Measures for Software Product Lines

Dave Zubrow
Gary Chastek

October 2003

Software Engineering Measurement and Analysis Initiative

Technical Note
CMU/SEI-2003-TN-031

Contents

List of Figures

List of Tables

Acknowledgements

The authors wish to thank Wolfhart Goethert and Grady Campbell for reviewing previous versions of this technical note.

Abstract

This technical note characterizes the status of measurement associated with the operation of a software product line, suggests a small set of measures to support its management, and provides guidance for those establishing measurement activities within a software product line. It is intended to help managers of software product lines develop a set of base measures for tracking those categories of needs most relevant to their organization's products, projects, and processes. The measures suggested here range from relatively mature to those whose general utility have yet to be validated. Therefore, an organization using this paper needs to assess its ability to generate the measures and the value they are likely to return to the organization. In most cases, an organization may wish to start with a subset of the measures described.

1 Introduction

1.1 Overview

This technical note characterizes the status of measurement associated with the operation of a software product line, suggests a small set of measures to support its management, and provides guidance for those establishing measurement activities within a software product line. The first section provides some background on software product lines and defines managerial roles and responsibilities. The management information needs associated with the software product line management roles form a matrix that is utilized to identify relevant measures. The second section presents product line measures for the information needs associated with the management roles. The selection and definition of these measures heavily relies on existing literature describing software project management and software reuse. The third section presents several measures that are used to characterize the performance of projects to produce core assets, the value of these assets to product development projects, and ways of managing core assets. The final section offers recommendations for improving the development and use of measures for the acquisition and development of systems based on software product lines.

1.2 Background

A *software product line* is a set of software-intensive systems sharing a common, managed set of features that satisfies the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way [Clements 01]. A product line's *scope* is a description of the products that constitute the product line or what the product line is capable of producing. Within that scope, the disciplined reuse of *core assets*, such as requirements, designs, test cases, and other software development artifacts, greatly reduces the cost of product development.

Software product lines capitalize on the commonalities and bounded variabilities among similar products, and as such can address problems such as

- dissatisfaction with current project/product performance
- need to reduce cost and schedule
- complexity of managing and maintaining too many product variants
- lack of staff
- need to quickly respond to customer / marketplace demands

A key component enabling the effective resolution of these problems is the use of a *product line architecture* that allows an organization to identify and reuse software artifacts (e.g., code and test case) for the efficient creation of products sharing some commonality, but varying in known and managed ways. The architecture, in a sense, is the glue that holds the product line together.

Measurement in a software product line[1] must address the following:

- the development of core assets and their use to produce products
- characteristics of product development project performance
- overall product line performance [Clements 01]

These are reflected in the spheres of management responsibility within the product line.

1.3 Management Responsibility in a Software Product Line

While there are multiple ways to structure a product line organization [Clements 01], the operation of a software product line typically involves three spheres of management responsibility:

- management of the product line: the overall business endeavor of the enterprise
- management of core asset development: the development and sustainment of reusable core assets and infrastructure
- management of product development: the development and sustainment of individual products

In the organizational scheme shown in Figure 1, the product line manager is responsible for the overall operation of the software product line (the functions within the box).[2] Asset development can occur either within the context of product development (as indicated by the box surrounding Product A) or independently (as part of product line sustainment). Finally, products are developed (as shown in the dark gray oval) by using the core assets. The arrows show the utilization of the core assets in product development and the changes to the core assets resulting from product development, and product line sustainment efforts.

[1] "Data Collection, Metrics, and Tracking" is one of the practice areas in the Product Line Framework and is viewed as necessary for successful product lines [Clements 01].

[2] This figure is adapted from a similar figure by Bergey & Goethert [Bergey & Goethert 01].

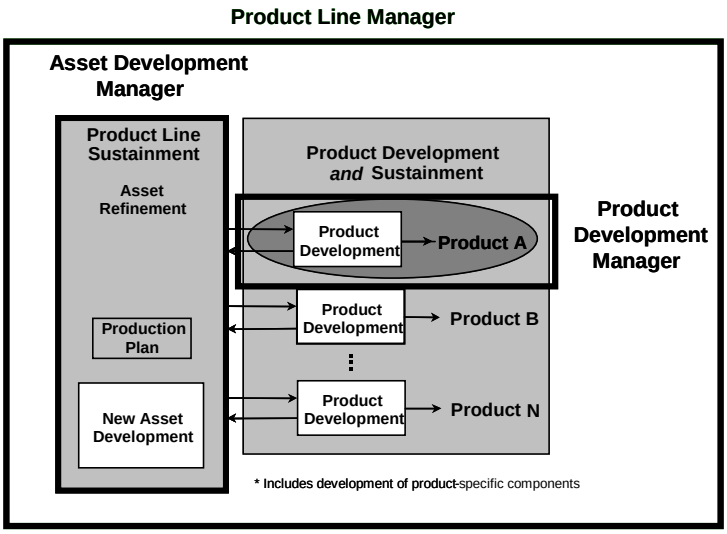

Figure 1: Product Line Management Roles

Within a product line, there may be multiple managers fulfilling each role, or a position could exist that supports all of these roles. For ease of description and explanation, each role is discussed separately.

A *product line manager* is responsible for the overall product line business enterprise and, therefore, is concerned with the entire set of past, current, and future products that make up the product line. The scope of a product line is generally characterized in terms of the targeted market or mission of the product line organization, and is identified by representative customers and the types of problems that these particular products are meant to solve. Furthermore, the product line manager should be able to demonstrate the benefits associated with this particular approach to software development to senior management and the rest of the organization.

A *core asset development manager* is responsible for the set of core assets and associated infrastructure that enables the systematic development of the products in a product line. All products in the product line are built using the core assets and infrastructure. The asset development manager needs to provide high-quality assets on a timely basis to the product development projects.

A *product development manager* is responsible for a single product. A *product* is the outcome of a project intended to address a specific customer or market need. The product is also a member of the product line and reflects the product line's scope and architecture. The product development team is constrained by the development approach prescribed for the product line as documented in the production plan. By doing so, it is expected that the overall goals of the product line will be realized. These could include realizing economies of scope, achieving cycle time targets, and improving product quality. The product development manager is

focused on providing high-quality products, on time and within budget, while adhering to the product line processes and contributing to the achievement of the product line goals.

Measurement-related concerns or responsibilities for the managerial roles can be grouped into three categories: performance, compliance, and effectiveness. *Performance* characterizes the extent to which development projects actually meet cost, schedule, and quality targets as compared to an organization's traditional means of product development. A product line should set relatively aggressive goals. *Compliance* characterizes the extent to which development projects utilize the processes, practices, and standards designed for leveraging and reusing the product line's core assets. Compliance with the product line processes and standards is crucial to realizing the benefits of a product line approach. *Effectiveness* characterizes the extent to which the product line meets its overall goals. To effectively exploit commonality, the product line must analyze the potential market for its products in terms of desirable features and develop products utilizing its product line approach.

These categories of managerial concerns reflect information needs for those managing the software product line. The information needs can be used to identify a set of useful measures for each of the three described managerial roles. These categories of concern, in essence, define the goals or issues to be addressed by software measures. Some areas of concern are familiar, such as those associated with project management. These would include meeting cost, schedule, and functionality requirements. The existing literature on software project management contains measures, such as cost and schedule variance and customer reported defects, which would apply to both core assets and product development alike. Also, literature on enterprise management is tapped to address measures for the product line manager. Finally, literature on software reuse served as a source of several measures associated with the product line approach.

The measures in the following table[3] are not to be construed as an exhaustive set; rather, they represent an initial set of suggested measures. Over time, an organization should evolve the set of measures to more closely meet its needs and its development process.

[3] This table is taken from the unpublished paper, "Measures for Software Product Lines," authored by Dave Zubrow, Grady Campbell, and Wolf Goethert in February 2000.

Table 1: Product Line Indicators and Measures

Objective	Product Line Manager	Asset Development Manager	Product Development Manager
Performance	Total product development cost Productivity Schedule deviation Time to market Effort distribution across life-cycle activities Number of products (past, present, and future) Trends in defect density	Cost to produce core assets Cost to produce infrastructure Schedule deviation Defect density in core assets Number and type of artifacts in asset library Core asset quality	Direct product cost Defect density in application artifacts Percent reuse*
Compliance	Mission focus* Architectural conformance Process compliance*	Mission focus Process compliance	Process compliance
Effectiveness	Return on Investment Market satisfaction Market feature coverage	Core assets utility Core assets cost of use Percent reuse	Customer satisfaction

In the following sections we discuss the proposed measures of performance, compliance, and effectiveness for each of the management roles.

* Indicates that the measure is only described once in the section.

2 Measures for Software Product Line Management

The measures in this section are intended to help a product line manager execute his/her responsibility for the overall performance and viability of the software product line. Many are derived from existing literature on enterprise and project management.

2.1 Total Product Development Cost

Total product development cost measures the engineering costs incurred by the product line organization to create a new software product. It consists of two components

- the direct product development costs incurred by the product development group
- a prorated share of the asset development costs

The product development group's costs should be actual expenditures captured by the effort tracking system. If significant costs are incurred to purchase commercial-off-the-shelf software (COTS) components, these too should be included with the product development expenditures. Various schemes [DISA 95, Gaffney & Durek 89, Poulin 97] have been proposed for prorating the asset development costs to a specific product development project/product. For our purposes here, we propose a simple formulation of

$$AllocatedCost = \frac{Annual\ Asset\ Development\ Costs}{Number\ of\ New\ Product\ Development\ Projects}$$

for determining the amount of core asset cost to be allocated to product development projects. Annual Asset Development Costs could initially be based on the budget for an asset development group if the actual expenditures are not available.

This metric could be tracked on a quarterly basis depending upon the rate of new product releases. In this case, costs associated with actual core asset and product development projects completed during the quarter would be tracked. Note that there will necessarily be a gap between expenditures on core asset development and the availability of the assets produced for inclusion in product development projects. The significance of this gap should diminish as the product line matures.

Total product development cost should also decrease as the product line matures. Initially, it may be higher than traditional product development costs since the costs to establish the core asset infrastructure will be spread over a relatively small number of projects. Over time, however, it is expected that relatively less effort would go into the infrastructure and that the

assets produced would be highly leveraged by product development projects. As a result, the total cost to an organization for new products should be reduced.

As an alternative to the preceding formulation, some percentage of core asset development could be allocated to future projects. For example, the overall product line business plan may call for 25 new products to be released during its first 2 years of operation and another 75 during the next 3 years. In addition, the plan may call for $1M to be invested in developing core assets during that 5-year period. Ignoring for now the time value of money, $10,000 would be allocated to the cost of each product development project to cover the investment in core asset development.

Whatever method is selected, it should be based on the product line business plan, the product line strategy (proactive, reactive, incremental) [Clements 01], and follow accepted accounting procedures for cost recovery on investments with no depreciation in asset values. Also, costs allocated to future projects should be increased in comparison to costs allocated to current projects due to delay of use and, hence, the likely increased value of the earlier investment to develop the core assets.

Note that as formulated above, this metric does not seek to measure how the core assets are used. It assumes that a trend toward reduced development costs is due to reuse of core assets. If such a trend is not observed, this metric provides little insight as to why. This metric could be evolved over time to become more precise and insightful. For instance, the prorating of asset development costs might use a formula that evenly apportions non-code costs across all projects and code products based on historical and projected use.

2.2 Productivity

Measuring and tracking productivity provides insight as to whether the product line approach enables an organization to develop more products with the same or fewer resources. *Productivity* is defined as the ratio of the amount of product or output relative to the resources consumed to produce it. Output can be measured in terms of the number of products fielded, number of features delivered in the products, or some measure of the size of the product(s) such as lines of code (LoC) or function points (FPs). Resources consumed will most likely be represented by the effort expended. Effort may be measured in terms of hours and should include not only the effort expended by product development but also the effort expended on asset development activities. It is suggested that an organization initially compute productivity based on a cumulative accounting of products delivered and effort expended. This simplifies the accounting with respect to aligning asset development costs with the delivered products.

To compute a productivity figure, an organization would measure the amount of product delivered since the inception of the product line effort. This would then be divided by the cumulative effort of the corresponding application development projects and the total

investment in the development of core assets during the same time period. Tracking productivity for the product line can be done on a quarterly basis. More frequent reporting may not be worthwhile unless the product line is releasing products on a monthly basis. That is, the update of the productivity measure should match the flow of products released by the product line.

This measure of productivity reflects on the entire product line enterprise. Over time, it should lean toward higher productivity as the investment in core assets and infrastructure are spread across an increasing number of products, and as the cost of creating a product using the assets and infrastructure decreases. To determine whether the rate of improvement in productivity is acceptable, the productivity needs to be compared against some expectation. Therefore, it is useful to estimate productivity as part of the overall product line plan, if only at a high level. A high-level estimate could be made using the budgets for core asset and infrastructure development, the budget for product development, and the estimated size of the products to be released during the year.

This measure indicates one dimension of overall performance, but does not reveal what is driving change in performance. Additional detailed measures of the performance for developing core assets, their use in product development, and the performance of product development teams is needed to gain insight into the underlying factors driving productivity trends.

2.3 Schedule Deviation

The product line manager is responsible for the delivery of all product-line-related products, those produced by asset development and product development. Since less new development is required as the product line matures, the product line should meet schedules more reliably as it matures. *Schedule deviation* for the product line manager is the sum of the variance of all product schedules. This measure should be calculated separately for completed projects and those underway. Schedule deviation reflects both on the accuracy of the planning and the ability of the project to execute the plan.

Variants on this measure include weighting the variances by the planned amount of effort to be expended by the project. Additionally, the absolute value of each schedule variance can be summed to prevent projects that are ahead of schedule from offsetting those that are behind schedule. If the organization uses an earned value management system, schedule measures such as the schedule performance index and schedule slip could also be calculated.

2.4 Time to Market

Time to market captures an organization's capability to deliver products and features faster.[4] This measure is based on the duration or calendar time of completed projects and the functionality they deliver. To measure the duration of the project, an organization must establish operational definitions of when projects begin and end. As an example, the start of a project is marked when approval to undertake the project is received and work to build the product has commenced. The end of the project could be marked by the completion of acceptance testing or by product release. If the product line organization delivers products containing largely varying amounts of functionality, it should either normalize the measures or bin similar projects into relatively homogenous groups. To normalize the measure, we suggest using the organization's standard size measure (e.g., LoC, FP, or number of requirements). The measure should be reported as the duration to deliver a given amount of software (e.g., months/KLoC). Grouping may be accomplished by creating categories of small, medium, and large.

Time to market can be tracked and reported on a periodic basis (e.g., quarterly, annually) determined by the number of projects in the product line and their spacing in terms of completion. Product development project duration should be analyzed in terms of both its central tendency (i.e., average, median) and dispersion (i.e., variance, range). This measure should be also used for product planning, especially estimating overall project schedules.

2.5 Number of Products

Number of products characterizes the scope of the product line as well as its accomplishments to date. Each product produced by the product line represents a contribution to the return on the investment in establishing the product line. At a gross level, the number of products produced gives an indication of the return on investment. Number of products produced also is an indicator of the utility of the core asset base and compliance with the product line processes. The biggest challenge with such a measure is being precise about what it means to have a product produced from the product line. That is, how much of its content must come from or contribute to core assets? Or, is it sufficient that the product be within the scope of the product line (i.e., adhere to the product line architecture and its points of commonality and variation)? In general, a product should adhere to the product line architecture and be composed of or contribute to core assets. The number of products produced is also used to normalize other measures for purposes of comparison.

[4] We recognize that for various business reasons, the pace of new product introduction may be managed to be slower than the capability of the product line to deliver new products.

2.6 Trends in Defect Density

Defects in delivered products reflect their quality. As defects are removed from core assets, the quality of products developed using the product line approach should increase. All defects in delivered products should be tracked to determine if product quality is improving. To compute this measure, defects reported by customers are divided by the size of the delivered product. Only unique defects reported against a product are tallied. A time period must be set for accumulating and reporting the defect count. The time period depends upon the use and complexity of the products, but it should be long enough to allow users to thoroughly test the product's features and capabilities. It may be as short as three months or as long as one year. Trends in the average defect density per reporting period for all products should be plotted. In addition, the variation in defect density as represented by the individual values for each product should also be plotted.

2.7 Mission Focus

Mission focus measures evaluate the degree to which products produced by the product line organization fit within its defined scope. These measures attempt to ensure that the product line maintains a coherent focus consistent with the strategic objectives delegated to the product line organization by higher management. Initially, this is a requirement for the product line for which an objective indicator may not exist. Therefore, as the product line is established, the data for this measure could come from manager surveys that ask for their judgments regarding whether the products produced by the product line are, in fact, consistent with its mission and scope. Once the product line and its corresponding architecture are more established, another indicator would be to track reuse of early life-cycle artifacts such as requirements and designs. It is assumed that products using a significant proportion of these types of core assets in their development are more aligned with the product line than those using relatively fewer of these types of core assets. Mission focus should be reported on a periodic basis (e.g., quarterly). As mission focus improves, other measures such as productivity and quality should improve due to the increasing use of core assets.

2.8 Architectural Conformance

Measures of software architecture could be used to compare product designs with the product line architecture to assess conformance [Avritzer & Weyuker 98, Bass 03]. They may even be used to determine whether a particular product should be developed as part of the product line or as a custom-developed project. Such an analysis would help to preserve the integrity of the product line and facilitate efficient use of core assets for product development. Not every product produced will be within the scope of the product line. Architectural assessment

can help inform this decision. Measures in this area are just emerging and are based on subjective ratings.[5]

2.9 Process Compliance

Establishing a product line approach to software development will yield few results if it is not followed and implemented. *Process compliance* captures the degree to which products are produced using the product line process. The data for this measure could come from process audits conducted by an organization's software quality assurance function. For the product line manager's purposes, a simple count of the number of deviations discovered, resolved, and open across all projects is an adequate starting point. Both core asset development and product development projects should be included. These data should be reported in concert with the periodic management reviews of the product line operation. Over time, additional analyses of the data can be used to improve the product line development process. These analyses could look at deviations between asset development projects and product development projects, variation in deviations by process or life-cycle activity, or type of project (e.g., maintenance, enhancement, or new products).

2.10 Return on Investment

Estimates of the return on investment (ROI) are often made to support a change in business practices. *ROI* is computed as the ratio of the estimated savings (or returns) for each dollar invested. By estimating the ROI for various alternatives for improvement, the alternatives can be compared and ranked in terms of their likely economic benefit to an organization. This applies to the adoption of a product line approach to software development. The challenge for this measure is to clearly and objectively estimate the savings on the investment, the costs to transition to a product line approach, and the costs to develop products using the product line approach. Savings need to be estimated with respect to the current way of producing software. This implies that an organization should track its project costs. To estimate savings, an organization needs a model of how the product line will operate within an organization. It needs to set a time period during which a given volume of products will be produced.

The ROI formula for a given volume of product produced is

$$ROI = \sum \frac{Costt - Costp}{Costi}$$

Where Costt is the cost of traditional software development, Costp is the cost associated with using the product line approach, and Costi is the cost of establishing the product line. The differences in development costs are summed across the expected number of projects.

[5] For instance, see "Predicting Project Risk from Architecture Reviews" by Elaine Weyuker, 1998.

This comparison is simplest if the comparison is done for the product line operation as a whole (asset development and product development) with a comparable amount and mix of product produced using the organization's current approach. Costs to establish the product line include non-recurring start-up costs, such as developing a concept of operations, product line architecture, and the acquisition of tools unique to the operation of the product line.

Furthermore, these costs can be amortized across the expected life of the product line. This will avoid charging the total cost of these assets and the infrastructure solely to the early projects. The time horizon or usable life of the core assets selected and the method for allocating costs across this time period should be consistently followed for all analyses utilizing these data. This includes analysis of total development costs as described above.

While ROI is often computed to compare alternative courses of action prior to implementation, it is important to continue to monitor it to measure whether the anticipated return is actually being achieved. Therefore, the measurement system employed must capture the needed data from both core asset and product development projects. Also, the costs to create infrastructure components to support the overall product line must be captured.

2.11 Market Satisfaction

The product development manager should measure customer satisfaction when purchasing products. This might easily be accomplished through a survey. In addition to general satisfaction, the survey should address reliability, functionality, usability, and *value* (the relation of these attributes to price). Trends in satisfaction value for the general rating and its constituent dimensions (e.g., reliability, usability) should be analyzed. Similarly, variation in the responses from customers should be analyzed to help identify potential improvement opportunities. For instance, certain products or customer segments may reveal trends that the product line can leverage to its benefit. A supporting measure could include the number of complaints received related to the product. (See the discussion on defect density above.) This number should be normalized by volume of product delivered if the volume has been changing over time.

2.12 Market Feature Coverage

Market feature coverage captures the extent to which the features currently available in the product line cover those related to the target market. This could be expressed as a percentage. The key toward assessing the product line's progress, and hence the utility of the measure, is the identification of features that are relevant to the market. This set should be used to establish the denominator of this measure. Additionally, it should be reassessed periodically.

3 Measures for Asset Development Management

These measures are used to characterize the performance of projects to produce core assets, the value of these assets to product development projects, and the management of the assets. They do not cover all of the measures that an asset development manager might use to actually manage a project.

3.1 Cost (Effort) to Produce Core Assets

An organization should track the cost to produce its various core assets. Not only are these costs important from the typical project control or management perspective, they are a major input for tracking the success of the product line approach. Additionally, understanding how the cost of designing for reuse compares to traditional product development costs is a key component for deciding what components are worthwhile to construct for reuse. It is expected that developing assets for reuse will be more expensive than developing them for a single specific use. The degree of additional expense needs to be monitored and controlled. Note that Poulin and Boehm [Poulin 97, Boehm 00] assume that the cost to produce these assets may run 1.5 to 2 times higher than producing the same functionality for a one-time use. On the other hand, the use of such assets runs about 1/5 the cost of developing the functionality.

These costs resemble typical software project costs. Corresponding measures should track the development of both software components as well as the development of non-code assets such as domain requirements, architectures, and user documentation. These costs should include all direct labor costs and be associated with the specific asset that is made available to application developers.

3.2 Asset Development Schedule Deviation

Because core assets are planned for use in subsequent product development projects, their availability as planned is crucial to the operation of the entire product line. The performance of the asset development project against its planned schedule should be tracked. Furthermore, this schedule needs to be visible to product development projects that are dependent upon the core assets under development. As revisions to the schedule occur, this history should be retained to serve as an input for future improvement in estimation and scheduling. Typically, *schedule deviation* is computed as the duration between the planned and delivered dates, where planned dates are based on projected needs of product development projects. As with cost, this measure should be tracked against the development of both code and non-code

assets. Critical to the functioning of the product line is development of its architecture. Slipping the schedule in this non-code asset would affect multiple product development projects.

3.3 Defect Density of Core Assets

It is extremely important to manage the development/acquisition of the product line's core assets, as core assets will influence the quality, reliability, and cost and schedule of every derivative product development. The success of the product line relies on the quality of the core assets and the capabilities that they offer. This measure indicates the quality of the core assets used to produce applications. To compute this measure, defects reported by product development projects and customers are divided by the size of the core asset. Only unique defects associated with core assets are counted. The defects need to be attributed to the core asset in which they originated. A time period must be set for accumulating the defects from customers, as discussed above. Trends in the average defect density for core assets and its variation should be plotted.

3.4 Core Asset Quality

Software product lines have additional quality requirements beyond those of a single product. Core assets in particular need to meet the standards of the product line in addition to those of the product's customer(s).[6] These could include conformance to various standards, specifications, and the architecture. In addition, there may be non-functional attributes that are important from the product line's perspective. The status of these measures with respect to practical application and validation is unknown.

3.5 Number and Type of Assets Available

Knowing the number and types of assets available is useful in several ways. As noted, a software product line is composed of core assets that span all the types of artifacts used to produce software products. This is in contrast to reuse libraries that provide software code to implement frequently used functions. Asset development management needs to monitor and guide the development of core assets in addition to code. Counts of requirements, their corresponding designs, test plans, and procedures can inform asset development management about the growth in reusable development packages (e.g., assets that permit the complete implementation of a requirement).

[6] For a list of several potential measures, see the National Institute for Standards and Technology's paper on *Quality Characteristics of Reusable Software* [Salamon & Wallace 94].

3.6 Process Compliance

Process compliance captures the degree to which core assets and tools are being produced in accordance with the organizational product line standards. These measures provide insight into the extent to which product development projects comply with an organization's product line processes and practices. Such data are useful for establishing a context for interpreting other performance data associated with the product line. They also provide insight into the adoption of the product line practices by product development projects during product line start-up. The data for this measure could come from process audits conducted by an organization's or product line's software quality assurance function. For the purpose of asset development management, a simple count of the number of deviations discovered, resolved, and open across the asset development effort is an adequate starting point. These data should be reported in conjunction with the periodic management reviews of the product line operation.

3.7 Core Assets Utility

Asset development management should track the extent to which core assets provide value to product development projects. This accrues cumulatively across product development projects and individual projects. Core asset utility provides insight into the extent to which core assets are able to satisfy the goals of the product line as well as helps determine whether the assets in the asset library are the assets actually needed by product development projects. Knowledge as to which core assets are used to create products through product development projects should be made available through a configuration management system.

One measure of the utility of core assets is the effect their use has on product development project performance. This can be measured in terms of cycle time, cost, and quality. This measure requires that an estimate be made as to what the performance of the product development project would have been had the core assets not been available. These estimates also support computation of the overall return on investment of the product line, as discussed earlier. This measure would then be represented in a cumulative manner for all products developed as well as for individual projects (e.g., median and range or mean and standard deviation).

If the data required for this measure are not available, an initial metric would be *percent reuse* [DISA 96]:

$$\frac{SizeofCoreAssetsUsed}{(SizeofCoreAssetsUsed + SizeofNewandModifiedApplicationArtifacts)}$$

In addition to percent reuse, a count of the number of uses of each core asset divided by the number of products in the product line can provide insight into those assets that are commonly used and those that are used less frequently. This will help determine whether the assets in the asset library are the assets actually needed by product development projects. Ferri & Pratiwadi et al. discuss several metrics (e.g., code profiles and reuse growth factor) that provide further insight into the use of core assets [Ferri & Pratiwadi et al. 97].

Another option is to count the number of product requirements met by core assets.

3.8 Core Assets Cost-of-Use

Core asset management needs to be cognizant of the costs incurred by product development projects in order to use core assets effectively. If the costs become too large, product development projects may seek other ways of satisfying product requirements. From the perspective of the product development project, the costs they incur are those associated with

- identifying appropriate core assets
- understanding how to apply assets to the project, and if need be, adapting them for use
- integrating and testing

Costs in this category would primarily be driven by the effort spent by engineers to find, understand, and tailor assets for use on their projects. If the effort tracking system does not support this level of effort tracking by activity, a study could be done as the basis for estimating the cost. Core assets cost-of-use should be tracked across all projects and for each asset type. The finer-grained the data, the greater insight core asset management will have into how this cost can be reduced.

In addition to the costs incurred by product development projects, costs may be incurred, for example, to create and implement a repository that facilitates the management and use of core assets.

3.9 Measures for Product Development Management

These measures are to be used to characterize the performance of product development projects to produce products. They do not cover all of the measures that a product development manager might use to actually manage a project; additional measures or variants on the measures might be used. For the most part, this role requires few metrics beyond those typically used for software project management. On the other hand, the application project setting is the source of much data on the performance of the product line. Therefore, there must be a measurement system that provides for the collection of the needed data.

3.9.1 Direct Product Cost

These costs resemble typical software project costs, and should include all direct labor costs incurred while producing (i.e., requirements analysis, design, construction, and testing) the product. Application development management should monitor the relative proportion of project costs going to produce application-specific code. The remaining costs can be interpreted as the cost to integrate core assets into the product. Over time, it would be expected that the former would decline as a proportion of the latter and that the latter would decrease in absolute terms as an organization gains experience and the quality of the components improves.

3.9.2 Defect Density of Application-Specific Code

The product development management role needs to understand the quality of the code it is producing as well as the quality of the assets it is utilizing. To compute this measure, defects reported by customers are divided by the size of the application-specific code in the product. During repair, each defect must to be attributed to either application-specific code or a core asset. Only unique defects are tallied. A time period is usually set for accumulating the defects from customers, as discussed above. Trends in the average defect density for application-specific code and its variation should be plotted. This information can be used to gauge the performance of the application development software processes.

A related use of the data is to track trends in the types of defects discovered by users. Such information is useful for identifying areas for process improvement.

3.9.3 Process Compliance

The product development manager needs to monitor compliance with processes associated with the operation of the product line. Since it is likely that the processes will be incorporated into the estimating and planning of product development projects, failure to comply with these processes may put the project at risk. As described above, reports from audits and reviews should be provided to the product development manager. Furthermore, these reports should make it easy for the manager to assess the degree to which the project is adhering to the product line concept of operations.

3.9.4 Percent Reuse

Planned reuse plays an important role in estimating the cost and schedule for product development projects. Product development management should track which core assets are used in the product and the extent to which each is used. Knowledge as to which core assets are used should be available from the configuration management system. The use of core assets in all work products, not just code, should be tracked. Failure to achieve planned reuse levels may signal performance trouble for the product development project and may also trigger a causal analysis by core asset management. For instance, the product development

project may find itself creating functionality from scratch when it planned to only integrate a core asset with the desired functionality. Hence, there would be a likely impact on cost and schedule.

Percent reuse can be computed as follows:

$$\frac{SizeofCoreAssetsUsed}{(SizeofCoreAssetsUsed + SizeofNewandModifiedApplicationArtifacts)}$$

As the above example illustrates, a product line reuse of assets goes beyond code. A size measure such as pages or requirements is necessary for computing this measure. Additionally, a summary measure can be created by converting assets and artifacts to a common measurement unit such as lines of code.

Lim cautions that such a measure can overstate the percent reuse, and hence, any measures derived from such a measure (such as costs avoided) [Lim 98]. The reason for this is that the reused asset may include functionality not required for the product being developed.

3.10 Customer Satisfaction

Product development managers need insight into how well the products produced by their projects satisfy their intended customers. As products are delivered to customers, the product development manager can ask for feedback via a customer satisfaction survey. This survey can feature questions regarding whether the product satisfies requirements and can also address satisfaction with respect to how well the project met its cost and schedule commitments. These data should be retained and analyzed cumulatively as more products are produced.

4 Summary and Conclusions

The management of software product lines poses additional challenges to those currently managing software-intensive projects. As has been presented, there are several proven measures that are applicable and documented for managing software product lines. However, a few measures have been developed to meet the special needs of software product lines. Work remains to be done to validate the utility of these measures. Software product lines also demand ways to assess the scope, features, variants, and underlying architecture from a business perspective. This will require additional data and measures that are not usually gathered by software organizations. While the measures discussed in this technical note provide ideas about how to meet the information needs of the various management roles, there is clearly room for improvement in the measures and further research.

References

[Avritzer & Weyuker 98] Avritzer, A. & Weyuker, E. J. "Investigating Metrics for Architectural Assessment," 4-10. *Proceedings of IEEE Fifth International Symposium of Software Metrics (Metrics 98)*, Bethesda, Maryland, November 20-21, 1998. Los Alamitos, CA: IEEE Computer Society Press, 1998.

[Bass 03] Bass, L.; Clements, P.; & Kazman, R. *Software Architecture in Practice (second edition).* Reading, MA: Addison Wesley, 2003.

[Bergey & Goethert 01] Bergey, J. & Goethert, W. *Developing a Product Line Acquisition Strategy for a DoD Organization: A Case Study*. (CMU/SEI-2001-TN-021, ADA395202). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2001. <http://www.sei.cmu.edu/publications/documents/01.reports/01tn021.html>

[Boehm 00] Boehm, B. et al. *Software Cost Estimation with COCOMO II*. Upper Saddle River, NJ: Prentice-Hall, 2000.

[Clements 01] Clements, P. & Northrop, L. *Software Product Lines: Patterns and Practice.* Reading, MA: Addison Wesley, 2001.

[DISA 95] Defense Information Systems Agency (DISA). *Software Reuse Business Model (SRBM)*, Department of Defense Software Reuse Initiative, January 1995.

[DISA 96] Defense Information Systems Agency (DISA). *Reporting Metrics and Measures*, Department of Defense Software Reuse Initiative, April 1996.

[Ferri & Pratiwadi et al. 97] Ferri, R. N. & Pratiwadi, R. N. et al. "Software Reuse Metrics for an Industrial Project," 165-173. *Proceedings of the Fourth International Symposium on Software Metrics*, Albuquerque, New Mexico, November 5-7, 1997. Los Alamitos, CA: IEEE Computer Society Press, 1997.

[Gaffney & Durek 89] Gaffney, J. E. & Durek, T. A. “Software Reuse—Key to Enhanced Productivity: Some Quantitative Models.” *Information and Science Technology 31*, 5 (June 1989): 258-267.

[Lim 98] Lim, W. C. *Managing Software Reuse: A Comprehensive Guide To Strategically Reengineering The Organization For Reusable Components*. Upper Saddle River, NJ: Prentice-Hall, 1998.

[Poulin 97] Poulin, J. S. *Measuring Software Reuse: Principles, Practices, and Economic Models.* Reading, MA: Addison-Wesley, 1997.

[Salamon & Wallace 94] Salamon, W. J. & Wallace, D. R. “Quality Characteristics and Metrics for Reusable Software (Preliminary Report).” National Institute of Standards and Technology, NISTIR 5459, May 1994.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	October 2003	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Measures for Software Product Lines	F19628-00-C-0003

6. AUTHOR(S)

Dave Zubrow, Gary Chastek

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2003-TN-031

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

This technical note characterizes the status of measurement associated with the operation of a software product line, suggests a small set of measures to support its management, and provides guidance for those establishing measurement activities within a software product line. It is intended to help managers of software product lines develop a set of base measures for tracking those categories of needs most relevant to their organization's products, projects, and processes. The measures suggested here range from relatively mature to those whose general utility have yet to be validated. Therefore, an organization using this paper needs to assess its ability to generate the measures and the value they are likely to return to the organization. In most cases, an organization may wish to start with a subset of the measures described.

14. SUBJECT TERMS	15. NUMBER OF PAGES
measurement, measures, software product lines	35

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL