Utilizing DMAIC Process to Identify Successful Completion of SRAD Phases of Waterfall Development

Niamat Ullah Ibne Hossain, Alexandr M. Sokolov, Tim Petersen, and Brian Merrill
Engineering Management and Construction Management
College of Engineering and Computer Science Arkansas State University
State University, AR 72467, USA
asokolov@astate.edu, nibnehossain@astate.edu, Timothy.Petersen@smail.astate.edu,
brian.merrill1@smail.astate.edu

Abstract

The Systems Requirements Analysis (SRA) and System Architecture Design (SAD) (often combined into one acronym SRAD) phases of projects in the waterfall development cycle often pass-through design gates without proper pass/fail criteria. In addition, completion of project designs is often put off onto later design phases (Preliminary Design and Critical Design) in favor of meeting schedule/budget early in the project lifecycle. Currently in industry, schedule, and budget dictate project phase completion over proper metric tracking/utilization. This is normally due to the fluidity of the design in early phases of project development. This thinking can be dangerous for organizations/industries as it consistently leads to defects late in the development cycle where fixes are costly. It is cheaper to change designs and defects as early in a design as possible. This paper will outline the DMAIC (Define, Measure, Analyze, Improve, Control) process to help track completion of the SRAD phases for proper completion of design review phase gates. By using DMAIC, projects will also be able to reduce latent defects in designs that can become costly to projects in later design phases such as testing, and production. These phase gate completion metrics can be implemented and refined, as the DMAIC process is an ongoing methodology.

Keywords
DMAIC, Waterfall, SRAD, Systems Engineering, Project Management

1. Introduction
This Waterfall Development Lifecycle, often called the Vee Model, is a phased approach to project management. This model is broken into phases, where the phases are done sequentially. Where the next phase cannot be started until the current phase is complete. Completeness of two of these phases will be discussed in this paper. The original intent o of this model was for each phase to end in a design review, where completeness would be assessed. The phases of the waterfall development cycle are outlined in the Figure 1, and as follows: Systems Requirements Analysis (SRA), Systems Architecture Design (SAD), Preliminary Design Review (PDR), Critical Design Review (CDR), Test Readiness Review (TRR, also interchanged with Systems Verification Review or SVR), and Production Readiness Review (PRR). This model is maintained within the following two sources: (Walden et al.2015) and (ISO/IEC/IEEE, 2015). The first being the Systems Engineering Handbook by The International Council on Systems Engineering (INCOSE, and the second being the specification ISO/IEC/IEEE 15288:2015 by the International Organization for Standardization (ISO), The International Electrotechnical Commission (IEC), and the Institute of Electrical and Electronics Engineers (IEEE).

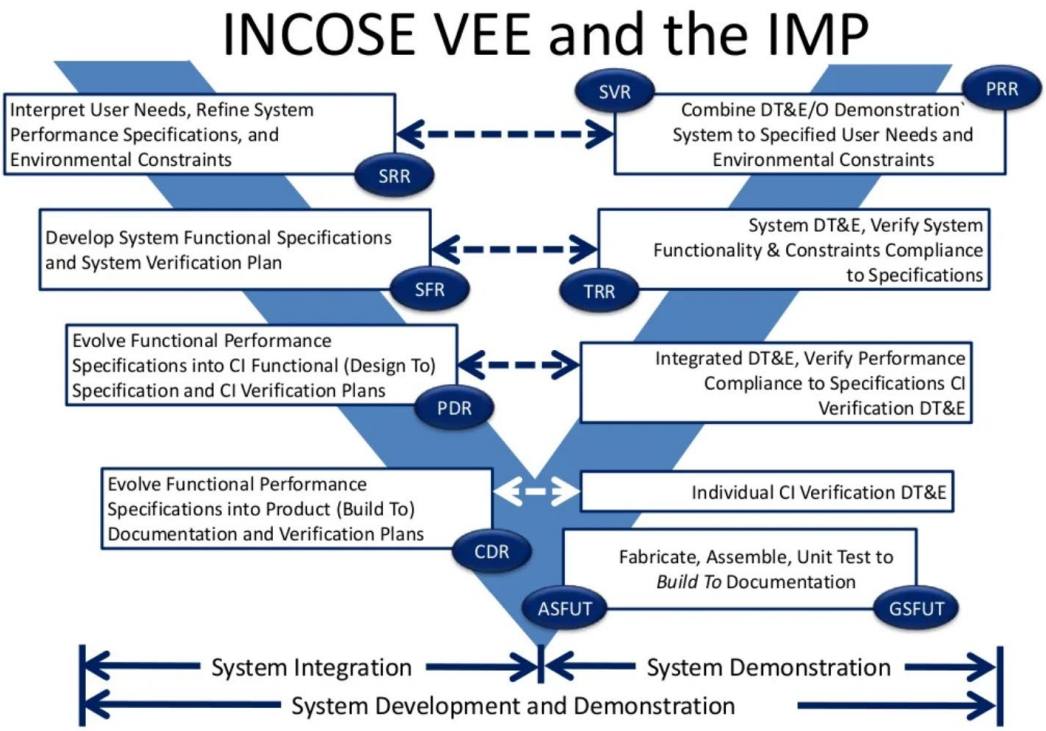

Figure 1. The Waterfall Lifecycle Model (Walden et al. 2015).

As shown by (Walden et al. 2015) in Figure 1. System design and integration are done on the left side of the V (SRR to CDR), while testing and demonstration is done on the right side of the V (ASFUT to PRR). The waterfall methodology has been in practice since 1970, first developed by (Royce 1970). Since then, this model has been put into practice in many industries, including but not limited to the automotive industry, construction, information technology (IT), and most notably the Department of Defense (DoD).

2. Literature Review

(Gilbert et al. 2014) suggests a quantitative process to monitor the state of projects to see if they become problematic. This was ill-defined as problematic was up for interpretation of the reviewers which ranged in experience levels, making each reviewer range in accuracy based on experience. More often than note in their experiment the reviewer ended the review unsure whether the project they were reviewing problematic (specifically 40% of the time). Furthermore, (Thesing 2021) has determined that one of the biggest problems with waterfall development is the difficulty to formulate all requirements in detail at the beginning of the process i.e., SRA. Creating a set of guidelines and metrics to make sure that all requirements are formulated during SRA to be iterated on during SAD, PDR, and CDR is crucial for project kickoff. (Touissant 2021) created their own model to track requirement elicitation. This model was adapted from SysML, ReqIF, and AP242, and requires the engineering team to track decisions, meetings, and team member responsibilities through object attributes and custom object classes. Although this possibly could solve some of the previously mentioned problems, this paper does not cite active use of the process, and in addition does not go into the time required to implement the process, as the engineering could be spending more time implementing a process, rather than working on the project design and requirements. Instead of going over project completion metrics, many papers such as (Hickey & Davis 2004), (Davis 1998), (Kotonya et al. 1999), (Macaulay, 1996), and (Maiden & Rugg 1996) have been found on the topic of requirement elicitation process, more specifically when a process should or should not be implemented

3. Methods

After the kickoff of the project, the first phase of waterfall development is the Systems Requirements Analysis process. Since this is the first phase of the project, there are no inputs to this project. Although there are no inputs to this process for systems engineering the project team should have in place a contract with the customer in addition to a

scope of work (SOW). Additionally, the customer often provides customer functional requirements as well. From this information, the systems engineering can go through requirement elicitation with the customer. Requirement elicitation requires the staff to go through requirement discovery with the customer using technical interchange meetings (TIMs). These meetings provide the team clarity to the intent of the provided requirements. In addition, these meeting go through whether requirements provided were a must-have (often indicated by the word shall in the requirement), or a want (indicated by the word may or will within the requirement).

Once customer requirements intent and considerations are considered the systems team can complete the concept of operations (CONOPS) documentation. This process usually breaks down the product into functional use cases for the team to document how those use cases would affect the system and how data would be processed. After the CONOPS is completed, the customer requirements can be derived into system requirements. These requirements are often maintained in a requirement management tool such as IBM DOORS or documented in a System/Subsystem Specification (SSS). The SRA phase is complete with a review that goes over the final outputs which are system requirements and concept of operations documentation.

The systems architecture design phase starts at the conclusion of a successful gate review of the systems requirements analysis phase. The inputs to this phase are the outputs of the previous SRA phase. Documentation included for the input are system-level requirements (which were derived from the CONOPS and customer requirements), and the concept of operations. Using those documents, the systems team can then start designing the architecture. The architecture would need to be broken into subsystems. Those subsystems would then need to be classified as a hardware or software configuration item (CI)/subsystem.

This design process must be collaborative. Ideally the systems team has already gotten input from other disciplines during the requirement phase of the program. Those requirements will still be in flux, during this phase and ongoing design phases, as the customer feedback rolls in, and the design changes. Outputs of this design phase would be an architecture description document. Architecture designs can be captured in a Systems/Subsystem Design Description (SSDD). Additional information such as interface descriptions should be captured in additional documentation such as an Interface Control Document (ICD). Once appropriate documentation is finalized and requirements are iterated on, the team can go into a design review. Upon successful completion of the design review, the project team goes into the preliminary design phase.

For the industries that still use the waterfall methodology, a constant pain point is how to know when a project has completed the design phase. Notice how previously in this paper, it was described how the design review was met when appropriate documentation was met. This is ambiguous. Often in industry, cost and schedule is of more concern early on in a program then design completion. This is because since the program is so early in development, program/project managers think this can be time/cost can be made up later in the project lifecycle. This is a fatal flaw.

It is most cost-effective to address design defects early in the program. By letting defects linger, unnoticed by an incomplete design, or an incomplete review can be costly to a program. (Widerman 2001) shows this clearly in Figure 2. While in the design phase, it is easy for a project to pivot, change designs and documentation and adapt at relatively low cost to schedule and budget compared to the production phase. Take for example a project that must design and produce a piece of custom electrical equipment like a PCB. If a defect is found in the PCB design after it has been produced, this will be more costly to fix, as the project will need to order more materials, put in more hours for defect resolution, as well as manufacturing and testing the new PCB. It would be much easier, simpler, and cost-effective to make sure the design is complete while still in the design phase.

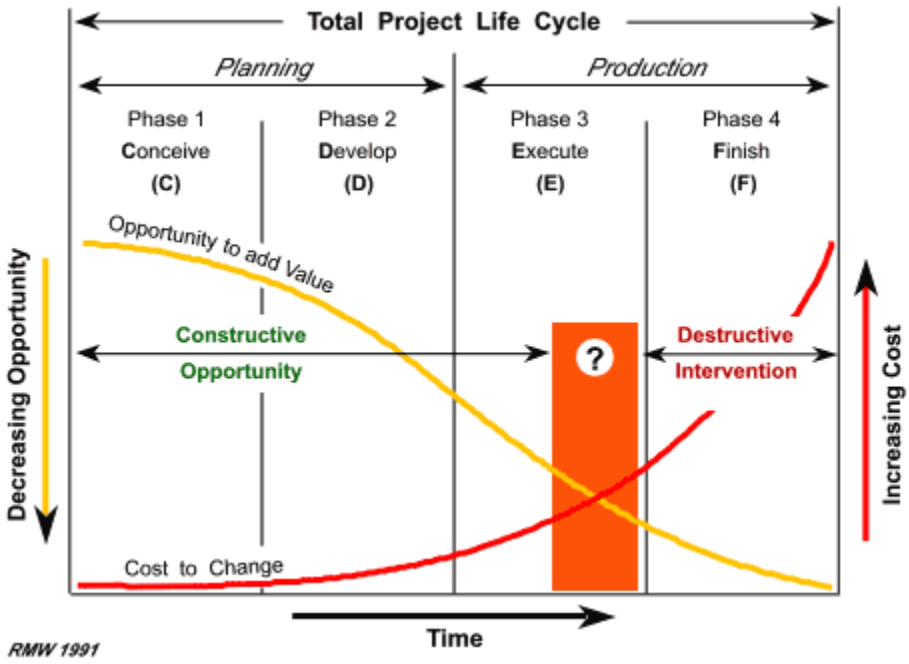

Figure 2. Cost of Defects Throughout Project Lifecycle (Widerman, 2001).

4. Data Collection

Using DMAIC (Define, Measure, Analyze, Improve, Control) a project team can better utilize project management tracking of the design. Cost and schedule management have been well-researched and defined. Tracking maturity of the design is complex. Therefore, it is best to utilize a process control method such as DMAIC. This way organizations can better track, analyze, and most importantly control their status for multiple projects to best determine what is and is not working for their specific organization. For this exercise, DMAIC will be used to best understand metrics to be utilized within the SRAD process, to determine design maturity completeness for design review passage.

SRA is iterative, but it is difficult to know when to stop iterating in the requirement creation process. Not iterating on the requirements enough yields an immature design, which can later turn into costly defects. Iterating to much, can lead to schedule overrun, and overspending for the budget, but yielding a much more complete design. A survey completed by (Jacob 2017) concluded that 37% of companies surveyed indicated that poor metrics were the leading cause of project roadblocks. To combat these problems projects should define the work products they are responsible for creating at the start of the project, and revisit this during the beginning of each phase. This will set expectations for the project team prior to starting the next phase worth of work. Defining work outputs will be useful for the end-of-phase review in the control phase to keep programs honest.

While creating requirements, it is essential to measure completeness by a few metrics. Built into most requirement management tools are traceability analysis tools. Using this tool engineers can track how many customer requirements are yet to be derived into system requirements. In addition, this can be tracked in the reverse direction. By checking that every systems requirement has a parent customer requirement traced to it, the engineering team can verify that they are limiting scope creep. Scope creep occurs when extra requirements are added to the design that are not required by the customer. For example, if the customer requires the input voltage to compatible with 120VAC standard electrical systems for residential US households, and the project team makes a device that is compatible with US electrical standards and EU electrical standards of 220VAC, then that would be scope creep, since the 220VAC is not required.

To track completeness of design, some projects create custom excel sheets, as seen below in Table1.

Table 1. Custom SRA Completion Excel Sheet.

	Concept of Operations	Requirements			
Use Case 1	100%	95%		Completion Status	Percent Complete
Use Case 2	5%	0%		Working	5%
Use Case 3	5%	0%		Drafted	33%
Use Case 4	95%	95%		Reviewed	95%
Use Case 5	33%	5%		Submitted	100%
Use Case 6	33%	0%			
			CONOPS Completion	45%	
			Requirement Completion	33%	
			SRA Completion	31%	

Using something like the above excel sheet, the project team can better track SRA completion in terms of the design. Although the above excel sheet tracks design completion only, and not document completion. While creating requirements, a verification method among other attributes such as risk, will be necessary to track, so measuring completeness of attributes besides the requirement text will also be helpful for requirement elicitation. While also measuring the completeness of the creation of systems requirements, the team can also review and track proper requirement creation. Proper requirements are requirements in the format defined by (Grady, 2006). Proper requirements include a subject, a shall verb, a sentence ending, and a unique paragraph title and number. (Grady 2006) gives the example of turning the requirement: Closure time less than or equal to 0.5 seconds into Valve closure time shall be less than or equal to 0.5 seconds.

In addition, requirement management tools usually also provide tools to track requirement volatility. This can be understood as the number of times a requirement has been changed between baselines. Highly volatile requirements should be iterated to develop a higher level of understanding for the team. It is normally helpful to analyze requirements in groups by use case, or functionality to make sure the proper stakeholders are involved in addition to burnout prevention

Since the SRA phase is iterative, it is important to improve on the requirements during each iteration. A simple approach to iteration was shown in Table 1. The iterative stages being shown as Working, Drafted, Reviewed, and Submitted. Working means it has been assigned to someone to complete, and not in the backlog of work to be done. Drafted does not have to be the first draft, but that an ongoing draft is being done/complete. Review is the process in which the team reviews the work and provides feedback along with comments. Lastly submitted means submitted for baselining to maintain a controlled version of what is being worked on. Improving the metrics and process are routinely evaluated within the control phase. This should be done by constant lessons-learned meetings at the end-of-phases, and knowledge transfers within the organization.

5. Results and Discussion

For a large organization controlling the process is key, especially across programs and maintaining standardization. (Pradham & Nanniyur 2021) cites standardization as a major cause preventing repeatability, and ill standardization makes the analysis of the metrics taken to be troublesome, as the process might not correlate well. One portion of the control process should take place during the end-of-phase review. This review should keep the team accountable for the phase outputs they originally defined in the beginning of the phase (and within the define phase of the DMAIC process). It is worth noting that without senior management buy-in, reviewers will not be able to hold projects accountable. Project management can continue to the next phase of the project without proper action and control from management. Management must also not be afraid to put complete designs as a priority over budget and schedule.

This is often embedded in the ethics of an organization, as a good organization would not want to produce an incomplete product.

In addition, the organization should keep control between a multitude of projects to maintain the statistics of defects found in the reviews, and especially categorize those defects, to home in on specific design problems, and use the I within DMAIC to Improve the process, and Measure change. Minor defects can be things such as spelling errors, missing attributes, bad requirement wording, missing shall verb. Major defects are defects that require functional changes to the architecture or increased/missing functionality. Functionality can often be done via requirement traceability in the requirement management tool.

5.4 Validation

During this paper, the topics of project management for the systems engineering development lifecycle phase of Systems Requirement Analysis were delved into. Regardless of industry, management of engineering teams and projects are within scope. Engineering management is agnostic to disciplines within engineering whether that be software, hardware, quality, manufacturing, or in this case systems engineering. Managing the requirement elicitation and pushing back against budget and schedule in favor of architecture/project design is something every manager should prioritize.

6. Conclusion

One of the major problems a waterfall project faces starting out is moving through early design gates with an incomplete design. This is a preventable problem. As discussed, ending the SRA and SAD phases early can lead to defects showing up later in the design process where it is more costly to fix, as proved by (Widerman, 2001). To prevent defects embedded in the design architecture metrics must be kept, by utilizing preexisting infrastructure within most requirement management tools. By iterating and tracking CONOP and requirement progress and iterations, project teams can better understand their progress within a waterfall phase and be less susceptible to early phase completion from budget, or schedule pressures.

References

Davis, A. M., A comparison of techniques for the specification of External System Behavior. *Communications of the ACM*, *31*(9), 1098–1115, 1988.

Gilbert, D., Yearworth, M., & Oliver, L., *Systems approach to the development and application of technical metrics to systems engineering projects.* Procedia Computer Science, 28, 71–80, 2014.

Grady, J. O., *System Requirements Analysis.* Academic Press, 2006.

Hickey, Ann M., & Davis, Alan M. (2004). A unified model of requirements elicitation. *Journal of Management Information Systems*, 20(4), 65–84. https://doi.org/10.1080/07421222.2004.11045786

ISO/IEC/IEEE., *Systems and software engineering -- system life cycle processes.* Geneva, Switzerland: International Organisation for Standardisation / International Electrotechnical Commissions / Institute of Electrical and Electronics Engineers, 2015.

Jacob, D., Quality 4.0 impact and strategy handbook: getting digitally connected to transform quality management, 2017.

Kotonya, G., & Sommerville, I., Requirements engineering with viewpoints. *Software Engineering Journal*, *11*(1), 1996.

Macaulay, L. A. , Requirements engineering. *Applied Computing*, 1996.

Maiden, N. A. M., & Rugg, G., Acre: Selecting methods for requirements acquisition. *Software Engineering Journal*, *11*(3), 183, 1996.

Pradhan, S., & Nanniyur, V., Large scale quality transformation in hybrid development organizations – A case study. *Journal of Systems and Software*, *17,* 2021.

Royce, W. W., Managing the development of large software systems: concepts and techniques. In Proceedings of the 9th international conference on Software Engineering (pp. 328-338), 1987.

Thesing, T., Feldmann, C., & Burchardt, M., Agile Versus Waterfall Project Management: Decision model for selecting the appropriate approach to a project. *Procedia Computer Science*, *181*, 746–756, 2021.

Toussaint, M., Krima, S., Feeney, A. B., & Panetto, H., Requirement elicitation for Adaptive Standards Development. *IFAC-PapersOnLine*, *54*(1), 863–868, 2021.

Walden, D. D., Roedler, G. J., Forsberg, K., Hamelin, R. D., & Shortell, T. M., *Systems engineering handbook: A guide for system life cycle processes and activities,*2015.

Wideman, Max., Project Management Simply Explained A Logical Framework to Help Your Understanding, 2001.

Biographies

Niamat Ullah Ibne Hossain, PhD., is an assistant professor in the Department of Engineering Management at Arkansas State University. Dr. Hossain received his PhD in Industrial and Systems Engineering in 2020 from Mississippi State University (MSU), Starkville, MS. He obtained his bachelor's degree in Mechanical Engineering in 2010 from Khulna University of Engineering and Technology (KUET) and his MBA in Management Information Systems in 2013 from the University of Dhaka, both in Bangladesh. His main research interests include systems engineering, systems model-based systems engineering/SysML, Systems thinking, Systems resilience, & sustainability management, System dynamics, and systems simulation.

Alexandr M. Sokolov, Ph.D., is a faculty in the Engineering Management Department of the College of Engineering and Computer Science at A-STATE. He holds a B.S., where he focused on Bioinformatics from the University of Tennessee Knoxville, an M.B.A., in Finance from Lincoln Memorial University, and a Ph.D., in Industrial Systems Engineering, Engineering Management from the University of Tennessee Space Institute. Alexandr has over 15 years of infield and teaching experience. His teaching experience includes multiple institutions dealing with Engineering, Management, and Technology disciplines. He is focusing on research dealing with Engineering Management, Performance Management, and Interdisciplinary Studies.

Tim Petersen; is a graduate student in the Management Department of the College of Engineering and Computer Science at A-STATE and currently pursuing the Master's in Engineering Management. He has previously been employed at the Naval Air Weapons Station in Lakehurst, New Jersey, Norfolk Southern in Fort Wayne, Indiana, L3Harris Technologies in Camden, New Jersey, and most recently at Apex Systems in Moorestown, New Jersey. Tim received his Bachelor of Science degree from Rutgers University New Brunswick in Electrical & Computer Engineering.

Brian Merrill; is a graduate student in the Management Department of the College of Engineering and Computer Science at Arkansas State and currently pursuing the Master's in Engineering Management. He is an Electro-Mechanical Engineer at a Pharmaceutical Manufacturing Facility located in North Las Vegas, Nevada. Holds a B.S., in Electrical Engineering from Nevada State College, with minors in Mechanical Engineering, and Biomedical Engineering and a certificate in Engineering Management from the universities engineering department.