Table of Contents

Table of Contents ........................................................................................................................................... i
1. Introduction.................................................................................................................................................. 1
2. Related Work ............................................................................................................................................... 2
3. Research Plan.............................................................................................................................................. 3
  3.1 Stage 1 – Exploratory................................................................. 3
    3.1.1 Basic study ........................................................................... 5
    3.1.1 Advantages and disadvantages ........................................... 5
    3.1.2 Variations ........................................................................... 6
    3.1.3 Integrating results .............................................................. 8
  3.2 Stage 2 – Hypothesis testing ..................................................... 8
    3.2.1 Basic study ........................................................................... 8
    3.2.2 Variations and threats to validity ...................................... 9
  3.3 Stage 3 – Process improvement ................................................. 10
    3.3.1 The Quality Improvement Paradigm .................................. 11
    3.3.2 Example scenario ............................................................. 12
4. Educational activities ................................................................................................................................. 13
  4.1. Course development ............................................................. 14
  4.2. Student research ................................................................. 14
5. Conclusion ................................................................................................................................................. 15
1. Introduction

Software maintenance offers a significant payback opportunity for process improvement. It is commonly believed that maintenance consumes a large proportion of total system cost (often up to 85% as reported by Lientz and Swanson [14] in 1978; no more recent studies have attempted to update this finding). For this reason alone, lowering maintenance cost is a profitable goal. Further, improving the quality of maintenance practices could lengthen the useful life of a software system, thus providing a greater return on investment with respect to the system’s original development costs.

In examining software maintenance processes for improvement opportunities, an obvious choice is information flow. Curtis et al. [8] have shown that information flow is a key factor in the success of software development and maintenance efforts. In maintenance in particular, obtaining accurate, up-to-date, and understandable information about the system being maintained is a major task. It is also a difficult task because these sources of information are often limited, inaccessible, or unknown. Clearly this impacts both maintenance productivity, simply because of the time it takes, as well as the quality of system changes, which depends on the quality of the system information available. The situation is often further complicated by the growing trend towards component-based (or COTS-based, or package-based) systems because maintainers must find sources of information about the components as well as the system being maintained.

Currently, software maintainers rely on various sources for information about the system they are trying to modify, adapt, or update. These sources range from artifacts of the development process (requirements documents, design documents, documentation within code, test plans and reports, etc.) to user documentation (user manuals and configuration information) to the system itself (both the running system and the source code) to direct access to the system’s original developers and current users. Many of these sources, especially those in the first category, are produced at least in part to facilitate maintenance. A lot of effort is required to both produce and maintain the documentation, but little effort has been made to determine how useful it is for maintenance.

The objective, then, of the work proposed here is to improve the way maintainers gain information about the systems they are maintaining. The approach to this work is empirically based. This means that any proposed improvements must be derived from previous empirical evidence concerning how maintainers currently gain information about the systems they are maintaining. The NSF has recently publicly recognized the importance of empirical studies in software engineering [7]. Because of the lack of previous empirical work in this area (see section 2), this proposal includes studies aimed at gathering empirical knowledge about the problem, as a necessary prerequisite to empirical validations of process improvements suggested by this knowledge. The research plan is organized as a series of empirical studies that, at first, aim at gaining knowledge about how maintainers gather system information, then building on that knowledge to experiment with changes in how they gather this information. The education plan is tied into this work as well by involving students in all aspects of the research and by incorporating results into the design of new courses.

Industrial participation is a key ingredient in the success of this proposed work. This participation is crucial not only in order to obtain valid and useful results, but also to ensure that the significant long-term potential of this work is realized. The impact of this work, because it will provide basic insights that we do not currently have, extends far beyond the scope of this proposal. It could affect maintenance (and development) practice all over the industry, but the findings and proposed improvements will not be accepted if they are not based on experience in industrial settings. This principal investigator is particularly well qualified to carry out this work, given her experience and background in partnering with industry in conducting research, in applying a variety of empirical methods and paradigms to software engineering studies, and in conducting empirical studies of software maintenance.
2. Related Work

There have been few empirical studies of maintenance practice in the literature. Probably the most famous and most often referenced is Lientz and Swanson’s [14] survey of 69 software maintenance organizations in 1978. This in-depth study produced the often-cited figure of 50-85% as the proportion of system costs devoted to maintenance. However, this study is quite dated at this point and we have no way of knowing if these results still hold in today’s maintenance environments. A more recent study of maintenance practice is Singer’s [24] study of 10 maintenance organizations. This study, unlike the earlier one, was a qualitative study and focused more on process than outcomes. That is, maintainers were asked (in interviews) about how they went about doing maintenance and what resources they used and needed. The eventual objective of this study was to inform the development of maintenance tools. This study contributes quite a bit to the proposed work, in terms of preliminary data on information sources used in maintenance, as well as an example of the use of qualitative methods in the study of maintenance.

Another example of using qualitative methods in studying maintenance, in which this principal investigator was involved, was Briand et al.’s study of a NASA maintenance organization [6]. In this study, interviews were used, along with an organizational modeling technique called Actor-Dependency modeling [27], to gain insight into the organizational issues and problems in the maintenance environment. The results uncovered problems such as organizational bottlenecks and a lack of participation from certain stakeholders in key process steps. An additional contribution that this study makes to the proposed work is the role it played in the methodological experience and training of this principal investigator. As a graduate student, she played the role of methodologist in this study, taking the lead on organizing and analyzing the qualitative data. This became her first experience in this area, in which she now is regarded as an experienced practitioner [20].

There is a major body of literature in program comprehension that addresses how programmers study source code in order to understand a software system. Much of this work ([26] and [5] are two good examples) focuses on the comprehension processes of maintainers in particular. This body of work differs from the proposed work in two important ways. First, the program comprehension work focuses on only one source of information, the source code, while the proposed work will study the use of all kinds of information sources. Second, the program comprehension work addresses a different step in the process than we propose to look at. The proposed work focuses on information gathering activities, i.e. how maintainers get information, while the program comprehension work focuses on information understanding activities, i.e. how maintainers process information once they have obtained it. In short, while the two bodies of work appear highly related, they are really addressing two different (but not entirely unrelated) questions. However, one contribution of the program comprehension literature to the proposed work is an analysis, or taxonomy, of tasks and objectives that maintainers are motivated by when they seek information. This analysis is particularly well worked out in the work of von Mayrhauser and Vans [26]. This taxonomy of tasks may inform the design of one of the variables used in the proposed research.

Another contribution of the program comprehension work that will be very useful to the proposed work is an analysis of the literature in psychology that applies to this area of software engineering. Clearly psychology plays a role in how people gather and understand information, how they go about seeking information to address a particular question, etc. The literature on learning styles and comprehension processes is certainly relevant to the proposed work, as it was to the program comprehension work. However, this principal investigator has no background in psychology (only an appreciation of its importance) and is not qualified to navigate and analyze this literature herself. Instead, we will rely on the analyses already done by other software engineering researchers (in particular from the program comprehension literature) and on her colleagues in the Information Systems department at UMBC, several of whom have extensive backgrounds in psychology (see the Departmental Endorsement and Certifications page).

Another outside area that will be consulted during this project is organization theory, in particular work on communication in organizations and what hinders and facilitates information flow. This is relevant to the accessibility and visibility of human sources of information for maintenance. There are viewpoints in the organization theory literature that consider communication the central issue in organization design [16].
The purpose of organizational structure is often articulated as the reduction of uncertainty [9], which is defined as a function of the difference between the information needed to complete a task and the information currently possessed by the organization. There are a number of structural mechanisms by which organizations manage uncertainty and lack of information. Allen’s work [1] in technological organizations (such as software development and maintenance organizations) points out that the needed information is encapsulated in material products as opposed to existing in literature, as in many scientific disciplines. Allen’s thesis is that, because of the difficulty of extracting information from physical products, direct communication between technologists themselves is vital. The theme that can be found again and again in the work cited above is that organizational structure and information processing in organizations are linked by design. Organizations are either designed to facilitate information flow, or they evolve over time to meet this need. In the proposed work, one objective is to find organizational mechanisms to facilitate the flow of information about systems under maintenance. This PI already has considerable experience with this literature and with applying it to software engineering [21, 22, 23].

3. Research Plan

The proposed work falls into three stages. The last stage is the proposal of improvements to maintenance practice and the empirical demonstration of their effectiveness. Before that can be done, however, we must learn more about how maintainers currently gain information about systems. In particular, we need to learn what information sources are efficient, effective, and available in different situations. This is the focus of the second stage of the proposed work. But there is another prerequisite even to this type of study. It is not currently clear what variables and factors need to be considered in the study of information sources in maintenance. What is the best measure of the “effectiveness” of an information source? What constitutes an “information source”? What other factors (programmer background, application domain, system size, etc.) might affect how a maintainer uses an information source? If we do not gain a better understanding of these questions than currently exists, then the second stage of this research may be seriously misguided. Thus, the first stage of the proposed work is an exploratory stage that addresses these and similar questions. These three stages and the relationships between them are depicted in Figure 1.

3.1 Stage 1 – Exploratory

The objectives of the studies conducted in this stage are to identify the relevant variables and hypotheses to be used in the second stage. These studies are exploratory in nature, meaning that they are intended to generate, not test, hypotheses. The starting point for this stage, then, is not a set of hypotheses, but a set of research questions:

- What do maintainers consider “information sources”?
- How is an information source judged to be “good”?
- What factors affect whether or not a given information source is “good”?

Answering these questions will provide input into defining independent variables (the “information sources”), dependent variables (various measures of “good”), and intervening variables (factors that affect “goodness”) and for defining hypotheses that describe the possible relationships between them. There is some literature that provides suggestions for these variables. For example, the source code is one obvious information source that is the subject of much of the program comprehension research [26, 5], as mentioned earlier. Other information sources suggested by [24] are other people and execution traces. How to measure “goodness” of an information source has less support in the literature, but some measures that seem reasonable are the amount of time required to consult the source, whether or not the source provided the information the maintainer sought, and the quality of the system change that was being made when consulting the source. Some intervening factors that are suggested by the literature are program structure (e.g. unstructured vs. structured vs. object-oriented) and programmer background [5]. Others that seem obvious are the maintenance task being performed, the perceived quality of different sources, and the availability of different sources.
Qualitative research methods, especially those aimed at generating theory, are particularly well-suited to this type of research, and this PI has considerable experience in applying such methods to software engineering questions [6, 19, 20, 21, 23]. Below is described the design of the basic study in this stage, followed by the advantages and disadvantages of that design. Then several variations will be described that will overcome some of the disadvantages of the basic design.

Figure 1
3.1.1 Basic study

The first study in this stage will take place in the context of a graduate-level software engineering course in the Information Systems department at the University of Maryland, Baltimore County, in the fall of 1999. As a class project, teams of students will be required to choose a real software system, in use in some organization. Most students in the class will be working IS professionals, so the software systems will most likely come from the companies in which they work. Some teams may also use previous student projects, or projects provided by the instructor. Some students may have been involved in the development of the systems they choose for the project. Each team will be required to identify one small-to-medium, relevant modification to their system and implement that change, using the software engineering practices taught in the class.

Once the teams have chosen their systems, each team member will be asked to fill out a form listing all the sources of information available on that system, including documentation from the development process, user documentation, comments and source code, development personnel, and users. The students will also be asked what form these various sources take, e.g. paper-based, electronic, hypertext, or what level of availability a human source has. This information will help answer the first of the three questions addressed by this stage.

During the semester, while the teams are working on their projects, they will be required to maintain a record of each information gathering activity they engage in. That is, each time they attempt to gather information of any kind about the system they are maintaining, they will have to record information including the information source they attempted to use, how long the activity took, what information they were seeking, and whether or not they found that information. This information is aimed at testing out several possible measures of the “goodness” of an information source. At the end of the semester, each team member will also be interviewed to gather data on their views of “goodness” of different information sources. They will be asked which information sources were the most useful, and how they define “useful.” These subjective responses will be compared to the more objective data collecting during the course of the project to determine how perception compares to reality, i.e. which measures best reflect the maintainer’s perception.

In these interviews, the teams will also be asked the relevant “why?” questions. That is, they will be asked why they relied on certain information sources rather than others, why some sources were more successful in providing information than others, and why using some sources took longer than using other sources. These questions address the third question, concerning the various factors that affect the “success” of an information source.

All of the data gathered from this study will be analyzed, using a grounded theory approach [10], to yield a set of variables and a set of hypotheses to be used in the next stage. Grounded theory methods are used extensively in education and the social sciences to explore an area that has little existing theory to guide its study. These methods have been used extensively by this PI [6, 19, 20, 21, 23] as well as others [24, 18] in studies of software engineering.

3.1.1 Advantages and disadvantages

When an empirical study is described in the literature, it is common to discuss the threats to validity associated with the study design. Often, this discussion is organized around a set of common threats, such as those described in [12], including threats to external, internal, and construct validity. These types of validity, however, usually describe the validity of the causal relationships that the study is aimed at demonstrating. Thus, in an exploratory study such as the one described here, that does not aim to show causal relationships, the relevant discussion is a bit different. Other concerns related to exploratory studies are representativeness (which is related to the concept of external validity), data accuracy (which has to do with internal validity), and validity of methods (which affects all aspects of validity).

The study will cover a variety of different types of systems. All of the student projects will be real-world systems, but will come from different types of organizations and from different application domains. Thus, the study will be somewhat representative in terms of types of systems. However, it lacks in
representativeness in several other areas. First, and probably most important, the study will involve students working on academic exercises, not real maintainers doing real work. An effort will be made to make the experience as real as possible, but in the end it will be an academic exercise. This means that the subjects will not be representative of the population of software maintainers either in terms of experience or motivation. Also, the process that they follow may not be representative of industrial maintenance processes. Finally, the systems they work on, although they will represent a variety of systems, will probably lie on the smaller end of the system size spectrum than exists in industry.

One advantage of this study design is the immediacy of the data collected. That is, the subjects will be asked for information about activities that they are either currently engaged in or that are very recent. This will reduce the effects of time and memory on the accuracy of the data. It will also allow the collection of data at a very detailed level, i.e. at the level of each information-gathering activity. Other measures that will be taken to ensure data accuracy are audiotaping interviews (field notes will be written from the audiotapes) and triangulation (data will be collected separately from each team member). One threat to data accuracy, however, is the fact that students may be motivated to distort their responses because they believe that the instructor will reward certain answers with better grades. For example, a team might consult a wider variety of information sources (or report that they consulted a wider variety) than they might otherwise simply because they believe the instructor wants them to. This behavior will be countered by explaining the study’s objectives to the students, in particular the non-evaluative aspect of the study, and by arranging the grading system so that participation in the study is graded, but not performance in the study. For example, students will be penalized if they do not provide the data requested but will not be graded on the quantity or content of that data. Also, students will have the option to request, at any time, that the data they provide not be used in the study. This request can come even after the end of the semester, so that there is no chance for such a request to affect their grade. These measures are also related to the ethical issues involved in using student subjects, in particular that a student’s participation in the study does not affect their grade, and that students are not coerced in any way to participate.

The qualitative, grounded theory methods used are designed for this type of inquiry, where there is little theory in the literature that can serve as a starting point, i.e. that provides measures and variables, or hypotheses. These methods have a long history of use in many disciplines, and have been exercised in studies of software engineering as well. This PI, in particular, has had experience in applying these methods in software engineering, and has studied their use in this and other disciplines. The disadvantage of qualitative (in particular grounded theory) methods is that the results they produce are sometimes difficult to accept in the software engineering research community because they seem “softer” and “fuzzier” than statistical findings. In fact, qualitative results are hard results, and in many cases they provide a truer and richer picture of the phenomenon being studied than any quantitative method could provide. This PI also has had experience in successfully presenting (and publishing) her qualitative results and has contributed to the growing (although slow) acceptance of these methods in her field.

### 3.1.2 Variations

To counter the disadvantages outlined in the previous section, several variations on the basic study will be performed. These variations will address the same research questions, will collect the same form of data (qualitative), and will use the same data analysis methodology (grounded theory). However, the sources of data and the data collection methods will vary.

The first variant will consist of a survey questionnaire sent to software professionals actively working on maintenance projects. The survey recipients will be asked to reflect on their maintenance experience to provide data on the information sources they have used to carry out maintenance tasks. Although the questionnaire has not yet been designed, it will address the three research questions presented at the beginning of section 3.1:

- What do maintainers consider “information sources”?
- How is an information source judged to be “good”?  
- What factors affect whether or not a given information source is “good”?
The survey recipients will be chosen with the aid of industrial contacts in a minimum of three different software organizations. The principal investigator already has contacts that may be used in such organizations as NASA, Social Security Administration (letter of support attached), IBM Canada, Lucent Technologies, Telcordia, Nortel, and Computer Sciences Corporation. The subjects will be maintainers with at least 6 months experience on at least 2 different maintenance projects. Permission to distribute the survey to these maintainers must be procured from their managers. Their managers will also be asked to issue a statement supporting the study, in order to increase the response rate.

The industrial survey increases the representativeness of this set of exploratory studies, by including a variety of industrial sources and reflecting maintenance projects in a variety of settings and application domains. The tradeoff, however, is that data accuracy may be diminished. Instead of providing information about activities in the immediate present (about which we have a lot of background information, as in the classroom study), the survey respondents will be asked to reflect on their previous experience and provide information on their experience in general. The survey will collect information on the types of systems the respondents have worked on, but the level of detail and the effect of memory may impact data accuracy.

The second variation also takes place in an industrial setting. A major design concern in this study is to collect more detailed and immediate data (to enhance data accuracy, as discussed earlier), while at the same time conserving the time and effort of professional participants. For these reasons, the study will be narrowly scoped and small, restricted to a handful (~6) of maintainers in one organization. The organization will be chosen based on the variety of information sources available for maintenance and the variability among the other relevant factors that emerge from the two previous exploratory studies. In other words, the subjects for this study will be chosen based on the results of the previous studies. This approach is often called theoretical sampling [17]. The idea is that intermediate propositions can be checked by focusing subsequent data collection on data that might support or refute the proposition. For example, if the results of the first two studies indicate that programmer experience has a strong influence on how different information sources are used, then subjects will be chosen for the third study that have a variety of different levels of experience. This is not the same as hypothesis testing (which comes in the next set of studies), but serves to gain a deeper understanding of the findings and the conditions under which they hold.

This study will begin with period of observation, where the researcher will observe the subjects at various times while they are performing maintenance work, including information gathering activities. The purpose of these initial observations is to gain a better understanding of how these subjects carry out maintenance tasks, in order to inform the design of the data collection instrument that will be used in the rest of the study. The instrument must be designed to fit as seamlessly into the process as possible and ask for data in a form that is natural for the subjects to express. Using an instrument that is incongruent with subjects' work practices leads to confusion and discomfort on the part of the subjects, and thus to inaccurate data.

After the initial observation period, and design of the instrument, the subjects will be asked to keep a log of their information-gathering activities. For each such activity, they will be asked to record what information source they consulted, whether or not they considered it successful and in what ways, and what factors affected their use of the information source. The questions asked of these subjects are more open-ended and the instrument less structured than in the first exploratory study. This, hopefully, will allow us to learn from the more extensive experience of these professional subjects. After the specified logging period (about one month), each subject will also be interviewed and will be asked to give more detail on their answers.

Although the subjects for this study will be chosen to be representative according to some specific factors of interest, the overall representativeness of the sample is severely restricted by the size of the study. On the other hand, the accuracy, specificity, and degree of detail of the data will be very high.
3.1.3 Integrating results

The exploratory studies described in this section are designed to balance each other’s advantages and disadvantages. For example, the representativeness issues present in the first and third studies are balanced by the wider representation of industry in the second study, the industrial survey. The tradeoff with the second study, however, is the generality and retrospective nature of the data. The other studies counter this by collecting very detailed data on current activities. The other tradeoff that is balanced among these three studies is cost. Using student subjects has a number of disadvantages, but it makes it possible to collect reasonable amounts of detailed data, while the time and effort of industrial subjects must be carefully conserved. A related issue is opportunity. Opportunities to work with industrial subjects are rare, compared to the opportunities to use student subjects, so again those opportunities must be exploited carefully.

Another factor that makes these three studies fit together nicely is that they are designed to be more and more focused as we progress and gain understanding of the relevant issues. While all the data collection instruments are fairly open-ended, the choice of subjects in the last study (and possibly to some extent in the second study) will be guided by previous results.

All of the data collected from all three studies will be in narrative form, and will be analyzed in the same way, as described earlier. The data from each study will be used to strengthen, refute, or modify propositions generated in the previous studies. The final stage of data analysis will be a round of member checking [15], where the results of the analysis are presented to and feedback solicited from the subjects who provided the data in the first place. The end result will be one set of proposed variables and the relationships between them (represented as a set of proposed hypotheses). These variables and hypotheses serve as the input for the next set of hypothesis testing studies, described in the next section.

3.2 Stage 2 – Hypothesis testing

The starting point for this stage is the set of hypotheses that are generated in the previous stage. Of course, we cannot know what these hypotheses are ahead of time, but below are some examples showing some possible form and content:

- Asking the developer of the system under maintenance is more efficient (in terms of taking less time) than consulting documentation when trying to gain information about side effects.
- Requirements documents are more useful (in terms of the probability of finding the information) than the source code for finding out the format of user input.
- If a data-flow diagram of the system is available and it is perceived to be accurate, then it is consulted more often than the source code.
- Source code is chosen as an information source over any other form of documentation if the documentation is not perceived to be accurate.

It is not known at this point how many hypotheses will be generated by the first stage, how many of those will be of sufficient interest to warrant testing, or how many different studies will be required to test them (often a single study can test several different, but related, hypotheses). So, in order to define the scope of this proposal, this section describes a set of studies that will be used to test a single set of hypotheses that are sufficiently related that they can be tested together from the same set of data (e.g. they reference the same variables). The details of the studies designed to test these hypotheses will depend on the hypotheses themselves. Thus, this proposal is limited to discussing an overall strategy, variations that different specific studies might take, and the balance between different threats to validity of these variations.

3.2.1 Basic study

The data collection instrument for this study will be a form that is based on the relevant variables (independent, dependent, and intervening) that emerge from the data collected in the previous set of studies. The participants in the study will be asked to fill out a copy of this form for each information gathering activity they engage in as part of their maintenance work. The first question on the form will ask the subject to check off, from a predefined list, the information source consulted. This predefined list will
contain all the information sources revealed in the exploratory studies (and that are available in the setting for this study). Then the subject will be asked to indicate the value of each of the intervening variables that emerge from the exploratory studies. Some of these variables will have a checklist of possible values, while others will have to be filled in by the subject. Also, some of these variables (such as programmer experience, for example) can be filled in ahead of time as they will be the same for each information gathering activity for the same subject. Then, the subject will be asked to fill in a value for each of the “goodness” measures, which evaluate how successful each information source was. Finally, there will be a few open-ended questions that will ask the subject to describe any other factors that are relevant that are not covered by the predefined choices on the form. In Figure 2 is shown an example of such a data collection form. Of course, the actual variables and values on the example form are not the same as those that will appear on the form used in the study, but they represent some initial guesses at what these variables and values might be.

The exact form of the data analysis will depend on the hypothesis being tested. One possible scenario is that the data points (information gathering activities) can be partitioned by independent variable (information source), so that the information source is considered the “treatment”. The partitions corresponding to the information sources in the hypothesis being tested will be compared with respect to the values of the dependent variables (“goodness” measures). These comparisons will be made statistically, with different statistical methods depending on the nature of the variables being tested. For example, the $\chi^2$ test could be used if the dependent variable is nominal or ordinal. If the dependent variable is continuous and normally distributed, then a $t$ test might be used. Other non-parametric tests exist for non-normal data (e.g. Mann-Whitney and Kruskal-Wallis, etc. [11]).

3.2.2 Variations and threats to validity

In the study description above, no mention was made of the subjects or setting of the study, and this will be the main source of variation among the studies in this stage of the proposed work. The combinations of subjects and settings are designed to balance the various threats to validity, which are described below. The choice of settings for the different studies will also be made with respect to the hypotheses being tested. Sites will have to be chosen that exhibit sufficient variety among the independent and intervening variables cited in the hypothesis being tested.

The first study to be conducted in this set will be a pilot study, which will be conducted in an academic setting with students as subjects. This approach will help test the study design and remove any flaws before committing the resources required for a study in an industrial setting. The pilot study will take place in a graduate-level software engineering course in the Information Systems department at the University of Maryland, Baltimore County (the same course, in a different semester, as that described in section 3.1.1). Although valid results will come from applying the study design in this setting, the most important outcome will be a debugging of the design and of the data collection form. This will enhance construct validity [12] in particular, as it will help ensure that the subjects understand the questions being asked and thus the answers provided will correspond to the concepts we intend to measure.

Another issue that must be considered when designing these studies is that of controlled vs. uncontrolled conditions. Conducting a controlled experiment with the above design in an industrial setting would involve a large number of maintainers working on the same system, with certain information sources withheld from some maintainers while made available to others. In addition, all of the relevant intervening variables would have to be controlled. Such a study would be difficult to implement. Opportunities to conduct such an experiment will probably be rare, but any such opportunities will be exploited.

The advantage of a controlled experiment is that internal validity (the ability to draw a causal inference [12]) is strengthened. Thus, it will be important to conduct as many of these studies in controlled settings as possible. Since controlled experiments in an industrial setting are so costly and difficult, it seems likely that the controlled experiments in this set of studies will take place in a classroom setting. The disadvantage of this plan, of course, is that it threatens external validity, i.e. it makes it more difficult to generalize the results because the conditions (subjects and setting) may not be representative of the software industry.
Thus, in order to balance cost of experimentation with internal and external validity, at least three studies, besides the pilot study, will be conducted in this stage. One will be an uncontrolled experiment in an industrial setting, one will be a controlled experiment in an industrial setting, and one will be a controlled experiment in a classroom setting. This will provide a significant amount of data from at least two different industrial settings (thus strengthening external validity), as well as two different sets of findings from controlled situations (thus strengthening internal validity).

3.3 Stage 3 – Process improvement

The next step in this line of investigation is to reflect on the results of the first two sets of studies and translate them into concrete process improvements. The studies comprising this stage must be described in this proposal in an even more general way than those in the previous stage because the process
improvements that will be evaluated are completely unknown at this time. However, what can be described here is a well-defined approach to process improvement that will be followed regardless of the nature of the process changes or the setting.

Although we cannot know at this point what process improvements will be proposed, several example scenarios are envisioned. If, for example, the results of the previous studies indicate that human sources of information (i.e. developers and users of the system) are always the most effective and efficient sources of information, then this has implications for the organizational structure of software organizations. It implies that maintenance can best be facilitated by making those human information sources as available to maintainers as possible. This may mean making maintenance support part of the job description of the developer of a system after it is delivered and explicitly budgeting time for such support. Or it may simply mean that developers are not transferred too far away (either organizationally or geographically) from the maintainers of the systems they develop. It may mean incorporating users into the maintenance organization as well. In terms of documentation, an improvement may simply be to produce less documentation during the development process, given that it will probably not be used during maintenance. Another improvement, in this case, might be to require developers’ names on all system documentation to make it easier for maintainers to identify potential sources of information.

As another example, if a particular piece of documentation is used heavily during maintenance, but is often not effective in providing the information maintainers expect from it, then the development process might be modified to put more effort into producing that document with higher quality. On the other hand, if a lot of development effort is put into producing and maintaining a particular document, but it is rarely used by maintainers, then an improvement might be to eliminate that document from the development process altogether, thus cutting development effort.

In the sections below, a general framework for implementing and evaluating process improvements, the Quality Improvement Paradigm (QIP), is described. This paradigm will provide the underlying strategy for all of the work in this stage, but it does not provide details, such as experimental design, that must wait until we know more about what kinds of improvements will be evaluated. After the discussion of the QIP, a possible experimental scenario will be described. This is an example, used to help clarify the strategy used in this phase.

Again, a word must be said about the scope of this proposal, as compared to the real scope of this work. If successful, the previous two stages will suggest numerous process improvements in many different software maintenance organizations. The implementation and evaluation of many of these improvements will hopefully be undertaken by my colleagues and other researchers and practitioners who become familiar with this work. The scope of this proposal includes only the work that I intend to perform to begin this line of research. Specifically, in this stage, this proposal covers only 2-3 iterations of the QIP, in one software maintenance organization, in order to implement and evaluate one identifiable process improvement (or set of closely related improvements that can be evaluated together).

3.3.1 The Quality Improvement Paradigm

The Quality Improvement Paradigm (QIP) is a high-level process for iteratively understanding and improving software engineering in a particular context. The QIP comprises six steps, which are performed iteratively in a cyclic fashion [2, 4]:

1. **Characterize.** Understand the organization being studied based on the available data, models, experience, and insights. Establish baselines with the existing products and processes in the organization and characterize them. In the context of this proposal, this would include examining the values of the relevant independent and intervening variables (from stage 1) as they apply to the organization. The baselines would be profiles of the “typical” values of the dependent variables from stage 1. Also, in this step, the hypotheses from stage 2 would be examined to see which are relevant in this environment.
2. **Set goals.** Based on the characterization of the organization, set quantifiable goals for project and organizational performance (and improvement). The reasonable expectations should be based on the baselines provided by the characterization step. In our context, this would mean choosing the dependent variables that we wish to affect and setting goals for how much we want to affect them.

3. **Choose process.** Based on the characterization and goals, and on the hypotheses and other experience gained in the previous stages of this research, choose the process changes, including specific tools and techniques, to be evaluated as means to achieving the goals. Also, a plan is made for evaluating the proposed process changes, including the project(s) used for evaluation. The process changes and the evaluation plan must be consistent with the goals and constraints set for the project(s) being studied.

4. **Execute.** Carry out the project, with the proposed process changes and the evaluation plan in place, by constructing the products and collecting data to monitor progress.

5. **Analyze.** At the end of the project, analyze the data and the information gathered to evaluate the process changes, determine problems, record findings, and make recommendations for future projects.

6. **Package.** Consolidate the experience gained in the form of new or updated models, documents, processes and other forms of knowledge and store it so that it is available for future projects.

The iterative nature of the QIP comes in after the packaging step, in which decisions are made about what relevant issues need further study, what questions remain unanswered, and what further improvements could be made. With these in mind, the whole cycle is repeated.

### 3.3.2 Example scenario

Suppose that a particular software development organization wishes to reduce its maintenance costs and has asked us to help them do this, using the QIP as a basic framework and based on the results from our previous maintenance studies. Suppose also that one of the hypotheses generated from our exploratory stage and supported in our hypothesis testing phase states that, when available, using class diagrams of a system in addition to the source code is more efficient than using the source code alone in designing enhancements to the system. The development organization would like to explore the possibility of using class diagrams during maintenance.

The first step would be an in-depth characterization of the organization’s development and maintenance processes. Special attention would be paid to how class diagrams are currently or could be incorporated into these processes. Also, any independent and intervening variables relevant to the class diagram hypothesis would be evaluated. Finally, baseline values would be calculated, with as much detail as is possible given the available data, for any relevant measures of maintenance efficiency and cost (the dependent variables). At the same time, we would check to see if any of the other hypotheses from our previous work might be relevant in this organization.

The second step is goal setting, in which we define exactly what we expect to gain from changing the process. In this case, this would most likely include goals for target levels of efficiency and cost. As well, there may be goals concerning quality and project duration, to reflect the fact that the organization is concerned that these aspects do not degrade for the sake of decreased cost.

The third step is where the process improvements are designed in detail. Here, we would decide how and when class diagrams would be produced during the development process, how they would be transferred to the maintenance organization, how they would be updated during maintenance, and how they would be consulted as part of the maintenance process. Measures must be defined to allow data collection during enactment of the processes and analysis after the fact. The Goal/Question/Metric (GQM) [3] approach would be used to define these measures. The GQM method starts with a set of goals (defined in the second step) and derives a set of measures that will result in data upon which the goals can be evaluated.

The other important set of decisions that must be made in this step is how the process changes will be applied, to what projects they will be applied, and for how long. Often, it is possible and desirable to apply changes in a small, controlled environment to isolate the effects of the change from other confounding factors. This is often too expensive to be practical, and it does not ensure that the changes will work in the same way in the real environment as they did in the controlled environment. However, this approach is
useful as a first step when followed, in subsequent iterations of the QIP, by case studies applying the changes in real, uncontrolled settings. In either case (controlled or uncontrolled), the case study must be planned very carefully, with special attention to possibly confounding factors (many of which cannot be controlled even in a “controlled” experiment) and the instruments and measures that will be used to make the comparisons [13].

Suppose that, in this example, it is decided that a controlled setting is not practical for these types of changes. Instead, it is decided to divide the problem into two parts, to be addressed in two separate iterations of the QIP. First, we must answer the question of whether or not using class diagrams during maintenance provides a cost savings. The second question is whether or not this savings is sufficient to counter the added cost of producing and maintaining the class diagrams themselves.

To address the first question, a system is chosen that is currently under maintenance, and a set of class diagrams is produced for this system separate from the development effort (which is already over anyway). Producing these class diagrams might be a good student project. Then the diagrams are provided to the maintainers to use in subsequent enhancements to the system. The efficiency and cost of these enhancements are measured (using the measures defined in step 2) and compared against the baseline (determined in step 1). This constitutes steps 4 (execute) and 5 (analyze) of the first iteration of the QIP. Packaging (step 6) consists of updating process documentation, institutionalizing the measures, and disseminating the lessons learned.

The focus of the second iteration, then, would be on the development process. The development process would be characterized in the first step (adding to the information gathered in the first iteration) and goals would be set in the second step for how the process changes should affect development outcomes (cost, duration, quality). The cost goals, in particular, would be informed by the results of the first iteration, because the goal should be that the increase in cost in the development process should not exceed the cost reduction in the maintenance process. Then, in step 3, the process changes will be defined in detail, as well as the measures and the scope of this iteration (what projects, for how long, etc.). The process will then be executed (step 4) and the data analyzed (step 5). Packaging (step 6), again, will consist of updating the process documentation, standardizing the measures, and disseminating the lessons learned. In addition, the results of the two iterations should be disseminated together, as they are designed to work together.

Of course, there is the possibility that the results of one or both of these iterations may not be completely successful. The process changes may turn out to be effective only under certain circumstances, or their effectiveness might not be apparent at first. Repeated iterations of the QIP, each focusing on issues that surface in previous iterations, may be necessary. As stated earlier, the scope of this proposal includes only 2-3 iterations of the QIP in one organization. However, the hope would be that the organization where this work takes place would find it worthwhile enough to incorporate it into their own continuous process improvement efforts.

4. Educational activities

The principal investigator on this proposed project, Dr. Seaman, is an assistant professor in the Department of Information Systems (IFSM) at the University of Maryland, Baltimore County (UMBC). Her duties include teaching both undergraduate and graduate courses in systems analysis and design and database design. In addition, she is taking the lead in expanding the IFSM department’s coverage of software engineering topics, including maintenance, documentation, testing, and object orientation. Expansion of the department’s offerings in these areas is a stated goal of the department.

This proposal specifically supports two of Dr. Seaman’s pedagogical duties. The first is the design of a new software engineering course in the IFSM department that concentrates on the latter stages of the development process. The second is the involvement of students, in particular undergraduates, in research projects.
4.1. Course development

A new course to be designed and taught by Dr. Seaman in the IFSM department diverges from a traditional course in software engineering in several ways. First, it must have an information systems focus. That is, it does not have to be broad enough to encompass methods and techniques for implementing real-time, embedded, or scientific software. Instead, the specific techniques and examples used can be tailored to those normally used and encountered by IS professionals. Second, it will concentrate on the latter stages of the development process, specifically implementation, documentation, testing, and maintenance. Project conception, requirements analysis, and design are already covered well in existing IFSM courses. This means that coverage of the latter-stage activities can be far deeper than is normally possible in a traditional software engineering course. This affords Dr. Seaman the opportunity to address some current research areas in the course, in addition to well-established practice.

The proposed research plan, described above, will contribute to the design and execution of this new course in several ways. Because the students will serve as subjects in several of the studies, they will be exposed to current research issues in maintenance and documentation. They will also have the opportunity to get a close look at an empirical study in progress, to discuss some experimental design issues, and to debate their own views on information sources for maintenance. On the other hand, the studies themselves will serve as a tool for teaching maintenance practices. The data collection forms, for example, will also serve as instructional vehicles for teaching students about the steps in the maintenance process and the information sources that should be considered. It will also force the students to examine their own project work more closely by keeping track of the amounts of time they spend in various activities. This close scrutiny makes it nearly impossible for students to complete the projects in an ad hoc, “seat of the pants” manner that, although possibly more realistic, should be discouraged in a learning environment. In this way, students will get the maximum value from the class project experience. The results of the maintenance studies will also inform the teaching of maintenance practices. After preliminary results are obtained, students in subsequent classes can be instructed in the most effective and efficient ways of obtaining information on the systems they are maintaining under different conditions.

Finally, taking on this line of research is a crucial factor in Dr. Seaman’s own development as an effective teacher, of maintenance in particular and software engineering in general. Software engineering has been recognized as an important area of instruction by the IFSM department at UMBC, and Dr. Seaman is one of the department’s key instructors in this area. Providing effective, relevant, and comprehensive instruction in software engineering to IFSM students requires faculty members actively involved in research in current software engineering topics. Participating in a long-term, focused line of research, such as the one described above, will allow Dr. Seaman to be an active participant in the maintenance research community. As such, she will be exposed to the latest developments in the field and will be able to bring that exposure to her students.

4.2. Student research

UMBC, as an institution, has a strong emphasis on undergraduate research, in addition to the research components of the graduate degree programs. Dr. Seaman is also committed to involving students in all aspects of her research. This will range from short-term projects with undergraduate students, to semester-long independent study projects with graduate students (IFSM Ph.D. students are required to engage in four such projects with different faculty members), to mentoring the research of Ph.D. candidates. In particular, the work proposed here provides numerous opportunities for both undergraduate and graduate students to participate at various levels. Much of the analysis proposed is not overly sophisticated, and would provide a good opportunity for undergraduates to perform basic statistical procedures. Involving students at both levels in some of the qualitative data collection activities (survey preparation, interviewing) would also be appropriate. Helping to prepare a questionnaire, for example, would help a student learn about clarity of thought and writing, and the ways in which bias can be reflected in the wording of questions. Observing and/or scribing for an interview with a practitioner subject would provide an opportunity for a student to visit an industrial workplace and hear how practitioners talk about their work.
5. Conclusion

This proposal describes a long-term series of empirical studies, each of which addresses the issue of how software maintainers gain information about the systems they are maintaining. The studies are described in three stages. The first stage includes exploratory studies meant to understand what information sources maintainers currently consider using, how they evaluate information sources, and what factors affect their successful use. The second stage is meant to evaluate hypotheses that are derived in the first set of studies concerning the effectiveness of the information sources being used. The third stage constitutes a process improvement effort, based on the empirical evidence gained from the first two stages. The designs of the proposed studies include a variety of research methods, both qualitative and quantitative, in order to triangulate and strengthen the conclusions. This program of research also has an important educational component in that it will contribute to two of the principal investigator’s institution’s objectives: development of software engineering courses and involvement of graduate and undergraduate students in research.

The work described in this proposal constitutes a significant step in the principal investigator’s early career as an academic and a researcher. It builds on the skills, background, and education she has achieved over the last eight or more years. As a student, she has been advised and mentored by two highly respected software engineering researchers (Richard DeMillo and Victor Basili) and has studied in two of the nation’s most respected software engineering programs (Georgia Tech and the University of Maryland). She has also worked as a software engineering researcher and developer in industry (Unisys Corporation). Further, she possesses the particular background needed to conduct effective empirical research in software engineering. She studied and participated in empirical work as a research assistant for seven years in the Experimental Software Engineering Group at the University of Maryland, directed by Victor Basili, and currently is a part-time Research Scientist at the Fraunhofer Center for Experimental Software Engineering – Maryland, also under the direction of Dr. Basili. Her own empirical work has incorporated a variety of research methods, both qualitative and quantitative. She has spoken and written about the use of qualitative methods in particular in software engineering studies, both in the upcoming special issue of TSE on empirical methods, and in a forthcoming book on advanced topics in empirical studies of software engineering (edited by Janice Singer and Khaled El-Emam). The principal investigator has also had experience applying empirical methods to software maintenance [6, 22]. In addition, she has also been involved with the International Software Engineering Research Network (ISERN), an international collection of researchers conducting empirical studies in software engineering. Her continued contact with this network will facilitate the continuation of this work beyond the scope of this proposal, through collaboration and replication of studies with other ISERN members. The work described herein provides this principal investigator with an opportunity to apply her various areas of expertise and to launch her academic career with a significant and important piece of work that will have a long-term impact on software development and maintenance practice.
Departmental Endorsements and Certifications

in support of the NSF CAREER program proposal submitted by

Carolyn B. Seaman

July 20, 1999

This document is in support of the attached proposal entitled, "Information Sources for Software Maintenance." The research proposed by Dr. Seaman is congruent with the standards for scholarship in this department. I strongly support the research she describes, and feel it will contribute greatly to the overall research standing of the department. In addition, her plans for contributing to our educational mission are consistent with our long-term goals. In particular, it has been a stated goal of the department, in response to the wishes of our students, to increase our offerings in software engineering, and Dr. Seaman was hired in part to help fulfill this goal. She is and will continue to be supported by the department in many ways, including our ongoing junior faculty mentoring program, an excellent administrative and technical staff, equipment (granted as part of her hiring package) including two high-end PCs and an equivalent laptop, network access to a large variety of campus computing resources (UMBC was recently named one of the 100 most wired campuses in the country), access to the extensive library system of the University System of Maryland, a half-time graduate student assistant every semester (whose duties are at her discretion), and a set of esteemed colleagues with diverse backgrounds that can offer her assistance in a wide variety of areas. In particular, our faculty members are willing and able to assist Dr. Seaman in several areas specific to the attached proposal: psychology (Dr. Anthony Nascio and Dr. Henry Ensminger), survey techniques (Dr. Patricia Fitches), and quantitative experimental design (Dr. Henry Ensminger).

I would also like to verify that Dr. Seaman is currently in her first tenure-track appointment, which began here, in the Information Systems Department of the University of Maryland Baltimore County, on August 17, 1998.

Finally, I certify that I have read and endorse this Career Development Plan. Dr. Seaman's Ph.D. and postdoctoral work has already contributed substantially to the field of software engineering. This award would enable her to develop a successful academic career in a field in which she has much to offer.

Sincerely,

[Signature]

Jennifer Prince, Ph.D.
Dr. Carolyn Seaman  
Department of Information Systems  
University of Maryland Baltimore County  
1000 Hilltop Circle  
Baltimore, MD  21250

Dear Dr. Seaman:

As Director of the Software Process Improvement Staff in the Office of Systems at the Social Security Administration, I have great interest in your proposed work in studying information sources for software maintenance. In our organization, we have approximately 2,800 systems personnel of which approximately 400 are software maintainers working on literally hundreds of systems ranging from legacy batch COBOL to on-line CICS applications. Obviously, software maintenance is a big part of our business and a major source of cost.

As I understand your proposal, the participation of real-world software maintenance organizations is crucial to the research. One of the first steps in your proposal is to conduct a survey to collect information about maintenance practices (in particular how maintainers gather information about the systems they are maintaining) from a number of different organizations. Another early step is an in-depth but small study of a few maintainers in one organization, closely tracking their information gathering activities over a period of time. I believe that SSA could become involved in either or possibly both of these studies if they demonstrate value to the organization. This may lay the groundwork for further participation in later stages of your proposed line of research.

I look forward to our potential collaboration in the near future.

Sincerely,

Ron Raborg

Ron Raborg
Office of Systems  
Social Security Administration  
6401 Security Blvd.  
Baltimore, Md. 21235  
July 21, 1999