A Study in Process Simplification

Ashok Dandekar
Fujitsu Network
Transmission Systems
Richardson TX 75082
a_dandekar@fujitsu-fnc.com

Dewayne E. Perry
Software Production Research
Bell Laboratories
Murray Hill NJ 07974
dep@research.bell-labs.com

Lawrence G. Votta
Software Production Research
Bell Laboratories
Naperville IL 60566
votta@research.bell-labs.com

Abstract

One of the major problems with software development processes is their complexity. Hence, one of the primary motivations in process improvement is the simplification of these complex processes. We report a study to explore various simplification approaches and techniques. We used the available process documentation, questionnaires and interviews, and a set of process visualization tool fragments (`pfv`*) to gain an understanding of the process under examination. We then used three basic analysis techniques to locate candidates for simplification and improvement: value added analysis, time usage analysis, and alternatives analysis. All three approaches proved effective in isolating problem areas for improvement. The proposed simplifications resulted in a savings of 20% in cost, 20% in human effort, 40% in elapse time, and a 30% reduction in the number of activities.*

1. Introduction and Overview

The process system that is the context of our study came into being as a result of a *management edict* (they had heard that processes were a good thing and that they ought to be defined). This ‘‘good thing’’ then grew and evolved in an unmanaged fashion due to the lack of effective management of the process system architecture. This unmanageability was then exacerbated by the merger of two process systems for similar products.

In [5], [6] and [10] we reported our experience in trying to understand the current state of the process system and its architecture. One of the underlying root causes of the current state of both the system and its architecture is the complexity of the processes (both their inherent complexity and the complexity of their visualizations).

In this paper we report a trial study to simplify that complexity in part of the process system. We chose to simplify rather than to re-engineer for several reasons. First, simplification is easier to do than re-engineering and is much smaller in its scope. Second, we wanted to see what could be gained by this approach as an example for the other process subsystems. And third, it was our intuition that there was much to be gained in terms of complexity, cost and interval reduction without going to the lengths of full-scale re-engineering. Our intuitions have proven to be correct.

Our criteria for choosing a particular process subsystem were

- the process needed to be customer focused,
- the process needed to have known problems, and
- the people using the processes needed to be willing to participate in the study.

On the basis of these criteria, we chose the customer documentation processes as the trial study processes and the new documents process as the specific focus. We then spent about 4 months meeting weekly to determine appropriate simplifications and improvements, and concurrently, building the supporting empirical baseline and technology infrastructure.

In the following sections we discuss the scope of the study, our efforts to understand the processes, our analyses, the results, the insights and lessons learned, and the current status.

2. The Scope of the Study

The customer documentation process subsystem represents a substantial subset of the entire process system. From this subset we chose a smaller subset of the core processes to look at more closely (because they represented the production of 90% of the document artifacts) and from that set finally selected the particular process for this study. Since these processes are executed in a number of different locations in both the US and Europe, we narrowed the study to a single location as well: the location where the majority of the people work.

The simplification team consisted of process executors (the people writing the documentation), members of the quality team, and members of research. We initially used a ‘‘local big picture’’ [5] as an aid in scoping out the core processes since it is quite effective in illustrating the production and consumption of document artifacts. We then began the process of understanding and analyzing the new documentation production process.

3. Understanding the Processes

Watts Humphrey [9] distinguished between processes as described, processes as executed, and processes as they ought to be. To determine the last of these, we need to understand the first two. We use different techniques to determine the process as defined and the process as executed.

3.1 Processes as Defined

The entire set of processes governing the production of this product from first customer contact to customer support are defined in electronic form and available as an on-line methodology [13]. The process descriptions are in informal prose in structured documents, each of which includes sections on the input and its suppliers, the output and its customers, tasks, templates, and relevant roles. We used the relevant process descriptions as the primary basis for our understanding of the processes as defined.

We augmented it with what we came to call ‘‘the big picture’’ (see [5]) of the documentation subsystem to visualize its interconnections with the rest of the entire process system and its intraconnections among the various processes that make up this subsystem.

3.2 Processes as Executed

Our initial approach to understanding the executed process and its context was to issue a questionnaire to motivate and organize a two day group meeting with a representative group of documenters from all the various locations and various aspects of the process subsystem. The questionnaire was designed to cover the context of the processes, cost factors, modeling decisions and content, and execution issues.

The context discussions covered the organizational context with its geographical and management considerations, the means of inter-location interactions, and the computing equipment and environments. The discussion about cost factors included issues of personnel and resource costs, costs of the context (computing and communications), process intervals, and process and artifact costs. The process modeling discussions covered issues about roles, process and artifact structure, and process interfaces.

The discussions about the executed process focused on two different issues. The first issue was that of roles and interactions -- what the actual roles are, what their attributes are, and how roles interact in executing the various tasks, subprocesses and processes [4]. The second issue is that of how the executed processes differ from the defined processes — that is, where and why they deviated from the prescribed process.

3.3 Details of the Subject Process

In focusing on the specific documentation process for new documents, we decided that a detailed view of the process and its relevant data was needed. Using the previously developed tool fragments [5] as a base, we built a new suite of tool fragments to provide this detailed visualization.

The flowcharting visualization is supported by the PFV suite of tool fragments: `pfv` takes a set of defined process steps and decision points and generates commands to `dot` [7] to format the visualization.

The format of the visualization specification is in awk record style format with a tab character separating the fields in a one line record. A ‘step’ or a ‘decision’ specification is a sequence of records. Steps and decisions are separated from each other by one or more blank lines.

The steps in the process flow are illustrated in a record format that defines a number of items of interest in the process simplification process: the process step identifier and title, the input and output artifacts with their suppliers and customers, the document reference list, the list of relevant roles, the actual and elapse time to perform the process step, the list of resources needed by the process step, and what the next steps are.

The format for a step specification is:

Step:	StepID	StepTitle
Input:	ArtifactName	SupplierID
...		
Output:	ArtifactName	CustomerID
...		
Ref:	DocRefList	
Roles:	RoleList	
Times:	ActualTime	ElapseTime
Tools:	ToolsList	
Next:	(Step/Decision)ID	
...		

The step in the process (or task) is defined by a StepID and a StepTitle. It is the StepID that is referenced throughout the process flowchart description, not the StepTitle. The StepID may be used as a SupplierID and a CustomerID in the input/output specifications. The inputs and outputs are defined by an ArtifactName and either a SupplierID or a CustomerID. The suppliers and customers are not used for visualization but are there for interface analysis and generation.

The DocRefList, RoleList, and ToolsList lists are lists where each element in the list is separated by the tab character. For Example:

Ref:	FT PD	Task 1	Act 3
Roles:	Info Dev	ECMS Team	
Resources:	Computation	ECMS Help	

There are no restrictions on the number of next steps that may be specified. Where multiple exits occur from a process step, that means that the paths proceed in parallel. Where multiple exits occur from a decision, that means that the paths are alternatives. The destination in either case is a StepID or a DecisionID.

The format for the decision specification is:

Decision:	DecisionID	DecisionQuestion
Exit:	DecisionLabelID	Percentage
...		

Contrary to the StepID, the DecisionID is not printed but only referenced. The DecisionQuestion defines what will be printed in the decision diamond. The DecisionLabel defines an answer to the decision question and labels the line from the decision to the next step or the next decision.

There may be as many exits as are needed to define all the answers to the decision question. There are two ways to label the exits with process data: for loops back into the previous steps and for proportions of time in each of the branches. The latter is represented by specifying percentages. The percentage specified is the percentage of the time the exit is taken compared to all the other possible exits (the sum of all exits is 100). For loops back to previous steps, the number specified is the average number of times the loopback is taken.

The following is a flowchart fragment and its visualization is shown in Figure 1. The initial process flowchart is shown in Figure 2.

Step:	1	Plan FT Strategy	
Input:	FT Information	Project Management	
Output:	FT Strategy	Project Management	
Ref:	FT PD	Task 1	Act 2
Roles:	FT Planner		
Times:	1.0	3.5	
Tools:	ADEPT DB		
Next:	D1		
Decision:	D1	Errors?	
Exit:	Yes	2	35
Exit:	No	3	65

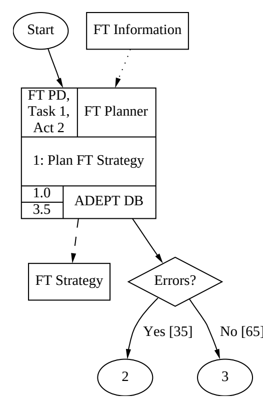

Figure 1: An example fragment description and its visualization.

4. Analyses of the Trial Process

For our simplifications and improvements, we used primarily three different kinds of analyses: value added analysis, time usage analysis, and analysis of alternatives. Moreover, we looked carefully at two obvious candidates for simplification: the looping structures of iterative process steps and extra or duplicated work.

The general goal of this study as a whole is to reduce the complexity of the process. To determine whether we had achieved our simplification goal, we surveyed the process executors and their management.

4.1 Value Added Analysis

One of the first problems faced in trying to simplify a process is that of what to use as criteria for choosing what to keep and what to cut. Harrington's advice [8] seemed to be a good place to start: first identify each activity as having either customer value, business value or no value. Then maximize customer added value, minimize business added value, and eliminate those activities which add no value. Activities that are required to maintain the product are considered to be of business value.[1]

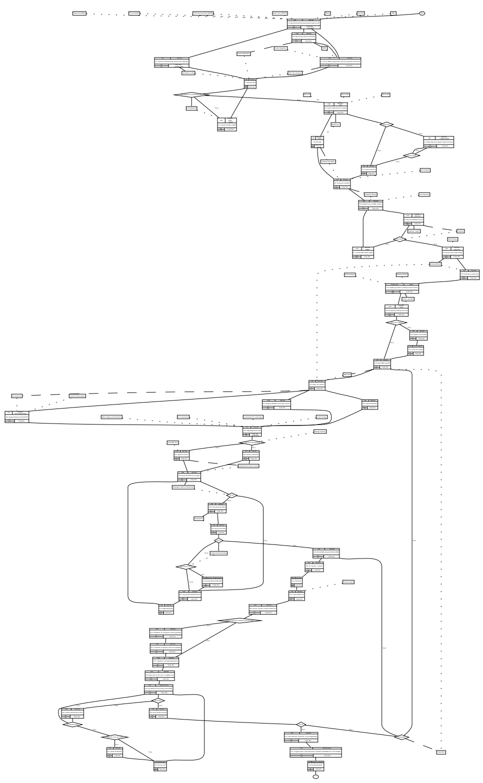

Figure 2: The initial process flowchart. In comparing this flowchart with that in Figure 6, you will notice that there are more iterations and a more complex control flow in the initial process than in the simplified one.

As an aid to this effort we extended `pfv` to include characterizations of any part of the flowchart information and the coloring of those characterizations. We did this with two mappings: one from flowchart elements to characterizations, and one from characterizations to colors. For example, the primary colored flowchart colored customer added value as blue, business value added as magenta, and no value added as yellow. This colorization was extremely effective in highlighting the differences between these activities.

Using this approach we classified 20% of the activities as adding customer value, 30% of the activities as adding business value, and the remaining 50% as adding no value.

Given this characterization of the various activities it became obvious that there was a lot of rework happening in the process. This rework came about because of the emphasis on business value added considerations in the process: the primary artifacts of the documentation process should be under configuration control and that they should be put under configuration control as early as possible. The net result was the large amount of rework embedded in the process.

Once the notion of customer added value was proposed as the most important aspect in the design of the processes, the thinking of the simplification team changed about the value of the business-oriented decisions that had been made. Early configuration control did not add anything to the value of the documents for the customer and, apparently, only caused a large amount of extra work. We used this intuition as the basis for our hypothesis in the paired match study we discuss in the section on analyzing alternatives.

4.2 Time Usage Analysis

One of the primary problems in large-scale software development is the time spent waiting for resources, responses, meetings, etc. One may be able to fill in the intervening time productively, but for a particular sequence of activities there may be a significant difference between the actual time spent and the time that elapses before completion (for example, see the discussions of how time is spent in [2], [11] and [12]).

To this end, we included two things in the data we needed to analyze the process: estimated race[2] and elapse times, and estimated percentages of time spent in various paths in the process (that is the percentage of time taking one path over another out of the decision points).

Using the insights from the value added analysis exercise, we decided to look at the part of the process where there was the most rework: the private, manual

1. It is interesting to note that Harrington considers all reviews, inspections and testing as adding no value to the product. Rather he views these as opportunities for technological improvement of validation.

2. The race time is the time that would be spent if there were no delays of any kind -- that is, it is the actual time spent.

and production builds (figure 3). The writers individually build their respective chapters in the context of the entire manual. Once a chapter has been successfully built, it is put under configuration control, handed off to another group who does the quality control builds on the entire manual. Once the manual has been successfully built, it is then passed to the production team for the production build. This process is the result of the business decision to put the artifacts under configuration control as early as possible. All three builds are part of the process.

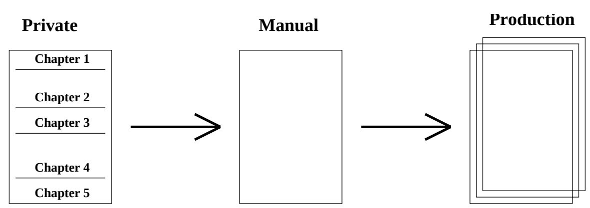

Figure 3: Private, Manual and Production Builds

We did a study to grade how successful these three build processes were. In the private builds of the chapters, a build is said to be *perfect* if it builds cleanly and has all the text laid out properly. A build is *successful* if it builds and returns some output. A build *fails* if it generates no output. 49% of the private builds were perfect, 34% were successful, and 17% failed (figure 4).

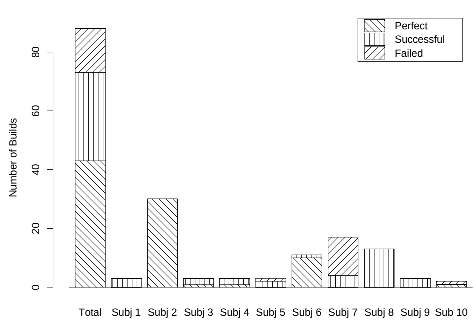

Figure 4: Private Builds

Once the private builds are perfect, the artifacts are handed off to the manual build team. 29% of the manual builds failed completely, 42% successfully resulted in some output with 29% considered to be perfect (figure 5).

Once they succeed, they are then passed on to production. Unfortunately, production builds have the same failure and success rates (one-third each). The reason for this is that the manual builds (as well as the private builds) are done in a non-production context -- the manual builds succeeded but in the wrong contexts.

So nothing but extra work was actually gained from the manual builds.

4.3 Alternatives Analysis

Alternative solutions need to be analyzed to make sure that they will be better than the steps they replace. To support this type of analysis, we added tool fragments to the PFV suite. We added a fragment to create a database of information from the process flowchart descriptions. This enabled us to query our process representation about various steps, artifacts, etc. A complimentary fragment enabled us to select subsets of the flowchart for individual alternative analysis.

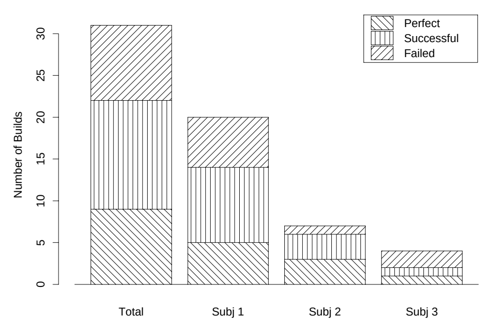

Figure 5: Manual Builds

The primary goal of time usage analysis is to find ways to reduce the production interval. In doing this, we often need to compare alternative process fragments formally to decide between them. The study design we used to meet this goal of deciding between two alternatives is that of a paired match study on a major variable (in this case, the time interval) [1].

Given that we eliminate the manual builds, we have two choices as to where to put the artifacts under configuration control: before the private builds or before the production builds. Our hypothesis was that early configuration control resulted in significantly more work and rework. In the paired match study, we used two people doing very similar features and looked at the difference in the time interval. We confirmed that hypothesis. The trial showed that chapter optimization was very expensive under configuration control -- a factor of two more expensive.

5. Process Improvements

On the basis of our various analyses, the simplification team thoroughly scrutinized the current process as it is executed and proposed a list of improvements. The accumulated effect of these simplifications and improvements amount to a significant savings in terms of cost (20%) and both race (20%) and elapsed (40%) intervals. Moreover, the number of activities was reduced by 30.

While the results here described represent the improvements to only one process in the process subsystem, this process is representative of the other development and evolution processes in this subsystem for this class of artifacts.

The improvements may be characterized as follows:

- improved feature impact assessment
- local graphical assistance
- local editorial staff
- simplified review activities
- elimination of manual builds

The resulting process flowchart illustrates the effect of the simplification effort (see Figure 6).

5.1 Improved Feature Impact Assessment

The process requires that writers refer to various documentation and information repositories to assess the impact of a certain feature on their manual.

The elapse time required to complete this assessment far exceeds the actual time spent because the information sources are incomplete, inaccurate and late. These problems result in wasted effort and time spent waiting for correct information. The state of the information sources results in part because developers do not know enough about documentation requirements.

There are several changes to the process to eliminate unprofitable effort and time delays: have the writers participate in the requirements and design reviews; have a writer be a member of the requirements specification and software development team; provide training for software developers on documentation; and develop a requirements assistant tool that will aid feature description developers to indicate documentation impact.

5.2 Local Graphics Assistance

When graphics help is needed, it is obtained from graphics department in a remote location. The activities performed are: send requirements to the remote graphics department, they prepare the graphics and return them, the senders review it and send comments back to the graphics department for correction, they make the corrections and return the graphics to the sender.

Obviously, the problem is the number of iterations that take place every time long distance help is sought. The process must be repeated a number of times until the picture is finally correct. This lengthy iterative process is compounded by the fact that a significant amount of help is needed in the use of `xcip` (the graphics tool used).

The proposal is to do graphics in-house. To make this happen, the following requirements were proposed: train all writers in basic graphics packages and make local expertise available for complex graphics help so the writers do not have to seek help from the remote graphics department.

5.3 Local Editorial Staff

Every document is sent to an editor in another location to ensure compliance with style standards, the consistency and accuracy of acronyms, and conformance to international standards. The editor's comments are then incorporated in the text.

The issues with the current process are: turnaround time is long, subjective opinion changes constantly, and meaning changes with the context.

The proposal is to eliminate the separate task of having an editor edit the document. However, in order to maintain the quality (compliance with standards, consistency/accuracy, etc.) the editor is included as a reviewer so that editing is done as part of the review process thus eliminating editing as a separate, sequential task. Also, to keep editing to a minimum, the following is made available to information developers: a single style guide, overview training, a style guide checklist, and editing training for all writers.

5.4 Elimination of Manual Builds

Presently, the information developers with feature impact do a private build followed by a manual build. Private builds ensure a perfect build of the part being written. It is done by individual writers. Manual builds are performed by the manual owners to ensure a perfect build of the whole manual. Finally, a production build is performed (by production builders) to produce the

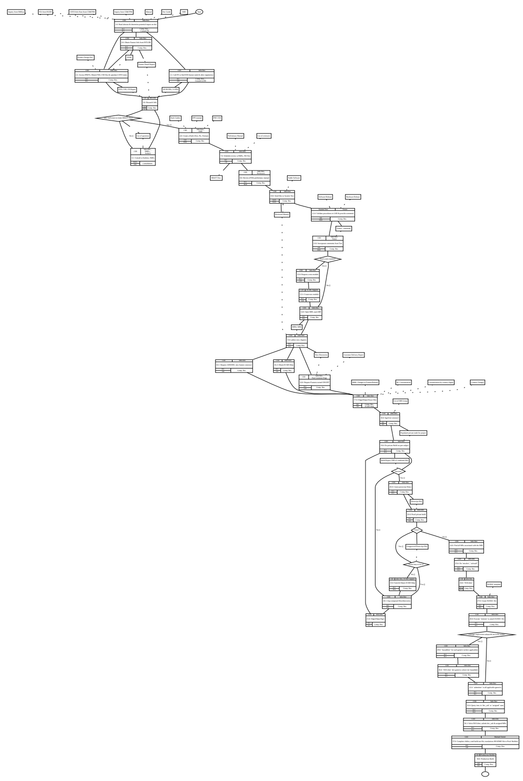

Figure 6: The resulting simplified process. Note how much simpler the control flow is and that there are fewer iterations.

deliverable documentation.

The issue is straight-forward: there are too many builds. A private build is unavoidable because this is where the basic unit of the manual is built and is where changes are verified. At present the production build cannot be eliminated. The candidate for elimination is the manual test build. The data shows that the main categories for failing manual test builds are modification request (MR) problems, featuring problems, non-process problems (e.g., file system out of space), ‘‘manual.vol’’ file problems, and various other problems.

After discussion, the team felt that the other category errors can be attributed to either MR or featuring problems. If we ignore the non-process problems, the featuring and MR problems account for 82% of the problems. To ensure that these problems are identified and fixed in private builds, writers must perform an `sget` with the MR number in the private build. This will help detect MR dependencies. The writers also need to plan ‘‘feature turn on/off’’ at least 2 weeks before delivery. Both of these items were added to a checklist.

5.5 Simplified Review Activities

The document review activities cause a significant amount of elapse time in this process. The process as documented in the on-line methodology under the Document Review Process involves selecting a group of reviewers, sending them the document to be reviewed, holding a review session, documenting issues raised, resolving the issues, and updating the document.

Again, the issue is rather straight-forward: the process is too cumbersome and takes too much time.

To solve this problem, we use the Fast Decision Process (FDP) instead of the current review process. In the FDP process, a group of reviewers meet in a room with terminals and the issues raised are resolved on the spot. Data shows that the saving in time will be at least 50%. Because of this automated support, FDP can also be used effectively to include reviewers from across different locations. FDP is documented as a procedure under the Document Review Process.

6. Lessons Learned

On the basis of our successful trial, we offer the following as insights that we either gained or confirmed in the course of this study.

- Finding the right level for process descriptions is difficult. Too much detail often results in overly complex descriptions; too little results in unusable descriptions. Focusing on the intent and the goals rather than on the details helps reduce some of this complexity.

- Effective process simplification requires a deep understanding of the processes and their relevant domains. We chose those executing the processes as the suppliers of this knowledge rather than those responsible for the process descriptions. This strategy worked effectively.

- Iterations in processes that span multiple locations can be very expensive in both actual time and elapse time costs. There are time lags, even when electronic means are used (especially when the multiple locations span different time zones or continents). Moreover, inter-location communication is still relatively primitive when high-bandwidth interactions are needed.

- Harrington’s advice of emphasizing customer added value, minimizing business added value and removing no added value was instrumental in generating several key insights into the fundamental nature of the process and why so much rework was being done.

- Even well established practices need to be reconsidered. There are always cost trade-offs that need to be revalidated. In this case, the cost of configuration control early in the process outweighed its benefits and rethinking its use as a business value added mechanism resulted in a reduction of effort without a significant cost in quality.
- Estimates and measurements of resource usage, and time and effort cost studies are necessary preconditions to any simplification or improvement effort (see [11] and [12] for a more complete discussion of these issues).
- No single simplification produced a major result, rather the accumulation of modest simplifications in the right places resulted in a significant reduction in time and cost. These modest improvements resulted in removing some of the accidental debris and enabled the process executors to focus on the essential problems of artifact production [3].
- To gain the maximum benefits from simplification and improvement efforts, the focus should be at the process system level where global improvements will have a more far reaching effect than they do at the individual process level where improvements have only local effects.
- Study designs of varying costs and complexity are available for studying alternative process fragments. There is often a direct correlation between cost/complexity and the degree of certainty, and a series of studies is often needed to arrive at a full understanding of the alternatives. Sometimes, however, an inexpensive study will yield such dramatic results that no further studies are needed.

7. Current Status

This trial has resulted in a set of process understanding and analysis techniques and tools that were found to be effective in simplifying and improving software-related processes (or any type of processes for that matter). Because of our success, this approach is being applied to about a half-dozen processes governing other aspects of the software production process.

The early results for these efforts look very promising. For example, in one case (a review process), the current process documentation has been reduced by 90% and the number of activities by 75%.

Acknowledgements

This work would not have been possible without the unflagging support of Al Barshefsky and Mike Poepping, and the active contributions of the other members of the simplification team: Boyce Grier, Mark Hamilton, Barbara Holmes, Angela Preston, Greg Ryba, Rich Schmidt, Anne Marie Stelter, Sherry Stratton, and Cecil Thomas.

References

[1] Thomas J. Allen. *Managing the Flow of Technology*. Cambridge MA: MIT Press, 1977.

[2] Mark G. Bradac, Dewayne E. Perry and Lawrence G. Votta. ‘‘Prototyping a Process Monitoring Experiment’’, *IEEE Transactions on Software Engineering*, 20:10 (October 1994), pp 774-784.

[3] Frederick P. Brooks, Jr. ‘‘No Silver Bullet: Essence and Accidents of Software Engineering’’, *Computer*, 20:4 (April 1987), pages 10-20.

[4] B. G. Cain and J. O. Coplien. ‘‘A Role-Based Empirical Process Modeling Environment’’, *Second International Conference on the Software Process: Continuous Software Process Improvement*, Berlin, Germany, February 1993, pp 125-133.

[5] David C. Carr, Ashok Dandekar, and Dewayne E. Perry. ‘‘Experiments in Process Interface Descriptions, Visualizations and Analysis’’, *Software Process Technology — 4th European Workshop, EWSPT'95* Wilhlem Schaefer, ed. Lecture Notes in Computer Science, 913, Springer-Verlag, 1995. pp 119 - 137.

[6] Ashok Dandekar and Dewayne E. Perry. ‘‘Barriers to Effective Process Architecture’’, submitted for publication.

[7] E. R. Gansner, E. Koutsofios, S. C. North, and K. P. Vo. ‘‘A Technique for Drawing Directed Graphs’’, *IEEE-TSE* 19:3 (March 1993)

[8] H. James Harrington. *Business Process Improvement*. New York: McGraw Hill, 1991.

[9] Watts Humphrey. *Managing the Software Process*. Reading Mass: Addison-Wesley, 1989 (reprinted 1990).

[10] Dewayne E. Perry. ‘‘Issues in Process Architecture’’, *Proceedings of the 9th*

International Software Process Workshop — The Role of Humans in the Process, October 1994, Airlie VA. IEEE Computer Society Press. pp 138 - 140.

[11] Dewayne E. Perry, Nancy Staudenmayer and Lawrence G. Votta. ‘‘People, Organizations, and Process Improvement’’, *IEEE Software*, July 1994, pp 36-45.

[12] Dewayne E. Perry, Nancy Staudenmayer and Lawrence G. Votta. ‘‘Understanding and Improving Time Usage in Software Development’’, in *Trends in Software: Software Process*, Wolf and Fuggetta, editors, John Wiley & Sons, 1996.

[13] Lawrence G Votta. ‘‘Comparing One Formal to One Informal Process Description’’, *Proceedings of the Eighth International Software Process Workshop*, Wadern, Germany, March 1993, pp 145-147.