

Project Management in a Software Product Line Organization

Paul C. Clements, Lawrence G. Jones, and Linda M. Northrop,
Software Engineering Institute

John D. McGregor, *Clemson University*

[M]any corporations are still working without overall guidelines, policies, and procedures, leaving virtually all such decisions in the project managers' hands.

—Richard H. Thayer and Arthur B. Pyster, 1984¹

Software product line organizations have unique practices and project definitions. These unconventional features offer new challenges and directions for traditional project management.

In traditional software engineering project management, managers provide focused guidance to a team responsible for producing a specific result in a specified amount of time. Today, however, organizations are increasingly taking a product line approach to software to exploit product commonalities. How does the traditional concept of a *project*—a temporary endeavor aimed at creating a unique product or service²—hold up under this new paradigm?

Here, we discuss this question, along with how the idea of a “project” and project management techniques must expand to fit a product line context. In particular, we’ll show how the “overall guidelines, policies, and procedures” that Thayer and Pyster spoke of 20 years ago remain crucially important in product line organizations today.

Overview: Software product lines

A software product line is a group of products that share a common, managed set of features. The products satisfy the specific needs of a particular market or mission, and are developed from a common set of core assets in a prescribed way.³ Beyond simple reuse or a

component-based development strategy, a software product line lets an organization manage and evolve its product family holistically, as a single, unified entity. Product line engineering is a growing software engineering subdiscipline, and many organizations⁴—including Philips,⁵ Hewlett-Packard,⁶ Nokia,⁷ Raytheon,³ and Cummins³—are using it to achieve extraordinary gains in productivity, time to market, and product quality.

Key product line activities

Essentially, fielding a product line involves three activities:

- developing core assets,

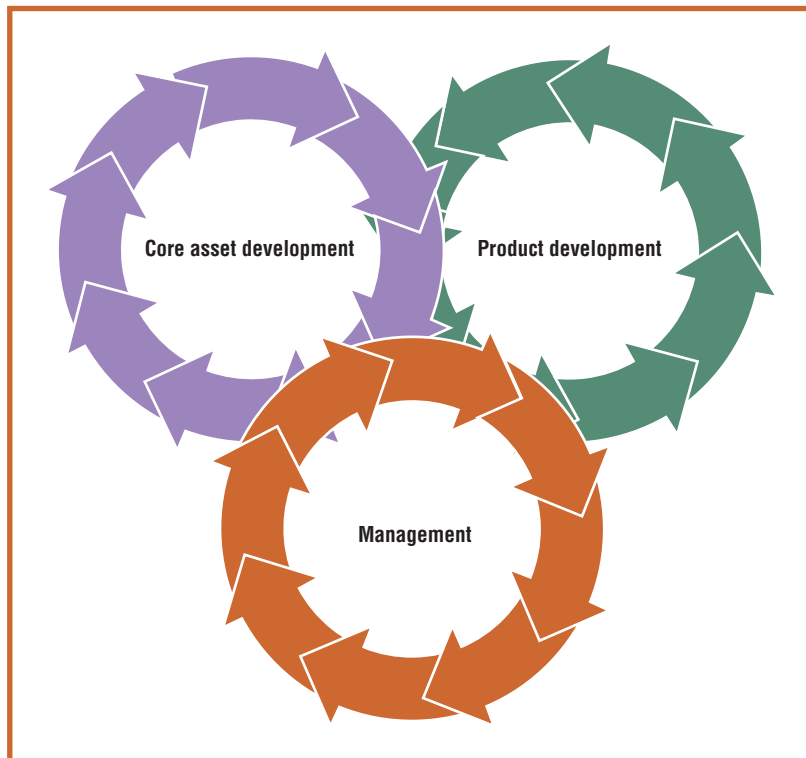
- developing products that use those assets, and
- managing these developments for the organization's overall benefit.

In figure 1, each rotating circle represents one of the essential activities. All are linked together, in constant motion, and highly iterative. Core assets, for example, are used in products, but product development often results in new or revised core assets. The figure is neutral regarding which process is launched first. In some contexts, organizations mine existing products for generic assets—such as requirements specifications, architecture, test plans, or software components—that they can use to populate their product line's core asset base. In other cases, an organization might develop or procure core assets first for later use in product production.

Management strategies

Many software product line organizations manage the activities that create and evolve core assets separately from those that create and evolve end products. The two development operations therefore constitute, in effect, separate projects. In this management strategy, *product development project* managers must understand each project's role as a core assets consumer, as well as its place in the overall product line. The supply chain for such projects is largely internal to the developing organization. At the same time, *core asset development project* managers must understand each project's role in the context of the products to be built from them. Their project's customer base is largely internal to the development organization. Managers of both kinds of projects must understand how their work supports the organization's overall product line goals.

Successful product line engineering requires management and coordination of both the core asset and product development projects to meet the organization's overall business goals. This coordination constitutes a new kind of software development "project" that's not dealt with in conventional project management practices—or, for that matter, conventional middle- or upper-management practices. Among other things, this overall coordination project must ensure that the products and core assets align and remain aligned with each other. That is, the products must make the best use of the core assets and



those assets must be useful to the products and their evolving needs.

Projects within product line development

Traditionally, a project is viewed as a temporary endeavor undertaken to create a unique product or service.² According to this definition, projects have

- a beginning and an ending point;
- clear outputs and goals (a charter for their existence);
- a clear chain of responsibility, including an owner or lead; and
- schedule and resource requirements.

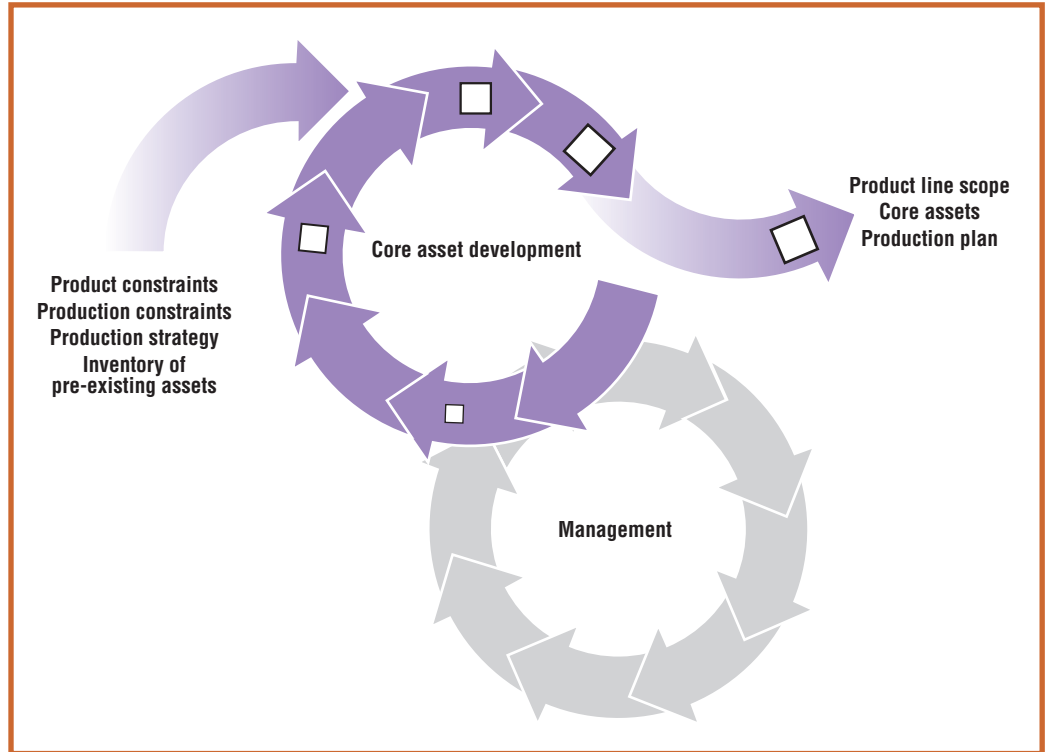
Product line organization projects (core asset development, product development, and organizational management itself) don't always match these characteristics. To be successful, managers of these projects must be sensitive to the differences.

Core asset development projects

Individual core asset development projects produce one or more reusable core assets that the organization will use to produce products in the software product line. Such projects sat-

Figure 1. Product line development. The three essential activities are core asset development, product development, and management to oversee and coordinate the process.

Figure 2. Core asset development projects. Project inputs are product constraints, production constraints, production strategy, and inventory of pre-existing assets. The outputs are a scope definition, core assets, and a production plan.



isfy the traditional definition. They might or might not be responsible for evolving these assets as new needs and understanding arise.

Beyond individual core asset projects, however, it's useful to think of the overall production of core assets as a project in itself as it's most often managed that way. Although this "project" establishes an organization's production capability for its product line, it doesn't quite fit the traditional project definition. In particular, it doesn't have a clear end point. Instead, its lifetime extends for as long as new products are produced and new core assets are developed or evolved. This type of project also requires more detailed management attention than conventional middle or higher management activities do.

Figure 2 illustrates the nominal inputs and outputs to the core asset development project. The project activity is iterative (as in figure 1), and has four inputs:

- *Product constraints*: what the products must have in common and how they vary.
- *Production constraints*: when the currently known products to be produced from the core assets are due and for whom.
- *Inventory of preexisting assets*: what legacy systems or components and artifacts

might be available from previous efforts.

- *Production strategy*: the overall approach for realizing the core assets and products.

The production strategy significantly affects how core asset projects are managed. It determines, for example, whether to build the product line proactively (starting with a set of core assets and building products from them), reactively (starting with a set of products and generalizing their components to create the core assets),⁸ or using some combination of the two approaches.

The core asset development project has three primary outputs:

- *Product line scope*: a description of the product line's existing or potential products (and, by implication, those it excludes). A scope often indicates what all products have in common and how they vary, and is often expressed in terms of product features. A proper scope definition determines what's "in" and what's "out" so that the product line is profitable.
- *Core assets*: the basis for producing product line products. These assets will typically include a software architecture and software components and their supporting

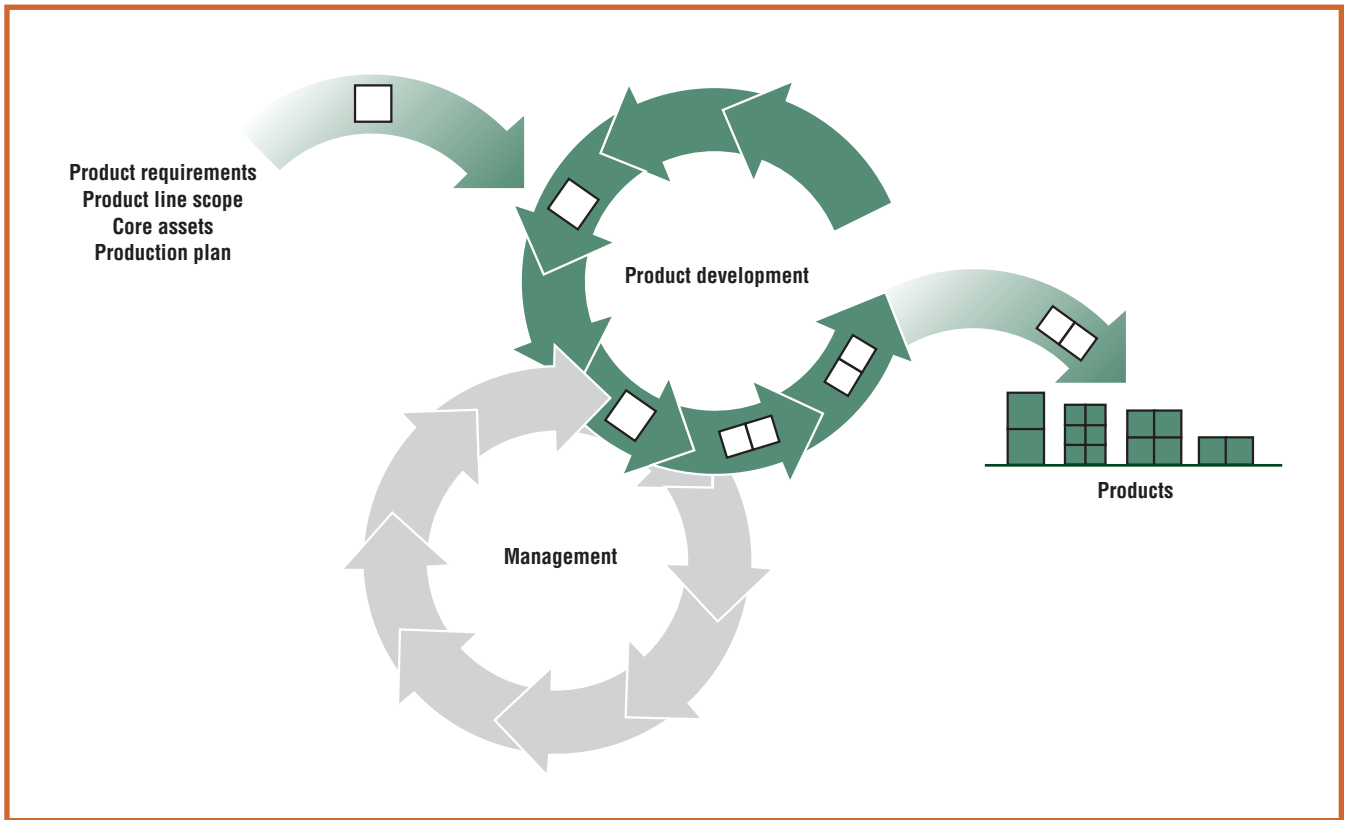


Figure 3. Product development projects. Project inputs are product requirements, the product line scope, core assets, and a production plan. The output is products.

documentation and artifacts (such as requirements specifications, design documents, and test plans). Other core assets include the business case for building the product line as well as management artifacts such as schedules and budgets.

- **Production plan:** a step-by-step description of how the products are produced using the core assets. Production plans typically vary depending on the amount of automation built into the production process.

Within the overall core asset development project, organizations might set up a subproject for creating and maintaining the product line's scope definition. Other subprojects might develop or refine one or more core assets, such as the product line architecture; develop the production plan; or establish tool support.

Product development projects

Product development's primary purpose is to produce products through the best possible use of the product line's core assets. As in core asset development, overall product development is often managed as a single project that

has the purpose of overseeing the individual product projects. Once again, this larger project has no clear ending point; it persists as long as the product line does.

Figure 3 shows the inputs and outputs. As always, iteration and feedback occur. For example, a new product design might reveal commonality with an existing product that will require new product line core assets to exploit.

As input, a product development project takes outputs of the core asset development project (and its constituent subprojects): the scope, core assets, and production plan. It also inputs requirements for the particular product, which are often expressed as a variant on a generic product description (itself a core asset).

Product line organizational management: A new kind of project

A successful software product line organization relies on close coordination among the core asset and product development projects. Product development projects must use core assets to the fullest practical extent. In turn, those core assets must be high-quality artifacts that are produced in a timely fashion and well suited for product development project use.

Some standard management practices that apply to all projects take on a different flavor in product line organizations.

Beyond mere coordination, however, a successful product line organization requires strong, visionary management. That management must continually invest resources in developing and sustaining core assets and precipitating the cultural change required to view new products in the context of the available core assets. This responsibility falls to (what we call) the *product line manager*, whose *organizational management* tasks constitute another project to be explicitly managed.

Several tasks fall under this purview:

- Ensuring the appropriate organizational units, with the right staff and resources.
- Ensuring an appropriate funding model for the creation and evolution of core assets.
- Having appropriately trained people.
- Establishing policies, process definitions, and a product line operating concept that defines how the product line development effort works day-to-day.
- Planning the organization's conversion to the product line paradigm.
- Planning and managing the product line's evolution.
- Establishing and monitoring the interaction mechanisms—such as communications, dependencies, feedback, and risk management—among product and core asset development projects.
- Establishing the organization's product line goals, as well as a measurement program to track progress in meeting them.
- Planning and managing external interfaces, particularly with customers and suppliers.

The product line manager might commission individual projects, such as a product line pilot demonstration project or a project for product line training. Product line managers play a dual role. First, they must provide the support system and constraints for individual projects. Second, they must establish Thayer and Pyster's "overall guidelines, policies, and procedures"—that is, who does what when. Leaving those items in individual project managers' hands could result in good decisions for each project that are potentially disastrous for the overall product line. A classic product line example is a project that checks out a core asset and unilaterally modifies it to suit its own special needs, rather than taking the steps nec-

essary to get the asset updated in a general way that serves the entire product line. The product line manager must enforce an operating concept with procedures that prevent this practice.

In addition, organizational management must set the software product line's overall strategic direction. As many organizations have discovered, a software product line is more than a set of products—it represents a manufacturing capability that the organization can use to tap into underdeveloped market segments. Cummins, for example, used its software product line for automotive diesel engines to enter and quickly dominate the industrial diesel engine market. That low-volume, highly specialized market is prohibitively expensive if a company must craft each product separately, but quite attractive if it can produce each product as a managed variant within an existing product family.³ Core asset or product projects managers—who are necessarily concerned with their more immediate responsibilities—cannot accomplish such strategic maneuvering.

Traditional projects vs. product line projects

Table 1 shows some differences between traditional software engineering projects and product line projects.

Additionally, some standard management practices that apply to all projects take on a different flavor in product line organizations,⁹ as table 2 (on page 60) describes. Project managers in a product line effort should recognize these differences and account for them in project planning and execution.

Case study

To illustrate the various product line projects, we offer a fictitious product line drawn from our experiences with several real clients.¹⁰ Figure 4 shows the product matrix for our fictional company, Arcade Game Maker. AGM produces three computer games for three distinct markets: personal computer freeware, wireless devices, and customizable software for company advertising. For the wireless market in particular, it must produce further minor product variations. The matrix's dimensions make some variation points obvious: game rules and interface, market constraints such as memory size and processor speed, and platform differences.

Table 1**Product line organization projects vs. traditional projects**

Projects and possible subprojects	What's involved	How it differs from traditional projects
Core asset development:	Producing the scope, production plan, and core assets; creating and maintaining the core asset base; measuring and tracking core asset production to the development organization; more oversight than conventional middle management	A continuing series of asset development and maintenance projects that must be coordinated; no predefined ending point; the customer base is internal
■ Scoping	Deciding what products are in the product line product-line context	Not usually performed as an explicit project in a non-
■ Writing the production plan	Linking attached processes	Not usually performed in non-product-line context
■ Developing individual core assets	Producing reusable assets (code and noncode artifacts)	Emphasis on reusability and support for variation as constrained and defined by the scope and architecture
Product development:	Creating products from a core asset base according to a production plan; measuring and tracking product production; providing feedback to the core asset development project	Most of the supply chain is internal to the development organization; the production plan replaces much of conventional middle management
■ Producing individual products	Assembling core assets and product-unique parts	Dependency on core asset development projects; constraints on product-unique aspects; possible dual role in creating reusable core assets; strong coordination among individual projects and core asset projects
Organizational management:	Managing the overall product line operations, including core asset and product development projects; establishing and tracking product line goals; setting new strategic directions; establishing organizational readiness	No predefined ending point; more oversight than conventional middle management; more strategic-level activities than in individual project management
■ Developing a business case	Documenting the business basis for entry into a market	Emphasis on the product line scope and market opportunities based on amortizing costs across a group of products and possibly markets
■ Product line adoption	Guiding the organization's systematic adoption of a product line approach	A comprehensive business process reengineering and technology change project with the goal of product line adoption
■ Training	Establishing the skills and knowledge necessary to support organizational goals	Product-line-specific training such as product line concepts and culture, use of architecture, production plan creation and usage, and product line operations
■ Producing a product line operational concept	Describing how the organization will do business	Different roles, responsibilities, structures, and interactions within a product line organization
■ Establishing policies for choosing among product opportunities	Describing how decisions about the product line's evolution will be made	Using business case, market analysis, technology forecasting, product line scope, and domain understanding to evaluate how to evolve the product line
■ Technology forecasting	Making sure that current and planned products are positioned to take advantage of upcoming technology trends	Consideration of technology life cycles over the usually much longer lifetimes of product lines compared with individual development projects; managing and coordinating technology insertion over a suite of projects

**Organizational management project:
Product line adoption**

Rapidly expanding game platform options left AGM's vice president for product development (VPPD) facing potential disaster. Product variations were greater than ever before, and

the time frame for developing new products was shrinking. To survive, AGM needed to increase both asset reuse and productivity.

To this end, the VPPD made the bold—and ultimately successful—decision to suspend the existing portfolio of product development

Table 2

Management practices that apply to all projects

Cross-cutting management practice	What it does	What's involved
Configuration management	Protects the integrity of core assets and products	Creating a unified system for asset sharing among products; multiple versions of multiple products, each using multiple versions of core assets; one group creates assets for parallel usage by other groups
Customer interface management	Manages interactions between the organization and external customers	Managing customer expectations and understanding of the product line approach's benefits and constraints
Data collection, metrics, and tracking	Uses data to guide the project	Creating three different metrics suites on quality and staff productivity: core assets, products, and overall product line organizational management
Funding	Establishes the finances necessary for successful project operation	Balancing allocation of funds between product development and core asset creation and evolution; funding the production infrastructure for producing products from core assets; investing "start up" funds at the correct moment for maximum return
Planning	Plans the basis for project execution and tracking	Producing various plans, such as product line adoption plans, core asset funding plans, transition plans, core asset development plans, production plans; different constraints and interdependencies
Risk management	Identifies and manages risks before they become problems	Managing interdependent risks resulting from project interdependencies
Structuring the organization	Creates a structure of units and projects to efficiently implement the product line	Creating unique roles, goals, responsibility scopes, and notions of what defines a project; assigning responsibility for core asset production and evolution

Figure 4. Arcade Game Maker's product matrix. The fictional AGM company produces three games for three distinct markets.

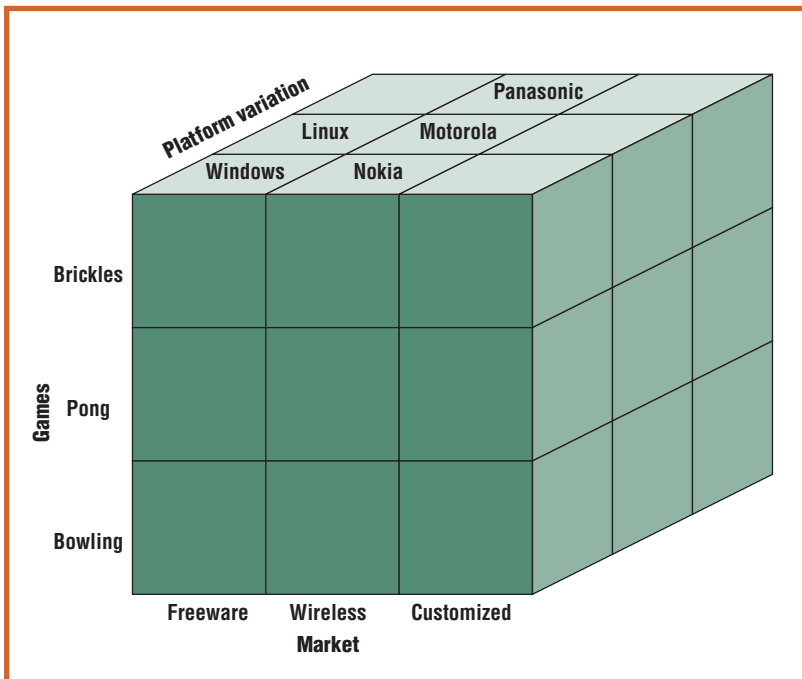
projects. She then chartered an organizational management project to initiate several adoption actions aimed at creating the AGM product line. (The real-life counterpart to this example is the Cummins software product line

for diesel engines.³) The project manager selected the "What to Build" product line pattern, described in *Software Product Lines: Practices and Patterns*,³ as the project's charter definition (see figure 5). Each element in the pattern directed the project staff to specific practice areas to guide their activities. This project chartered individual core asset development projects, including one to build a business case and another to define the product line's scope.

The adoption project was considered a success after the first three products were delivered on time and met quality standards. In addition, the company had developed a set of core assets that would make future deliveries even faster. The adoption project team coordinated the initial product and core asset development projects, providing input especially to projects that defined the product line's operational concept and measurement program. After that, the adoption project concluded.

Core asset development project: Defining a production plan

AGM chartered several core asset development projects, including an architecture proj-



ect and several component development projects. During the initial product development project, developers realized that they didn't have enough information about the core assets to use them effectively. The product developers discussed this with the core asset builders (the product development circle's arrows in figure 1 represent such feedback).

In response to the feedback, AGM chartered another core asset development project. Its task was to develop a production plan that would explain how to select the appropriate core assets, exercise their built-in variabilities, and use them to produce products. (The real-life counterpart to this example is the US National Reconnaissance Office's Control Channel Toolkit.³) This new project was staffed by the product development project team members who recognized the problem and by members of the architecture project. The Brickles development project gave the resulting production plan its initial validation.

Product development project: The Brickles wireless product

In a software product line efforts, when to begin chartering product development projects is a critical decision. For its first development project, AGM planned to develop the Brickles game for wireless devices. Although AGM had much game domain experience, it had no software product line or software architecture experience.

To contend with this fact, the VPPD decided on a reactive production strategy⁸ in which the company would charter the Brickles project as soon as the scope document's first version was ready. (The real-life counterpart to this example is Salion's product line of revenue acquisition management software.¹¹) AGM also chartered a series of core asset development projects concurrently with the product development project so that the product and a preliminary set of core asset projects would complete at roughly the same time. In a reactive regime, one of the product development project's responsibilities is to continually integrate the core asset projects' efforts. This approach allowed AGM to continue product development and delivery, while laying the foundation for future products in the product line.

The scope of the original core asset project charters didn't encompass the full behavior range that all product line products would

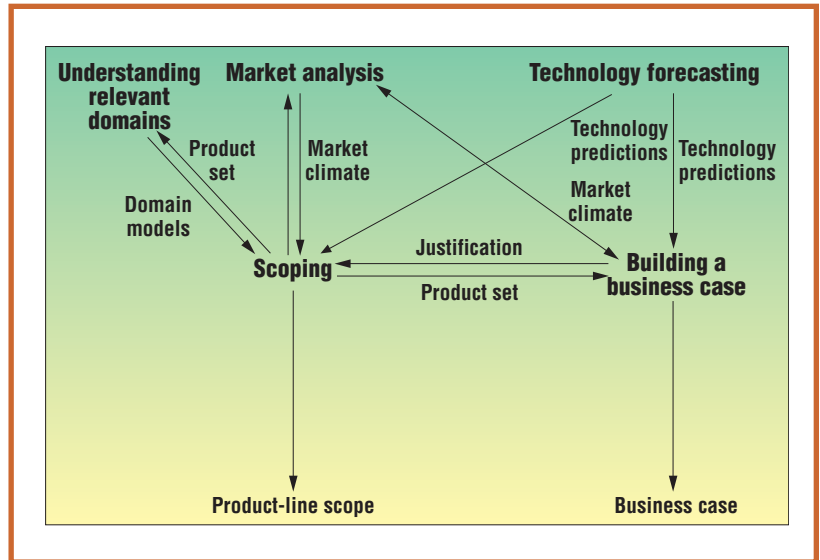


Figure 5. The "What to Build" pattern. Each element represents a specific practice area (bolder font) to guide project activities.

eventually require. Each charter was limited to the behaviors needed by the product under development. AGM would charter additional projects over time to refactor the architecture and renovate core assets for use in other product development projects. This approach limits each project's complexity and reduces the risk of the project failing to achieve its charter regarding schedule, budget, and quality standards.

This reactive approach reduced AGM's upfront investment but required closer coordination within its larger project portfolio. The company achieved this coordination partially by using certain extreme programming¹² practices, including

- daily builds in the product development projects,
- continuous testing in the core asset projects, and
- explicit contracts for the interfaces between modules.

This approach reduced the need for communication among projects, while validating that the coordination was successful. As the company's core asset base has matured, the number of concurrent projects necessary to produce a product has drastically decreased. The architecture's refactoring and the resulting refactoring of the other core assets have stabilized; they now occur when new customer opportunities arise outside the product line's current scope.

About the Authors



Paul C. Clements is a senior member of the technical staff at Carnegie Mellon University's Software Engineering Institute, where he's led projects in software product line engineering and software architecture documentation and analysis since 1994. He received his PhD in computer science from the University of Texas at Austin. He is coauthor of *Software Product Lines: Practices and Patterns* (2001), *Software Architecture in Practice* (1998; 2nd ed. 2003), *Evaluating Software Architectures: Methods and Case Studies* (2001), and *Documenting Software Architectures: View and Beyond* (2002), all published by Addison-Wesley Professional. Contact him at clements@sei.cmu.edu.


Lawrence G. Jones is a senior member of the technical staff in the Product Line Systems Program at Carnegie Mellon University's Software Engineering Institute. He received his PhD in computer science from Vanderbilt University. He's a member of the IEEE, vice-chair of the ABET (Accreditation Board for Engineering and Technology) Computing Sciences Accreditation Commission, and an ACM Director on the Computing Sciences Accreditation Board. Contact him at lgj@sei.cmu.edu.



John D. McGregor is an associate professor of computer science at Clemson University and a visiting scientist at the Software Engineering Institute. His research interests include software product lines, design quality, testing, and measurement. McGregor is coauthor of *A Practical Guide to Testing Object-Oriented Software*, forthcoming from Addison-Wesley. He is a member of the IEEE Computer Society and the ACM. Contact him at johnmc@cs.clemson.edu.

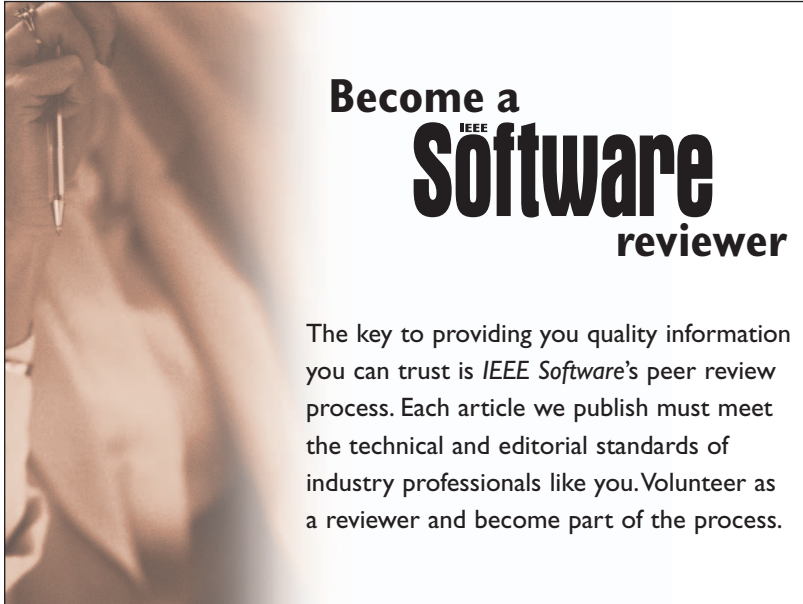
Linda M. Northrop is director of the Product Line Systems Program at Carnegie Mellon University's Software Engineering Institute. Her research interests include software architecture, software product lines, predictable assembly from certifiable components, aspect-oriented system development, and ultra-large scale systems. She is coauthor of *Software Product Lines: Practices and Patterns* (Addison-Wesley Professional, 2001). She is a member of the IEEE Computer Society and the ACM. Contact her at lmn@sei.cmu.edu.



Although the basic knowledge areas for traditional project management apply, product lines require specifically targeted management practices and techniques, recognition of new kinds of projects, and closer coordination among projects. Knowing this, software project managers must examine their existing practices to determine how they add or diminish value in product line development and how they might be modified to succeed in a product line context. 

References

1. H. Thayer and A.B. Pyster, "Software Engineering Project Management," *IEEE Trans. Software Eng.*, vol. 10, no. 1, 1984, pp. 2-3.
2. Project Management Inst., *A Guide to the Project Management Body of Knowledge*, 2000; www.pmi.org/prod/groups/public/documents/info/pp_pmbok2000welcome.asp.
3. P. Clements and L. Northrop, *Software Product Lines: Practices and Patterns*, Addison-Wesley, 2002.
4. Software Engineering Institute, *Product Line Case Studies*, 2004; www.sei.cmu.edu/plp/plp_case_studies.html.
5. P. America et al., "CoPAM: A Component-Oriented Platform Architecting Method Family for Product Family Engineering," *Software Product Lines: Experience and Research Directions*, P. Donohoe ed., Kluwer Academic Publishers, 2000, pp. 167-180.
6. P. Toft, D. Coleman, and J. Ohta, "A Cooperative Model for Cross-Divisional Product Development for a Software Product Line," *Software Product Lines: Experience and Research Directions*, P. Donohoe, ed., Kluwer Academic Publishers, 2000, pp. 111-132.
7. A. Heie, "Global Software Product Lines and Infinite Diversity," keynote address, Second Software Product Line Conference, 2002; www.sei.cmu.edu/SPLC2/keynote_slides/keynote_1.htm.
8. P. Clements and C. Krueger, "Point-Counterpoint: 'Being Proactive Pays Off' and 'Eliminating the Adoption Barrier,'" *IEEE Software*, vol. 19, no. 4, 2002, pp. 28-31.
9. P. Clements and L. Northrop, *A Framework for Software Product Line Practice*, version 4.2; Software Engineering Inst., Carnegie Mellon Univ., 2004; www.sei.cmu.edu/plp/framework.html.
10. J. McGregor, *Arcade Game Maker Product Line*, 2004; www.cs.clemson.edu/~johnmc/productLines/example/frontPage.htm.
11. P. Clements and L. Northrop, *A Software Product Line Case Study* (CMU/SEI-2002-TR-038, ADA412311), Software Engineering Inst., Carnegie Mellon Univ., 2002; www.sei.cmu.edu/publications/documents/02.reports/02tr038.html.
12. K. Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 2000.



**Become a
IEEE
Software
reviewer**

The key to providing you quality information you can trust is *IEEE Software's* peer review process. Each article we publish must meet the technical and editorial standards of industry professionals like you. Volunteer as a reviewer and become part of the process.

Become an *IEEE Software* reviewer today!
Find out how at www.computer.org/software.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.