12. Elm, Joseph, Dennis Goldenson, Khaled Emam, Nicole Donatelli, Angelica Neisa, NDIA SE Effectiveness Committee. 2008. "A Survey of Systems Engineering Effectiveness – Initial Results (with detailed survey response data)". CMU/SEI-2008-SR-034. NDIA.

13. Enterprise Architect. 2009. Enterprise Architect User Guide. Sparx Systems. March 2009.

14. Fortune, Jared. 2009. Estimating Systems Engineering Reuse with the Constructive Systems Engineering Cost Model (COSYSMO 2.0). Dissertation. University of Southern California.

15. Frohnoff, Stephan. 2007. "Use Case Points Effort Estimation Based on Different Specification Formats". MetriKon 2007: Measurement for Software Professionals.

16. ___. 2009. Use Case Points 3.0 Implementierun einer Use Case bezogenen Schatzmehtode fur das Software-Engineering betrieblicher Informationssysteme. Dissertation. University of Paderborn.

17. Frohnoff, Stephan, G. Engels. 2008. "Revised Use Case Point Method – Effort Estimation Development Projects for Business Applications". In Proceedings of the CONQUEST 2008, 11th International Conference on Quality Engineering in Software Technology, Potsdam 2008.

18. Frohnhoff, Stephen; T. Engeroff. 2008. "Field Study: Influence of Different Specification Formats on the Use Case Point Method". In Proceedings of ISWM/Metrikon/Mensura 2008, Munich, Germany.

19. Gencel, Cigdem, L. Buglione, O. Demirors, and P. Efe. 2006. "A Case Study on the Evaluation of COMIC-FFP and Use Case Points". Proceedings of Smef 2006.

20. Grecki, Mariusz, et. al. 2008. "Application of SysML to design of ATCA based LLRF Control System." 2008 IEEE Nuclear Science Symposium Conference Record.

21. Heltewig, Sebastian. 2008. *Improving the Use Case Point Method*. Dissertation, Technische Universitat Kaiserslautern.

22. Karner, Gustav. 1993. *Metrics for Objectory*. Dissertation. University of Linkoping, Sweden.

23. Kusumoto, Shinji, F. Matukawa, K. Inoue, S. Hanabusa, and Y. Maegawa. 2004. "Estimating Effort by Use Case Points: Method, Tool, and Case Study". Proceedings of the 10th International Symposium on Software Metrics (METRICS '04).

24. Lane, Jo Ann. 2009. "Cost Model Extensions to Support Systems Engineering Cost Estimation for Complex Systems and Systems of Systems". 7th Annual Conference on Systems Engineering Research 2009.

25. Li, Jinhua, Z. Ma, and H. Dong. 2008. "Monitoring software projects with earned value analysis and use case points". Seventh IEEE/ACIS International Conference on Computer and Information Science. Pgs 475-480.

26. Lin, Ann Chih. 1998. "Bridging Positivist and Interpretivist Approaches to Qualitative

Methods". Policy Studies Journal. Vol. 26, 1998.

27. OMG. http://www.omgsysml.org/ . Last accessed Feb. 22, 2010.

28. OMG. 2008. OMG Systems Modeling Language (OMG SysML™) Specification. Final adopted specification. formal/2008-11-01.

29. Periyasamy, Kasi and A. Ghode. 2009. " Cost Estimation using extended Use Case Points (e-UCP) Model". International Conference on Computational Intelligence and Software Engineering 2009.

30. Reed, Paul R. Jr., Developing Applications with Java and UML, Addison-Wesley Professional, Boston, MA, 2001.

31. Smith, John. 1999. "The Estimation of Effort Based on Use Cases". IBM Rational Software.

32. Stowell, Frank, Donna Champion. 2002. "A unified mechanism for information systems definition in action". Systems engineering for business process change: new directions. Pgs 138-149. Springer-Verlag New York, NY, USA.

33. Soares, Michel dos Santos and J. Vrancken. 2008. "Model-Driven User Requirements Specification using SysML". Journal of Software. Vol.3 No. 6. June 2008.

34. Valerdi, Ricardo. 2005. *The Constructive Systems Engineering Cost Model (COSYSMO).* Dissertation. University of Southern California.

35. ___. 2006. academicCOSYSMO Users Manual: A practical guide for industry and government.

36. Valerdi, Ricardo, B. Boehm and D. Reifer. 2003. " COSYSMO: A Constructive Systems Engineering Cost Model Coming of Age". INCOSE 2003 Symposium June 29-July 3 2003, Washington DC, USA.

37. Valerdi, Ricardo, G. Roedler, M. Wheaton, J. Rieff and G. Wang. 2007. "Lessons Learned From Industrial Validation of COSYSMO". 17th INCOSE Symposium.

38. Vijay, J. Frank and C. Manoharan. 2009. "Initial Hybrid Method for Analyzing Software Estimation, Benchmarking and Risk Assessment Using Design of Software". Journal of Computer Science 5 (10): 717-724. 2009 Science Publications.

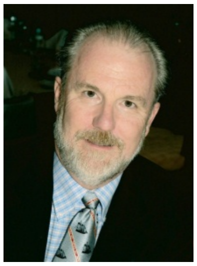
39. Vinsen, Kevin, Diane Jamieson, Guy Callender. 2004. "Use Case Estimation – The Devil is in the Detail". Proceedings of the 12th IEEE International Requirements Engineering Conference (RE'04).

40. Ward, Daniel. 2010. "Faster, Better, Cheaper Revisited: Program Management Lessons from NASA". Defense AT&L. March-April 2010. http://www.dau.mil/pubscats/ATL%20Docs/Mar-Apr10/ward_mar-apr10.pdf. Last accessed April 20, 2010.

Biography

Mary Alice Bone has worked as a Systems Engineer for GE Transportation, BAE Systems, and Rockwell Collins. She holds a B.S. in Aerospace

Engineering from the University of Missouri – Rolla and a M. Eng. in Systems Engineering from Iowa State University. She is currently a Research Assistant under Dr. Robert Cloutier at Stevens Institute of Technology where she is pursuing her PhD in Systems Engineering.



Dr. Robert Cloutier is an Associate Professor of systems engineering in the School of Systems and Enterprises at Stevens Institute of Technology. He has practiced systems engineering & architecting, software engineering, and project management in both commercial and defense industries for over 25 years. Industry roles included lead avionics engineer, chief enterprise architect, lead software engineer, and system architect on a number of efforts and proposals. His research interests include model based systems engineering and systems architecting using UML/SysML.