Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

2011 ISPA/SCEA
Joint Annual Conference & Training Workshop
Albuquerque NM

7-10 June 2011

Software Cost Estimation Using A Decision Graph Process:
A Knowledge Engineering Approach

Sherry Stukes and Dr. John Spagnuolo, Jr.
Jet Propulsion Laboratory, California Institute of Technology

Abstract

At The California Institute of Technology/Jet Propulsion Laboratory, methodologies for Flight Software (FSW) cost estimation and documentation are determined that allow for efficient concurrent and consistent analysis within a tight schedule constraint. This knowledge is structured or "engineered" to facilitate the implementation of FSW cost estimation by others who wish to serve as practitioners in the field.

Knowledge Engineering (as defined by Edward Feigenbaum and Pamela McCorduck in 1983[1]) "… is that discipline that involves integrating knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise". Embedded in this definition is the acquisition and structuring of the related information characterizing the knowledge domain of interest. The effort described in this presentation relates to these ideas in 2 specific ways: (1) It gives a decision graph relating to the acquisition, structuring and representation of the knowledge used for the computation of FSW estimates for space missions at JPL and (2) Although we do not fully automate the processes described here, various aspects of the work are embedded in and related to computer activity. Further, the work is done in such a way as to facilitate further automation of its procedures.

We present an overview of FSW cost estimation techniques used for Independent Cost Estimates (ICEs), proposals, and for the validation of Cost Analysis Data Requirements (CADRe) submissions. The aforementioned decision graph illustrates the steps taken in the production of these estimated costs and serves as the basis of discussion for this paper and the corresponding presentation.

General principles for the estimation of FSW are presented using the SEER-SEM computer program as an illustration of these principles when appropriate. A discussion of various Source Lines of Code (SLOC) data sources and their uses for the preparation of the estimates is given as code size is a major driver for software costs. A computerized methodology used to map the SEER-SEM output into the JPL Work Breakdown Structure (WBS) is illustrated. Finally, an "Across the Board" tally of the SEER-SEM runs and their corresponding input is given in a single sheet for a set of several proposals at JPL.

This paper is not a description per se of the efforts by two software cost analysts. Rather, it is an outline of the methodology used for FSW cost analysis presented in a form that would serve as a foundation upon which others may gain insight into how to perform FSW cost analyses for their own problems at hand.

---

[1] The Fifth Generation : Artificial Intelligence and Japan's Computer Challenge to the World, Addison-Wesley , Reading, MA, 1983.

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

Table of Contents

Figures

Tables

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

1   Introduction

Due to the recent and seemingly ever present economic climate, cost estimation at CalTech's Jet Propulsion Laboratory is becoming a crucial part of the mission formulation (proposal) process. Further, the rigor and exactitude of its methods is attaining an importance that is becoming more and more pronounced with the passage of time. Implicit in these analyses is a reliable and accurate estimation of the software costs involved in spacecraft, instruments (payload), simulation and testbeds, ground systems for commanding the spacecraft, and instruments, and science data processing. Such software analysis is also required to support two other kinds of activities: Independent Cost Estimates (ICE's) and Cost Analysis Data Requirement (CADRe) documents.

ICE's are an integral part of the cost verification process to ensure that costs are reasonable. They may be requested by the project, NASA's Independent Program Assessment Office (IPAO) [2] or Division 2x (Costing & Pricing) at JPL or all three entities. They are required at milestone reviews and are performed separately from the project. Depending on how many ICE's are performed, a reconciliation exercise may be conducted in order to understand the differences in content and scope of the estimates. This allows for a single best estimate. During this process, interactions with project personnel are usually discouraged. One or more project independent data sources must be used to derive a software cost estimate.

A CADRe is a report that provides present and future researchers with an encapsulated presentation of the technical and cost data of a project. The project could have already reached 'End of Mission' (EOM), or could be ongoing. A 'Software Metrics Section' in the report is used to categorize the modules of the software as defined by the project. Correspondingly, various parameters relating to the module are listed such as Source Lines of Code (SLOC), programmer and analyst experience, security requirements, multi-site development, work hours etc. These project given parameters (except for work hours) are used in conjunction with a computer program of choice to produce a software cost estimate for the project. The validation of the project given parameters is attained if the work hours produced by the program come close to those given by the project. If not, further project interaction and analysis is needed.

This paper focuses on the work done in computing the FSW costs for N_0 proposals done in the Engineering Costing Analysis Group of Section 2x. The techniques embedded in this work overlap considerably with those used for ICE's and CADRe's but differ in the sense that the work had to be done quickly and for many missions at once. It was therefore imperative that certain techniques and procedures had to be developed which not only streamlined the flight software analysis process but which also provided instantaneous confirmation that the data and processes used for these estimates was consistent across the board.

The above discussed execution of software cost analyses for so many projects suggested the existence of general patterns that could be followed which were, in effect a part of all software cost analysis. Therefore, aside from presenting the results of the analysis and describing what was done to get them, a high level generalized decomposition and illustration of the above mentioned techniques and procedures in a clear form is presented. Typically, a decision tree is used for such purposes. However, to give the reader insight as to what direction he or she should take for the creation of a cost analysis for a given project, it was decided that a decision tree with all its inherent detail would blur the high level concepts and direction for developing such an analysis. Hence, the embodiment of the implemented considerations took the form of high level directive 'boxes' followed by tree like alternatives given rise to as a result of these 'boxed' directives. The resulting structure will be referred to as a decision graph. In essence, this decision graph represents the structuring of the thought processes and data ac-

---

[2] The main role of the IPAO is to enable the independent review of the NASA's Programs and to ensure mission success.

quisition necessities of SW cost estimates as they were done here. This will be referred to as Knowledge Engineering the estimate.

Formally, Knowledge Engineering (as defined by Edward Feigenbaum and Pamela McCorduck in 1983[3]) "… is that discipline that involves integrating knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise". Embedded in this definition is the acquisition and structuring of the related information characterizing the knowledge domain of interest. The effort described in this presentation gives a decision graph relating to such acquisition, structuring and representation of knowledge as it is applied to the computation of FSW estimates. Although the process is essentially not automated, various aspects of the work are embedded in and related to computer activity. Further, the work is done in such a way as to facilitate further automation of its procedures.

This paper is not only a description per se of the efforts by two software cost analysts. It is also an outline of the methodology used for FSW cost analysis presented in a form that serves as a foundation upon which others may gain insight into how to do FSW cost analyses for their own problems at hand.

2   Flight Software Cost Estimates for N_0 Type X Class Proposals

As mentioned in the introduction, this paper focuses on the development of FSW cost estimates for N_0 Type X class missions at the Jet Propulsion Laboratory of the California Institute of Technology. A Type X class mission is defined by the Type X Announcement of Opportunity (AO) issued by the National Aeronautics and Space Administration (NASA). In accord with this specification, the missions under discussion are of 3 types: Inner Heliosphere, Earth Orbiter, and Primitive Body Encounter.

Aside from the rigor and detail inherent in the execution of such analyses, the work was further complicated due to the relatively short deadlines and the varying schedules and availability of the cost leads for each proposal. This made it difficult if not impossible to get the job done in a timely fashion unless several FSW cost estimates were done simultaneously. Work was done on 1 or more proposals to the greatest extent possible, and then as 'information/personnel for discussion' became available on others, work proceeded to them. Further, when more personnel or data became available for analyses previously initiated, work resumed on them and so on. To maintain consistency in the analyses as well as to facilitate an immediate view of data obtained and data needed at any point of the estimation process, the results obtained at all stages of the work were tabulated in a large Excel™ spreadsheet . In the end, all data used in the computation of all flight software estimates were on this spreadsheet. The sheet thus stemmed as a necessity due to the parallel nature of the work being done.

It became clear to the authors, however, that the above spreadsheet not only served as an encapsulation of the data used in the N_0 estimates. It also helped illustrate the process involved in obtaining such estimates, suggesting a 'decision tree' structure[4] which in essence characterized the flight software cost estimation process used. A drawback is that such a tree structure is somewhat tedious and intricate to the point of hiding concepts and procedures important to the decision making process. To illustrate these fundamental building blocks of the FSW analysis work done here, a variant of the decision tree is used. Boxes giving directives followed by node structures listing the possibilities resulting from these directives are used. Such a structure is, in this paper, referred to as a 'decision graph'. This decision graph structure is more compact and intuitively palatable than a decision tree and expresses high level relationships and concepts to the point of suggesting to the reader how to construct his or her own decision graph for their own estimates. A decision tree would easily follow (see Section 5, "Summary and Future Work").

---

[3] The Fifth Generation : Artificial Intelligence and Japan's Computer Challenge to the World, Edward A Feigenbaum, and Pamela McCorduck, Addison-Wesley, Reading, MA, 1983.

[4] Decision Analysis, Howard Raiffa, Addison-Wesley, Reading, MA, 1968.

A notional view of the full decision graph is given in Figure 1. Details are discussed in subsequent sections. The variation in text coloring in the decision graph corresponds to the breakout of the sections which will discuss it. The corresponding color bars that appear above/below the portions of the graph designate the section numbers that the graph portions are discussed in. Note further that the decision boxes and column nodes are numbered C_i and D_j for ease of discussion.

Even though the necessities of the FSW estimation process 'mothered' the need for a spreadsheet which gave rise to a decision graph, the ensuing discussions will not follow that ordering. The authors feel that the ideas implicit in this paper are best conveyed by discussing the details of the decision graph first and then illustrating the compiled spreadsheet data resulting from the process indicated by the graph.
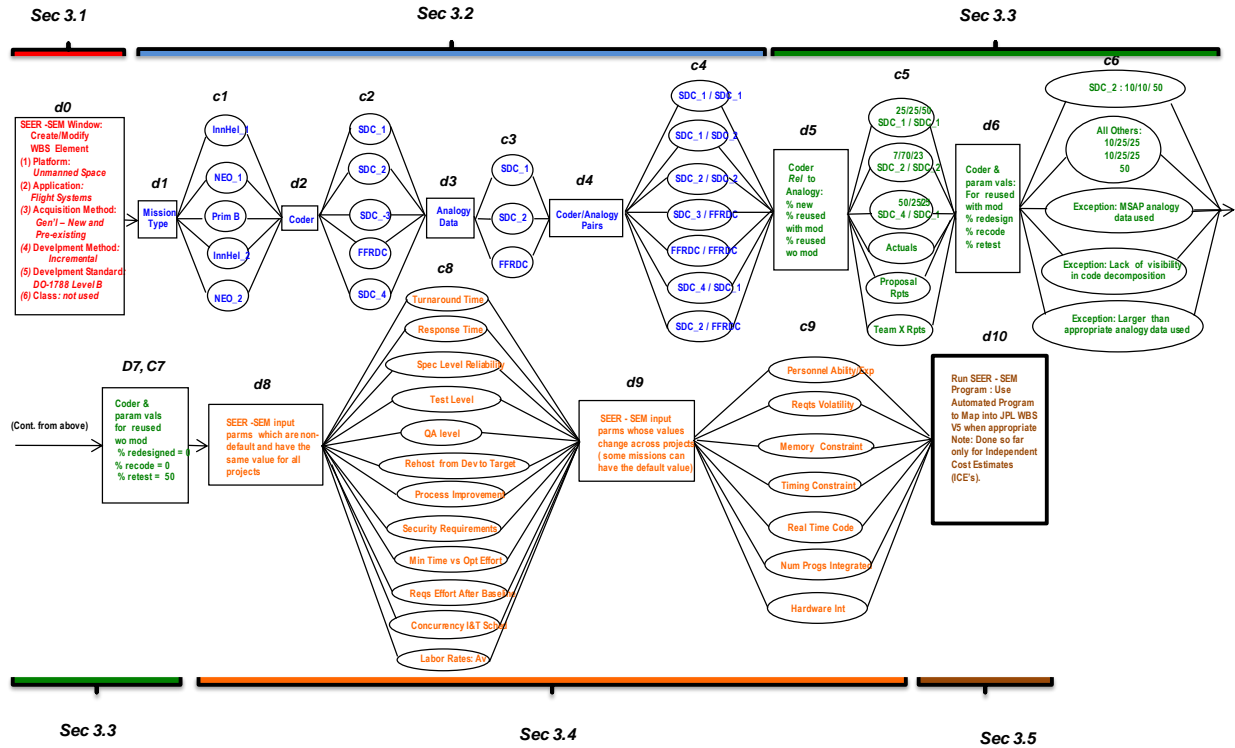
Figure 1: Decision Graph for Flight Software Cost Analysis of N_0 Type X Class Proposals

3   The Decision Graph

As shown in Figure 1, there are five 'color sub-graphs'. Each will have its own subsection in this chapter, be reproduced in a larger form for ease of reading, and discussed in detail.

As often occurs when discussing research with respect to real life data and organizations, much of the inherent information is of a proprietary nature. To allow useful discussion of the issues, the following variable representation of the real life entities are given. Table 1 gives the variable representation of the mission types for which the FSW analyses were done.

Mission Type	Description
Inn_Hel_1	Inner Heliosphere Mission (examples: Venus, Mars)
Inn_Hel_2	Inner Heliosphere Mission (examples: Venus, Mars)
NEO_1	Near Earth Orbiter (examples: Earth , Moon)
NEO_2	Near Earth Orbiter (examples: Earth , moon)
Prim B	Primitive Body Encounter (examples: comets, asteroids)

Table 1: Variable Representation of Real Life Mission Types

In the actual cost exercise, there could exist several missions for each mission type. Variable names for each mission are not necessary for the purposes of this paper.

The organizations responsible for developing the code are roughly of two types: SW Development Contractors (SDC's) and Federally Funded Research Development Centers (FFRDC's). The SDC's are represented by SDC_1, SDC_2, SDC_3, SDC_4 and any FFRDC is simply represented by the acronym FFRDC.

3.1   Establishing Initial SEER-SEM Inputs

Due to the large number of proposals that needed to be estimated in a short time, a parametric approach was used to create the FSW estimates. The parametric model selected and the required input data are described in the following paragraphs.

3.1.1   SEER-SEM Overview

The System Evaluation and Estimation of Resources - Software Estimating Model (SEER-SEM), version 8.0, developed by Galorath Incorporated was the selected for use in the proposal effort because it is widely accepted within NASA and industry and uses as an input the types of data that are available.

The model is based on approximately 6,700 historical data points that are used to create the internal model equations. Approximately 30% (2,000) of the historical programs are based on Commercial environments and the remaining 70% are defense related programs. The model's internal database is significant because it is the basis for the default Knowledge Bases that represent cost drivers for the FSW estimate.

SEER-SEM requires four basic categories of information that represent the input data to the model. These categories include:

- **Software Systems Work Breakdown Structure (WBS)** - Identification of the software modules being developed (to the configuration item where possible, but often times to the software subsystem level).
- **Software Size** - The number of logical source lines of code (SLOC). This includes code that is anticipated to be reused from similar software developments. SLOC may be entered into the model using least likely, most likely, and highest likely values to reflect the uncertainty of the software size.
- **Knowledge Bases** - SEER-SEM contains industry data that is supplemented with related historical data that was used to calibrate, or adjust, the model parameters to reflect historical experience.
- **Parameter Settings** – Parameter settings are initially established by the selected SEER-SEM knowledge bases and have been adjusted to reflect proposal-specific knowledge. Parameter settings may also be entered as least likely, most likely, and highest likely values to reflect uncertainty.

3.1.2   Initial Input Data

Decision box **D0,** shown in Figure 2, identifies the initial categories of information required by SEER-SEM (e.g., Platform, Application, etc.). These categories are referred to as Knowledge Bases (KB's) and the values selected for them (e.g., unmanned space, flight systems, etc.) are the basis for the creation of an initial set of qualitative input parameters for SEER-SEM. A subset of these parameters will be adjusted based upon the procedures and techniques discussed throughout the paper.

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

D0

SEER -SEM Window:
Create/Modify
WBS Element
(1) Platform:
   Unmanned Space
(2) Application:
   Flight Systems
(3) Acquisition Method:
   Gen'l – New and
   Pre-existing
(4) Develpment Method:
   Incremental
(5) Develpment Standard:
   DO-1788 Level B
(6) Class: not used

Figure 2: Initial SEER-SEM inputs

To expand on the KB's, Table 2 provides definitions for the KB's and identifies the selection made for use in establishing the initial model input parameters.

Knowledge Base	Definition	Selection
(1) Platform	Establishes a collection of input parameter settings that characterize a particular host environment.	Unmanned Space
(2) Application	Establishes a collection of input parameter settings that characterize an application or application technology type.	Flight Systems
(3) Acquisition Method	Establishes a collection of input parameter settings that characterize from where the software will come.	New and Reuse
(4) Development Method	Establishes a collection of input parameter settings that characterize the particular Software Development Life Cycle method that will be used.	Incremental Development
(5) Development Standard	Establishes a collection of input parameter settings that characterize the software development process standard that will be used.	DO-178B Level B
(6) Class	A knowledge base calibrated to a specific set of data or domain.	Not used

Table 2: Knowledge Base Definitions and Selections

Aside from the initial inputs established by the SEER-SEM KB, additional data reflecting other facets of FSW cost analysis is required to run the SEER-SEM model. The approach used for obtaining this data included collecting historical data and descriptive information required as input to the model. This process is described in detail later in the paper.

During this initial phase, the software architecture, related new and reused code estimates, knowledge base selections, and parameter setting adjustments, were closely coordinated and reviewed with the technical points of contact.

3.2   Mission Category and Coder/Analogy Data pairs

After the initial inputs are fed to the program, further numerical and quantitative characteristics deemed important with respect to the FSW cost evaluation process have to be determined for each

Computer Software Configuration Item (CSCI). The authors, in every case, found the initial reasoning as given in Figure 3 to be crucial in this respect. In what follows, we discuss the nature of each node structure under its 'decision box' heading.
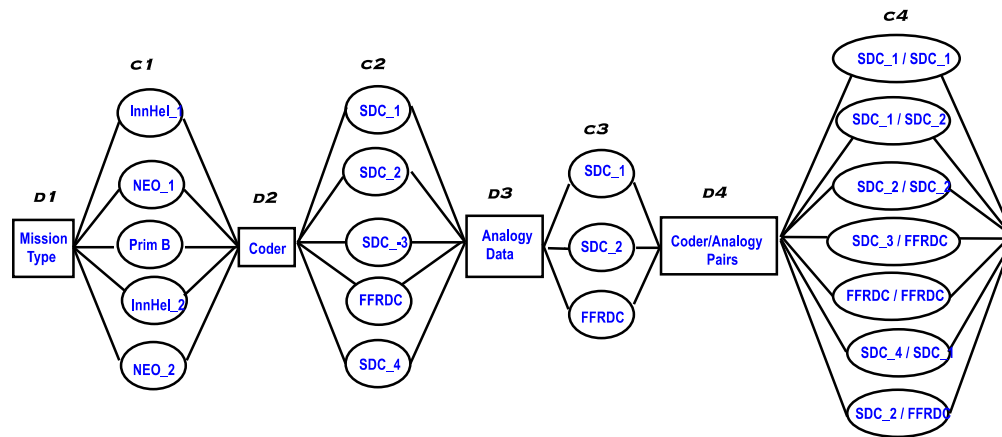

Figure 3: Mission Types, Developers, and Analogy Data Decision Dynamics

3.2.1   Mission Type

The first concern is the nature of the mission. If the project does not give SLOC values for FSW, the estimator will have to locate data to determine approximate SLOC values for the FSW. The SLOC data used depends upon how similar the mission that it was developed for is to the one of current interest. It is therefore important to classify the missions of interest to the level that that similarity can be established. The classifications for the Type X proposals as given in column $C1$ in Figure 3 were inner heliosphere, near earth orbiters and primitive bodies and are listed in according to the variable names presented in Table 1.

3.2.2   Software Developer: SDC or FFRDC

Knowledge of the organization assigned to develop the FSW for the proposal is important because that information, in conjunction with the analogy data available, will determine subsequent numerical SEER-SEM inputs (as discussed in section 3.3 below).

This knowledge is not always known at the beginning of a proposal. Sometimes it changes during the course of a proposal. This can, and does, cause a significant cost change during the estimation process. In the absence of any knowledge of the coder, the analyst and cost lead agree on a best guess as to who the coder might be and the estimate is made with that assumption. The options are listed in $C2$.

3.2.3   Analogy Data

Once the nature of the mission has been studied, the appropriate analogy data must be determined. The analogy data used consisted of code developed by the organizations as listed in $c3$. This data can be obtained by stored samples of code, reports (previous step 2 proposals, for example) or Technical Data Packages. In one case, there was a step 2 report giving actual SLOC values from a previous version of the mission of interest. In another case, there was a Technical Data Package (TDP) for a mission which was deemed very analogous to the proposal of interest. This TDP had SLOC values in it, and these were used. The vast majority of cases, however, required a search for data when the proposal gave inheritance directives without SLOC values or, in fact, when no inheritance directives were given at all. It was then up to the FSW analyst to determining appropriate data analogy sources for SLOC values approximating those that would apply to the Type X proposal at hand.

8

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

3.2.4   *Software Development Contractor/Analogy Data Pairs*

If it is known that Company X is writing the SW, and we have analogy data that Company X developed, then that affords a cost advantage as compared to the case where Company X is doing the FSW and analogy data from Company Y is being used. In the first case, code already exists that company X can use to do the present mission with. Further, having done the code, Company X has experience and infrastructure for that code. In the latter case, even though the Contractor Y analogy data can approximate the amount of code needed, there may be a lot of code and corresponding resources that Company X has to develop that it may not have developed enough to be consummate with Company Y. The coder/analogy pairs shown in the final tree structure indicates those combinations experienced in the Type X proposal experience[5]. The determination of these pairs is important to the FSW cost computation process in ways which will be discussed in the remaining Decision Graph subsections.

3.3   **Quantitative Parameter Determination**

This portion of the decision graph uses the coder/analogy data pairs to determine several sets of numerical inputs to SEER-SEM which, in addition to SLOC values, are major cost drivers. Once the SLOC values are obtained, it is crucial to the cost estimating process to determine how much of the SLOC is new, reused without (wo) modifications (mods) and reused with mods. It is also important to determine, for the code that is reused with modifications, the percentages corresponding to redesign, recode and retest. These 3 percentage categories also apply to code that is reused as is, but in these analyses they are given fixed values of 0%, 0% and 50% for all proposals. Details and justifications regarding the elements described above are as follows.

For purposes of explanation, the triplet:

$$(\text{\% new, \% reused wo mods, \% reused with mods})$$

is referred to as vector 1 and the triplet:

$$(\text{\% redesign, \% recode, \% retest})$$

is called vector 2.

3.3.1   *Coder Relationship to Analogy Data as it Determines Vector 1*

As can be seen in Figure 4, the decision box, *D5*, indicating the need for determination of vector 1 is followed by *c5* which shows several alternative sources for determining the value of this vector as encountered during the proposal cost estimating process. When actual % values for the vector were obtained with analogy data, proposal reports or Team X reports, they were used. In the absence of this data, default values based on cost estimating experience were used. For example, assume SDC_1 was developing the code and the analogy data (SLOC values) used was developed by SDC_1 as well. If the delivered logical SLOC size of the analogy data was X, Then if SDC_1 were to write code for the project , it typically would be approximately the same delivered size, X, as that of the analogy data but would but would be such that:

$$New\ code = 25\%\ X$$
$$Reused\ Code\ wo\ mods = 25\%\ X$$
$$Reused\ Code\ with\ mods = 50\%\ X$$

which gives:

$$Vector\ 1 = (25, 25, 50).$$

---

[5] Note that the number of coder/analogy data pairs shown (7) indicates a subset of all combinations of coder and analogy data possibilities (15).

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

The same reasoning is applied if SDC_2 were the contractor for the S/C and SDC_2 analogy data was used. In this case, based upon FSW cost estimating experience, vector 1 would have the entries:

$$New\ code = 7\%\ X$$
$$Reused\ Code\ wo\ mods = 70\%\ X$$
$$Reused\ Code\ with\ mods = 23\%\ X$$

which gives:

$$Vector\ 1 = (7, 70, 23).$$

Entry of this vector as opposed to the one corresponding to SDC_1 generally results in a lower FSW cost due to the coupling of a lower new code % and a higher reused wo mod code %. This is consistent with the fact that SDC_2 code development is more of a 'production line' as compared to SDC_1's.

In the case where the analogy data used was _not_ developed by the assigned contractor, experience dictates that the % of new code developed would be somewhat larger than the 2 previous cases mentioned. The degree to which this is true depends on the assigned contractor. In the case of SDC_4 being the assigned contractor where SDC_1 analogy data of delivered size X is used.

$$New\ code = 50\%\ X$$
$$Reused\ Code\ wo\ mods = 25\%\ X$$
$$Reused\ Code\ with\ mods = 25\%\ X$$

yielding:

$$Vector\ 1 = (50, 25, 25).$$

As stated above, when actuals are obtained with the vector 1 values, or if available reports give these percentages (sometimes with SLOC values), then the above discussed default reasoning is overridden and those values are used.

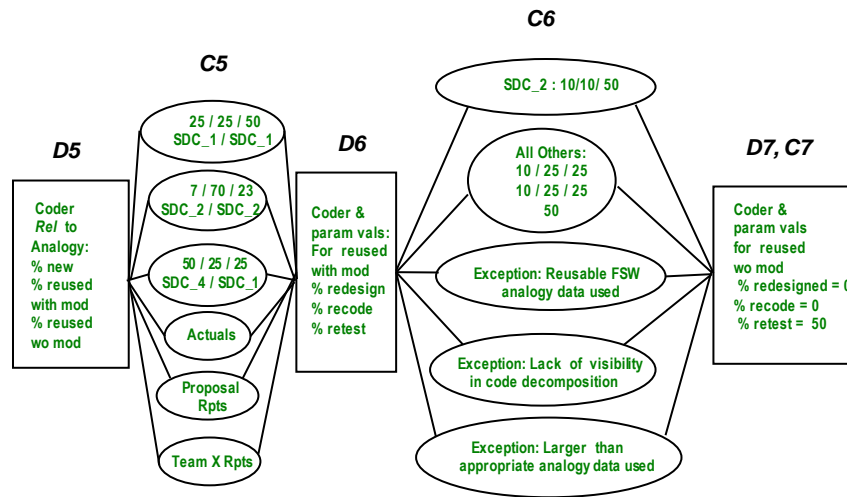Figure 4: Reasoning for Quantitative Input Determination

3.3.2 _Software Development Contractor and Parameter Values as they Relate to Vector 2_

The reasoning involved with vector 2 is computationally and conceptually similar to that of vector 1. In the case of SDC_2, experience with its production line code indicates that, in general, if x represents the amount of reused modified code, then:

$$Redesigned\ code = 10\%X$$
$$Rewritten\ code = 10\%X$$
$$Retested\ code = 50\%X$$

yielding:

$$Vector\ 2 = (10,\ 10,\ 50).$$

In general, for those cases where the SW contractor was not SDC_2 and the analogy data was not SDC_2, each of the entries characterizing vector 2 entries was not given a single value but a range of values[6]. More precisely for the '% redesign' SEER-SEM input parameter, there are 3 values for input into SEER-SEM: A 'Least Likely' value for % redesign, a 'Highly Likely' value for % redesign and a 'Most Likely' value for % redesign. Similarly for the values %recode and %retest. The values used for the non SDC_2 cases thusly are:

$$Least\ likely\ value\ for\ \%\ redesign = 10$$
$$Highly\ likely\ value\ for\ \%\ redesign = 25$$
$$Most\ likely\ value\ for\ \%\ redesign = 25$$

$$Least\ likely\ value\ for\ \%\ recode = 10$$
$$Highly\ likely\ value\ for\ \%\ recode = 25$$
$$Most\ likely\ value\ for\ \%\ recode = 25$$

$$Least\ likely\ value\ for\ \%\ retest = 50$$
$$Highly\ likely\ value\ for\ \%\ retest = 50$$
$$Most\ likely\ value\ for\ \%\ retest = 50.$$

These values were used because they tended to represent the cases where a non-production line FSW contractor (not at the level of experience of SDC_2) was used. Exceptions to this rule included cases where reusable FSW data was used. In that case, lower single valued % values, equal to those used for SDC_2, provided sufficient accuracy. Other exceptions existed as shown in Figure 4.

3.3.3   Treatment of Reused Code without Modification

Finally, SEER-SEM requires inputs for % redesign , % recode and % retest for the code designated as reused without modification as a means to measure how 'New' the code is.

For all missions across the board, it is assumed that that % of code designated as new does not require any redesign or recoding. It however is assigned a value of 50% retest. This is due to the fact that in SEER-SEM 100% retest means that 52% of the code includes the work relating to test plans, test procedures, test drivers, and test scripts. The 48% requires the actual retesting and integration of the code. Again, it is the assumption that this new code does not require the activities which comprise the 52%. It only requires pure testing and integration. Hence 50% was chosen for convenience as it was close enough to 48%.

3.4   **Non - Default Parameter Identification**

This section deals with the assignment of Type X mission values to the SEER-SEM parameters not yet discussed in this paper. Figure 5 gives the decision graph component dealing with this issue.

---

[6]In SEER-SEM, each value for a '% type' parameter can be given a range of three values corresponding to least likely, most likely and highly likely. The fact that previous to this discussion, only one % value was used means that that one value was given to all three possibilities.

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach
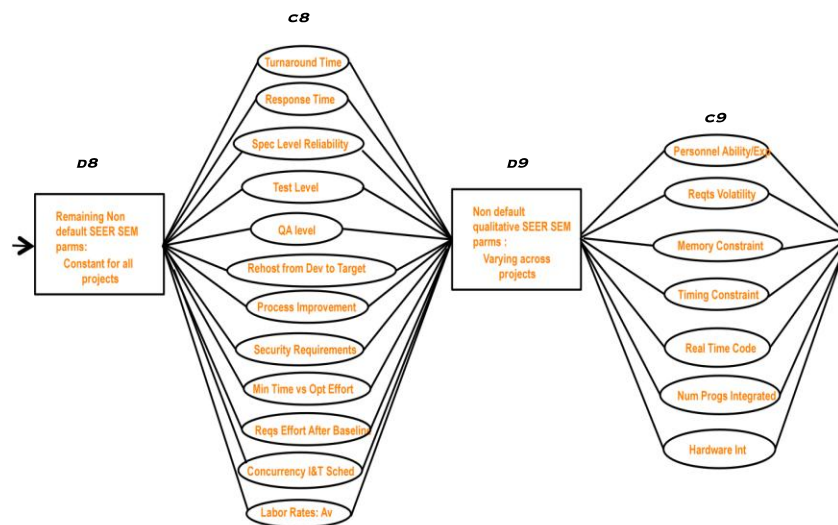
Figure 5: Non Default Knowledge Base Breakout

Decision box **D8**, Figure 5, indicates those input parameters in SEER-SEM for which:

(1) the default values assigned by the program are not appropriate for the Type X missions

and

(2) which can be assigned a value *which is the same for all of the missions*.

The nodes in column **C8** give a listing of all parameters for which this is true.

Decision box **D9** in Figure 5 indicates the existence of parameters whose values varied from mission to mission followed by a listing of those parameters in **C9**.

Any parameters appearing (other than the ones discussed in the sections above) that are not of the types described in this section have the SEER-SEM default values assigned to them. This is the case because, at that early stage in the cost estimation process, it was unrealistic to assign anything else.

Table 3 gives the name and description of those parameters corresponding to the first decision box and the reasons as to why the default values are not appropriate in the missions studied. Further, the table justifies the values assigned in this costing effort.

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

Definition	SEER-SEM Default Value	Reason for Not Using Default	Value Given	Reason for Value Given
The time required to create a release version of the software solution.	LOW-	*Outdated Default Value*	VLO	*More relective of Recent HW/SW Reality*
Rates the average transaction response time from the moment a developer presses a key or click a command, until that command is acknowledged and its action is completed.	NOM+	*Outdated Default Value*	LOW	*More relective of Recent HW/SW Reality*
Rates the level of documentation required. The level of documentation is often dictated by the development standard being used with government contracted software developments.	HI	*Outdated Default Value*	HI-	*More relective of Recent HW/SW Reality*
Rates the rigor and formality of testing for the software or integrated component. Systems intended to be more reliable must be tested more stringently.	HI	*Outdated Default Value*	HI	*More relective of Recent HW/SW Reality*
Evaluates the completeness of the Quality Assurance (QA) activities. The Quality Assurance effort is usually directly related to the impact that a failure in the software would have during its operational phase.	HI	*Outdated Default Value*	HI	*More relective of Recent HW/SW Reality*
Rates the effort to convert the software from the development system to the target system on which the software will execute.	NOM+	*Outdated Default Value*	HIGH-	*More relective of Recent HW/SW Reality*
Evaluates the impact of improving development technology by comparing current, established development practices with those planned for this development.	HI	*Outdated Default Value*	NOM	*More relective of Recent HW/SW Reality*
Rates development impacts of security requirements for the delivered target system. (All classifications are identified in the Orange book.)	HI	*Security default value is too High for the work at hand*	NOM	*Security is Nominal for NASA Unmanned Space Work*
Choose between optimizing the schedule or the effort estimate. Optimizing for schedule (minimum time) assumes the development will be finished as quickly as possible. Optimizing for effort assumes the software will be developed as cheaply as possible, but will take longer to complete.	*Min Time*	*Min Time has inappropriately high cost used only in special time constrained cases*	*Optimal Effort*	*Min Time would yield unallowable and unrealistic FSW costs*
Identifies if software requirements effort should be costed after the software requirements phase is complete.	YES	*Will almost always be a YES response*	YES	*Used default*
The degree of concurrency between the development activities and integration and testing activities. Enter degree of overlap between development and integration.	HI	*Will almost always be a HI response*	HI	*Used default*
The average monthly labor rate for all personnel working on the project.	$28,400 per WM (FY10)	*Used an average of industry and JPL Values*	$xx per WM (FY10)	*Appropriate to use an average of industry and JPL Values*

Table 3: Reasoning for Non-Default Non-Varying Parameter Assignment

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

The following list of parameters from Table 3 requires additional elaboration:

Outdated Default Value – Due to the model not keeping pace with the state-of-the-art in software development.

Security Default Value is Too High – Based on the National Security Agency (NSA) "Orange Book".

Min Time is not applicable in these missions - Min Time has inappropriately high cost used only when there is a schedule constraint.

More Reflective of Recent HW/SW Reality – The default parameters were originally defined for older systems and do not applied to current systems.

Security is Nominal for Unmanned Space Work – JPL does not require high security for their systems.

Min Time Would Yield Unallowable and Unrealistic FSW Costs – Min Time compresses schedule and increases cost.

Table 4 gives the name and description of those parameters corresponding to the second decision box and the reasons as to why the values varied from mission to mission.

Parameter	Definition	Reason for Variation
Personel Ability/Experience	Characteristics of the software development personnel	Coder/Analyst Ability varies from company to company
Requirements Volatility	How frequently the customer changes the software development requirements	Industry varies from JPL
Memory Constraint	Is there sufficient memory to meet the systems requirements	Altered to adjust for lack of visibility in decomposition
Timing Constraint	Can the meet the timing requirement	Varies with GN&C complexity
Real Time Code	The amount of code that requires an instantaneous response	Altered to adjust for lack of visibility in decomposition
Number of Programs Being Integrated	How many CSCIs are concurrently being integrated	Have Different Numbers for Different Projects - Some not Broken out
Hardware Integration	The complexity of interfacing the hardware elements	Altered to adjust for lack of visibility in decomposition

Table 4: Reasoning for Variable Assignments to Parameters

The following list of parameters from Table 4 similarly requires additional elaboration:

Coder/Analyst varies from company to company - Different organizations have different standards their programmers and analysts.

Industry Requirements vary with respect to JPL - The Defense Industry are more stringent than JPL and unmanned NASA projects.

Altered to adjust for lack of visibility in decomposition - Lack of visibility into the breakout of CSCIs required additional adjustments.

Varies with GN&C complexity – Dependent on the type mission (planetary, earth orbiter, lunar, etc.).

Different Numbers of CSCI's for different projects - Some project had more granularity than others based on their financial and engineering requirements.

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

3.5   Program Output Mapping into JPL FSW Work Breakdown Structure

The final decision box $D10$ (see Figure 6) alludes to the fact that when a FSW estimate is done, it is sometimes mandated that the costs be mapped as much as possible into the JPL Work Breakdown Structure (WBS). Although this requirement has thus far been in force only for the production of Independent Cost Estimates (ICE's) and Cost Analysis Data Requirements (CADRe's), it would be of no surprise if, in the future, the mapping were requested for proposals as well. Therefore, due to the coupling of the potential importance of realistic FSW WBS element costs with the fast turnaround time often required, the algorithm for and automation of the mapping from SEER-SEM FSW costs to the JPL WBS is an essential component to effective FSW costing activity.

The construction of a mapping from SEER-SEM to the JPL WBS first consists of choosing the format of the cost output in SEER-SEM. The format should be one which groups the output costs in such a way as to facilitate a clear and direct mapping to the JPL WBS. This is important for the abstract understanding of the JPL cost groupings a well as the practicalities of automating the mapping process. To this end, it was deemed that the "Cost by Labor Category" was the SEER-SEM output of choice. This format not only satisfied the above criteria but also served as a basis for cost analysis and cost comparisons by FSW analysts at JPL for many years. Having made this choice, the task is now to map this output into the JPL FSW WBS. The JPL FSW WBS essentially consists of *FSW management*, *FSW systems engineering*, *FSW testbed*, *FSW I&T* and *Coding Related Activities* (which correspond to the following S/C elements : Command & Data Handling (C&DH) , Guidance Navigation & Control (GN&C) , Engineering Models, Payload & Instrument Control SW, SW Systems Services).

$D10$

> **Run SEER - SEM Program : Use Automated Program to Map into JPL WBS V5 when appropriate Note: Done so far only for Independent Cost Estimates (ICE's).**
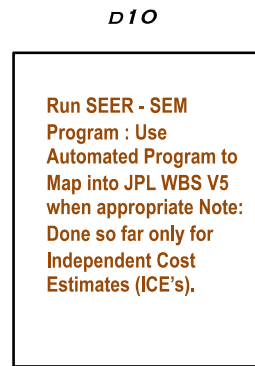
Figure 6: Terminal Decision Box WBS correlation to SEER-SEM Output

For each CSCI (Computer Software Configuration Item) for which SLOC is available , the Cost by Labor Category of SEER-SEM produces column costs which can be grouped into all the above WBS elements except for FSW I&T for which it has a row cost and FSW testbed for which a calculation outside of SEER-SEM is done (see below). Note that the SW costs will have to be mapped into merged S/C elements of the JPL WBS if the SLOC values fed into SEER-SEM representing those S/C elements are correspondingly merged. For example, if a separate breakout of S/C GNC SLOC and S/C C&DH SLOC is not available to the FSW analyst, a breakout of costs into the GNC and C&DH JPL WBS elements is not feasible. Therefore, because these costs will be merged into the SEER-SEM input/output, they will be mapped into a merged WBS category consisting of both GNC and C&DH data. Figure 7 represents the SEER-SEM output and mapping to the JPL WBS for the more extreme (and most common case for the Type X proposals) where only one SLOC value is available for the total of all S/C elements.

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

Figure 8 represents the other extreme, which is more typical of ICE's and CADRe work, where SLOC values are given for each (or at least many) S/C elements. When several arrows with the same color coding from the SEER-SEM outputs merge into one and flow into a JPL WBS element entry (like orange for SW management), that means the cost for that WBS element is formed by taking the sum of the individual costs of the SEER-SEM outputs each of which is enclosed by a box of that same color. Similarly, for I&T, except those costs are enclosed by an ellipse rather than a box for purposes of illustration. Further, note that the SEER-SEM columns which are interpreted as coding related activity (in green) have a one to one mapping directly into the S/C element to which they correspond.

There are some calculations in the illustrated JPL WBS structure for both figures that do not have SEER-SEM as their Basis of Estimate (BOE). The reader will note that equipment and facilities costs appearing at the top of the JPL WBS listings were computed using formulae relating to assumptions on the number of computers used for the coding, average square footage of the coding facilities and composite labor rates. The software testbed cost appearing near the bottom of the WBS listings is computed by taking 4% of the sum of all SEER-SEM costs for all other WBS elements from Project Management through FSW I&T.
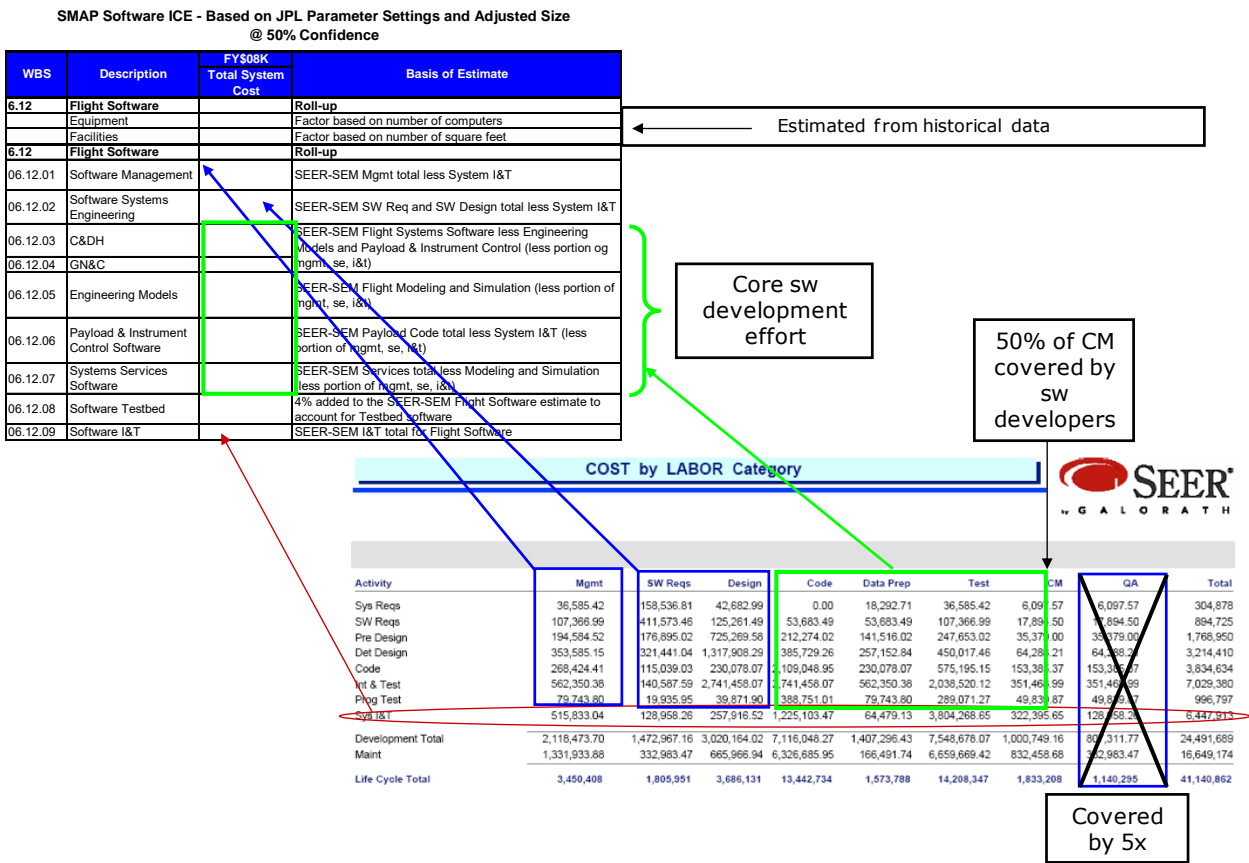

Figure 7: SEER-SEM /JPL WBS Mapping for all SW Elements Combined (Notional Sample Data)
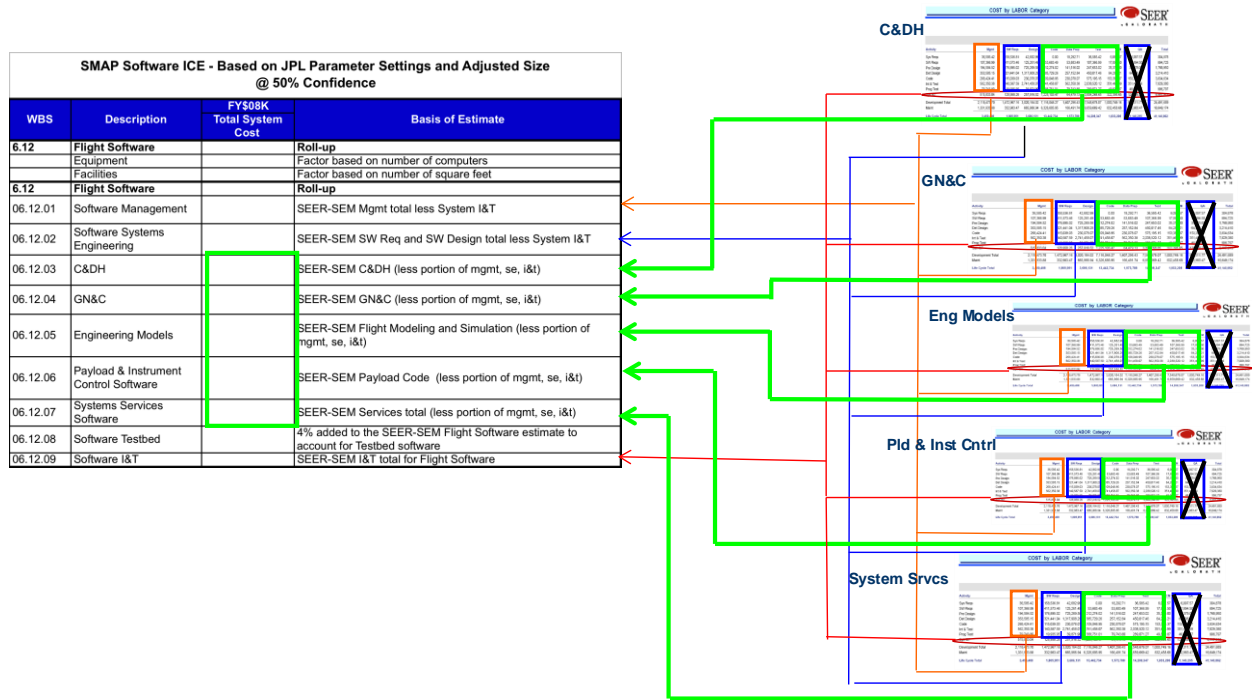
16

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

Figure 8: SEER-SEM / JPL WBS Mapping for Individual FSW Elements (Notional Sample Data)

4   The Spreadsheet

As mentioned in the introduction, the comprehensive nature of the spreadsheet yielded a deeper perception regarding the nature and processes of FSW cost estimation. For each Type X proposal, the sheet included general mission information together with detailed and significant SEER-SEM input parameter data. Since there were N_0 missions that had to be costed, this yielded a spreadsheet whose size parameters made its complete inclusion in this paper prohibitive. A smaller portion of the sheet containing all of the parameters for only five of the missions sufficiently conveys the sense of expanse and detail implicit in the sheet and is displayed in Figure 9 (a) and (b)[7]. Each mission has its own column. The rows display pertinent information for the corresponding column mission. The first 9 rows represent various 'mission facts such as the mission category, name, cost lead etc. Note that certain proprietary data have been blanked out such as the final cost (dollars and work hours) and the proposal name. Note the contractor/analogy data row refers to the flight software contractor and analogy data parameters discussed earlier in the paper. The following groupings (colored in aqua) show the knowledge base inputs, Software sizing parameters (vectors 1 and 2) and parameter settings (non-default constant and varying). All SEER-SEM parameters not shown in the rows are default across the board. Figure 10 is notional and displays a mapping from the main components of the decision graph to the corresponding row parameters that those components determine the values of.

Throughout the cost estimation process, hardcopy of the evolving spreadsheet was made (taped together) in its entirety to reflect the status of cost and cost estimation progress to higher level management. The use of a large paper sheet on a big table with pencils in hand added to the analysis and monitoring process in a way that might not have been achieved otherwise. Further, a better understanding of the nature and justification of the costs was achieved by the cost leads when columns representing only their proposals were distributed to them. Finally, the bird's eye view of the mission data and SEER-SEM inputs/outputs facilitated the cost estimation consistency analysis by the cost es-

---

[7] The full spreadsheet is too large for display

17

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

timators. By checking the parameters mission by mission (i.e. column by column) and comparing costs resulting from the use of these parameters together with mission categories and contractor /analogy data, the analysts were allowed insights in a way consistent with the 'one picture is worth 1000 words' philosophy.

Category	Inn_Hel_1			Inn_Hel_2	
Proposal Name	1	2	3	4	5
Cost Lead	A	B	C	D	D
Spacecraft Provider	SDC_1	SDC_1	SDC_2	SDC_3	FFRDC
Analogy Program(s) Used	from SDC_1	from FFRDC	from FFRDC	from FFRDC	from FFRDC
Contractor/Analogy Data	SDC_1/ SDC_1	SDC_1/ SDC_2	SDC_2/ SDC_2	SDC_3/ FFRDC	FFRDC/ FFRDC
Software Cost Estimates (SEER-SEM) (FY$10M) (excludes testbed, equip, facilities)	$XX	$XX	$XX	$XX	$XX
SEER-SEM (- ATLO, SQA, CM 50%)	$XX	$XX	$XX	$XX	$XX
Team X Estimate (for reconcilliation)	$XX	$XX	$XX	$XX	$XX
Software Duration (SEER-SEM) (mo)	27	30	23	30	26
Knowledge Bases					
SEER-SEM Window Name: (Create/Modify WSB Element)					
Platform (Operating Environment)	Unmanned Space	Unmanned Space	Unmanned Space	Unmanned Space	Unmanned Space
Application	Flight Systems	Flight Systems	Flight Systems	Flight Systems	Flight Systems
Acquisition Method	New/Reuse	New/Reuse	New/Reuse	New/Reuse	New/Reuse
Development Method	Incremental	Incremental	Incremental	Incremental	Incremental
Development Standard	DO-178B Level B	DO-178B Level B	DO-178B Level B	DO-178B Level B	DO-178B Level B
Software Size (SLOC)					
Size BoE	Used actual SLOC counts from SDC_1. Assumed 25% new, 25% reused "as is", and 50% reused modified.	Used an average actuals from FFRDC projects with the inheritance percentages fromFFRDC.	Used SDC_2-derived SLOC values for new, reused, reused modified. Added correction factor to convert code counts.	Used FFRDC TDP information.	Used FFRDC size estimates. Duplicated reasoning used for FFRDC estimate.
ESLOC	69,888	92,238	61,848	85,533	61,450
Delivered Software (SLOC) - most likely	153,812	202,000	204,990	221,664	180,000
Software Size (SLOC)					
New SLOC - most likely	38,453	60,600	25,000	46,404	30,000
% of new SLOC	25%	30%	12%	21%	17%
Reuse SLOC (as is - no mod) - most likely	38,453	35,350	97,700	117,424	70,000
% of reused (as is) SLOC	25%	17%	48%	53%	39%
% re-design	0	0	0	0	0
% re-implementation (Re-coding)	0	0	0	0	0
% re-test	50%	50%	50%	50%	50%
Reuse SLOC (modified) - most likely	76,906	106,050	82,290	57,836	80,000
% of reused (modified) SLOC	50%	53%	40%	26%	44%
% re-design	10%, 25%, 25%	10%	10%	10%, 25%, 25%	10%
% re-implementation (Re-coding)	10%, 25%, 25%	10%	10%	10%, 25%, 25%	10%
% re-test	50%	50%	50%	50%	50%

(a)

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

Parmeter Settings Notes					
Personnel Capabilities & Experience (7 parameters)	Leave at KB setting. This reflects an industry average which is appropriate since we do not know the composition of the software development team so early in the proposal process.				
Analyst Capability					NOM-
Analyst's Application Experience					NOM
Programmer Capabilities					NOM-
Programmer's Language Experience					VHI
Developkent System Experience					HIGH
Target System Experience					VHI
Practices & Methods Experience					VHI
Development Support Environment	Leave at KB settings with the exception of:				
turnaround time	VLO	VLO	VLO	VLO	VLO
response time	LOW	LOW	LOW	LOW	LOW
Product Development Requirements	Leave at KB settings with the exception of:				
requirements volatility	HIGH	HIGH	HIGH	HIGH	HIGH
spec level - Reliability	HIGH-	HIGH-	HIGH-	HIGH-	HIGH-
test level	HIGH	HIGH	HIGH	HIGH	HIGH
quality assurance level	HIGH	HIGH	HIGH	HIGH	HIGH
rehost (development to target)	HIGH-	HIGH-	HIGH-	HIGH-	HIGH-
Product Reusability Requirements	Should always be NOM (no reusability required by the contract). If the parameter is set to NOM the percentage value is meaningless.				
Development Environment Complexity	Leave at KB settings with the exception of:				
process improvement	NOM	NOM	NOM	NOM	NOM
Target Environment	Leave at KB settings with the exception of:				
memory constraint	NOM	NOM	NOM	NOM	NOM
timing constraint	NOM+,NOM+,HIGH-	NOM+,NOM+,HIGH-	NOM+,NOM+,HIGH-	NOM+,NOM+,HIGH-	NOM+,NOM+,HIGH-
real time code	NOM, NOM, NOM+	NOM, NOM, NOM+	NOM, NOM, NOM+	NOM, NOM, NOM+	NOM, NOM, NOM+
security requirements	NOM	NOM	NOM	NOM	NOM
Schedule & Staffing Constraints	Leave at KB settings with the exception of:				
start date	11/25/2012	11/25/2012	11/25/2012	11/25/2012	11/25/2012
Min Time vs Optimal Effort	Always start with **Optimal Effort**. Where possible, verify that the schedule duration is achievable. If not, evaluate schedule constraints to accommodate the estimated schedule. If the software development time is less than the **Minimal Time**, the SEER-SEM model contends that it is not possible to complete the software. Identify this as a significant risk issue!				
Confidence Levels	Both effort and schedule should be run at 50% and 70% confidence. SQI recommends the 70%				
Requirements	Leave at KB settings with the exception of:				
requirements after baseline	YES	YES	YES	YES	YES
System Integration					
number of programs being integrated	5	5	7	5	5
concurrency of I&T	Hi	Hi	Hi	Hi	Hi
hardware integration	N-, N, N+	N-, N, N+	N-, N, N+	N-, N, N+	N-, N, N+
Ecomonic Factors	Labor rate based on NASA Center contractor developed software survery conducted in FY08. Escalated to FY$10 using the NASA New Start Inflation index (5.6%).				
cost base year	2010	2010	2010	2010	2010
labor rate (FY$2010) work months	$xx	$xx	$xx	$xx	$xx

(b)

Figure 9: Portion of Final Spreadsheet

Software Cost Estimation Using a Decision Graph Process:
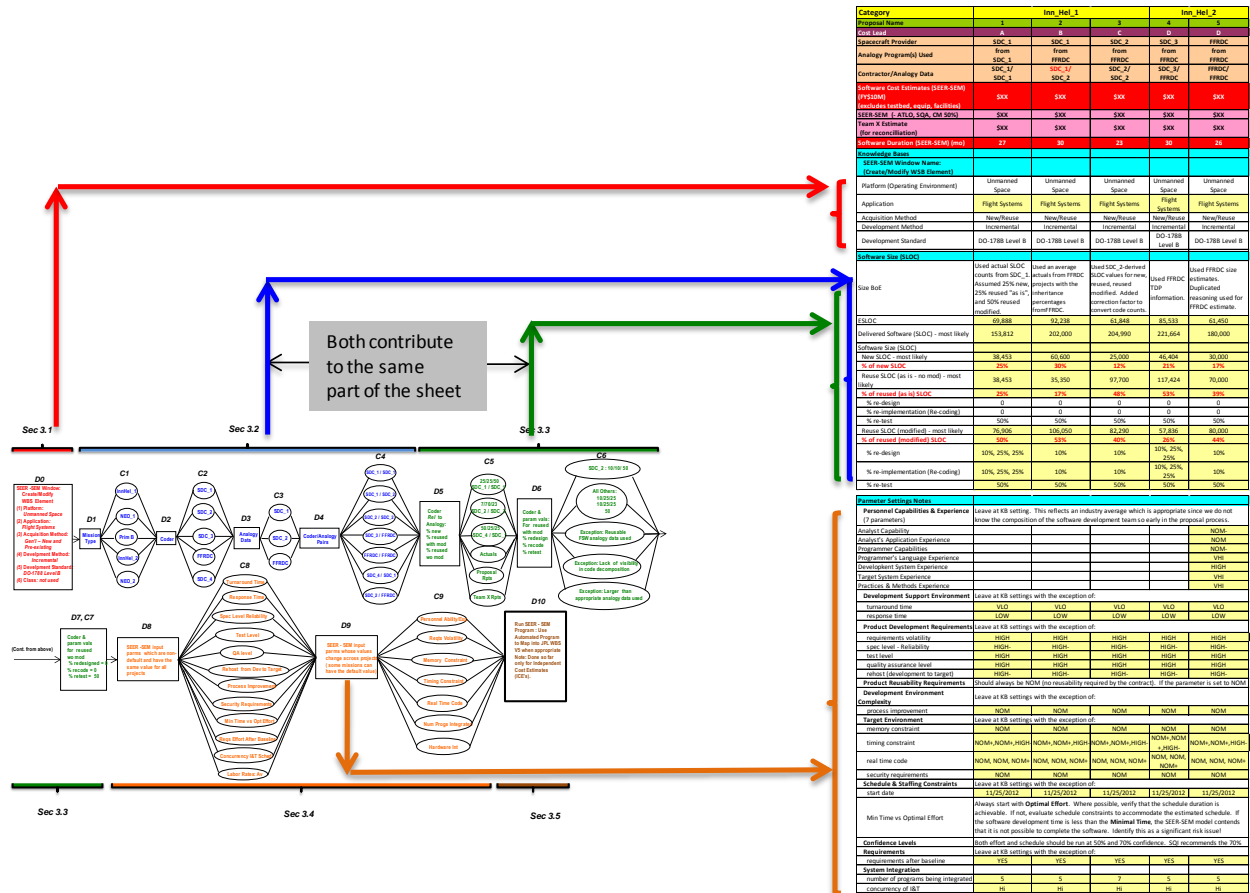A Knowledge Engineering Approach

Figure 10: A Notional Retrospective of Decision Graph and Spreadsheet

5   Summary and Future Work

The ideas presented in this report are best encapsulated by the tried and true sentiment "Necessity is the mother of invention". During the process of producing timely and accurate flight software cost analyses, it became clear that efficient and effective methodologies for consistency checking had to be established. Intuitively, the best way to accommodate this requirement was to develop a comprehensive spreadsheet which, for each mission, showed all parameters which served as the blueprint for that mission's FSW cost estimate. At first this only included certain parameters which were inputs to the SEER-SEM FSW costing model. It was subsequently determined that it was equally important to include mission information that gave rise to parameter value selection. The consideration by the expert of the interplay of data /reason for data forced upon both authors a progressively exacting analysis not only regarding the FSW costing exercise but also with respect to an examination of the thinking processes behind the execution of such an exercise. This resulted in the invention of the decision graph.

Logically, the next step would be to start a construction of a meaningful rule based system which could serve as an aid in the determination of FSW analysis. One approach would be to use the decision graph developed in this paper for the N_0 Type X proposals and build a simple decision tree (and hence a rule base) from that. Gaining practical experience in the construction of expert systems of this type would not be the only benefit of building a costing program based on these rules. Constructing this simple rule based system based on the Type X proposals could allow future analysts to run it, get the determined cost associated with the proposal of interest and have a complete explanation from the computer as to how these costs were obtained based upon the triggered rules.

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

An example of how a decision tree corresponds to the decision graph can be seen in the following example. Figure 11 shows an initial part of the decision graph. As can be seen, the first column of nodes contains all the mission types that appear on the spreadsheet while the second column lists all the FSW contractors. No correlation between the two is given. Figure 12 shows the actual pairings of Mission Type/Coder that actually appeared in the spreadsheet. This serves as direction for the design of the expert system. One way in which this can be seen is that in a constructed system, after the user enters the Mission Type based upon the possibilities of column 1 nodes, the computer could then query the user to select from the limited range of FSW contractor possibilities based upon that initial input.
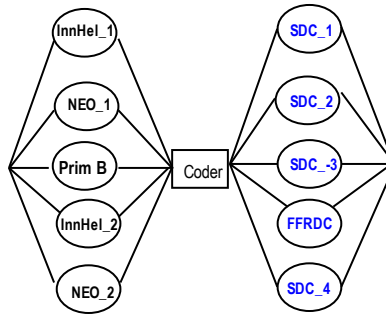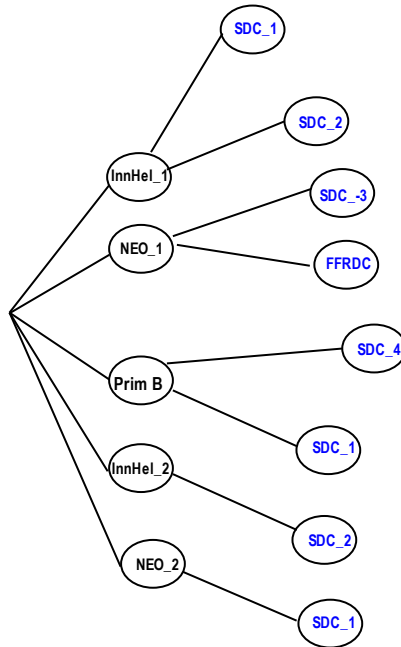

Figure 11: Portion of Decision Graph

Figure 12: Corresponding Decision Tree

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

So, for example, if the user chose that he was interested in a primitive body mission, the computer could then say:

"In the Type X Mission Proposal Data Base for Year 2010, the two possible FSW contractors for Primitive Body Missions were SDC_4 and SDC_1. SDC_4 worked on a comet mission to examine its core and SDC_1 worked on a main belt comet and an asteroid mission. Please select which of these missions (as listed below) you would like to see a derivation of FSW cost for."

Expansion of the rest of the decision graph into a tree structure would allow a complete step by step explanation and justification of the how's and why's of the FSW cost estimate for the mission of interest. Details of precisely what the format, nature and expanse of the explanations at each step are one of the subjects for future research even within the strict confines of the Type X mission FSW costing effort.

The graph developed in the last section was a first attempt to add a high level quantitative perspective that can serve as a quick reference for spotting trends in FSW cost estimation. One area of further work could be to compile more and more estimated costs and create corresponding graphs. Corresponding graphs could also be made for actuals. These graphs could serve as an aid in determining and graphing a 'difference metric' between actual and estimated FSW costs. Analyses could be done to improve the estimation based on the comparisons. Further estimate data and graphs could be compiled to see if the difference metric is improving. A cycle of estimation-cost metric determination-analysis-improvement could occur through time as the cost estimation data base increases.

Finally, as previously discussed, the mapping scheme which takes the SEER-SEM output into the JPL FSW WBS is semi-automated. Complete automation of the mapping for fast delivery of precision FSW cost estimates would be a natural follow up to the efforts thus far. Integrating this system into an expert system as discussed above would, with proper interaction and feedback from appropriate personnel, yield a powerful interactive costing tool within the JPL community and beyond.

Software Cost Estimation Using a Decision Graph Process:
A Knowledge Engineering Approach

Biographies

Sherry Stukes

Ms. Stukes is a Software Systems Engineer/Cognizant Engineer at The Jet Propulsion Laboratory in Pasadena, California. She specializes in software estimating and software data collection in support of JPL Independent Cost Estimates, proposals, and CADRe development. Ms. Stukes manages a research project that will provide a software estimating tool for NASA Headquarters, which is not dependant on software lines of code.

Some of Ms. Stukes prior accomplishments include: development and maintenance of two large databases for the Air Force Space and Missile Systems Center (AF SMC): the Software Database (SWDB) and the Operations and Support Database (OSDB); instructor for the Army Logistics Management College (ASDC_2C) Software Estimating Models course; and advisor to Air Force Institute of Technology (AFIT) students conducting thesis projects in the area of software model calibration. Ms. Stukes was the 1997 International Society of Parametric Analysts (ISPA) Parametrician of the year.

Ms. Stukes holds a BS degree in Business Administration from California State University, Long Beach and an MBA from California Lutheran University.

Dr. John Spagnuolo, Jr.

Dr. Spagnuolo is presently serving as an Engineering Cost Analyst in the Engineering Cost Estimation Group at The Jet Propulsion Laboratory in Pasadena, California. He specializes in the preparation of Cost Estimation Data Requirements (CADRes), Cost Estimating relationships (CERs) and software cost estimation and analysis. He holds degrees in mathematics from Clarkson College of Technology, University of California at Los Angeles, and the Doctor's degree from Rensselaer Polytechnic Institute.

In the past, Dr. Spagnuolo has received several awards and certificates of recognition relating to his work in neural networks and solar physics. He received the Space Act Award for his paper on the Computation and Visualization of Archimedean spirals in 3 dimensions. Along with numerous conference and journal articles, he has published several abstracts in the NASA Tech Briefs Journal. He also developed an architecture for a hierarchical planning system for computerized military war gaming and an expert system for use in automated decision making in a computer war game for which he won best session paper at the Military Operations Research Society (MORS) Conference.

Presently, Dr. Spagnuolo is studying Newton's Principia from a mathematical perspective and doing research in mapping cognition onto an artificial neural network.