Mark A. Hart

Launch Pad

Insights on Brooks' Law and launch

Mark A. Hart, NPDP, *Visions* Launch Editor, Founder of OpLaunch
(mark_hart@oplaunch.com)

What do you do when your launch date is approaching and there are unexpected obstacles? What if you have options to enlist more resources? This article explores the implications of adding resources and minimizing risks.

Alistair Cockburn states, "Software development is a series of goal-directed, resource-limited, cooperative games of invention and communication. The primary goal of each game is the production and deployment of a software system; the residue of the game is a set of markers to assist the players of the next game."[1] Cockburn (pronounced Coburn) describes himself as "project witchdoctor and IT strategist." He is also a coauthor of the Agile Development Manifesto.[2]

"According to...Abdel-Hamid and Madnick... the later in the project that the additional resources are added, the higher the potential for problems."

Cockburn's insight applies to new product development (NPD). It provides an approach to develop products that are validated by abundant sales at launch. It provides guidance on how to select the appropriate development resources. It embraces NPD concepts such as Product Lifecycle Management and system-level training to improve contributors' skills for the next NPD effort. Brooks' Law provides another important insight.

Summary of Brooks' Law

Frederick P. Brooks, Jr. is a software engineer, computer scientist, and the author of the highly influential book, *The Mythical Man-Month*, which was first published in 1975.[3] An oversimplified explanation of Brooks' Law is "adding manpower to a late software project makes it later." Brooks' Law provides an appropriate warning for managers of software projects that are behind schedule.

Brooks cites learning curve factors and communication factors as the primary reasons for the additional strain on software projects that are likely to cause them to fall further behind schedule. For example, training engineers new to the project diverts resources and diminishes the productivity of the established staff. Code produced by engineers new to the project is more likely to require rework because these engineers make errors consistent with their neophyte status. Communication overhead increases as more contributors are added to the project. According to the model developed by Abdel-Hamid and Madnick in the 1991 book *Software Project Dynamics: An Integrated Approach* that is described by Brooks, the later in the project that the additional resources are added, the higher the potential for problems.[4]

Sometimes, Brooks' Law is cited inappropriately as an argument for not adding resources to an understaffed project. In addition, Brooks' Law does not apply directly to contributors that perform tasks that can be easily partitioned and isolated—those tasks that do not have a significant learning curve and require minimal communication.

Brooks' Law and new product development

In NPD, an intrinsic aspect of preparing for product launch is that resources are added late in development. Therefore, the impact of Brooks' Law may be amplified while preparing for launch. Problems that can occur due to the late addition of resources or limited communication include the following examples:

- Because of terminology differences, communication between disparate disciplines (such as engineering and public relations) can be difficult.
- Because contributors are likely to be geographically dispersed, there will be multiple channels of communication.
- Since team members do often not have the same managers, there are additional learning curve and communication factors.
- Since the preparation for a successful new product launch requires contributions from many disciplines, there are additional learning curve factors—outside of traditional production—required to harmonize launch goals. Common conflicts include the sales organization requesting additional product features while engineering considers eliminating planned features in order to meet the launch deadlines.

Brooks' Law impacts launch strategy decisions. For a given obstacle, should resources be added? Or should the team make other adjustments to overcome the problem? Using a concept from martial arts can be helpful as a launch strategy is developed.

Shu, Ha, and Ri levels of mastery

Shuhari is a martial arts concept that Cockburn and others have popularized to provide software development insights. When differences in the level of mastery among team members are considered, misunderstandings are minimized. Shuhari concepts can be used in NPD to reveal insights about learning curves and to facilitate communication. This applies to intradisciplinary and interdisciplinary situations. It applies to both individual contributors and their managers, as seen in Exhibit 1.

From a discipline specific perspective, a Shu-level contributor should be encouraged to follow templates and adopt a predefined

Exhibit 1: Using the Japanese Martial Art of Shuhari to Identify Stages of Learning in NPD

Level of mastery	*Characteristics in a martial arts context*	*Characteristics in discipline specific context*	*Characteristics in a systemic, NPD context*
Shu—Follower	Beginners copy the techniques presented by their instructor. Students learn techniques advocated by a particular discipline. Students do not explore the limits of these techniques or alternative techniques.	Contributors are likely to embrace techniques consistent with numerous templates gleaned from commonly available sources within their discipline.	Management and development activities are not symbiotic. Development priorities across disciplines are not aligned. Inconsistent launch results due to sub-optimization.
Ha—Breakaway	After many years of training and after attaining a high-level black belt, students explore the limits of techniques they have learned and are allowed to explore exceptions.	Contributors select from a variety of techniques that could be used to solve a specific problem and implement the most appropriate technique based on context.	After exploring a large percentage of the disciplines required for NPD (from ideation to launch, and including Product Lifecycle Management) and mastering more than one specific contributing discipline, practitioner is able to recognize and evolve nonaligned procedures to produce more consistent, systemic NPD results.
Ri—Fluent	Performs martial arts movements automatically. Shift techniques instantaneously.	Mastery of theory, fundamentals, and application enables contributors to provide solutions to new problems in diverse environments.	Adapts to emergent conditions (such as change in market conditions or loss of key team members) regardless of development environment to deliver innovative solutions that dramatically contribute to NPD success. Inspires and mentors other discipline specific contributors to produce exceptional results.

SOURCE: The Author

Shuhari is a martial arts concept that Alistair Cockburn and others have popularized to provide software development insights. Shuhari concepts can be used in new product development to reveal insights about learning curves and to facilitate communication.

vocabulary. A Shu-level contributor is not ready to explore/evaluate multiple techniques. The productivity of a Shu-level contributor can range from mediocre to exceptional.

From a systemic, NPD perspective, a development team that operates at a Shu-level selects generalized best practices in the hope of achieving success at product launch.

Ha-level contributors are more likely to understand the guidelines and goals. They often improve the methodology. They are more likely to understand the intent of the procedures and find unique ways to contribute to the development effort. Within a given discipline, Ha-level contributors are ideal mentors for Shu-level contributors because they share a common vocabulary.

Ri-level contributors are pioneers that transcend historical methodology. They are the most likely to provide solutions to difficult problems, such as identifying critical paths and minimizing integration risks.

A contributor can be characterized with respect to their mastery within a specific discipline and their ability to impact system-level results. For example, a software developer may be a Ha-level coder but not have the understanding to collaborate with other team members to maximize NPD success.

Improving NPD communications in an agile environment

Incorporating better tools is one way to improve communications. For example, NPD collaboration tools can improve the communication of geographically disbursed teams because they facilitate:

- Multiple channels of communication (such as face-to-face, voice, video, and instant messaging)
- Management of multiple versions of documents and other files
- Sharing of current prototypes and other deliverables
- Secure communication for internal and external contributors
- Maximum effectiveness of group meetings
- Quiet and effective thinking time for developers.

The careful selection of team members can minimize communication ambiguity. The best option is to enlist highly productive resources that have experience with your team. Shared experiences minimize assimilation delay (that is, the time required for a new resource to become productive). Another option is to select resources that are better positioned to interpret your intent and your specifications. A Ha-level contributor is more likely to develop an innovative plan for a domain specific action that relates to an NPD goal than a Shu-level contributor.

Open-source paradigm and massive collaboration

Brooks' Law includes a warning about adding resources. Linus' Law advocates massive collaboration. An organization's culture determines which generalization predominates.

Brooks' Law is based on observations from traditional software development projects that include an organizational hierarchy, constrained resources, and a scheduled product launch date. Linus' Law provides insights from open-source approaches characterized by a network of contributors and the evolution of a code base. As defined by Eric S. Raymond, a computer programmer, in the essay "The Cathedral and the Bazaar," Linus' Law is "given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone."[5] This central thesis is named after Linus Torvalds, the leader of the Linux kernel project. Raymond's essay provides the following insights that relate to effective communication:

An open-source development environment makes it far easier for testers and developers to "develop a shared representation grounded in the actual source code and to communicate effectively about it."

“Brooks' Law indicates that under some conditions, it would be prudent to decline additional resources.”

Parse tasks so that the "halo developers work on what are in effect separable parallel subtasks and interact with each other very little." Ensure that "code changes and bug reports stream through the core group."

The essay also includes the following insights that relate to the mastery level of the developers:

"Good programmers know what to write. Great ones know what to rewrite."

"The next best thing to having good ideas is recognizing good ideas from your users."

Fortunately, insights from Brooks' Law and Linus' Law can be harmonized to develop a system-level perspective that maximizes the potential for a successful launch.

Managing resources that impact launch

To maximize the potential for an on-time launch, evaluate the mastery level of the proposed development resources and the communication implications. Brooks' Law indicates that under some conditions it would be prudent to decline additional resources. Alternatively, a comprehensive understanding of Brooks' Law can help project leaders identify exceptions that enable resources to be added to a project that can improve productivity.

When you are adding capability to prepare for launch, select resources with a high level of domain mastery to ensure that the widest variety of problems (known and emergent) can be resolved quickly. Selecting resources that have a sharable vocabulary and up-to-date, valuable information facilitates interdisciplinary communication. Because a product launch requires system-level contributors to select the appropriate development resources that will solve the appropriate problems in the appropriate sequence in order to deliver a product at the appropriate time, insights from Brooks' Law can guide your NPD staffing decisions. §

Endnotes

1. "Alistair Cockburn" (http://alistair.cockburn.us).
2. "Agile Development Manifesto" (http://agilemanifesto.org/).
3. Frederick P. Brooks, Jr., *The Mythical Man-Month: Essays on Software Engineering*, Anniversary Edition, (Boston, Mass.: Addison-Wesley, 1995).
4. Tarek Abdel-Hamid and Stuart E. Madnick, *Software Project Dynamics: An Integrated Approach*, (Upper Saddle River, N.J.: 1991).
5. . Eric S. Raymond, "The Cathedral and the Bazaar" (http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/).

DIVE RIGHT IN

LOOKING FOR A MORE POWERFUL CUSTOMER IMMERSION PROCESS?

Turn to Innovare for a deep dive into your customer's world:
- Voice of the Customer
- Ethnography
- Team Immersion Events
- Process Facilitation
- Training

Uncover customer needs throughout your value chain.

INNOVARE®
Grounded in Research, Driving for Innovation

www.innovare-inc.com

617-479-3818
ideas@innovare-inc.com