SOFTWARE PROCESS REUSE IN AN INDUSTRIAL SETTING

Craig Hollenbach William Frakes

Department of Computer Science
Virginia Tech
2990 Telestar Ct.
Falls Church, VA 22042
703-698-4712 Fax: 703-698-6062
wfrakes@vt.edu

Abstract

This paper describes a method for creating reusable processes and our experience using them in an industrial environment. A notation and process for creating and tailoring reusable processes is described and applied to the building of a 120 process library at PRC Inc. Initial data collected on use of the library indicates large potential payoffs from process reuse such as a 10 to 1 improvement in the time to develop a project specific process.

Keywords

Software Reuse, Process Reuse, Domain Analysis, Process Notation, Process Tailoring

Introduction

Many software organizations are actively pursuing software process maturity. Often they use measures like the Capability Maturity Model [Paulk 93] to structure their process improvement initiatives. One method often cited for accelerating process improvement within an organization is to replicate a standard, reusable organizational process within other projects. However, the creation of a process that is usable on diverse projects is a difficult task. What is needed is an effective method to capture the common and variant elements of project-specific processes and create process definitions that can be applied in a variety of situations, i.e., that are reusable.

A literature search shows that groundwork for process reuse exists. Research by [Castano 93] concerns the abstraction of common objects from a set of object-oriented process definitions. Research at the DoD-sponsored STARS project [Ett 95], the Software Engineering Institute [Over 94], and the Software Productivity Consortium [Bechtold 94] has developed a set of process notations and methods for defining single processes. These process notations are primarily built upon the ETVX (Entrance criteria, Task, Validation, and eXit criteria) format developed at IBM [Radice 85]. Some large US companies have demonstrated instances of process reuse. As examples, IBM has replicated a defect prevention program [Gale 90] and peer review program [Fagan 86] within its divisions. Other companies have developed process enactment tools, such as Process Weaver [ProcessWeaver 92], Synervision [Synervision 93], and PEAKS [Ett 93].

The creation of reusable processes is dependent on domain analysis. Many domain analysis techniques for creating reusable software have been developed. [Arango 94] has surveyed these techniques and created a composite domain analysis technique . This composite technique is used as a basis for the methods described later in this paper.

This paper investigates how to pragmatically and systematically standardize and replicate project-specific processes in an industrial software setting. A systematic and standard method for process reuse is presented. The methodology is applied to a set of PRC business units undergoing software process improvement, and measurements of the level of reuse before and after the application of the process reuse methodology is reported.

Using domain analysis and implementation techniques, software reuse principles, process architecture research, and process modeling mechanisms, project-specific processes can be abstracted into a reusable form that can be retrieved and tailored by process engineers.

What is a reusable process?

[Feiler 92] defines a process as "a set of partially ordered steps intended to reach a goal." We define process reuse as the usage of one process description in the creation of another process description. It is not multiple executions of the same process on a given project.

What makes a process reusable? To answer that question it is important to examine what makes software reusable. Significantly, that question has not been definitely answered even though it has been researched for over ten years; there is no foolproof external indicator that can determine the reusability of a software module [Frakes 96]. The best answer to date is simply that the software module is reusable only if it is reused. The same can be said for software processes; if reused, they are reusable.

The 3Cs model of reuse design provides a framework that has been found effective in the design of reusable assets. [Latour 91] The model indicates three aspects of a reusable component - its *concept*, its *content*, and its *context*. The concept specifies the abstract semantics of the component, the content specifies its implementation, and the context specifies the environment necessary to use the component.

3 C's Aspect	Code	Process
Concept	Abstract Data Type (ADT)	Informal specification of: • General information • Customer description • Interface description
Content	"Implemented in C with link lists"	Procedural description (graphic & text)
Context	"Operating system = UNIX, hardware = Sun, compiler = ANSI C"	Contextual description

Table 1 - Examples of the 3Cs model

Table 1 shows an example of the 3Cs model applied to a software component and to the reusable process notation described in appendix A. For a software component, the concept might correspond to an abstract data type (ADT), whose implementation might be a C module using linked lists. This component's context might require a Sun workstation running UNIX, and an ANSI C compiler.

Analogously, a reusable process will have for its concept an informal specification of: the general information, the customer description, and the interface description. The content will include the procedural description utilizing both a textual and graphical representation. The reusable process context will be defined in the contextual description.

Generally speaking, a process is more reusable when it can be used in various situations without changing its concept. The concept in this case is the portion of the process description that defines the customer and associated requirements, the interfaces, and other general information. An example of a reusable process is shown in Figure 1. Here the "estimate software complexity" process is shown. In the course of the process, a complexity measure is needed; there are three methods used to measure complexity: a Halstead measure, a McCabe measure, and a lines of code measure. The use of any of these methods should not affect the concept of the process; indeed, the degree to which it affects the concept is the degree to which it is less re-usable.

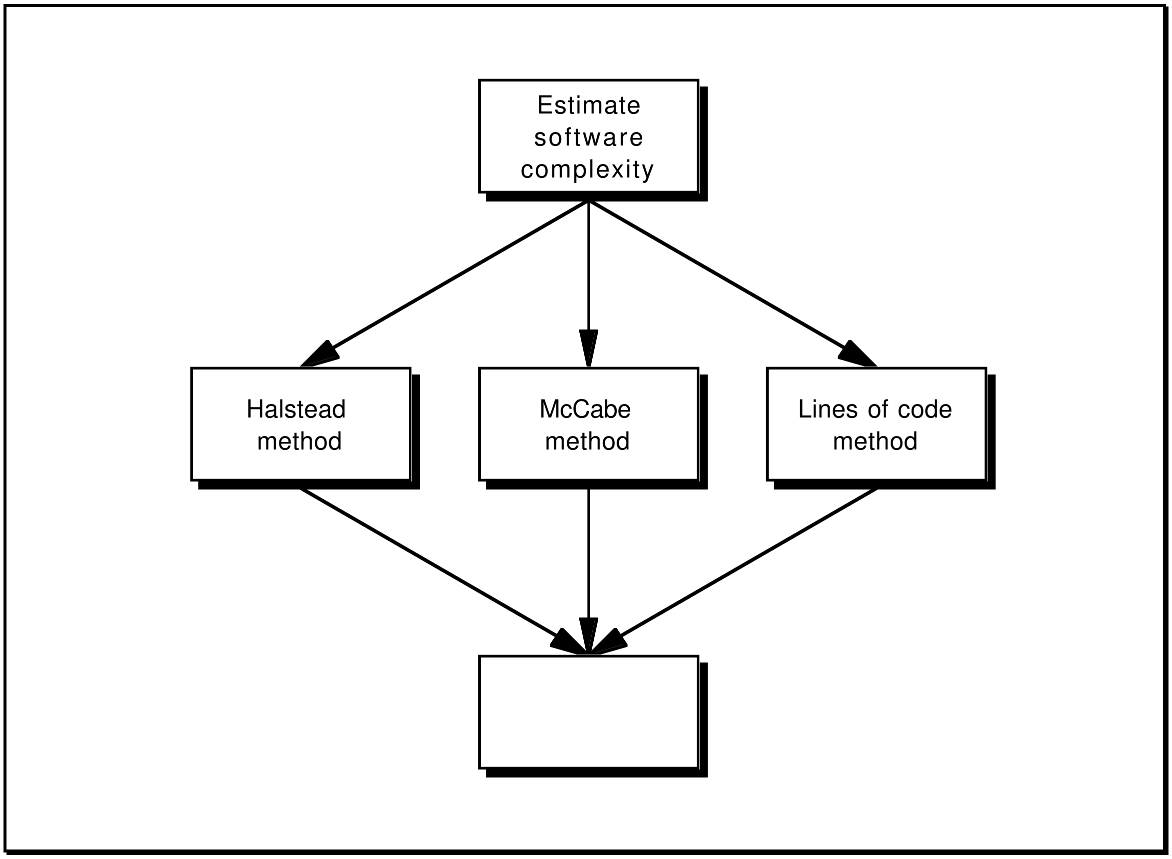

Figure 1. A Process Reusability Example

1. Method

1.1 Notation for Recording Reusable Processes

For processes to be reusable, organizations need a way to express common and variant elements within a process. Frameworks provide one mechanism to accomplish this. Appendix A shows a framework for process definition which is used within PRC. Several portions of the framework reflect the integration of PRC's software process improvement and quality improvement programs; see [Arthur 93] for details on this approach to quality.

This framework is organized into several sections. The first three sections, known as the general information, customer description, and interface description sections, specify the process concept, or the *what* specification of the process. These sections describe *what* the process is, *what* is does, and *what* its interface is to other processes.

The next section, the procedural section, specifies the process content, or the *how* of the process. This section describes how the process is performed by specifying the people who do the process and their interactions, the tasks and the order of their execution, and the tools and other necessary resources. The procedural description is both textual and graphical.

The graphical notation uses flowcharting symbols. The flowchart describes the process, its customers, and the indicators used to measure compliance in meeting customer requirements. The graph is divided vertically into columns; each column represents a participant in the process. Process steps for each participant are placed in the associated column. The graph can also be divided into horizontal bands which represent sequential steps of the process. Each process starts with an oval symbol describing the customer's need. The steps in meeting those needs are described using rectangles for processes and diamonds for decisions. Boxes with double side bars represent a collection of sub processes that is defined graphically in a separate diagram. The arrows represent control flows. No attempt is made to capture data flows. Indicators, described below, are denoted using circular symbols.

The context description describes the context in which the process can execute. The process context, which is strongly bound to the process content, defines domain, organizational, project, and managerial criteria for application of the process.

The final section, the measurement description, spans both the *what* and *how* specification roles. The quality indicators measure how well the process meets its goal and are therefore part of the *what* specification. The process indicators measure how well individual steps perform. Because the process indicators measure steps, they are part of the *how* specification.

1.2 Context for Defining and Tailoring Reusable Processes

The purpose of the process definition methodology is to create reusable processes so that projects within an organization can cost-effectively tailor them to their own requirements. Reusable processes aid in transferring process knowledge between projects, reducing training costs, planning common project activities based on process data, and increasing productivity and quality in a continuously improving process-oriented environment.

The context of the process definition and tailoring methods are depicted in Figure 2, which shows the life cycle of a process description. The figure describes graphically the steps necessary to create a standard reusable process description, instantiate it for use on a project, refine its usage, and feed back information to improve the process description for subsequent use.

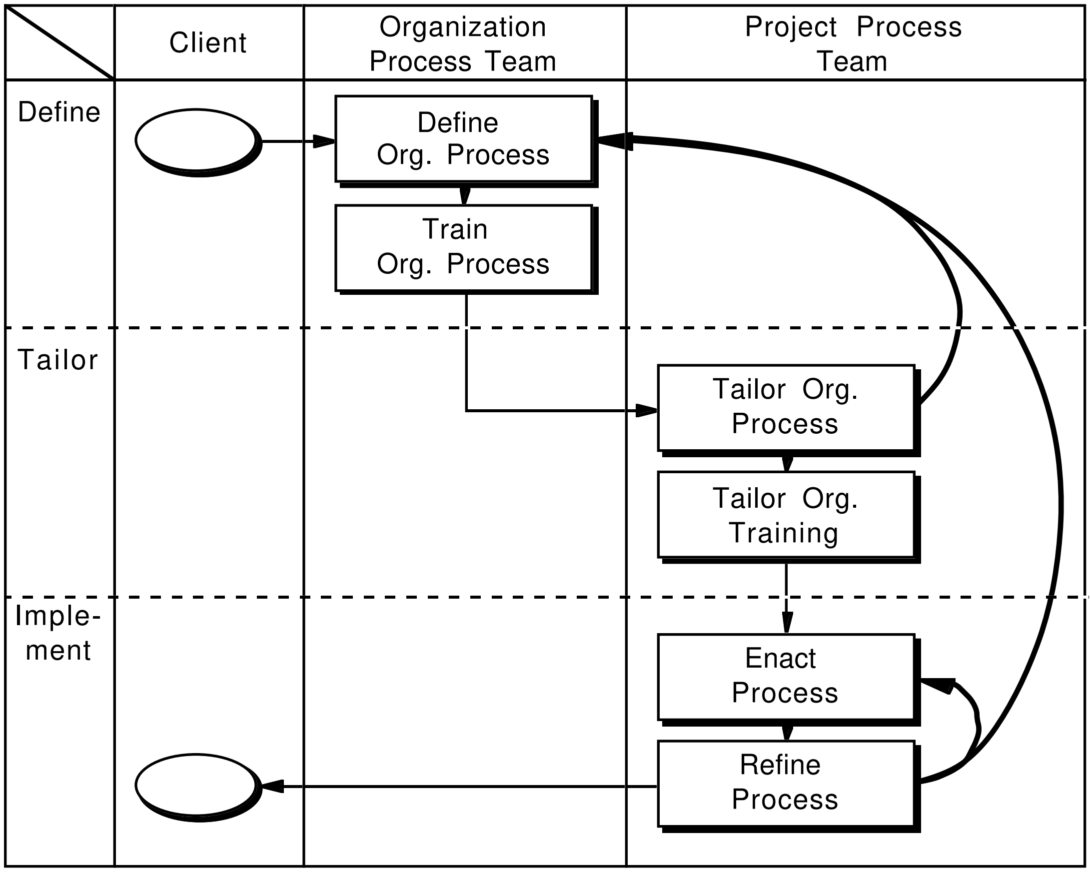

Figure 2. Context for Defining and Tailoring Reusable Processes

This process life cycle contains the following steps:

<u>Define the Reusable Process.</u> The output is one or more process descriptions, along with tailoring guidance and output work products; these are usually packaged into a process manual or handbook. The process descriptions also are integrated into the life cycle and statically tested to ensure they are fit for (re)use.

<u>Develop Training for the Reusable Process.</u> Once the reusable process is completed, training for that process is developed. That training is tailored as described below.

<u>Tailor the Reusable Process on a Project.</u> The reusable process is tailored to meet the specific requirements and environment of a project. Project-specific process descriptions are the result, sometimes packaged into a handbook or set of desk instructions.

<u>Tailor the Reusable Process Training to the Tailored Project Process.</u> The training for the reusable process is tailored to match the project-specific

process. The training is given to the appropriate staff just before they need to enact (or execute) the process.

Enact the Process on the Project. The project-specific process is enacted, i.e., put into practice on the project. CMM level 3 project management practices track the enactment of the process and use it to monitor and control the project. The quality assurance function audits the project staff to ensure that the process is faithfully enacted. Measurements are taken and used in the following step.

Refine the Process. Based on measurements collected in the previous step, the process is evaluated to see if it is stable and capable. If not, an analysis is performed to determine why the process failed. The process definition is then refined, new training is developed, and the process is re-enacted.

1.3 Method for Defining Reusable Processes

This section describes how to create a reusable process description. The objective of this methodology is to develop and maintain a reusable set of software process descriptions that improve process performance across the projects and provide a basis for cumulative, long-term benefits to the organization. The method assumes there is a system to maintain and support reusable process definition efforts, and supporting policies and adequate funding and resources.

The customers of a reusable process description are: 1) the managers and SEPGs (Software Engineering Process Groups) [Paulk 93] of projects that want to reuse a process, either during proposal development, at project startup, or for a process improvement initiative; 2) the developers of the associated training for the reusable processes; and 3) corporate SEPGs tasked with managing process implementations and changes. These customers require that the reusable processes be enactable, comply with standards, and improve productivity and quality.

The methodological steps to create a reusable process are described below. The method derives from a generalization of standard software development and domain analysis techniques, particularly Arango's domain analysis of domain analysis methods [Arango 94].

1. Process Requirements Analysis: This step defines the operating conditions under which the process will be enacted and the goals and requirements that the process will satisfy. A process definition and evaluation action plan is developed that contains this information plus basic project planning data.

2. Process Analysis: The process analysis step supports generic reusable process definitions by examining current project-specific processes and relevant industry process data. Domain boundaries and content are specifically described and current processes and process data are collected The data is evaluated to locate entities, events, operations, or relationships which are then modularized. Finally, these items are analyzed to determine similarities, variations, combinations, and trade-offs that help structure the process for reuse.

3. Process Preliminary Design: In this step, precise and accurate descriptions of the process concept, or *what* specifications, are created. For each reusable process, the customers and their requirements are defined as are the process inputs and outputs, entrance and exit criteria, and quality indicators. Major sub process blocks are defined for upper level process hierarchies.

When designing the process interfaces, it is important to integrate the process into other existing processes. This implies that a process architecture has been constructed consisting of processes from all software engineering disciplines. The architecture is the starting point for integration of the reusable process. It also provides a framework for reference during reusable process definition.

To determine a process's quality indicators, the goal-question-metric paradigm [Basili 84] is applied to the definition of the process customers and their requirements, as defined in the preceding step.

4. Process Detailed Design: The process detailed design step defines the process content, or the procedural, *how*-related process characteristics. They include methods/procedures, roles, process indicators, and feedback and control mechanisms.

It is important to use roles rather than agents in the reusable process. Roles refer to required task-oriented functions. Agents are the people or groups that perform the roles. Agents differ greatly between projects, based on project size, contractual commitments, subcontract arrangements, past project organizations, and other project environment concerns. Because of this variability, the reusable process descriptions describe functions and allow the projects to map the functions to the organizational agents as appropriate.

5. Process Code and Unit Test: The process code and unit test step is used to automate a reusable process or a portion of the process. Trade studies are performed that weigh the benefits of commercially available tools versus the construction of an internally developed tool set. If commercial tools are selected, there is the inevitable task of tailoring the tool to meet the characteristics of the organization or project. In this case, the process definition and its contextual process architecture provide the necessary information to effectively tailor the tool; in other words, process definition is a necessary precursor to process automation. If local automation is done, the process description then becomes the functional specification for the automated process. In this case, the process code and unit test step transforms the enactable process description into an executable process program.

6. Process Integration and Testing: The process integration and testing step ensures that the defined reusable process can meet product and process goals. This is usually accomplished through static analysis of the process and then the process architecture which contains the integrated process.

The method to create a reusable process can itself be measured using quality and process indicators. Some potential quality indicators are:

- Number of projects consulted per organization process.
- Percent of project-specific process reuse per organization process.
- Standard set of project tracking metrics, including total effort, milestones missed/made, and cost.

1.4 Method for Tailoring Reusable Processes

Process tailoring consists of importing a reusable process into a new environment and changing it, if need be, for a new project's requirements. The major steps, illustrated in Figure 3, are usually performed by a team of project or business unit staff or by a expert in the process area. The output of the tailoring method is a set of process descriptions that textually and graphically describe the project-specific process. The tailoring is performed at the direction of the local SEPG; the tailoring plan is usually documented in the SEPG's software process improvement plan.

After initial planning of the tailoring effort, the tailoring team selects a reusable process. The team then adds detail to the customer definition section of the reusable process so that it describes the project's customer. Through discussions with the customer, requirements are elicited, adding project-specific details to those abstracted in the reusable process description.

Using knowledge of the specific project input and output work products and the existing process architecture, the team adds project-specific detail to the set of interface specifications in the reusable process descriptions. The team also tailors the reusable process metrics to measure how well the specific customer’s requirements are satisfied. The metrics are tuned to the peculiarities of the project contract, available collection methods, perceived risks, and maturity level, among other things.

Next, the process methods and associated indicators are tailored to accommodate the specific functional assignments on the team, expertise in currently used methods, differences in task sequencing, and existing project processes. The team then identifies opportunities to (semi-)automate the process, integrate it into the process architecture, and test it. The testing is usually accomplished through peer reviews, augmented by standard testing techniques for the automated portions of the process.

Finally, the tailored process is given to the local SEPG for final approval, along with all applicable waivers and deviation requests, and when approved, added to the organizational repository.

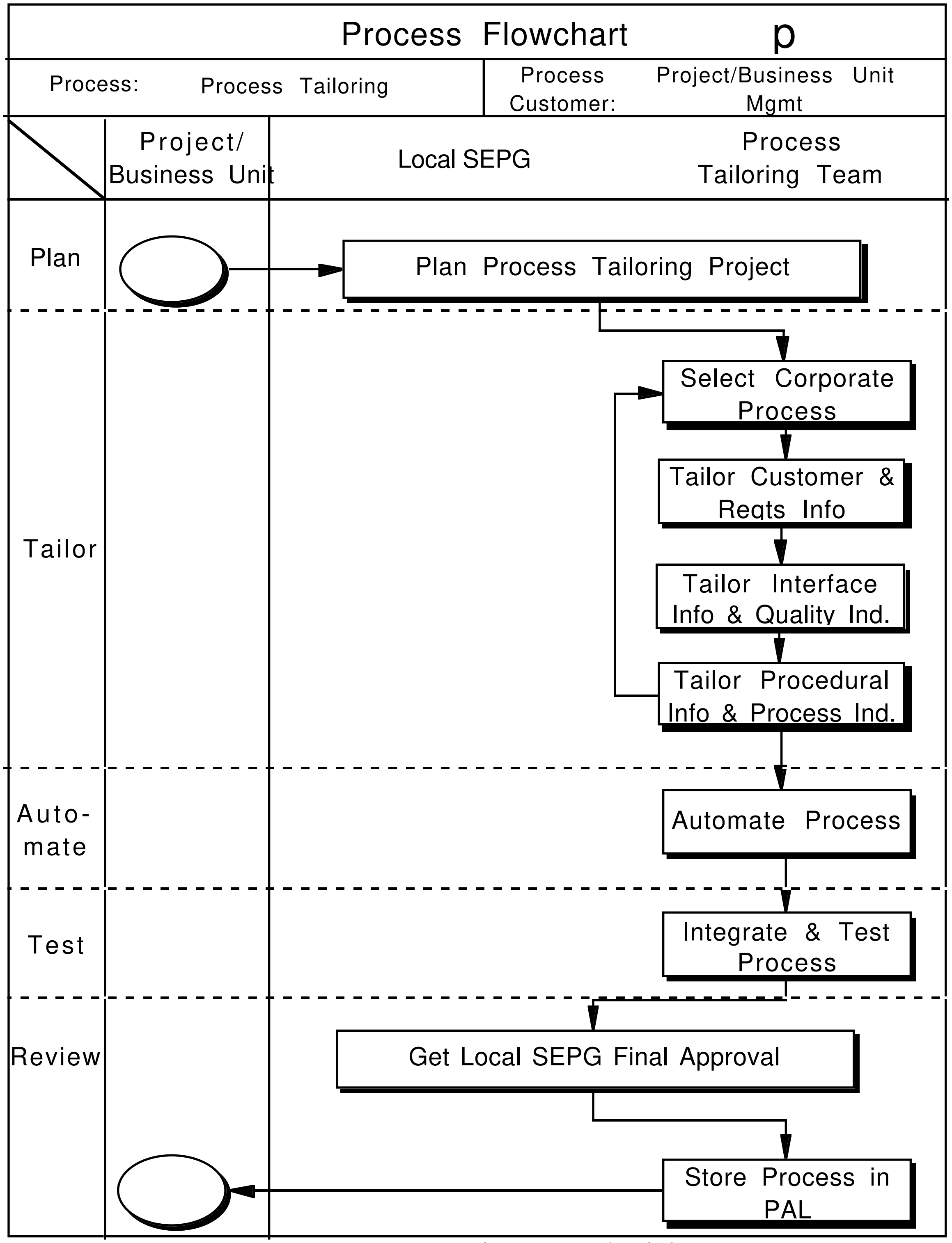

Figure 3. Process Tailoring Methodology

2. Case Study

Although PRC has been involved in software process improvement since 1989, this case study begins in 1991. PRC then began to work with individual projects to increase their software maturity. Efforts were centralized into one group of full-time staff in the company's technology center. Based in part upon the experience of a pilot project, this group developed a set of approximately 300 corporate process descriptions in the level 2 key process areas, and also the peer review key process area. This set of process descriptions, referred to as Release 1.0 below, was collected into a set of process manuals. Although the pilot project used these process descriptions, there was little interest among other projects in software process improvement and process adoption.

In 1993, a corporate initiative merging the company's software process improvement and quality improvement programs secured the involvement of eleven key software projects, assisted by the full-time group from the technology center. These projects and the technology center staff formed the "phoenix" team. In this collaboration, process definition and improvement operated was more decentralized. Projects defined their own processes, sometimes using existing process definitions. To maximize process maturity, the concept of "best of breed" (BoB) processes was tried; attempts to replicate processes from the most mature projects to more immature projects were employed. This failed almost totally. The team determined that a more methodical approach to process reuse was needed, one that abstracted away project-specific details so that other projects could reuse the processes.

Consequently, in mid-1994, research was begun on how to maximize process reuse. Project representatives were surveyed to characterize process reuse as of June 1994. A working group, initially formed to create corporate training for level 2 key process areas, was also tasked to modify or redefine processes so that they were compliant with version 1.1 of the Capability Maturity Model. They used the method defined above to collect project-specific processes and data and to create reusable processes.

Release 2.0 of corporate process descriptions, numbering about 120 processes, was created primarily as a result of their efforts. These processes were stored in a web-browsable Process Asset Library (PAL) which is accessible by PRC employees. Specific KPA web pages were built to house the reusable processes, project-specific instantiations, related process assets, and training material. The training associated with these processes was given first in pilot and then in corporate training situations. In each of these, the students were taught how to tailor the corporate/reusable processes to meet their project-specific needs. Finally, instances of process definition and/or reuse over the 1991 to 1995 time period were collected.

The results have several key points. First, contrary to past trends where reuse was tied to previous involvement in corporate process development, reuse increased for business units that had no clear link to the development of the reused organizational processes. Significantly, the time to develop a project specific process was dramatically lowered through reuse, showing at least a 10 to 1 decrease in time needed to create a process. This is significant in that two strong process improvement requirements voiced by PRC projects are 1) time to improve, and 2) effort to improve, since most improvement efforts are performed on tight, limited budgets.

Reuse Levels Compared: Has use of the method increased process reuse? To answer this question, we compared process reuse percentages in 1994 before the method was used and in 1995 after the method was introduced. As Table 2 shows, process reuse increased from 41% in 1994 to 55% in 1995. We believe that the domain analysis and process tailoring methods and training contributed to this increase. All business units participated in the training, and business units #4 and #7 have set a standard to begin all process definition efforts by identifying reusable processes. More time is needed to measure the effects of the domain analysis and process tailoring methods. The decrease in total processes from 1994 to 1995 is due to incomplete data from 1995 (through July only), and normal fluctuations in the process creation and use cycle.

Business Unit	**1994**			**1995 (to 8/1)**		
	Reused	**Total**	**% age**	**Reused**	**Total**	**% age**
Business Unit #1	0	38	0.00	0	0	0.00
Business Unit #2	0	50	0.00	1	14	0.07
Business Unit #3	26	31	0.84	37	42	0.88
Business Unit #4	0	0	0.00	3	4	0.75
Business Unit #5	35	41	0.85	0	9	0.00
Business Unit #6	37	79	0.47	10	26	0.38
Business Unit #7	0	1	0.00	3	3	1.00
Total	98	240	0.41	54	98	0.55

Table 2. Process Reuse in 1994-1995 by Business Unit

Benefits of Process Tailoring Method: One positive thing that came from implementing the process tailoring methodology was a greater awareness of the time and effort saved by tailoring a process rather than defining it from scratch. Metrics from both corporate and project process definition efforts place the effort spent to design processes for a KPA at around 800 to 1000 or more person-hours, depending on the breadth and depth of the KPA. One project process team serves as an example: this team built four processes in 800 hours, their efforts spanning over one full year. It could be extrapolated that one process took 200 hours.

Results from process tailoring show that a project can instantiate a process in two hours, one to tailor the process textual description, and one to tailor the graphical description of process roles and tasks. When a peer review of the process is added, the total time to instantiate and produce a project-specific process is around 20 hours. Thus, conservatively speaking, our PRC experience shows a 10 to 1 improvement in time to define a project-specific process.

Process Tailoring Consultation: Our experience with process tailoring covers several business units and KPAs. Release 2.0 processes have been instantiated and modified in the following KPAs: peer reviews, requirements management, software project tracking and oversight, software quality assurance, and intergroup coordination. In each of these cases, an "instantiated" or "instantiated and modified" process was created in two to three hours. Participants in this process are usually excited about its power, surprised at the results, and eager to apply the same method to other processes.

Reuse of processes within PRC is not limited to corporate processes. One business unit has taken advantage of processes from another business unit and also from a tool vendor. In the first case, no particular tailoring was required; the processes which involved cost planning and tracking were reused without modification. In the case of the process from the tool vendor, the business unit used the process and tool documentation as a reference for process definition, but were not able to instantiate or instantiate and modify the tool's process information.

3. Conclusions and Future Work

The application of reuse technology to software processes is a valuable addition to process engineering methods. Process reuse enables organizations to apply carefully crafted processes to a wide variety of projects and to apply them in a shorter time frame, thus saving time and effort. In this paper we presented a method for process reuse and reported on its use in an industrial setting. In our case study at PRC we found at least a 10X improvement in time and effort to create a project or business unit process description when we instantiated a reusable process rather than building the process from scratch. Future work should continue this case study to fully measure the effects of process reuse over a statistically significant set of process definition and tailoring instances. Additionally, more research is needed in analyzing the effects of process reuse on quality, cost reduction, and time to market.

4. Acknowledgments

The authors would like to thank Voytek Kozaczynski, Rebecca Joos, Shirley Browne, Martin Griss, and Maggie Davis for their comments during WISR-7, and PRC Inc. for their involvement in and support of the case study and paper.

5. References

[Arango 94] Arango, G. "Domain Analysis Methods," from Software Reusability, edited by Schafer, W., Prieto-Diaz, R., and Matsumoto, M., Ellis Horwood, 1994

[Arthur 93] Arthur, L., Improving Software Quality: An Insiders Guide to TQM, New York: John Wiley, 1993

[Basili 84] Basili, V., Weiss, D. "A Methodology for Collecting Valid Software Engineering Data", IEEE Trans. Software Engineering, Vol. SE-10, no. 3, November 1984, pp. 728-738

[Bechtold 94] Bechtold, R., Brackett, J. Process Definition and Modeling Guidebook, Volume 1: Concepts and Principles of MPDM, SPC-92041-CMC, Version 02.00.02, Software Productivity Consortium, Herndon, VA 22070, March 1994

[Castano 93] Castano, S., De Antonellis, V. "Reusing Process Specifications", from Information System Development Process, Proceedings of the IFIP WG8.1 Working Conference on Information System Development Process, Como, Italy, 1993

[Ett 93] Ett, W., Terrel, J., Lineham, A. Software Process Management System User's Guide for the IBM RISC System/6000 (SPNS/R*) Version 1.0 Renamed Process Engineering and Analysis Kernel System (PEAKS), Electronic Systems Center, Contract No. F19628-93-C-0129, Task IU03, IBM Federal Systems Company, 800 North Frederick Avenue, Gaithersburg, MD 20879, July 31, 1993

[Ett 95] Ett, W., Kellner, M., Over, J., Phillips, D. Defining Manually Enactable Processes Using the Process Definition Information Organizer Templates, Electronic Systems Center, Contract No. F19628-C-029, Task IV02.1, CDRL Sequence A014-010, Defense

Technical Information Center, Cameron Station, Alexandria, VA 22304-6145, March 6, 1995

[Fagan 86] Fagan, M. "Advances in Software Inspections," IEEE Transactions on Software Engineering, Volume SE-12, Number 7, July 1987, pages 744-751

[Feiler 92] Feiler, P., and Humphrey, W. Software Process Development and Enactment: Concepts and Definitions, Software Engineering Institute CMU/SEI-92-TR-04, Pittsburgh, PA, September 1992

[Frakes 96] Frakes, W., Terry, C. Software Reuse Models and Metrics: A Survey (to appear), ACM Computing Surveys, 1996

[Gale 90] Gale, J., Tirso, J., and Burchfield, C. "Implementing the Defect Prevention Process in the MVS Interactive Programming Organization," IBM Systems Journal, Vol. 29, No. 1, 1990, pages 33-43

[Latour 91] Latour, L., Wheeler, T., Frakes, W. Descriptive and Prescriptive Aspects of the 3 C's Model: SETA1 Working Group Summary. Ada Letters, 1991. XI(3): p. 9-17.

[Over 94] Over, J., Kellner, M. Fundamentals of Software Process Definition, Tutorial given at the 1994 SEPG National Meeting in Dallas, TX, Software Engineering Institute, Pittsburgh, PA, 1994

[Paulk 93] Paulk, M., Curtis, B., Chrissis, M., and Weber, C. Capability Maturity Model for Software, Version 1.1, CMU/SEI-93-TR-24, February 1993

[ProcessWeaver 92] ProcessWeaver User's Manual and Reference Manual, Version PW1.2, Grenoble, France, CapGeminin Sogeti, 1992

[Radice 85] Radice, R., Roth, N., O'Hara, Jr., A., Ciarfella, W. "A Programming Process Architecture," IBM Systems Journal, Vol. 24, No. 2, 1985

[SynerVision93] Introduction to SynerVision: Models of Use, Hewlett-Packard document B3261-90002, Draft May 5, 1993

Appendix A - Sample Process Notation

Name	Name the process/sub process that is described within the document.
GENERAL INFORMATION	
Process ID	Unique process identifier
Process Purpose	Provide a brief description of the purpose and objective of the activity.
Standards	Identify the applicable process and product standards, including the SEI CMM KPA reference.
Related Processes	Identify processes that are related to this process, especially if this process is part of a set that is normally viewed as a whole.
Version Number	Configuration management version number used in PAL
CUSTOMER DESCRIPTION	
Customer	Identify the internal and external groups who benefit directly (receive products/services) from the results (outputs) of this process.
Customer Requirements	List each of the legitimate requirements that have been negotiated and agreed to with the identified. These requirements should follow the RUMBA criteria in that they should be Reasonable, Understandable, Measurable, Believable, and Acceptable.
INTERFACE DESCRIPTION	
Entrance Criteria	Identify the criteria that must be satisfied before the activity can be initiated. The criteria might say how to tell when a process can be started, for example at the conclusion of another activity or process.
Inputs	Identify the work products that are used at any point in the process.
Outputs	Identify the work products that are produced during the process.
Exit Criteria	Identify the criteria that must be satisfied before the activity can be considered complete. Exit criteria summarize the salient measurable tasks of the process.
PROCEDURAL DESCRIPTION	
Responsibilities	Describe the groups that participate in the process.

Tasks Describe the tasks that must be accomplished within the process. For ease of reference, the tasks should follow the QIDW process diagram referenced as the main exhibit. If the process is procedural, describe the tasks in the order that they must be accomplished, numbering each task step. Parenthesis the responsible group to the left of the task, as shown below:

(<Participating group>) <Task description>

Use action verbs to describe the tasks. Reference by process ID all tasks that are further described elsewhere. Note any particular procedures, practices, or methods that are employed in any step.

Tools Describe suggested or mandatory tools used during any step of the process.

Resources Describe resources that are necessary to enact the process.

CONTEXT DESCRIPTION

<u>Domain</u>

Applicable Domain List the application domains to which this process is applicable.

Required Domain Knowledge Describe the knowledge of the application domain that participants in this process must exhibit.

Usage Information Describe how past projects have used this process, including the results of the process (i.e., associated metrics) and the lessons learned.

<u>Organization</u>

Organization Size Briefly describe the organization size that limits this procedural method.

Organization Structure Define specific groups or functions that must be in place to execute this procedural method.

<u>Project:</u>

Project Duration Describe the applicability of this process in regards to project length, e.g., any project that requires over 1 person-month.

Problem Complexity Describe complexity constraints. Generally phrased in terms of software size.

Physical Team Locality Define whether the team must be collocated or can be physically distributed.

Communication Infrastructure Describe what information resources and inter group communication paths or mechanisms must exist.

Management

Cost/Benefit Describe the costs and benefits of this process in both the short and long term.

Risks List management and technical risks associated with the execution of this process.

MEASUREMENT DESCRIPTION

Quality Indicators List and briefly describe those measurements that will be used to track the performance (or outcome) of this process in terms of the product or service delivered to the internal or external customer. These indicators should be linked closely to the valid customer requirements and should be used to monitor performance of the entire process. These measures should be measurable, verifiable, and cost effective.

Process Indicators List and briefly describe those measurements that are to be taken at critical points during the process and used to track and assess the effectiveness of the process itself. These in-process measures should also be measurable, verifiable, and cost effective.