

Product Lines in Practice at Three Major Corporations

Bill Thomas

Cummins Engine, Raytheon, and Hewlett-Packard all knew that they had a lot to gain from product line practice, and they were right: The companies have enjoyed substantial reductions in time-to-market, cost, and risk, and significant gains in efficiency and quality. But significant technical and cultural changes are also required, and all three companies have learned some valuable lessons from their product line programs.

Cummins Slashes Development Time, Costs

Cummins Engine Company launched a product line program for software in 1994, beginning with one legacy system for a diesel engine and modifying the software to extend it over several other engines.

Using product line practices presented difficult challenges on several fronts, says Joseph Gahimer, director of the Core Controls Group, which develops all core assets for embedded controls software and hands them off to application teams that apply the core assets to specific products.

First there is the complexity of Cummins's mix of engines, which range from light-duty engines (less than 200 horsepower) to high-horsepower engines (6,000 horsepower) used in a variety of applications, such as pickup trucks, large trucks, mining, rail, construction, power generation, and the military. Cummins, which has annual sales of more than \$6 billion, sells to a variety of original-equipment manufacturers who all require varying degrees of customization to meet particular specifications. In addition, Cummins engines must conform to varying emissions regulations throughout its worldwide markets.

Prior to 1994, one team would develop each engine, including all the software for that engine. At that time, Cummins managers looked at the company's projections for new engine products and new features and calculated that it would need far too many engineers to develop enough new products to satisfy the expected demand. "The company realized that it would need to do something to get a lot of reuse" from one product to the next, Gahimer says.

Cummins wanted not only to develop many new engines; it also wanted to improve its time to market with those engines. A common reuse base would help that, too. Finally, customers were asking for more uniformity in the look and feel of Cummins's engines, while still demanding customization for different products.

Five years later, Cummins has declared its original product line program a huge success. Before using product lines, it took Cummins 150-250 person-months of effort to develop software for the electronic control module (ECM) just to run an engine.

“It’s an embedded system, and we had to reinvent everything. Using core software assets, a lot of building blocks are available, and it’s a matter of assembling the building blocks and modifying them.” Cummins can now develop new ECM software to run an engine in less than 10 person-months. “In one instance, where we were applying the same fuel system across engines, we were able to build the software to run the new engine in just three days,” Gahimer says.

“Seventy-five percent of our product software originated from the core assets,” he adds. “That’s an incredible amount of reuse and it has saved a lot of cost across the company. Also, the reliability and quality are very good.”

Now that product line practices have proven so successful for software, Cummins is embarking on a second-generation effort, and is extending the product line concept to other areas of the company. “The company has seen the benefit and is saying, ‘Hey, where else can we apply this?’” Gahimer says.

Government Project Leads to Commercial System at Raytheon

Managers in Raytheon Company’s Space Systems Division examined the company’s satellite command and control systems and saw that there was a high degree of overlap in software capabilities. Although the company did not have a product lines program in place at the time, Raytheon believed it could significantly reduce cycle time, cost, and risk by developing software only once and using it across its government and commercial markets.

Raytheon’s first product line effort, begun in August 1997, was a joint initiative with a U.S. government agency and was named the Control Channel Toolkit (CCT) program. The goal was to develop a reference architecture, extendible components, and reuse documentation that would cover most software requirements for satellite command and control on government satellite ground-system programs. In fall 1997, the CCT program brought in technical staff from the Software Engineering Institute’s Product Line Practice Initiative to serve as consultants on the program.

The effort was proven successful on the first reuse of the CCT product. Compared with its original bid, which called for largely creating the new system from traditional “code cloning” reuse, Raytheon reduced by 80 percent the lines of code engineers had to write. “That was a reaffirmation that by coming up with a common core set of capabilities—if

we do the domain engineering right—a large majority of the capability should be there,” says Jeffrey Shaw, who served as program manager for CCT.

Shaw says thorough domain analysis is critical—not just for identifying common capabilities but also the hotspots where things tend to always vary among spacecraft and missions. Knowing about the variability helps Raytheon engineers develop well-engineered “variation points” that can be easily adapted to new applications, which helps to further reduce costs and cycle time.

Shaw points out that in reusing CCT, Raytheon engineers spend significantly more time on software architecture and design work, but afterward the implementation work proceeds relatively quickly. Based on that experience, Raytheon is beginning to adjust its cost and bidding models.

From the beginning, Raytheon planned to extend the experience it gained with CCT by applying that experience later to commercial products.

Shaw has since moved on from the CCT project to become product line manager of Eclipse, which, as Raytheon intended, will be a commercial application of the principles learned with CCT. The existing Eclipse software architecture, which is object-oriented and modular, will be migrated to a true product line architecture.

Unlike CCT, which is a toolkit with a reference architecture, Eclipse is an end-to-end system. Raytheon’s transition plans call for moving components from the existing system, one at a time, into a product line architecture. “One key in any product line plan is to get the common baseline under control, and to then have a well-planned strategy for migration.”

Evolving Platforms at Hewlett-Packard

Hewlett-Packard Company has long developed hardware products on common platforms, but platform development for software is relatively new—within the past 10 years for many applications, says Emil Jandourek, the Practice Area Section Manager within HP’s Product Generation Solutions (PGS) group. PGS partners with HP’s product divisions to help them evolve their product-generation capabilities and competencies.

The company’s product line approach for software involves whole products, such as families of printers, and tends to be very effective when it drives products that have multiple simultaneous releases and involves rapid time-to-market schedules.

As of 1997, core assets made up 89 percent of the software in one line of HP printers. The other 11 percent consisted mostly of adaptations that matched the software to a particular printer in the line.

A common Hewlett-Packard strategy is to evolve new features into an existing platform. For example, duplexing (printing on both sides) was originally enabled on only one printer, but is now available to all printers, though not all printers use it because they lack the necessary hardware.

“We’re doing things on the hardware side, coupled with the software,” Jandourek says. “An overall platform approach is emerging in both disciplines: one that leverages what you have and extends it forward. In multiple simultaneous releases, you can reduce effort between products and have a shorter time to market.”

The next step in HP’s product line approach involves “an overall shift from a more monolithic, integrated, and tightly coupled asset base to a modular component-based architecture with corresponding assets,” Jandourek says. This component-based approach is more efficient and gives developers more flexibility. The challenge is to correctly establish the boundaries for the components.

No Universal Prescription

Jandourek points out that the product line approach “is not a universal prescription.” For example, there would be too many unknowns with a startup product for a new customer base. It would be a poor strategy to delay that product in order to develop a product line platform because the product could fail—and perhaps take the entire organization down with it. “But when it’s a product category that is well established, even if it’s new, you could design it to be platform-based.”

For Cummins, the greatest challenge has been to conduct good domain analysis—a process that only comes with experience. “With product-line engineering, until you do it for the first time, you don’t have a feel for what it takes to do it,” Gahimer says. “Our domain experts conducted domain analysis based on their experience, but we identified many improvements later. So the lesson is that you need thorough domain analysis to help you anticipate future needs. Still, you can’t forecast or project all the different uses for the product line until you go through it.”

Cummins also learned that it is important to get the workflows and rules established, and the necessary tools and training in place for the product line effort, which Gahimer says make up the infrastructure that keeps the effort moving over the long term.

The product line effort also requires discipline. “There are a lot of conflicting pressures during development,” Gahimer says. There could be a desire to quickly get software written for a specific application, and some members of the organization might not want to take the extra time to make that software available for common use. But unless the software is available for common use, the next group will also have to start from scratch.

Finally, Gahimer says, the architecture for software product lines has to be portable and able to move between different products and technologies because the lead time to create a new architecture is often longer than the company’s ability to predict the market’s needs.

Raytheon’s Shaw also emphasizes the importance of the product line’s software architecture. “The whole key is architecture,” Shaw says. “We’ve gone through object modeling, object orientation, component-based reuse—but those were baby steps. It’s not until you have a reusable architecture that provides a context for all those enabling technologies that you can reap the benefits that everyone’s been trying to get for 20 years.” With that architecture now in place, Shaw says, “we’re excited. We’re seeing the bottom-line business numbers.”

Management Champions and Cultural Change

Gahimer also points out that Cummins’s program might have failed without a champion in senior management, which for Cummins was the vice president of electronics. “He had the vision, the authority, and the persistence to establish the first generation concept and keep it flowing.”

Raytheon has also learned that the organization and its culture have to change to support product line practices. Shaw explains that with Eclipse, Raytheon is marrying two architectures: a high-end system architecture and an underlying product architecture. On one hand, the product line organization is looking for core assets to be used across products, but on the other hand, individual programs are used to developing “stove-piped” systems, with a single end use in mind. “The biggest question is: How do we integrate processes and have cultural change to get the overall efficiencies that we need from product line practice?” Shaw says.

Such efforts can expose underlying concerns, even at the best companies. “Stove pipes typically involve someone’s sphere of control,” Shaw says. “Despite everyone’s desire to be good company people, there is a lot vested in the current organization.” Shaw says the solution is to have alignment at the senior-management level. “It flows down from there. Everyone has to see that there are incentives to adopting the new culture. If they think they will end up as losers, they’ll fight it.”

Copyright © 1999 by Carnegie Mellon University

The Software Engineering Institute (SEI) is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University.

SM Architecture Tradeoff Analysis Method, ATAM, CMM Integration, CMMI, IDEAL, Interim Profile, Personal Software Process, PSP, SCE, Simplex, Team Software Process, and TSP are service marks of Carnegie Mellon University.

® Capability Maturity Model, Capability Maturity Modeling, CERT, CERT Coordination Center, and CMM are registered in the U.S. Patent and Trademark Office.