December 2001

A Metamodel for the Capability Maturity Model for Software

Jeffrey Ingalsbe
Ford Motor Company

Dan Shoemaker
University of Detroit Mercy

Vladan Jovanovic
University of Detroit Mercy

A METAMODEL FOR THE CAPABILITY MATURITY MODEL FOR SOFTWARE

Jeffrey Ingalsbe
Ford Motor Company
jingalsb@ford.com

Dan Shoemaker
University of Detroit Mercy
shoemadp@udmercy.edu

Vladan Jovanovic
University of Detroit Mercy

Abstract

This document presents a metamodel for the "Capability Maturity Model for Software". This Metamodel embodies seven fundamental concepts (common types), which are neither new nor unique to the SW-CMM and which occur across all key process areas in the same relationship to each other. It can be (and has been) used to build a model for each of the 18 key process areas in the SW-CMM and an idealized model for the entire SW-CMM. It represents an orthogonal view of the SW-CMM and can be used to define the fundamental skills an organization must instantiate across all 18 key process areas.

The Software Engineering Institute's (SEI) Capability Maturity Model (SW-CMM)

The "Capability Maturity Model for Software" (SW-CMM) describes five levels of maturity through which an organization can progress to define, assess, and improve their software processes. The five levels are Initial, Repeatable, Defined, Managed, and Optimized. Each level of maturity (except level one) comprises several key process areas (eighteen across all maturity levels) that must be addressed to establish and subsequently demonstrate capability at that maturity level. Organizations are certified at a maturity level by participating in a CBA-IPI assessment (CMM[SM]- Based Appraisal for Internal Process Improvement) conducted by SEI authorized Lead Assessors. The SW-CMM is documented in two SEI technical reports ("Capability Maturity Model for Software, Version 1.1" and "Key Practices of the Capability Maturity Model, Version 1.1"). Combined, the documents exceed 500 pages and contain only a handful of graphics. Their actual and perceived complexity is high. This makes it difficult to both understand and apply the model in practical business settings.

The Problem(s) With Implementing CMM

Organizations embrace the SW-CMM for two primary reasons. First, they want to **establish and improve** software processes in order to improve software quality, increase productivity, and decrease product development cycle time. Various organizations have reported these benefits after embracing the SW-CMM (Brodman and Johnson, 1996). Doing this requires that they understand the principles set forth in the SW-CMM and implement them according to their culture and the experience of their personnel. Second, they want to formally **demonstrate** their maturity in order to differentiate themselves from their competitors or obtain contracts restricted to organizations with a given SW-CMM rating (Saiedian, and Kuzara, 1995, maintain that it should be a potent ingredient for winning and maintaining contracts).

Although the previous paragraph would seem to indicate that the SW-CMM is the silver bullet everybody has been looking for, there are four serious roadblocks to implementing it: **Communication**: The SW-CMM is large and text based. Communicating about the SW-CMM with one person or disseminating information about the SW-CMM to many persons is difficult. **Comprehension**: The SW-CMM is complex. It is difficult to master the entirety of its complexity. Additionally, comparing and contrasting the SW-CMM to other models for software process development and improvement is nearly impossible because there is no common frame of reference. **Time:** An organization typically takes 18 to 30 months to move up one full maturity level (Hayes and Zubrow 1996). Working on improving software processes means not working on projects. Paulish and Carleton (1994) report that some organizations have trouble finding the time to work on software process improvement because of their current commitments to deliver products to their customers. Many companies will be out of business in the three to five years

it would take to move from level 1 to level 3. **Money**: Formal certification at a given maturity level is obtained by participating in a CBA-IPI assessment conducted by SEI authorized Lead Assessors. The assessors' services are typically in the neighborhood of $50,000 per assessment depending on the size of an organization and the travel involved. There are typically two assessments performed at each level of maturity, a baseline and a certification. In order to overcome the communication and comprehension roadblocks, organizations typically employ consulting services between assessments at an average cost of $156,000 per maturity level *(based on an informal survey of 10 of the most active assessors).* Using averages, in order to reap the benefits of a SW-CMM level 3 rating, an organization will spend in excess of $500,000 over 3 to 5 years (Hayes and Zubrow 1996). Saiedian, and Kuzara (1995) report it is nearly a universal complaint that moving from level to level can costs thousand or even millions of dollars.

Who is Affected

Small to medium sized organizations that don't have the time or money (or both) to embrace the SW-CMM are not likely to utilize it even informally because of the communication and comprehension roadblocks. The roadblocks are even worse in the international community where English language might be an issue and SEI authorized Lead Assessors (and their consulting services) are in short supply (especially in the Asia/Pacific region, SEI Website at www.sei.cmu.edu.).

Solution

This paper distills the essence of the SW-CMM into a metamodel of seven fundamental concepts neither new nor unique to the SW-CMM, which occur across **all** key process areas in the same relationship to each other. The metamodel can be (and has been) used to produce a model for each key process area (outside this paper) and an idealized model of the entire SW-CMM (also outside this paper). The metamodel (and the models that can be built using it) are useful in the following ways:

- As a communication tool for organizations of all sizes
- As a method for small to medium sized organizations to understand and implement the intrinsic or indispensable principles of the SW-CMM when time, money, and personnel are impediments to attaining formal SW-CMM certification or acquiring SW-CMM consulting services.
- As a method for comparing and contrasting models for software process development and improvement

The metamodel provides a practical basis for understanding the application of the SW-CMM. They metamodel elements exist in two logical groups. The first is work products, issues, and maintenance. The second is policies, processes, plans, and resources. By understanding that the SW-CMM requires all work products, issues raised against those work products, and maintenance performed on those work products to be handled in the same manner (whether the work product is a Software Requirements Specification for level 2 or a Continuous Process Improvement Plan for level 5) an organization can put the infrastructure in place that will carry them through all 5 levels of maturity. The actual certification to a given level then, is a simple matter of certifying that the goals of the KPAs of that level are directly addressed by the processes already in place (or perhaps the better analogy for how this works is to pour the old wine into a CMM shaped bottle).

Approach

The original intent of the author's work was to manage the complexity of the SW-CMM by critically examining each KPA and producing a (graphical) model (using the UML notation) for that KPA using the principles of decomposition, abstraction, and hierarchy (Berzins and Naumann, 1986). It quickly became clear that the same entity abstractions (Booch 1994) and their associations reoccur in every KPA with minor specializations. Additionally, the abstractions were largely form-based written pieces of work created, routed for approval, distributed for communication, associated with other forms or data and completed based on deadlines making them ideal candidates for business process automation (WFMC, 1998). The analysis proceeded with the modified objective of first modeling the reoccurring abstractions and their associations (the metamodel) and then using that model to produce a model for each key process area. All the while keeping business process automation in mind. Upon completion, the vocabulary (verbs and nouns) of the entire SW-CMM was examined to see if the metamodel abstractions and associations adequately represented the similarities between those verbs and nouns across all KPAs. No effort was made to determine whether the author had discovered the *right* abstractions for this problem domain, as there is no quantitative method for doing so. This paper presents the metamodel elements but not the models for all KPAs or the idealized model of the entire SW-CMM because of length limitations.

Suitability for Business Process Automation

The metamodel comprises recurring concepts that are woven into the fabric of all key process areas. Therefore, through the utility of business process automation, a concept needs to be automated only once for the entire SW-CMM. As an organization evolves to higher levels of maturity it simply re-uses the metamodel implementations each time a new one is required or encountered. All of its constituent elements can be represented electronically as form-based artifacts (with some elements comprising a collection of forms, possibly heterogeneous). The associations between constituent elements can be represented electronically as rules that manage the creation of electronic forms (according to a documented procedure), their routing for approval (by authorizing personnel), their access for communication (of size or status), their association with other electronic forms or data (for the purpose of cascading issues and maintenance), and their completion based on programmable deadlines. Additionally, an electronic implementation of the some of the metamodel elements facilitates the controlled specialization of those elements. What follows is a discussion of each of these elements.

The Work Product Metamodel Element

Every software artifact discussed in the SW-CMM is a work product. This includes, but is not limited to, policies, processes, plans, specifications, code, standards, and software process assets. They are produced at the organization level (e.g. the System Requirements Allocated to Software) or the software project level (e.g. the Software Requirements Specification). The common discussion around all software artifacts is represented visually in Figure 1.

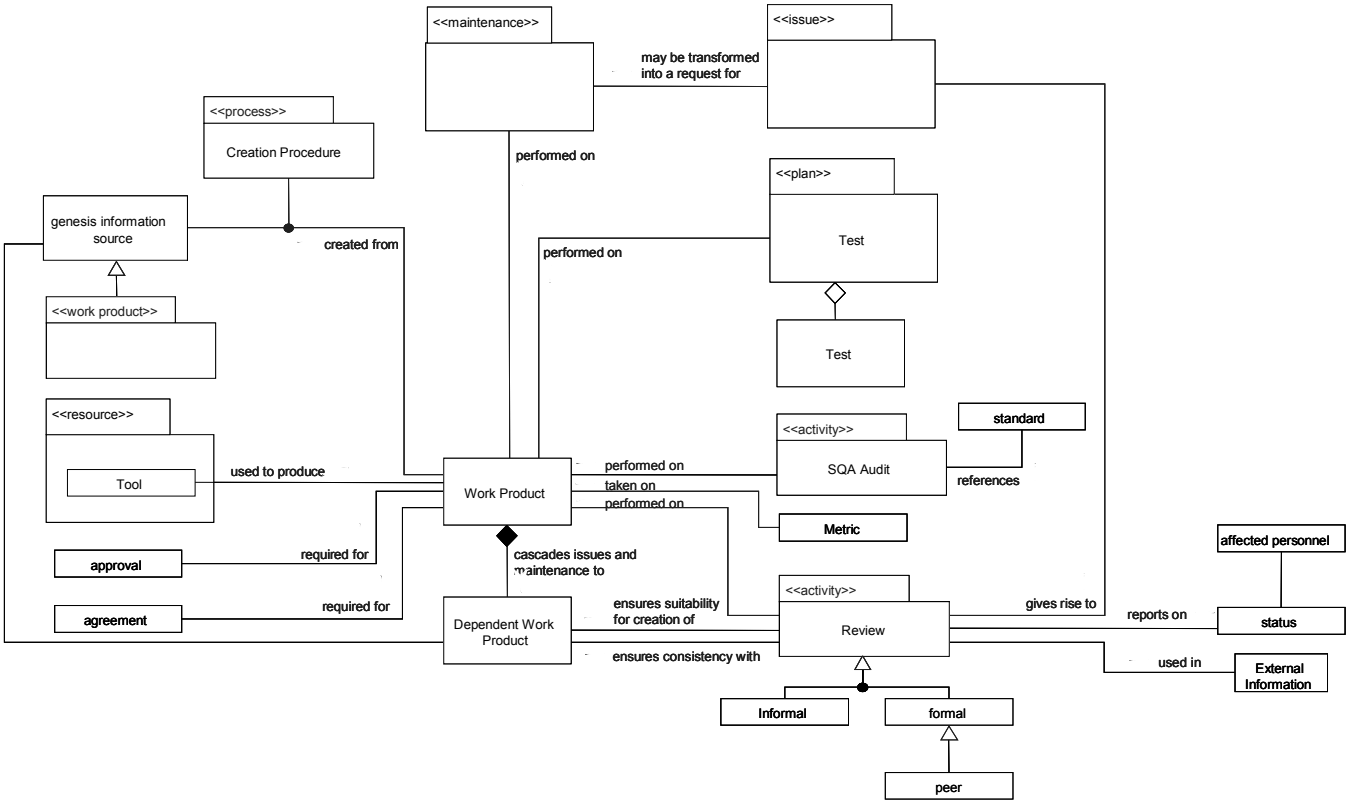

Figure 1. Work Product Metamodel Element

As the diagram shows, a work product is always created according to a documented procedure from some kind of genesis information by trained, available, allocated, responsible personnel using approved, available tools and technology. The genesis information could be another work product, a document or an amalgam of both. Whatever its form, a work product is subordinate in that changes to or issues with the genesis information source are cascaded to the work product. Similarly, a work product may be associated with zero, one, or many dependent work products. Dependent work products are subordinate to their superior work product in that changes to or issues with the superior work product are cascaded to the dependent work product. A work product is always reviewed by affected personnel. A review is an activity in which the work product is looked over, studied, or examined

in a critical fashion according to a documented procedure relative to a standard and objectives. A review could be formal or informal and could be conducted from a technical or business standpoint. As a result of a review, issues (matters that need to be investigated) may be raised. Those issues are analyzed, tracked to closure, and cascaded to dependent work products. Additionally, those issues may be transformed into maintenance requests. Maintenance is performed on work products according to a documented procedure by responsible personnel with the consensus of affected personnel, and the approval of authorizing personnel and is cascaded to dependent work products. A work product is measured. Measurements are used in reviews where they are compared to previous or estimated measurements to determine quality or status. Finally, work products are approved by authorizing personnel, verified, reported on, remembered, and comprised of one to many versioned, constituent elements. The creation, maintenance, approval, and measurement of work products are subject to software quality assurance audits (activities) to ensure compliance with documented processes relative to standards.

The Issue Metamodel Element

Issue management is discussed by implication in all key process areas of the SW-CMM and by direct reference in several. The Intergroup Coordination, Defect Prevention, and Peer Review key process areas are fundamentally concerned with issue management. Issues are points of debate, controversy, or concern that must be documented, assigned to responsible personnel, analyzed, categorized, and reported on (all according to a documented procedure). Sometimes analysis reveals that other personnel should handle the issue or that an escalation is proper. If so, the issue is assigned to appropriate personnel where additional analysis is performed. In addition to traditional analysis, there are three special types of analysis that may be performed on the issue as required in several key process areas: risk analysis, root cause analysis, and impact analysis. Root cause analysis is required in Defect Prevention (Level 5: Optimized). After analysis, issues may be categorized as defects and subsequently transformed into maintenance requests. If the maintenance request is for a work product that has dependent work products, multiple maintenance requests may be generated. An SQA Audit is performed to ensure that issues are handled according to the documented process.

It is important to note that issues are logged against work products only. Every point of debate, controversy, or concern enters the system as an issue. Defects are issues that have been categorized as errors, omissions, or inconsistencies relative to genesis information (remember, genesis information may include standards, guidelines, other work products, etcetera). Maintenance requests are issues, which require changes in work products.

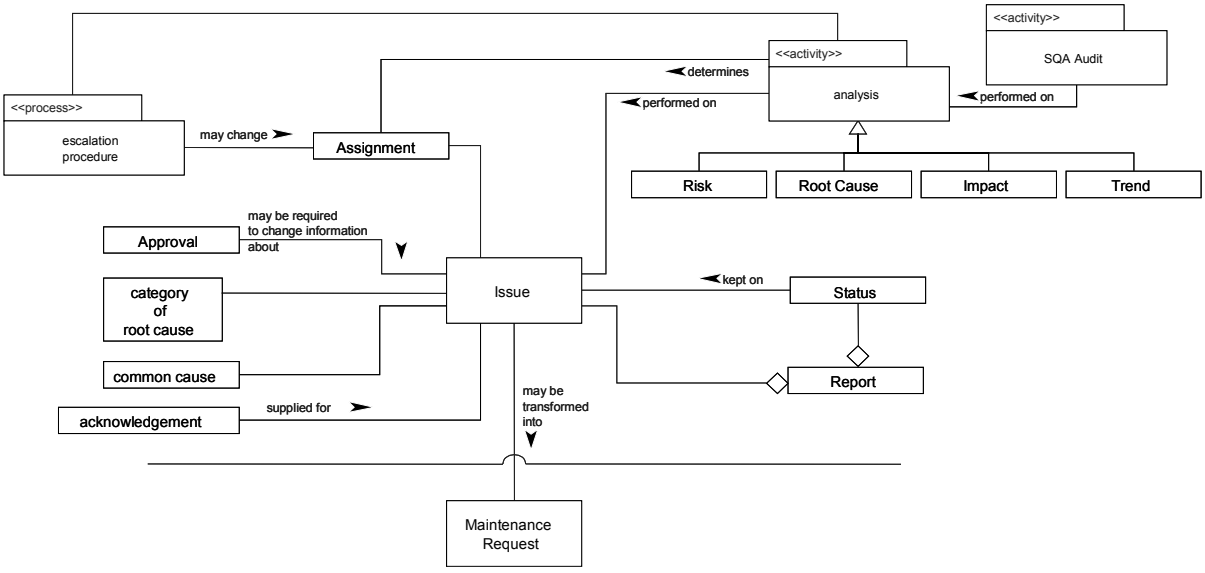

Figure 2. The Issue Metamodel Element

The Maintenance Metamodel Element

Maintenance is changing work products. Maintenance arises from issues (that have been transformed into maintenance requests), cascaded maintenance (resulting from maintenance on genesis information), and the input of temporal metrics. Maintenance on work products is performed according to a documented procedure by responsible personnel negotiating with affected personnel, with the approval of authorizing personnel and is cascaded to dependent work products. Analyzed, negotiated, reviewed, planned, prioritized, approved, executed, tracked, and remembered. Status information is communicated to all affected personnel.

For example, a software development plan is updated with actual schedule, resource, effort, size, and cost metrics. This change to the software development plan is different in nature than the two previously discussed: It doesn't require a review, an approval to incorporate, or a plan to incorporate However, it is a change to the document that should be documented and remembered For example: the effort expended on reviews during the requirements analysis phase of the project was updated on June 10th, June15th, and June 30th. Instead of constructing a new type to handle to include this change, it was deemed simpler construct the maintenance type with the understanding that it includes changes from the input of temporal metrics. To reflect this understanding, some of the associations are optional instead of require for example, reviews, approvals, and plans have cardinality of 0 to many on their associations with the maintenance request.

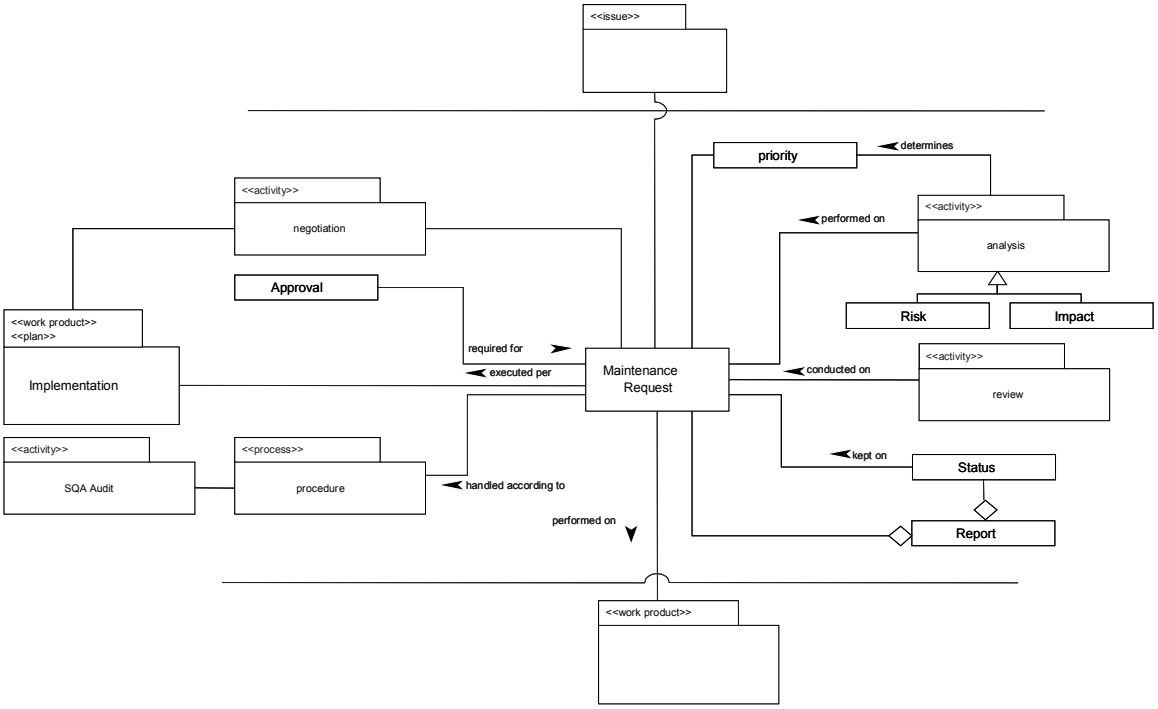

Figure 3. The Maintenance Metamodel Element

Policies, Processes, and Plans Qualifying Statement

The policy type, the process abstraction, and the plan type are all activity aggregates (and work products). They could have all been derived from an element called "Activity Aggregate". However, since these models are really domain models it seemed appropriate to consider a plan, a process, and a policy as atomic elements and not introduce an artificial element that is not referenced in the SW-CMM for any other purpose than to acknowledge their similarity. An implementation of these types would likely involve an activity aggregate element and a generalization specialization hierarchy. The important thing to note here is that a policy is a high level list of activities and (possibly) their resultant work products that provides the framework for a process. A process is an ordered, sequential list of activities, their manner of execution, and their resultant work products that specifies the preset format for a plan. Finally, a plan is an instance of a process. For example, a policy specifies in general what will be done with regard to requirements management across all projects; a process specifies in detail what will be done with regard to requirements management across all projects; a plan specifies in detail things like the person who will be responsible for requirements management on **this** project and how long it should take them. Policies, processes, plans, and projects are all work

products so it is implied that they are created according to a documented procedure from some kind of genesis information by trained, available, allocated, responsible personnel using approved, available tools and technology. Additionally, they are reviewed by affected personnel, measured, approved by authorizing personnel, verified, reported on, remembered, comprised of one to many versioned, constituent elements, linked to any number of dependent work products, and maintained according to a documented procedure.

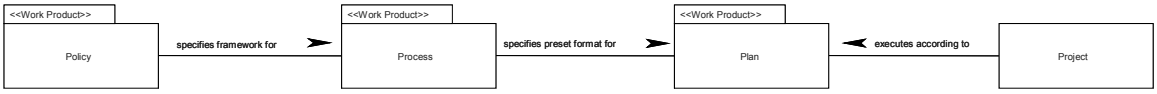

Figure 4. Relationships between Policies, Processes, Plans, and Projects

The policy states in general that an activity must be performed, the process states in detail that an activity must be performed, and the plan details its predicted and actual execution

The Policy Metamodel Element

Policies are activity aggregates that specify activities to be executed and (possibly) associated work products but not necessarily the order or manner of execution of those activities or the characteristics and handling of the work products. They are always accompanied by a discussion of their typical content. Remember that policies are generalizations of processes or, conversely, processes are specializations of policies. With regard to policies, the responsible personnel are senior management and the Software Engineering Process Group (they compose policies and manage them), affected personnel are all engineering groups but primarily the software engineering group(s) (they adhere to policies, require notification of changes, and should be involved in discussions surrounding policies), and authorizing personnel are senior management (they approve policies).

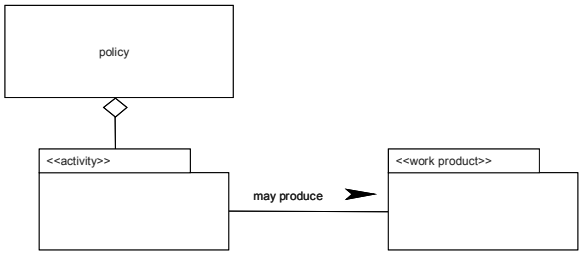

Figure 3.4. The Policy Metamodel Element

The Process Metamodel Element

Every process, procedure, method, and guideline discussed in the SW-CMM is a process. Processes are activity aggregates that specify activities to be executed, their order and manner of execution, their associated work products, and the characteristics of those work products. Processes are templates specifying the preset format for plans. They are specializations of one or more policies. Additionally, processes delineate the interrelationships between activites, dependent activities, work products, and dependent work products. Activities may produce work products. Those work products may be required for subsequent activities in which case the subsequent activities are **dependent** and the work product must be examined to determine whether it is ready and appropriate for use. Work products produced in dependent activities must be compared with the work products produced in superior activities to ensure that they address any requirements set forth by the superior work products. The integrity of the dependencies is maintained by cascading work product changes to all dependent work products. This means, in some manner, executing the activities that produced the dependent work product again. Processes may be constituent parts of other processes. With regard to processes, the responsible personnel are the SEPG (they write processes and manage them), affected personnel are all engineering groups but primarily the software engineering group(s) (they use processes, require notification of changes, and should be involved in discussions surrounding processes), and authorizing personnel are senior management (they approve processes). The two most notable instances of the process type are the Organization's Standard Software Process and the Project's Defined Software Process, which is tailored from the former.

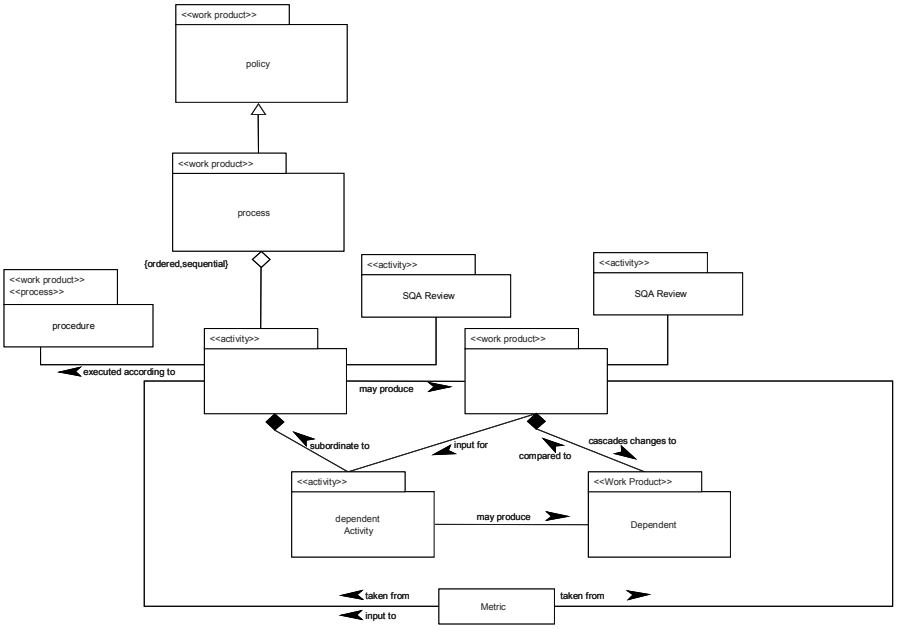

Figure 5. The Process Metamodel Element

The Plan Metamodel Element

Plans are activity aggregates that delineate in detail the execution of a project (or program) beforehand and dynamically change during project (or program) execution. Plans delineate project (or program) execution in detail beforehand and dynamically change during project (or program) execution. They are instantiations of the process type. That is, the process provides a template; the plan provides the specific details to populate that template. During the execution of a project, metrics are taken on activities and work products and recorded in the plan (hence the plan changes dynamically as the project executes). During certain planned activities, the metrics and the estimates are compared. Subsequently, issues may be raised which, after analysis, consensus, and approval could result in maintenance on the plan (i.e. estimates for future activities revised, or new activities added). Plans exist at the organization level (e.g. Process Development and Improvement Plan) or the software project level (Software Development Plan) and can be constituent elements of other plans (e.g. a project plan may comprise a Software Development Plan, a Software Quality Assurance Plan, and a Software Configuration Management Plan).

Plans are produced during activities that are planned themselves. A Software Quality Assurance Plan might be produced in an activity that is specified in a Software Development Plan. A Software Development Plan might be produced in an activity that is specified in an overall Project Plan.

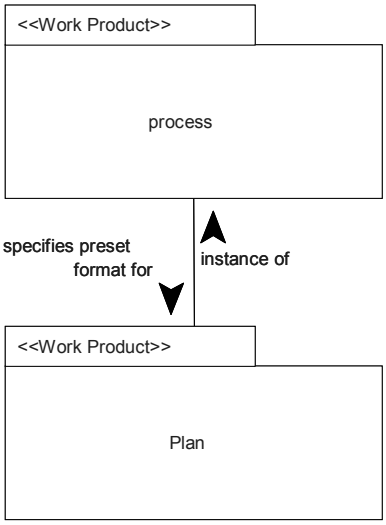

Figure 6. The Plan Metamodel Element

The Resource Metamodel Element

Resources are personnel, tools, and technologies that are drawn upon to perform activities are discussed throughout the SW-CMM and can be referred to as resources. These resources and the common discussion around them are abstracted in the resource metamodel element.

Personnel, tools, and technology must be allocated for consumption by a project as detailed in a plan. Personnel must be trained and available. Personnel create and maintain a personal training plan which is subordinate to an organization training plan (specifying employee development training requirements) and must be reconciled with a project training plan (specifying the project's training requirements) before allocation to a project. Those personnel must have approved tools and technology that have been evaluated relative to documented criteria allocated for their use.

There are only three types of personnel in an organization: those who are responsible for performing an activity (**responsible personnel**), those who are affected by the performance of an activity (**affected personnel**), and those who authorize the performance or outcome of an activity (**authorizing personnel**). The responsible personnel are those that are trained, available and allocated to perform the work. The affected personnel are those that receive or have access to all appropriate activity information (and hence work product information). This would include (but is not limited to) status reports, maintenance requests, and open issues. The authorizing personnel are those that ensure personnel training and allocation and later approve work. Personnel who are responsible on one project may be affected on another.

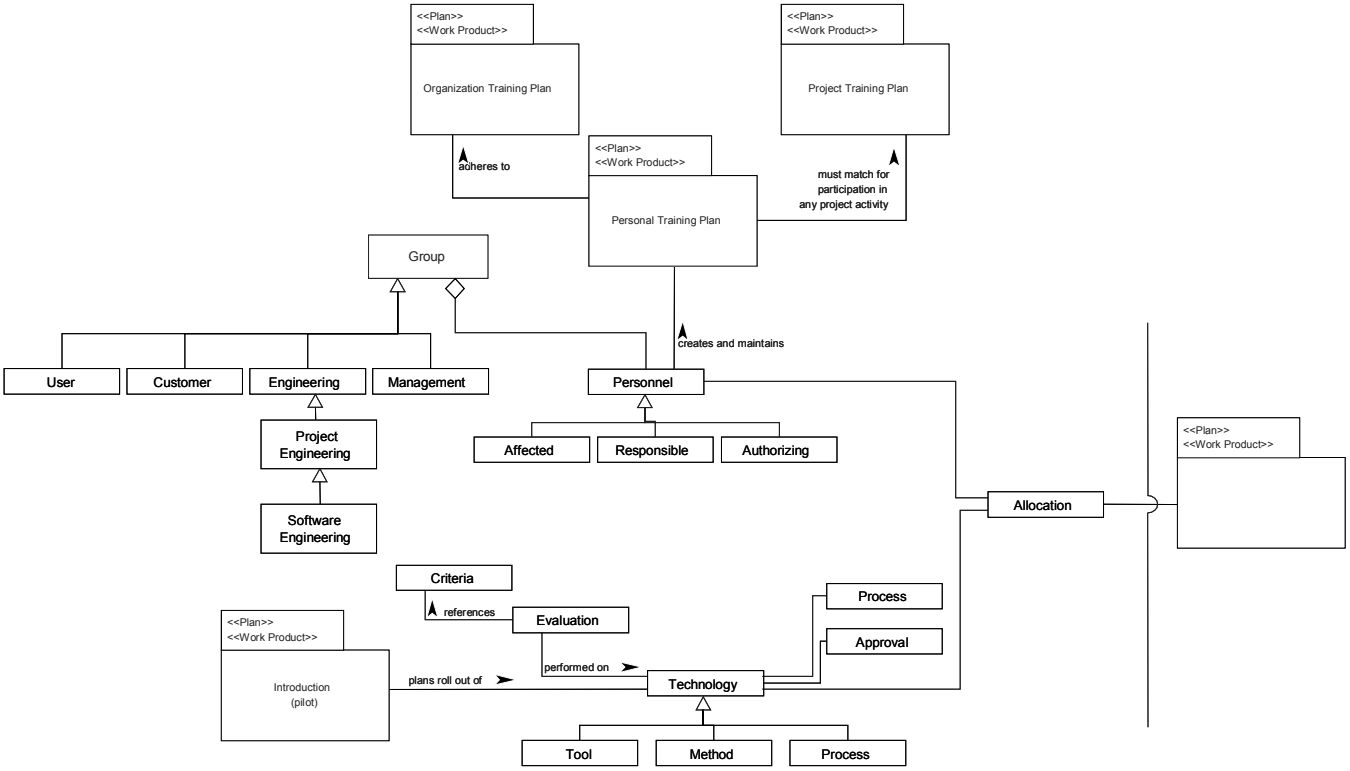

Figure 7. The Resource Metamodel Element

Pulling It All Together

The previous sections presented seven common types that have been identified in and abstracted from the SW-CMM. Their interrelationships form a framework (figure 8) that is the fundamental structure upon which the models for each key process area are built. Work products are produced by activities during the execution of projects. Issues may be logged against a work product and may, subsequently, result in maintenance on that work product. Projects are executed according to a plan. Plans specify the consumption of allocated resources (personnel, technology, and tools) and are instantiated from predefined templates called processes. Processes are subordinate to organizational policies. The utility of the framework is twofold. First, the uniqueness of each key process area becomes simply a specialization of the framework. As an organization evolves to higher levels of maturity it simply specializes (re-uses) the framework each time a new key process area is addressed. Second, a business process automation of the entire SW-CMM becomes simply specializations of the framework. Metaphorically, the framework is the common platform upon which very different vehicles are built. Alternatively, it is the common framing upon which very different houses are built.

References

Bberzins, V., Gray, M., and Naumann, D. May 1986. "Abstraction-Based Software Development" *Communications of the ACM vol. 29(5), p. 403.*

Booch, Grady, "Object-Oriented Analysis and Design with Applications", Benjamin/Cummings Publishing Company, Inc. Redwood City, CA, 1994.

Brodman, Judith G. and Donna L. Johnson, "Return on Investment from Software Process Improvement as Measured by U.S. Industry," CrossTalk, April 1996, pp. 23-29.

Hayes, Will and Zubrow, Dave "Moving On Up: Data and Experience Doing CMM-Based Process Improvement", Software Engineering Institute, CMU/SEI-95-TR-008, August 1996.

OMG Unified Modeling Language Specification Version 1.3, 1999, Object Management Group, Framingham, MA 01701

Paulish, Daniel J. and Anita D. Carleton, "Case Studies of Software Process-Improvement Measurement," IEEE Computer, 1994, pp. 50-57.

Paulk, Mark C., Bill Curtis, Mary Beth Chrissis, and Charles V. Weber, "Capability Maturity Model for Software, Version 1.1", Software Engineering Institute, CMU/SEI-93-TR-24, DTIC Number ADA263403, February 1994.

Paulk, Mark C., Charles V. Weber, Suzanne M. Garcia, Mary Beth Chrissis, and Marilyn W. Bush, "Key Practices of the Capability Maturity Model, Version 1.1", Software Engineering Institute, CMU/SEI-93-TR-25, DTIC Number ADA263432, February 1994.

Saiedian, Hossein and Richard Kuzara, "SEI Capability Maturity Model's Impact on Contractors," IEEE Computer, January 1995, pp. 16-25.

Software Engineering Institute, web site at www.sei.cmu.edu. 1998 (accessed January 2001)

"Workflow and Internet: Catalysts for Radical Change. A WFMC White Paper", Workflow Management Coalition, at www.wfmc.org, 1998

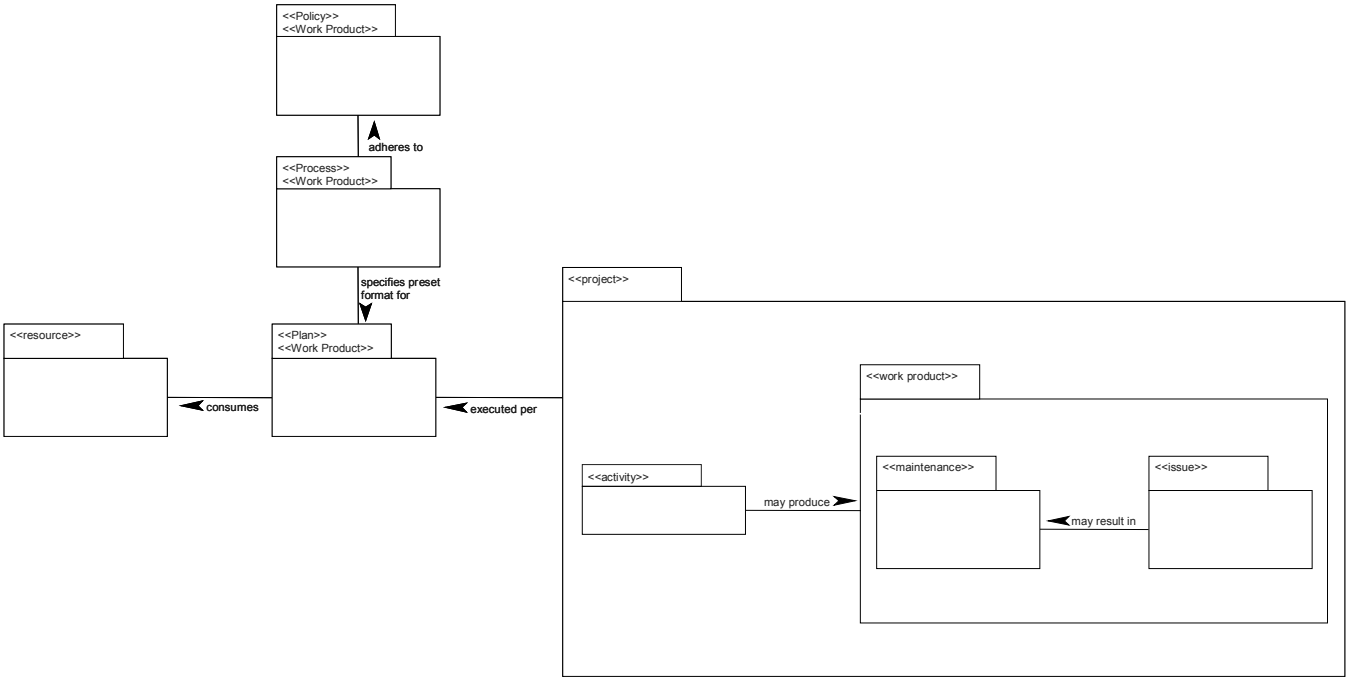

Figure 8. Framework Built Using the Metamodel