

Selecting a Software Development Life Cycle (SDLC) Methodology

A Practical Decision Framework to Maximize Business Value (The D³ Cube)

The IT Project Manager's Challenge

One of the greatest challenges facing IT project managers today is the seamless integration of project management methodology and SDLC methodology. The D³ Cube is a practical bridge for project managers seeking to select the appropriate SDLC methodology for their project.

Selecting a Software Development Life Cycle (SDLC)ⁱ methodology is a challenging task for many organizations. What tends to make it challenging is the fact that few organizations know what criteria to use in selecting a methodology to add value to the organization. Fewer still understand that a methodology might apply to more than one Life Cycle Model.ⁱⁱ Whether you are a Fortune 500 firm or an Armed Services agency, today's demanding time-to-market conditions have organizations focused on how to deliver the needed solutions faster. Business budget, time, and functionality requirements are driving business value and defining market changes.

The enterprise is becoming more and more dependent on the timely delivery of these systems, the solution's ability to reliably sustain the business operation, and the practitioners ability to provide solutions that support changing demands. The net effect is that practitioners are seeking methods of developing solutions that give them control over complexity (feature, content, and architecture), sizing (point solution, departmental or enterprise), service level (ad hoc, reliable, and mission critical), and delivery (buy vs. build decisions). These are not simple challenges and the selection of a methodology to accommodate these demands is critical to the success of the business. Consider, for example, the methodology utilized to develop an informational web site may be inappropriate to use in constructing software that handles flight controls for a fighter jet. Why? One may need high reliability but the other is life critical where no room for error exists. Seeking the balance of time verses rigor is the practitioner's challenge. Selecting the right methodology and then acquiring and building process, body of knowledge, and competency become critical success factors to delivering the businesses value proposition.

A common misconception is that methodology is synonymous with technology or industry constructs. Structured programming, Event Driven, Client-Server, Object-Oriented, n-Tier, and Service-Oriented Development of Applications (SODA) are all examples of constructs that have evolved over time to deliver faster or better systems. They are not methodologies. A methodology is the process within which any or all of these might be utilized and controlled to deliver the solutions. While technology constructs may influence your selection of a methodology in the short term, value to the business should be the predominant driver of methodology selection. Additionally, the type of organization and its focus may significantly influence the selection. Case in point; an Information Technology (IT) or Management Information Systems (MIS) organization will likely select

Selecting an SDLC Methodology

a significantly different methodology from that of a Line of Business (LOB) or Functional organization. The appropriateness of either's selection is dubious without a common decision criteria.

To further illustrate this point and its significance, consider the following. When selecting a SDLC methodology, technical organizations (i.e., IT, MIS, etc.) are most often looking to the technical merits of the various methodologies to drive their decision. In reality, these technical merits are subjective measures such as tool kit availability to support the methodology, staff familiarity with a given methodology, or the "perceived" benefits of the methodology where perceived benefits often include additional subjective measures such as ease of implementation, fit for existing platforms, or time-to-delivery for today's hot project. This is in sharp contrast to the approach a given LOB or Functional organization might take. Such an organization is more likely to look for and SDLC methodology that reduces their IT resource costs, reduces project risk, is easy to understand, and provides known management metrics.

From this perspective, both IT and the LOB could select a SDLC which they whole heartedly believe will fit their needs. And, in fact, it may. However, it is more likely that the selection would be one of good fortune than good science. Without objective measures, the selected SDLC is simply a matter of subjective choice with no demonstrable value to the organization above that which is perceived by those selecting the methodology.

This paper seeks to provide an objective framework from which IT and LOB organizations alike can make an effective selection of SDLC methodology. The decision process presented is one which allows IT and the LOB organizations to select a methodology in concert with one another and aligned with the purpose of the organization.ⁱⁱⁱ

Methodology Overview

Before considering a framework for selecting a given SDLC methodology, one needs to understand the choices. In this paper we are focused on constructing software and will seek to address the primary approach of the various methodologies in supporting software development initiatives.^{iv} Systems development, systems engineering, integration, and Commercial Off The Shelf (COTS) software package implementation initiatives are included within these life cycles as well, however the majority of our discussion will focus on software development specifically. Our discussion seeks to represent the principles and concepts of each methodology without a bias toward one methodology over another.^v

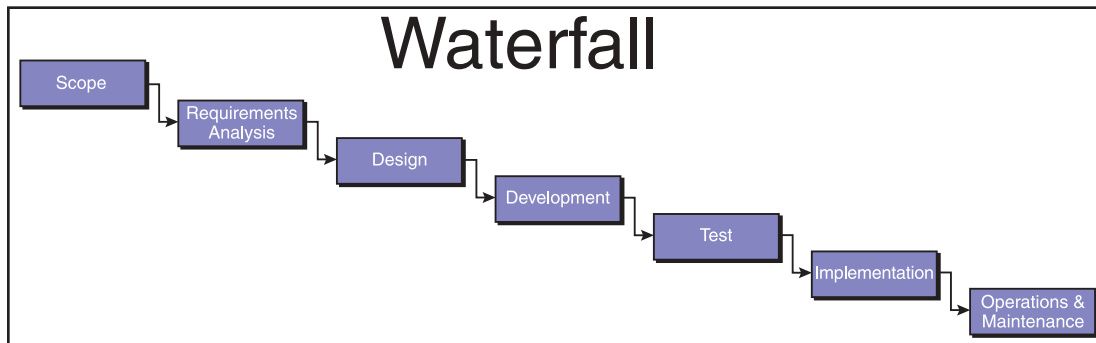
SDLC methodologies all focus on the common goal of defining the steps and processes from the beginning of a software project through its completion. Traditional engineering models have depicted these steps as Requirements Analysis, Design, Development, Test, and Implementation. However, contemporary thinking provides for a cradle-to-grave perspective by which IT and LOB are closely linked in the process. Accordingly, the models herein presented add a step to the beginning (Scope) and Ending (Operations and Maintenance) of each methodology. Scope (sometimes referred to as Strategy, Feasibility, or Concept of Operations) captures the preplanning, business case, and boundaries of the software project. Operations and Maintenance (sometimes referred to as Production Support, Post Install, or Execution) captures the day-to-day ongoing activities necessary to sustain the system.

While there are various derivatives of the basic methodologies, there are predominately six SDLC methodologies in practice today. Following is a short description of each:

Selecting an SDLC Methodology

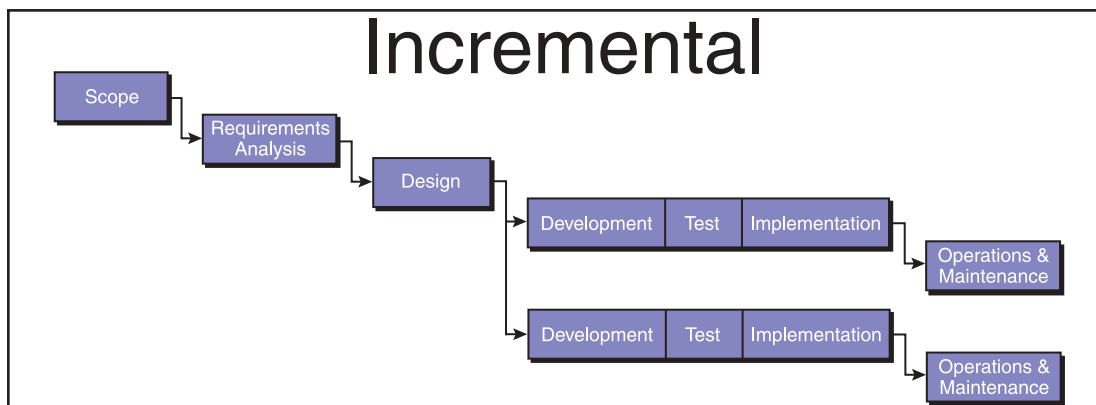
Waterfall

This methodology was the first formalization of a process for controlling software development. The Waterfall methodology was, and still is, the foundation for all SDLC methodologies. The basic phases in this methodology are utilized in all other methodologies as descriptors of processes within a given SDLC. The Waterfall methodology, as its name implies, is a series of phases that are distinct and cascading in nature. As the graphic below portrays, each phase is dependent on the preceding phase before it can begin, and requires a defined set of inputs from the prior phase. Subsequent phases are driven to complete the requirements defined in the Analysis phase to assure the resulting software meets these requirements. A slight derivative of this methodology exists, typically known as Modified Waterfall, whereby the end of one phase may overlap with the beginning of another, allowing the phases to operate in parallel for a short time. This is normally done to avoid gaps in phase schedules and still necessitates the completion of the prior phase primary deliverables before the subsequent phase is fully started.



Incremental

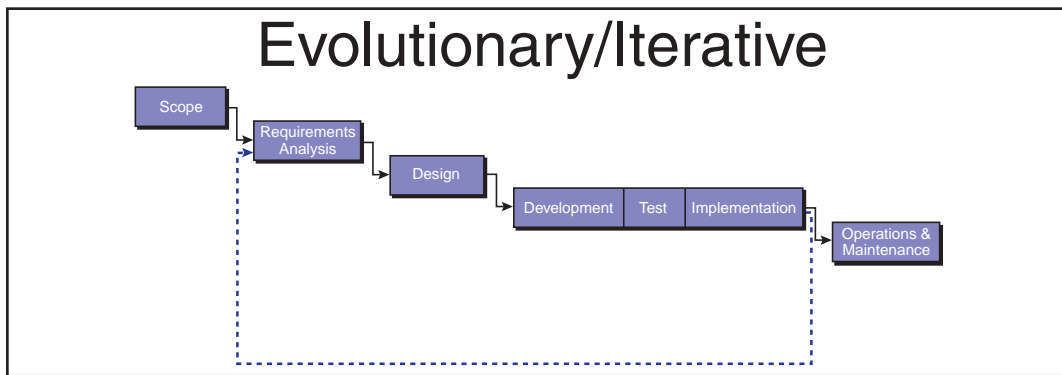
The Incremental methodology is a derivative of the Waterfall. It maintains a series of phases which are distinct and cascading in nature. Each phase is dependent on the preceding phase before it can begin and requires a defined set of inputs from the prior phase. However, as the graphic below portrays, in the design phase development is broken into a series of increments that can be constructed sequentially or in parallel. The methodology then continues focusing only on achieving the subset of requirements for that development increment. The process continues all the way through Implementation. Increments can be discrete components (e.g., database build), functionality (e.g., order entry), or integration activities (e.g., integrating a Human Resources package with your Enterprise Resource Planning application). Again, subsequent phases do not change the requirements but rather build upon them in driving to completion.



Selecting an SDLC Methodology

Evolutionary (Also known as Iterative)

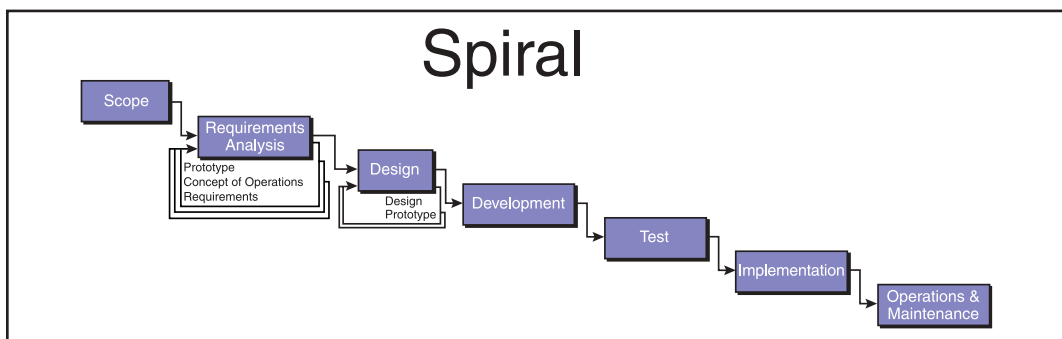
The Evolutionary methodology also maintains a series of phases that are distinct and cascading in nature. As in the other methodologies, each phase is dependent on the preceding phase before it can begin and requires a defined set of inputs from the prior phase. As the graphic below portrays, the Evolutionary methodology is similar to the Incremental in that during the design phase development is broken into a distinct increment or subset of requirements. However, only this limited set of requirements is constructed through to implementation. The process then repeats itself with the remaining requirements becoming an input to a new requirements phase. The “left over” requirements are given consideration for development along with any new functionality or changes. Another iteration of the process is accomplished through implementation with the result being an “Evolved” form of the same software product. This cycle continues with the full functionality “Evolving” overtime as multiple iterations are completed.



4

Spiral

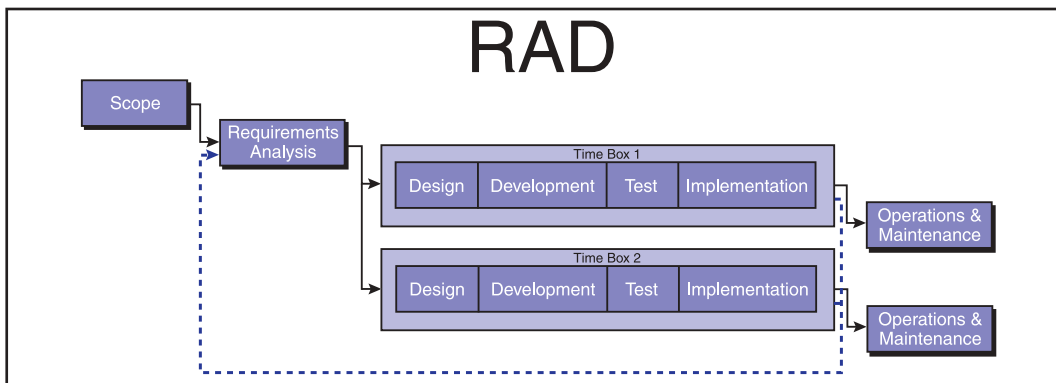
Much as the other methodologies, the Spiral methodology maintains a series of phases that are distinct and cascading in nature. As in the other methodologies, each phase is dependent on the preceding phase before it can begin and requires a defined set of inputs from the prior phase. However, as the graphic below portrays, the Spiral methodology iterates within the Requirements and Design phases. Unlike the other models, multiple iterations are utilized to better define requirements and design by assessing risk, simulating, and validating progress. The Spiral methodology also relies heavily on the use and evolution of prototypes to help define requirements and design. Prototypes become operational and are utilized to finalize detailed design. The objective of this being to thoroughly understand requirements and have a valid design prior to completing the other phases. Much like the Waterfall methodology, subsequent processes of Development through Implementation continues. Unlike Incremental and Evolutionary, no breakdown of development tasks nor iteration after implementation respectively are utilized.



Selecting an SDLC Methodology

RAD (Rapid Applications Development)

The RAD methodology is a significant departure from the other methodologies in that it is time driven rather than requirements driven. Surely requirements are what define the functionality of the software. However, under RAD, time to delivering the functionality of the software is paramount with required functionality second in importance. Similar to the other methodologies a requirements phase is completed. However, during this phase functional and technical requirements are prioritized and timeframes for delivery established. Design through Implementation then take place inside of a pre-defined “Time Box.” Requirements are addressed and functionality developed starting with the top priority requirements and working down the priority list as time allows. The length of the Time Box dictates how much functionality is developed and, therefore, which requirements are completed. As the graphic below depicts, multiple Time Boxes can be employed in an incremental fashion to achieve additional functionality. As in Incremental, these can be sequential or in parallel. Likewise, Evolutionary techniques can be employed as shown via the dashed line, to evolve functionality and achieve requirements over time.

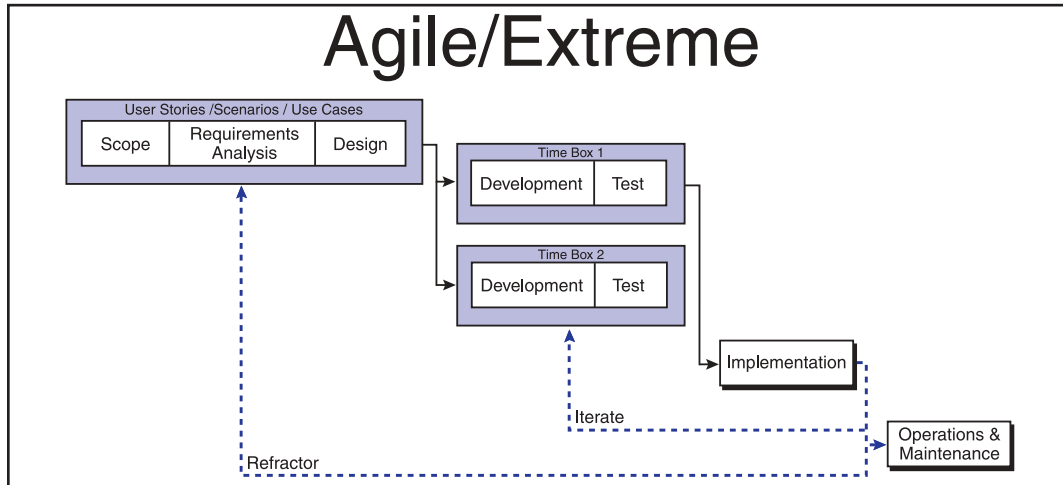


Agile/Extreme

Agile or Extreme Programming is just emerging as a potentially viable SDLC methodology. The basic tenets in philosophy are consistent with other valid SDLC methodologies and the basis in processes are founded on the Waterfall methodology as are other SDLC methodologies. However, the robustness and consistency of process have yet to be defined, standardized, and proven. Nevertheless, we will cover this emerging methodology as the others.

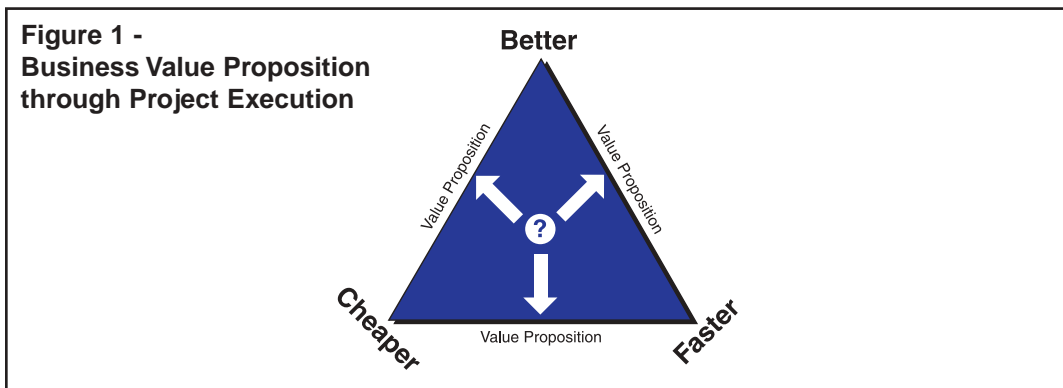
Much like the RAD methodology, Agile/Extreme is a significant departure from the other methodologies in that it too is time driven rather than requirements driven. However, rather than focusing on documenting requirements, Agile/Extreme seeks to compress time in the Scope through Design phases by utilizing Stories, Scenarios, and Use cases as replacements for formal documentation. Likewise Development and Test take place inside of a pre-defined “Time Box” as an additional means of time compression. The emphasis is on reaching Implementation as rapidly as possible. The goal is often to reach Implementation in days rather than months or years. Additionally, as the graphic below depicts, multiple Time Boxes can be employed in an incremental fashion to achieve additional functionality. As in Incremental, these can be sequential or in parallel. Likewise, Evolutionary techniques are employed to achieve additional software functionality shown via the dashed lines. Development and Test are iterated to generate rapid evolution of product functionality and “refactoring” used to redesign or re-scope the effort.

Selecting an SDLC Methodology



TARGETING VALUE PROPOSITION

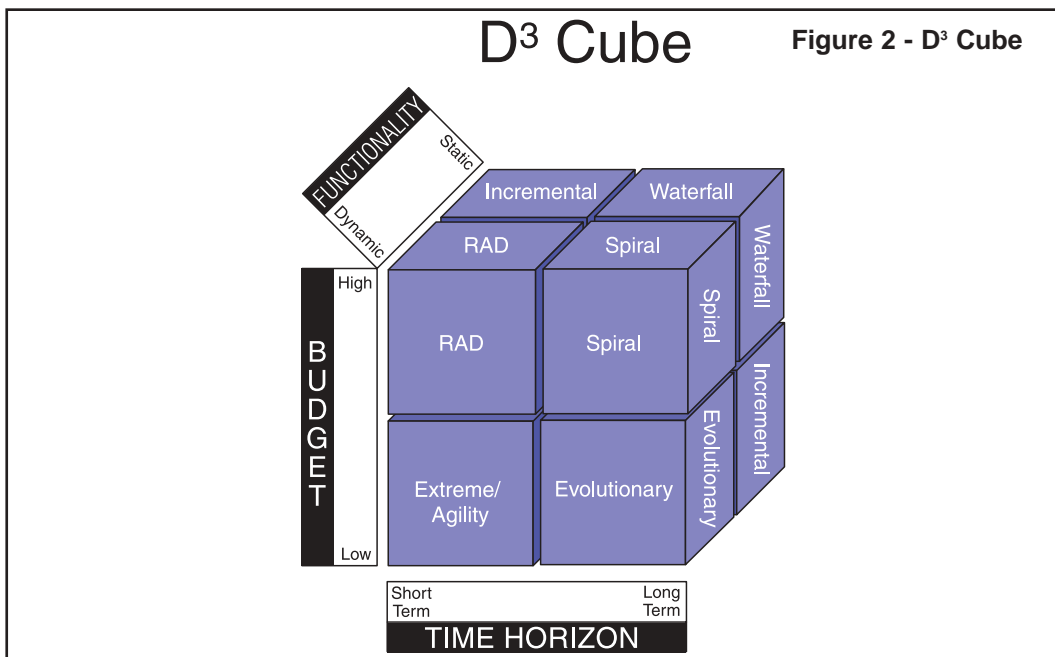
With an understanding of our methodology choices we can now discuss the selection of a methodology focused on adding value to the organization through objective measures. In order to establish an objective measure, we need to establish a common value to the business and a common approach to achieving that value. That common approach is to look at the SDLC methodology selection from the standpoint of what it will deliver. It will deliver projects. Albeit software projects, nonetheless projects which result in a product or service offering having come through the SDLC. The common value to the business is the effectiveness of the delivered project and the impact it can have on the bottom line of the business, market share or differentiation, and ultimately value to the stakeholders. Today's business world revolves around doing things better, cheaper, or faster. From a project management viewpoint the stakeholders can have any two of those they desire. But only two! The third is the price you'll pay for the other two. So if you want better and cheaper it will cost you time (you cannot have it faster), if you want better and faster it will cost you money (you cannot have it cheaper), or if you want cheaper and faster it will cost you quality (you cannot have it better). These three measures, although informal, become the value proposition of the business. Such a proposition is demonstrated visually in Figure 1.



Selecting an SDLC Methodology

Formal project nomenclature equates; Better to Quality, Cheaper to Cost, and Faster to Schedule. Such a model provides the business a mechanism to strategically manage projects.^{vi} Likewise, instantiation of these principles inside a business' Project Management (PM) Methodology is an indicator of the organization's Project Management Maturity^{vii} and directly correlates to the overall results a business will realize through its value proposition. Still, the realization of these benefits from software development projects is often illusive. This is primarily due to the lack of integration of an organization's PM methodology and SDLC methodology. Sadly, many organizations attempt to execute one to the exclusion of the other. The results being a well managed project that doesn't deliver the required functionality when needed (PM methodology only) or a system delivered yet so poorly managed that the value it was to provide was diminished or eliminated due to ineffective control of quality, cost, or schedule (SDLC methodology only). It's important to note that PM maturity and SDLC maturity are codependent in delivering on a business' value proposition. Simply put, an organizations success (value proposition) is directly linked to its project success which, in turn, is directly linked to its PM and SDLC methodologies.

With value proposition linked so tightly with SDLC methodology, it's important for organizations to select and utilize the SDLC methodology that aligns with their planned objectives. A structure for selecting an appropriate SDLC methodology can be operationalized through a graphical framework. The D³ CubeSM (Decision Cube) is a framework that provides a common approach to selecting one or more SDLC methodologies. There are three selection criteria each of which map to a specific element of business value and, when combined, form the basis for selecting an SDLC methodology to optimize value to the business. As shown in Figure 2, the three key criteria are Functionality, Budget, and Time.



Using these three criteria an SDLC methodology can be selected to best fit the needs and direction of a given organization based on its business drivers. To use the Cube simply plot the project style(s) of your organization along each of the three criteria and the resultant quadrant is the recommended methodology.

Selecting an SDLC Methodology

The key to selecting the appropriate methodology is correctly quantifying each criteria through its objective measure. Specifically, when assessing a project's **Functionality** we are making a selection based on the whether the functional requirements of your projects will be **Dynamic (changing)** or **Static (non-changing)** throughout the life cycle. Likewise we are assessing the **Budget** of your projects along a high-low continuum and the **Time Horizon** of your projects as **short term** versus **long term**. Each organization will have a different quantification of these objective measures based on that organizations risk/reward model. We'll discuss how this should be handled in more detail in the "Setting Objective Quantities" section of this paper. For now, suffice it to say, a team seeking to select one or more SDLC methodologies for an organization could perform an initial selection based on their general knowledge of the organization's predisposition of each criteria and the characteristics of their projects. To add fidelity to the decision process a confirmation of methodology selection should be performed by assessing the advantages and disadvantages of the selected methodology in light of the organizations goals, objectives, and fiscal realities.

SETTING OBJECTIVE QUANTITIES

Each criteria of the Decision Cube (Budget, Time, Functionality) is an objective measure for selecting a methodology. However, the quantification of these measures will vary by organization. Following are some guidelines for quantifying the criteria to be used in the Decision Cube for your organization:

Budget

Organization size, average project size, and return on investment (ROI) requirements are all considerations when quantifying budget for the Decision Cube. Additionally, budget can be quantified in terms of dollars, hours, resources, etc. depending upon how your organization budgets. A good guideline for determining your quantification for Low to High budget is to create a range for your projects and then place them on a 10-point scale. Start by looking at the midpoint as your break point between low and high. Then take that number and see if it fits your organizational risk tolerance. Starting with that number you can move up or down on your 10-point scale until you reach your organization's optimal break point.

For example: Your organization's projects will range from \$10,000 to \$250,000. Placing them on a 10-point scale might look like this:

- | | | | |
|----|---------------------|-----|---------------------|
| 1. | \$10,000 - 25,000 | 6. | \$125,000 - 150,000 |
| 2. | \$25,000 - 50,000 | 7. | \$150,000 - 175,000 |
| 3. | \$50,000 - 75,000 | 8. | \$175,000 - 200,000 |
| 4. | \$75,000 - 100,000 | 9. | \$200,000 - 225,000 |
| 5. | \$100,000 - 125,000 | 10. | \$225,000 - 250,000 |

The midpoint is \$125,000. Assess the \$125,000 against your organizations risk level (e.g., is spending \$125,000 a high risk venture). If it doesn't fit your risk tolerance start moving up or down the scale until it does (e.g., if \$125,000 is too high ask the question again at \$100,000 and again at \$75,000 etc.). You might decide that \$50,000 is your tolerance thus making that your break-point for the difference between a Low and High budget. Likewise you could perform the same process based on person-hours, number of resources, or whatever metric you use to budget.

Time

Time is actually referred to as Time Horizon because we are interested in knowing the projected completion of the project. The nature of your business is critical to quantifying this measure. For instance if your business is a high commodity or seasonal business you might consider 90 days to be a long-term project. Similarly, if your business is a Federal Government agency

Selecting an SDLC Methodology

or service with a high level of complexity and mission criticality you might consider 90 days to be extremely aggressive and short-term. In selecting your quantifying measure for short-term versus long-term look at the nature of your business and select a timeframe in accordance with what your business needs to succeed.

If you are having problems quantifying this for your business try applying the following formula and then adjust it according to your business constraints:

$$TH = (SD_L - LD_S)/4 + LD_S$$

Where TH is your Time-Horizon break point

Where SD_L equals the Shortest Duration of your Longest project (i.e., if we expect our longest project to finish in 1000 days but optimistically it could finish in 700 then SD_L=700)

Where LD_S equals the Longest Duration of your Shortest project (i.e., if we expect our shortest project to finish in 30 days but pessimistically it could finish in 90 then LD_S=90) Using our example: TH = (700-90)/4 + 90 or 243 days. Then assess if 243 days fits your business constraints (i.e, ROI requirements, time to market, etc.) well.

Functionality

When quantifying functionality, we suggest keeping it as black and white as possible.

Either the functional requirements will change (Dynamic) or they will not (Static).

However, reality dictates that business, and the software supporting it, will change over time. Therefore, the question is not really “if” the requirements will change but “when” the requirements will change. Consider the functionality Static if the requirements may change after the product is released for the first time. Consider the functionality Dynamic if the requirements change before the product is released the first time. Products already in production for which you are assessing changes should be assessed for the functionality of their planned next release rather than their first release and you should consider their change history carefully when making the Static/Dynamic determination.

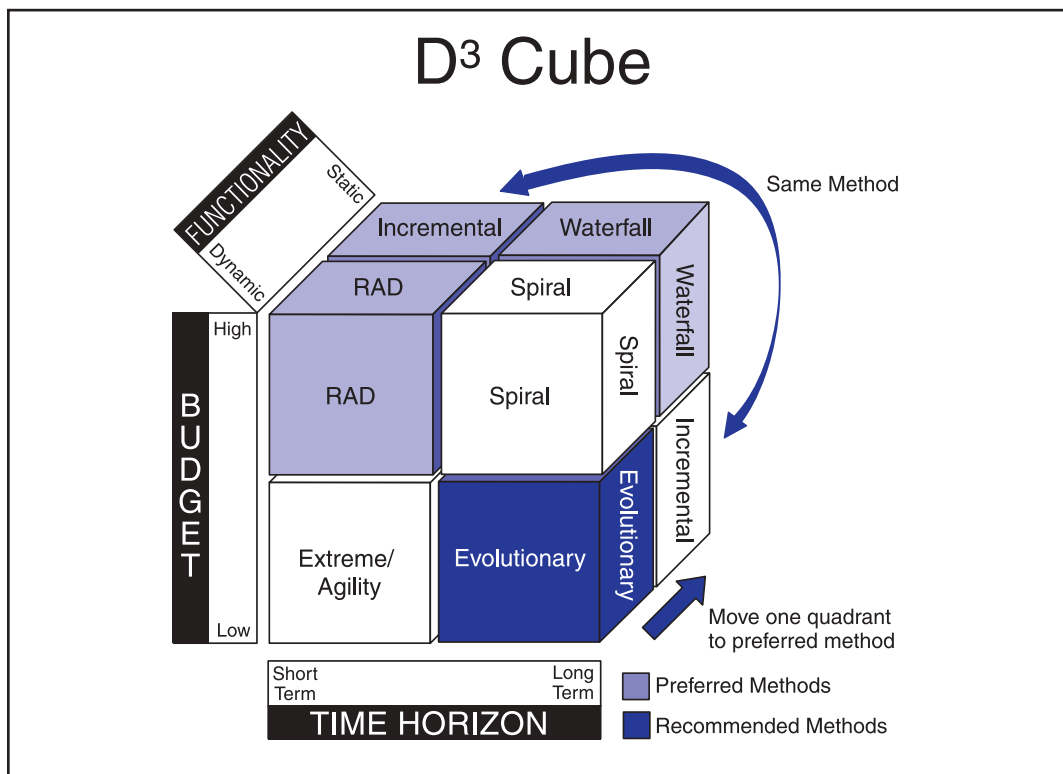
ADDRESSING ANOMALIES

We do not live in a perfect world and there will always be those projects that present themselves as anomalies to the preferred methods. When the criteria of a given project (Time, Budget, Functionality) applied to the Decision Cube results in a methodology recommendation that is not one of the preferred methodologies for your organization, you can effectively force the recommendation into one of your preferred selections by moving to the preferred methodology that is in the quadrant nearest the recommendation. In order to select the methodology most effective for your project, you should move no more than one quadrant in any direction. The result of the move is a change to the make-up of the project and requires a change to the given criteria for that project. This, in turn, will necessitate changes to the requisite project management tasks of the given project entailing organizational and team alignment with the change necessary to carry out the preferred methodology. Likewise, if you are making this assessment for all projects of an organization, consider that changing these characteristics of your organization’s projects is a significant paradigm shift and should likely require a strategic planning process be followed to align these changes with the organizational direction, goals, and objectives before selecting a methodology.

For example: If your preferred method is Incremental and the criteria of a given project result in a recommendation of Evolutionary, you would move one quadrant to the nearest preferred method which would be Incremental. In this example the project would be required to change the scope of functionality from Dynamic to Static and adjust your

Selecting an SDLC Methodology

planning accordingly. The implications for such a decision could be far reaching and effect both tactics and strategy of the organization. In our example, this may mean locking down the requirements for the desired functionality to establish a defined increment of development. Then communicating to the team, organization, and stakeholders that requirements will be held static for this increment. Making such a decision may require management authority, consensus, or investor buy-in based on the expectations around the project. Re-planning would need to take place within the project management team and a reassessment of the activity set for the project. Likewise, the project management team will need to implement mechanisms (i.e., change control processes, issue resolution processes, communication planning, etc.) to assure requirements for this increment do not change and stakeholder expectations are managed.



ADVANTAGES & DISADVANTAGES

Each methodology has its advantages and disadvantages. Once you have used the D³ Cube to select one or more methodologies, validate that the key drivers of the organization and its value proposition are consistent with the characteristics of that methodology. Look to both the advantages and disadvantages for potential critical gaps between the methodology and the business drivers. If inconsistencies exist, seek alternatives by reviewing the organizations project styles and/or seek an alternative methodology as described in the “Addressing Anomalies” section of this paper and adjust your organization’s approach to executing software projects accordingly. The table on the following page presents each methodology and its advantages and disadvantages for the reader’s consideration.

Selecting an SDLC Methodology

Methodology & Criteria	Advantages	Disadvantages
Waterfall Budget: <i>High</i> Time: <i>Long Term</i> Functionality: <i>Static</i>	<ul style="list-style-type: none"> Clearly defined stages Assures delivery of initial requirements Well documented process and results 	<ul style="list-style-type: none"> Lack of measurable progress within stages Cannot accommodate changing requirements Resistant to time and/or budget compression
Incremental Budget: <i>High</i> Time: <i>Short Term</i> Functionality: <i>Static</i> or Budget: <i>Low</i> Time: <i>Long Term</i> Functionality: <i>Static</i>	<ul style="list-style-type: none"> Early and periodic results Measurable progress Supports parallel development efforts 	<ul style="list-style-type: none"> Demands increased management attention Can increase resource requirements No support for changing requirements
Evolutionary Budget: <i>Low</i> Time: <i>Long Term</i> Functionality: <i>Dynamic</i>	<ul style="list-style-type: none"> Supports changing requirements Minimizes time to Initial Operating Capability (IOC) Achieves economies of scale for enhancements 	<ul style="list-style-type: none"> Increases management complexity IOC only partially satisfies requirements and is not complete functionality Risk of not knowing when to end the project
Spiral Budget: <i>High</i> Time: <i>Long Term</i> Functionality: <i>Dynamic</i>	<ul style="list-style-type: none"> Supports changing requirements Allows for extensive use of prototypes More accurately captures requirements 	<ul style="list-style-type: none"> Increased management complexity Defers production capability to end of the SDLC Risk of not knowing when to end the project
RAD (Rapid Application Development) Budget: <i>High</i> Time: <i>Short Term</i> Functionality: <i>Dynamic</i>	<ul style="list-style-type: none"> Minimizes time to delivery Accommodates changing requirements Measurable progress 	<ul style="list-style-type: none"> Increases management complexity Drives costs forward in the SDLC Can increase resource requirements
Extreme/Agile Development Budget: <i>Low</i> Time: <i>Short Term</i> Functionality: <i>Dynamic</i> or Budget: <i>Low</i> Time: <i>Short Term</i> Functionality: <i>Static</i>	<ul style="list-style-type: none"> Rapid demonstrable functionality Minimal resource requirements Supports fixed or changing requirements 	<ul style="list-style-type: none"> Not conducive to handling complex dependencies Creates Quality Assurance (QA) risks Increased risk of sustainability, maintainability, and extensibility

IN SUMMARY

While selecting the right SDLC methodology is challenging, the challenge is not insurmountable. With a clear understanding of the business and a framework for guidance, selecting a fitting SDLC can be readily achieved. The D³ Decision Cube is an effective mechanism for assisting in the selection of one or more SDLC methodologies. The Decision Cube assists the business in making a selection based on objective criteria specifically relevant to the business. The Decision Cube is an objective framework that allows IT and the LOB organizations to work together in the selection of a methodology that

Selecting an SDLC Methodology

brings value to the business. However, achieving success through this selection means utilizing the SDLC methodology, seeking to align this methodology with your PM methodology, and driving to maturity in both. A proper methodology in a maturing environment will enable the business to build software that assists the business in realizing its value proposition.

References

ⁱ For the purposes of this white paper we are focusing on Software Development rather than Systems Development or Integration. However, while not detailed here, the D³ Cube is equally suited for selecting the later.

ⁱⁱ While the D³ Cube focuses on selecting one or more SDLC Models, this paper utilizes the term Methodology to signify that methods and rules are operationalized inside a given Model. We will, therefore, refer to SDLC Models as SDLC Methodologies.

ⁱⁱⁱ Organizations have a myriad of structures and challenges in dealing with the sponsorship and/or governance of both the selection process and daily management once in practice. These topics are not covered in this paper. However, we are planning to address these as topics for future publication.

^{iv} Systems engineering, Commercial Off The Shelf (COTS) software package implementation, and system integration initiatives fit within these life cycles as well, but are not emphasized in our discussion.

^v We recognize that when choosing to outsource software development or assessing a vendor's methodology, objective value assessment needs to be performed. However, we will not address such in this document.

^{vi} Further details on this can be found in the Project Management Institute's (PMI) *Project Management Body Of Knowledge Guide* (PMBOK® Guide).

^{vii} For further information on Project Management Maturity we recommend reading "Project Management Maturity Model", J. Kent Crawford, Marcel Dekker, Inc. NY

About the Author

Dr. Alan E. Dillman directs all technology integration efforts for PM Solutions. He also serves as President and CEO of Park Hill Technologies LLC, a consulting firm dedicated to providing rapid business focused information solutions to commerce. Dr. Dillman's nearly 20 years of experience ranges from being the project manager for the Information System implementation of the United States Strategic Defense Initiative Organization (known as former President Reagan's Star Wars Initiative), to managing large systems projects for Fortune 500 companies.

Contributors: Karen R.J. White, PMP, Director, Resource Management, PM Solutions and Steven Jackett, Senior Consultant, Park Hill Technologies

About PM Solutions

PM Solutions is a management consulting, training, and research firm dedicated to optimizing business performance through project management initiatives. Core services include organizational project management maturity assessments, Strategic Project Office deployment and enhancement, methodology development, corporate training, project management technology integration, project portfolio management, and professional staffing and outsourcing services. PM Solutions comprehensive approach integrates business strategies with project management practices, enabling sound management decision-making and higher profitability. Headquartered in Havertown, a suburb of Philadelphia, Pa., PM Solutions has built its reputation by providing premier service to clients through the most knowledgeable and expert consultants in project management. For more information, visit www.pmsolutions.com.