Concept Maturity Levels

Article · September 2016

DOI: 10.1002/j.2334-5837.2018.00570.x

2 authors, including:

Louis S Wheatcraft
Wheatland Consulting LLC
19 PUBLICATIONS **47** CITATIONS

Concept Maturity Levels

Louis S. Wheatcraft
(281) 486-9481
louw@reqexperts.com

Layne Lewis
208-250-0211
llewis@willowviewconsulting.com

Abstract:

In this paper the authors introduce and expand on a method for organizations to access the maturity of a system concept referred to as Concept Maturity Levels (CMLs). In the first part of the paper, we introduce the benefits and need for organizations to include CMLs in their systems engineering processes. We then go into more detail defining and explaining the activities and outcomes for each CML. Next we show how CMLs are an implementation of an older concept referred to as "The Doctrine of Successive Refinement. Tools to help implement, manage, and mature the project through the various CMLs are presented, followed by a discussion on applying CMLs to other product development approaches including incremental and spiral development. In the final section, we discuss the need for organizations to establish a collaborative work environment to help concept maturation and design teams to quickly mature system concepts through the early CMLs.

Introduction:

In a previous blog, Lou discussed the need to define and baseline your scope before writing requirements. In that blog, he discussed how organizations can use a Scope Review, defined success criteria, and assessment of risk to determine whether or not the project is ready to proceed to the next product development lifecycle phase. (The Scope Review may also be referred to as a Scope Baseline Review, System Concept Review, Mission or Project Concept Review, etc.).

In this paper we introduce Concept Maturity Levels (CMLs) as a way of measuring the maturity of a project's system concept for all gate reviews defined by an organization, including the Scope Review.

Some projects may have their funding approved from the beginning, but must pass specific gate reviews at the end of each phase of product development. When this is the case, the organization defines specific entrance criteria that needs to be met to perform the review, as well as success criteria that the project has to demonstrate have been met before passing the gate review and proceeding to the next lifecycle phase.

Other organizations may use a two-step, knowledge based process for funding projects. Step 1 involves the submission of new project ideas and proposals to obtain funding and resources needed to mature the system concept to a specified level. Step 2 projects that successfully accomplished the activities associated with Step 1 are evaluated for continuation in terms of concept maturity, feasibility, risk, and return on investment (ROI). Based on this evaluation, additional funding required to complete development of the proposed system is approved.

(Note: I am using the term ROI generically to refer to the benefits to the organization associated with a given project.)

With both funding approaches, a dilemma faced by many program/project managers when deciding whether to fund a project or schedule a gate review is how to:

- evaluate the feasibility (cost, schedule, technology) of the proposed system concept (project, product, system, mission) and its fulfillment of the project's Need, Goals, and Objectives (NGOs) and stakeholder expectations within the defined drivers and constraints,
- access whether or not the project will deliver an acceptable ROI with acceptable risk,
- determine if **the maturity of the concept,** critical technologies, available resources, and associated planning are sufficient to approve the project for additional funding or conduct the gate review, baseline the deliverables associated with the review (scope, requirements, design, project and technical plans), and proceed with the next phase of product development.

The answer? "Concept Maturity Levels (CMLs)". We were recently made aware that Mark Adler at NASA's Jet Propulsion Laboratory (JPL), had developed CMLs as described in a paper you can access at: http://hdl.handle.net/2014/44299.

JPL introduced the idea of CMLs so that system architects, proposal teams, and system engineers have a way to measure, assess, and communicate the fidelity, accuracy, technical progress, and maturity of a system (science mission) concept through the system's lifecycle.

Note: The CML definitions and other descriptive information in the JPL paper is very domain specific in terms of NASA science space missions. In this paper we have revised and expanded the definitions to be more generic and applicable to other domains. Added to the CML definitions are Technology Readiness Assessments as described in the recently released GAO Technology Readiness Assessment Guide. We have also aligned the CMLs to be consistent with the scope definition and requirements development activities we present in our classes and seminars and defined in NASA's NPR7123.1B, Systems Engineering Processes and Requirements. While for NASA space missions the traditional waterfall development approach is used, we have defined CMLs in this paper so they are applicable to other development approaches including incremental, spiral development, rapid prototyping.

As shown in Figure 1, CMLs are expressed in a scale from 1-9 that provides a repeatable way to assess and describe the maturity of system concepts and provides a single numerical scale, comparable to the Technology Readiness Level (TRL) scale, to document the maturity of different system concepts. (To learn more about TRLs, see Lou's blog "Using Technology Readiness Levels to Manage Risk".)

CMLs provide a standardized method to allow management to:

1) determine how much work (funding, resources, and effort) has been placed into the definition and maturation of a system concept;

2) compare competing project system concepts in terms of relevance to meeting the organization's strategic goals, objectives, and ROI with acceptable risk;
3) determine which system concepts have had the same level of work and can be compared on the same terms;
4) understand the maturity of critical technologies needed to meet the project's goals and objectives,
5) understand how much future work a system concept will require to get to a subsequent level of maturity; and
6) have the information needed to determine when a proposed project's system concept is mature enough to proceed to the next stage of system development.

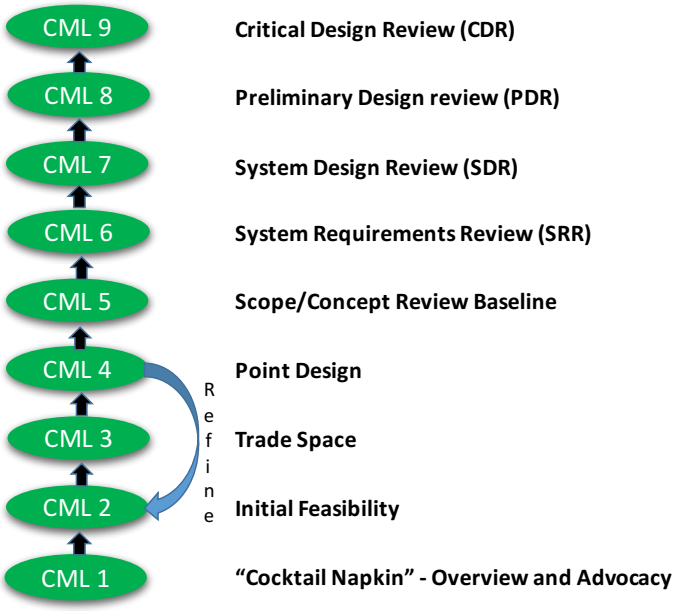


Figure 1: CML summary

The CML organizing structure corresponds to an increasing level of maturity as the system concept, planning (project and technical), design, architecture, and risks are analyzed and evolve. CML(s) provide the ability to measure a system concept's maturity guided by an incremental set of maturity criteria. This defined maturity criteria can be tailored to correspond to the processes specific to a particular organization, domain, and project within that domain.

Concept Maturity Levels Defined (tailored to be domain independent)

The CML rating scale shown in Figure 1 is expanded below. The idea for the CML scale is to address the common path of progression through the project initiation phase from initial idea through Critical Design Review (CDR) or a comparable gate review. Varying levels of concept maturity may entail differing levels of fidelity of project planning and engineering analysis

needed for broader vs. more localized trade studies, and varying techniques for cost, schedule, and risk analysis.

CML 1 - "Cocktail Napkin"- Overview and Advocacy - used to launch a new concept. In the overview, the problem (or opportunity) is defined, communicating why this project is worthwhile. (For more details on defining the problem, see Lou's blog: "What problem are you trying to solve?") The essence of what makes the concept unique and meaningful is clearly stated. The project vision and "Need" statement is defined showing the top level reason for developing the system and providing a focus for the team. The overview lays out a broad outline of the intention of the project, high level benefits, overall advantages, but provides limited details of cost, schedule, or other programmatic or design information. Think of the overview as the "30 second" elevator speech addressing what the problem being solved or opportunity being pursued, is and what the Need is – in other words, the desired end result at the end of the project once the system is operational.

From an advocacy viewpoint, information needed to sell the idea is documented. Key stakeholders are identified, the benefits, advantages, and ROI that result from pursuing this concept are defined for the stakeholders and for the organization. Needed capabilities are defined in terms of people, process, and products along with what needs to be done to provide these capabilities. A rudimentary "back of the napkin" sketch of the system concept is developed.

The advocacy perspective expands on the overview, going into more detail as to both what needs to be done to meet the Need and how management will know if the project is successful. Preliminary goals and objectives are defined including preliminary measures of effectiveness (MOEs) and Key Performance Parameters (KPPs) including preliminary key figures of merit concerning safety, reliability, sustainability, affordability, extensibility/evolvability, agility/robustness, effectiveness, and innovation.

Assumptions are documented and system level drivers and constraints are identified including cost, schedule, standards, regulations, technology, resource availability, interaction with existing systems, and higher level requirements allocated to the system of interest.

CML 2 - Initial Feasibility: The system concept is expanded and assessed on the basis of feasibility (cost, schedule, technology) from ROI, technical, and programmatic viewpoints. Preliminary system life-cycle concepts are documented including concepts for development, procurement, system verification, system validation, manufacturing, transportation, deployment, operations, maintenance, upgrades, and disposal. Enabling systems required during development and operations are identified.

The system concept is expanded to address how the proposed system will fit within the macro system of which it is a part. A preliminary functional architecture of the macro system and system of interest is developed. How the systems that make up the macro system are related (connectivity, interaction, and flow of information) to each other and to

the system of interest is defined. External interfaces are identified. High-level schedule, resource, and cost estimates are developed. Key risks are identified.

Outside influences the system needs to take into account are addressed. Need, Goals, and Objectives (NGOs), MOEs, and KPPs are refined based on the information gained at this level.

CML 3 - Trade Space: The system functional architecture is transformed into an initial physical architecture. This physical architecture is used to access the options of the system to be developed meeting the NGOs within the defined drivers and constraints.

The study/design team will perform architecture trades between the system, subsystems, support systems, critical technologies, and enabling systems with elaboration to better understand the relationship between the parts of the architecture and impacts to ROI, cost, schedule, and risk.

The system (product, device, application, etc.) concept is demonstrated to be viable within the defined drivers and constraints, technology maturity, proposed macro architecture, and operating environment.

Objectives, MOEs, and KPPs are refined based on the information gained at this level. Measures of Performance (MOPs) are defined. Key risks are investigated, updated, and possible mitigation strategies outlined.

A key consideration at this stage is understanding the partials, that is how ROI changes as a function of some system parameter (e.g., mass, data volume, communication bandwidth, computing power, operating environment, technologies used, etc.). How will the ability of the system to meet the NGOs, MOEs, MOPs, and KPPs be affected as a function of changes in cost, schedule, technology, or other key parameter? How will the cost , schedule, technology be effected by a change in an KPP?

For complex systems where many of the parameters across the system architecture have a dependency (directly proportional or inversely proportional) it is very hard to track and understand the partials “manually”. For complex systems, we recommend the study/design team use Systems Engineering (SE) software tools to develop a model of the system including the subsystems that make up the system as well as the macro environment in which the system will operate. With such a model, the team can make changes to the MOPs and KPPs as well as external constraint parameters to access the impact of the change to meeting the NGOs and other key system parameters. This knowledge will be of great benefit in choosing or optimizing the system architecture within the defined drivers and constraints. (See Lou’s blog on Model Based Systems Engineering (MBSE): MBSE and Requirements.)

Another key consideration when performing architectural trade studies is the TRL of the candidate parts of the architecture. Trade studies will guide which technologies are needed to best meet the project’s objectives. Critical technologies needed to meet the goals and objectives, MOEs, and KPPs are identified. An initial Technology Readiness Assessment

(TRA) is performed per the processes outlined in the GAO *Technology Readiness Assessment Guide* and the resulting TRA Report is generated. If the current capability to meet an MOP or KPP is beyond current state-of-the-art for that part, then new parts/technology with the potential to provide the desired capability will have to be considered. Best practices and guidance from the Government Accountability Office (GAO) state that the system architecture should not rely on parts/technology with a TRL less than 3 at the time of scope baseline (CML 5) and a plan needs to be in place to mature the technology to at least TRL 6 by the time of the Preliminary Design Review (CML 8) and TRL 7 by the time of the Critical Design Review (CML 9). For more information on TRLs, see Lou's blog "Using Technology Readiness Levels to Manage Risk".)

CML 4 – Point Design - Architecture selected within Trade Space: Candidate system architectures are identified that will implement the system concept selected to meet the project NGOs and achieve the desired ROI providing the needed capabilities, functionality, performance, and quality within the specified drivers, constraints, and trade space. These architectures are defined to the level of major subsystems with acceptable cost, schedule, risk, and resource margins and reserves. (For a detailed discussion on the importance of defining development and operational margins and management reserves at this stage of the development lifecycle, see Lou's four-part blog: Resource Margins and Reserves.)

Subsystems' trades are performed. Descope and backup options are defined. The objectives, MOEs, MOPs, KPPs, cost, schedule, and risks are further refined based on the information gained at this level.

Operational scenarios and/or use cases are defined (nominal, alternate nominal, and off-nominal) and the system concept and candidate architectures are assessed based on their ability to successfully address each of the operational scenarios and/or use cases. Stakeholder roles and functions are defined and how various stakeholders (operators, maintenance and software update personnel, etc.) will interact with the system during operations are documented. The focus in on roles and functions – what the stakeholders do with and how they will use the system once it is operational. A "day-in-the-life" of the system from the stakeholder's viewpoint is addressed. How does data and information flow between architectural elements? How are the capabilities of the overall system used to accomplish its intended purpose in the operational environment?

Organizations may use "rapid prototyping" of potential physical solutions of a single component, subsystem, or system. With advances in 3D printing, study/design teams can develop prototypes quickly to be used as part of the architectural trade activities. When prototypes are developed, various configurations can be made available to the actual users. These prototypes can be functional, non-functional, or just "form and fit". A prototype is useful to help the study/design team understand the user needs and expectations which will drive the system requirements. Prototypes are developed based on the currently known requirements. By using prototypes, the users can get an "actual feel" of the system in representative operational environments.

Also referred to as “discovery learning”, user interactions with the prototypes can enable both the user and study/design team to better understand the requirements for the desired system. Rapid prototyping allows for user feedback much earlier in the development lifecycle concerning errors and problems, such as missing functions, substandard performance, safety issues, and confusing or difficult user interfaces. This is an interactive process that allows users to understand, modify, and eventually approve a working physical model of the system that meets their needs.

From the operational scenarios and user interactions with system prototypes, a draft set of system requirements are developed including core functional requirements and associated performance requirements, requirements to implement defined objectives and MOPs, requirements addressing defined drivers and constraints, and system level interface requirements.

The TRA and resultant TRA report are updated. Based on the results of the TRA a Technology Maturation Plan (TMP) is created per the processes outlined in the GAO *Technology Readiness Assessment Guide* to mature the critical technologies needed to meet the goals and objectives, MOEs, and KPPs.

Based on the knowledge gained to this point and the activities completed, the study/design team needs to determine whether or not they are ready to proceed with the activities associated with CML 5 or go back and repeat activities associated with CML 2-4 to further refine the system concept.

Note: *it is a myth that design work shouldn’t start until after the scope and requirements are baselined. A key outcome of the Scope Review (CML 5) is that there is at least one feasible concept that will result in the NGOs being achieved within the given drivers and constraints with acceptable risk. Lessons learned from key programs indicate that design work during the early concept maturation phases can identify important issues that are difficult to determine until a design concept is of sufficient maturity to evaluate a more complete set of development, test, operations, and evaluation considerations. Using the knowledge gained from prototypes and trade studies, a design concept should be developed, as early as possible in the project formulation phase, to allow a more complete understanding of programmatic implications and development challenges and risks. The design work at this stage enables validation of the draft requirements, providing analysis to determine where something is missing or perhaps not needed. This validation is a key part in accessing feasibility of the concept. Prototyping results in a system concept that is much more than just an artist rendering or a parametric model.*

CML 5 – Scope/Concept Review or analogous gate review Baseline: The implementation approach is defined including project management, systems engineering, contracting mode, integration, system verification and system validation approaches, cost and schedule. Relationships and dependencies, partnering, key risks, mitigation plans, and system “buy (from an external source), build/code (make internally), reuse/modify (existing systems)” or buy/try/decide approaches are defined. The TRA is completed verifying a minimum of TRL 3

for all parts of the architecture and a plan for advancing the TRLs to at least TRL 6 by PDR is complete, for systems going operational. (The purpose of some projects may be to advance the TRL. In this case advancing to TRL 6, may be the goal of the project.) A viable, feasible, and stable system concept is defined that will meet the NGOs, MOEs, MOPs, and KPPs within the defined drivers and constraints with acceptable risk.

The proposed system concept along with operational scenarios/and or use cases is documented in a system concept document or model. Project management and systems engineering documentation is developed to the maturity required by the organization for this lifecycle stage of the product development. The TMP is baselined. (For a more detailed discussion on baselining scope, the entrance criteria, exit criteria, and risk identification, see Lou's blog: Baseline Your Scope Before Writing Requirements.) For a two-step project funding process, this is the maturity that the system concept needs to be at for a Step 1 funding proposal as discussed earlier. By baselining scope, the project has demonstrated a system concept that is mature enough to proceed to the next lifecycle stage and document its requirements.

Note: *In the above text I referred to a "system concept document" rather than an Operational Concept or Concept of Operations Document. See Lou's blog: ConOps vs OpsCon – What's the Difference?) For those moving away from a document-centric approach and towards a data-centric systems engineering approach, the system concept may be documented in a model developed using a systems engineering modeling tool. Using this approach, requirements and other systems engineering lifecycle artifacts can be linked to the model. With a data-centric systems engineering approach, documents are generated from the data in the form of "reports', where the "ground truth" data is maintained and managed within the model. These documents can then be shared with other internal and external organizations. For more details on data-centric systems engineering and selection SE tools to implement this approach, see Lou's blog: Features an SE Tool Set Should Have.*

Traditionally, projects focus on developing an operational concept or concept of operations document that is baselined as part of the scope review. Too often the focus is on writing this document. This is a mistake. The focus shouldn't be on the document but on maturing your system concept to the point you can document a feasible concept that has reached CML 5 and is ready to be baselined. Developing and documenting a system concept is not an exercise in writing, rather it is an exercise in systems engineering!

CML 6 – System Requirements Review (SRR) or analogous gate review: Requirements are finalized and baselined. Design and planning commensurate for a SRR is complete. From the baselined system concept, stakeholder needs and expectations are transformed into "design-to" requirements. In this context, requirements resulting from this transformation represent an agree-to obligation of what is expected by the customer and other stakeholders and communicates clearly these expectations to the design/development team. For more details on the transformation process, see Lou's blog: Going from system concepts to requirements.

Using SE tools, requirements are traced to their source and allocated to the subsystem level, schedules for the subsystem development defined, and system verification and validation approach defined. Interfaces are identified and reflected in interface requirements documented in the set of requirements being baselined. Expanded details on the technical, management, cost, risks, and other elements of the system concept have been defined and documented.

Project management and systems engineering documentation is developed to the maturity required by the organization for this lifecycle stage of the product development. The system concept document or model is updated as necessary. The TRA, resultant TRA report, and TMP are updated. The requirement set is baselined. (For a more detailed discussion on baselining requirements, the entrance criteria, exit criteria, and risk identification, see Lou's blog: Why Do I Need to Baseline My Requirements?)

CML 7 – **System Design Review (SDR) or analogous gate review**: Preliminary Implementation Baseline. Design and planning commensurate for a SDR is complete. System architecture trades have been completed and a design approach to meet the baselined system requirements has been defined and is ready for baseline. Preliminary subsystem-level requirements and analyses, demonstrated (and acceptable) margins and reserves, prototyping and technology demonstrations, risk assessments and mitigation plans are completed. Preliminary integrated cost-schedule-design is baselined.

The system concept document or model is updated as necessary. The Preliminary Project Plan is ready for review. Other project management and systems engineering documentation is developed to the maturity required by the organization for this lifecycle stage of the product development. The TRA, resultant TRA report, and TMP are updated. For a two-step project funding process, this is the maturity that your concept needs be at for a Step 2 project funding proposal as discussed earlier.

Note: *Some organizations do not have a standalone SDR. Some may combine the SDR with the SRR or with the PDR. For those cases, criteria for determining whether or not your concept is at CML 7 would combined with criteria for either CML 6 or CML 8 depending on how your organization defines their gate reviews.*

CML 8 – **Preliminary Design review (PDR) or analogous gate review**: Design and planning commensurate for a PDR is complete. System design documentation, "build-to" requirements and drawings are 10%-20% complete. Interfaces are defined in Interface Control Documents (ICDs). Final integrated cost-schedule-design is baselined. All parts/technologies have reached an TRL of at least 6. The system concept document or model is updated as necessary. The Project Plan is baselined including budget and schedule. Other project management and systems engineering documentation is developed to the maturity required by the organization for this lifecycle stage of the product development. The TRA, resultant TRA report, and TMP are updated.

CML 9 – **Critical Design Review (CDR) or analogous gate review**: Design and planning commensurate for a CDR is complete. Detailed system design documentation, "build-to"

requirements and drawings are 80%-90% complete. ICDs are complete. The system concept document or model and other project management and systems engineering documentation is updated as necessary. The TRA, resultant TRA report, and TMP are updated.

Notice that the CMLs apply to the left side of the Systems Engineering "Vee" Model. Ensuring all the activities are completed successfully on the left side of the SE Vee Model go a long way in minimizing problems that often occur on the right side of the SE Vee Model during system integration, verification, and validation. This is a major reason, and benefit, for using CMLs to manage your system development activities.

The Doctrine of Successive Refinement

The CMLs defined previously show how a system concept matures as a function of time. Note that between CMLs 2 – 4 there is a feedback loop, indicating an iterative nature between initial feasibility, trade space, and point design/architecture selection based on the trades completed. This highlights the fact that system concept definition is not a linear process, especially early on in a concept's development and maturation. Some organizations refer to this loop as Design Analysis Cycles (DACs). Management should not expect the team will get it "correct" with only one iteration. To be successful, there will often be multiple iterations or DACs. The idea of multiple interactions, or DAC cycles, is referred to as the "Doctrine of Successive Refinement" defined in the NASA Systems Engineering Handbook [NASA 1995] shown in Figure 2.

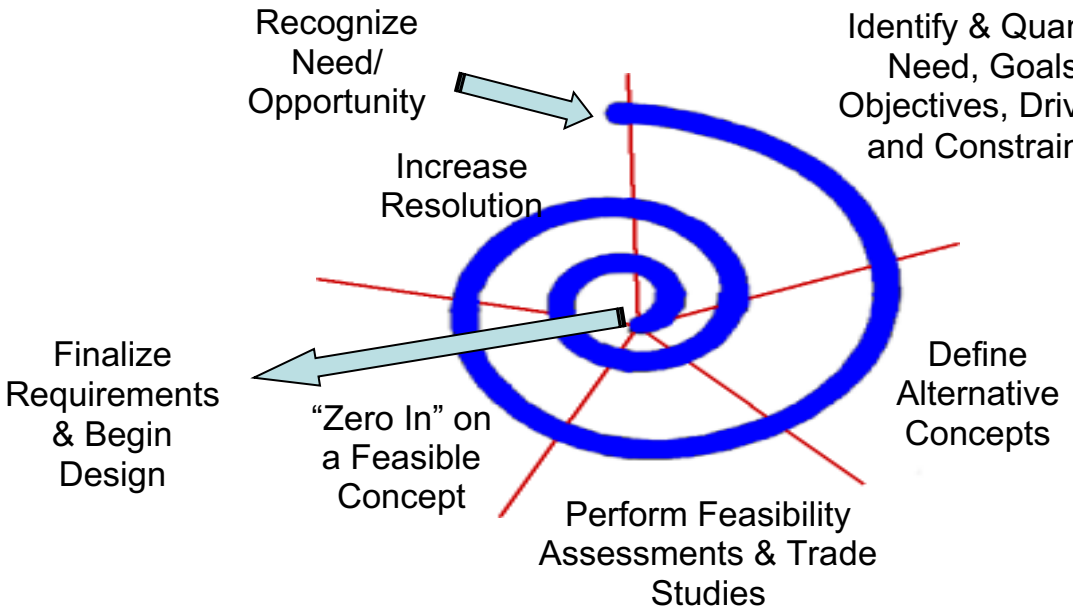


Figure 2: Doctrine of Successive Refinement

In many cases, at the beginning of concept development there are many uncertainties making it difficult to define a feasible system concept in one attempt. Projects must invest the time it takes, in the early phases of the project, to understand the problem and recognize the need/opportunity. The study/design team works with the stakeholders to identify and quantify

the stakeholder needs and expectations as well as develop a plan to refine and mature the system concept. The result is a feasible concept to meet the agreed-to NGOs within the defined drivers and constrains.

The project starts with a clear problem statement and recognized need/opportunity which remains constant throughout the system development lifecycle. Goals, objectives, and system concepts start out at a somewhat ambiguous level, but are refined with increasing resolution and precision with each inward spiral. With each iteration the study/design team gains the knowledge they need to "zero-in" on a feasible approach. The first attempts at gaining the knowledge may be a little fuzzy. Candidate system architectures and enabling technologies are identified and trade studies completed. The study/design team will assess the maturity (TRL) of the enabling technologies. Prototypes are developed and evaluated by the users. With each iteration, the prototypes will be modified based on feedback from the users. The study/design team may start with key performance statements containing threshold values as well as objective values. (See Lou's blog "How to Communicate Threshold vs Objective Requirements.)

From these activities, the study/design team increases their knowledge, zeroing in on a feasible solution, and refining the goals, objectives, drivers, and constraints. If uncertainties still exist, the study/design team can then start the cycle over at the next level of detail. With each iteration, the study/design team hones the system concept. As the enabling technologies are matured and the study/design team understands what is possible, within the stated drivers and constraints, they will be able to select a feasible concept, define the scope, develop requirements for the final product, and baseline a design.

This feedback loop is a key advantage of using the CML approach to mature a project's system concepts. This loop allows study/design teams to return to an earlier stage of concept development if system implementation issues are encountered. Typically, study teams return from a point design or architectural selection to trade space exploration to find an alternate approach for meeting the NGOs, maximizing ROI within the defined drivers and constraints. Trade space exploration is a key part of concept maturation and is needed to provide an increased likelihood that a global optimum is identified in selection of a viable system concept architecture.

In the JPL reference paper, the authors note that "prior to the development of CMLs, some study teams would move directly into a specific system architecture to implement their system concept without first having done sufficient trade space exploration. This resulted in system concepts that 1) did not have a maximum ROI, 2) had inefficient system and support system designs, and 3) had a less efficient overall system concept because trades between NGOs, [organizational ROI] expectations, the system, risks, and support system design for a particular cost point never occurred." Another issue we commonly see is projects failing to assess the TRL of the enabling technologies at the beginning of the project, yet relying on those technologies for project success. In my experience this, together with the other issues stated in the JPL reference paper, are major causes of program and project failure or significant cost overruns and schedule slips.

CMLs provide an efficient methodology for describing the maturity of a system concept and achieving the level of maturity needed for a particular project lifecycle.

Systems engineering processes are knowledge based. "*The devil is in the details.*" Your system concepts are continually evolving as your knowledge increases during the system development lifecycle. Your system concept document is a living document that needs to be kept current as your knowledge increases and system development process progresses and matures through each of the CMLs. Also, if you are using a MBSE approach, your models will also continue to evolve and get more detailed as you progress through each of the CMLs and product development lifecycle stages.

Tools to help implement the CML concept

In the JPL reference paper, the authors define two key tools that will help the study/design team to advance from one CML to another: the CML Matrix and the CML checklists. The CML Matrix and CML checklists allow the study/design team and management to assess and know the current state of a system concept, enabling them to compare the state of the system concept to that required to pass a specific gate review and proceed with the next set of activities to further mature the system concept and proceed with the next development lifecycle phase. The result is an explicit list of areas where the concept is at the desired level of maturity and a list of deficiencies (more work and resources are needed to address these deficiencies). Armed with that information, the study/design team can efficiently plan their work to get their mission concept up to the desired level of maturity.

Note: *There is an overlap of the CMLs 5-9 corresponding to a major gate review occurring at the end of major system development lifecycle phases. Because of this, standard systems engineering and project management processes and guidelines can be referred to when generating the CML Matrix and corresponding CML Checklists for these levels. For example, NASA's SE process definition standards (NPR 7123.1) define specific entrance and success criteria for each of the lifecycle gate reviews corresponding to CMLs 5-9. As stated previously, some organizations may combine the MDR (CML 7) with either the SRR (CML 6) or PDR (CML 8). In this case, the CML Matrix and associated checklists described below will need to be tailored to reflect the organization's specific gate reviews.*

CML Matrix

The CML matrix defined in the JPL reference paper contains columns corresponding to each CML (they show CML 1-7) and rows for key attributes associated with a concept. The rows are grouped into areas of interest: science, technical, management, cost, and "other". (Figure 3 shows the science area of interest.) Instead of science the first area could be labeled MOEs or KPPs, whatever best fits your project domain. Because each domain is different, we leave it up to your organization to develop a CML matrix tailored to your organization, domain, and system using the JPL reference paper CML matrix and the updated CML descriptions as examples.

Life Cycle Phase	Pre-Phase A					Phase A	
	Advanced Studies			Concept Development		Early Formulation	
CML	1	2	3	4	5	6	7
Name	Cocktail Napkin	Initial Feasibility	Trade Space	Point Design	Baseline Concept	Integrated Concept	Preliminary Implementation Baseline
Life Cycle Gate	—	—	—	Concept Gate (Draft AO Out / Mission Study Report)	Baseline Commitment Gate / MCR	Step 2 Submittal	PMSR / VDR
Science							
Attribute							
Science Objectives & System Requirements	Science objectives described in one sentence	Objectives described to levels that allow comparison with previous investigations and NASA science community documents	Objectives linked to investigations and measurements Science return as a function of cost, risk and programmatics quantified	Produce draft Science Traceability Matrix Initial Level 1 requirements considered Specifying one Baseline and one Threshold Science investigation Key Performance Parameters listed	Science Traceability Matrix (or equivalent) produced Preliminary PLRA produced (assigned projects)	Proposed Level 1 requirements documented Level 2 & 3 driving requirements listed Full and minimum success criteria defined Baseline PLRA submitted @ SRR (assigned projects)	Update PLRA if necessary Preliminary Level 2 & 3 requirements listed
Science Data System	—	Identify science data drivers	Science data rates and volume included in trade space analysis	Science data system sizing	Science data processing architecture, release and archive approach defined	Science data management approach (includes Level 0, 1, 2 data products) defined	Same as for CML 6

Figure 3. Example CML Matrix described in the JPL reference paper.

The intent of the CML matrix is to serve as a high-level guide for study/design and proposal teams through the stages of system concept maturation and architecture selection. The matrix can be used by management and the study/design team in several ways:

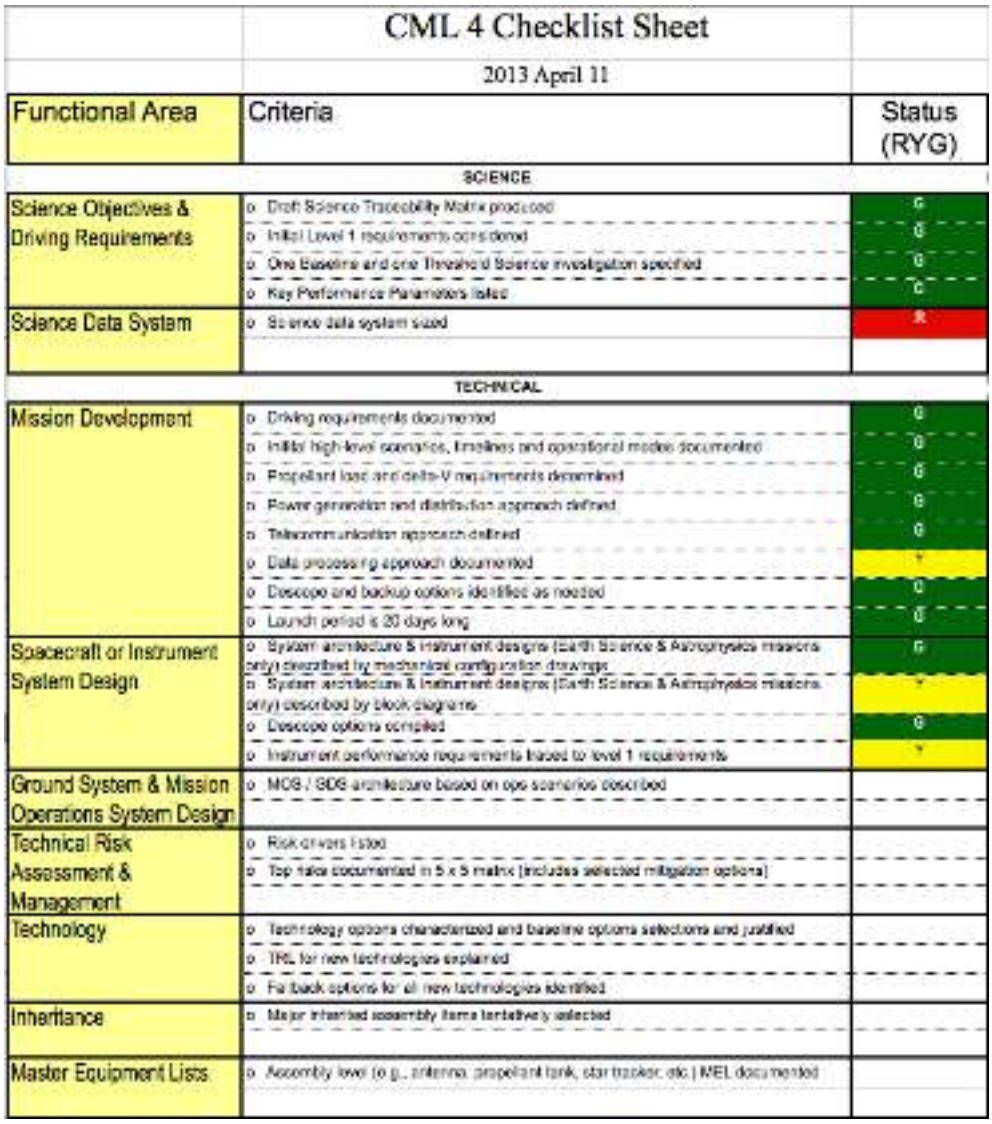
1) To determine the maturity of a system concept at the time of a particular gate review. As an example, by looking at the contents of the cells in the CML 5 column, a system architect can quickly see the material that is needed for a study/design team to pass their Mission Concept or Scope Review. Of course, the matrix only identifies what is needed at each CML, not the quality of the deliverable nor what is necessary to achieve a winning proposal.
2) To understand the deliverables and their maturity required as a function of time.
3) As discussed below, the contents of each column can be used to generate a CML checklist.

CML Checklists

The CML Checklists defined in the JPL reference paper provide study/design teams and management a tool to assist in CML evaluation. (An example of a CML checklist is shown on the next page in Figure 4.) The CML Checklists can be applied quickly to help identify gaps for the various CMLs. The CML Checklists:

1) Allow management and the study/design team to quickly measure the system concept's maturity,
2) Are reusable, i.e., the checklists can be applied to any project that is maturing their concepts, providing the same level of maturity score for concepts with the same level of maturity and,
3) Identify deficiencies and provide clear information as to what areas of the concept need

additional work to get to the overall mission concept to the desired level of maturity.



CML 4 Checklist Sheet		
	2013 April 11	
Functional Area	**Criteria**	**Status (RYG)**
	SCIENCE	
Science Objectives & Driving Requirements	o Draft Science Traceability Matrix produced	G
	o Initial Level 1 requirements considered	G
	o One Baseline and one Threshold Science investigation specified	G
	o Key Performance Parameters listed	G
Science Data System	o Science data system sized	R
	TECHNICAL	
Mission Development	o Driving requirements documented	G
	o Initial high-level scenarios, timelines and operational modes documented	G
	o Propellant load and delta-V requirements determined	G
	o Power generation and distribution approach defined	G
	o Telecomm unication approach defined	G
	o Data processing approach documented	Y
	o Descope and backup options identified as needed	G
	o Launch period is 20 days long	G
Spacecraft or Instrument System Design	o System architecture & instrument designs (Earth Science & Astrophysics missions only) described by mechanical configuration drawings	G
	o System architecture & instrument designs (Earth Science & Astrophysics missions only) described by block diagrams	Y
	o Descope options compiled	G
	o Instrument performance requirements traced to level 1 requirements	Y
Ground System & Mission Operations System Design	o MOS / SDS architecture based on ops scenarios described	
Technical Risk Assessment & Management	o Risk drivers listed	
	o Top risks documented in 5 x 5 matrix (includes selected mitigation options)	
Technology	o Technology options characterized and baseline options selections and justified	
	o TRL for new technologies explained	
	o Fa back options for al new technologies identified	
Inheritance	o Major inherited assembly items tentatively selected	
Master Equipment Lists	o Assembly level (e g., antenna, propellant tank, star tracker, etc.) MEL documented	

Figure 4. Example portion of a CML Checklist described in the JPL reference paper.

The checklists are based on the activities and products identified in the specified column of the CML Matrix. Again, because each domain is different, we leave it up to your organization to develop CML checklists tailored to your organization, domain, and system, using the JPL reference paper CML checklists, the expanded CML descriptions in this paper as examples.

The CML checklists are intended to provide guidelines for management and the study/design team to correct areas that are found deficient. The CML checklists provide an independent check on where the system concept is weak and where resources should be applied to make the system concept implementable and robust. Once the system concept maturity is assessed, management and the study/design team is faced with a choice. Of the areas that are identified as “not at the desired maturity level,” where should the team’s limited resources be applied to

help achieve their overall system concept's desired maturity? In many cases, study teams do not have the time, resources, and funds to correct all deficiencies identified by the CML checklists. It is a decision for management, the study/design team lead, project's lead systems engineer and/or any other key team personnel to decide how best to apply the team's limited resources to most efficiently correct the most critical omissions. This should be a risk-based decision.

Using CMLs as part of other development approaches

The discussion concerning CMLs to this point assume a scenario for system development where the complete system is to be delivered with all the capabilities, functionality, performance, security, quality, etc. per the design-to requirements baselined at CML 6. This is the case for a project like the NASA space science projects managed at JPL where the CMLs were originally defined. This approach is used when the end system requirements can be baselined and all the capabilities, functionality, performance, and quality needs to be realized in one delivery. This approach is often referred to as the "waterfall" development approach.

However, one size doesn't fit all. There are other approaches that may be more appropriate depending on the domain and needs of your organization. For example, in the US Department of Defense, different philosophies to system development and acquisition are gaining popularity including Rapid Acquisition Program (RAP) and Quick Reaction Capability (QRC). RAP and QRC promise the ability to:

- quickly develop, communicate, and validate requirements to resolve unforeseen threats to mission and/or casualties;
- identify solutions that can be rapidly fielded with no, or very limited, development; and the willingness to accept less than a 100% solution;
- immediately apply funding to selected solutions and begin executing a program; and
- execute a tailored acquisition that accepts "acquisition risk" but delivers solutions in a time frame consistent with the urgency of the mission requirements, thus reducing the warfighter's operations risk.

Other organizations are using approaches like Rapid Application Development (RAD), agile development, rapid technology infusion cycles, and middle-out or bottom-up design. In consumer products, time to market is a key consideration. For example, the Apple watch. The first hardware release provided functionality and a high degree of quality, but it was a "push" product not driven by consumer needs for the watch, but rather what Apple thought consumers would want. Apple's vision for the Apple watch is far greater than what was delivered in the first increment. Some functionality and performance was left out due to technologies available at the time as well as the need to get it to market given the smart watches being release by others. From a software perspective, iWatch 1.0 was clearly a "beta" version. iWatch 2.0 fixed problems and added some new features and performance. However, many feel that iWatch 3.0 is the first real consumer release based on lessons directly from the consumers when using the earlier versions. It seems to be a common practice to let not only developers but consumers beta test software products. Incremental releases that consumers

must pay for also form many organization's business model. When using this approach, the organization can apply CMLs to mature their system concept for each of the incremental release as well as the final system release.

One key issue is risk. When is a product concept mature enough to proceed with development and release the product for use? If you release too early, bugs, poor performance or poor quality, can result in negative consumer/operator feedback and can doom the product to failure. If you wait too long, a competitor may introduce a product that dominates the market, making it difficult to break into this market or gain market share. There is no easy answer.

Because systems engineering is a knowledge based process, CMLs provide an excellent roadmap to follow no matter which approach an organization chooses to follow. No matter the system to be developed and the development approach used, project managers need to understand the problem, the need, goals, drivers, and constraints. They need to define a feasible system concept that will address the problem/opportunity and understand user expectations. These expectations need to be transformed in to a set of requirements that represent an agreed-to obligation (contract) for the developers and to which the designed and built system will be proven to meet (system verification and system validation.)

In addition to the classic waterfall development approach, two historical develop approaches that can be used to meet the evolving need for more rapid processes for system development and acquisition are incremental and spiral development.

Incremental Development

Incremental Development is similar to Waterfall – where the organization can define a core set of system requirements upfront -- however, rather than delivering the whole system at one time, as shown in Figure 5 the organization delivers the system in increments, with each subsequent increment delivery adding additional capabilities, functionality, performance, quality, etc. based on stakeholder priorities and changing needs. The first increment results in a useable system that is delivered in a shorter time that meets the stakeholders top priority needs and expectations. Subsequent increments add to the first delivered system lower priority needs/requirements, additional capabilities, functionality, performance, quality, etc.

Incremental Development can be combined with iterative approaches as well. Where Incremental Development focuses on planned deliveries, adding new capabilities with each delivery, iterative focuses on making improvements on existing deliveries as well as updating requirements for future increment deliveries. With an iterative focus, an organization can use the subsequent increments to include updates to the previous increments based on lessons learned during testing and operations of the system "in the field" or infuse new technologies that were not mature enough to be included in the first increment. This allows the organization to correct problems found during the previous increment delivery and use this knowledge to improve the system capabilities, functionality, performance, quality, etc.

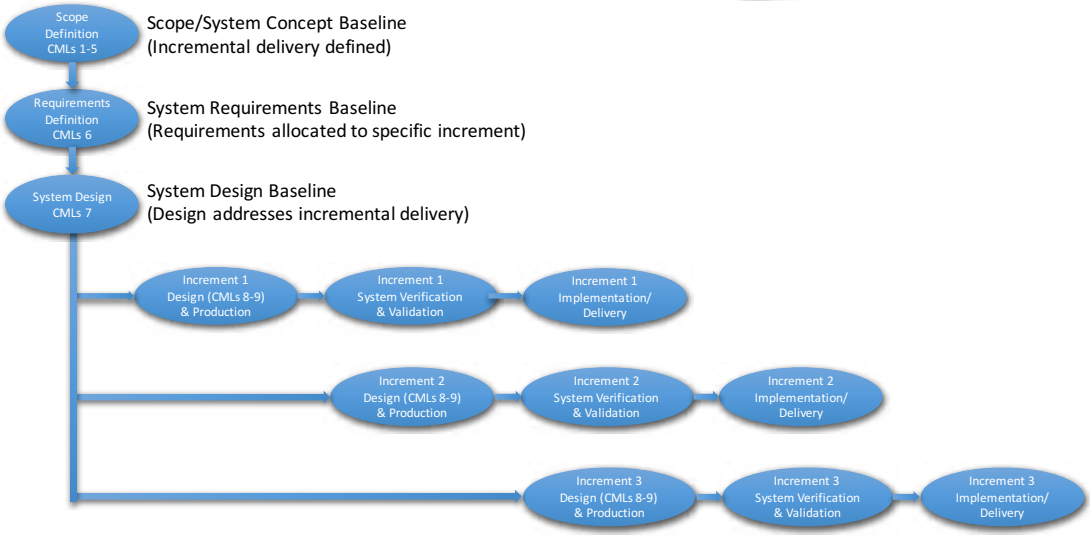


Figure 5. Incremental Development and Delivery

The Incremental Development approach supports organizations that have adopted a rapid development philosophy as discussed earlier. Key provisions that enable Incremental Development include using a modular design philosophy and defining the interfaces (external and internal) at the beginning of the project so that modules added or upgraded in subsequent increments will be able to be integrated into the system. A key tenant of Incremental Development is that a 75% solution is acceptable and useful. A product delivered in 1 year, that provides 75% of the needed capabilities, is more useful and beneficial to the current users, than a product delivered 2 years later with 100% of the needed capabilities.

Spiral Development

Spiral Development is a good approach when the requirements are difficult to define upfront. As shown in Figure 6, each spiral can result in a prototype that can be shown to the stakeholders to get their feedback. It often takes several spirals to nail down the requirements. Once the stakeholder needs, expectations, and desired outcomes are understood, the requirements can be defined and the organization can transition into either a waterfall approach and deliver the whole thing or an incremental approach and deliver the system in increments driven by stakeholder priorities and changing operational needs.

Spiral Development is very similar to the Doctrine of Successive Refinement discussed earlier. The Spiral Development model results in the same fundamental outcome. While the Doctrine of Successive Refinement started on the outside and spiraled inward, zeroing in on a solution, Spiral Development is shown as starting in the middle and spiraling outward toward the same outcome. With each spiral our knowledge increases and so does the maturity of our concept and requirements to the point we are ready to build the final product. The Spiral Development approach is very applicable to the loop nature of CMLs 2-4 as discussed earlier. The key difference is that the NGOs, KPPs, MOEs, etc. defined during CML 1 activities are very preliminary at the beginning where the stakeholders have a problem but, initially, a concept to

address that problem is elusive, requiring more work to develop an acceptable solution. Once an initial concept is able to be formulated, then it can be matured into a feasible concept that addresses the problem and the stakeholders agreed-to NGOs. Thus, for projects like this, the loop is between CML 1 and 4.

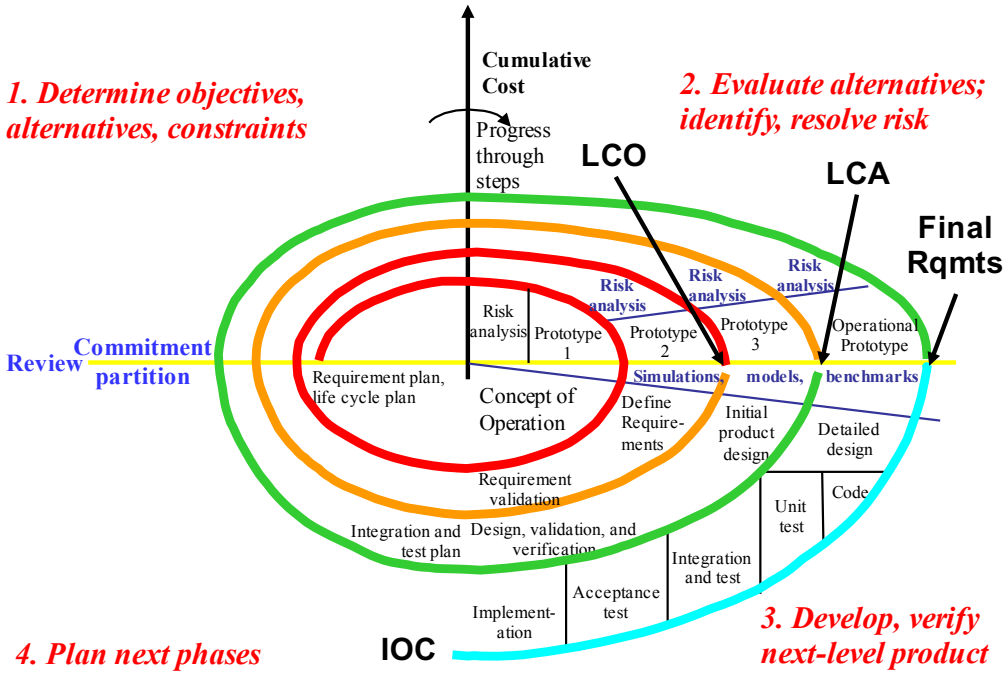


Figure 6: Spiral Development derived from [Boehm 2001]

Spiral Development was developed by Dr. Barry Boehm, primarily addressing software development projects. Spiral Development addresses projects where the requirements aren't clearly understood at the beginning, users often don't know what they want until they see a prototype, and based on what they learn, the user requirements often change. Dr. Boehm defines spiral development as [Boehm 2001]:

> "*The spiral development model is a **risk-driven process model** used to guide multi-stakeholder **concurrent engineering** of software-intensive systems. It has two main distinguishing features. One is a **cyclic** approach for incrementally growing a system's degree of definition and implementation while decreasing its degree of risk. The other is a set of **anchor point milestones** [life-cycle objectives (LCO) – what should the system accomplish, life-cycle architecture (LCA) - what is the structure of the system, and initial operational capability (IOC)- first release] that are key decision points for ensuring stakeholder commitment to feasible and mutually satisfactory system solutions.*"

Prototypes are used to support the risk analysis process that results in a better understanding of the user's needs and the challenges in meeting those needs. Once a prototype has been validated it will often become the simulation, model, or benchmark for the next step in the spiral.

Each trip around the spiral starts with determining and refining the NGOs, MOEs, and KPPs as well as one or more system concepts that will result in meeting the NGOs within the defined drivers and constraints. Multiple spirals (CML 1-4 loops) may be required until the NGOs are finalized and a feasible system concept has been defined to meet those NGOs. Prototypes are developed, matured, and evaluated by the stakeholders. Trades are completed. Once the team has a feasible concept (CML 5) they can proceed to finalize the design-to requirements (CML 6), and proceed with development of the end product (CMLs 7-9). With each spiral stakeholder needs, expectations, drivers and constraints are better understood, alternative concepts are identified and accessed, and the system concept matures.

Each spiral considers these main elements: 1) critical-stakeholder objectives and constraints, 2) system concept alternatives, 3) risk identification and resolution 4) stakeholder review, and 5) commitment to proceed. These elements are very similar to what is defined for CMLs 1-5. There is enough flexibility in the spiral model enabling the study/design team to complete several spirals before going on to the next CML – it depends on the results of the risk analysis and the decisions made along the review/commitment partition shown in the diagram.

Boehm's inclusion of anchor point milestones is especially important. Each milestone is a stakeholder commitment point. The purpose of the LCO review is to make sure that there is at least one feasible system concept to meet the stated NGOs of the project. The purpose of the LCA review is to align the stakeholders to a common vision and agree on the scope of the project (CML 5). Boehm states: "*The LCA milestone is particularly important, as its pass/fail criteria enables stakeholders to hold up projects attempting to proceed into evolutionary or incremental development without a life cycle architecture*. [Boehm 2001]" Risk analysis is a key activity for each spiral and the project must have a risk management plan in place. At each milestone the project risks are reviewed and the resolution of each risk is agreed to by the stakeholders.

To better understand how a project can use the anchor point milestones as part of the CML approach, Boehm provides the following Stud Poker Analogy[Boehm 2001]: "*A valuable aspect of the spiral model is its ability to support incremental commitment of corporate resources rather than requiring a large outlay of resources to the project before its success prospects are well understood. Funding a spiral development can thus be likened to the game of stud poker. In that game, you put a couple of chips in the pot and receive two cards, one hidden and one exposed. If your cards don't promise a winning outcome, you can drop out without a great loss. This corresponds to canceling a project at or before LCO. If your two cards are both aces, you will probably bet on your prospects aggressively (or less so if you see aces among other players' exposed cards). Dropping out of the second or third round of betting corresponds to canceling at or before LCA. In any case, based on information available, you can decide during each round whether it's worth putting more chips in the pot to buy more information or whether it's better not to pursue this particular deal or project*."

Establishing a collaborative work environment

To support "rapid development and acquisition" philosophies, organization's need to provide a collaborative work environment conducive to the activities that take place during CML 1 - 4.

To streamline the system development process, organizations need to move to a "lean" project management and system engineering organization and adopt an agile product development approach. Key is for the organization to enable their teams to work in these environments, cutting out unnecessary bureaucracy, streamlining development and acquisition processes, reducing the number of reviews, etc.

To be successful, the study/design team needs to include subject matter experts (SMEs) with the experience and expertise to foresee potential issues during later project phases and include actual users who understand the problem and what an acceptable operational solution needs to include. When forming the team, it is also important to use younger personnel who are not afraid to ask "Why are we doing it this way?" and older, more experienced personnel who can answer that question, but are open to new ideas and innovative approaches to addressing the problem solution. The team also needs to include personnel who represent the various lifecycle stages, especially those involved in design, acquisition, manufacturing, test, operations, and maintenance. To be effective, these teams need to be kept small: 7 +/- 2 is a good number.

NASA has recently released a two volume document titled: "Expanded Guidance for NASA Systems Engineering". In volume 2, Section 7.2, Concurrent Engineering, they discuss the need for and advantages of establishing a collaborative work environment. In a collaborative work environment, team members are often collocated and are provided the tools, data, and supporting information technology infrastructure in an integrated support environment that can be immediately used by the team. In this type of collaborative environment, questions can be answered immediately, or key participants can explore assumptions and alternatives with the stakeholder team or other study/design team members and quickly reorient the whole team when a design change occurs. The collaboration triggers the creativity of the engineers and helps them close the loop and rapidly converge on a feasible concept.

Organizations that are pursing this approach may call the team and associated work environments various names: Team X, A Team, Rapid Mission Architecture, Integrated Design Center, Advance Concepts Office, Concept Design Center, Concurrent Design Facility, Skunk Works, etc.

One example is NASA's Glenn Research Center's COMPASS (Collaborative Modeling for Parametric Assessment of Space Systems) Team, which was established to meet the need for rapid mission analysis and multi-disciplinary systems design for in-space and human missions. The COMPASS Team is a multidisciplinary, concurrent engineering group whose primary purpose is to perform integrated systems analysis, but it is also capable of designing any system that involves one or more of the disciplines present in the team.

Another example is the US Special Operations Command (SOCOM). They have created SOFWERX, an institute designed to facilitate communication between the technology community and U.S. Special Operations Command. SOFWERX is both a program and a place near Tampa Bay, Florida. SOFWERX creates a network of collaborators enabling a very agile and rapid acquisition process. A goal of SOFWERX is to create processes and venues to make it easier for individuals with innovative ideas and technologies to collaborate to solve a problem and rapidly identify, procure, and release the solution into the field. The program holds rapid prototyping sessions where a group gets together to discuss and analyze a particular problem to figure out how to move a state-of-the-art capability forward to solve a specific problem. SOFWERX regularly holds capability collaboration sessions bringing together operators, technical experts, acquisition officials, industry representatives, and academia to work on new ideas. SOFWERX is equipped with 3-D printers and other tools for rapid prototyping to quickly get the job done in-house.

Summary

The use of CMLs provide management and study/design teams an approach that facilitates discussions of concept maturity, provides a basis for improved concept development practices, and helps establish reasonable expectations based on the maturity of the concepts in consideration. It is not advisable to assume that a project "magically" has reached CML 5 and its scope is ready for baseline just because someone has written a System Concept Document. By adopting the CML approach, project teams will be expected to do the work necessary to mature their system concepts and progress from CML 1 to CML 5 before they are allowed to baseline their scope and proceed with the next development lifecycle stage.

Management must determine when the concept is mature enough to fund the next lifecycle stages within the acceptable risk profile. For identified risks that are manageable, a plan for mitigating the risks needs to be put into place. CMLs provide a standardized mechanism for describing and communicating the products/activities required for achieving a given CML and for identifying work remaining to proceed to the next level. CML(s) address the broad scope of systems engineering, technology, and programmatic parameters, and are useful for identifying analysis gaps and areas requiring more in-depth evaluation. It is important to note, that for each CML level achieved, the system fidelity, system definition accuracy, and its implementation is more clearly understood. As a result, the development risk is lowered as comparted to approaches that do not use CMLs to define and manage system development.

CMLs can be used as part of any system development approach whether waterfall, incremental, agile, or spiral as well as any procurement/acquisition philosophy: classic or rapid. To implement a successful CML approach, it is important for the organization to enable the formation of a collaborative team and provide facilities and capabilities to allow the team to perform the work needed to quickly mature their concepts.

In the classes that Requirements Experts teach, we make the point that all projects should strive to deliver a winning product – one that delivers what is needed, within cost and schedule constraints, with the desired quality. To do this, projects need to have a clear vision,

knowledge, and be able to stay on course. Using CMLs will help to do this so that project teams can deliver a winning product – the first time!!!

References:

Association of the United States Army, *Rapid Equipping and the U.S. Army's Quick-Reaction Capability*, October 2015, https://www.ausa.org/file/518/download?token=cpmV_uNh

Boehm, B., *Spiral Development: Experience, Principles, and Refinements,* CMU/SEI-00-SR-008, Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000.

Boehm, B. and Hansen, W., *The Spiral Model as a Tool for Evolutionary Acquisition,* CrossTalk - The Journal of Defense Software Engineering, May 2001: 4-11

Government Accountability Office, *Technology Readiness Assessment Guide*, draft, August 2016, http://www.gao.gov/assets/680/679006.pdf

Judson, Jen, *Special Operations Command Breaks Down Buying Barriers,* Defense News, May 2016, http://www.defensenews.com/story/defense/policy-budget/budget/2016/05/10/special-operations-command-socom-sofwerx/84193372/

NASA *System Engineering Handbook*, SP6105, June 1995.

NASA *Expanded Guidance for NASA Systems Engineering*, Vol 1 & 2, March 2016

NASA, Glenn Research Center, *COMPASS*, https://re.grc.nasa.gov/compass/

NASA, *Systems Engineering Processes and Requirements*, NPR7123.1B, April 2013, NASA Online Directives Information System (ODIS) Library: http://nodis3.gsfc.nasa.gov

Oehmen, Josef, Editor, *The Guide to Lean Enablers for Managing Engineering Programs*, Version 1, Joint MIT-PMI-INCOSE Community of Practice on Lean in Program Management, May 2012, http://dspace.mit.edu/bitstream/handle/1721.1/70495/oehmenetal2012-theguidetoleanenablersformanagingengineeringprograms.pdf?sequence=4

Romero, Pia, *Quick wins show the benefits of DoD's Rapid Acquisition Program*, Federal News Radio, June 2012, http://federalnewsradio.com/defense/2012/06/quick-wins-show-the-benefits-of-dods-rapid-acquisition-program/

Wessen, Randii R., Borden, Chester, Ziemer, John, Kwok, Johnny, *Space mission concept development using concept maturity levels*, NASA JPL, AIAA SPACE 2013 Conference & Exposition, San Diego, California, September 10-12, 2013
http://hdl.handle.net/2014/44299

Wheatcraft, Louis S. & Hooks, Ivy F., *Scope Magic*, 2001.
http://reqexperts.com/wp-content/uploads/2016/04/scope_magic.pdf

Wheatcraft, Louis S., *Delivering Quality Products That Meet Customer Expectations*, CrossTalk, Vol.16 No.1, January 2003.
http://reqexperts.com/wp-content/uploads/2016/04/Delivering-Quality-Products-Crosstalk_Wheatcraftjan03.pdf

Wheatcraft, Louis S., *Developing Requirements for Technology-Driven Products*, paper presented at the INCOSE 15th International Symposium 2005, Rochester, NY.
http://reqexperts.com/wp-content/uploads/2016/04/Rqmts-for-Technology-Driven-Products-Wheatcraft-INCOSE-012612.pdf

Wheatcraft, Louis S., *ConOps vs OpsCon – What's the Difference?*, blog, June 2013,
http://reqexperts.com/blog/2013/06/conops-vs-opscon-whats-the-difference/

Wheatcraft, Louis S., *Going from system concepts to requirements*, blog, July 2013,
http://reqexperts.com/blog/2013/07/going-from-system-concepts-to-requirements-part-1/

Wheatcraft, Louis S., *MBSE and Requirements*, blog, February 2014,

http://reqexperts.com/blog/2014/02/mbse-and-requirements/
Wheatcraft, Louis S., *How to Communicate Threshold vs Objective Requirements*, blog, July 2014, http://reqexperts.com/blog/2014/07/how-to-communicate-threshold-vs-objective-requirements/
Wheatcraft, Louis S., *What problem are you trying to solve?*, blog, August 2015, http://reqexperts.com/blog/2015/08/what-problem/
Wheatcraft, Louis S., *Baseline Your Scope Before Writing Requirements*, *blog, September 2015,* http://reqexperts.com/blog/2015/09/baseline-scope/
Wheatcraft, Louis S., *Why Do I Need to Baseline My Requirements?*, *blog, September 2015,* http://reqexperts.com/blog/2015/09/why-do-i-need-to-baseline-my-requirements/
Wheatcraft, Louis S., *Using Technology Readiness Levels to Manage Risk*, blog, October 2015, http://reqexperts.com/blog/2015/10/using-technology-readiness-levels-to-manage-risk/
Wheatcraft, Louis S., *Features an SE Tool Set Should Have*, blog, December 2015, http://reqexperts.com/blog/2015/12/features-a-re-tool-set-should-have/
Wheatcraft, Louis S., *Resource Margins and Reserves – Part 1*, blog, March 2016, http://reqexperts.com/blog/2016/03/resource-margins…-reserves-part-1/
Wikipedia, *Rapid application development,* https://en.wikipedia.org/wiki/Rapid_application_development

Biography

Lou Wheatcraft is a senior instructor/consultant for Requirements Experts (RE) who educates organizations on the importance of writing good requirements and helps them implement Requirement Development and Management (RD&M) processes based on industry best practices. Lou has taught over 180 requirement seminars over the last 16 years. Lou works with both government and industry clients to tailor training for their organizations and provides just in time team training for specific projects. Lou has spoken at Project Management Institute (PMI) Chapter meetings, International Council of System Engineering (INCOSE) conferences and NASA's PM Challenge and delivered tutorials to PMI and INCOSE chapters at multiple locations. Lou has had published and presented a multitude of papers on requirement RD&M topics for NASA's PM Challenge, INCOSE, INCOSE INSIGHT Magazine, and Crosstalk Magazine. Lou is a member of INCOSE, co-chair of the INCOSE Requirements Working Group, a member of PMI, the Software Engineering Institute (SEI), the World Futures Society, and the National Honor Society of Pi Alpha Alpha. Lou has a BS degree in Electrical Engineering from Oklahoma State University, an MA degree in Computer Information Systems from the University of Houston – Clear Lake, an MS degree in Environmental Management from the University of Houston – Clear Lake, and has completed the course work for an MS degree in Studies of the Future from the University of Houston – Clear Lake. Lou is the primary contributor to RE's blog on requirements best practices. The blog can be assessed at: http://www.reqexperts.com/blog.

Layne Lewis runs Willowview Consulting, LLC where she works as a Systems Engineer consulting for commercial companies, the Department of Defense, and other Federal agencies. **Layne** also works with small companies to help them enter the world of government contracting. Previously, she founded Motionetics, Inc., a company developing innovative energy harvesting and sensor technology products for military and commercial applications. **Layne** started her entrepreneurial journey as the co-founder and chief operations officer of TenXsys, Inc., which developed small devices for monitoring human health and tracking animal movements. Prior to that, she was a program and project manager and software development engineer for Hewlett Packard, where she developed firmware for HP's LaserJet printers. **Layne** worked as a senior systems engineer for Booz Allen & Hamilton Inc., where she worked on various NASA projects, including the Space Station communications systems monitor and control subsystem for the NASA enhanced mission communications system, and the interface for the Space Shuttle aft flight deck to the Space Station docking module. Layne holds a BS in aerospace engineering from the University of Arizona and an MBA from the University of Washington.