Appropriate life cycles depend upon enterprise strategy

Stan Rifkin
Master Systems Inc.
2604B El Camino Real 244
Carlsbad, California 92008 USA
+1 760 729 3388 sr @ Master-Systems.com

ABSTRACT: *Selecting the best life cycle depends upon a number of factors. One of them is the market discipline or strategy of the organization. Certain strategies are aligned, that is fit, with corresponding strategies.*
Keywords: *Life cycle, strategy, fit, Discipline of Market Leaders.*

What is the best life cycle (to use in a particular case)? The answer should be, "It depends." What does it depend on? Contingencies. Well, then, what is the best why to decide on the (best) life cycle? The answer should be, again, "It depends." But the method used to decide should be a process that takes into account some contingencies. Which contingencies is a matter of research and speculation; there have been a range of suggestions.

1 SAMPLE OF PRIOR WORK

Davis et al. [1] proposed a cost-benefit method of selecting among conventional (waterfall), evolutionary, and incremental delivery life cycles. Basically, it tried to trade off functionality and delay. Waterfall, for example, could have a large – but late – function, while incremental delivery had small – but quick – functionality. See Figure 1.

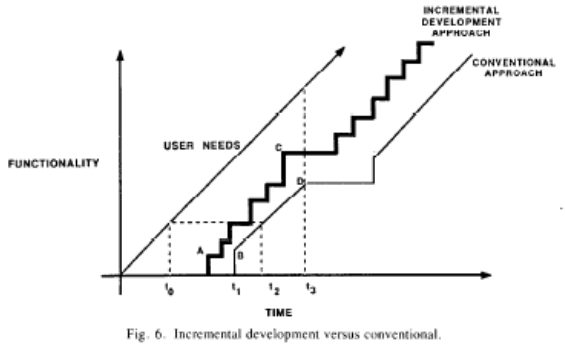

Fig. 6. Incremental development versus conventional.

Figure 1. A method of deciding on an appropriate life cycle by considering functionality vs. duration. [1, p. 1457]

Boehm [2] claims that the best way to decide on a life cycle is to use a risk driven approach where objectives, constraints and alternatives are considered. This was later operationalized into MBASE [3-5]. See Figure 2.

Boehm and Turner [6] suggest that there are five decision factors: size, criticality, dynamism, personnel, and culture. Using their framework one can tailor life cycles that range from agile to heavily plan-driven. See Figure 3.

Todd Little [7] in a time frame that is parallel with Boehm and Turner appropriated a matrix used for other purposes to characterize the mix of plan-driven and agile project management practices. It is a 2 x 2 matrix, showing the appropriate treatment (i.e., life cycle and development method) of uncertainty and complexity. See Figure 4.

2 CURRENT SPECULATION

The present paper proposes a decision process that is considerably higher in level of abstraction than those suggested so far. It depends upon the way the organization distinguishes itself among those seeking its services.

2.1 The market disciplines

We can distinguish three software product development life cycles and each corresponds to a market discipline. A market discipline is a

choice of value proposition and becomes the reason a consumer prefers one product or service compared with another provider's or even just keeping his/her money. Market discipline in our usage here is synonymous with the term strategy, that is, the long-term marketplace differentiator. Relying first on *The discipline of market leaders* [8], which in turn depended upon *Competitive advantage* [9], we identify three value propositions:

Operational excellence – Price leadership via cost leadership via excellent quality. That is, organizations that seek operational excellence have excellent quality that yields lowest cost that can yield lowest price. This is the Deming chain of logic. These organizations offer a limited menu of choices.

Product innovativeness – Unique combination of features. Organizations that seek product innovativeness hunt for product combinations to dominate the product space by creating the most interesting, seductive, and convenient products. Such products are often value priced.

Customer intimacy (total solution) – Single provider for a range of applications and their integration; one-stop shopping; "one throat to choke." Organizations that seek to become the total solution offer a broad range of products that, taken together, can address a large niche. The consumer is paying for the integration across point solutions (i.e., individual, possibly standalone, products). These organizations offer essentially an infinite menu.

Table 32: Process Model Decision Table

Objectives, Contraints			Alternatives		Model	Example
Growth Envelope	Understanding of Requirements	Robustness	Available Technology	Architecture Understanding		
Limited			COTS		Buy COTS	Simple Inventory Control
Limited			4GL, Transform		Transform or Evolutionary Development	Small Business - DP Application
Limited	Low	Low		Low	Evolutionary Prototype	Advanced Pattern Recognition
Limited to Large	High	High		High	Waterfall	Rebuild of old system
	Low	High			Risk Reduction followed by Waterfall	Complex Situation Assessment
		High		Low		High-performance Avionics
Limited to Medium	Low	Low-Medium		High	Evolutionary Development	Data Exploitation
Limited to Large			Large Reusable Components	Medium to High	Capabilities-to-Requirements	Electronic Publishing
Very Large		High			Risk Reduction &Waterfall	Air Traffic Control
Medium to Large	Low	Medium	Partial COTS	Low to Medium	Spiral	Software Support Environment

Figure 2. Decision table for deciding on life cycle model. From MBASE guidelines [4].

Each of the three strategies has its own set of characteristics in reference to software products. Operationally excellent organizations are predictable and deliver world-class quality. There are few of these in the software realm, but two are long-time standards: unmanned space flight dynamics at Goddard Space Flight Center in Greenbelt, Maryland USA, and NASA Shuttle on-board software developed across the street from the NASA Johnson Space Center in Houston, Texas USA. These organizations typically use a highly-disciplined, plan-driven life cycle that is normally called waterfall.

Product innovative organizations seek to create unique sets of features so are engaging in risky development. They consciously, tangibly manage risk and often accomplish the riskiest tasks first. Their life cycle is a spiral of ever-decreasing riskiness, where each spiral is selected to expose the technical, program, and problem space threats and opportunities and how they will be addressed. Examples of product innovative software products are games and learning programs, the software that drives high-resolution Hollywood movie cameras, and controls the drive-by-wire Mercedes S-class automobile. So are software that: autonomously lands an unmanned helicopter on a bouncing boat's deck, analyzes the Dop-

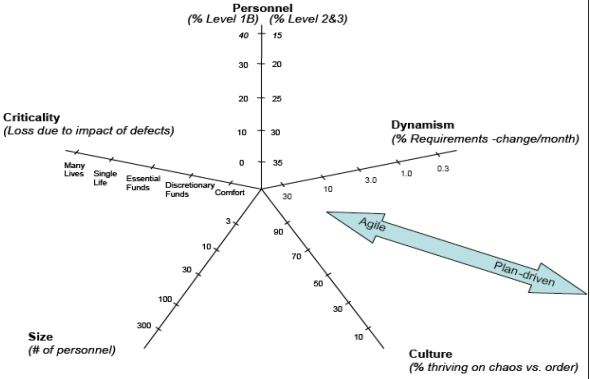

Figure 3. The five axes that determine how agile vs. plan-driven a life cycle should be. [6]

pler shift in ultrasound patterns to show the direction and pressure of flow of blood in a living body, creates in real time what looks like a 3D image from an X-ray so that a surgeon or radiologist can navigate through a patient's body with a catheter, controls a 3D cutting machine that makes a crown for a tooth from the image captured in a waiting patient's mouth by a very small digital camera, and suggests products to buy based on complex patterns of previous purchases.

Organizations seeking to develop a total solution are selling change: they will readily add or subtract features in order to tailor the product to the consumer. Examples include the current crop of customer relationship management suites and enterprise resource management suites. And the IBM Rational development environment is another example: end-to-end coverage of the job of a software development team. The software development life cycle for total solution providers is driven by architecture because that is the key to the ability to quickly and cheaply change.

It is worth stating at while the highest performing firms select a single market discipline and focus assiduously on it, all successful firms must at least reach a threshold in the other two disciplines. So, for example, total solution providers must have acceptable quality and price, but they do not distinguish themselves that way.

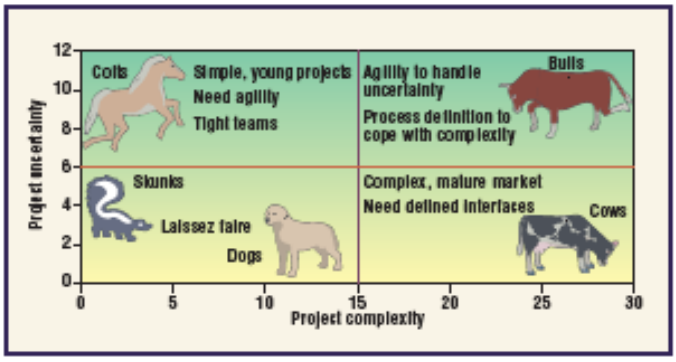

Figure 4. The selection of an appropriate life cycle depends upon uncertainty and complexity. [7, p. 31]

2.2 Their distinctive life cycles

Life cycles for all three of the market disciplines are of the phase-gate type, but only one can have pre-determined phase definitions and durations, the one for operationally excellent organizations. That is because operationally excellent organizations assume that (a) any questions during development (in the problem, solution, or project domains) can be addressed within the plan, and (b) everything will fit within the limited menu offered. And further that if the questions require additional project resources then a new plan can be reasonably created and approved in light of what has become known so far.

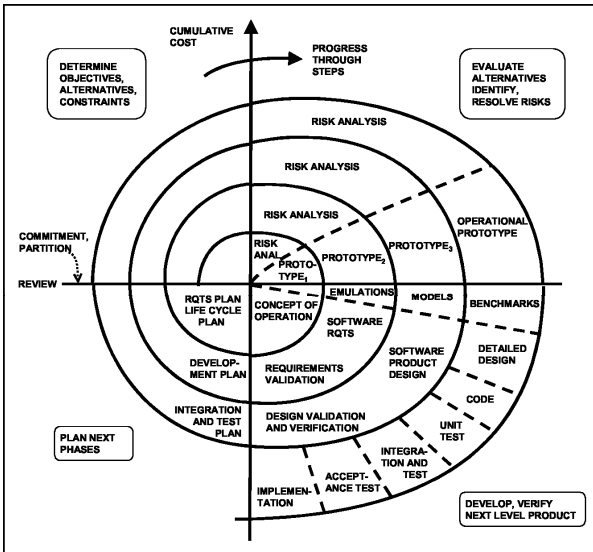

Figure 5. Original spiral life cycle model. [10]

Innovative and customer intimate organizations cannot plan as well as operationally excellent ones because their projects are intended to discover issues and prove approaches as they proceed. Therefore, the gates in innovative projects tend to ask whether the novelty is sufficient, and the gates in customer intimate solutions tend to ask whether the architecture defined so far (a) will meet the requirements for changeability, and (b) is implementable within the constraints, usually of duration and spending.

In principle, after a sufficient number of spirals both of these life cycles become waterfall, with traditional quality phase gates.

2.3 Spirals

The reason innovative and customer intimate organizations cannot plan well is that they do not know in advance the number of spirals they will have to invoke to achieve a given level of functionality, and whether that level of functionality is even possible.

A brief review of the original spiral **[10]** indicates that each rotation is meant to address a specific set of concerns a priori; and in the end the last rotation is a waterfall. See Figure 5. In practice, and as the spiral model was implemented and refined, it became evident that each rotation cannot be defined in advance, the nature of each new rotation depended upon what was learned in the last one and what new threats and opportunities were discovered. Each rotation, then, became a go/no go point where stakeholders could evaluate the progress so far in light of the risks identified and agree to continue to invest or not.

3 REAL EXAMPLES

Here are some real applications:

Operational excellence. HP OpenView, a collection of 60 components developed in parallel by about 800 people in 15 countries, regularly misses its quality gate dates, so the program manager extends the life cycle to accommodate the new, expected duration.

Product innovative. Kalisto developed Dark Earth, a game that was created by imagining that Jules Verne had come back to France in the mid-1990s and wrote the story. Like most other games, this one expected its users to have state-of-the-art personal computers and to have high regard for graphic artistry. The game was created in about two years, and then it took an additional three to translate the texts into multiple languages and tune the game so that the user experience was extraordinary. No one at Kalisto imagined that the game would take five years to complete; the underlying quasi-Verne story (that is, the functional requirements) did not change during that last three year period.

Customer intimate (total solution). Datatel provides a total back office automation solution to colleges and universities, primarily in the USA. Datatel clients each configure their systems in any one of an uncountably large number of ways. In order to make sense of these "infinite" variations Datatel has established an architecture and has promised its clients that it will adhere to the architecture so that clients can rapidly make changes by writing programs themselves. Cisco has agreed to the same condition: it will adhere to its published architecture for its Internet Operating System (the heart of its programmable routers and hubs) so that customers can make changes with confidence as long as they, too, adhere to the rules of the architecture.

4 FUTURE WORK

The approach described here needs much more field experience, particularly in mapping the anchor points among the competing life cycle models. Anchor points, to borrow a term from MBASE (Model-Based (System) Architecting & Software Engineering) [3-5], a generalization of the spiral life cycle, are life cycle definition-independent milestones for synchronization and stabilization.

5 CONCLUSIONS

There are many life cycle models to chose among. At the highest level the best way to select is to make the life cycle fit with the market discipline, the strategy, of the organization. Fit is an important and powerful attribute [11]. Fit means that an organization tailors every aspect of its operations in the furtherance of the strategy. When there is fit then all components of an enterprise contribute to the strategic success; when there is not fit then at least some component consumes energy, attention, and/or resources away from the strategic goals. The software product life can be selected so that it is among the components that fit.

6 REFERENCES

[1] Davis, A., Bersoff, E., and Comer, E. A strategy for comparing alternative software development life cycle models. *IEEE Transactions on Software Engineering, 14*(10), 1453-1461 (1988).

[2] Boehm, B. Implementing risk management.

In *Software risk management tutorial* (B. Boehm, Ed.), pp. 433-440 . IEEE Computer Society, Washington DC. (1989).

[3] Jain, Apurva. *MBASE*. http://sunset.usc.edu/cse/pub/research/mbase/.

[4] Jain, Apurva. *MBASE guidelines*. http://sunset.usc.edu/cse/pub/research/mbase/MBASE_Guidelines_v2.4.0.pdf.

[5] Metha, N. *MBASE electronic process guide*. http://sunset.usc.edu/research/MBASE/EPG.

[6] Boehm, B., and Turner, R. *Balancing agility and discipline: A guide for the perplexed*. Addison-Wesley, Boston (2004).

[7] Little, T. Context-adaptive agility: Managing complexity and uncertainty. *IEEE Software, 22*(3), 28-35 (2005).

[8] Treacy, M., and Wiersema, F. *The discipline of market leaders: Choose your customers, narrow your focus, dominate your market*. Perseus Books, Cambridge, MA (1997).

[9] Porter, M. E. *Competitive advantage: Creating and sustaining superior performance*. Free Press, New York (1985).

[10] Boehm, B. A spiral model of software development and enhancement. *IEEE Computer, 21*(5), 61-72 (1988).

[11] Burton, R. M., and Obel, B. *Strategic organizational diagnosis and design: Developing theory for application, 2nd ed.* Kluwer, Boston (1998).

7 BIOGRAPHY

Dr. Rifkin specializes in deployment, implementation, technology transition, getting best practices into actual practice. He worked at the SEI in 1988-1990, where he co-wrote the *Software Engineering Process Group Guide*, and in 2002 he issued "What I would do differently if I wrote the *SEPG Guide* today," an outline of an update to one of the most referred to SEI publications. He also wrote, "Why software engineering processes are not adopted" in 2003 for *Advances in Computers*.

Dr. Rifkin is with Master Systems Inc., an advisory services firm he started 20 years ago at the request of the National Headquarters of the American Red Cross. He is an associate editor in chief of IEEE *Software* magazine and a founding member of the editorial board of *Empirical Software Engineering*. He is a member of the IEEE, ACM, Academy of Management, and Project Management Institute. He is one of the founders of the San Diego Software Process Improvement Network, which he operates.

Dr. Rifkin has a bachelor's degree in business, a master's degree in engineering, and a doctorate in sociology.