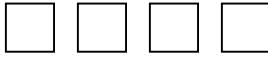


# Six Sigma Meets Personal and Team Software Processes

Creating Disciplined Software Development Processes for Financial Services Providers

An MKS White Paper  
By Christian Middel  
Senior Consultant  
MKS



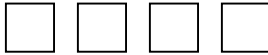
## Introduction

A Gartner internet usage survey from January 1999 reported that about 181 million users worldwide received financial products or support via the internet. That number had more than doubled to 407 million users by the middle of 2001<sup>i</sup>. Equally significant is the emergence of the “virtual enterprise”, which requires financial services providers to improve accessibility to stakeholders and integrate easily with the systems of other members of the supply chain<sup>ii</sup>.

While these developments are exciting, changing the face of banking as we know it, they also represent tremendous challenges for financial services providers. New internet business models bring with them a need for rapid deployment, seamless operation and scalability of software applications. New software development tools are emerging to ease this transition, but the ultimate success of any given software application depends on its quality. Quickly deploying a highly scalable application is a commendable achievement in itself, but if users encounter bugs and error messages, preventing them from receiving a company’s financial products or support, then all the work is for not.

Software configuration management (SCM) and change management tools cannot guarantee the quality of software applications by themselves. Defining and implementing solid processes to support software development is the key ingredient to producing high quality software applications. To this end, more and more financial services providers are adopting company-wide methodologies for improving processes at all levels. Methodologies such as Total Quality Management (TQM), Balanced Scorecard and Six Sigma are all popular, with Six Sigma receiving the most attention recently.

This white paper examines two software development methodologies that support Six Sigma by improving both individual and team development processes. The ultimate goal of these methodologies is to provide a framework that support the production of high quality software applications. These methodologies (Personal Software Process (PSP) and Team Software Process (TSP)) should be viewed in the larger context of Six Sigma, which is an organization-wide approach to improving processes in large and diffuse organizations.



## 1 Six Sigma

Six Sigma is a business improvement approach with systematic, mathematical and management underpinnings that position it as a methodology for accelerating improvements in quality, productivity and overall operating efficiencies<sup>iii</sup>. With the world economy slowing, revenues at financial services providers are shrinking, creating a need to drive down costs. This environment has also spawned a spate of mergers, which has naturally led to environments where some rationalization of systems and processes are in order<sup>iv</sup>. As a result, numerous financial services providers have already instituted Six Sigma initiatives:

### Six Sigma Initiatives in Financial Institutions

Financial Institution	Key Executive(s)	Stated Goal
<b>Bank of America</b>	Charles Goslee, process-management executive, formerly at Kodak; Joseph Valasquez appointed Technology & Operations Quality and Productivity executive, formerly at Sunbeam	Reduce errors, streamline processes and enable cross-platform selling. Goslee has promised CEO Ken Lewis to save \$1 billion in 2002
<b>Conseco</b>	Ruth Fattori, executive vice president for process and productivity, formerly at General Electric	"Process Excellence" program being launched with an initial 231 projects aimed at improving customer satisfaction and lowering costs in the range of \$100 million in the first year
<b>J.P. Morgan Chase</b>	Martha Gallo, managing director of Six Sigma program and newly named director of LabMorgan	LabMorgan being reoriented from external focus to focus on Six Sigma and process quality improvements

Source: Gartner Dataquest (November 2001)

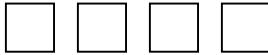
One of the most visible areas targeted for process improvement, because it so dependent on processes, is software development. By implementing well defined, scalable and disciplined processes at the individual and team levels, financial services providers benefit in two ways. First, because processes become streamlined and uniform across the organization, they are able to deliver applications to their customers (internal and external) more quickly. This meets the need for rapid deployment. Second, and most importantly, the quality of software applications improves because mechanisms are in place to prevent and track defects throughout the development process. This meets the need for maintaining customer satisfaction through well functioning web sites and online resources.

## 2 Disciplined Software Process

Experts and analysts agree that the greatest potential for improved overall software development lies in the improvement of development processes. In past decades, scientific and academic researchers did essential groundwork in this area. There are now numerous software process models available such as the CMM Model<sup>v</sup> and the CMII Model<sup>vi</sup>. The focus of this white paper is the "Disciplined Software Development" model<sup>vii</sup>, which is a software process based on the CMM model. This model can be separated into two parts: The Personal Software Process (PSP) and the Team Software Process (TSP).

### 2.1 The Personal Software Process

In the early 1990s, the SEI (Software Engineering Institute) at Carnegie Mellon University in Pittsburgh developed the Personal Software Process (PSP). The originator was Humphrey Wells, who also developed the CMM model. PSP's goal is "to enable individual developers to deliver high quality software products as quickly as



possible<sup>viii</sup>. The PSP concentrates on the role of the developers and exclusively on the coding stage of development. It takes as its starting point the software project's lifecycle, which encompasses a requisition analysis, rough planning, coding and integration/testing of the system.

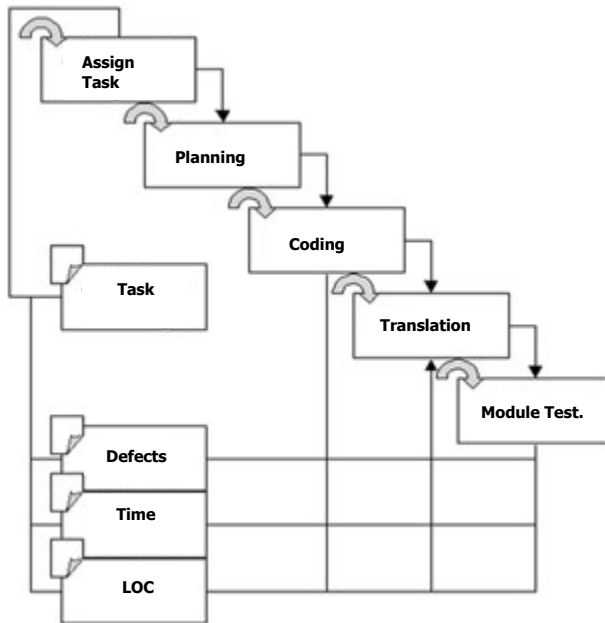


Diagram 1: PSP Summary

During implementation of the process, the PSP stage starts with a development task, which continues through the fine planning, coding, translation and module testing (see Diagram 1).

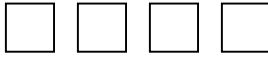
The PSP presumes that each step will be done in sequence; back tracking is not planned.

To ensure continuous improvement, the PSP includes three measurements. First, the process registers and records defects for the development task and the PSP stage. Second, the time needed to complete the development is assigned to the development tasks and, third, the extent of the entire development in Lines-Of-Codes (LOC) is reported for each development task (see Diagram 1).

The PSP quality model is based on empirical data, which can be defined as "Produced Correct Software"<sup>ix</sup>. To achieve this 'high goal' the PSP uses various quality measurements such as a review of the individual developer based on reported defects. It tries to eliminate individual and typical errors. Based on the recorded data, the PSP produces checklists, which are an important basis for the entire quality model as it sets out clearly defined instructions for the production of expense estimates and for the general improvement of personal processes. Empirical research on the PSP shows strong improvements in the areas of expense estimations and product quality.

## 2.2 Team Software Process

The Team Software Process (TSP) is an extension of the PSP. In this case, the most important component of TSP is the team whereas the PSP deals exclusively with methods and applications used for the management of tasks for an individual developer. The TSP emphasizes the identification and the implementation of various roles for the team, including the creation of backups of each job. In the process, the TSP assigns jobs such as Customer Interface Manager and one (or more) Design Manager(s), Implementation Manager, Planning Manager, Process Manager, Quality Manager, Support



and Test Managers. The project leader's position represents the only link between the Software Project Team and the rest of the company's organization (see Diagram 2).

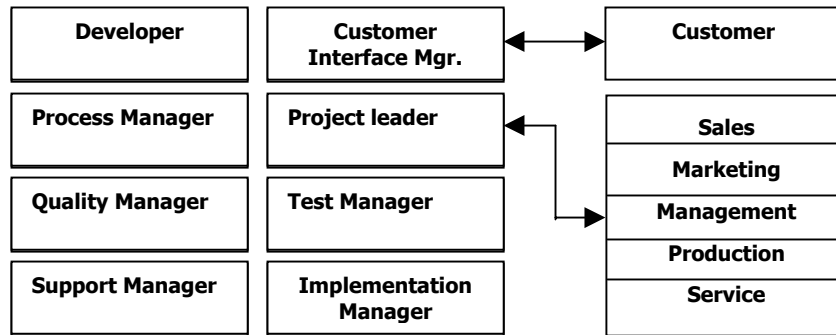


Diagram 2: Organizational aspects of TSP

The project launch is the first methodical step. For the launch there are detailed descriptions of the application process, which include among other things the creation of a quality plan, a project plan, an analysis of the project risks and, in the next step, the establishment of the individual tasks.

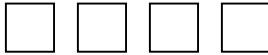
During weekly meetings the previous project stage is analyzed. Based on this analysis, the planning for the next project stage occurs. A quality profile for each module of the software project is created, which reflects the frequency of defects during translation or the frequency of defects during the module testing. After this stage, further project planning follows and the team sends progress and status information to management at regular intervals. Experience with TSP shows that deadlines are met with regularity and the quality of the products is notably high<sup>x</sup>.

### 3 SCM and Change Management as a Base Technology in Software Processes

"Configuration Management is the discipline of organizing and controlling evolving systems ... Software Configuration Management (SCM) is the specialization of Configuration Management for software systems."<sup>xi</sup> One of the biggest problems in software development, Tichy describes, is the "inadequate documentation of changes," which results in a, "painful loss of information"<sup>xii</sup>. SCM counters this loss of information, "with an identification of all involved elements through control mechanisms and examinations of the changes made in these elements. It also produces a report system that reviews the status of the product"<sup>xiii</sup>.

Change Management, "includes the process and the procedures for the identification, documentation, validation and release of software changes that software configuration management administers"<sup>xiv</sup>. Together, SCM and change management offer a broad methodological base for daily use in developing systems and implementing changes within these systems. These basic methods must be integrated in the appropriate places of the software development process to implement a software process model. And since SCM and change management provide the basic methods, the ultimate challenge lies in the identification of the suitable software process, finding the correct place for integration of the tools, and directing efforts for formalizing the software process through these tools.

The following section describes the application of tools within the PSP and the TSP.



### 3.1 Application of tools in the PSP

Understanding the PSP is the basis for software development when assigning a development task to a particular developer. Modern change management systems allow for the administration and workflow control of such tasks. For example, the MKS Integrity Manager™ introduces an entity or 'Task', for which there is a defined, detailed and gradual process for each stage. The developer is able to reproduce each task whether it is situated in the fine planning stage, the coding stage or in any other stage within the PSP.

The software configuration management system records all necessary elements that are required by the developer to fulfill his development task. This usually includes elements such as the original source code and design documents. The SCM tool will automatically register all changes that the developer carries out within his or her development task.

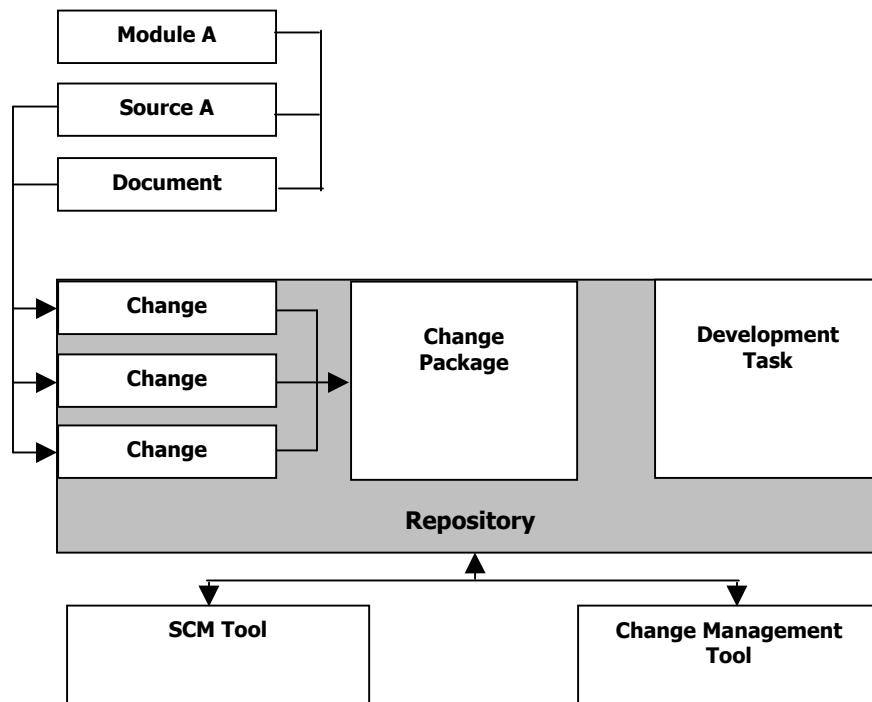
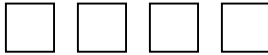


Diagram 3: Tools and Developing Tasks

Modern, advanced SCM tools allow for the reproduction of all connections between the original task and the sum of all changes in the software system (final version). These are called "Change Packages" when using MKS Integrity Manager and MKS Source Integrity® Enterprise Edition (see Diagram 3). In addition to reproducing the task, most change management tools (including MKS Integrity Manager) allow for the designation of various entities, including 'Defect'. The Defect entity records the defects that occur during the PSP and traces the resolution of the defects within the PSP.

By introducing the entities 'Task' and 'Defect', three important measurements of the process are covered. First, the 'Defect' entity of the change management system completely reproduces the measurement of the defects. By assigning PSP stages to corresponding task stages, metrics are produced, which function as the base for the PSP quality model (i.e. frequency of defects during the stage module testing).

Second, there is a timing aspect to the PSP, which records all time expenditures for the completion of the task. This can be achieved in two ways. Either the entities 'Task' and 'Defect' have the corresponding time-related



information (i.e. summary areas), or the process introduces a new entity for timekeeping, which must be connected to defects and tasks. Time expenditure is important for estimating future expenditures and assessing relevant resource capacity.

Third, the measurement for 'Lines-Of-Code' (LOC) is automatic because the SCM tool stores the so-called 'Delta' records wherever there are changes to the elements. As a result, the number of changed or added code (lines) is always available. It is important to note that Change Packages not only make an individual connection with the task but also to the defect. Consequently, other measurements can be determined such as the relative expense for the resolution of defects in each PSP stage and task.

### 3.2 Application of Tools in TSP

The TSP, as a process for the entire team, has different priorities than the PSP. This is reflected in new requirements. The priority of the TSP is to assign team members different roles and corresponding tasks. Besides internal tasks, such as the appointment of development tasks in order to achieve the next project milestone, or a review of quality measurements to improve the process and identify risks, there are also external tasks for the team (i.e. customer service, reports to the management and other departments of the company – see Diagram 4). Here, the change management system can be an important and helpful 'tool'. Within the change management system – analogous to the explanation of the PSP application – further entities can be designated, with which a base for information and corresponding workflows can be defined. For example, an entity called 'Error Message', which is important for the communication between team and customer, can be designated. The team determines what information or content is necessary to analyze and solve any given error. The entity 'Error Message' is defined according to the requirements, and made available to the 'outside world' (customers) in an online form. The workflow definition guarantees the route that the error follows through the system. Simple status-transfer models and 'escalation mechanisms' make sure that no single error is forgotten.

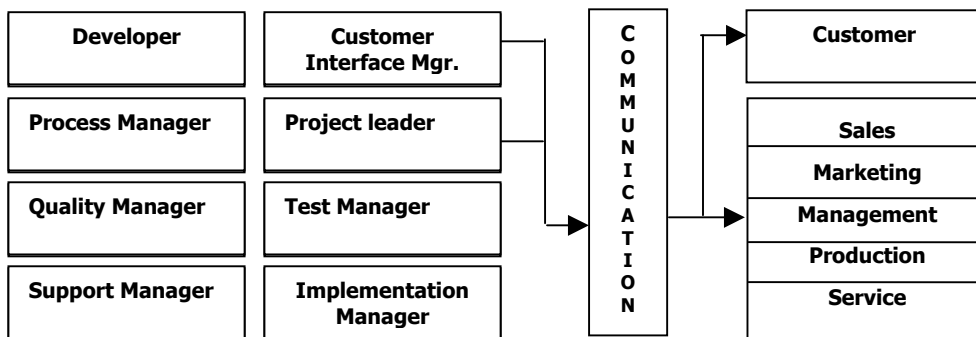


Diagram 4: Intersection of communication

Occasionally, other entities are identified and defined as a result of continuous process review. These new entities can be defined and integrated into external team processes and then reproduced through the change management system. Examples of these are requests for extensions, technical queries, test reports, status requests, expense estimations etc. In summary, with the help of a change management system, all of a company's incidents in the 'real world' can be formalized (see Diagram 5).

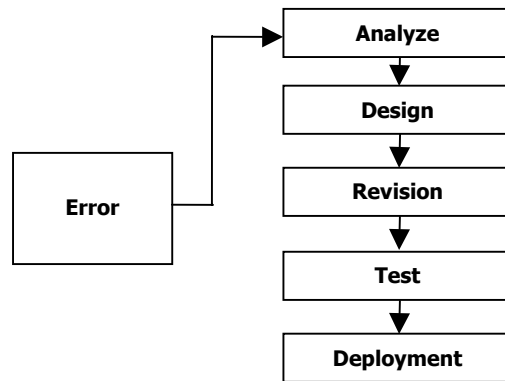
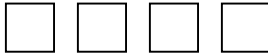


Diagram 5: Entity 'Error' and corresponding workflow

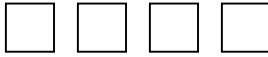
After the connection between entity 'Task' and the other entities is made, all possible information related to that task that is useful to the team can be generated through the created database (repository). By doing so, a complete "package" of all changes in the source code and all changes within the documents related to the task will be connected to the original error message. Design changes can, therefore, be investigated at their source (i.e. design changes by customer request).

The application of a change management tool and defined process is fundamental to the TSP, enabling suitable and appropriate assignment of tasks to team members. Only when a standardized course of action of identical structured demands for the team (Intersection of communication, see Diagram 4) is taken, can useful and complete distribution of tasks occur. Then the tasks can be allocated their proper roles.

The Change Package plays an important role in the interaction between change management and software configuration management systems, especially MKS Integrity Manager and MKS Source Integrity Enterprise Edition. By knowing which version of source code or document was produced to implement a critical change, further quality measurements can be introduced. Before the delivery of the completed software product, one is able to compare the entire set of all critical changes to the configuration of the delivered product. Consequently, one can automatically discover sources of errors that are based on certain elements in delivered versions that are tainted with critical errors. These are only some examples of the application of software tools in software processes.

### 3.3 Opportunities for Tools from a Technological Standpoint

As shown in the previous sections, modern software configuration and change management tools are parts of integrated processes. As a result, a standardized and complete database (repository) of software changes and corresponding tasks is produced. Of course, from a technological standpoint there are many starting points for the application of these tools. One idea is the introduction of online forms that are generated by the change management system (i.e. MKS Integrity Manager). With the aid of such online forms, one is able to connect directly with customers or internal departments to collect information without using additional means of communication like email and phone. Within MKS Integrity Manager there is even a notification option that informs process constituents of any changes. Customers can receive these notifications continuously and can be updated about the status of their error messages.



## Conclusion

Software development in the financial services industry, with its dynamic and changing business and IT requirements, is a challenging discipline. SCM and change management tools continue to increase in functionality and sophistication, but the tools themselves do not present the entire solution for quality, on-time software development. The tools must be wedded to processes that will facilitate both individual developer and team success on the project. Nowhere is this more relevant than in financial services organizations that have implemented Six Sigma initiatives. This white paper provides financial services providers with a starting point for integrating SCM and change management tools with proven development methodologies. The PSP (Personal Software Process), which is based on the CMM, concentrates on the coding stage of development and provides a structured method for ensuring individual developers have a repeatable process with ongoing mechanisms for quality and continuous improvement. The TSP (Team Software Process) addresses the collective processes that the overall project development team requires to ensure speedy delivery and high quality. There is a heavy emphasis on task management and team communication along with quality and continuous improvement measures. MKS Source Integrity Enterprise Edition and MKS Integrity Manager can help teams transform these ideas for software development into tangible, day-to-day activities with measurable results.

## About the Author

Christian Middel studied Business/Computer Science at the Ravensburg Vocational Academy (Berufsakademie). Immediately following his studies, Mr. Middel worked as a Software Engineer in the development of software for accounting systems. Based on his accumulated experience as a software developer, he worked as a consultant for the successful implementation of Software Configuration and Change Management solutions. Today is Mr. Middel a senior consultant for MKS GmbH in Germany.

---

<sup>i</sup> Financial Services 2006: Delivery of Banking and Finance, Gartner Research, October 2001, p 16.

<sup>ii</sup> *ibid*, p 19.

<sup>iii</sup> Financial Services Quality Initiatives: Is Six Sigma On Your Radar Screen?, Gartner Inc. Dataquest Alert, November 9, 2001, p 1.

<sup>iv</sup> *ibid*, p 2.

<sup>v</sup> Analysis and Evaluation of Software Development in Germany: GfK Marktforschung (market study) GmbH, Fraunhofer Institute for Experimental Software Engineering, IESE, Fraunhofer Institute for System Engineering and Innovation Research, ISI, December 2000.

<sup>vi</sup> CMM – <http://www.sei.cmu.edu>

<sup>vii</sup> CMII – <http://www.icmhq.com>

<sup>viii</sup> Software Technology Support Center, US Air Force: Disciplined Software Development Report, 4/30/99, Utah.

<sup>ix</sup> The Personal Software Process (PSP): An Empirical Study of the Impact of PSP on Individual Engineers, Will Hayes and James Over, CMU/SEI-97-TR-001, 1997, Utah.

<sup>x</sup> Using the TSP on TaskView Project, D. Webb and W.S., Crosstalk, February 1999, pp. 3-10.

<sup>xi</sup> Walter F. Tichy: Configuration Management, 1994, Karlsruhe

<sup>xii</sup> ISO-9000-3:1991(E): Quality Management and Quality Assurance Standards – Part 3: Guidelines for the application of ISO 9001 to the development, supply and maintenance of software, International Organization for Standardization, 1991.