FORENSIC DOCUMENTS OPTIMIZATION

Final Report

OR 680
Spring 2007

5 May 2007

Prepared by: Document Optimization Group

Chris Catalano
Chun-Yu Chang
Chris Joson
David Matthes

Sponsored by:
Huron Consulting Group
Aerospace Corporation

GEORGE MASON

EXECUTIVE SUMMARY

This study evaluates the operational, functional, and financial structures of three systems for the Huron Consulting Groups' forensic discovery lab. The study satisfies the requirements of two project sponsors, the Huron Consulting Group and the Aerospace Corporation. Huron is a financial consultant company that provides financial investigation services. Aerospace Corporation is a federally funded research and development center that provides technical expertise to various US National Security Space programs. The Huron Consulting group requested the study to evaluate and compare their in-house Electronic Data Discovery System with two new software intensive alternatives. The Aerospace Corporation is specifically interested in determining if SysML is an effective tool for model based systems design.

In order to provide viable recommendations to the Huron Consulting group, the initial approach was to understand, model, and analyze the three EDD systems. The models for the three systems were created iteratively. Each iteration incorporated new information from discussions between the design team, the domain experts, and the client. The SysML diagrams fostered communication and allowed the stakeholders to complete multiple design iterations clearly and effectively. A trade study provides cost-performance analysis of three EDD Systems.

The Baseline System: The Baseline system must remain in place. The process has excellent potential because of its flexibility and adaptability. It is advisable to develop new pieces of software and additional scripts to reduce the amount of human interaction with the system. The alternative systems gain traction by providing a single piece of software to execute multiple functions. The baseline requires significant human interaction to accomplish the same tasks. By creating new scripts the baseline can be made to mimic the more expensive alternatives while maintaining the flexibility in components.

The Autonomy System: The Autonomy system provides high speed at high fixed costs. The $2 million initial cost for the software and the $250,000 maintenance fee are steep. Fortunately through the NPV analysis it was shown that the Autonomy system could be profitable if the number of projects accepted is sufficiently increased.

The Attenex System: The benefits to the Attenex system are its speed and its zero initial cost. The steep downside is the $500 per Giga-byte processing charge. The analysis shows that Attenex can process data faster than the baseline but it cannot outperform the baseline in terms of profitability under the same pricing structures. In order for Attenex to increase its profitability the pricing structure Huron offers its customers would have to increase to offset the operational costs.

Aerospace Corporation asked the Document Optimization group to use SysML with Rational System Developer to model a system and provide feedback on the modeling language. Rational System Developer supports the SysML language and can be used to model systems and maintain consistency and requirement links across different models.

The Document Optimization team used SysML to model and help evaluate the different choices for the Huron Consulting Group. The Document Optimization team created a survey by decomposing the evaluation criteria into a Multi-Attribute Utility Assessment Evaluation Hierarchy to support the evaluation of SysML and Rational System Developer. The results of this evaluation provide the Aerospace Corporation with an analysis of the feasibility and effectiveness of Rational Systems Developer's implementation of SysML.

The Document Optimization Team found that SysML supports system design as intended and effectively maintains consistency across various models. However, SysML is difficult to learn and requires substantial training. Aerospace Corporation is likely to see benefits with using SysML for complex National Security Space programs.

TABLE OF CONTENTS

Table Of Figures

1 Introduction

This study evaluates the operational, functional and financial structures of three systems for the Huron Consulting Groups' forensic discovery lab. The study satisfies the requirements of two project sponsors the Huron Consulting Group and the Aerospace Corporation. The Huron Consulting group has commissioned the study to evaluate and compare their in-house Electronic Data Discovery System. The Aerospace Corporation is specifically interested with the use of SysML as a tool for model based systems design.

Electronic Data Discovery (EDD) is the collection, processing, and review of electronic information in the discovery phase of litigation. The EDD industry is a niche market of specialty vendors who service law firms, private corporations, and government agencies. The Huron Consulting Group is uniquely positioned in the EDD market place by providing the full spectrum of EDD services. Huron has three interconnected teams that handle the collection, processing, and review phases of the EDD chain. The middle link in the chain, the processing group, is the focus of this study.

The processing group uses an EDD system that focuses on a high level of flexibility and variability. The system is designed to maximize revenue while sacrificing throughput. Competing vendors rely on off the shelf systems that can increase throughput at an increased cost. The business climate has evolved to an environment where processing speed is now paramount to clients.

To adjust to the shifting market demands Huron's baseline system is to be evaluated against two off the shelf EDD solutions. The migration to either of these software products would require a dramatic overhaul of the entire forensic document processing system.

The study is subdivided into three complimentary pieces. The first piece describes the three systems in terms of SysML. The SysML model is used to compare and contrast the baseline system against the alternatives. The second piece of the study is a capital budgeting analysis of the three alternatives with Net Present Value. The third piece of the study is an analysis of the utility of SysML with a multi-attribute hierarchal decomposition.

2 EDD System Background

The forensic document processing team within the Huron Consulting Group is tasked with operating and managing the processing component within the EDD chain. The EDD processing system systematically converts electronic documents into image files and a structured database format. The system accepts common electronic files such as spreadsheets, text files, graphics files and electronic mail files and outputs TIFF images for all native files and a structured database. The TIFF images provide a snapshot of the native documents. The database provides an aggregated reference to all the related metadata information. The final deliverables are reviewed and searched by Attorneys to find documents relevant to their case.

The baseline system was designed for flexibility and variability. The system has evolved to satisfy most client requests and specifications. The strength of the baseline system is the emphasis on modularity. The process is constantly evaluated to maintain a high level effectiveness. Individual steps and components can be tweaked, changed, or upgraded independent of the other pieces of the system. The baseline has nearly two-dozen steps. Each step is a discrete process that requires unique software. The benefit of the configuration is that individual steps can be modified and upgraded independently. The disadvantage is that each step and component must be managed by significant human interaction.

The Autonomy system is commercially available software developed to search, categorize and process native files. The software is used by large corporations and government agencies for a wide range of functions. The Autonomy system can be tailored to operate in an EDD environment where it can quickly process and output native files. The benefits of the Autonomy system are its speed and process simplification. The disadvantages are the significant initial and maintenance costs. The Autonomy system costs $2 million to procure and an additional $250,000 a year in maintenance.

The Attenex system is commercially available software that directly supports EDD processing. The software has been designed and marketed towards EDD vendors. The benefits of Attenex are its speed and its simplified process. The major disadvantage is the increased operational costs. The Attenex system requires a $500 operational fee for each Giga-byte of data processed through the system.

The balance between throughput, revenue and costs is a difficult situation to mitigate. This study provides an analytical approach to which system is best suited for the Huron Consulting Groups' document forensics team.

3 SysML Overview

The Systems Modeling Language (SysML) is an extension of the Unified Modeling Language (UML) and provides a uniform framework for system designers to capture and communicate different viewpoints of a system. The viewpoints of a system are represented in terms of visual models that are defined by traditional UML diagrams and diagrams specific to SysML. For system designers SysML brings the capabilities to produce analytical and executable system models to a degree that was previously unattainable.

The systems engineering community is plagued by an approach to system design that includes ad-hoc graphics and white paper reports. SysML is a UML-based modeling language for systems engineering. SysML is positioned to be the solution to sloppy design work by providing a standardized but flexible model-driven environment. SysML is positioned to become the standard modeling approach for all major systems engineering efforts.

SysML addresses the need for the integration of systems engineering models with other discipline-specific models such as software, hardware, simulation, and analysis. SysML was designed to provide the systems engineering community a standard modeling language to replace the myriad tools currently being used.

SysML assists in managing complex system development. It provides standardized techniques for hierarchical modeling, requirements analysis, and supporting incremental and evolutionary development. Errors and ambiguities are reduced throughout a projects lifecycle by allowing designers to developer a more detailed and rich representation of systems.

SysML incorporates some elements of UML 2.0 and expands on others. The techniques ported from UML to SysML include data structures for modeling actions, activities, classes, general behavior, information flows, interactions, models, profiles, state machines, structures, and use cases. Data structures extended by SysML are constraint blocks, ports & flow, activities, allocations, requirements, and blocks.

The four pillars of SysML are the behavior diagram, requirements diagram, structure diagram, and parametric diagram. The behavior diagram can be decomposed into activity, sequence, state machine, and use case diagrams. The structure diagram can be decomposed into a block definition diagram, an internal block diagram and a package diagram. The requirements hierarchy describes the requirements for the system. The parametric diagram is used to express constraints and equations between value properties. The figure below shows graphically the structural hierarchy of SysML.

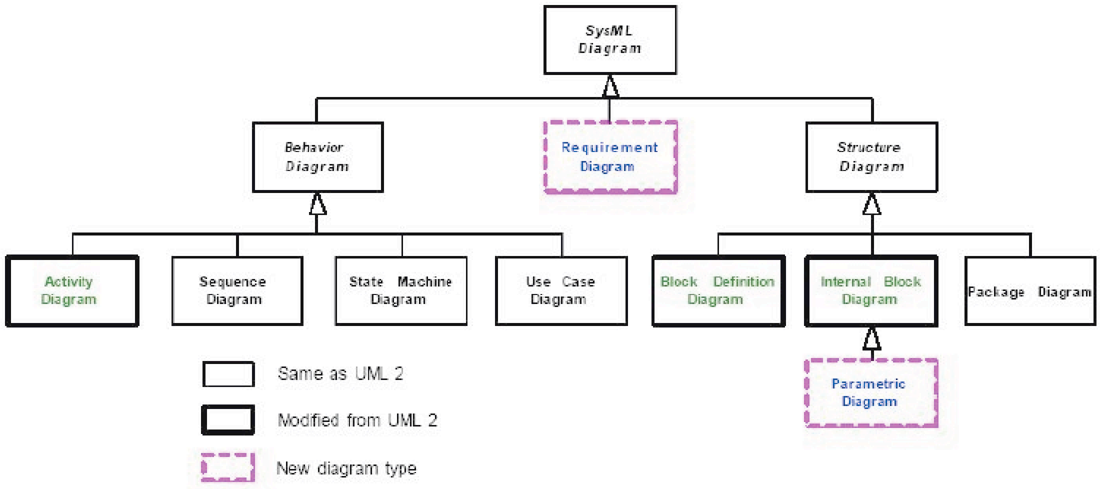


Figure 1: SysML Hierarchy

3.1 Electronic Data Discovery Systems in SysML

One of the objectives of this study was to provide a comparative performance analysis of three Electronic Data Discovery (EDD) systems. In order to provide viable recommendations, the initial approach was to understand, model, and analyze the three EDD systems. The EDD model was created in Rational System Developer with the SysML toolkit. The models for the three systems were created iteratively. Each iteration incorporated discussions between the design team, the domain experts, and the client. The SysML diagrams fostered communication and allowed the stakeholders to complete multiple design iterations clearly and effectively.

Several SysML diagrams were developed to holistically abstract the three EDD systems. Each type of diagram captures and depicts different perspectives of the EDD systems. The following section describes why specific SysML diagrams were developed and how each model contributed to the overall design effort.

3.2 Requirements Diagram

The Requirements Diagram describes the specifications of a system. It provides a resource to communicate critical information about individual requirements. The information includes details about the source of the requirement, the author, content, and parent-child hierarchical relationships. A hierarchical structure of requirements can be modeled to describe derived requirements nested within parents. In SysML, requirements can be assigned to functional elements throughout the model. Then the requirements can be traced to determine if and where the requirements are satisfied.

The Requirement Diagram was used to effectively communicate the EDD systems requirements. The diagram provides a visualization of the requirements hierarchy. A visual requirements model provided an effective means of communicating the inter-relationships between requirements. The visualized requirements model allows stakeholders to quickly understand and communicate the specified functionality of the system. This minimized the need to develop and read lengthy textual requirement documentations to understand the relationships between requirements.

Figure 2 describes the basic requirements that all three EDD systems must satisfy.

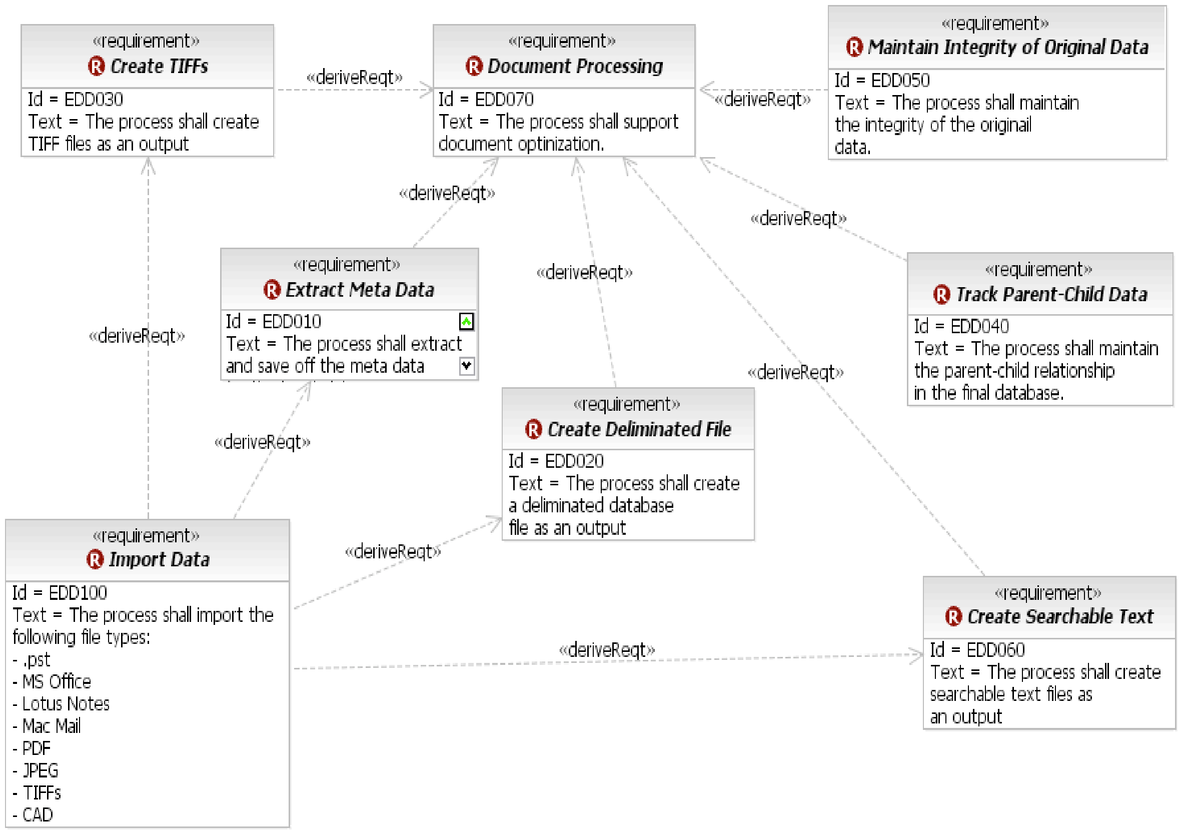


Figure 2: Requirements Diagram

3.2.1 Use Case Diagram

A Use Case Diagram describes how stakeholders interact with the system. The Use Case Diagram shows the interactions between the system and external entities. A use case diagram will paint a clear picture of system behavior and promote a common understanding between the stakeholders.

In this study, the Use Case Diagram provided a high level context of interactions between the customers and the EDD service providers. In Figure 3 below, the customer provides raw data, the Document Processing Team receives and processes the raw data, and finally the review team will read and evaluate the finished data.

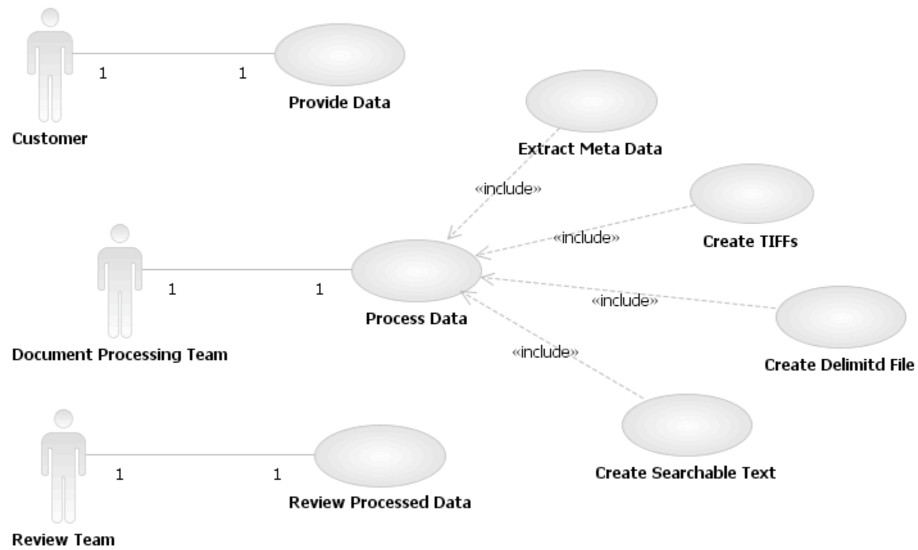


Figure 3: Use Case Model

3.2.2 Block Definition Diagram

The Block Definition Diagram provides a method of organizing the attributes and functions of the EDD system into a structural model. The diagram also enables elements from other SysML diagrams, such as activities and requirements, to be allocated to a physical component in the EDD System. A Block Definition shows the structural composition as well as the characteristics and behaviors of each component. The diagrams communicate physical relationships between different components in a system. The Block Definition Diagram can show the decomposition of a complex component into multiple sub-components.

In the EDD system the first level decomposition included five sub-components. The five subcomponents were a server component, a client computer component, a data component and a network component. Further decompositions of each sub-component fleshed out the structural hierarchy of the EDD System. Figure 4 shows the first level decomposition.

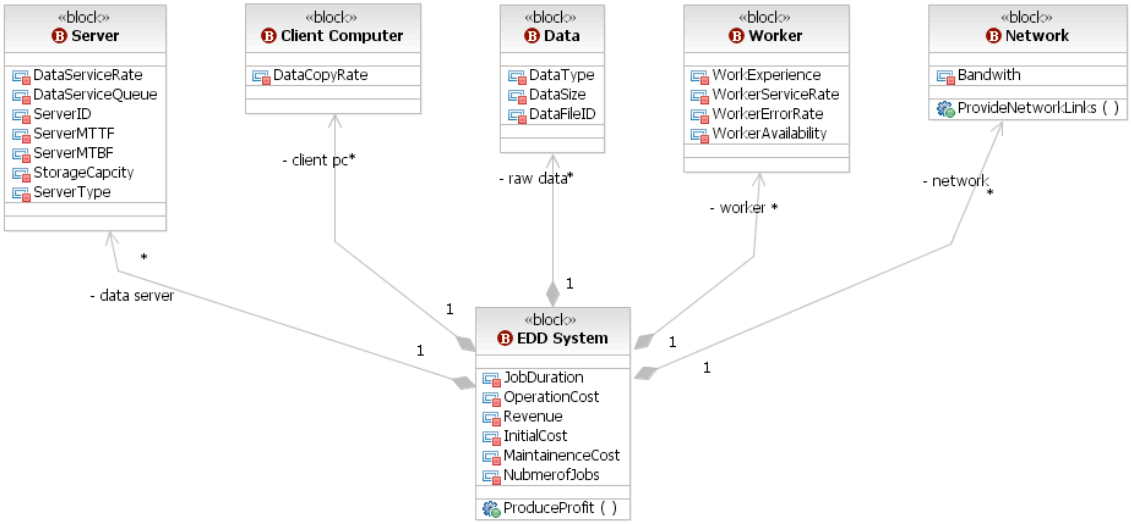


Figure 4: Top Level Block Definition Diagram

Each component may include a set of attributes that further refines its functionality. For example, a data server has a storage capacity attribute to specify how much data it can hold. A more detailed inspection of the Worker component is shown in Figure 5. The decomposition reveals that the Worker can be subdivided into a Review Team, a Document Processing Team, and a Project Manager. Note that the Worker component has a Work Experience attribute that is inherited by each subcomponent. Each component may also contain a set of operations or activities. The Document Processing team component contains three operations: copy data to server, determine data type, and select data server.

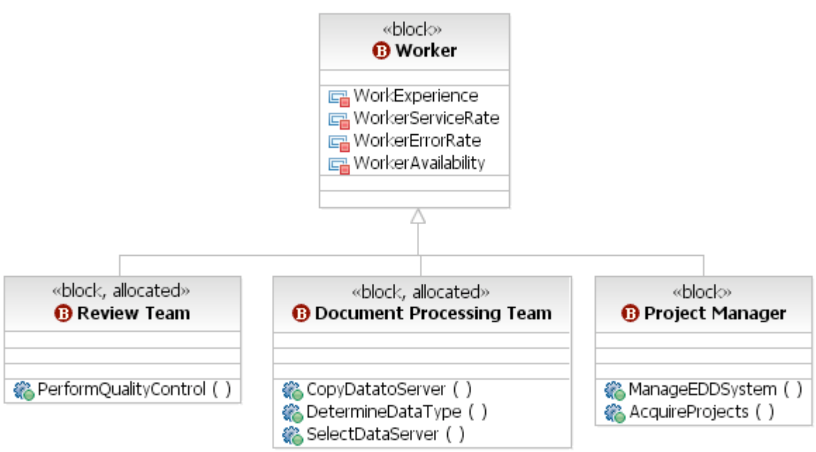


Figure 5: Decomposed Block Definition Diagram of Worker

3.2.3 Parametric Diagrams

Parametric diagrams allow designers to model performance and reliability using constraint blocks and block definition diagrams. Parametric diagrams are used to express system constraints in the form of variables, constants, and equations. The diagrams also show the context of the constraints. Constraint blocks are used to capture equations, which can then be used by a computational engine for analysis. Parametric diagrams allow for the integration of engineering analysis with design models and are the stepping-stones needed to make the SysML model executable.

A Parametric Model was not developed in this study due to time constraints.

3.2.4 Activity Diagrams

In SysML, an activity diagram provides a view of the behavior of a system by describing the sequence of actions. Activity diagrams are similar to flowcharts. Each diagram shows the relationship between the processes and the activities. Activity diagrams have the added benefit of showing parallel, concurrent, and alternate flows.

The activity diagram, shown in Figure 6, describes the baseline EDD process used by the sponsor. An iterative approach was used to properly derive the baseline systems' activity diagram. The process involved continual efforts from the stakeholders including significant input from the sponsor and domain experts.

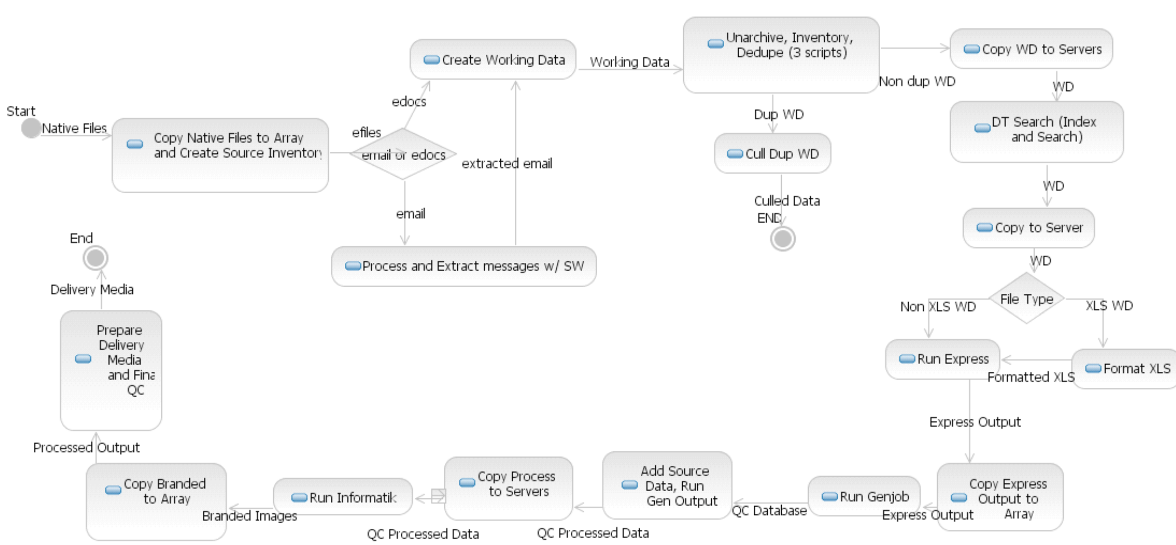


Figure 6: Process Diagram

The activity diagram presented in Figure 6 provides an overview of the EDD baseline process. The native files are received from the customer. The native files are copied to a storage array and a source inventory is created. Any electronic mail is then extracted from the source mail programs (e.g. Microsoft Outlook, Lotus Notes, etc.). The email is then processed into a common text delimited format and any email attachments are extracted. All the electronic data along with the processed email create the working data, which is then un-archived and inventoried. A program is then used to remove any duplicate electronic documents. An index is created of all of the working data and a search can be performed for "key words". Any Microsoft Excel files are then processed into a common format. Next, the metadata, which is pictures and text, is extracted from the working data. The metadata output is then put into a database and all the data is then extracted into a delimited format to be delivered to the customer. All pictures are then watermarked, or branded, before the final delivery media is created. Quality checks are performed throughout the entire baseline EDD process.

3.2.5 Activity Diagrams with Partitions

Partitions are a secondary construct within SysML that explicitly show responsibilities for the activities using swim lanes. Horizontal lines evenly distributed throughout the activity diagram represent swim lanes. Swim lanes provide a view of the behaviors in an activity, but do not affect the control flow or object flow. The swim lane's respective owner performs the activities that fall within a swim lane. The swim lanes are then allocated to components from the block definition diagrams.

The activity diagram, with swim lanes, of the baseline EDD process is presented in Figure 7. With the current process, there are nine separate programs that are involved to process all of the legal documents on the processing server. In addition with the current process, the engineer has three decision points and has to move or copy data nine separate times. The current process also requires the use of six Perl scripts to process the legal documents into a format needed for the lawyers. The process of the current EDD system requires six separate quality checks by the review team. As can be shown from the activity diagram, significant human interaction is involved with the current baseline process.

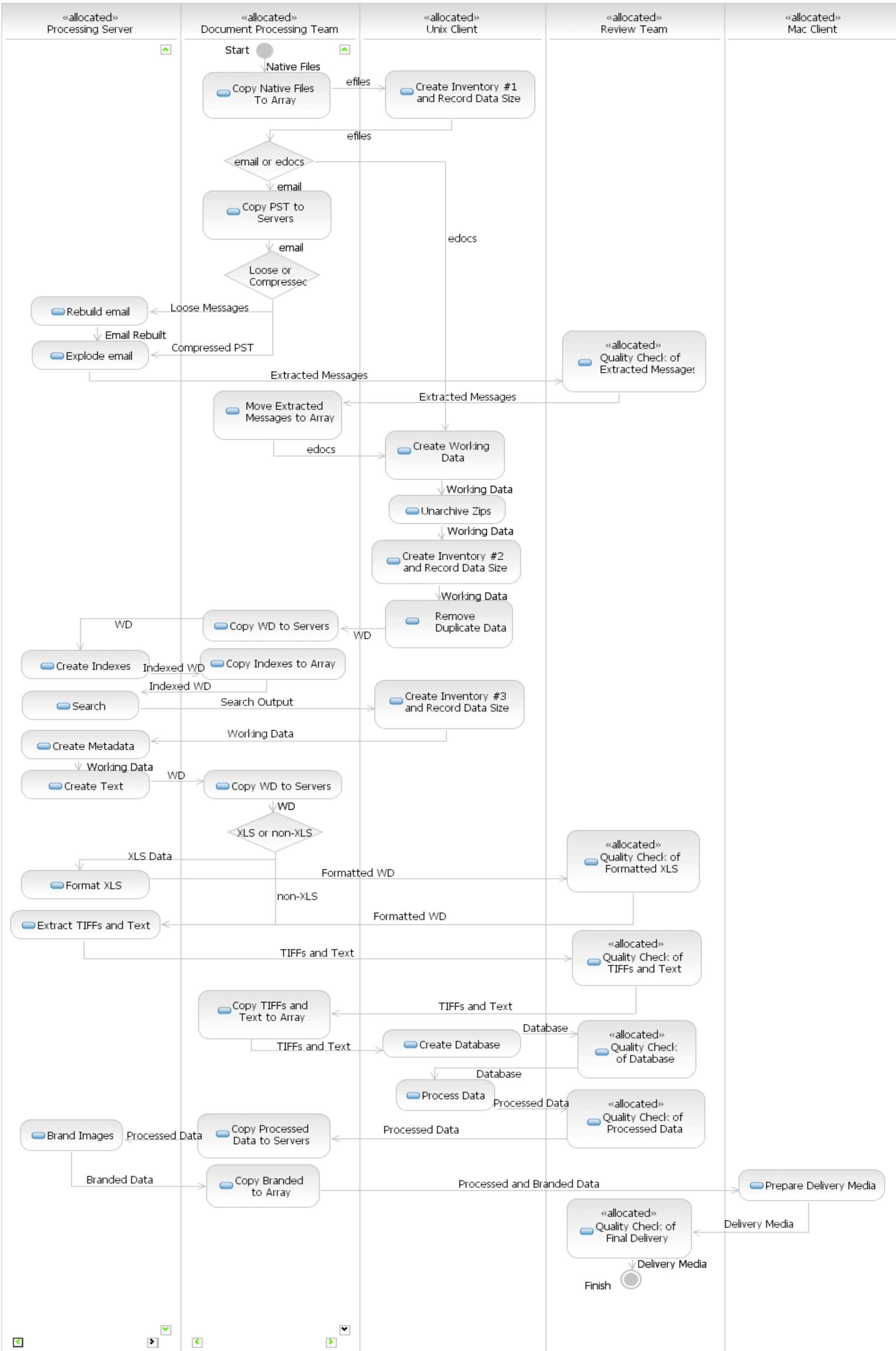


Figure 7: Baseline Activity Diagram (With Swim Lanes)

The activity diagram of the alternative EDD process is presented in Figure 8. If Huron Consulting decided to move to either the Attenex or Autonomy software, there would be only one program running on the processing server that would perform sixteen separate activities. With the alternate process, the decision points are removed and the operator has to only copy data twice. The alternate process only requires the use of two Perl scripts and requires only two separate quality checks by the review team. As can be shown from the activity diagram, the Document Processing Team and the Review Team are not as heavily interlaced with the process.

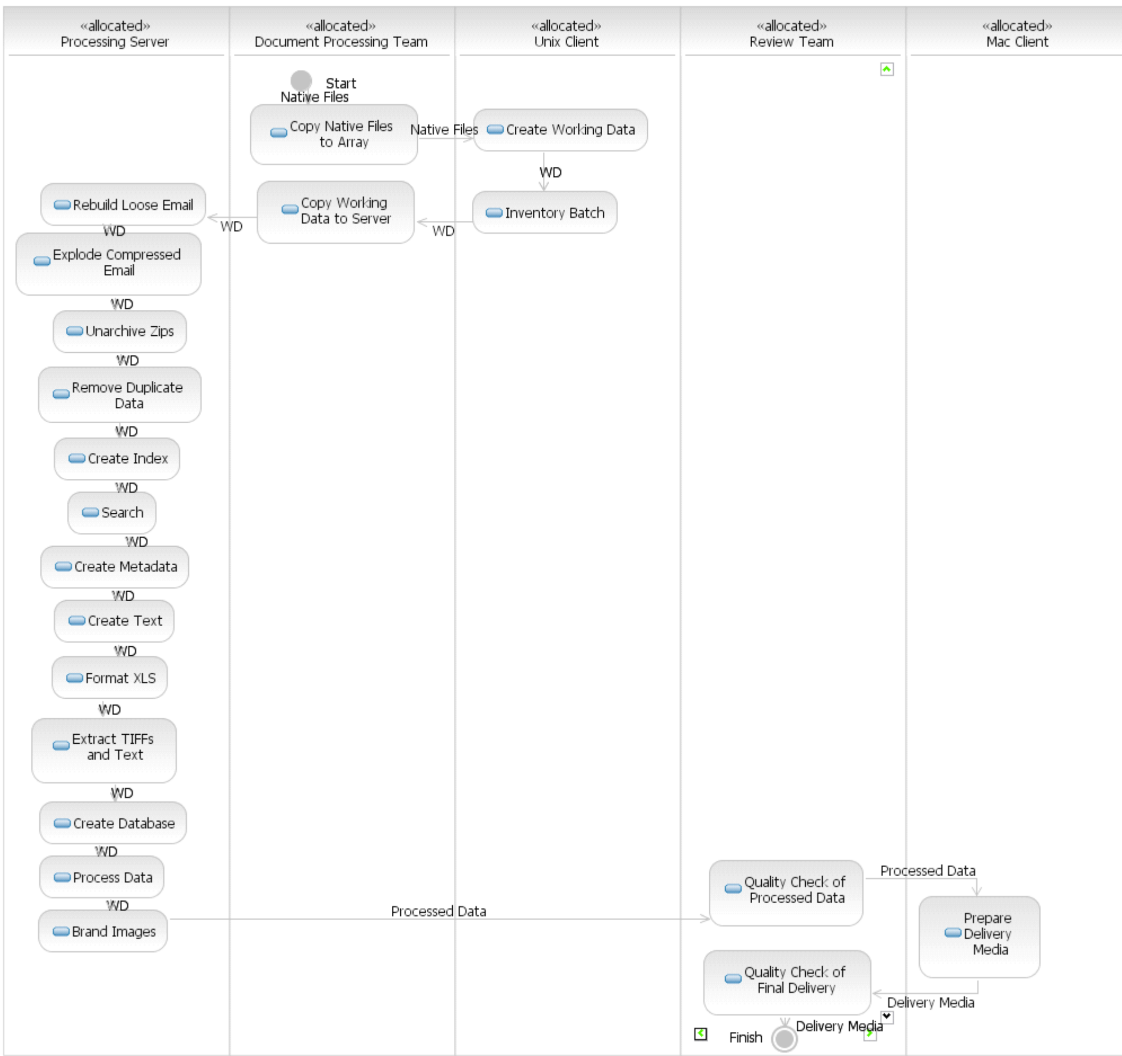


Figure 8: Alternative Activity Diagram (With Swim Lanes)

3.2.5.1 Process Comparison

Comparing the two SysML activity diagrams together, it can be shown that the alternative process is a lot simpler than the current process. Not only are there fewer activities to perform overall, but there are also fewer decisions that have to be made in the alternative process. There is a little bit more leniency with regards to the sequence in which the sixteen activities have to be executed in the alternative process, since all sixteen activities are performed using a single software program. From a maintenance standpoint, it is easier to maintain one executable and two Perl scripts than it would be to maintain nine programs and six Perl scripts. Instead of calling several help desks for questions about their individual programs and Perl scripts, a single vendor could handle all of questions for the sixteen activities pertaining to the alternative software.

From a process standpoint, it is certainly easier to train a new employee with the alternative process than with the current process. Currently, it takes approximately three months for a temporary or new employee to be able to perform the current process by him or herself. The training predicted for a new employee to follow the alternative process would be approximately one month.

In the current EDD process, the activities must be performed in an exact sequence and if one step is forgotten, the process needs to be repeated from the forgotten point forward. The decision points and the manual copying of data each introduce potential operator error. This would provide the process with less points of failure and a lower probability of error. The alternate process would save both the engineers and the quality team more time because they would not have to be as manually involved with the data processing. They would be able to spend their time processing more jobs through the alternate process and they would potentially have more time to perform other tasks.

3.3 SysML Lessons Learned

Throughout the duration of the project, the experiences related to the development process of the EDD system models provided us with a set of valuable lessons learned. The lessons learned are intended to provide a source of guidance to future SysML users regarding some of the best practices of modeling in SysML as well as potential challenges one might encounter during the design process.

One of the challenges encountered during the development of the SysML models for the EDD Systems is discovered during the integration of the various types of SysML diagrams. A typical system design team is consisted of more than one developer to produce diagrams for a system model. This implies that the diagrams can come from various developers and requires a process to integrate the various diagrams. Even though SysML provides a framework to define relationships from one diagram type to another, it does not explicitly provide the capability to automatically align all the diagrams to each other.

The methodology of integrating the diagrams into one cohesive model is left to the consent of the design team. This issue was evident after the preliminary iteration of diagrams was developed for the EDD Systems. When the team gathered to integrate various diagrams into a single system model, there were inconsistencies that made it difficult to relate one diagram to another. One of the inconsistencies can be attributed to the lack of proper naming convention for the various types of elements between each type of SysML diagram. For example, the element that performed the processing of documents was named “engineers” in the activity diagrams but it was called “document processors” in the block definition diagrams. Although the two diagrams were both referring to the same entity, the inconsistent naming could result in confusion as to whether or not there are two entities that perform the document processing operations. As the result of this inconsistency, several hours were spent between the developers of all of the different diagrams to ensure that the naming of all the elements across the different diagrams is consistent for the model.

In addition to providing a naming convention for the model elements, a connection between the different SysML diagrams needs to be established. This step is essential to provide a relationship between the types of SysML diagrams that represent the different perspectives of the system. Allocation of elements from one type of diagram to another type of diagram provides this ability. An allocation represents a relationship that maps one model element to another model element. While attempting to allocate the activities in the Activity diagrams, it became apparent that it was especially difficult to allocate activities to other components without dividing the activity diagrams into partitions. Partitions are a secondary construct within SysML that visually divides the flow of activities by responsibilities. The activities that are grouped by the divisions can be allocated to components. This establishes that the activities are performed within the allocated components.

A problem encountered during allocation of activities is that the partitions are difficult to add if the activity diagrams already exist. Whenever a partition is inserted, it is placed directly on top of the existing activity diagram but not in a location where it is desired. There is no simple way to shift or adjust the partitions so that it can group the desired activities together. A new activity diagram had to be created with the partitions inserted first before placing the appropriate activities in the desired partitions. Once the activities are placed in the swim lanes it became very simple to allocate the partitioned activities to components represented in the block definition diagrams.

There are also issues that pertain to the integration of different versions of diagrams. The continual development and modification of the EDD system models resulted in multiple versions of the same diagrams. The intention is to update and improve the system model and to verify the design with the subject matter experts. This led to the need to establish an approach to manage the configuration of the system models. The approach to maintain configurations had to be simple because there were no specialized configuration management resources available. The approach is to assign a centralized repository for the model and to assign revision numbers for each model baseline. Each update to a baseline model is uploaded to the online repository to centralize the storage location. Each approved update to the model also resulted in an increment of revision number. This approach regulated the updates to the model and ensured that the appropriate changes incorporated in to the final baseline.

Another major challenge encountered is to develop an executable model for the EDD Systems. Parametric diagrams were needed in order to make an executable SysML model and perform a trade study of cost and performance. From research of the SysML Tool and discussion with the other SysML team, converting the SysML model into an executable model requires extensive knowledge in JAVA programming as well as extensive knowledge of model-based programming. The time and effort required to fully understand the intricacies of developing an executable model far exceed the scope and resources available in this project. However, the benefits of developing an executable model should not be overlooked.

The benefits of the executable model include trade study analysis. An executable model can be used to perform sensitivity analysis for parameters such as cost and performance. Other parameters that could be analyzed in an executable model include reliability, safety, and weight. After all the analysis is performed with the baseline system, another model can be created with an alternative component and made executable. The results of the sensitivity analysis performed on the alternative component can then be compared to the baseline component for a formal trade study. The trade study performed with the executable model would become criteria for the company's formal decision analysis.

4 Trade Study with Net Present Value

The benefits of the alternative systems are speed and a simplified process. The trade offs to the alternative systems are higher operational and maintenance costs. In this section Net Present Value (NPV) is used to determine the financial implications of the two alternatives systems relative to the baseline. The major question is: Can the increased processing speed of the alternatives overcome the high operational and maintenance costs? The NPV model was used to characterize the cash inflows (revenue) and cash outflows (costs) for each alternative over three years. The result was a set of probability density functions (PDFs) that describe the net present value for each system.

4.1 Method

The revenue and cost streams were decomposed into basic model inputs. The inputs were defined using a mix of data mined from a database of historic data and staff expertise. Once the inputs were properly determined, a Monte Carlo simulation was run to generate NPV probability density functions for each alternative. Finally, an analysis of the resulting distributions was conducted to generate recommendations on each of the alternatives to the client.

The formula for Net Present Value is summarized below.

$$\mathrm{NPV} = \sum_{t=1}^{n} \frac{C_t}{(1+r)^t} - C_0$$

Where:
t – time
n – total project time
r – discount rate
Ct – net cash flow
Co – Initial capital expenditures at time zero

The net cash flow (Ct) parameter forms the foundation of the comparative NPV model. The net cash flow is decomposed into net cash inflows (revenue) and net cash outflows (costs). The revenue and cost parameters can be further decomposed into individual model inputs. The inputs and their inter-relationships were developed with the expertise of the client side financial and process experts.

The NPV model was used to project revenue and cost streams for each alternative over a three-year period. The model was run through 25,000 iterations for each alternative. Annual results were calculated by segmenting each year into 12 monthly estimates. Monthly estimates were derived from a distribution of monthly cash flows, which was derived from the initial model inputs.

The NPV study focuses on the ability of the alternative systems to generate enough revenue to compensate for the higher costs. In this model the limiting factor is the ability for the EDD processing team to accept projects into the system. The more jobs entered into the system, the more revenue is generated.

The discount rate (r) was held constant at 1.10 for each alternative over the three-year period.

4.2 Model Input Decomposition

The input variables and their relationships can been seen in the figure below.

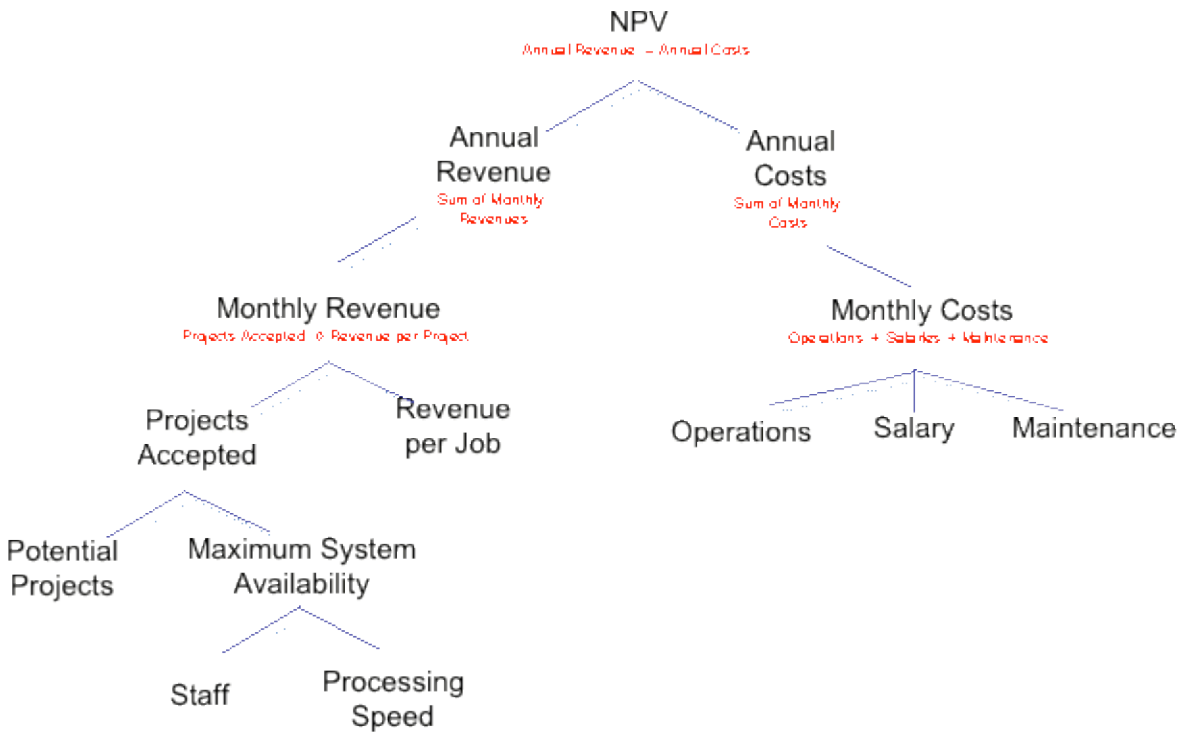


Figure 9 NPV Parametric Diagram

The below is an in-depth explanation of each variable.

Annual Revenue: Annual revenue is derived from the distribution generated from the monthly revenue results. The annual revenue is the sum of twelve monthly revenue estimates.

Monthly Revenue: The monthly revenue is dependant on the number of projects accepted into the system each month and the revenue for each project. The monthly revenue is the sum of the revenue for each job accepted and completed in a month.

Revenue per Project: The revenue per project is the amount of revenue in dollars that a project generates. In practice the revenue per project is dependant on the size and complexity of a job. The revenue for each job is stochastic to reduce the complexity.

Projects Accepted: This value is the total number of projects entered into the system each month. The parameter is based on the maximum availability and the number of perspective projects. This parameter is the key to determining if the alternative systems can generate enough revenue to counter their higher costs. With a higher number of projects accepted the total revenue each month should increase.

Maximum level of System Availability: The maximum level of system availability is the largest number of projects that can enter into the system each month. The availability is a function of two parameters, the number of staff available and the rate at which projects in the system can be completed. The number of projects accepted may not exceed the maximum level of availability. The model will test to see if an increased level of availability will allow the alternative systems to recoup the losses on the cost side.

Number of Prospective Projects: The number of prospective projects describes the number of projects that are available to be entered into the system. The number of prospective projects is linked to the efforts of management throughout Huron to solicit new EDD projects. If the number of projects per month exceeds the system availability, then the project is not accepted and the revenue for the project is lost.

Number of Staff: The number of staff plays a critical role in limiting the number of jobs that can be entered into the system each month. The maximum level of system availability is directly related to the number of trained and capable employees. The number of staff within the group may fluctuate considerably each month. The fluctuation is a balance between the number of current employees, incoming new hirers, and people leaving the team.

Processing Speed: Processing speed describes the rate at which projects can be pulled through the system. The processing speed is a critical element in the model. It works in conjunction with the number of staff to define the maximum availability of the system.

The cash out flows are described below:

Initial Costs: The costs used to procure new software and equipment for the alternative systems at the onset of the migration. The initial costs are incurred once at the beginning of the project.

Maintenance Costs: Monthly costs associated with maintaining the software and hardware systems. The maintenance costs include repairing machines, software upkeep and spare parts.

Salary Costs: Monthly costs related to employee salaries.

Operational: Monthly costs related to procuring additional equipment, software and the overhead costs related to the building and facilities.

4.3 Assumptions

The following assumptions have been applied to frame the scope of the NPV model.

Number of Projects Limitations: The number of projects entering into the system cannot be greater than the maximum level of availability.

Projects Start and Completion Time: All projects started in a month are assumed to be completed within that month. In practice this assumption can be interpreted as larger scale projects are started early in the month while smaller projects are started later in the month.

Minimum Revenue: $2500 is the minimum amount of revenue accepted for a job.

Autonomy Costs: The Autonomy system has an initial cost of $2 million dollars and an operational cost of $250,000 annually.

Attenex Costs: The Attenex system has an operational cost of $500 dollars per GB processed.

Prospective Projects: The level at which prospective projects are found is consistent for all systems.

Availability Parameter: The availability parameter is being used to model the size and availability of the queue for incoming projects.

Pricing Scheme: The pricing scheme is constant for each system over the three-year period. No adjustments have been made to the pricing schemes of the higher cost alternatives.

Migration Costs: With the exception of initial software costs, all migration costs are ignored in this model.

4.4 Results and Discussion

Results were generated by calculating a net present value probability distribution function for each system and then comparing the results. The graph below shows probability density functions for the net present value assuming that the alternative systems will increase the maximum level of system availability by 1.5 times the baseline thus increasing the number of projects accepted into the system.

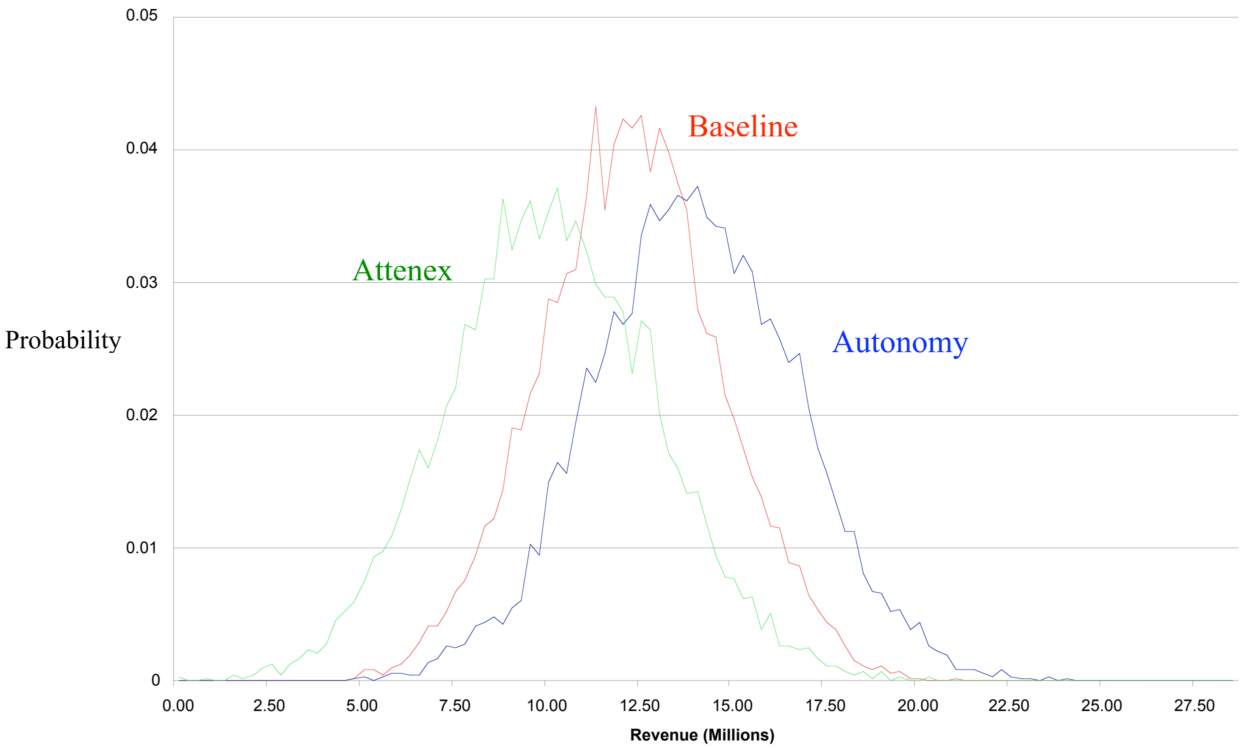


Figure 10: Net Present Value Probability Density Function

The Attenex NPV range is shifted downward below the NPV range of the baseline. The shift suggests that increasing the job availability by 1.5 times will not generate enough revenue to offset the additional operations costs. The Autonomy PDF is shifted slightly higher relative to the baseline, suggesting that the Autonomy system may be able to overcome the higher costs.

The costs structures of Attenex and Autonomy are driving the difference in their NPV distributions. The Autonomy system has an initial cost of $2 million dollars and

$250,000 dollars in maintenance fees each year. The fees are fixed annual costs and can be overcome by increasing the number of jobs accepted into the system.

The Attenex system has an operational cost of $500 per gigabyte processed. The average revenue per Gigabyte is $2,000. The costs of Attenex reduce revenue by approximately 25%. For Attenex to outperform the baseline the higher costs must be recouped by increasing the per gigabyte charge.

The table below shows the average number of jobs accepted per month by year for the three alternatives.

Average Number of Projects Accepted Per Month

	Year-1	Year-2	Year -3
Baseline	12	13	13
Autonomy	14	16	17
Attenex	13	15	15

The table shows that the alternative systems have a definite edge on the number of jobs brought in each month. The two alternatives' average number of projects in year-1 exceed the baselines; average in year-3. The critical difference is that the increased number of jobs is not enough to recoup the high costs related to each system.

The following graph shows the net present value PDFs if the alternative systems can increase job availability by two times the baseline thus further increasing the number of jobs accepted each month.

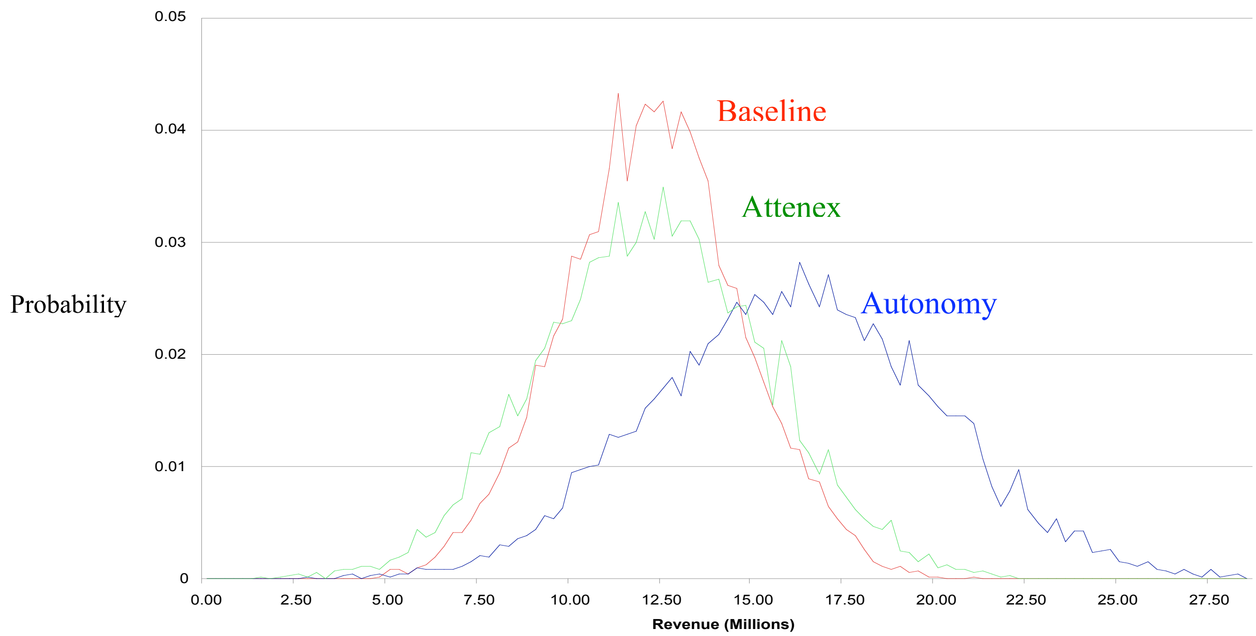


Figure 11: Net Present Value Probability Density Function

The Autonomy systems' NPV distribution is being pulled out into a higher range relative to the baseline and Attenex systems. The PDF for the baseline and Attenex systems are nearly overlaid. By increasing the number of jobs accepted each month the Autonomy system is taking advantage of revenue that would be lost in the baseline system. By harvesting the previously lost revenue the NPV of the Autonomy system is beginning to separate from the baseline and from Attenex.

The table below details the average number of jobs accepted each month. In this table the availability parameter of the alternatives is set to two times the baseline.

Average Number of Projects Accepted Per Month

	Year-1	Year-2	Year -3
Baseline	12	13	14
Autonomy	15	17	17
Attenex	15	17	17

The model suggests that the target range for average jobs accepted each month is in the mid-to-high teens. By maintaining a higher level of jobs accepted into the system the Autonomy system is being pulled into a higher range of profitability. The Attenex system continues to lag due to its pricing mechanics.

The model output suggests that the costs of the Autonomy system can be overcome by increasing the number of jobs accepted into the system each month. The higher revenue can offset and overcome the higher costs making it potentially more profitable than the baseline. For the Attenex system to overcome its costs the pricing structure charged to the client must be raised to compensate for the increased per gigabyte charge.

4.5 Conclusions

The alternative systems gain processing speed but suffer from high operations and maintenance costs. The NPV analysis indicates that under the models assumptions the Autonomy system will begin to show signs of increased profitability only if it can increase the number of jobs accepted into the system. A word of caution, even when the maximum level of availability is twice the baseline there is still a reasonable chance that the Autonomy system will only be as profitable as the baseline. The Attenex system is hamstrung by the $500 per Gigabyte processing cost. The revenue for each job in the Attenex system is cut by approximately 25% with respect to the baseline.

There is hope in the alternative systems! The nature of the EDD market demands that vendors be able to adapt quickly to new kinds of data and processes. A single EDD process will not fit the dynamic market demand. The strength of the baseline system is its adaptability and its robustness. Each stage can be tweaked and adjusted to meet client demands.

The Autonomy system is ideal for an environment with standard high volume data. In that environment the uncertainty of the type of data being delivered is low and the speed of the software can be exploited. The Autonomy software also provides excellent methods to connect the EDD processing group and the EDD review groups, a factor outside the scope of this study. The Attenex software has processing speed to match Autonomy. The per gigabyte cost makes it ill suited for a full migration. Attenex can be used for clients that can bear the burden of higher costs to receive data quickly.

4.5.1 System Summary

The Baseline System: The Baseline system may remain in place but in a decreased role. The process has excellent potential because of its flexibility and adaptability. It is advisable to develop new pieces of software and scripts to reduce the amount of human interaction with the system. The alternative systems gain traction by providing a single piece of software to execute multiple functions. The baseline requires significant human interaction to accomplish the same tasks. By creating new scripts the baseline can be made to mimic the more expensive alternatives while maintaining the flexibility in components.

The Autonomy System: The Autonomy system provides high speed at high fixed costs. The $2 million initial cost for the software and the $250,000 maintenance fee are steep. Fortunately through the NPV analysis it was shown that the Autonomy system could be profitable if the number of projects accepted is sufficiently increased.

The Attenex System: The benefits to the Attenex system are its speed and its zero initial cost. The downside is the $500 per gigabyte processing charge. The analysis shows that Attenex can process data faster than the baseline but it cannot outperform the baseline in terms of profitability under the same pricing structures. In order for Attenex to increase its profitability the pricing structure Huron offers its customers would have to increase to offset the operational costs.

5 SysML Evaluation

Aerospace Corporation is a federally funded research and development center that provides technical expertise to various US National Security Space programs. These space programs are complicated systems that require insight into and tradeoffs between different engineering disciplines. One of the jobs of systems engineers is to find an acceptable balance between the necessary tradeoffs in areas such as performance, cost, and schedule. Systems Modeling Language (SysML) is a language designed to help model systems. Conceptually, SysML has the potential to help Systems engineers effectively model a complicated system and link requirements to applicable subsystems. However, SysML is largely untested in today's industry. Aerospace Corporation is interested in determining if SysML as a language was mature enough to begin using in industry. For this reason, Aerospace Corporation asked the Document Optimization group to use SysML to model a system and provide feedback on the modeling language. Additionally, the Document Optimization group was asked to evaluate Rational System Developer, a tool that implements SysML. The Document Optimization team used SysML, as implemented by IBM Rational System Developer, to model and help evaluate the different choices for the Huron Consulting Group.

The purpose of the survey was to evaluate SysML as a modeling language for designing systems. Specifically, the Document Optimization team wanted to evaluate SysML maturity and determine if SysML is a useful tool for systems development. Additionally, the Document Optimization team wanted to evaluate IBM Rational System Developer and determine if it supports SysML usage.

5.1 Decomposition Hierarchy

To support the evaluation, the Document Optimization team decomposed the evaluation criteria into a Multi-Attribute Utility Assessment Evaluation Hierarchy. The Document Optimization team based the decomposition and survey questions on the Handbook for Evaluation Knowledge Based Systems, by Adelman and Riedel. The decomposition was modified for the purposes of the project and the tools being used. Figure 12 shows the evaluation hierarchy.

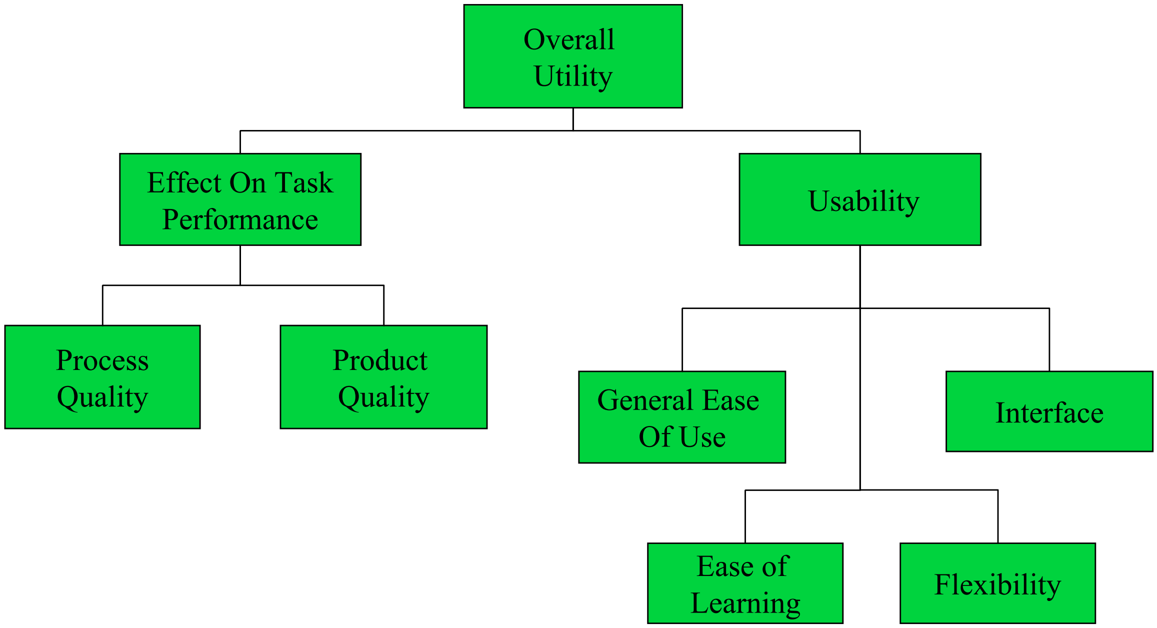


Figure 12: Evaluation Hierarchy

The overall question of SysML utility was broken down into two categories; Effect on Task Performance and Usability. These two categories were further broken down into subcategories. Effect on Task Performance was broken down into Process Quality and Product Quality. Usability was decomposed into Ease of Use, Interface, Ease of Learning, and Flexibility. Table 1 provides a brief definition of each of the terms.

Table 1: Evaluation Definitions

Evaluation Term	**Definition**
Overall Utility	Questions in this section focus on overall impressions of the usefulness of SysML / Rational System Developer. Includes data from lower levels of the hierarchy.
-Effect on Task Performance	Focuses on determining the overall positive or negative effect that SysML and Rational System Developer have on overall task performance.
--Process Quality	Questions focus on whether SysML / Rational System Developer is faster than the current methods.
--Product Quality	Questions focus on whether the resulting quality is adequate.

-Usability	Focuses on determining the overall usability of SysML / Rational System Developer.
--Ease of Use	Measures whether participants feel SysML / Rational System Developer is difficult or easy to use.
--Interface	Questions focus on the Rational System Developer interface and navigation in the tool
--Ease of Learning	Focuses on whether SysML / Rational System Developer is difficult or easy to learn.
--Flexibility	Measures the adaptability of SysML and how language can represent changing designs.

The purpose of decomposing SysML utility is to help focus the rating on specific segments of the SysML language and Rational System Developer. This approach provides insight into the different aspects and highlights areas that users consider shortcomings. This provides understanding as to which areas of the SysML language may be ready for use in industry and which areas may need to be mitigated by additional training or further development of the SysML language. The entire survey is provided as Appendix A.

5.2 Evaluation Approach

All the members of the Document Optimization team filled out the evaluation survey. Another group in the class, the Tactical Surveillance Satellite (TSS) team, also used SysML and Rational System Developer and agreed to fill out the survey to provide additional feedback.

Each team member filled out some basic information, including group project, software development experience, Unified Modeling Language (UML) experience, and some estimates on number of hours spent using SysML and number of hours it took to become a functional user of SysML. This information was collected because prior experience with UML may make SysML easier to use due to the commonality of between the languages. Conversely, little or no experience with UML may prompt different ratings on learnability and ease of use. Software development experience could also skew the results.

The survey consisted of forty-one questions. Each question was phrased as a statement and participants were asked to rate their agreement on a 1 to 5 Likert scale, from strongly disagree (1) to strongly agree (5) with 3 indicating neither agree or disagree. The questions were all positively worded, so that higher ratings were consistent with a better system.

5.3 Evaluation Results

The ratings were averaged between the 8 participants in both groups to get average scores for the different areas. Scores for subcategories were averaged with top-level ratings to provide ratings for Effect on Task Performance and Usability. These were further compiled together to determine an Overall Utility. Table 2 shows averages from questions pertaining to SysML. Table 3 shows averages for questions pertaining to Rational System Developer. Some areas did not have questions that pertained to either SysML (such as interface) or Rational System Developer (such as process performance). The results are further discussed below.

Table 2: SysML Ratings

SysML	Average	TSS	EDD
Overall Average Utility	3.7	3.7	3.7
Overall Questions	4.1	4.2	4.0
Effect on Performance	3.6	3.8	3.5
Process	3.6	3.8	3.6
Product	3.5	3.8	3.3
Usability	3.4	3.3	3.6
Ease of Use	3.8	3.9	3.8
Learnability	2.4	2.3	2.5
Flexibility	3.7	3.4	4.0

Table 3: Rational System Developer Ratings

Rational	Average	TSS	EDD
Overall Average Utility	3.5	3.3	3.7
Effect on Performance	3.5	3.4	3.6
Product	3.5	3.4	3.6
Usability	3.5	3.3	3.7
Interface	3.4	3.3	3.5
Ease of Use	3.8	3.9	3.8

Learnability	2.8	2.8	2.9

When presented with questions about the overall utility, participants rated the SysML language 4.1, indicating general satisfaction with SysML as a valuable tool and efficient approach for system design. There was a small difference between the two teams, with the TSS teams average rating 0.2 higher than the Document Optimization team. Individual rankings ranged from about 3.6 to 4.4. Participants with software development experience and/or UML experience tended to rate SysML overall utility marginally more positively.

Process Quality and Product Quality, which measure the positive or negative effect on task performance, were rated between 3.5 and 3.6 for both SysML and Rational System Developer. This indicates that participants viewed these aspects as positive to a lesser extent than their view of the overall utility.

Questions about SysML overall usability averaged a rating of 3.9 across both teams. However, the Document Optimization team rated the overall SysML usability as 4.3, which the TSS team rated it as 3.5. Similarly, questions about Rational overall usability averaged a rating of 4.0. Again, however, the ratings provided by the Document Optimization team were 4.8 versus the 3.3 given by the TSS team. This could be due to the difference the usage of SysML and Rational Systems Developer between the two teams. The Document Optimization team used SysML and Rational System Developer to model and evaluate a process for Huron. A cost model in Excel was used to perform the trade study and sensitivity analysis. The analysis was performed in Excel due to time constraints. The TSS team used SysML and Rational System Developer to model part of a satellite and used parametric models and equations in SysML to perform a trade study. Presentations by the TSS group indicate that they found executing a parametric model in SysML both time consuming and difficult.

The differences in ratings between the two teams diminished when presented with decomposed questions pertaining to the interface, ease of use, learnability, and flexibility. SysML and Rational System Developer ratings were similar between the two teams.

Questions that measure how easy SysML and Rational System Developer are to learn are the only ones that were ranked below three. This indicates that both teams felt that SysML, while a good language, was difficult to grasp. Similarly, Rational System Developer received low ratings, evidence that both groups felt the time required to learn Rational System Developer was a weakness for the tool.

5.4 Survey Limitations

The evaluation survey is limited primarily by the small number of participants. While a group of 8 students provides a quick look at aspects of SysML and Rational System Developer, a larger number could help further evaluate if prior exposure to UML or experience in certain fields beyond software development affect the rating given to different aspects of the decomposed evaluation.

Another limitation is the variance between the Document Optimization and TSS projects. The Document Optimization project was to use SysML to model and evaluate a complicated process with little hierarchy and evaluate three different implementations. The TSS project was to model part of a satellite and show how SysML parametric models could help make various tradeoffs. The variance in types of project added variability to the results.

5.5 Conclusions

Overall, respondents felt that SysML was a good language for modeling systems. SysML was rated positively on flexibility; SysML allows users to model a variety of systems and to modify the design while working on the system. Ease of use was also rated positively. Many of the remaining measures centered around 3.4 to 3.6. The exception was learnability, which was rated significantly lower than all other measures.

Similar results can be seen in the Rational System Developer results. When measured without the learnability taken into account, Rational System Developer was rated positively for usability. Ease of use was also scored positively. The main area that received poor ratings was the learnability.

6 Final Recommendations

6.1 Recommendation to Huron Consulting Group

A combination of the baseline and the alternatives is preferable. No process studied is ideal as a full solution. The approach should be to target specific processes at specific projects. Clients that opt to incur higher rates for Attenex can have their projects completed more quickly compared to the other systems. For clients that can tolerate a 1-2 day lag the baseline is ideal.

The Autonomy software should only be purchased if the costs can be distributed across the three groups within Huron's electronic discovery chain. This study considered the impact to the processing group, the second stage in a three-staged chain. If the costs of Autonomy can be distributed across the three teams the initial and operational costs incurred by each individual team is reduced. All three groups would be responsible for recouping the investment instead of a single group within the chain. The benefits of the software can touch each group in the chain making the aggregated EDD process smoother.

The Baseline system can flourish with improvements. The baseline system must be updated with scripts to handle the data transfers between processes. Scripts and software can automate tasks that are a burden on the engineers' time for example creating scripts that can distribute data onto processing machines, configure Adlib Express Settings and handle basic quality control operations. The key is to have a dedicated software engineer tasked with developing new software.

6.2 Recommendation to Aerospace Corporation

Rational System Developer supports the SysML language and can be used to model systems and maintain consistency and requirement links across different models. However, both groups found SysML and Rational System Developer difficult to learn. Aerospace should be aware that a migration to SysML would require investment in training and time before a benefit is seen. Additionally, SysML may not be practical for smaller systems with limited complexity where the analysis can be performed in Excel. The real benefits of SysML will likely be seen in complicated systems with significant hierarchical decomposition such as systems in the National Security Space arena, which Aerospace works with.

7 Future Work

If more time were available to continue developing the project, the first goal would be to update the SysML model to the point where it would become executable. First, the constraints need to be well defined in the component block diagrams. The allocations would have to be double-checked to ensure each diagram is properly linked with each other. The ports, which specify interaction between blocks and parts, would have to be made consistent; i.e. same kind, type, direction, etc. Encapsulation would have to be ensured between blocks and internal blocks and between activities and sub-activities. The parametric diagrams would then be the inputs for the Simulation Toolkit in order to make the model executable.

If extra time permitted, a state-machine could be designed to make a simulation of the document process itself. A state machine is typically used to represent the life cycle of a block. State machines support event-based behavior, which is generally asynchronous. The UML portion would then translate the state-machine diagrams into executable code. The state-machine would only be needed to simulate the document process to find out the time needed to process a job in each of the three alternatives.

Time would then be required to learn the Rational System Developer Simulation toolkit and Java or C++. The SysML model would then have to be manipulated by both the Simulation toolkit and a programming language to make the model truly executable. As found with the TSS team, there are plenty of caveats that need to be followed to make the model executable, i.e. no spaces in any of the SysML element names. A recommendation would be to ensure at least two team members have very knowledgeable programming skills.

In addition to the task of making the SysML model an executable model, future students can use the SysML survey as a starting point to further measure the utility of SysML and Rational System Developer. The survey results can be made cumulative with each semester SysML is used in the Research Project course. The survey would then be taken by a potentially larger student population and the survey results then can be further analyzed. Future professors of this course can distribute the survey to SysML users and keep the surveys for a team that is willing to analyze all of the results.

8 Acknowledgements

The Document Optimization team would like to acknowledge and thank several people who helped us throughout the project. We would like to thank Heather Howard, Shana Lloyd, and Julie Street, from Aerospace Corporation, for their continual involvement and guidance throughout the course of this project. Their comments and suggestions proved invaluable to the final product. Heather Howard, specifically, provided helpful recommendations on the project direction and our presentations. Additionally, we want to thank Chris Genter, from Huron Consulting Group, for allowing us insight into Huron's document review process so that we could use it for our project. We would also like to recognize Professor Laskey for her mentorship throughout the project. We will graduate as Systems engineers with more clear and professional presentation skills as a result of Professor Laskey's recommendations and advice. We would also like to give thanks to Professor Adelman for directing us to some of his previous work in system evaluations. We are also thankful for the time that Sanford Friedenthal took to give us a SysML training session. Lastly, we would like to acknowledge David Alexander, Kevin Sadeghian, Siroos Sekhavat, and Tom Saltysiak, from the TSS team, for filling out the survey providing us with additional responses.

9 References

Attenex Patterns E-Discovery Software.
http://www.Attenex.com/products/eDiscovery/

Autonomy
http://www.Autonomy.com/content/home/

SysML Spec
http://www.omg.org/cgi-bin/doc?ptc/06-05-04

OMG
www.omgsysml.org

SysML Tutorial
http://www.omgsysml.org/SysML-Tutorial-Baseline-to-INCOSE-060524-low_res.pdf

Adelman, Leonard and Riedel, Sharon L.. Handbook for Evaluating Knowledge-Based Systems, Concepts Framework and Compendium of Methods, Kluwer Academic Publishers, Boston, 1997.

Adelman, Leonard, Rook, Frederick W, and Lehner, Paul E.. “User and R&D Specialist Evaluation of Decisions-Support Systems.” IEEE Transactions of Systems Man, and Cybernetics, Vol. SMC-15, No. 3, May/June 1985.

Appendix A – Evaluation Survey

The questions that correspond to each area of utility are listed below.
Overall Utility: 1, 2, 3, 4, 7, 8, 9, 10
Process Quality: 5, 6, 11, 12
Product Quality: 13, 14, 15, 16, 17, 18
Usability: 19, 20
Interface: 21, 22, 23, 24, 25, 26, 27, 28, 29, 32
Ease of Use: 30, 31, 40, 41
Ease of Learning: 33, 34, 35, 36, 37
Flexibility: 38, 39

EDD SysML Survey
OR680, Spring 2007

This survey is intended to gather your opinions of the SysML language and Rational System Developer tool. For the questions on the following page, please circle 1-5 or N/A (not applicable) for each question.

Name: ___________________

Group Project: ________________

Software Development Experience (circle one):

Limited | moderate | advanced

Previous UML Experience (circle one):

minimal experience | moderate | advanced/currently user

What is your current modeling language / process?
(e.g., None, Power Point, UML, etc): _________________________

Estimated Hours Spent using SysML: _______

How many hours did it take to become a functional user of SysML? _______

How many hours did it take to become comfortable navigating the System Developer tool? _______

Overall System Utility

1. SysML is a valuable tool for modeling.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

2. Overall, SysML supports an efficient approach for system modeling.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

3. Use of SysML improves the system design process.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

4. SysML allows a designer to gain insight into the system design and component interaction.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

5. The SysML diagrams available were adequate for my project.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

6. SysML helped clarify the project processes.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

7. SysML is a valuable tool for modeling software systems.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

8. SysML is a valuable tool for modeling hardware systems.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

9. SysML is more useful than my current modeling language.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

Effect on Task Performance

10. I would recommend using SysML for work-related design projects.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

11. Overall, SysML is a useful approach for optimization.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

12. Modeling a system with SysML is faster than the current process (i.e., Power Point).					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

13. I would feel comfortable using SysML under schedule pressure.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

14. Overall, SysML provides me with useful results.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

15. I found using SysML for modeling my project resulted in acceptable results.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

16. Using SysML to model my project improved the quality of my work.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

17. Using Rational System Developer, I was able to easily diagram my design.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

18. Rational System Developer supports the preparation of high quality results.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

System Usability

19. The Rational System Developer diagrams are easy to read.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

20. The SysML diagrams are easy to understand.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

21. Rational System Developer uses the same layout for all screens.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

22. Rational System Developer presents the information in a consistent way.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

23. The same commands produce the same actions throughout Rational System Developer.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

24. It is always clear where the user is in Rational System Developer.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

25. It is easy to navigate around Rational System Developer.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

26. The Rational System Developer interface has an intuitive layout.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

27. I can easily add model elements to the project in Rational System Developer.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

28. Rational System Developer provides feedback when processing user commands.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

29. Rational System Developer provides the user with effective directions so that I know what to do next.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

30. SysML is easy to use.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

31. Rational System Developer is easy to use.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

32. The Rational System Developer display diagrams are easy to read.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

33. Rational System Developer has sufficient help features.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

34. One can learn SysML in an 8-hour training session.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

35. SysML was easy to learn.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

36. A person without a SysML background can understand the models with minimal additional instruction.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

37. Rational System Developer is easy to learn.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

38. SysML allows for adaptation to different scenarios.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

39. SysML allows me to see the effects of changes throughout the model.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

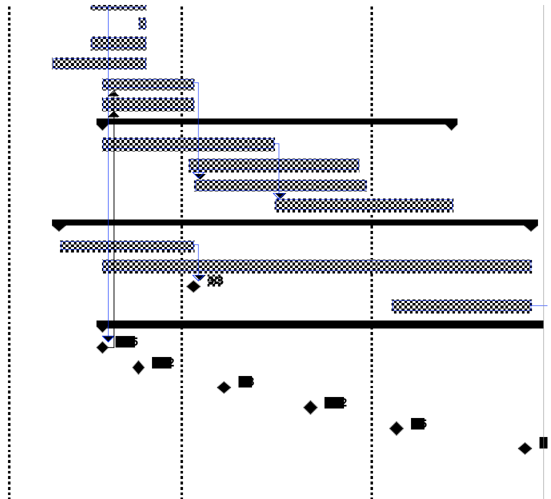
40. Modifying SysML diagrams is easily accomplished.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

41. Rational System Developer allows me to easily modify my diagrams.					
Strongly Disagree				Strongly Agree	
1	2	3	4	5	N/A

Thank you for completing the survey.

Appendix B – Project Schedule

8	Contact Aerospace	1 day?	Thu 2/22/07	Thu 2/22/07
9	Determine a day to visit Huron	7 days	Wed 2/14/07	Thu 2/22/07
10	Research Background Literature	11 days	Thu 2/8/07	Thu 2/22/07
11	Research Autonomy	11 days	Fri 2/16/07	Fri 3/2/07
12	Research Attenex	11 days	Fri 2/16/07	Fri 3/2/07
23	**Model Processes in SysML**	**42 days**	**Fri 2/16/07**	**Fri 4/13/07**
24	Implement Model of Current Process in SysML	21 days	Fri 2/16/07	Thu 3/15/07
25	Implement Attenex Model in SysML	21 days	Fri 3/2/07	Thu 3/29/07
26	Implement Autonomy Model in SysML	21 days	Sat 3/3/07	Fri 3/30/07
27	Optimize Models and Perform Trade Analysis	21 days	Fri 3/16/07	Fri 4/13/07
28	**Track SysML Issues and Metrics**	**56 days**	**Fri 2/9/07**	**Thu 4/26/07**
29	Learn SysML	16 days	Fri 2/9/07	Fri 3/2/07
30	Use SysML	51 days	Fri 2/16/07	Thu 4/26/07
31	SysML Training	0 days	Sat 3/3/07	Sat 3/3/07
13	Develop Final Presentation	17 days	Wed 4/4/07	Thu 4/26/07
14	**In Class Deliverables**	**56 days**	**Thu 2/15/07**	**Thu 5/3/07**
15	Proposal and Presentation	0 days	Thu 2/15/07	Thu 2/15/07
16	Status Report	0 days	Thu 2/22/07	Thu 2/22/07
17	Progress Report	0 days	Thu 3/8/07	Thu 3/8/07
18	Status Report	0 days	Thu 3/22/07	Thu 3/22/07
19	Formal Progress Presentation	0 days	Thu 4/5/07	Thu 4/5/07
20	Dry Run of final Presentation	0 days	Thu 4/26/07	Thu 4/26/07
21	Final Project Presenation	0 days	Tue 5/1/07	Tue 5/1/07
22	Final Report	0 days	Thu 5/3/07	Thu 5/3/07



A Schedule was created in the third week of the semester to measure and track the project's progress. Hard firm dates for the project proposal, two status reports, two progress reports, dry run of final presentation, final report, and final presentation were provided by the course instructor Professor Laskey. All other project- related tasks were monitored and tracked. Plans were created and enforced for any tasks that were not on schedule.

Appendix C - Team Roles and Responsibilities

Team Member	**Roles and Responsibilities**
Chris Catalano	Team Lead EDD subject matter expert. Primary responsibilities included trade study using a net present value model and Huron sponsor interface.
Chun-Yu Chang	Team Member Implemented SysML block definition and use case models. Designed and developed the team web site. Responsible for maintaining the web site with current models, documents, and presentations.
Chris Joson	Team Member Developed the SysML activity diagrams to model the Huron EDD process. Managed the integration of SysML models and final paper sections.
David Matthes	Team Member Primary responsibilities included SysML Multi-Attribute Utility Assessment Evaluation Hierarchy survey and subsequent analysis. Modeled the system requirements in SysML.

Appendix D - Web Site

URL: http://mason.gmu.edu/~cchang7/index.html

A website was created and updated to describe the project to people not involved in the class. The project's sponsors are identified with links to their respective web sites. The project team members are identified with quick biographies. The problem definition, significance, SysML model, and main results are available. An attempt was made to be visually appealing and relative addresses were used for use by the professor and future OR 680 students.

Appendix E - Net Present Value Parameters

Stochastic inputs for all alternatives:

Input Parameter	Yr-1	Yr-2	Yr-3
Staff	U(7,9)	U(8,10)	U(9,10)
Processing Speed	N(1.9, 1.3)	N(1.9, 1.3)	N(1.9, 1.3)
Potential Projects	N(17, 11)	N(18, 11)	N(19, 11)
Revenue Per Job	N(35,000, 60,000)	N(35,000, 60,000)	N(35,000, 60,000)

The table below summarizes the NPV output for each alternative. The alternatives under configuration 1 had an increased job availability of 1.5 times the baselines; configuration 2 was 2 times the baseline.

	Baseline	Autonomy Config 1	Attenex Config 1	Autonomy Config 2	Attenex Config 2
Mean ($)	12,046,680	13,823,510	9,949,792	16,183,098	12,310,707
Standard Deviation	2,441,528	2,748,596	2,840,322	3,742,575	2,983,890

The table below summarizes the mean NPV by year and total.

	Year-1	Year-2	Year-3	Total
Baseline	3,588,734	3,387,927	5,068,447	12,046,680
Autonomy Config 1	2,843,882	4,350,760	6,628,868	13,823,510
Attenex Config 2	2,800,066	2,780,519	4,369,765	9,949,792
Autonomy Config 2	3,633,222	5,035,964	7,513,057	16,182,243
Attenex Config 2	3,593,232	3,453,712	5,264,467	12,310,707