Studies in Process Simplification

Ashok Dandekar
Fujitsu Network
Communications, Inc.
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 set of studies 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’’ began with various people and groups defining the process they were responsible for more or less in isolation from all the other processes being defined. These processes then grew and evolved in an unmanaged fashion, due primarily 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 [6], [7] and [11] we reported our experience in trying to understand the current state of this 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). There are two factors that contribute to system complexity: the exceedingly large number of interactions among the processes and the exceedingly large number and wide variety of artifacts exchanged among these processes. Factors analogous to these contribute to the complexity of individual processes: the number of interactions among process elements, the number and variety of artifacts exchanged among the process elements, and the contours of the various paths through the process.

In this paper we report a set of studies in which our goal is to simplify a specific process — that is, to understand and resolve the problems of internal complexity. We chose simplification rather than re-

engineering as our approach 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 within that set of processes 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. During this period we used three different experimental techniques in our iterative approach to gaining insight and understanding.

In the following sections we introduce our philosophical approach to process improvement, delineate the scope of the study, discuss our efforts to understand the processes, summarize our studies and their results, and distill the insights and lessons we learned.

2. The Approach We Found Useful

We used a basic straightforward process improvement paradigm in which we iteratively build our understanding of the process and its context, cost, quality and interval characteristics, the supporting technology, and the various alternative improvements.

We classify the experimental techniques that we use to build this understanding into two classes: retrospective and prospective. Retrospective techniques use various existing legacy documents and data, as well as various forms of personal and group experience reports. Prospective techniques use various forms of proactive data gathering and experiments to gain detailed understanding of cost, quality and interval factors, as well as various process control factors (that is, causal connections among process elements).

One common technique for studying processes is GQM (Goal, Question, Metric) [2]. This is primarily a formalization of the empirical paradigm tailored to understanding software products and processes. As such it is primarily prospective and not retrospective. Since understanding the process before improving it is fundamental [12] and since retrospective studies can supply a large degree of understanding at very low experimental cost, the GQM approach did not seem appropriate for the problems we addressed.

Thus, in our iterations over various aspects of the studied process we use a mixture of retrospective and prospective studies.

3. 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.

There are several advantages that the subject process has for our studies. First, it has a relatively short interval from start to finish. In the context of large-scale software developments that last for months and in some cases several years, an interval of three months is very useful for both retrospective and prospective studies. The advantages for retrospective studies are that the details are still fresh in the developers' memory and there are fewer problems with execution reconstruction. The advantages for prospective studies are that the studies can be done relatively quickly and generally without high costs or commitments for the developers. Second, the process is executed frequently. This is particularly useful for prospective studies since studies are best done iteratively, each successive study building on the previous studies. There is a significant advantage to being able to vary designs and instrumentations for the multiple studies. For retrospective studies, the primary advantage is the accumulation of experience and the likelihood of more generalizable results.

The simplification team consisted of process executors (the people writing the documentation), members of the documentation process team, members of the quality team (that is, the process management team), and members of research. The team composition varied depending on the simplification phase. In all cases, members of the quality team and members of research were the core members of the teams.

In phase I (the data collection phase) all the interviewees were executors of the process. The goal of this phase was to collect realistic data about process elements such as activity descriptions, sequences of actual and elapsed times to execute an activity, sequences in which they are executed, roles, tools used on the process, etc.

The rationale for using the executors is that they have the expert knowledge on day-to-day operation of that process, especially since most had been executing this process for quite some time.

The team composition for phase II (the analysis phase) had different needs since the goal was to critically examine the process and consider improvements. Here we used the following mix of people: process executors from the first phase, engineers with quality and process management expertise, and a few novice engineers who had absolutely no knowledge of the process.

The rationale for using this mix is that the novices can objectively question every step of the process, the executors can explain the purpose and details of each step, and the quality experts can compare the two viewpoints and suggest an improvement which can be debated objectively by all before accepting it as part of the improved process.

To ''prime the pump'' for the simplification, we used a local version of the ''big picture'' [6] 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.

4. Understanding the Processes

Fundamental to improving any process is a sufficient understanding of that process and the various alternative improvements to be able to choose wisely amongst those alternatives and achieve the desired improvement. Perry, Staudenmayer and Votta [12] argue this point and illustrate it with a series of studies aimed at understanding how people spend their time in a specific software development process. We follow that strategy here first with retrospective studies followed by a set of prospective studies.

Watts Humphrey [10] 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.

4.1 Processes as Defined

The entire set of processes governing the production of this product from first customer contact to post-delivery customer support are defined in electronic form and available as an on-line methodology [14]. 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.

There are a number of problems with the defined processes. First, they generally are processes defined in isolation, defining what they need and what they supply independent of what the supplier and customer processes define (which are also generally done in isolation). Second, these problems were not ameliorated when the process systems were merged for two large projects that made similar products. In some cases, the processes were truly merged, in others merely mangled. In many cases, it is not at all clear how well the process definitions predict actual process execution.

We augmented it with what we came to call ‘‘the big picture’’ (see [6]) 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.

4.2 Processes as Executed

Our initial approach to understanding the executed process and its context was to issue a questionnaire (see Appendix A) 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 [5]. 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.

The questionnaire worked well as a focus for the discussion and provided useful guidance. Where we could not get information directly from those in the meeting, people volunteered to find the answers and relay them to us.

4.3 Processes as They Ought To Be

There are two distinct forces that determine what the process ‘‘ought to be’’. First, there are the company corporate and marketing strategies that determine what the product composition is and what the cost, quality and interval attributes are. The former implies what the basic activities for the process should be and the latter implies certain choices about the way these activities are structured and interrelated in the process.

Second, there are less tangible, but none-the-less important matters of style that affect what the processes ought to be. As in programs themselves there are differences in styles, so in processes there are differences in styles. These differences tend to become part of the basic culture supporting the processes. Sometimes the underlying culturally supported style is at odds with the desired strategic attributes and must be changed. Obviously, these are much more subtle aspects in determining the structure and content of processes.

The first-mentioned force is, of course, paramount in our studies here. The two primary attributes that need to be changed are those of cost and interval. The quality of the product is generally considered to be satisfactory, but the costs are too high and it takes too long to produce the documents.

5. Visual Support for the Studies

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 to supplement the knowledge we had gathered at the initial two day meeting. Rather than rely on just the written process definitions, we focused on the process as executed as the basis for creating the needed process information to use as the basis for process improvement.

Our previous efforts at understanding the process system architecture and the interactions among those processes were successful largely because we had various forms of visualizations to help us understand the system in different ways. We capitalized on this work and, using the previously developed tool fragments [6] as a base, we built a new suite of tool fragments to provide a customized and detailed form of visualization.

The flowcharting visualization is supported by the PFV (Process Flowchart Visualization) suite of tool fragments: `pfv` takes a set of defined process steps (together with their inputs and outputs and flow directions), and decision points (and their flow directions) to generate commands to a directed

graph drawing program (`dot` [8]) to format the visualization.

The format of the process step and decision 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 for the process step,
- the input artifacts and name for the supplier (either a process step or an external process or subprocess),
- the output artifact and name for the customer (again, either a process step or an external process or subprocess),
- a list of the relevant documents for this step,
- 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 or decisions are.

The initial choice of elements for the process step was based on our previous experience in analyzing and improving processes.

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 StepTitle is used to make the process step more understandable for the readers. 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	

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. Thus, decisions should be made explicitly outside process steps. 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.

Figure 1 contains a flowchart fragment showing each of the basic flowchart elements and pictures the fragments visualization. The initial process flowchart is shown in Figure 2.

As a further aid to visual analysis, we provided a means of coloring the process flowchart. We did this by means of two tables: a component characterization table and a characterization coloring table. The first table defines a set of characterizations of components in the process descriptions. For example. the primary characterization we used was that of characterizing the process steps as to whether they added customer, business or no value. We experimented with others as well, but that characterization proved to be the most useful. One could just as easily characterize the artifacts that are produced according to their complexity or according to whether they are used locally or externally.

The format for the component characterization table is again in an `awk` record format where the components are listed in alphabetical order.

component	characterization

The characteristics used in the characterization table are then given their corresponding colors in the coloring table. How one colors those characterizations is entirely open and limited only by the set of postscript-supported colors and the colors provided by the plotter. We chose in our primary example to make ‘customer value’ as strong a color as possible, ‘business value’ less intense, and ‘no value’ to be fairly pale (cyan, magenta, and yellow).

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

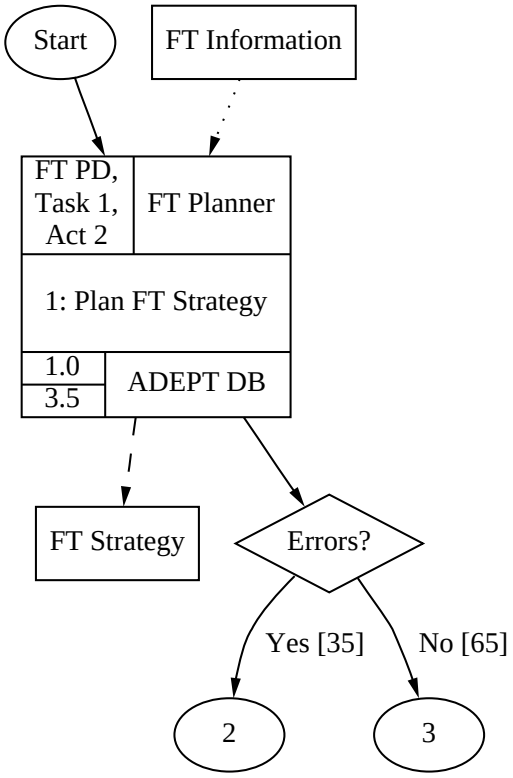

Figure 1: An example fragment description and its visualization.

The format for the characteristic coloring table is identical to that of the component characterization table and also must be in alphabetical order.

characterization	color

6. Prospective Studies of the Trial Process

For our simplifications and improvements, we used primarily three different kinds of studies: value added studies, time usage studies, and assessing alternatives studies. 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.

6.1 Value Added Studies

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 [9] 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.*

The rationale for this is that the primary reason for the process is to produce an artifact that is meant for customer use. The parts of the process that provide things that are of use and interest to the customer are of paramount importance. Thus, they should be maximized. Improvements that increase customer added value are to be considered first.

Next in importance are those activities that are needed to produce, maintain and evolve the product. In most cases, they are activities that are of no interest to the customer, although customers who are critically dependent on the artifacts may well be concerned about these activities to convince themselves that appropriate measures are being taken to ensure the product will be properly maintained and supported. These activities are usually centered around such activities as configuration management, maintainability studies, process management, etc.

An obvious class of activities which add no value are those which are redundant, often existing because of historical reasons. Another class of such activities are those which we often consider to be quality checking activities. 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. One might also view them as opportunities for finding ways of building the quality into the product rather than checking for it afterwards. If we cannot eliminate them, at least we can ensure that they are as effective and appropriate as possible.

Of course, in considering which activities to maximize, minimize and eliminate, one has to consider questions of cost both to the customer and to the business. Expensive features that the customer is unwilling to pay for are of no value and merely add expense. Minimizing business value beyond what is needed for effective and manageable evolution of the product is detrimental rather than beneficial. Clearly, one aims to achieve a balance of various inter-related costs and benefits.

We used the characterization and coloring facilities of `pfv` to help with this analysis. We first characterized the various activities as adding customer, business or no value and then colored these

activities. We colored customer added value as blue, business value added as magenta, and no value added as yellow. This colorization proved 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 were numerous examples of no value added activities and that there was a lot of extra work and rework happening in the process. This extra work and 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 they should be put under configuration control as early as possible. The net result was the large amount of extra work and rework embedded in the process.

This emphasis on customer added value was critical in changing the way the process was thought about. It was instrumental in identifying a number of critical changes to the processes and was a key element in the success of the simplification effort.

6.2 Time Usage Studies

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 [3], [12] and [13]).

To this end, we included two things in the data we needed to analyze the process: estimated race (that is, the time that would be spent if there were no delays of any kind, the actual time spent) and elapse (that is, the calendar time from beginning to completion) 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).

Where value added studies expose what is important to be done and what is not, time usage studies expose how well various parts of the process are being done — that is, how efficiently the different activities are executed. There are two primary strategies in time usage studies: determine whether the race time can be reduced and determine whether the elapse time can be reduced. The first reduces the actual cost of the activities in terms of labor and the second reduces both the time to market costs and the labor costs (especially if work cannot be done while blocked and waiting).

As we show in the proposed process improvements, quite a lot can be done to reduce the elapse time even if you cannot reduce the actual work time. Prime candidates for elapse time reduction are interactions of various kinds, especially if they occur across geographical and temporal boundaries.

Eliminating iterations is a way of reducing both the actual and the elapse times in process activities. Just as with program, you need to look at the parts of the process where significant time is spent to achieve useful results. Optimizing an activity or part of the process that gets done infrequently does not gain much in the way of global improvement. You may want to do it for a certain feeling of elegance in the process, but not for a serious expectation of overall cost reductions or overall process

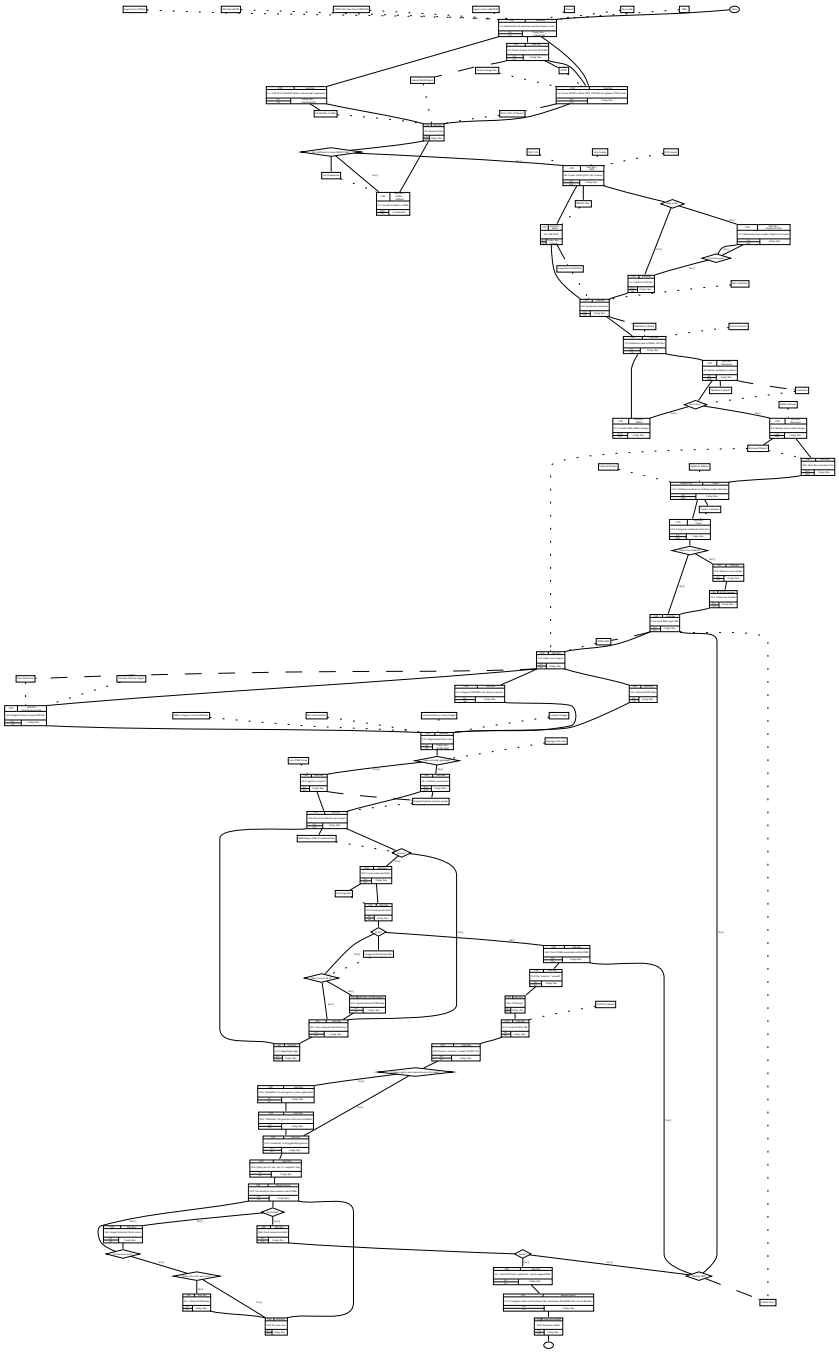

Figure 2: The initial process flowchart. In comparing this flowchart with that in Figure 6, you will notice that there are more iterations and a more complex control flow here than in the simplified one there.

improvement.

6.3 Analysis of Alternatives Studies

Alternative solutions need to be analyzed to make sure that they will be better than the steps they replace. We have argued above that the basis for effective improvement is a substantial understanding of the process as it exists. Only then can you make effective improvements and know with some assurance that the improvements will be as effective as you think they will be.

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

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 complementary fragment enabled us to select subsets of the flowchart for individual alternative analysis.

This feature enabled us to select existing fragments of the current process and compare them in various ways with alternative process fragments.

7. An Example Set of Studies

To illustrate these three kinds of studies and how we used them to simplify the process under consideration, we focus on one of the proposed improvements — that of eliminating manual builds in the document production 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.

One of the key business value added activities is that of configuration management. It supplies the necessary control over the artifacts needed to maintain and evolve the documentation for the customers. As such, there is no question that it is a necessary business value activity. What was noticeable from the value added analysis was that it generated a significant amount of extra work and activity.

While it was agreed that configuration management was essential, it was not clear that the use of configuration management early in the process resulted in any extra value for the customer over using it later in the process. What was clear was that it caused a large amount of extra work. We used this intuition as the basis for our hypothesis in a paired match study to analyze alternative improvements.

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, quality-control 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 by the quality control group, 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.

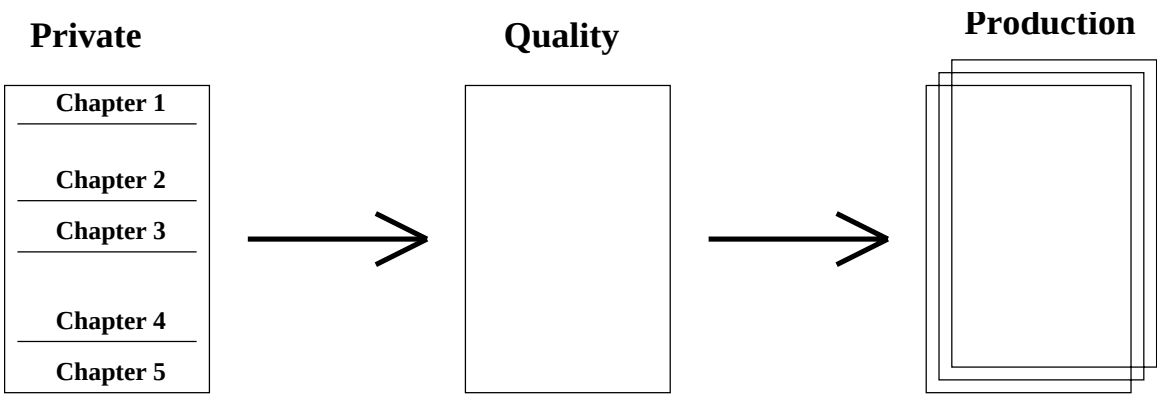

Figure 3: Private, Quality-Control and Production Builds

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).

Once the private builds are perfect, the artifacts are handed off to the quality-control build team. 29% of the these 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 approximately the same failure and success rates (one-third each) as the quality-control builds. The reason for this is that these latter builds (as well as the private builds) are done in a non-production context -- the quality-control builds succeeded but in the wrong contexts.

So nothing but extra work was actually gained from the quality-control builds.

Given that we eliminate these quality-control builds, we have two choices as to when 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. The results confirmed our hypothesis. The trial showed that chapter optimization was very expensive under configuration control -- a factor of two more expensive.

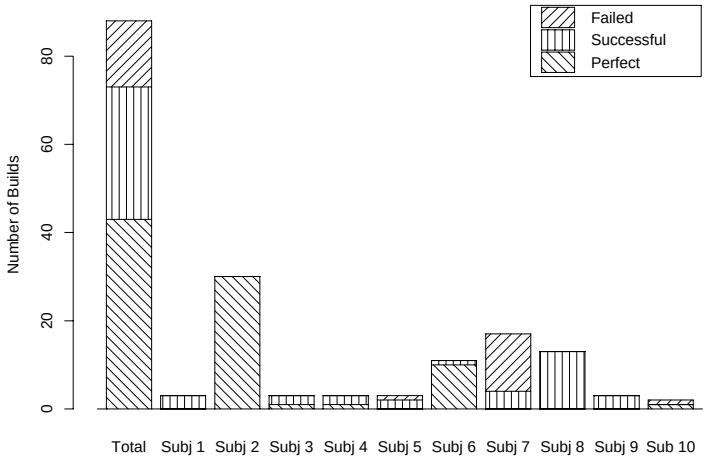

Figure 4: Private Builds

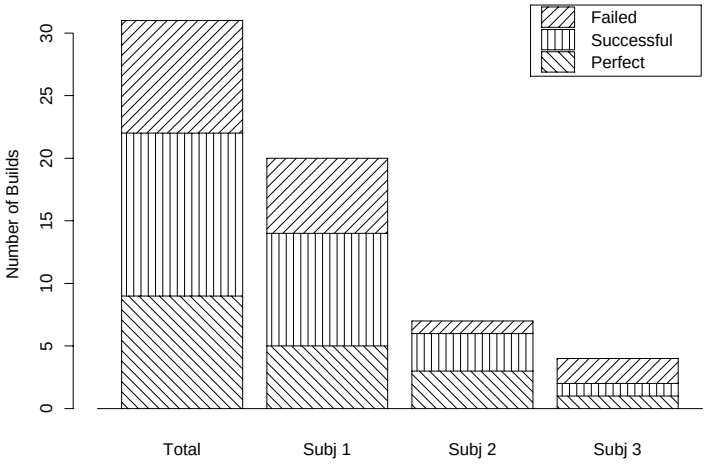

Figure 5: Quality-Control Builds

Thus, the suggested improvement is to first remove the quality-control builds entirely and second delaying configuration management of the chapters until it is time to do the production build. The first reduces both the race time and the elapse time by removing a significant amount of work. The second also improves both the race time and the elapse time by making the private builds significantly easier and thus less time consuming and less delayed.

8. 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 described here 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 quality-control builds

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

8.1 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.

8.2 Local Graphics Assistance

When graphics help is needed, it is obtained from a 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.

8.3 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 document writers: a single style guide, overview training, a style guide checklist, and editing training for all writers.

8.4 Elimination of Quality-control Builds

Presently, the information developers with feature impact do a private build followed by a quality-control build. Private builds ensure a perfect build of the part being written. It is done by individual writers. Quality-control 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 deliverable documentation.

The issue is straightforward: 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 quality-control build. The data shows that the main categories for failing quality-control 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.

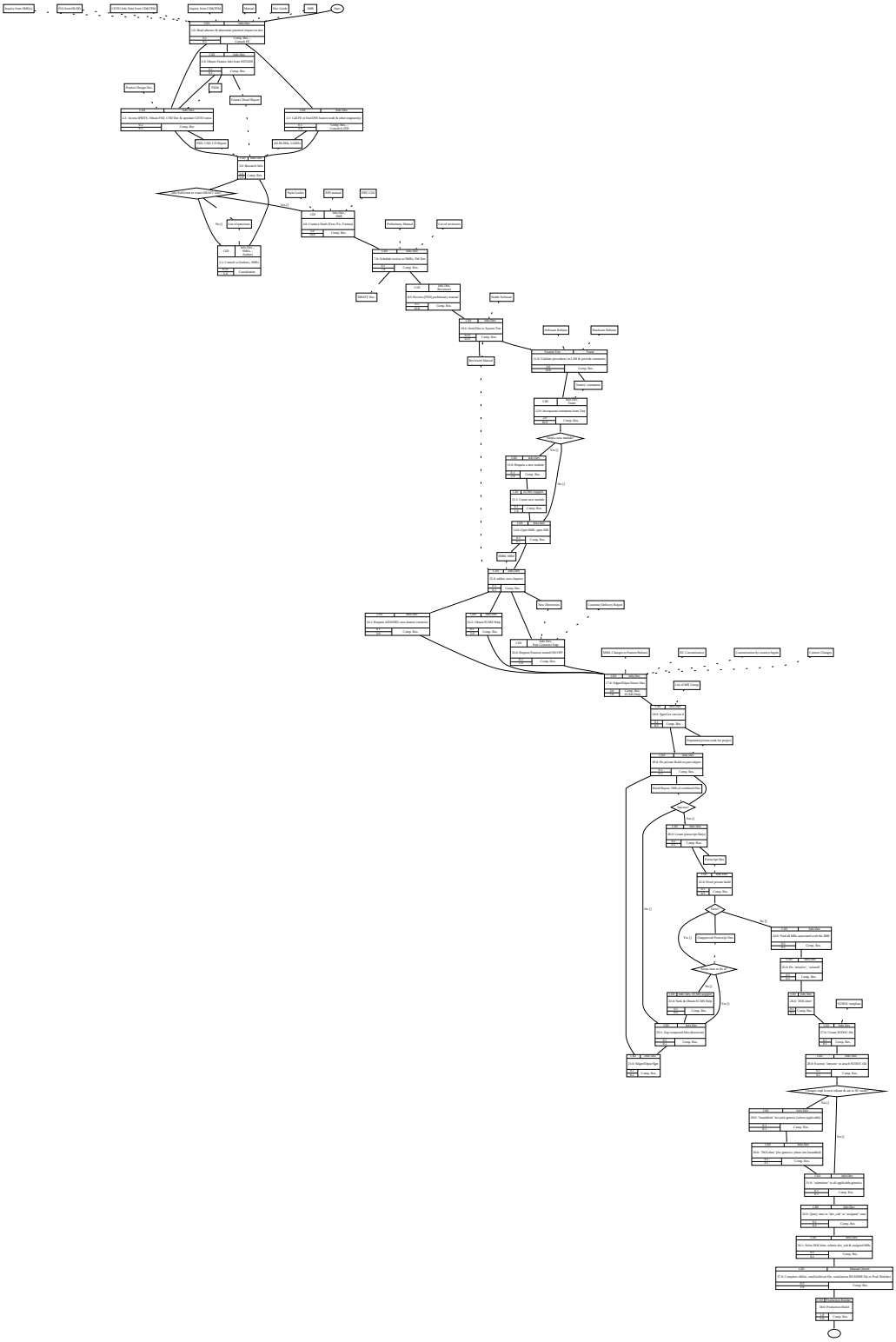

Figure 6: The resulting simplified process. Note how much simpler the control flow is and that there are fewer iterations.

8.5 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 straightforward: 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.

9. 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. However, the right level and a uniform description of the activities in the process at this level are critical as means of determining how much work is involved in the different parts of the process. High-level descriptions for a particular involved part of the process and detailed descriptions of a relatively small part confuse the issue of how much work is actually done where.
- 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. This is an instance of a more general rule: use the appropriate people for the right job at the right time.
- 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. First, iteration is always a prime candidate for scrutiny, for just as in software systems, significant optimization is often found where the most execution time is spent. It is often the case that the inner loops are just those kinds of places.
- 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. As software developers, we often lose sight of the fact that the work is ultimately done so that the product can be used and that those users are the ultimate arbiters of what is important about both the product and the process to produce it. Business value is extremely important, but the use and timing of these activities is critical.

- Even well established practices need to be reconsidered. There are always cost trade-offs that need to be revalidated. The rationale for the original decision may have changed. The marketplace almost certainly changes faster than we can respond to it. Rationales, then, must be constantly revaluated and the parts of the process dependent on those decisions must be re-evaluated. 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 [12] and [13] for a more complete discussion of these issues). Only by knowing where the work is being done and how long it takes can we know where to look for significant reductions and improvements.
- 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 [4]. Significant improvements, if they are found, often are the result of automation specifically tailored to eliminate repetitive, time consuming work done by people. Of course, if the process is really bad, then significant improvements are not only possible but mandatory.
- 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. Local effects, such as we have gained here are always useful, but some care must be taken to ensure that one process does not gain at the expense of another. Only by looking at the process system from a global view does one get the proper perspective of the entire system. We note in passing that a theory for global process improvement is an important research topic with as yet very few results. We have nothing to compare with the well-known results of global optimization of production lines.
- 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. Most importantly, do not ignore existing data that is often available for retrospective study techniques.

10. Summary

We have described an improvement approach using both retrospective and prospective studies that we have found to be useful and effective. We began by using several retrospective studies to gain an understanding of the process and then used a sequence of three prospective studies to isolate and deepen our understanding, and compare alternative improvements.

Because of the success of these studies, this approach has been effectively applied to a half-dozen or so processes and several process subsystems 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%.

While there are a variety of approaches that one might take to process simplification, ours has proved to be one which provided us with a rich set of lessons learned and useful process simplifications and improvements.

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] Victor Basili and David Weiss. ‘‘A methodology for collecting valid Software Engineering Data’’, *IEEE Transactions on Software Engineering*, 10:6 (November 1984), pp 728-738.

[3] 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.

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

[5] 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.

[6] 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* Wilhelm Schaefer, ed. Lecture Notes in Computer Science, 913, Springer-Verlag, 1995. pp 119 - 137.

[7] Ashok Dandekar and Dewayne E. Perry. ‘‘Barriers to Effective Process Architecture’’, *Software Process: Improvement & Practice* 2:1 (March 1996). pp 13-20.

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

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

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

[11] 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.

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

[13] 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.

[14] 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.

Appendix A: The Questionnaire

Process Information

Introduction

The following sections outline a general set of information that we think is important as a basis for process understanding and improvement. There are four major areas that we wish to cover:

- the context of the process,
- the various costs factors in executing the process,
- the process as it is modeled, and
- the process as it is actually executed.

Under each section we pose a number of questions to elicit an understanding of the current modeled and actual process together with its various costs.

Process Context

- *The Organizational Structure*
  - — What is the geographical and management organization within which the process takes place?
  - — To what extent do these geographical and management boundaries help or hinder the execution of the process?
  - — How are the various processes or subprocesses aligned with respect to organizational or geographical boundaries?
- *Interaction Mechanisms*
  - — What are the primary means of communication among the people doing the various processes?
  - — How effective are they? What are their strengths and weaknesses?
- *Computing Equipment*
  - — What sort of equipment is used to support the process?
  - — Is it centralized or distributed?
  - — Do you use PCs, workstations, or terminals? What kinds?
  - — Are they time and cost effective?
- *Computing Environment*
  - — What is the basic supporting operating system?
  - — What are the tools that you use in executing the process?
  - — How much of the time do you use each tool?
  - — How effective are they in supporting your process?
  - — Are there better tools that you might use?

Process Costs

- *Personnel and Resource Information*
  - How many people are involved in the various processes, subprocesses and tasks?
  - What are the resources that are needed for the effective execution of the process?
  - What effect does a shortfall of resources or people have on the process?
- *Contextual Costs*
  - What are the costs of the interaction mechanisms?
  - What are the costs of the computing equipment?
  - What are the costs of the computing environment?
- *Process, Subprocess and Task Intervals*
  - What are the intervals for the various processes, subprocesses and tasks?
  - What is the ratio of elapse time to race time? That is, what is the factor needed to compute the scheduled finish time from an estimate of effort?
  - Where are the time sinks? For example, meetings, waiting for resources, etc.
  - What is the ratio of work to rework?
- *Process and Artifact Costs*
  - What are the various process costs in terms of personnel, resources, salaries, overhead, etc.?
  - What are the various artifact costs in terms of personnel, resources, salaries, overhead, etc.?
- *Possible Cost Improvements*
  - Where are the most effective process improvements likely to be found from the standpoint of costs? From the standpoint of interval?
  - Where are the most effective improvements likely to be found in reducing the costs and intervals to produce the various artifacts?

The Process as Modeled

- *Defined Roles*
  - What are the identified roles?
  - What are the levels of experience, education, etc required for these roles?
  - Do people enact multiple roles concurrently?
  - What are the relationships between roles and processes, subprocesses and tasks?
- *Artifacts and their Structure*
  - How are the artifacts structured? For example, are there templates for the artifacts, formal definitions, etc?
  - Are they supported by particular tools?
  - What are the important events in the lives of the various artifacts? That is, are there particular states that are important in measuring progress or in determining relationships with other artifacts?
  - Can various parts of particular artifacts be worked on concurrently? Or are they too

interdependent or sequential, etc.?
  - Are there well-defined criteria for progress and completion of artifacts? What are they?

- *Processes and their Structure*
  - What are the criteria for dividing a process into subprocesses?
  - What are the criteria for defining tasks?
  - How much of the process structure is strictly sequential?
  - How much concurrency is there among the various processes, subprocesses and tasks? Where?
  - How much iteration is there within the various processes and subprocesses? Among the processes and subprocesses?

- *Process Interfaces*
  - What are the entry and exit criteria for each process?
  - What are the various permissions required for the execution of a process?
  - What are the various obligations entailed by executing a particular process?

- *Process Interrelationships*
  - Which processes, subprocesses and tasks are strictly independent of each other?
  - Which processes or subprocesses have a more or less subroutine like use -- that is, they are used from within another process?
  - Which processes are artifact dependent on each other -- that is, which processes are dependent on the final output of other processes?
  - Which processes cooperate in a co-routine like fashion -- that is, which processes share intermediate states of their artifacts and work concurrently, iteratively and cooperatively?

- *Possible Process Improvements*
  - What kind of improvements have been made to the process?
  - Where might we look for improvements? Tools, improved communication, simpler processes, elimination of duplicate effort, etc.?

The Process as Executed

- *Roles and Their Interactions*

  How do the various roles interact with each other in executing the various tasks, subprocesses and processes? What are the attributes of these various roles? The importance of this approach is to provide an understanding of what the actual process is as opposed to how it is supposed to be.

- *Non-conformance to Process Model*

  Given that we view the process model as a prescription for how the processes should be done, it is important to understand where the process model is not being followed and why. These reasons for non-conformance may range from not understanding the process (for various reasons) to having discovered improvements and optimizations to the process.