SOFTWARE OUTSOURCING:

DECISION SUPPORT VIA SOFTWARE PROCESS MODELING

by

Stephen Thomas Roehling

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

ARIZONA STATE UNIVERSITY

December 2000

SOFTWARE OUTSOURCING:

DECISION SUPPORT VIA SOFTWARE PROCESS MODELING

by

Stephen Thomas Roehling

has been approved

December 2000

APPROVED:

_____ , Chair

_____

_____

Supervisory Committee

ACCEPTED:

_____

Department Chair

_____

Dean, Graduate College

ABSTRACT

Business pressures for reduced costs, reduced development schedules, and a shortage of skilled software developers have prompted many software organizations to outsource product development processes or components. Outsourcing can offer several benefits to an organization, including a scalable development staff, externalization of non-core activities, and project schedule reduction. However, a non-zero interaction and support overhead accompanies software outsourcing.

Software process modeling is proposed as a value-added tool to explore and gain insights into the positive and negative impacts of software outsourcing. There are time-sensitive, threshold, and feedback-oriented behaviors inherent to software outsourcing relationships, and software process modeling is especially suited to representing these types of behaviors. A high-level framework and road-map, of sorts, is provided for organizations to more appropriately and effectively leverage modeling support. By understanding outsourcing-specific software process modeling's scope, applicability, roles in organizations, and roles with respect to complimentary tools, organizations can more effectively allocate and leverage modeling resources.

To evaluate first hand the synergistic relationships between software process modeling and outsourcing, including the types of outsourcing-related behaviors described above, research also included the development of a proof of concept software process model. This model concretely illustrates modeling's applicable roles and the decision support to be expected. In terms of its required set of inputs, the model also suggests a required level of organizational commitment.

To my wife, Tammy

ACKNOWLEDGMENTS

I am grateful to my advisor and committee chair, Dr. Jim Collofello, for his tremendous support, guidance, and encouragement. He has been exceptionally responsive, accessible, and patient in support of my research. He also encouraged me to present my research at a professional conference and publish a paper; as a graduate student, these type of experiences were extremely rewarding.

I would also like to thank the rest of my committee, Dr. Dwight Smith-Daniels and Dr. Joseph Urban. Dr. Smith-Daniels provided a unique management perspective, and enthusiastically participated in all aspects of the research. Dr. Urban has been a careful reviewer and enthusiastic committee member from the very start.

I would also like to thank Brian Hermann for his involvement in my research. On several occasions, he provided insightful inputs and careful peer reviews.

Finally, several individuals participated in an evaluation of a prototype software process model. This evaluation involved several hours of commitment from each participant, and was an essential part of the research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1: Background and Motivation

Poorly managed software projects routinely suffer from schedule overruns of one hundred to two hundred percent, with wide degrees of staff performance, poor software quality, customer dissatisfaction, and other problems [10]. To address these types of problems, many processes, methodologies, and tools have been developed to more effectively manage software projects [2, 5, 10]. The undertaking for in-house software managers to deliver software on time and within budget is risky and challenging. Outsourced projects are arguably riskier than in-house ones, since customers have less direct control over project decision making. Software outsourcing participants may also benefit from carefully developed processes, methodologies, and tools.

There are synergistic relationships between the capabilities of software process modeling and the interactions and feedback inherent to software outsourcing relationships; therefore, organizations stand to benefit from the integration of software process modeling into their suite of project management tools. For example, over the life of a simulated project, a software process model can accurately represent the notion of interaction and support overhead between in-house and vendor organizations. Moreover, where outsourcing provides a type of pressure relief valve for in-house organizations, software process modeling can represent work backlog or schedule pressure thresholds as a trigger for a part-time outsourcing need, or as a mechanism to preempt in-house staff to assist with an outsourced component. Finally, software process modeling can represent the cascading effect of rework originating from an outsourced component, such that additional in-house rework and support are required. In general, software process modeling effectively captures the types of cause and effect relationships and feedback-oriented behaviors inherent

to software outsourcing.

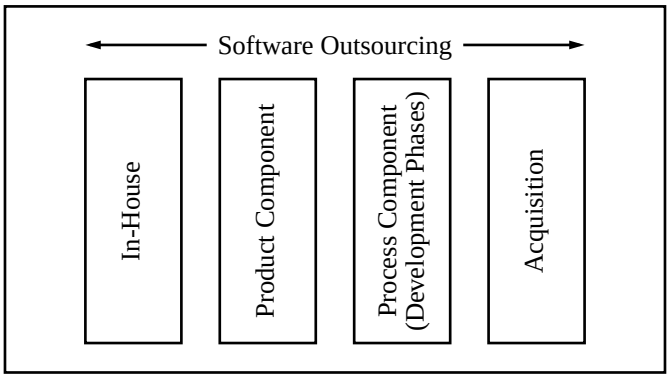
1.1 Software Outsourcing Defined

A comprehensive research effort is underway at Arizona State University to better understand the software outsourcing problem, and build a suite of software tools to provide insight and guidance to software practitioners in decision making roles. Research to date has been focused on software development, engineering, and process activities, rather than information technology functions, such as computer and network support. Research to date has also not included individual sub-contractors or consultants. Within the context of this research effort, software outsourcing was defined as:

- contracting with an external organization to develop one or more product components; or
- contracting with an external organization to assist with one or more process components, such as testing or coding.

Figure 1 illustrates how total in-house development and total acquisition are included in this definition and represent the boundaries of software development outsourcing.

Figure 1: Dimensions of Software Outsourcing



Nearly every software acquisition involves some project oversight, tracking, and relationship management, just like typical product or process component outsourcing. At the other end of the spectrum, in-house efforts frequently integrate commercial product components, which is like projects where part of the product development is outsourced. Software outsourcing is in the middle of the spectrum presented in Figure 1, where to some extent, development and process responsibilities are shared between an in-house and one or more external organizations.

1.2 Why Develop an Outsourcing-specific Model?

To some extent, software process models developed for in-house software projects may suffice for modeling outsourced projects, since outsourced and in-house projects may have common characteristics. For example, software process models tailored to internal processes could be run several times with each prospective vendor's expected productivity and cost drivers. However, software process models tailored to internal processes do not emphasize some unique overhead and collaborations between shared in-house and out-sourced efforts; for example:

- Outsourcing may be used as a strategy to provide a “pressure relief valve” for a core internal development staff. For example, in the months leading up to a major product release, the internal development staff may wish to focus exclusively on new development work, but, to the maximum extent possible, outsource maintenance efforts for an existing release.

- An organization may wish to outsource an entire development phase, such as testing or maintenance. However, schedule pressure may preempt in-house staff from their normal duties to assist the outsourced team. An outsourcing-specific model can help to identify time-frames and scenarios where the in-house staff might be preempted, and to develop sensible policies and thresholds for diverting in-house staff.

- A vendor and in-house organization may have incompatible defect tracking tools. With respect to a total in-house effort, additional delays are introduced into the maintenance fix process, because the in-house organization must manually synchronize its defect tracking system with the vendor.

- Rework generated from an outsourced effort may increase in-house support efforts. A model can represent the feedback and cascading effect of rework on an in-house organization, such that rework attributed to an outsourced vendor can be factored into project planning and decision making.

- Outsourced efforts require a non-zero support effort from an in-house organization. A model can illustrate how an in-house support commitment must scale over time to support an outsourced component.

Research into outsourcing-specific modeling is worthwhile, since it maps out and establishes the extent to which software process modeling can be leveraged with respect to outsourcing-specific project behaviors, goals, and productivity drivers. To the extent that a particular behavior or goal is minimally or not dependent upon outsourcing-specific project parameters, there is no reason existing models cannot be leveraged in a value-added manner.

1.3 Simulation Modeling and Software Outsourcing

In general, the research was first motivated by the recognition of a close relation-

ship between software outsourcing-related processes and concerns, and simulation and modeling capabilities which can effectively represent these processes and concerns. In other words, there is a close mapping between simulation modeling's capabilities and outsourcing-specific processes and concerns, such that organizations can potentially leverage and benefit from modeling support. In this regard, Table 1 summarizes different simulation modeling characteristics, and explains how each can benefit software outsourcing relationship management.

Table 1: Simulation Modeling Characteristics and Outsourcing

Simulation Modeling Characteristic	Benefit to Software Outsourcing
Simulation allows for the study of a system with a long time frame in compressed time [8].	Outsourcing decisions are time-constrained, but simulation models allow customers to make insightful observations in minutes, rather than weeks or months.
Simulation allows for more control over experimental conditions than would be allowed from experimenting with the system itself [8]	• Customers and vendors can cooperatively experiment with what if development scenarios, and select one with the desired characteristics. • Key outsourcing-related planning, such as selecting a vendor, or choosing which components to outsource, occur *before* a project actually starts.In terms of modeling, the system, or software project, would not be available for experimentation.
Explicit, logical, and precise assumptions must be made in order to build a simulation model [8].	• In support of a modeling effort, customers will be further motivated to provide explicit assumptions about the software's requirements, expected costs, and expected schedule. • Where measurement activities are shown to directly benefit the modeling effort (i.e., where collected metrics serve as inputs to the model), organizations may be further motivated to carefully measure development efforts.
Simulations can present complex information in an intuitive, easily digestible manner.	Non-technical outsourcing stake-holders, such as managers and contract officers, can readily gauge the effects of outsourcing decisions.
Dynamic simulation models can model a system over time.	Many aspects of outsourcing relationships are time-sensitive, including workloads and learning curves of vendors' staffs.

Another way to think about the relationship between simulation modeling and outsourcing relationships is to summarize modeling's support for several outsourcing-related

activities. To this end, Table 2 summarizes the support simulation modeling can provide for several outsourcing related activities.

Table 2: Simulation Support for Outsourcing-Related Activities

Outsourcing-Related Activity	Potential Simulation Modeling Support
Vendor Selection	Highlight outsourcing goals which are sensitive to vendors' productivity-related model inputs, such that vendors with the most attractive set of drivers can be selected.
Risk Management	If a vendor were to go out of business, how quickly could a project recover by bringing the outsourced component or process back in-house?
Negotiations	Provide a greater understanding as to which vendor productivity drivers have the greatest impact on project outcomes. Both sides of an outsourcing relationship can then more carefully negotiate for key productivity drivers.
Proposal Development	Where certain model inputs, such as team size, are negotiable, a vendor could run a model with several sets of inputs.

1.4 Modeling Limitations

While Table 1 and Table 2 describe the extent to which modeling can represent and support outsourcing-specific relationship management, there are limitations to modeling-based decision support. In particular, simulation models can only represent quantifiable aspects of outsourcing relationships, such as schedule, cost, and productivity; where models are inappropriate or incomplete, additional project management tools may be required (see "Complementary Tools" on page 23). Moreover, where an existing model is not readily available or requires extensive customization, a modeling effort introduces additional overhead to a project.

1.5 Motivational Example

During the establishment of an outsourcing relationship, vendors' proposals are a pivotal negotiation and communication tool. In this regard, Table 3 below considers ways simulation modeling can directly or indirectly benefit the proposal development and vendor selection process.

Table 3: Modeling Integration into the Vendor Selection Process

Representative Proposal Questions	Potential Simulation Modeling Benefits
How much will the work cost?	Based upon lower and upper bound cost estimates, vendors may carefully adjust their bids.
What will the software do?	Models require explicit and precise assumptions, which in turn requires explicit and precise customer requirements.
How many people will work on the project?	Increased staffing levels increase communication overhead, which may result in lower per-staff-member productivity.
What is the experience level of staff in the target application domain?	A model can represent the time-sensitive nature of staff experience. Assuming the vendor will learn the technology as the project progresses, a customer may choose a less experienced, less expensive vendor. A model can also be used to better understand bait and switch risk scenarios, where a vendor proposes a team with a different experience level than the one that actually works on the project.
Where will the work be performed?	Assumptions about communication overhead can be factored into model inputs, such that resulting outputs may be useful to both customers and vendors; for example: • A customer may negotiate for a lower price, to recover the cost differential between an off-site development arrangement and an in-house effort. • Where the model demonstrates the indirect effects of communication overhead on cost, a vendor might negotiate for effective communication channels, such as a high-speed dedicated data link, or regular in-person meetings.
What is the job's size (function points, lines of code)?	Where an assumed job size is an input to the model, customers can observe non-linear direct or indirect effects on software outsourcing related variables. These observations will motivate customers to be more explicit and precise regarding project requirements, and prioritize these requirements accordingly.

Table 3: Modeling Integration into the Vendor Selection Process

Representative Proposal Questions	Potential Simulation Modeling Benefits
Change Rates	• Determine realistic penalties and rewards for schedule changes. • Factor the rising cost of outsourced labor over time.

A written proposal compels a vendor to explicitly quantify preliminary estimates of development costs, schedule, and staffing requirements [9]. After the vendor has prepared an initial written proposal, a model will give both customers and vendors a collaborative tool to better understand, and more carefully negotiate important project variables. Furthermore, a written proposal can only statically represent a small set of development choices. A simulation model can represent and communicate an endless number of outsourcing scenarios.

1.6 Research Goals and Contributions

The main research goal was to substantiate a claim that software process modeling, represented by a dynamic simulation model, can be a value-added outsourcing decision support tool [11]. Substantiating this claim gives software organizations justification to leverage simulation modeling in support of their outsourcing efforts. The primary research contributions are a rationale and framework for software organizations to properly leverage dynamic simulation modeling, and a proof of concept outsourcing model to demonstrate an actual model can fill many of its proposed roles.

1.7 Research Activities

A research effort was conducted to map out the relationships between modeling

and software outsourcing relationships, describe how these capabilities can be integrated into software organizations, and to build a proof-of concept model which demonstrate first hand modeling's expected scope and benefits.

Towards realizing the research goals and contributions described above, several approaches were considered. In particular, working with several commercial software organizations to develop and validate a commercially deployable simulation model would have been ideal. A primary concern with this approach is the proprietary nature of software outsourcing relationships; for example, cost and schedule estimations, or vendor selection criteria are arguably sensitive. Even if open access was granted to several projects, it is unlikely the results could be openly discussed and analyzed in an academic setting. In terms of a manageable research effort, the time frames involved in following several projects through their entire lifecycle would also have been prohibitive. In consideration of the constraints described above, the following approach was settled upon:

- perform a careful analysis of the project conditions, roles, and expected benefits to be gained by modeling the outsourcing problem;
- implement a "proof of concept" outsourcing model, which represents a single outsourcing type, and illustrates a proposed set of outsourcing-specific behaviors other modelers can reuse or leverage; and
- evaluate the model with respect to the conditions, roles, and expected benefits where the model shows promise of being a value-added tool.

With the approach described above, a central theme of the research was establishing the feasibility, scope, and applicability of outsourcing-related models. The work therefore serves as a useful reference for other modelers to determine when and what to model,

or when and what not to model. Moreover, where organizations may wish to leverage many complimentary tools and capabilities in support of outsourcing-related decision support (e.g., spreadsheet models, expert systems, scheduling tools, and metrics collection), research efforts have defined software process modeling's unique contribution with respect to these other tools and capabilities.

Although the research effort did not include studies of actual software projects, the work can be viewed as a proposed roadmap, or framework for actual projects to incorporate software process modeling. An emphasis upon scope, feasibility, and applicability provides a potential advantage to project managers, in terms of making judicious, well-informed planning and resource allocation decisions.

1.8 Summary

There is a synergistic relationship between the inherent capabilities of software process modeling, and some key outsourcing relationship behaviors, planning activities, and decision making.

Where an outsourcing software process model is implemented using dynamic simulation modeling tools, simulation constructs including feedback, information flows, and thresholds map to outsourcing-related behaviors, such as a vendors' rework impacting in-house support overhead, or work backlog thresholds triggering an in-house organization to outsource as a pressure relief valve. Simulation modeling also captures the time-sensitive aspects of outsourcing relationships, such as the learning curves of vendors' staff, or work-loads varying over time.

Software process modeling is also an appropriate tool, given the time-frames and constraints associated with outsourcing relationship planning and decision making. In particular, software process modeling can be leveraged during initial planning activities such as vendor selection or proposal development, when metrics for the actual project itself are still unavailable. Using some up-front assumptions, or ranges of assumptions about project cost, schedule, and staff, a software process model can highlight overheads and in-house support functions which are the least and most sensitive to these assumptions. With this information, an organization considering outsourcing can more appropriately focus its vendor selection criteria, risk management resources, selection of project management tools, and in-house staff. Since modeling efforts require explicit and precise input assumptions, the use of software process modeling is also indirectly linked to positive activities, such as explicit requirements specification, or keeping metrics databases for past projects.

There are many different types of software outsourcing, such as the outsourcing of software development, or the outsourcing of software configuration management. Similarly, a researcher could study outsourcing relationships from many different perspectives, including risk management and resource planning. To keep the research effort focused and manageable, but concretely demonstrate how software process modeling can be a value-added project management tool, research focused on a single outsourcing type and a single perspective. Along these lines, a proof of concept model illustrates modeling's applicable scope, representations of outsourcing-specific processes, benefits to be expected, and required organizational commitments.

Having presented the background, motivation, and rationale for modeling outsourcing relationships, Chapter 2 discusses ways software organizations can integrate modeling into their outsourcing relationship management processes. Chapter 3 presents a prototype, or proof-of-concept, model for a single outsourcing type; this prototype serves as a concrete example of a model, and provides the basis for an evaluation in Chapter 4. Chapter 4 presents the results of an evaluation of the model, including independent feedback from a group of software professionals. Chapter 5 summarizes and reinforces the research contributions and describes additional research efforts which may build upon existing work.

Chapter 2: Integrating Modeling into the Software Outsourcing Process

With the ultimate goal of software organizations leveraging process modeling for direct and indirect benefits, addressing the ways that outsourcing-specific software process modeling can be leveraged by and integrated into software organizations is important. In this regard, the applicable scope of modeling efforts is also important. With an applicable scope defined, the extent to which key outsourcing-related management questions can be addressed via modeling outputs, or through sensitivity analyses, can be considered. Where software process modeling is just a single tool, defining an applicable scope also helps to define the roles of complementary tools.

2.1 Scope and Applicability

Several factors contribute to the applicable scope of modeling efforts. First, there is no need to model processes which can be more easily captured by other tools. Secondly, since software process models require quantitative inputs, the scope of any modeling effort is constrained by the types of available quantitative information, either in the form of metrics or assumptions from project managers.

With an understanding of the applicable scope for outsourcing-related modeling efforts, project managers can plan more effectively up-front, and make judicious use of modeling resources. For example, modelers can choose what to include and what to exclude from their models. Or, if modeling is deemed inappropriate, project managers can direct resources towards project management tools which more effectively address the problem at hand.

Simulation models, and system dynamic models, in particular, are especially use-

ful tools for representing time-sensitive and feedback-driven processes. Modeling efforts are also constrained by the types of quantitative outsourcing-specific information which may be collected within a software organization, then represented in a continuous, dynamic, software simulation model. Table 4 below lists the types, or classes, of metrics which may be collected during an outsourcing relationship [14].

Table 4: Outsourcing Metrics Classes

Finance & Budget
Customer Satisfaction
Work Product Delivery
Quality
Time and Schedule
Business Value
Operational Service Levels
Human Resources
Productivity

Outsourcing-related metrics hint at the types of quantitative information which may be collected or estimated for modeling efforts, or serve as outsourcing performance gauges. Using the metric classes listed in Table 4 as a guide, appropriate modeling efforts could emphasize productivity, staffing, rework (quality), cost, and schedule; in this regard, Table 5 lists several metrics classes, and describes several modeling components which are closely related to these metrics classes.

Table 5: Quantitative Modeling Emphasis Areas

Modeling Emphasis Area	Potential Modeling Components
Finance & Budget	• Labor rate increases over time. • Outsourcing-specific compensation systems.
Quality	• Defects generation. • Rework.
Time and Schedule	Schedule performance with respect to productivity assumptions, learning curves, and hiring delays.
Human Resources	• Staff attrition, rehiring delays, bait and switch, learning curves. • Over and under utilization of work-force.
Productivity	• Staff learning curves and experience levels over time. • Collaboration overhead associated with managing an out-sourced effort.
Operational Service Levels	• Work backlogs with respect to time. • Response times to complete customer requests.

2.2 Expected Benefits

An organization may wish to use a modeling tool to better understand its potential to achieve outsourcing goals, such as shortening development schedules, or responding more rapidly to customers' maintenance requests. Where model inputs are correlated with model outputs, and model outputs can gauge whether or not an organization is achieving its goals, it is useful to consider relevant model inputs and outputs with respect to common outsourcing goals. In this regard, Table 6 below lists the top five outsourcing goals, according to Hermann's outsourcing survey [6], and discusses types of model inputs and outputs which may impact or relate to these goals.

Table 6: Outsourcing Goals and Applicable Sensitivity Analyses

Outsourcing Goal (Ranked According to Survey)	Outputs Versus Inputs
1. Obtain particular expertise	• Schedule and cost versus vendor experience level and learning rate. • In-house support and interaction overhead versus vendor experience level and learning rate.
2. Shorten schedule duration	Schedule duration versus the amount of outsourcing, staff diversion thresholds, and experience levels.
3. Improve responsiveness to the customer	• Work backlog versus vendor staff size. • Work completion response times versus vendor staff size. • Work backlogs and response times versus staff diversion thresholds.
4. Add people	• Schedule and cost versus the number of people and hiring delay of additional people. • Required in-house support versus the number of people added.
5. Improve product quality	Number of defects generated and rework required versus vendors' experience levels and learning curves.

Where modeling inputs and outputs are related to outsourcing goals, as shown in Table 6, sensitivity analysis is another important modeling capability. Sensitivity analysis allows a modeler to determine which input parameters have the greatest impact upon model outputs; and, given the relationships between model outputs and outsourcing goals, sensitivity analysis arguably plays an important role in determining which goals are the most sensitive to model inputs. For example, if an organization considers outsourcing to obtain particular expertise, a sensitivity analysis using various assumptions about in-house or outsourced experience levels could be performed to gauge the overall impact on schedule duration. This type of sensitivity analysis promotes more realistic goal setting, and

allows a project manager to more carefully focus energies on the project/model inputs with the greatest impact upon outsourcing goals.

2.3 Modeling Roles within Software Organizations

At this point, modeling is envisioned primarily in a strategic role, where observed trends and insights gained indirectly influence project planning and resource allocation. This type of role is also appropriate to outsourcing relationships, where sub-contractor data is difficult to obtain, or parameters of the relationship are confidential. In a strategic role, modeling could provide the following types of strategic benefits:

- elevate decision makers' sensitivities to the trends, dynamics, and constraints of outsourcing relationships;
- encourage software practitioners to focus resources towards activities most conducive to successful outsourcing, such as outsourcing goals which are most sensitive to outsourcing relationship variables (see Table 6 on page 20);
- give customers the ability to perform "what if" analyses to choose from any modeled outsourcing scenario, rather than the one or two described in vendors' proposals; and
- since models require explicit, logical, and precise inputs, all parties in outsourcing relationships will benefit from estimations and metrics gathered to support modeling efforts.

Furthermore, as shown in Table 7 below, modeling holds promise for both sides of an outsourcing relationship.

Table 7: Outsourcing Participants, Their Objectives and Concerns, and Potential Modeling Benefits.

Party	Objectives and Concerns	Potential Modeling Benefits
Customer (In-house organization)	• Build the right product. • Minimize cost. • Minimize time to market. • Maximize quality. • Stable in-house staffing levels. • Flexible and scalable commitments with vendors.	• Where modeling can demonstrate the most important outsourcing capabilities per the customer's requirements, customers can choose the most qualified vendor. • More development choices - buyers can choose any development scenario. • Commit to more realistic and fair outsourcing relationships.
Vendor	• Maximize profit. • Customer/buyer satisfaction. • Commit to a reasonable development schedules. • Maintain stable staffing levels. • Earn future business.	• Manage non-technical customers' expectations regarding cost and schedule. • Literature suggests feature creep is sometimes a problem in outsourced development [7]. A model could motivate customers to be more explicit and precise about feature requirements. • Commit to more realistic and fair outsourcing relationships. • Input historical project metrics data to a model as a selling point for future business. • Refocus on activities demonstrated to be most important to successful outsourcing (e.g., continuous training).

Organizations will provide varying levels of commitment and access to a modeling effort. For example, if a project is proprietary, an organization may only be able to use an outside modeling expert to provide generic insights. Similarly, if an organization is small, and does not have the resources for dedicated modeling efforts, they may wish to simply gain insights by running pre-existing examples. Where modeling is deemed inappropriate

or has limitations, it is also important to recognize the role of complementary tools. With these points in mind, the following are some suggested strategies organizations to integrate modeling:

(A) At the bottom of a top-down decision support process:

1. A customer meets with an outsourcing domain expert, and describes objectives and concerns.
2. The domain expert uses a qualitative, informational framework, or an expert system, to identify potentially successful outsourcing types, strategies and tactics.
3. The simulation model is customized to represent the customer's outsourcing project characteristics.
4. The domain expert walks the customer through several simulation runs, which corroborates results from step 2.

(B) Training sessions, where the model is used in a stand-alone manner:

1. A customer meets with an outsourcing domain expert, and describes outsourcing goals and constraints.
2. A modeling domain expert chooses a set of pre-modeled outsourcing types to illustrate key properties associated with the customer's outsourcing goals and constraints.
3. The domain expert walks through several simulation runs and analyses, which collectively elevate the customer's sensitivity to benefits and drawbacks of various candidate decisions.

(C) As a complement to other relevant tools and practices, for example:

1. An estimation model, such as COCOMO, is used to determine the software project's expected cost, effort, and schedule.
2. The customer asks the question, "Does it make sense for me to outsource a piece of the development effort?"
3. The customer leverages an outsourcing specific model to more closely understand expected outsourcing related consequences.

2.4 Complementary Tools

The respective roles and contributions of other types of tools to the overall goal of improved outsourcing relationship management are important. An understanding of the expected roles and contributions of other tools further defines the useful scope and appli-

cability of outsourcing-specific modeling, as described in “Scope and Applicability”, on page 16; in other words, where complementary tools provide robust support, there is no need to pursue a modeling solution. Leveraging complementary tools allows outsourcing-specific modeling to be used in a broader context, as described in “Modeling Roles within Software Organizations”, on page 21. With these points in mind, Table 8 below lists several types of project management tools, and summarizes their relationship, role, and contribution with respect to outsourcing-specific modeling.

Table 8 below focuses on software tools, but to a large extent, high-level, qualitative outsourcing issues can also be addressed through management efforts, such as due diligence with different vendors.

Table 8: Complementary Project Management Tools

Software Tool	Relationship to Modeling
Software Estimation Tools	• When actual metrics are unavailable, software estimation tools can be used to provide realistic model inputs. • Some software estimation tools, such as Construx Estimate (see http://www.construx.com), leverage a hybrid of estimation methodologies, such as simulation, and a statistical analysis of past metrics data. Outsourcing-specific features could feasibly be incorporated as an extension to such hybrid software estimation tools.
Historical Project Metrics Database.	• During up-front planning, vendors can leverage metrics from past projects as model inputs. • During up-front planning, customers can leverage metrics from past projects as benchmarks for different vendors.
Expert Systems	Expert systems, such as the one developed by Hermann (see [5]), are more capable of addressing qualitative aspects of outsourcing. These types of tools may be appropriate for determining whether or not successful outsourcing is indicated up-front, such that a modeling effort may or may not be called for.
Spreadsheet Models	For calculations not involving time or feedback, spreadsheet models often suffice.

2.5 Summary

Software process modeling has a role in addressing schedule, cost, and staffing aspects of outsourcing relationships. An approach to establish these roles is to observe that modeling inputs are dependent upon quantitative metrics or assumptions, so available metrics and assumptions determine the applicable scope of modeling efforts as a whole. Furthermore, individual modeling outputs can be linked to outsourcing goals, such as schedule reduction, improved quality, or adding experience to an in-house organization; then, using either a fixed set of inputs, or a range of inputs and sensitivity analyses, software outsourcing goals can be analyzed in terms of their linked modeling outputs. This

type of analysis can be leveraged in a strategic context, such as encouraging organizations to focus resources towards productivity drivers with the greater impact on successful outsourcing. This type of decision support allows managers to gain insights into the outsourcing relationship, and manage and prioritize outsourcing relationship goals shown to be sensitive to or impacted by outsourcing parameters. This method also accommodates the confidential nature of outsourcing relationships, where sub-contractor data may not available, but a range of possible inputs can be given to a model.

Chapter 3: Outsourcing Decision Support Model

Chapters1 and 2 provided general motivation, rationale, and a framework for organizations to leverage software process modeling, but the research effort also included the development of a concrete, proof-of-concept process simulation model. For research, discussion, and analysis, a concrete implementation of a single outsourcing type was needed. Chapter 2 presented a notional view of software process modeling as a useful and value-added tool to help manage software outsourcing relationships, but the development of a model concretely illustrates the types of outsourcing-specific decision support to be expected.

Important insights and knowledge were also gained during the implementation process. In particular, after some experimentation, a certain functional decomposition of modeling components emerged. These components have a particular mapping, relevance and importance to the outsourcing problem, which can be analyzed and discussed.

Model development consisted of an analysis to define a realistic software project scenario, followed by the implementation of a software process model to represent this project scenario. Proof-of-concept model development was restricted to maintenance outsourcing, a single outsourcing type and microcosm of outsourcing as a whole. The model provided a pivotal discussion and research example. Implementation of a single outsourcing type also promoted a focused, manageable research effort.

Model development also included the establishment of a concrete, workable set of model inputs and outputs. Understanding the model's input requirements and output capabilities conveys the level of organizational commitment required to leverage a model; for

example, where the model has certain input requirements, an organization must either collect metrics or estimate specific values for those inputs. Secondly, where particular model outputs are aligned with outsourcing-specific goals (see “Scope and Applicability”, on page 16), and inputs are correlated with outputs, the model can be evaluated for completeness and feasibility; a strategy for formulating inputs and analyzing outputs to address outsourcing-specific problems can also be established.

3.1 Analysis of Maintenance Outsourcing

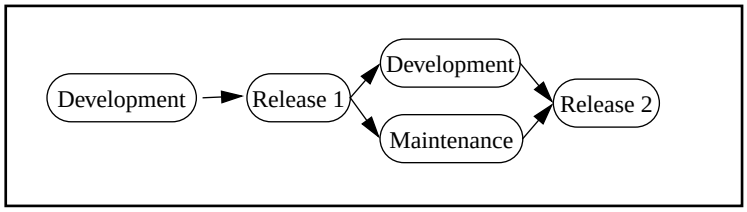
Prior to model implementation, an analysis of the chosen outsourcing type motivated particular features in the model and provided some early insights into the benefits that the model could provide.

Selecting an appropriate outsourcing project type was a major consideration that factored into choosing maintenance outsourcing. For example, modeling the outsourcing of proprietary software development would not be appropriate, since the outsourcing of proprietary development efforts is often contra-indicated [14]. For several reasons, however, outsourcing maintenance efforts makes sense under certain conditions; in particular:

- For maintenance outsourcing, a version of the product has already been released, such that the code base would be less proprietary.
- Moreover, a vendor would be working on fixes and enhancements to existing features, rather than ground-breaking new development; the in-house staff would be expected to work on maintenance requests which are proprietary in nature, and provide some direction as to which maintenance requests are assigned to the vendor.

- Outsourcing could also give a core internal development staff the opportunity to work on new development in parallel with maintenance activities, such that schedule reduction for the second release of the product might be possible. Figure 2 below depicts maintenance outsourcing, where a version of a product has been released, and maintenance and new development for the second release of the product occur in parallel.

Figure 2: Software Development and Maintenance Lifecycle



Where “Why Develop an Outsourcing-specific Model?”, on page 3, proposes several benefits to modeling outsourcing relationships, a goal in choosing an outsourcing type to model was to exemplify many of these proposed benefits. Versus a total acquisition, or total in-house maintenance effort, maintenance outsourcing scenarios can involve several interesting interactions and feedback between an in-house organization and outsourced vendor; therefore, a model which represents maintenance outsourcing interactions would have several differentiating features and capabilities; for example:

- To remain responsive to the software’s customer base, the in-house organization may be required at peak times to help the outsourced vendor with maintenance requests.
- Some maintenance requests will require rework, such that the in-house organization will need to provide additional support and integration overhead. A model can distinguish the amount of in-house support effort originating from vendors’ rework.
- There could be front-end overhead associated with formally communicating, translating, and routing a maintenance request to the vendor. Maintenance requests which are proprietary in nature, for example, could be routed to the in-house organization.
- There could be back-end support overhead associated with accepting a completed maintenance request, and integrating the fix into the main development code base (e.g., acceptance tests to ensure the fix was done correctly). Since the vendor may work off-site, the vendor may have an incompatible source code version control system, such that integration of maintenance fixes could require a manual integration effort on behalf of the in-house staff.

- Rather than salaries common to full-time staff, the vendor may be compensated on a per maintenance request, or fixed hourly basis. Moreover, the vendor may work part-time, and not be compensated for idle time, or time spent on other projects. Therefore, a model must have the capability to model different compensation systems inherent to outsourcing relationships.

An actual maintenance outsourcing relationship would be expected to have some up-front assumptions and limitations which could impact the feasibility of a modeling effort. These assumptions and limitations are a function of project type, and the respective commitment and process maturity of the in-house and vendor organizations. The following are a minimum set of assumptions before beginning a modeling effort:

- In general, an outside vendor may be used for some or all maintenance efforts. For example, it is assumed that general management support is available for outsourcing, and the project is not overly proprietary.
- After the first release of the product, development and maintenance tasks occur in parallel.
- Based upon the number of backlogged maintenance requests, in-house development talent is diverted from new development to handle maintenance requests.
- Based upon historical data, the stream of new maintenance requests over time can be approximated. The in-house organization, for example, would be expected to have a defect tracking system which can generate this historical information.

Subject to these interactions, feedback, and project scenario assumptions described above, Table 9 below describes the management support that a software process model can provide with respect to outsourcing goals from Hermann’s outsourcing survey:

Table 9: Outsourcing Goals and Maintenance Outsourcing Model Capabilities

Outsourcing Goal (Ranked According to Survey Results)	Related Model Capabilities
1. Obtain particular expertise	A model can factor the experience and learning curves of different potential vendors.
2. Shorten schedule duration	Where shortening the schedule duration for a product's second release is an outsourcing goal, a model can determine when and how much outsourcing makes sense.
3. Improve responsiveness to the customer	Modeling the backlog of maintenance requests over time serves as an indirect measure of responsiveness to the customer.
4. Add people	A model can capture the in-house support overhead that comes with adding additional people.
5. Improve product quality	The amount of rework generated by the outsourced staff serves as an indirect quality measure.

Chapter 2 describes a general approach for linking modeling outputs to specific outsourcing goals, such that a sensitivity analysis with a range of modeling inputs can be used to identify outsourcing goals sensitive to vendor productivity drivers. With an actual model implementation, particular model outputs can be linked with outsourcing goals. In this regard, by describing how particular maintenance outsourcing modeling capabilities, and hence related outputs and analyses, are related to individual outsourcing goals, Table 9 concretely illustrates the analysis approach proposed in Chapter 2.

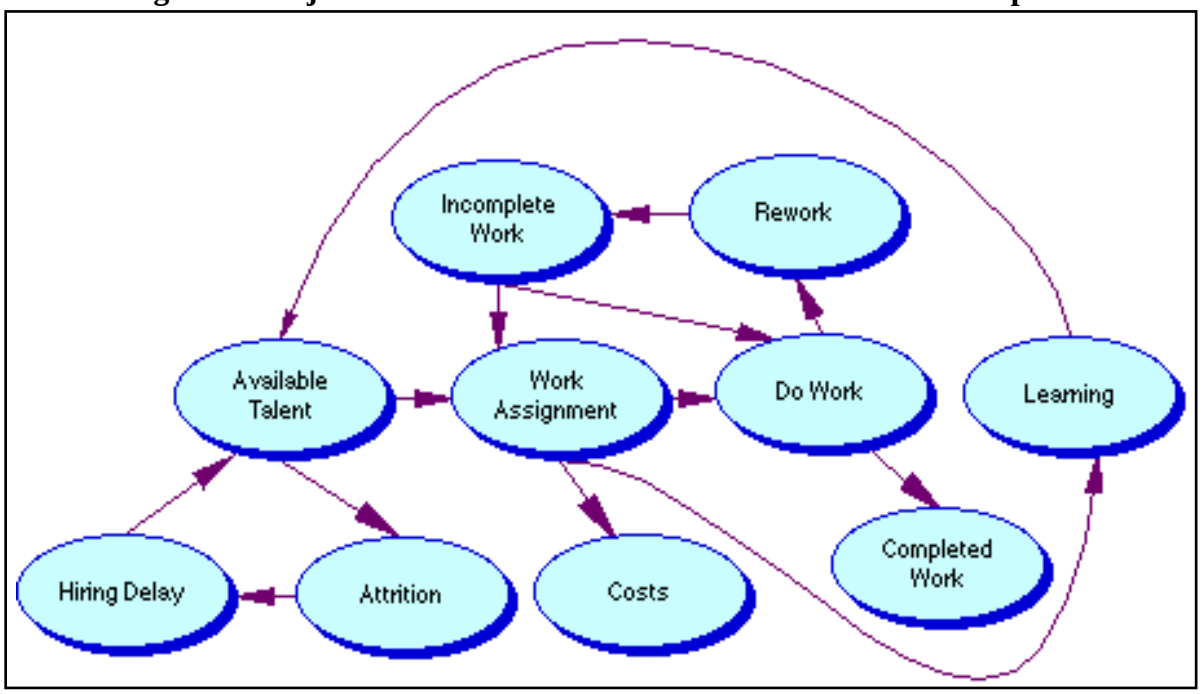
3.2 Model Overview

At its core, the model is a basic software project simulator, where people staff the project, tasks are worked on with a certain productivity level, rework is generated, and

costs accrue. In this regard, Figure 3 below illustrates the basic information flow and feedback within the model.

In addition to the basic project simulator depicted in Figure 3 below, the model has some distinctive, outsourcing-related features and capabilities. Most importantly, the model simultaneously represents both an in-house and vendor organization, such that interactions, dependencies, shared work efforts, support overhead, and information flow between two organizations can be represented.

Figure 3: Project Simulator Information Flow and Feedback Loops



The model also can be instrumented with particular inputs and outputs, such that outsourcing-specific analyses are possible. For each component, required support for outsourcing scenarios also motivated implementation decision-making; for example, the model supports part-time staffing and hourly compensation, since some vendors may bill hourly for the work they perform, and only work during peak times, serving as a pressure relief valve for in-house organizations (e.g., provide testing help at the end of a project).

Schedule (i.e., completed work) and cost serve as the model's primary outputs. Therefore, schedule and cost serve as the primary metrics for analyzing the results of one simulation run versus another. In the model's current implementation, schedule and cost do not serve as decision variables to trigger more or less outsourcing, but using cost and schedule as decision variables is an area of possible future research.

3.3 Model Implementation

In general, the model can be classified as a dynamic, deterministic, and continuous simulation model. But, research efforts were partially motivated by similar modeling efforts to represent software development processes [1]. Systems dynamic modeling is a type of simulation modeling tool that software project managers can use to gauge the effects of various project management decisions and policies [3, 4, 13].

The *Extend* simulation tool was used for model implementation (see <http://www.imaginethatinc.com>). Although *Extend* is not a system dynamics modeling package per-se, it does allow modelers to construct continuous, system dynamics models with feedback, stocks, and flows. In particular, feedback was useful for representing rework, and the accumulation of available work capacity where learning is involved. Stocks represent pools of talent, incomplete and completed work, and cost accrual. Finally, flows capture the flow of rework, work items, and the routing of available work capacity to handle work items.

3.4 Staffing

Talent pools represent available staff for both in-house and vendor organizations. In other words, collective talent, rather than individual staff members are represented. There are separate talent pools for both the in-house and vendor organizations, and the relative experience levels within those organizations. The following attributes are associated with each talent pool:

- learning rates - duration, not including overtime or idle time required to advance to

the next experience level;

- attrition rate - percentage of talent which turns over each year;
- hiring delay, in project weeks, to hire new talent, or replace talent which leaves due to attrition;
- initial talent immediately available at the beginning of the project phase;
- needed talent at the beginning of the project phase; and
- overtime availability.

Talent is available for one or more tasks. For each talent pool, and for each task, there are several input variables:

- productivity - for each experience level, the amount of time it takes to complete a work item; and
- rework generation rate - number of new work items (e.g., maintenance requests) generated for each completed one.

For both in-house and outsourced talent, a learning rate is associated with each experience level. However, if based upon work assignment, some talent remains idle, the learning rate will decrease. Similarly, if some talent works overtime, then the learning rate will increase. The model takes into account idle time and overtime in speeding up or slowing down the learning of in-house and outsourced talent.

The model supports an “idle time learning rate” input parameter. This parameter controls the learning rate during idle time. This parameter may approach 1 for the in-house talent, since this talent is immersed in an environment doing complimentary tasks, where applicable learning occurs even during idle time. However, this parameter may approach 0 for off-site, outsourced talent, where applicable learning does not occur, because the out-

sourced talent is assumed to be working on dissimilar tasks during idle time.

3.5 Work Flow and Assignment

The model represents separate in-house and outsourced teams interacting to complete project work. In this regard, the model includes appropriate work input, assignment, and scheduling capabilities.

A user defined work input distribution fills an incomplete work pool, which is drawn from when talent is assigned to a task and the work is completed. For each simulation time step, the amount of new work can be specified. Observe that “requirements creep”, or a mid-project change of scope, can be supported by inputting new work during later simulation time steps.

Units for work are generic, but for a particular set of inputs, and depending upon the type of outsourcing, units of work could represent such things as maintenance requests, function points, or test case executions. The current model represents work in the form of maintenance requests, and assumes all maintenance requests are of equal complexity. Future revisions of the model may allow a user-specified distribution of maintenance request complexities. With such a feature, the work assignment policy for maintenance could factor complexity, and include rules like, “outsource the easy maintenance requests first”.

Several types of work assignment rules support task prioritizing, parallel tasks, and tasks requiring simultaneous efforts. These rules can be combined to implement a customized work assignment policy for a particular type of outsourcing. For example, within the

current model, work is assigned such that work started on existing maintenance requests will be finished before work on new maintenance requests begins. Based upon work assignment, a certain amount of work capacity is available for each task. This work capacity is used to draw from the pool of incomplete work, then generate rework and completed work. In the current model, the following work assignment policy represents outsourced maintenance:

1. Assign some or all of *both* the in-house and outsourced teams to simultaneously work on back-end communication overhead associated with the outsourced team completing maintenance requests (e.g., the work associated with acceptance testing).

2. Assign in-house and outsourced talent to work in parallel on maintenance requests. A backlog threshold can be set, whereby in-house talent will only be assigned to work on maintenance requests if the threshold is exceeded.

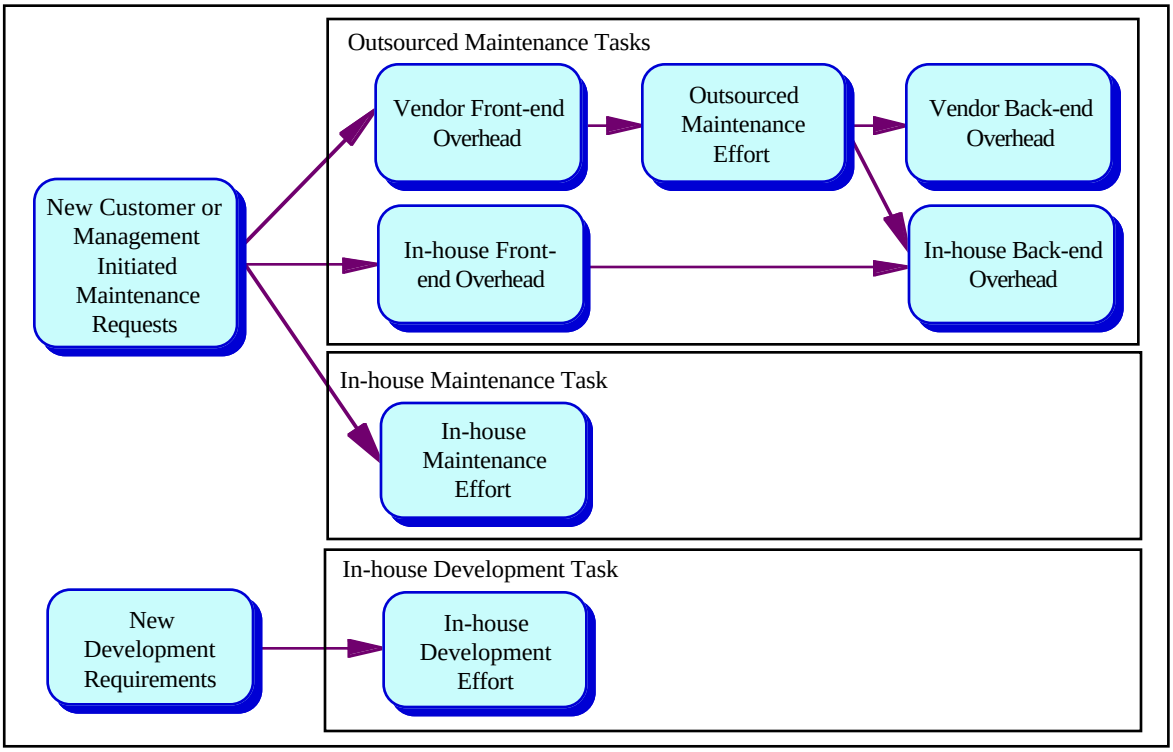
3. Assign both in-house and outsourced talent to work on front-end communication overhead work associated with outsourcing maintenance requests. For maintenance, this work could represent extracting the maintenance request from the problem tracking system, then formally communicating the request to the outsourced team.

4. In-house talent not already assigned is assigned to work on new development.

5. Any remaining in-house talent not assigned to new development is assigned to work on any remaining maintenance requests.

In addition to an input work distribution, and a work assignment policy, the model also implements schedule dependencies between different tasks. Figure 4 below shows the schedule dependencies for outsourced maintenance tasks. These schedule dependencies influence staff utilization within the model, and ultimately affect schedule durations in the model's outputs. New maintenance requests are either routed to the in-house or outsourced team, subject to the work assignment policy described above.

Figure 4: Schedule Dependencies between Tasks



3.6 Front-end and Back-end Overhead

In terms of outsourcing-specific work efforts, a representation of front-end and back-end overhead is an important model capability. Front-end or back-end overhead represent communication and support interactions between the in-house and outsourced organization. For example, front-end overhead to outsource maintenance requests could include one or more of the following:

- based upon the nature and complexity of the maintenance request, the overhead of routing maintenance requests to either the in-house organization or vendor;
- formal software specifications needed to complete the work under a legally binding contract;
- communication and language overhead between the in-house organization and vendor; and

- where the in-house organization and vendor have incompatible or geographically distributed software tools, the manual labor associated with extracting and synchronizing information between the in-house organization's and vendor's tools.

Similarly, back-end overhead could include the following types of overhead:

- careful acceptance testing to ensure completed work is done to specification, meets quality standards, and meets contractual obligations; and
- where the in-house organization and vendor have incompatible or geographically distributed software tools, manual labor associated with extracting and synchronizing information between the in-house organization's and vendor's tools.

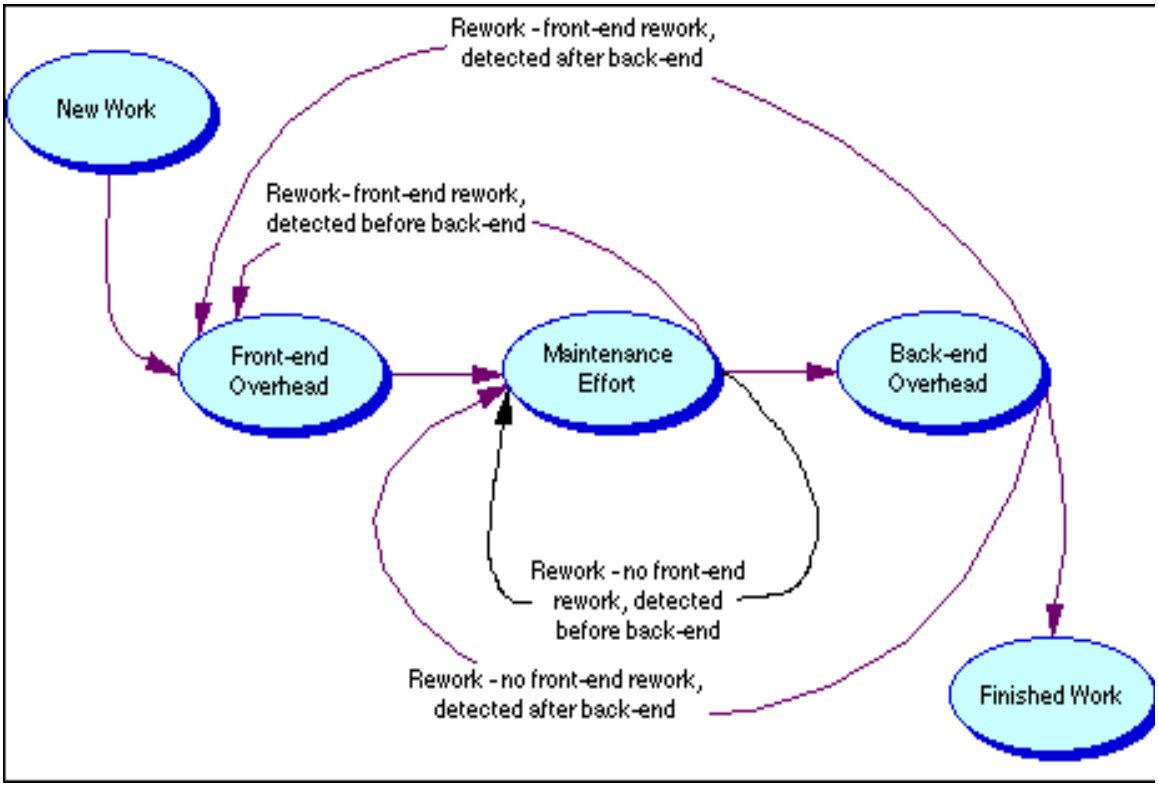
The model assumes front-end and back-end overhead require different levels of effort on behalf of in-house and outsourced talent. For example, as part of back-end overhead, it could take more effort for the vendor to prepare for a formal acceptance test than it would take an in-house representative to inspect test results.

3.7 Rework

Rework is an important model feature, since rework generated from outsourced efforts also requires additional in-house support. Rework is a means to gauge the sensitivity of overall support overhead, schedule, and cost to the relative work quality of different vendors.

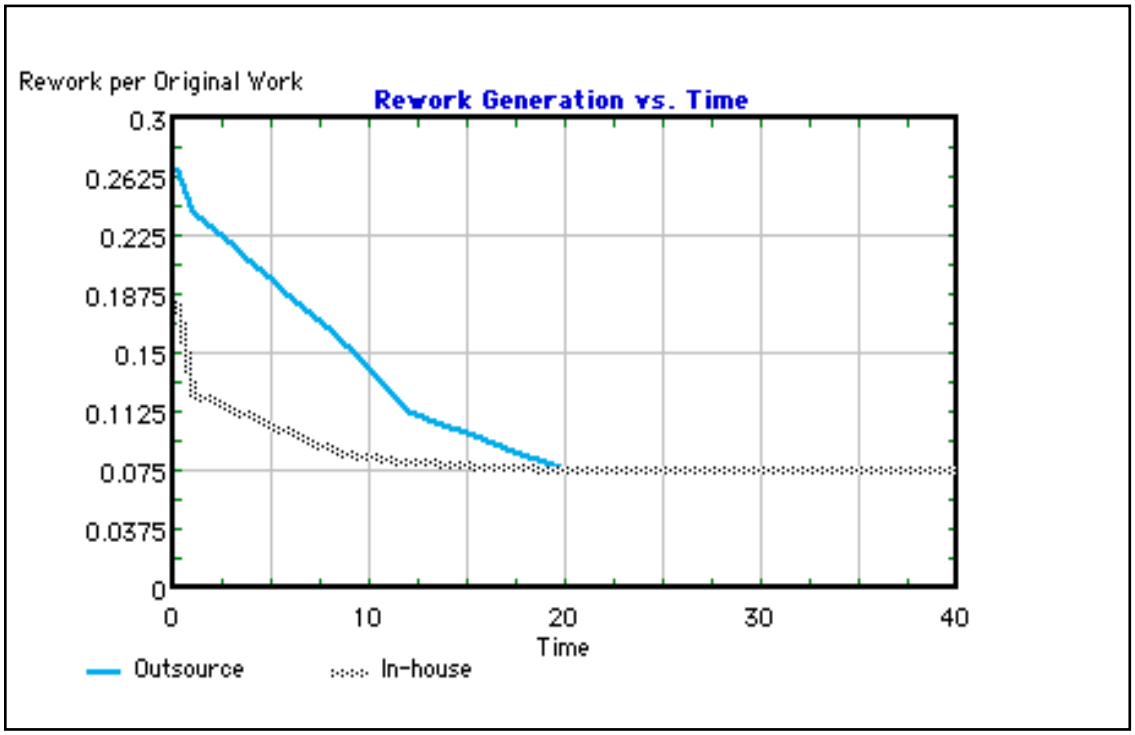
As shown in Figure 5 below, the model currently represents four possible rework types for outsourced maintenance efforts. The assumption is that efforts required to rework a maintenance request may or may not require additional front-end overhead, but all rework is assumed to require some additional back-end overhead. The model also represents *when* rework is detected: either before or after back-end overhead effort.

Figure 5: Rework Flow and Feedback for Outsourced Maintenance Efforts



The model assumes the effort associated with rework will be less than or equal to the original effort. In terms of model inputs, the effort associated with rework is expressed as a fraction of the original effort. Figure 6 below is an example plot showing rework generation rates for in-house and outsourced talent over time. In general, less rework is generated as more and more organizational learning takes place.

Figure 6: Rework vs. Time and Experience



3.8 Costs

Costs are a function of work assignment and talent utilization described above. The model only includes labor related costs, since labor costs can vary with time. However, where staff is utilized in different outsourcing related support and development tasks, such as rework, front-end overhead, maintenance, or development, cost outputs can be separated. Fixed costs which are not a function of an organization's size and utilization, are probably better handled in a spreadsheet.

The model includes the following cost input parameters, which are designed to generically represent several types of compensation methods for either in-house or outsourced efforts:

- normal rate per hour per developer;

- overtime rate per hour per developer, expressed as a fraction of the normal rate; and
- idle time rate per hour per developer, expressed as a fraction of the normal rate.

A “per unit of work completed” compensation type is not directly supported (e.g., pay the vendor for each function point, each maintenance request, or each line of code). However, the amount of work completed over time is an output of the model, so the calculation to determine cost under such a compensation system is trivial.

In general, the model’s project cost implementation supports many types of outsourcing arrangements. For example, an idle time hourly rate applies to both in-house and outsourced staff. During idle time, the model can represent scenarios where costs would continue to accrue for in-house, salaried staff. For outsourced work, the model can represent scenarios where the vendor only works during peak times, and is only paid for work completed. Table 10 below summarizes different ways cost inputs can be formulated in the model, and maps these formulations of cost inputs to project compensation structures.

Table 10: Formulating Cost Inputs

Cost Inputs	Project Compensation Structure
• Normal rate is average yearly salary divided by number of project weeks per year. • Overtime rate equals 0. • Idle time rate equals 1.	Salaried, in-house employees.
• Overtime rate equals 1. • Idle time rate equals 0.	Outsourced, possibly off-site effort, where vendor works on another project during idle time.
• Overtime rate equals 1. • Idle time rate equals 1.	Flat hourly rate.

For the most part, cost inputs for in-house and outsourced organizations are independent of one another. However, cost outputs are dependent upon the outsourcing relationship, and the model can be instrumented to separately output outsourcing-related costs. For example, if a vendor generates rework, the in-house organization may need to provide additional support to help the outsourced organization resolve the issue (e.g., further requirements clarification, and re-integration of the code). For the in-house organization, the amount of rework, and associated costs, can be determined; such information could be leveraged to link appropriate contract penalties or rewards to the amount of rework generated by an outsourced organization.

3.9 Model Configuration

The model includes custom input forms to allow users to provide and experiment with inputs. Within the *Extend* simulation tool, form elements, such as text boxes, sliders, and switches can be linked to inputs or outputs of graphical modeling elements. There are three basic forms in the model. The first form, as shown in Figure 7 below, is used to spec-

ify input parameters for the outsourced team; a second, similar form is also provided for the in-house organization. The control panel shown in Figure 8 below can be used to easily experiment with modeling inputs; for example, the size of outsourced team can be easily adjusted; or, a “learning rate multiplier” can be used to speed up the rate which outsourced talent reaches the top of their learning curve, and therefore peak productivity.

Figure 7: Baseline Outsourced Talent Input Profile

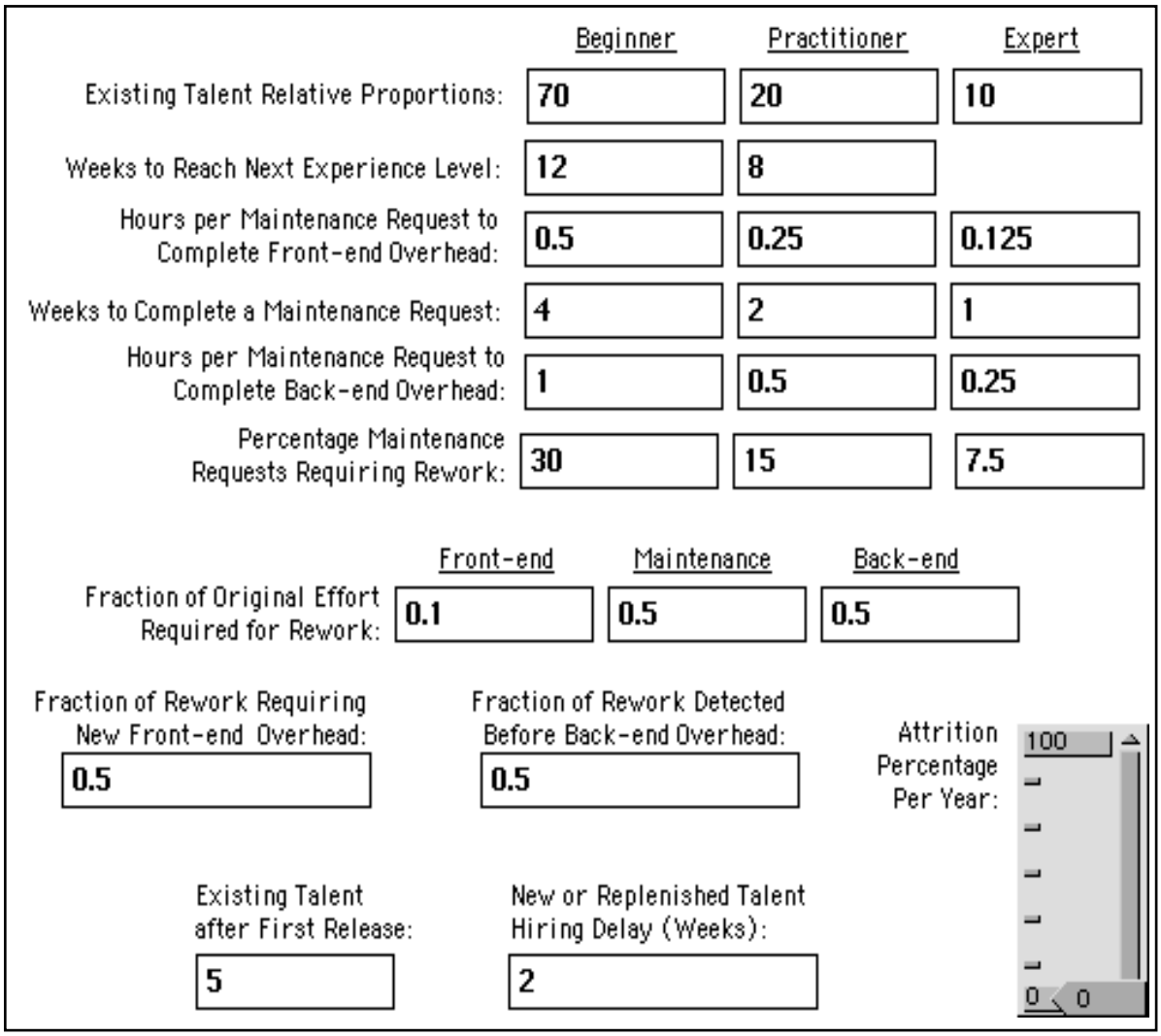
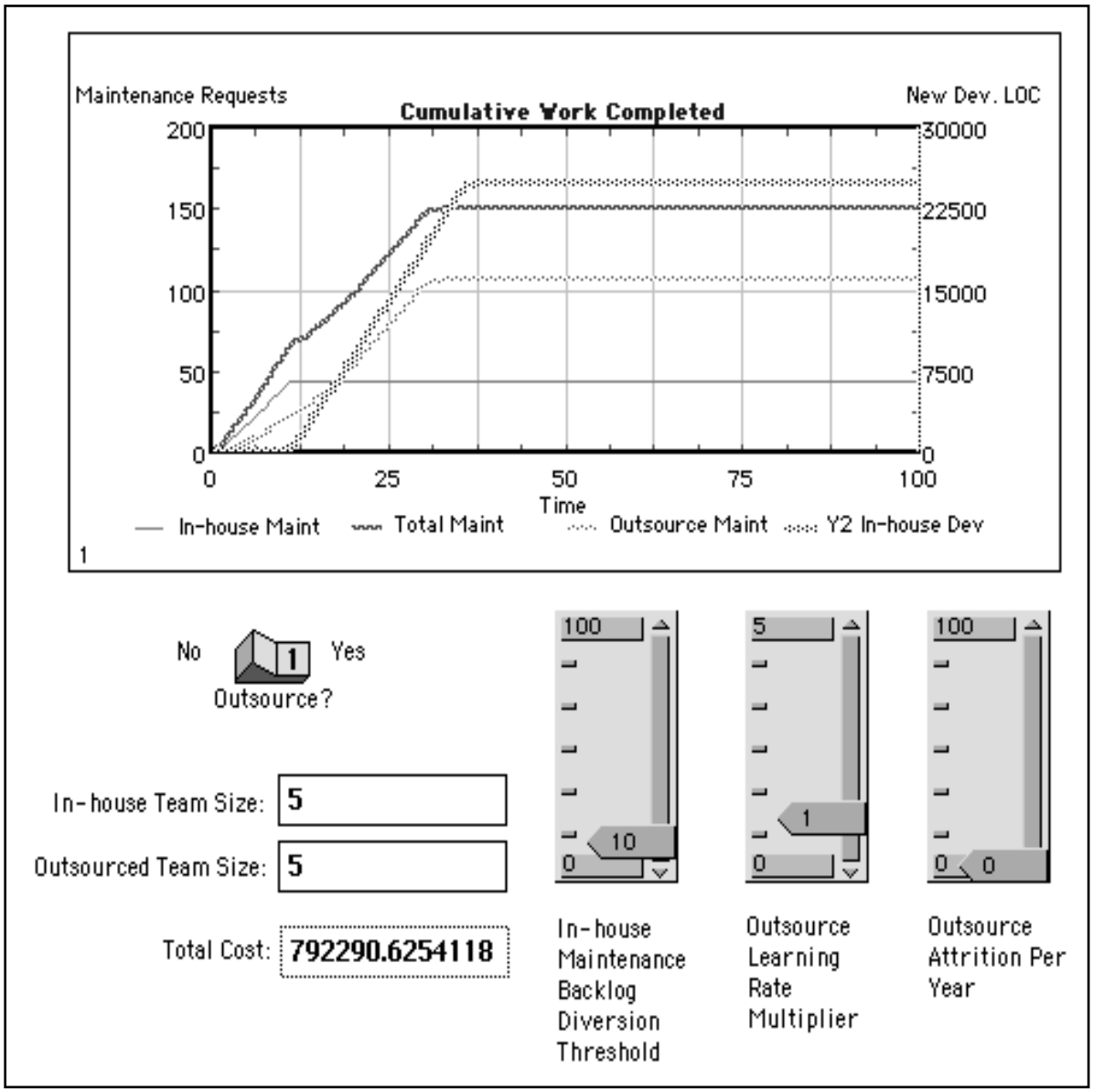


Figure 8: Simulation Control Panel



3.10 Input and Output Correlation

With a description of model components, and an understanding of the input parameters required to configure the model in place, the correlation between model inputs and outputs can be addressed. In particular, understanding of the correlations between inputs and outputs helps model users gain insights; where inputs are correlated with outputs and

outputs can be linked to individual outsourcing goals, the model can be leveraged for insights and decision support (see “Expected Benefits” on page 19 and “Analysis of Maintenance Outsourcing” on page 28). In this regard, Table 11 below describes some of the model’s input and output correlations.

Table 11: Input and Output Correlations

Input Parameter(s)	Output Correlation
• Existing Talent after First Release. • Existing Talent Relative Proportions. • Weeks to Reach Next Experience Level. • Relative Proportion of Beginning, Practitioner, and Expert Talent.	• These inputs determine the overall amount and relative proportion of beginning, practitioner, and expert talent with respect to time. • Higher levels of overall existing talent yield lower schedule duration outputs. • Higher proportions of less experienced talent, and/or longer durations to gain experience result in higher cost and schedule outputs.
• Hours per Maintenance Request to Complete Back-end Overhead. • Hours per Maintenance Request to Complete Front-end Overhead. • Weeks to Complete a Maintenance Request. • New Development LOC per Hour.	• For beginner, practitioner, and expert experience levels, these inputs specify development and maintenance staff productivity. • Increased productivity results in lower cost and schedule outputs.
• Percentage Maintenance Requests Requiring Rework. • Fraction of Original Effort Required for Rework. • Fraction of Rework Requiring New Front-end Overhead. • Fraction of Rework Detected Before Back-end Overhead.	• These fields collectively determine the overall amount, feedback, and flow of rework, as shown in Figure 5 on page 41. • Higher rework percentages represent lower quality, and increase schedule and cost outputs.
• Attrition percentage per year. • New or Replenished Talent Hiring Delay (Weeks).	• These fields specify turnover and rehiring delays. • Since newly hired talent starts with less experience (and lower productivity), higher attrition results in increased schedule and cost outputs.

3.11 Summary

The implementation of a proof of concept outsourcing software process model demonstrates first-hand the types of support outsourcing-specific modeling can provide to software organizations.

In general, the model represents outsourcing in two different ways. First, several modeling components are not specific to outsourcing, but their implementations are tailored to outsourcing, and can be easily instrumented with inputs for individual outsourcing types. For example, the model's implementation of cost factors a vendor's idle time. With this implementation, the model can represent situations where vendors are not compensated during idle time, or time spent working on other projects. Secondly, and more importantly, the model represents key interactions between an in-house and vendor organization. For example, the model represents work backlog thresholds, such that both the in-house and vendor organization work together until backlogs are cleared. Similarly, the model represents the front-end interactions needed to outsource maintenance requests, and the back-end interactions needed to re-integrate completed work back in-house.

The model is a dynamic, continuous simulation model, implemented using the *Extend* simulation package. Although Extend is not a systems dynamics modeling package per-se, the model leverages systems dynamics constructs, such as feedback, stocks, and flows, to represent key outsourcing behaviors, such as rework, experience levels, and work flow.

Chapter 4: Model Evaluation

In general, evaluation of simulation models addresses the extent to which a model's conceptual representation is an accurate reflection of the system under study [8]. To build confidence in a model's implementation, as well as the development approach in general, an evaluation was conducted.

The evaluation consisted of an independent, expert review of the model's features and capabilities. Where applicable, self-evaluation was conducted in the form of supporting, or confidence-building discussion and analyses with respect to evaluation criteria, and a systematic, structured review of the model's implementation. Information and feedback gained during the model's evaluation will help future modelers avoid the same mistakes, and motivates possible future modeling and research efforts.

4.1 Evaluation Scope

In general, evaluation efforts were organized around two primary concerns. The first concern was validation, or "building the right model". Since the model was developed as a proof of concept, rather than a commercially deployable implementation, verification, or "building the model right", was a legitimate, but secondary concern.

Within the system dynamics modeling domain, there are also established criteria for validating and verifying models. These criteria address both the static, or structural, and dynamic, or behavioral, aspects of a model. Table 12 below lists system dynamics modeling validation and verification criteria.

Table 12: Model Validation and Verification Criteria

	Validation	Verification
Model Behavior	Based upon evaluator feedback, do model outputs resemble a real system?	Is the model behavior and outputs appropriately sensitive to its inputs?
Model Structure	Based upon evaluator feedback, does the model's implementation resemble a real system?	Are dimensions consistent? Do variables and feedback address problems of study?

In terms of appropriately scaling evaluation efforts, and establishing acceptable evaluation criteria, it is useful to reconsider the research effort's overall goals and contributions (see "Research Goals and Contributions" on page 11), as well as modeling's expected role within commercial development organizations (see "Modeling Roles within Software Organizations" on page 21). In particular, recognizing the proprietary and unique nature of different outsourcing relationships, a research goal was to provide a roadmap for organizations to implement or incorporate modeling, rather than a 'shrink-wrap' modeling solution. Secondly, the modeling is primarily envisioned in a strategic, rather than tactical role, such that general behaviors and trends can be leveraged for insight, rather than specific outputs. In general, each of these points shifts the evaluation's emphasis towards validation, and away from verification activities.

With respect to the model's intended roles and the feasibility of incorporating modeling into commercial software organizations, evaluation efforts were focused upon validating the model's general behaviors and completeness, in terms of motivating an actual solution for software organizations. Based upon the intended research contributions

and expected modeling roles, as well as the criteria listed in Table 12, Table 13 below formulates a set of validation and verification related evaluation criteria, and maps supporting evaluation-related activities to these criteria.

Table 13: Evaluation Criteria and Supporting Activities

	Evaluation Criteria	Supporting Analysis and Tasks
Validation	Is the model complete?	• "Expert Evaluation", on page 63, traces the model's features and capabilities to each of the roles, outsourcing-specific problems, and expected benefits described in Chapter 2. • "Scope and Applicability" on page 16 describes applicable modeling areas and capabilities. • "Modeling Roles within Software Organizations" on page 21, discusses the model's role with respect to other, complementary decision support tools and methodologies. • The panel of experts reviewed and commented upon the analysis in "Expert Evaluation" on page 63.
	Does the model support the notion of outsourcing modeling as a worthwhile and valid concept? In other words, is the model a proof of concept?	• Intermediate research results were presented at a leading software process modeling industry conference, then reviewed and accepted for journal publication [12]. • The assessment in "Expert Evaluation" on page 63, is effectively a case-by-case proof of the concept. • Chapter 3 establishes that each model component, such as cost or rework represents a unique, meaningful aspect of the outsourcing problem; furthermore, each of these model components is differentiated from similar components which might be used to model an in-house project. • The panel of experts provided feedback on the concept's validity and feasibility.
	Do model outputs resemble a real system?	"Example Use Case and Analysis" on page 55, discusses the models outputs in general, and the panel of experts provided feedback regarding the believability of simulation outputs.
	Does the model's implementation resemble a real system?	As a proof of concept, the model does not represent an actual project, but the maintenance outsourcing project scenario, as described in "Analysis of Maintenance Outsourcing" on page 28, is arguably realistic.

Table 13: Evaluation Criteria and Supporting Activities

	Evaluation Criteria	Supporting Analysis and Tasks
Verification	Are dimensions consistent?	A self-evaluation ensured dimensions are consistent throughout the model.
	Do variables and feedback address the problem of study?	• "Analysis of Maintenance Outsourcing" on page 28 provides a rationale and discussion of key, outsourcing-specific variables and feedback loops. • A panel of experts assessed the relevance and believability of each outsourcing-specific variable and feedback loop, as presented in "Expert Evaluation" on page 63.
	Is the model behavior and outputs appropriately sensitive to its inputs?	• "Discussion of Results" on page 60 describes in general, how the model's inputs are directly correlated with its outputs. • No internal assumptions or calculations involving "magic numbers" are applied to inputs. • If an organization is uncomfortable with default inputs, they can provide their own. • Example runs and subsequent discussion established the sensitivity of various outputs to different input parameters.

4.2 Example Use Case and Analysis

In terms of evaluating the overall performance, running the model with an example outsourcing use case, or project scenario, serves several purposes.Example simulation runs and subsequent analyses collectively demonstrate the model's level of detail, flexibility and completeness in representing an outsourced maintenance effort.

Walking through an example spells out the type of metrics, corresponding to model inputs, that an organization needs to collect or estimate to effectively use the model. Setting up the scenario also illustrates the actual types of assumptions and limitations associated with the model. Analyses of model runs, based upon an example use case,

shows the sensitivity of cost and schedule outputs to outsourcing-specific productivity, cost, and quality drivers.

4.2.1 Project Assumptions

To eventually fit into each of the roles and realize each of the benefits described above, the model will need to support several types of outsourcing. The model *currently* supports maintenance outsourcing, where maintenance outsourcing is defined as using an external vendor to handle customer originated requests to fix or enhance a software product. The model can be used at the bottom of a top-down decision support process, as described in “Modeling Roles within Software Organizations” on page 21. In this role, the model can provide support to project decision-makers *after* a general, high-level indication of successful outsourcing has been established, for example:

- there is minimal risk of losing intellectual property, since maintenance development occurs on the code base for a publicly released product;
- the vendor has a positive track record;
- the vendor has sufficient domain expertise; and
- maintenance is considered a non-core activity.

The following productivity and size characteristics apply to the example project:

- developers are expected to write one to five lines of code per hour, based upon their experience level;
- to give enough time for final system testing, maintenance efforts must be completed *before* new development (simulation runs where maintenance efforts finish after new development are considered invalid);
- the vendor works off-site, and does not get paid for idle time;

- the vendor can provide five to ten people for the project;
- the maintenance request input distribution can be estimated from past projects;
- there are twenty thousand lines of code (KLOC) to be developed for Release 2; and
- seventy-five percent of maintenance requests are received the first month, and the remaining twenty-five percent in the second month.

4.2.2 Example Runs

Example model runs started with a set of baseline inputs for in-house and outsourced talent. For the baseline run, labor rates were assumed to be $50/hour for in-house talent, and $75/hour for outsourced talent. Attrition was assumed to be zero for all runs, since the project is relatively short. The input distributions for maintenance requests and new development were the same for each simulation run. The input control panels from“Model Configuration”, on page 44, were used to configure each example simulation run; the inputs shown in Figure 8 are actually those used in the baseline example simulation run.

Table 14: Example Simulation Runs

Run Number	Setup	Total Project Cost (US $x 1000)	Maintenance Phase Duration (Weeks)	Development Phase Duration (Weeks)
1	No setup required - run with baseline inputs.	792	31	37
2	1. Start with baseline inputs. 2. Turn off all outsourcing.	582	33	59
3	1. Start with baseline inputs. 2. Double outsourced team rework generation rate (decrease relative quality).	808	34	38
4	1. Start with baseline inputs. 2. Half outsourced team rework generation rate (increase relative quality).	781	31	37
5	1. Start with baseline inputs. 2. Double amount of outsourced talent.	909	25	32
6	1. Start with baseline inputs. 2. Half amount of outsourced talent.	710	40	43

Table 14: Example Simulation Runs

Run Number	Setup	Total Project Cost (US $x 1000)	Maintenance Phase Duration (Weeks)	Development Phase Duration (Weeks)
7	1. Start with baseline inputs. 2. Half outsourced talent learning rate (learn slower)	879	36	39
8	1. Start with baseline inputs. 2. Double outsourced talent learning rate (learn faster)	752	30	35
9	1. Start with baseline inputs. 2. Set outsourced labor rate to $50/hour (baseline is $75/hour).	648	31	37
10	1. Start with baseline inputs. 2. Set outsourced labor rate to $25/hour (baseline is $75/hour).	505	31	37
11	1. Start with baseline inputs. 2. Set outsourced labor rate to $25/hour (baseline is $75/hour). 3. Double the amount of effort required for front-end and back-end overhead.	508	31	37

Table 14: Example Simulation Runs

Run Number	Setup	Total Project Cost (US $x 1000)	Maintenance Phase Duration (Weeks)	Development Phase Duration (Weeks)
12	1. Start with base-line inputs. 2. Make initial out-sourced experi-ence level the same as in-house experience.	739	30	33

The simulation runs tested the sensitivity of schedule and cost to changes in the following outsourced talent input parameters (in-house parameters remain fixed):

- relative amount of experience;
- relative quality;
- relative size of outsourced team;
- relative learning rate; and
- relative labor costs.

4.2.3 Discussion of Results

In this example, the general motivation for outsourcing is to free up in-house developers to work on new development, and therefore finish the second release of the product earlier (i.e., reduce product release cycle time). In this regard, output results in Table 14 indicate when development and maintenance finish, and the total cost associated with the second phase of development and maintenance.

In addition to the indicators given above, literature also provides qualitative factors

and planning steps associated with successful outsourcing [7]. After running the model, successful or unsuccessful maintenance outsourcing is indicated by the following outputs and analyses:

- relative to other simulation runs, when will the second release of the product be completed?;
- relative to in-house development, what is the total cost of outsourced maintenance development?;
- maintenance request backlog vs. time (a measure of responsiveness to customer maintenance requests); and
- in-house developers' availability for new development vs. time.

In the context of outsourcing decision making, a manager would be expected to interpret the results from Table 14, then make a management decision about whether to outsourced, and with what input parameters. In other words, the model will not directly indicate which particular set of inputs will produce the best outsourcing results. To realize the type of outsourcing-related decision support described in "Expected Benefits" on page 19 and "Analysis of Maintenance Outsourcing" on page 28, the sensitivity of outputs to inputs, and the outputs themselves, can be linked to individual outsourcing goals. In particular, Table 15 below formulates an analysis of the example runs with respect to out-sourcing goals from Brian Hermann's survey results [6].

Table 15: Analysis of Outsourcing Goals with Respect to Example Runs

Outsourcing Goal (Ranked According to Survey Results)	Relationship to Example Runs
1. Obtain particular expertise	Labor costs are equal for in-house and outsourced talent in runs 2 and 9, but outsourcing still costs more. The outsourced talent is assumed to be less experienced at the start of the second project phase than the in-house team, and there is a certain support overhead associated with outsourcing maintenance requests.
2. Shorten schedule duration	If time to market is a manager's only driving concern, comparing runs 1 and 2 makes outsourcing look like an attractive alternative.
3. Improve responsiveness to the customer	The example runs do not directly indicate responsiveness, but runs with shorter maintenance phase durations will have improved responsiveness.
4. Add people	Run 5 doubles the amount of outsourced talent with respect to run 1, and the overall cost in run 5 is more than run 1. In run 5, more maintenance work is performed while outsourced talent is still on the learning curve and less productive. Therefore, based upon hourly billing, the outsourced talent in run 5 will work longer and cost more than run 1.
5. Improve product quality	For the example project, rework does not play a major role in cost or schedule.

In addition to analyses which can be linked to particular outsourcing goals, the following represent some additional insights from the example model runs:

- In general, the model will produce relatively more attractive cost and schedule results for larger projects, where outsourced talent has the chance to reach the top of their learning curve and work at peak productivity levels for longer periods. This result is consistent with literature, which also suggests larger outsourcing ventures tend to be more successful [14].

- Run 10 represents off-shore outsourcing, where labor rates may be lower. Run 11 also assumes front-end and back-end support overhead will be higher for off-shore outsourcing. However, the costs and levels of overhead in runs 10 and 11 do not include travel time and other costs which may be associated with off-shore out-

sourcing.

4.3 Expert Evaluation

From the perspective of experienced industry professionals, the role of an independent, expert review was to provide some independent feedback regarding the model's completeness and validity, and indirectly, the outsourcing-specific modeling concept as a whole.

Software professionals were asked to review the concept, feasibility, and believability of the proposed model. Each of these evaluators were qualified by having industry experience with outsourcing and some formal computer science or software engineering education. Additionally, several of the evaluators have a background in software process modeling, and one evaluator wrote a dissertation related to software outsourcing decision support. The evaluation questionnaire included questions to qualify the background and experience of each evaluator; an appendix lists the qualifying questions and responses from each evaluator.

Table 16 below lists the evaluation-related questions which were presented to the software professionals, and summarizes evaluation feedback for each question; an appendix provides the full set of responses for each evaluator.

Table 16: Model Evaluation Questions

Question	Evaluation Summary
Based upon the information provided to you, does the model's implementation of front-end and back-end overhead realistically capture the notion of in-house support overhead and interaction required to sustain outsourced maintenance efforts?	Each evaluator answered “yes” to this question.
Based upon the information provided to you, does the model's representation of rework effectively and realistically capture maintenance outsourcing quality issues?	Two evaluators answered “yes” to this question, and two evaluators answered with “partially”. One evaluator who responded with “yes” elaborated by suggesting the model may represent rework with more detail than is actually needed, but this makes the model better. A second evaluator who answered with “partially” was able to identify issues related to quality costs (e.g., lost schedule due to defects in the code), but did not see a direct representation of product quality.
Based upon the information provided to you, do the model's outputs look believable?	Two evaluators answered “yes” to this question and two evaluators answered “partially”. Additionally, one evaluator who answered “yes” suggested model outputs broadly matched his instincts. A second evaluator who answered “partially” elaborated by suggesting more information about the subcontractor/vendor historical performance, audit information, and SEI assessments would have been helpful.

Table 16: Model Evaluation Questions

Question	Evaluation Summary
In general, do you believe outsourcing-specific models, like the proof-of-concept model proposed herein, would be useful and value-added tools? In a research setting? In a commercial setting?	• All four evaluators responded with “yes” in regards to modeling being useful and value-added activity in a research setting. Additionally, one evaluator suggested hidden outsourcing qualities might be found if the model continues to evolve. A second evaluator suggested the thesis-related research and writing provided an excellent justification for the work. • Three evaluators responded with “yes” in regards to modeling being useful and value-added in a commercial setting. Additionally, one evaluator suggested if the model can be tuned to an organization's performance, it would be a great tool for manager's to do trade-off analysis before making project organization & staffing decisions. The fourth evaluator responded with “maybe”; he suggested the model could be used to understand the dynamics of outsourcing relationships, but should probably not be used as an ‘engineering metric’ to directly drive decision making.
In general, does the model better, and/or more uniquely, represent key aspects of software outsourcing relationships than currently available project management tools?	One evaluator responded with “yes” to this question. One responded with “maybe”, and two responded with “don’t know”. Evaluators were asked to elaborate on their “yes”, “maybe”, of “don’t know” answers. The “maybe” and “don’t know” answers can be linked to evaluators not having complete exposure to project management tools, such that they can claim the prototype model is definitely better than those tools. In general, evaluation responses indicated the model more comprehensively addresses outsourcing-specific issues than tools evaluators have been exposed to.

Table 16: Model Evaluation Questions

Question	Evaluation Summary
Based upon the information provided to you, can you suggest any areas where the model might be extended and/or improved?	The evaluation provided information relevant to identifying possible future enhancements and research. In this regard, Section 4.4 summarizes evaluation responses related to improving the existing model or suggesting future possible research.

In addition to the set of evaluation questions from Table 16, evaluators were also furnished with supporting documentation, including a detailed description of the model, and cross-references into evaluation-related analyses and discussion (see Table 13 on page 54). Evaluators were also given a copy of the simulation model, which they could run with their own inputs.

Modeling of actual projects was not part of the research effort (see "Research Goals and Contributions" on page 11), but it should be noted that the evaluators included experienced software professionals; therefore, in terms of assessing the models value and believability, feedback from this independent review partially grounds or links the research results to the commercial software industry.

4.4 Suggestions for Improvement and Future Research

In addition to evaluating an existing proof-of-concept model, evaluation responses also provided suggestions for enhancements and possible directions for future research.

4.4.1 Refinement and Calibration

In regards to modeling being a useful and value-added tool, leveraging models in

commercial settings, and making outputs more believable, the evaluation suggested the following activities:

- compare the model's performance against existing projects for tuning and prediction evaluation;
- tune the model to organizations' performance (both in-house and outsource); and
- establish closer linkage to, and perhaps additional outputs related to, subcontractor/vendor performance, audit information, and process maturity assessments.

4.4.2 Represent Vendor and Customer Liaison Relationships

One evaluator suggested the model should directly factor the experience and abilities of vendor and customer liaisons. The existing model represents the in-house and outsourced project teams as separate teams, but does not distinguish developers from liaisons. The model could be enhanced to include a separate set of experience and ability inputs for liaisons, representing such factors as communications skills, and whether or not both sides have a liaison.

4.4.3 Model Other Outsourcing Types

The prototype model looks at a single type of software maintenance outsourcing, with a specific set of business rules for the way vendor and customer organizations interact. An obvious direction for future research is to address other types of outsourcing (see Section 5.2.1), but several evaluators had specific suggestions for modeling other outsourcing types.

The existing model assumes a certain set of outsourcing business rules. For example, the model incorporates the rule, "outsource maintenance requests when the backlog of

maintenance requests exceeds X", where X is a model input value. One evaluator suggested future research efforts should be directed towards allowing the model to be customized with respect to additional business rules.

A second evaluator suggested additional model development could be organized around a taxonomy, with several dimensions, to address the many different types of outsourcing. According to the evaluator, these dimensions could include the following:

- type of relationship (including degree of integration) between the client and provider;
- degree of formality and oversight in overhead activities;
- degree of platform support for outsourcing relationship relative to the type of relationship;
- degree of technical dependence in work performed by client and provider; and
- degree of management support for outsourcing and number of levels of management.

The current model's implementation of front-end and back-end overhead is designed to generically capture the degree of formality and oversight in overhead activities, degree of technical dependence between client and provider, and degree of management support. The current model allows a single input estimate for front-end and back-end overhead, but future revisions of the model could include separate inputs for each of these overhead activities and dependency issues. The current model represents a single level of management, but it is possible to model multiple levels of management with a limited degree of detail using separate model runs.

4.5 Summary

The evaluation was generally positive, and provided helpful feedback in terms of how useful and value-added outsourcing-specific software process modeling can be. When evaluators were given the questionnaire, their expected contribution was summed up as answering the following questions:

- is outsourcing-specific process modeling a useful, value-added and worthwhile concept?; and
- does the model realistically capture the key issues associated with outsourcing relationships?

The evaluation clearly backed the claim that outsourcing-specific process modeling is a useful and value-added concept; in other words, the prototype model definitely proves the concept, as far as the evaluators were concerned. Every evaluator suggested software process modeling could be a useful and value-added tool in a research setting. Three of the four evaluators agreed that software process modeling could be a useful and value-added tool in a commercial setting. The fourth indicated such tools could be used to gain an understanding of outsourcing relationship dynamics, but may not be appropriate for direct decision making; this evaluation feedback is consistent with Section 2.3, which describes the expected roles for these tools in software organizations.

Where front-end and back-end interaction overhead are key outsourcing-specific model features, the evaluation confirmed the model realistically captures the notion of in-house support overhead and interaction required to sustain outsourced maintenance efforts. Evaluation results were generally positive with respect to the model's representa-

tion of rework and quality. One evaluator who indicated the model's representation of rework fully addressed maintenance outsourcing quality issues suggested the model may actually provide more detail than is needed. Another evaluator suggested the model's representation of rework captured quality costs as opposed to *actual* product quality; this is an important distinction, and it captures the notion that the model's representation of rework is an indirect quality measure.

Evaluation of the proof-of-concept model was a final part of the research effort. The evaluation supported the conclusion that software process modeling of the outsourcing relationship is a valid concept, and provided feedback in terms of possible future research and model development efforts.

Chapter 5: Conclusions and Future Research

5.1 Conclusions

This research effort originated from an observation that software process modeling may be able to effectively represent various aspects of software outsourcing relationships. Chapter 1 provided extensive motivation and rationale for modeling the outsourcing problem. Chapter 2 described how software process modeling can be integrated into software relationship management, including modeling's role in organizations, the linkage between model outputs and specific outsourcing goals, modeling's applicable scope, and modeling's role with respect to other complementary project management tools. Chapter 3 and Chapter 4 presented and evaluated a proof-of-concept, or prototype model. Development of a prototype concretely illustrated the types of decision support to be expected from outsourcing-specific models. Evaluation of the prototype confirmed software process modeling's role as a useful and value added tool for software outsourcing decision support.

5.2 Future Research

The maintenance outsourcing model described herein represents the first increment of the modeling efforts. The current model represents fundamental software project components, such as staffing, schedule, learning, and costs. These components are combined with outsourcing-specific rework, overhead, and work effort scheduling. The prototype model allows users to gain some interesting insights into software outsourcing, but there are plenty of opportunities for future research.

5.2.1 Modeling Other Types of Outsourcing

While the current model supports maintenance outsourcing, it was designed with

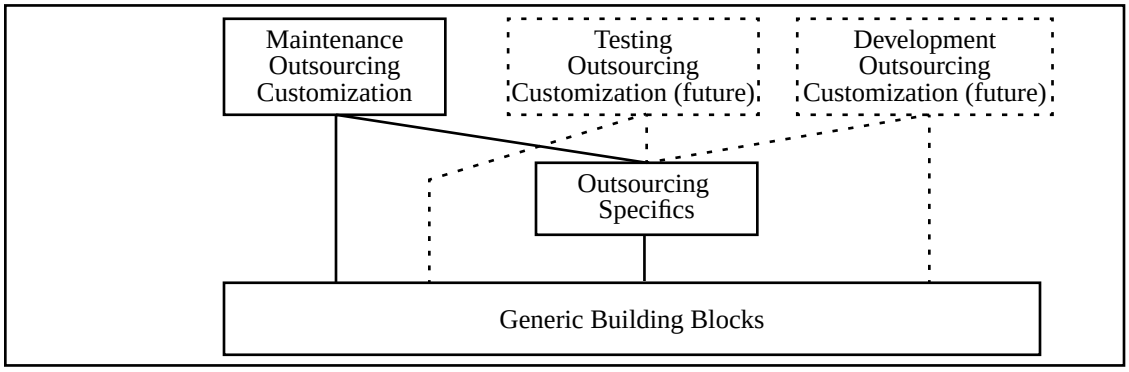
the underlying support to represent other outsourcing types. To this end, the current and suggested future organization, where dashed lines denote possible future work, is illustrated in Figure 9 below, and the following approach was used to develop the model's current maintenance outsourcing capabilities:

1. develop a set of generic, reusable building blocks to simulate software project fundamentals such as work, rework, costs, experience levels, and talent pools;

2. develop a set of potentially reusable outsourcing specific capabilities, such as shared efforts on behalf of in-house and outsourced teams; and

3. customize the model for maintenance outsourcing by implementing new behaviors or reusing generic building blocks and/or outsourcing specific capabilities.

Figure 9: Overall Current and Future Model Organization



Modeling other outsourcing types will establish the degree to which outsourcing can be generically modeled, and will allow software process modeling to more completely interface with other project management tools. This work will also motivate new features and determine the customization required to realistically simulate other outsourcing types.

Table 17 shows some candidate outsourcing types to be implemented in the future. The project characteristics given for each outsourcing type illustrate model features, such as front-end overhead, back-end overhead, and rework, which can be reused for other outsourcing types. Furthermore, these characteristics hint at the customization which will be required to support the other outsourcing types. For example, a new, customized work assignment policy will be needed to support a “pressure relief valve” in the development scenario.

Table 17: Future Outsourcing Types

Outsourcing Type	Project Characteristics
Development outsourcing	• Motivation for outsourcing is to have a "late in the project pressure relief valve" to keep in-house staffing levels stable, and ship the product sooner. • Both in-house and outsourced teams work on development. • Front-end overhead associated with screening features to be outsourced. • Back-end overhead associated with carefully ensuring outsourced development completed to specifications. • Thresholds can be used to model a pressure relief valve, where outsourcing is used to help the in-house organization get through a schedule crunch.
Development with outsourced testing.	• Testing not a "core" activity. • Test case input distribution a function of in-house development team work output. • In-house team must be large enough to ensure sufficient work output to keep outsourced team busy. • Development rework requires retesting. • Front-end overhead associated with developers carefully explaining features to be tested. • Back-end overhead associated with sign-off on testing completeness, test case design, and test execution.

5.2.2 Model Other Aspects of Outsourcing Relationships

The current model's implementation focuses on schedule, cost, and staffing issues, but there are other potentially valid modeling perspectives, such as risk management or package customization. For example, when a software product component, such as a major product subsystem, is outsourced, the in-house organization runs a risk that the vendor will go out of business, or be unable to fulfill the contract. In this situation, the in-house organization must either internalize and finish the work themselves, or find a

replacement vendor. Outsourcing relationships are also subject to risks like bait and switch, where a vendor staffs a project with a team less qualified than originally specified. To gauge the sensitivity of schedule and cost model outputs to bait and switch tactics, a model could be run with varying degrees of staff proficiency.

References

1. T. Abdel-Hamid and S.E. Madnick, *Software Project Dynamics: An Integrated Approach*, Prentice-Hall, New Jersey, 1991.

2. G. Booch, *Object Solutions: Managing the Object-Oriented project*, Addison-Wesley Publishing Company, New York, 1996.

3. J.S. Collofello, et al., "Modeling Software Testing Processes," *Proceedings of Computer Software and Applications Conference (CompSAC '95)*, 1995.

4. J.S. Collofello, et al., "A System Dynamics Process Simulator for Staffing Policies Decision Support," *Hawaii International Conference on System Sciences (HICSS)*, January, 1998.

5. B.G. Hermann, *A Decision Tool to Support Strategy Selection for Software Development Outsourcing the Applicability of Outsourcing Strategies to Specific Software Development Projects and Goals*, Ph.D. Dissertation, Arizona State University, Computer Science and Engineering Department, 2000.

6. B.G. Hermann, *Software Outsourcing Survey*, Arizona State University, Computer Science and Engineering Department, 1998.

7. C. Jones, "Marry in Haste, Repent at Leisure: Successful Outsourcing Requires Careful Consideration and Preparation," *Cutter IT Journal*, July, 1998, pp. 22-29.

8. A.M. Law and W.D. Kelton, *Simulation Modeling & Analysis*. McGraw-Hill, Inc., New York, 1991.

9. J.J. Marciniak and D.J. Reifer, *Software Acquisition Management*, John Wiley and Sons, New York, 1990.

10. S. McConnell, *Rapid Development: Taming Wild Software Schedules*, Microsoft Press, Redmond, Washington, 1996.

11. S. T. Roehling, *Software Outsourcing: Decision Support Via Dynamic Simulation Modeling*, Master of Science Thesis Proposal, Arizona State University, Computer Science and Engineering Department, 1999.

12. S. T. Roehling, et al., "System Dynamics Modeling Applied to Software Outsourcing Decision Support," *Software Process: Improvement and Practice.* Summer, 2000, pp. 169-182.

13. I. Rus, J.S. Collofello, and P. Lakey, "Software Process Simulation for Reliability Strategy Assessment," *International Workshop on Software Process Simulation Modeling (ProSim '98)*, 1998.

14. E. Yourdon, et al., “Offshore Outsourcing: The Benefits and Risks,” *Cutter IT Journal*, 1996.

Appendix:Evaluation Questionnaire Results

Table 18: Model Evaluator # 1 Responses

Evaluation Question	Responses
Please describe your background and experience.	• Bachelors in Electrical Eng (Computer Concentration). • Masters in Software Systems Management. • Doctorate in Computer Science (Software Engineering) • 13 years experience. • Position: Military Student • Responsibilities: After completing my current course of studies, I will be teaching software engineering at the graduate level for the US Air Force.
How many software development projects involving outsourcing have you participated in during the past five years?	6
What has been your exposure to software outsourcing in the last 5 years?	Independent operational tester.
Please briefly summarize the roles you've played and the good or bad experiences you've had with software outsourcing.	In all cases, I was a form of oversight checking to make sure the resulting software met requirements. Frequently we found the purchasing (contracting) organization did not really understand what the actual system users wanted/needed. Extensive nesting of outsourcing caused all kinds of problems including configuration control, division of responsibilities, etc. The further vendors were removed from the customer organization, the more trouble they had understanding how their task or product fit into an overall effort.
What is your experience with software process modeling?	I have been exposed to these tools, but haven't used them on a project.

Table 18: Model Evaluator # 1 Responses

Evaluation Question	Responses
In general, please describe your exposure and/or experience with modeling and simulation (even if you haven't been exposed to software process modeling, in particular).	I've studied this research and read the books, but haven't applied them to a project myself.
Based upon the information provided to you, does the model's implementation of front-end and back-end overhead realistically capture the notion of in-house support overhead and interaction required to sustain outsourced maintenance efforts?	Yes: I believe these overheads correctly model the effort required to estimate/manage a new requirement and to verify/validate/and test the product prior to acceptance by the in-house organization.
Based upon the information provided to you, does the model's representation of rework effectively and realistically capture maintenance outsourcing quality issues?	Yes: I believe these 4 types of rework are comprehensive. You may actually have more complexity here than you need, but that only serves to make the model better.
Based upon the information provided to you, do the model's outputs look believable?	My review is more based upon a comparison between baseline values and each scenario. In each case, the results broadly matched my insticts.
In general, do you believe outsourcing-specific models, like the proof-of-concept model proposed herein, would be useful and value-added tools?	• Yes, in a research setting: I think we might discover some hidden qualities of outsourcing if we continue to evolve this model. I would also like to compare its performance existing projects for tuning and prediction evaluation. • Yes, in a commercial setting: If the model can be tuned to an organization's performance (both in-house and outsource), this would be a great tool for manager's to do trade-off analysis before making project organization & staffing decisions.

Table 18: Model Evaluator # 1 Responses

Evaluation Question	Responses
In general, does the model better, and/or more uniquely, represent key aspects of software outsourcing relationships than currently available project management tools?	Yes: Although I don't claim to have complete knowledge of software process tools, the generally are focused on the internal processes and how they impact project outcomes. This presents a prototype for inserting outsourced processes into a larger project context. In that sense, this is a first step toward making software process modeling tools more realistic.
Based upon the information provided to you, can you suggest any areas where the model might be extended and/or improved?	This model really only looks at one type of software maintenance outsourcing (with specific business rules about how maintenance interacts with new development). The ultimate software outsourcing model will include any possible process or product component outsourcing and be customizable to different business rules too!
Do you have any general comments about the model?	I have no doubt the model is on the right track. It's an important first step toward making software process models more closely match the real world.

Table 19: Model Evaluator # 2 Responses

Evaluation Question	Evaluation Response
Please describe your background and experience.	• Bachelors in General Computer Science. • Masters in Signal Processing. • Doctorate in Process Modeling. • 18 years experience. • Current Position: Staff Engineer. • Responsibilities: Technical Lead / Manage, System Architecture & Design, Schedules, budget, personnel
How many software development projects involving outsourcing have you participated in during the past five years?	1

Table 19: Model Evaluator # 2 Responses

Evaluation Question	Evaluation Response
What has been your exposure to software outsourcing in the last 5 years?	• Vendor/sub-contractor for outsourcing. • Research on subcontracting metrics and success/ quality prediction model.
Please briefly summarize the roles you've played and the good or bad experiences you've had with software outsourcing.	• Technical definition, effort estimation, negotiation.
What is your experience with software process modeling?	I have been exposed to these tools, but haven't used them on a project.
In general, please describe your exposure and/or experience with modeling and simulation (even if you haven't been exposed to software process modeling, in particular).	Academic reading & research, examination of some of the models
Based upon the information provided to you, does the model's implementation of front-end and back-end overhead realistically capture the notion of in-house support overhead and interaction required to sustain outsourced maintenance efforts?	Yes
Based upon the information provided to you, does the model's representation of rework effectively and realistically capture maintenance outsourcing quality issues?	Partially
Based upon the information provided to you, do the model's outputs look believable?	Partially: I'd like to see more information on the subcontractor/vendor - historic performance, audit information, SEI assessment, etc.

Table 19: Model Evaluator # 2 Responses

Evaluation Question	Evaluation Response
In general, do you believe outsourcing-specific models, like the proof-of-concept model proposed herein, would be useful and value-added tools?	• Yes, in a research setting. • Yes, in a commercial setting.
In general, does the model better, and/or more uniquely, represent key aspects of software outsourcing relationships than currently available project management tools?	Maybe: Better than most of the standard commercially available tools, but I am not sure what the state of other academic/research tools are.
Based upon the information provided to you, can you suggest any areas where the model might be extended and/or improved?	As mentioned earlier, an assessment of in house and subcontractor process maturity, past performance metrics, etc.
Do you have any general comments about the model?	Interesting research. Is this the end, or do you plan on continueing to expand the model?

Table 20: Model Evaluator # 3 Responses

Evaluation Question	Evaluation Response
Please describe your background and experience.	• Doctorate in management of software technology. • 10 years experience. • Current Position: Software project leader. • Responsibilities: Leading a project for ongoing development and maintenance of a control system engineering application. Application is being co-developed and co-maintained in Phoenix and Bangalore.
How many software development projects involving outsourcing have you participated in during the past five years?	6

Table 20: Model Evaluator # 3 Responses

Evaluation Question	Evaluation Response
What has been your exposure to software outsourcing in the last 5 years?	Customer of outsourcing: I have been involved as a customer in a variety of outsourcing relationships, including support of 3rd party development, acquisition and productization of a prototype, outsourcing development of product modules to a corporate subsidiary for development, and co-development and maintenance with a corporate subsidiary.
Please briefly summarize the roles you've played and the good or bad experiences you've had with software outsourcing:	• Worst experience: a corporate subsidiary agreed to take on an upgrade release and failed to do it; I ended up taking it back and doing it myself. • Best experience: I am currently leading a project being co-developed by teams in Phoenix and Bangalore; we have developed working relationships and a common product knowledge that support concurrent development of interacting functions.
What is your experience with software process modeling?	I have used these tools on projects, and am quite familiar with them.
In general, please describe your exposure and/or experience with modeling and simulation (even if you haven't been exposed to software process modeling, in particular):	• Two ASU IE courses in simulation. • Worked with discrete event, continuous, deterministic, and stochastic models. • Worked with SLAM, Extend, and ithink. • Designed and conducted experiments on four software project simulation models. • Developed a software project simulation model for risk management of six major risk factors. • Published in the field of software process modeling.

Table 20: Model Evaluator # 3 Responses

Evaluation Question	Evaluation Response
Based upon the information provided to you, does the model's implementation of front-end and back-end overhead realistically capture the notion of in-house support overhead and interaction required to sustain outsourced maintenance efforts?	Yes: You have captured the most important factors: experience (including product knowledge) and time to learn. Other factors may be important but designed experiments should be used to measure their relative influence.
Based upon the information provided to you, does the model's representation of rework effectively and realistically capture maintenance outsourcing quality issues?	Partially: The question is broad and seems to include issues related to costs as well as to level of quality. The model represents quality costs but I don't see measures of quality represented because the model does not appear to include a product flow. (I may be missing it. Extend does not provide a distinct visual construct for flows.)
Based upon the information provided to you, do the model's outputs look believable?	Partially
In general, do you believe outsourcing-specific models, like the proof-of-concept model proposed herein, would be useful and value-added tools?	• Yes, in a research setting: You have demonstrated it. Your thesis provides excellent justification. • Yes, in a commercial setting: The model needs refinement and calibration, but the value is clearly demonstrable from your work.
In general, does the model better, and/or more uniquely, represent key aspects of software outsourcing relationships than currently available project management tools?	Don't know: I don't know of any project management tools that explicitly support outsourcing decision-making. And your work goes beyond generic project management tools by incorporating and focusing on factors particular to outsourcing.

Table 20: Model Evaluator # 3 Responses

Evaluation Question	Evaluation Response
Based upon the information provided to you, can you suggest any areas where the model might be extended and/or improved?	Your thesis provides an excellent case for this model. However, the flavors of outsourcing are many more. See below for more: I think it would be helpful to develop a taxonomy of outsourcing that includes dimensions such as (a) type of relationship (including degree of integration) between the client and provider, (b) degree of formality and oversight in overhead activities, (c) degree of platform support for outsourcing relationship relative to (a), (d) degree of technical dependence in work performed by client and provider, and (e) degree of management support for outsourcing and number of levels of management involved. I'm suggesting that these are all the right dimensions (that is a study in itself) but even an informal taxonomy based on such dimensions would be helpful in guiding future modeling of outsourcing by allowing you to account for factors pertinent to the ranges of these dimensions. I found the thesis excellent an excellent demonstration of how simulation can be applied as a decision-making tool for outsourcing.
Do you have any general comments about the model?	The model does not incorporate a factor for the experience and ability of the client and provider representatives who interface. I have found that this is critical. If neither side has a strongly qualified liason who knows the technical work and has good communications skills, the project is in trouble. If one side has such a person, the project can work. If both sides have such a person, the project can succeed very well.

Table 21: Model Evaluator # 4 Responses

Evaluation Question	Evaluation Response
Please describe your background and experience.	• Bachelors in Math, Computer Science. • Current Position: Staff Software Engineer • Current Responsibilities: Develop software.
How many software development projects involving outsourcing have you participated in during the past five years?	3
What has been your exposure to software outsourcing in the last 5 years?	• Customer of outsourcing. • Vendor/sub-contractor for outsourcing
Please briefly summarize the roles you've played and the good or bad experiences you've had with software outsourcing:	• I managed one of the groups using outsourced developers as part of the development team. • Being a team instead of competitors (most of the time) allowed us to work together very well. • I've also done some work on the side for other people.
What is your experience with software process modeling?	I have been exposed to these tools, but haven't used them on a project.
In general, please describe your exposure and/or experience with modeling and simulation (even if you haven't been exposed to software process modeling, in particular):	My only exposure to SPM was from Steve's paper, I think.
Based upon the information provided to you, does the model's implementation of front-end and back-end overhead realistically capture the notion of in-house support overhead and interaction required to sustain outsourced maintenance efforts?	Yes: the answer is 'mostly'. There are always going to be interpersonal issues that affect the outcome regardless of how well a group works out the technical and 'management' issues.

Table 21: Model Evaluator # 4 Responses

Evaluation Question	Evaluation Response
Based upon the information provided to you, does the model's representation of rework effectively and realistically capture maintenance outsourcing quality issues?	Yes
Based upon the information provided to you, do the model's outputs look believable?	Yes
In general, do you believe outsourcing-specific models, like the proof-of-concept model proposed herein, would be useful and value-added tools?	• Yes, in a research setting. • Maybe, in a commercial setting: Its not an 'engineering metric' - i.e. it should be used as a guide and a way to understand some of the dynamics, but to say 'well, the model says go this way' without understanding what is going on underneath would be 'a bad thing.'
In general, does the model better, and/or more uniquely, represent key aspects of software outsourcing relationships than currently available project management tools?	Don't know: The only 'project management tool' I've used is Microsoft project, and I find it woefully lacking in almost all respects.
Based upon the information provided to you, can you suggest any areas where the model might be extended and/or improved?	
Do you have any general comments about the model?	No.