Mental Models of Software Forecasting

J. Hihn
A. Griesel
K. Bruno
T. Fouser
R. Tausworthe

Jet Propulsion Laboratory/California Institute of Technology
4800 Oak Grove Avenue
Pasadena, Ca. 91109

Abstract

The majority of software engineers resist the usc of the currently available cost models. One problem is that the mathematical and statistical models that are currently available do not correspond with the mental models of the software engineers, In an earlier JPL funded study (1 lihn and Habib-agahi, 1991) it was found that software engineers prefer to use analogical or analogy-like techniques to derive size and cost estimates, whereas current CER's hide any analogy in the regression equations. In addition, the currently available models depend upon information which is not available during early planning when the most important forecasts must be made.

The work reported here is the first step in defining the mental models software engineers use to forecast effort and size. Protocol analysis (a technique for converting self reported narratives (verbal protocols) into data) and probability transition matrices are used to capture and model the processes used by software forecasters. These models then can be used to determine the structure of software costing methods, tools and databases which software engineers actually will be willing to u se.

This paper presents a summary of the results of one port ion of a JPL internally funded research task to study the costing process and parameters used by internally recognized software cost estimating experts. The results of the analysis include: the demonstration of a rigorous data collection and analysis technique; the identification of a core set of well defined costing activities, and the identification of several hypothesized basic software forecasting mental models.

1.0 introduction

Software cost, size and schedule forecasting is a very complex task requiring the prediction of an inherently stochastic process while incorporating many variables with varying degrees of confidence in their fidelity. It is well documented in the cognitive psychology (Slovic, 1982) literature that the way that the human mind deals with complexity is to develop a mental model which is a simplification of objective reality. Statistical and mathematical models are very similar in that they also are simplified models of reality. However, the mathematical models that are currently available neither correspond nor easily integrate with the mental models of the software engineers. 1 n a recent survey it was found that 11% or fewer of software engineers used formal CERs either as their primary or secondary method whereas 87% reported using some form of analogical comparison (I lihn and Habib-agahi, 1991). It is important for both to be readily available, as it has been shown that more accurate forecasts are produced from a 50-50

The research described in this publication was carried out at the Jet Propulsion Laboratory, California Institute of l'ethnology, under a contract with the National Aeronautics and Space Administration.

combination of expert judgement (mental model) and statistical model.(Bunn and Wright, 1991) This paper reports the. preliminary work in identifying and examining engineers' forecasting processes, or mental models. It presents the methodology which has been developed to collect and analyze data obtained from verbal self-reports. Results from applying the analysis methods to data from a previous study show the power of the method and also gives a few preliminary results.

Existing forecasting tools or models which support analogy-based reasoning either are highly structured, simplifying reality by limiting the application domain, or are so flexible that the user must provide a structure thus providing his own simplification. The Software Engineering Laboratory (SEL) at Goddard Space Flight Center (GSFC) has developed the Software Management Environment (SME). This resource model provides analogical cost estimates based upon an extensive history of Flight Dynamics Software Systems, where the estimator selects components of one of the three basic designs that are used for Flight Dynamics Systems. This approach has proven to be quit successful, Martin Mariet a developed an analogy-like software tool for usc by the Air Force, and the Institute for Systems Analysis has developed CostExpert for C2 and C3 systems. The Small Business Innovative Research program (SBIR) funded a software size estimation tool which demonstrated that an automated tool could estimate size at least as well as human estimators. This was tested on software in the COSMIC database. Although the SBIR sizing tool dicl predict size well, it fell into the classic trap of providing a classification scheme that was too foreign for potential users. These and other systems have demonstrated that analogy-based forecasting tools are feasible for solving portions of the estimation problem for narrow domains. During the past few years ESPRIT under the MERMAID Project has funded research on cost forecasting and metrics tools. Commercially these are sold under the name ESTEEM by VOI .MAC, a Netherlands based company. 'l'here are several modules which support analogy type forecasting: Analogy Based Estimation (ABE) (Corbet and Kirakowski, 1992), Experienced Based Est imation (EBE) (Cowderoy, 1992) and a similarity function.

Such narrow-domain applications of analogy-based forecasting take advantage of the simplification of reality introduced by the domain. In order to develop tools and databases which are not domain specific, we need to examine the forecasting method, or process, used by the engineers. This will allow us to identify the mental processing elements which are used across many domains. This is what we call the forecaster's "mental model". An appropriate research method for identifying the elements of software forecasters mental models are those used in cognitive psychology. While cognitive psychology has been used to study programming, human-computer interaction and design behavior (Ackerman and Tauber, 1990) it has received little application in the analysis of forecasting behavior. A variety of useful forecasting practices arc in use and are important to capture if a formal process is to be ultimately proposed and be acceptable to software engineers. The two preliminary attempts to develop such mental models of the forecasting process documented in the literature, by Vicinanza et. al. (1991) and Howard (1992), arc discussed in the next section.

2.0. Existing Mental Models of Software Forecasting

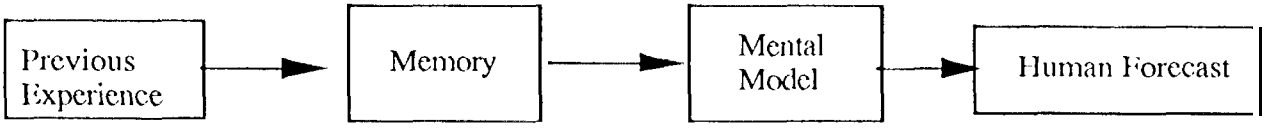

Figure 1: Simplified Model of Cognitive Process for Software Forecasting

Figure 1 provides a context for a discussion of forecasting mental models. Based upon previous experience a forecaster stores in format ion in memory. At the same time a forecaster evolves a mental model (Simon, 1959) of how to combine the information in order to use it effectively. This

mental model is a simplification of reality allowing the forecaster to reduce complexity and not be overwhelmed by cognitive overload.

This mental model of how information is combined can be represented by a sequence of activities or events which are repeatable. Some of the sequentiality is inherent in the model, e.g., when a decision must be made. However, some of the sequentiality is an artifact of most data collection techniques. "I'he activities are abstractions that correspond to the cognitive components. Therefore, to document this mental model requires determining the set of activities and their corresponding activity sequence.

The two previous attempts to identify the mental models of software forecasters both present interesting insights. Ilowever, they have some methodological and analytical problems.

Vicinanza et al completed an exploratory study of the methods used by experts. They had five respondents who ranked a series of cost drivers and then estimated the development effort for 10 projects. They categorized the forecasters' methods by four categories: algorithmic initial condition, algorithmic effort estimate, analogical initial condition, and analogical effort estimate. For a method to be algorithmic the forecaster had to mention and use productivity figures. For a method to be analogical the forecaster had to mention a reference project. Four of the estimators used an algorithmic approach and only one used analogy. Vicinanza et al propose a logic flow (mental model) for algorithmic ant] analogical forecasting (see Figure 2 for the analogy model), Given their simple categorization scheme it is unclear how they derived their mental model. Other problems are that the experimental design required that the engineers use COCOMO and function point descriptors, neither of which may have been natural to them; and the terms used in their mental model are neither goals nor the vocabulary that are commonly used by software engineers.

Howard reports the results of two surveys cm software cost estimation practices for standard information systems such as a bank transaction system. Approximate y SO observations were collected using a survey form Twelve observations were collected using semi-structured face to face interviews based upon a case description which the subjects were given before the interview. The objective was to develop a recta-model of the mental processing steps which estimators follow to support research on how cost estimates are developed in group settings. A very high level model with about 20 possible steps is proposed based upon cognitive processing theories. For an example of the mental model for the "bottom up" process see Figure 3. interestingly, aggregation is never mentioned even though "functional breakdown into components" is explicitly shown.

The model proposed is intuitively appealing. However, the respondents provided quite generic responses describing how they normally do cost estimates. Howard reports this is because the case example was found to be too poorly defined. Verbal reports of this type are well known to lead to biased and very likely inconsistent results (see Ericson and Simon , 1984).

Howard provides some interesting insights on analogy and "fuzzy" actions. She finds that analogy can be applied at any of the steps as a check on the cost forecasting process, Forecasters make intuitive leaps (fuzzy actions) especially with respect to how they transition from "read user demands" to developing a simplified model of the system and the incorporation of residual factors into their forecast. An example of a residual is adjusting for differences in development process.

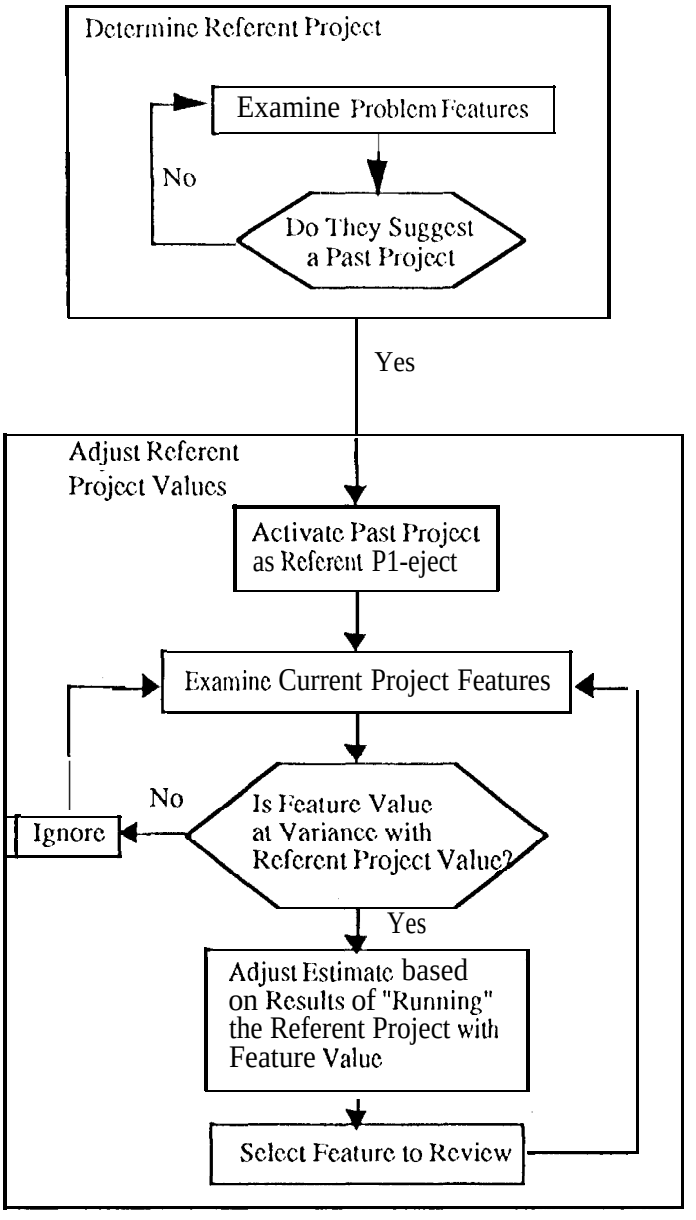

Figure 2 : Abstraction of Analogical-Estimation Strategy from Vicinanza et al (1991)

in both of the papers described above the bases by which the proposed software forecasting mental models have been derived is not explained, Howard is following some basic cognitive psychology techniques, but it is not clear that they were derived by a repeatable analysis. A significant problem from the perspective of identifying a more detailed picture of the underlying mental model is that most of what distinguishes an expert from a novice is in how they generate and "factor residuals" or, in other words, incorporate their cost drivers and adjustment factors. "l'his element of the model needs to be developed in greater detail in order to determine what processes, data and tools should be developed.

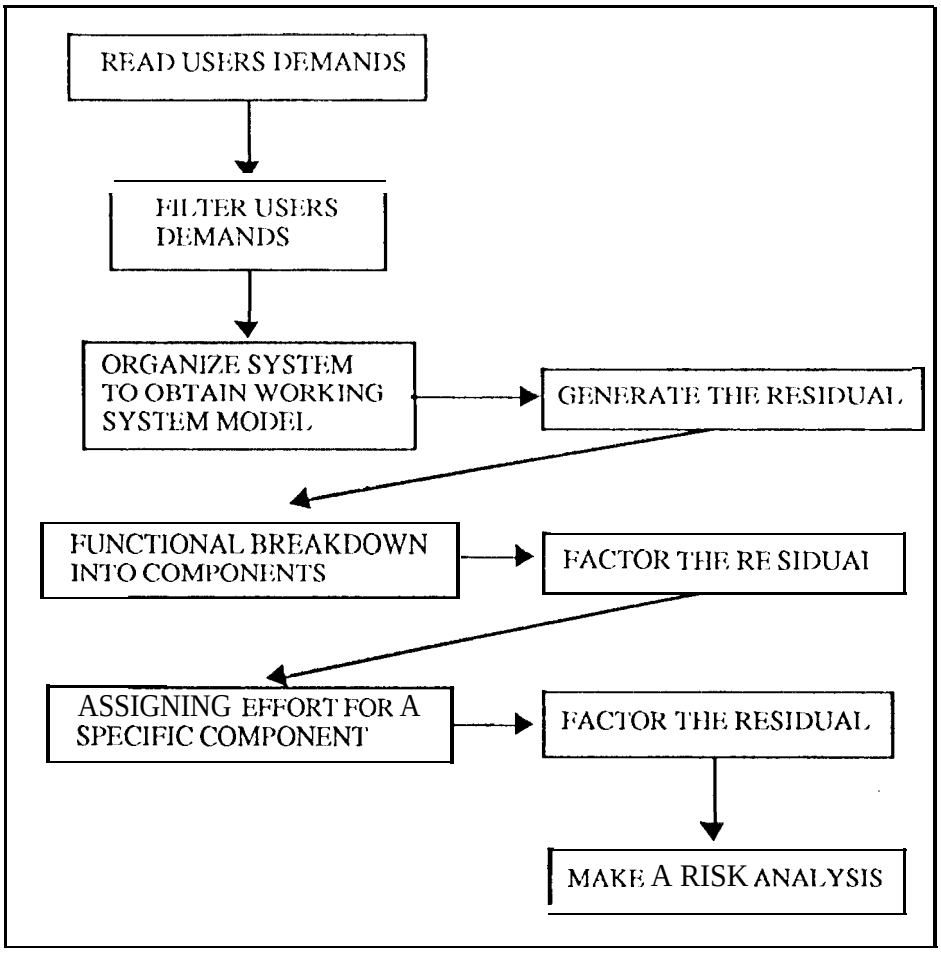

Figure 3 : A Bottom Up Approach to Estimation from Howard (1992)

From the analysis of the existing work we are now better able to identify a more rigorous methodology which combines the use of Protocol Analysis with Markov Processes. What is needed is:

(1) a very systematic way by which a finite set of well defined costing activities are identified;

(2) a well defined mapping of the actual verbal expressions to the costing activities (this provides the vocabulary and documents in a repeatable way how the mental model was constructed);

(3) mathematical and/or statistically based techniques by which the mental models are identified.

3.0 Methodology

Two basic methods arc used in the collection, derivation and analysis of the data, Protocol Anal ysis (Ericson and Simon, 1984) and the techniques associated with Markov processes (Phillips et al, 1976).

Protocol Analysis is a data collection and analysis technique used to convert unstructured and semi-structured verbal responses (verbal protocols) into data describing cognitive processes. The verbal protocol method is ideally suited to the task of learning what mental models experts apply to cost forecasting. Using Protocol Analysis, forecasters are allowed to actually estimate in their own domain using their own vocabulary yet still provide repeatable, domain independent information. 'l'his data collection technique will be used in the interview portion of the study.

'l'he Protocol Analysis technique was used to perform the secondary analysis of an existing data set to demonstrate its feasibility for identifying forecaster's mental models. Therefore, only the data analysis part of the Protocol Analysis technique is reported on here. The most important step in the data analysis is the construction of a scoring taxonomy which captures all the relevant characteristics of the different mental models. This taxonomy can be refined as the data is collected and anal yzed but all verbal reports must be scored in the same manner. The current analysis provides an initial scoring taxonomy for analyzing the data to be collected. The reliability of the scores is greatly improved if at least two different persons score the reports independentl y. Differences are resolved, yielding a refinement in the categories; this may require that all responses be re-scored using the new set of categories.

Once the verbal protocols have been converted into data, a method for constructing the mental models is needed. Because a significant portion of each forecaster's sequence of activities is clue to individual differences, which are presumably distributed over the population, the summary of the mental models over a group of engineers can be viewed as a stochastic process. While there is an underlying structure that is consistent over groups or at least subgroups of forecasters, the manner in which forecasters walk through this structure varies. "l'he process can be represented by the probabilities of transitioning from one activity to another. 'l'he derivation of these probabilities is greatly simplified if it is assumed that the transition from state i to state j is independent of the sequence of steps followed to get to state i. in other words future states are independent of the past and all that one need know is the current state to derive the transition probabilities to the next time period,

The assumption of time independence is being made for this analysis of the existing data and is not expected to hold when more detailed data is collected. This assumption is a strong one because it is expected that when a forecaster uses analogy or a CER there would be a specific sequence or sub-sequence of steps followed. For example, with a CER a typical sequence might be estimate size, estimate cost driver values, estimate cost by applying CER. Analysis of transition probabilities conditional on the time steps indicates that even for the existing data dependence on short sub-sequences of history exists, However, for short processes with initial and absorbing (terminating) states, stationary one-step transition matrices provide a reasonable initial description of the respective processes (Papoulis, 1991).

4.0 Sample Definition and Background information

The following brief overview of JPL is meant to provide the proper context to understand the survey data. JPL is an FFRDC run by the California Institute of "1'ethnology under a government contract with NASA. As a national laboratory, it performs research and development activities, primarily the development of robotic spacecraft for interplanetary studies. In addition, a portion of JPI .'s budget is supplied by non-NASA organizations such as the Department of Defense,

The methods used to estimate costs differ depending upon the required reliability. A "Class A" estimate is the most rigorous, and a formal review is required. For Class A estimates, model-based and other independent estimates are supposed to be obtained. However, most estimates are produced by a bottom-up exercise which may involve as many as 25 persons for the software activities portion of each task. Since bottom-up or grass roots estimates are the main process used for major JPL projects, a survey was conducted of the technical staff that actually performs the lowest level estimates during the summer and fall of 1989.

The original purpose of the survey was to study the ability of software engineers to estimate effort and size given an architectural design document. In addition, the survey included a brief description of the typical approach each used to estimate. Since data collected in this manner is not appropriate e for Protocol Aria] ysis, any conclusions drawn from this secondary analysis of the data arc tentative. The main purpose in attempting to rc-analyze the old survey data is to identify the linguistic units, to develop a prototype scoring taxonomy, and to investigate the appropriateness of different types of statistical analysis,

Over 185 software engineers were contacted for participation in the original survey. Of the 185 contacted, over 100 were identified who estimate effort, size and/or cost for software tasks. Of these, 83 were willing to complete the questionnaire on current software cost estimation practices. Of these, 28 responses provided sufficient information for use with the current analysis. For a detailed discussion of how the original data was collected see} lihn and Habib-agahi (1991).

Tables 1 and 2 provide a summary of the basic experience of the respondents included in this study. The data presented shows that the average person has substantial experience and makes estimates one or two times per year. On average the participants have approximately 16 years of experience working on software tasks and ten years making cost estimates, Eight of the 15 total years have been spent with managerial responsibilities (cognizant engineer or higher). This reveals a substantial amount of experience that is being used for cost forecasting.

Table 1: Summary of Respondent **Experience**

Type of Experience	Mean (years)	**Standard Deviation** (years)
Software Experience	16	8
Managerial	8	5
Estimation	10	7

Table 2: Summary of Frequency of Estimation

Estimate	**Mean *** (months)	Standard Deviation (months)
Most Recent	6	9
Second Most Recent	14	13
~'bird Most Recent	24	16

* **Number** of months from date of interview.

5.0 Analysis

5.1 Definitions of Activities

Vicinanzi and Howard's work indicate that there are only a few basic forecasting activities which are used by forecasters, sometimes repeatedly and in varying sequences. Such activities constitute an abstract vocabulary that can be used to describe the forecasting process. The activities and their definitions were derived from the literature, JPL experiences documented in Lessons 1 earned, and the personal costing experiences of the authors. All of this was modified by the data available in the verbal protocols to maximize the scoring of the linguistic units into one and only one scoring category, "l'able 3 contains the list of software forecasting activities used for this preliminary analysis and their definitions. Note that any sub-activities which have been identified so far are specified in the definitions. Appendix A contains the mapping of the vocabulary in the verbal protocols to these activities,

'] 'he level of granularity of the activities determines the information obtainable from analysis of the forecasters' activity sequences. An activity set defined at too coarse a granularity can not distinguish between sequences and all protocols will appear identical. An activity set defined with to much detail, at too fine a granularity, makes every protocol appear unique. Hence identifying the right granularity, or level of abstraction, is crucial.

The existing data is at a fairly coarse granularity. Sometimes verbal protocols at this level map into more then one activity. The term analyze requirements is a very complicated activity which occurs frequently in the protocols, However, from a cognitive perspective the main purpose of anal yzing the requirements is to develop a representation of the software system. A key component of this representation is to identify and encode the characteristics of the software system. This is the attribute identification activity. Attributes are used in a variety of ways; when using analogy, attributes are applied as discriminators to identify similar tasks, and when using CER's they are applied as multipliers or cost drivers. Aggregation was the only activity which at times was inferred from the context of the protocol, A number of cases occurred where decomposition was mentioned but an aggregation step was not explicitly stated. It was assumed that since the objective of the cost estimate was to forecast the total effort that estimates of pieces had to be eventually aggregated. Note also that verification was not included as one of the activities. Verification is seen as executing one or several of the software costing activities included in the current activity taxonomy. Hence, it is associated with a different level of abstraction which groups a number of activities or defines a specific sub-sequence. This suggests that at the next higher level of abstraction a forecasting life-cycle could be identified.

'l'able 3: **Hypothetical Software Cost Forecasting** Activities

Activity	Definition
Requirements Identification (RI)	The obtaining or retrieval of information. Kcy vocabulary words arc read requirements, talk to experts, review requirements, and obtain requirerments.
Attribute identification (Al)	Attributes arc key aspects of a task which arc used in forming the system mental model and arc also used as analogy discriminators and cost drivers. This is onc of the main products of the analysis of the requirements. Al is generally cic.scribed by the basic activity that was undertake.n with the result that precise attributes are rarel y specified at this point. These consist of product and process attribute.s. Kcy vocabulary words are identify, understand, analyze, and include.
Decomposition (DE)	The breaking down of a software entity (s ystem, subsystem, etc.) into smaller and simpler pieces. The types of decomposition which have so far been identified arc: functional, WBS, ncw vs inherited requiremen Is. Key vocabulary words are breakdown (functions), identi fy sub-tasks, develop (W Bs)
Estimation (ES)	The prediction of future cost and other key project management dimensions. The main items forecasted are: schedule size effort dollars ES can be further divided by type of technique used: analogical expert judgement explicit analogy algorithmic rules of thumb CERs Key vocabulary words arc usc (analogy, rule of thumb), estimate (SLOC, effort), and cost.
Attribute Application (AA)	The explicit usc of the system attributes to discriminate between systems for purposes of analogical comparison or as cost drivers when using an algorithmic approach. Identification primarily depends upon specific mention of attribute. While there is less homogeneity in the vocabulary some common phrases are adjust, use (fog factor), add (change, fog factor, etc.), multiply.
Aggregation (AG)	The combination of forecasted values associated with the system pieces produced by decomposition. Key vocabulary words arc add-up and run SRM (J PI, resource management tool)
Adjustments (AD)	Multipliers used independent of the system being costed. Usually applied at a higher level then attributes. Consist of adjustments for purposes of risk scaling bias(error) Kcy vocabulary word is add percent.

S.2 Data Summary

The data consists of 28 verbal protocols. The number of steps ranged from 4 to 11. 'l'able 4 shows the number of steps for each verbal protocol; the most likely number of steps is 7, the average number of steps is 8. 'l'he distribution is bimodal. This arises because some verbal protocols contain more then one sequence of estimation activities. The additional estimation activities support verification or an estimate such as size which was used in a subsequent effort activity. Those verbal protocols with only 4 or 5 steps are almost certainly due to a lack of verbalization of the mental model; therefore these observations were di scarded from the data set when deriving the mental models re.ported in Section 5.6.

'l'able 4: Frequency Count of the Number of Steps in Each Verbal Protocol

Number of Steps in Estimation Sequence	Number of Observations	Percentage
4	3	11%
5	2	7 %
6	3	11 %
7	7	25 %
8	3	11 %
9	6	21 %
10	2	7 %
11	2	790

'liable 5 contains a summary of all of the responses by activity and sub-activity. The table was constructed by scoring each recognizable linguistic unit in the-verbal protocols by their pre-defined mapping to an activity. These were then summed over all observations. 'l'here are more or fewer ticks per activity than the number of verbal protocols (28) because the engineers did not always execute every activity while others were repeated multiple times, The distribution of activities is relatively uniform around 12% except that the number of times an estimation activity was ment ioned is high (31%) and references to adjustments were relatively low (7%). The fact that estimation is high is partly due to a lack of articulation of the other activities and that the main focus of the survey was software cost forecasting. Therefore these results do not imply that engineers spend 2-3 times the effort performing estimation activities compared to other activities, However, looking at the sub-activities within estimation, engineers at JPL are more likely to use expert judgement ant] rules of thumb or informal methods than formal forecasting methods. The small number of reported adjustment activities, especially related to risk adjustments, probably reflects an engineer's tendency to hide risk in a variety of amorphous adjustments, for example, assuming a higher average wage rate for labor.

Table 5: Summary of Responses by Software Forecasting Activity

Activity	Number of Times Reported	Percentage	Percentage by Major Activities
Requirements identification	27	13	13
Attribute identification			11
Not Specified	20	10	
Product	2	1	
Process	0		
Decomposition			17
Not Specified	6	3	
Functional	12	6	
WBS	6	3	
New/Old	8	4	
Requirements	3	1	
Estimation			31
Not Specified	8	4	
Expert Judgement	24	11	
Explicit Analogy	11	5	
Rules of Thumb	17	8	
CER	6	3	
Attribute Application			9
Not Specified	3	1	
Product	15	7	
Process	3	1	
Aggregation	25	12	12
Adjustments			7
Not Specified	2	1	
Risk	6	3	
Bias	1	1	
Scaling	4	2	
Total	209	100	100

5.3 Analysis of Differences in the Distribution of Activities

Given the small sample size one way to split the data is between the Information Systems Division and all other technical divisions. The information Systems Division main objective is the development of ground based software for NASA and non-NASA sponsors. The other divisions develop software in support of one main objective and in general only work for NASA. 'l'here are 18 information Systems Division observations and 10 observations from other divisions. The summaries are displayed in Table 6 for all of the identified activities and in Table 7 for main activities only. Based upon a chi-square test the over all distributions arc not statistically different even at the 10% level, Looking at only differences of 570 or greater it appears that the Information Systems Division reports fewer estimation activities, specifically the usc of expert judgement and rules of thumb, and reports virtually no explicit risk adjustments.

"1'able 6: Summary of the Percentage Number of Responses by Forecasting Activity for the information Systems Division Compared to the Rest of the Divisions

Activity	information Systems Division (%)	other Divisions (%)
Requirements Identification	14	11
Attribute Identification		
Not Specified	10	8
Product	1	0
Process	o	0
Decomposition		
Not Specified	4	0
Functional	5	8
WBS	3	3
New/Old	5	1
Requirements	o	3
Estimation		
Not Specified	6	o
Expert Judgement	10	16
Explicit Analogy	6	3
Rules of Thumb	6	14
CER	3	3
Attribute Application		
Not Specified	2	0
Product	8	5
Process	1	2
Aggregation	11	14
Adjustments		
Not Specified	o	3
Risk	1	6
Bias	1	0
Scaling	3	o
Total	100	100

Table 7: Percentage Number of Responses by Major Activities for the information Systems Division Compared to the Rest of Divisions

Activity	Information Systems Division (%)	Other Divisions (%)
Requirements Identification	14	11
Attribute Identification	11	8
Decomposition	17	15
Estimation	31	36
Attribute Application	11	7
Aggregation	11	14
Adjustments	5	9
Total	100	100

The differences in the estimation activity are significant as shown by 'l'able 8. In Table 8 the estimation activities were grouped by whether they were to support: an intermediate estimate such as SLOC; effort estimation, whether an intermediate estimate was performed or not; or a verification estimate. The percentages shown are normalized for the estimation activities only. A chi-square test produced a score of 7.89 with 2 degrees of freedom which is significant at the 2.5% level. This is primarily due to a greater reliance on intermediate estimates such as size anti schedule by the other divisions. Also note that there is relatively little time spent on verification. in fact, only 7 out of 28 verbal reports mentioned verification. "

Table **8: Percentage Number of Reported** Activities by Estimation **Goal** for the information **Systems Division** Compared **to** the. **Rest** of the **Divisions**

Estimate Type	information **Systems Division (%)**	Other Divisions (%)
Intermediate	14	44
Effort	67	43
Verification	19	13
Total	100	100

S.4 Summary of Activities over Time

When the verbal protocols were scored the activities described were recorded in the order they were reported. Table 9 presents a summary of the main activities reported in time sequence. This should be viewed as a snapshot of where each respondent was for each time period. There is no information as to where they have been nor what activity they will perform next, However, we can say something about the costing behavior on average. First, if the activities ate listed in approximately the order they are performed then the matrix should be partially diagonal, which it is. The lower left hand corner and upper right hand corner consist of zeros. This means that there are indeed some activities engineers perform primarily at the beginning of the process and some primarily at the end, while in the middle of the process activities tend to be more jumbled. This is consistent with the forecaster individual differences that are expected of the forecasting process. On the other hand, there are some well defined modes for many of the time periods, which gives a sense of what activities are being performed over time, at least "on average".

A few general observations can be made based on the summary data, Steps 1 and 2 are dominated by requirements and attribute identification. Starting with step 3 the main activities bifurcate; while there is a major mode at decomposition there is a secondary mode at estimation. "l'his pattern is maintained in step 4 except estimation is now the main mode. Step 5 becomes uni-modal at estimation and then the process becomes bi-mods] again splitting between estimation and aggregation. After step 8 most of the protocols have terminated.

Table 9: Summary of Major Forecasting Activities over Time

Activity	Time Sequence 1	2	3	4	S	6	7	8	9	10	11
Requirements Identification	2.1	1	2	00	2	1	0	0	()	o	
Attribute identification	2	18	1	0	0	0	1	0	0	0	0
Decomposition	5	4	1	4	7	3	2	0	0	0	0
Estimation	0	5	8	1	0	1	5	8	9	7	2
Attribute Application	0	0	2	5	5	3	3	2	0	1	0
Aggregation	0	0	1	4	2	6	5	3	5	1	0
Adjustments	o	0	0	202	1	1	3	1	1		
Total	28	28	28	28	25	23	20	13	10	4	2

This static view of the data suggests there may exist a few well defined forecasting processes passing through the gates defined by the major and secondary modes. Figure 4 presents a dynamic view of the data, In this view the modes identified above arc virtually undiscernible. However, examining repeating activity behavior indicated that distinguishing attributes of similar sequences were those that had multiple decomposition steps and those with multiple separate estimation steps. The stationary probability transition matrices were calculated to examine the dynamic behavior of the processes for each of these groups.

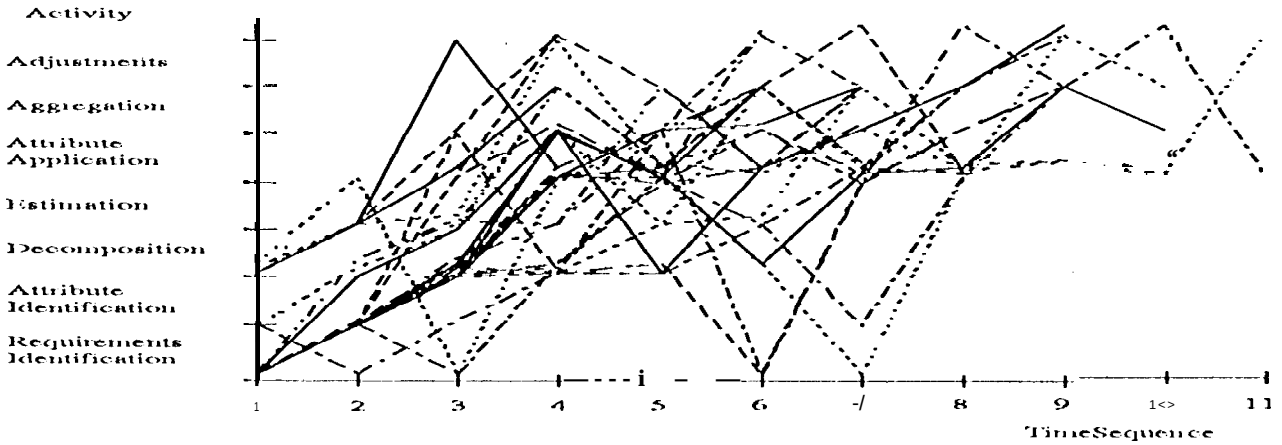

Figure 4 : Graphical Summary of Time Sequence of Activities

S.5 Transition Matrix Calculation

The probability transition matrix is a matrix of the form

$$\begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{18} \\ & & \cdots & & \\ p_{81} & & \cdots & & p_{88} \end{bmatrix}$$

where

$$\sum_j p_{ij} = 1$$

and p_{ij} is the probability of moving from activity i to activity j.

To compute the probability transition matrix for the stationary case requires simply counting the number of times that an engineer reports doing activity j after activity i. For example, assuming the order follows the list in Table 3 then p_{12} is the probability of performing requirements identification followed by attribute identification. Each time this was observed the count for p_{12} was incremented. once this was completed all the rows were normalized to sum to 1. These probabilities can then be used to construct an activity flow diagram where the activities arc the nodes and the branches are the transitions which can be assigned the probability computed as described above, This is what we call the graphical representation of the forecasting mental model. The basic structure of the graph captures that which is consistent across forecasters while the probabilities and multiple branches capture the random element of the process.

5.6 Analysis of Mental Models

Mental models can now be represented by activity flow diagrams giving transition probabilities for the flows. Models are illustrated for engineers who used single decomposition and single estimation steps (Figure 5), multiple decomposition and multiple estimation steps (Figure 6), those reporting analogy as an estimation activity (Figure 7) and those which do not report using analogy (Figure 8). Figures 5, 7 and 8 define orthogonal sub-populations.

Figure 5 shows the activity flow diagram for verbal protocols with single decomposition and single estimation steps. The sample size is 4. "l'his is the most simple of the mental models and will be used as a base for describing the rest. For this group all respondents perform requirements identification, attribute identification, decomposition and estimation as the first four steps. For step 5 there is a 43% probability of repeating the estimation activity, a 29% chance they will go to attribute application and a 14% chance they will either "jump" to the adjustment or aggregation activity. If they jump activities then the next step terminates the protocol, Upon moving to attribute application it is equally likely that the next step will be to terminate, repeat attribute application or do an effort adjustment. The mental model displayed in Figure 5 corresponds to a simple straight forward task for which the fundamentals of costing 101 apply with little modification,

Figure 6 reveals a much more complicated process consisting of multiple decomposition and estimation steps. The sample size is 5. The main differences arc the appearance of 6 feedback loops and termination from the estimation and attribute application activities. As labeled, the new

terminating branches arise due to verification of the estimate. The two feedback loops to estimation from the aggregation and adjustment activities arise due to verification estimates and also intermediate estimates. The new branches are consistent with several expected scenarios. One example is: estimate size, aggregate, estimate effort, apply attributes, then develop new estimate. to verify and report results. Two other new branches appear to be associated with the use of explicit analogy: the "jump" from identify attributes to estimation and the repeat estimation branch. The jump makes sense considering that when using analogy it is expected that attributes would be applied as discriminators to identify similar tasks in preparation for the estimation activity. Most likely there is an apply attributes activity that was not verbalized. This is partially supported by the data which shows that attribute, identification occurs 11% of the time while attribute application was reported only 9% of the time. The estimation repeat loop most likely reflects the fact that explicit analogy was most frequently used to compute the verification estimate not the original estimate.

To further explore the impact of analogy on the costing mental model the data was divided, excluding the single decomposition, single estimation group, into two sub-populations, one using analogy , Figure 7, and the other which did not usc analogy, Figure 8. The sample size for the analogy group is 9 and 6 for the non-analogy group. The differences in the figures are displayed in bold in Figure 7. The main differences due specifically to analogy are: the jump from attribute identification to estimation, the jump from decomposition to attribute application with an 80% probability of looping back to estimation, and the greater probability of sequential repetition of the estimation step. The other flows designated as different are most likely due to other causes such as the usc of multiple decomposition or verification. The implications of these results are that the use of explicit analogy has an effect on the whole mental model and therefore analogy needs to be understood in the context of the whole process. Also we can now propose some hypotheses which can be tested on the next data set.

6.0 Summary ant] Conclusions

The data analysis of the last few sections lets us begin to identify where in their cost and size forecasting process software engineers can best usc supporting methods, tools, anti data, and what kind of methods, tools and data they need. For example, the static analysis of this data would suggest supporting expert judgement. I rooking at the transitions indicates that there are often multiple sequential estimation steps. Supporting methods and tools should provide smooth transitions amongst these.

Previously published analysis of this data showed that for experienced forecasters, those who forecast frequently (at least every 6 months) on the average forecast effort 12% high, whereas those who forecast less frequently (at greater than 6 month intervals) cm the average forecast effort 44%3 low. This suggests examining the mental models for those activities and transitions most dependent on memory.

Finally, the idiosyncratic nature of the individual protocols indicates that supporting methods and tools need to capture and record the steps followed and information used by the forecaster. "l'his will provide a record of the assumptions and context within which the estimate was made, and improve the quality of updated estimates.

Thus we have demonstrated a viable process for capturing and analyzing the mental models software engineers use for cost and size forecasting which can provide requirements for supporting method, tool, anti database specification.

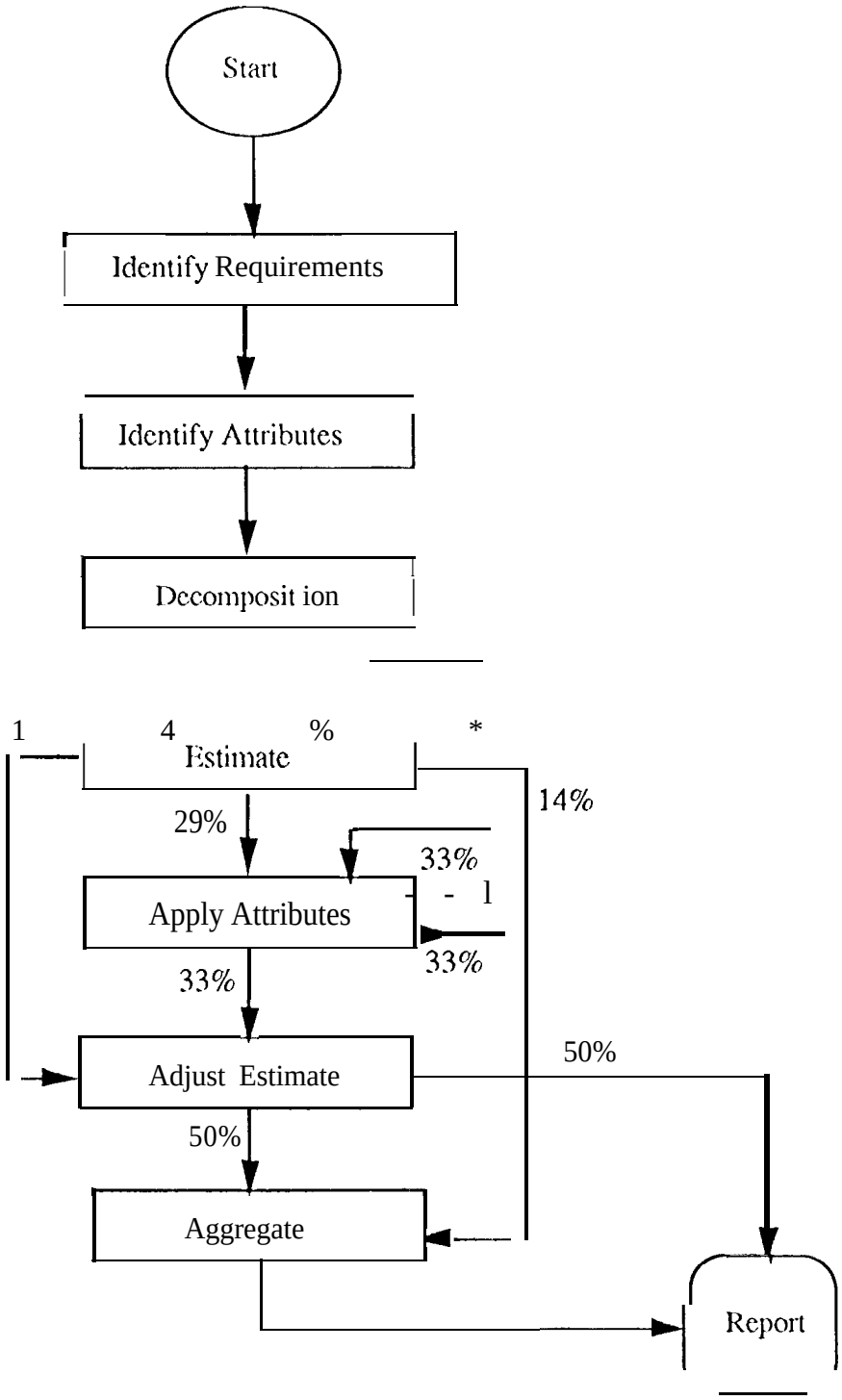

Figure 5 : Activity **Flow** for **Forecasters** Reporting Single Step Decomposition and Single or Sequential **Estimation** Steps

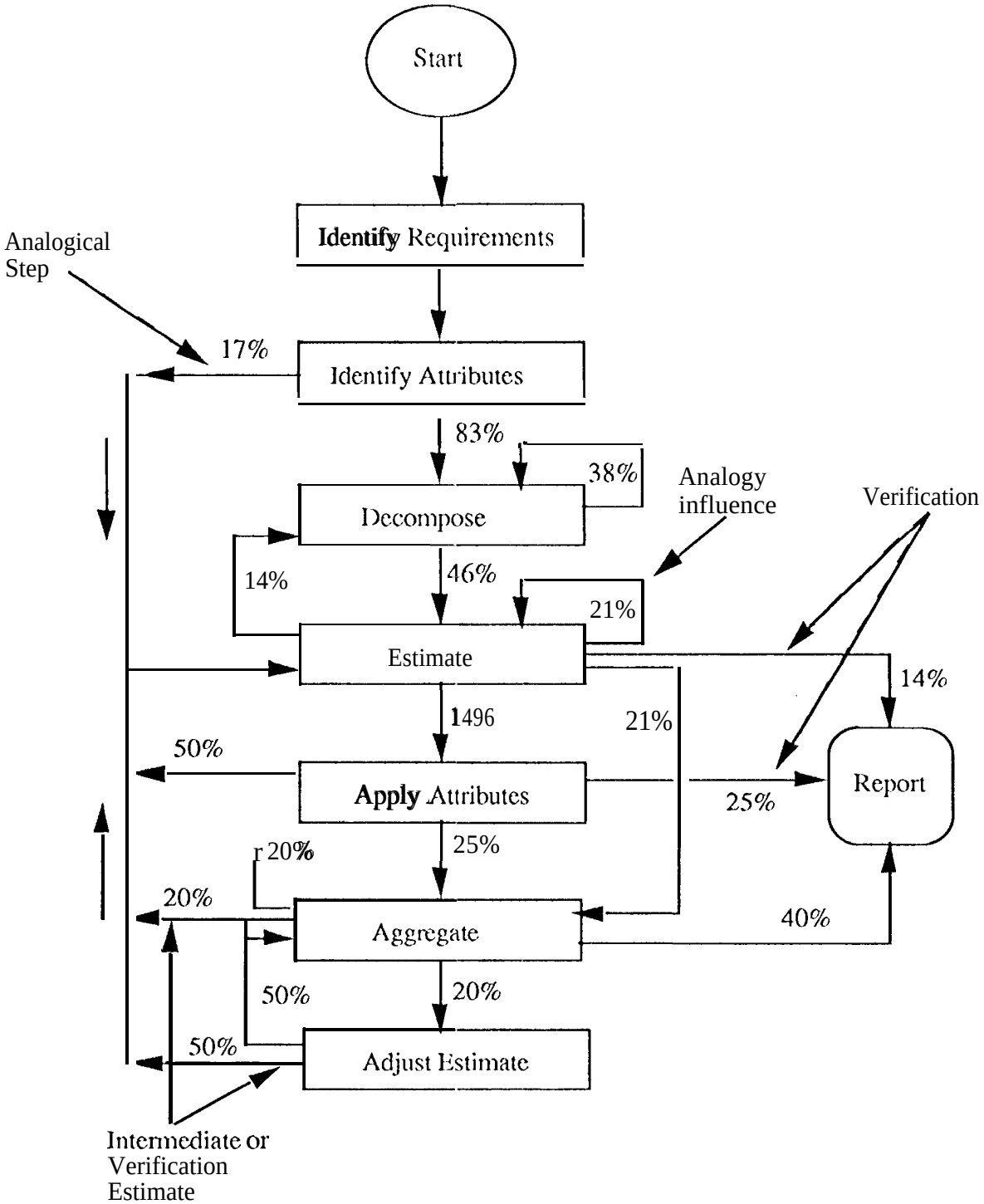

Figure 6 : Activity Flow for Multiple **Decomposition** Steps and Multiple Non-Sequential Estimation Steps

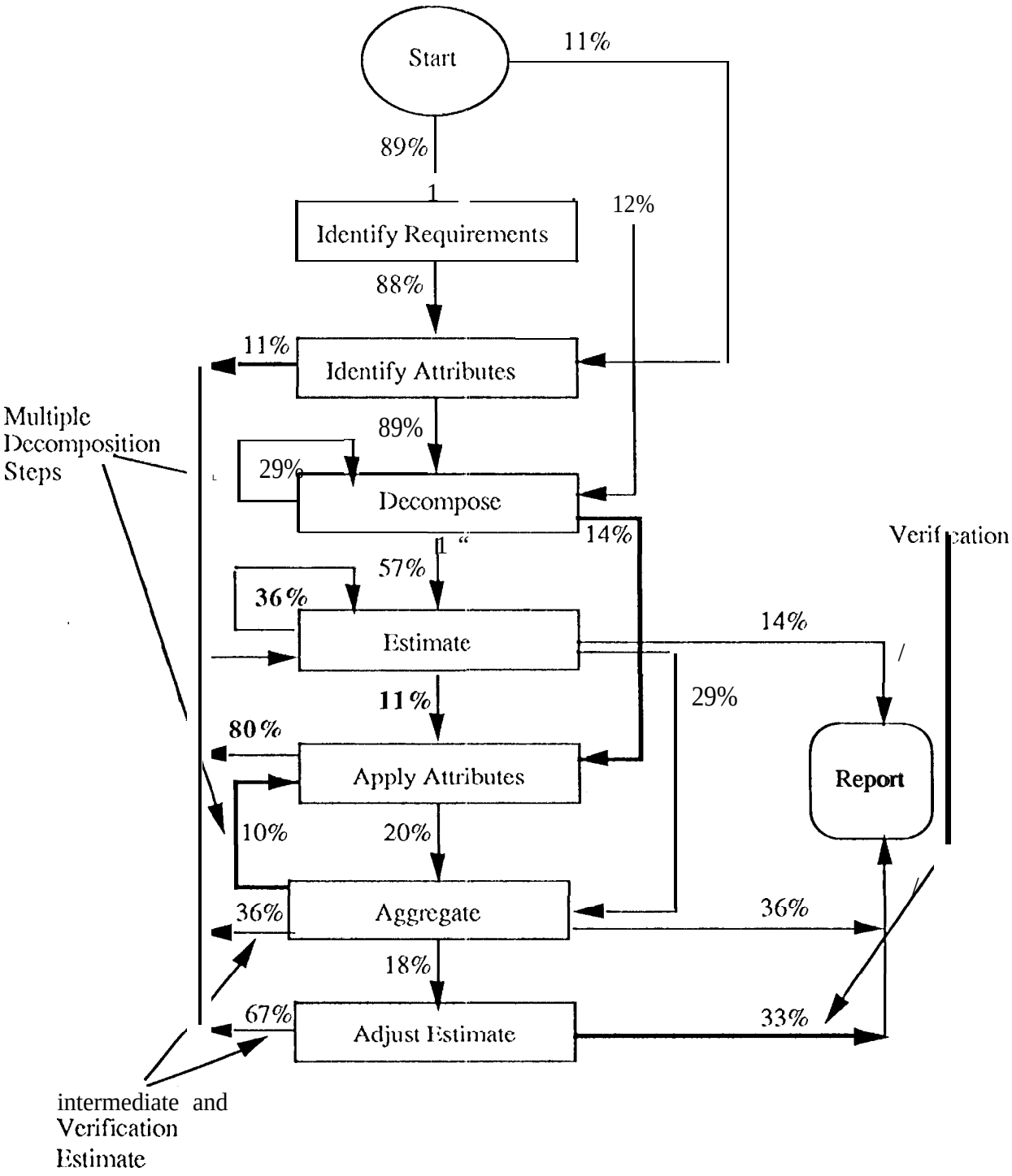

Figure 7 : Activity **Flow** for Cost **Forecasters Using** Analogy as a Major Component of '1'heir Mental Model

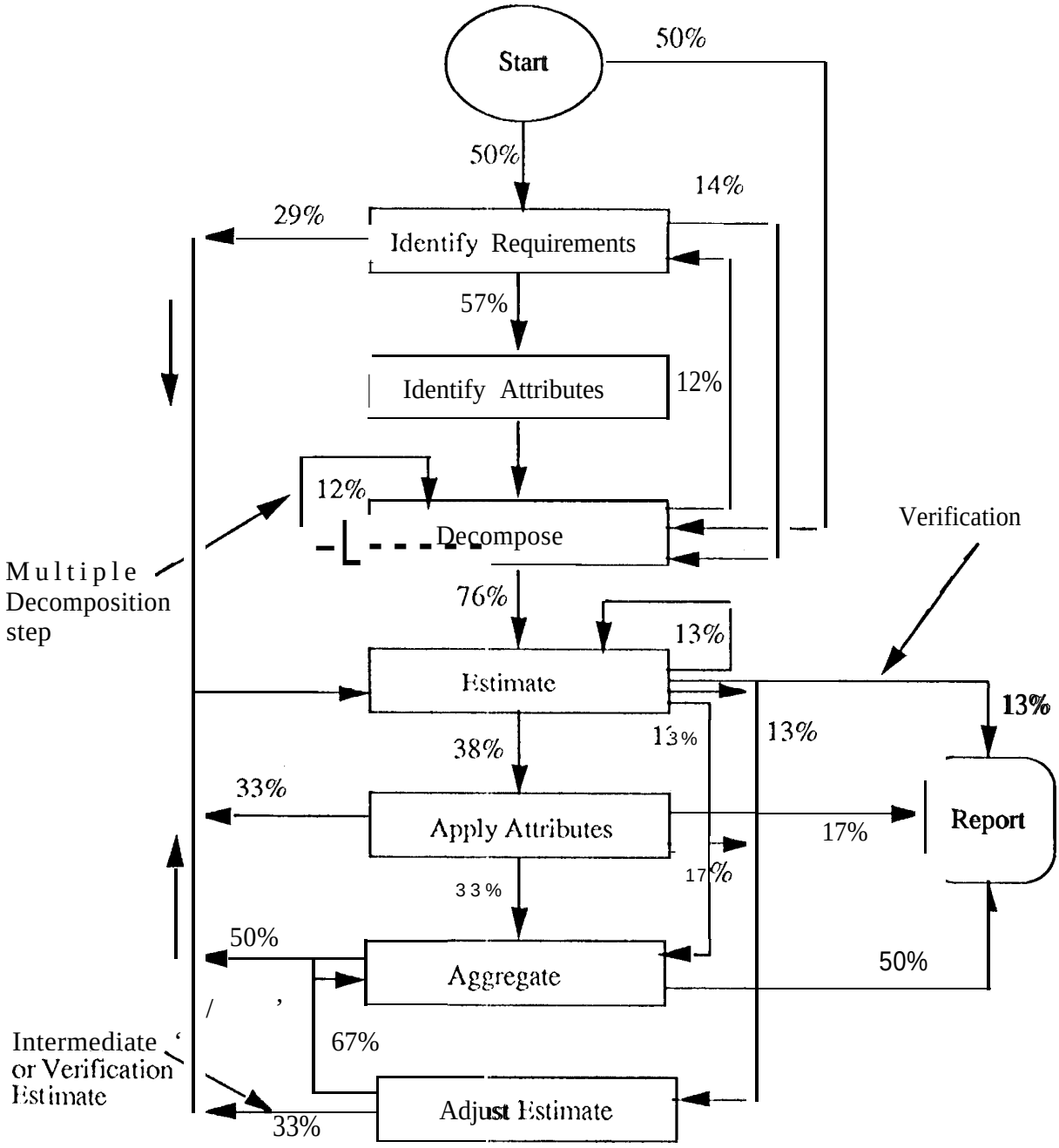

Figure 8 : Activity **Flow** for **Cost Forecasters NOT Using** Analogy as a Major Component of Their Mental **Model**

References

Ackerman, D. and Tauber, M., **Mental** Models **and Human-Computer** Interaction - 1, Elsevier, 1990.

Bunn, D. and G. Wright, Interaction of Judgmental and Statistical Forecasting Methods: Issues and Concerns, **Management** Science, Vol 37, May 1991, PP. 501-518.

Corbett, M. and J, Kirakowski, An Analogical Approach to Cost Intimation, The]'rocecdings of the European Software **Cost** Modeling Meeting 1992, Munich, Germany, May 27-29, 1992.

Cowderoy, A., Experience Based Estimation for Software Development, The **Proceedings** of the 1992 international Society of **Parametric** Analysts Conference, Munich, Germany, May 25-27, 1992, pp. CM-24-CM-51.

Ericson, K, and Simon, H., Protocol Analysis, Ml"I' press, 1984

Hihn, J. and Habib-agahi, H., Cost Estimation of Software Intensive Projects: A Survey of Current Practices, Proceedings of **the 13th International** Conference on Software **Engineering**, pp. 276-287.

Howard, M., "The Creation of a Research Model for Estimation", Proceedings of the **European** Software Cost **Modelling** meeting 1992 (ESCOM), Munich, germany, May 27-29, 1992.

Papoulis, A., Probability, Random Variables and Stochastic Processes, Mcgraw-Hill Jnc, 1991.

Phillips, D., Ravindran A., and Solberg, J., Operations Research Principles and Practice, Wiley, 1976.

Simon, H. A., Models of Man: Social and Rational, Wiley, 1954.

Slovic, P., Toward Understanding and improving Decisions, **Human** Performance and Productivity, ed. W. Howell and E. Fleishman, Lawrence Erlbaum Associates, 1982.

Vicinanza, S., Mukhopadhyay and Prietula, M., Software-Effort Estimation: An Exploratory Study of Expert Performance, Information Systems Research, vol 2, December 1991, pp. 243-262.

Appendix A: Vocabulary Mapping to Activity Taxonomy

Activity	Definition	Vocabulary (Linguistic Units)
Requirements Identification (RI)	The obtaining or retrieval of information. Key linguistic units are: read talk to obtain Some more complicated linguistic units which imply several events but are primarily encoded as EN are: review requirements study produce requirements	Read requirements Talk to experts talk to experts and sponsors to understand requirements better for new parts talk to experts to try to understand the sub-tasks talk to my group obtain budget information produce requirements because do not exist (AI) Study Review requirements
Attribute Identification (AI)	Atmbutes are key aspects of a task which are used m forming the system mental model and are also used as analogy discriminators and cost drivers. This is one of the main products of the analysis of the requirements. AI is generally described by the basic activity that was undertaken with the result that precise attributes are rarely specified at this point. At this point two types of attributes have been identified: product and process. Key vocabulary words are: identify, understand, analyze, include. Some more complicated linguistic units which imply several events but are primarily encoded as AI are: relate to some existing model take the baseline make assumptions find analogy	Identify standard Identify environment identify parts that should be prototyped identify complexity identify size identify scope identify functionality identify personnel quality include programmer experience include reliability relate to some existing model requirements analysis analyze requirements talk to experts and sponsors to understand requirements better. for new parts talk to experts to try to understand the sub-tasks consider similar projects what are the differences how much inheritance for inherited s/w use similar past applications make assumptions take the baseline tind analogy produce requirements because do not exist (EN)

Decomposition (DE)	The breaking down of a software entity (system subsystem, etc.) into smaller and simpler pieces. One of the most fundamental ways that we deal with complexity is to break a problem down into smaller pieces so that each piece can be analyzed separately. DE occurs throughout the costing process. The types of DE described in the current protocols are: WBS, functional, new vs inherited, requirements. Key vocabulary words are: breakdown identify sub-tasks break it into develop In some cases the product of the breakdown is mentioned without mentioning the activity itself. Therefore it is necessary to know the different types of software break-downs.	Breakdown to CSC level break it into modules breakdown to primitive requirements for new s/w breakdown into known and unknown pieces Identify subtasks needed to complete task Develop preliminary design develop s/w wbs develop WBS develop functional sub-tasks Functional Breakdown WBS to small/medium level separate inheritance from new for new parts look around for analogy assume design environment functional design H/W S/W Architecture WBS by functionality bottom up from component level

Appendix A Continued

Estimation (ES)	The prediction of future cost and other key project management dimensions. The main items forecasted are: schedule size effort dollars FO can be further divided by type of technique used: analogical expert judgement explicit analogy algorithmic rules of thumb CERS Key vocabulary words are: use (analogy, rule of thumb) estimate (SLOC, effort) cost	use analogy to est effort use analogy use best judgement use rule of thumb to estimate effort at module level use expert judgement to est effort **use expert** programmers to est effort used basic COCOMO use rules of thumb and past experience to est effort use rules of thumb to adjust SLOC for unknowns ??? use analogy and ratios of sameness for inherited s/w use similar past applications **use rules** of thumb to get SLOC/Requirement use expert judgement to convert requirements to effort for new s/w unknown pieces use rules of thumb known pieces use formal analogy use analogy to est effort sit with programmers use analogy to estimate effort estimate effort estimate SLOC for similar parts est effort for new parts est SLOC/ IO SLOC/day estimate effort based upon 8 SLOC/day experts make estimates based on seat of pants for each component estimate complexity for each component estimate SLOC by complexity cost each sw module compute equivalent new SLOC industry standard for $/LOC multiply by dollars/SLOC determine schedule of activities and assume 1 module/month for 3 people for new parts look around for analogy DSN runs around 10-12 SLOC/day operations is 11,000/ month get cede effort and schedule from CDE's(EN)

Appendix A Continued

Adjustments (AD)	Multipliers used independent of the system being costed. Usually applied at a higher level then attributes. Consists of adjustments for purposes of risk scaling bias(error) Key vocabulary word is add percent.	add % for risk use 50% safety factor for new parts do same but add fog factor add 10-20% based upon % of new code multiply by 5 to get total effort add a percent to get total effort use section cost rates double rate to add cm, test, doc etc. add 100% for test and doc add 5070 for requirements and design add 200% see how well estimate before see how well estimated before and use fog factor to adj estimate could be 100%

Appendix A Continued

Attribute Application (AA)	The explicit use of the system attributes to discriminate between systems for purposes of analogical comparison or as cost drivers when using an algorithmic approach. Identification primarily-depends upon specific mention of attribute. At this point two type of attributes have been identified: product and process. While there is less homogeneity in the linguistic units some common phrases are: adjust use (fog factor) add (change, fog factor, etc.) multiply	Use fog factors : language, personnel quality put on physical limitations old task used assembly new is high level language complexity is the same more COTS personnel quality personnel availability assign complex parts to more competent persons adjust based on cost drivers manpower quality. manpower availability schedule constraints degree of flexibility use analogy to map the requirements to similar tasks multiply fog factor to get sub-task effort add change for new work modified code level of language use analogy and ratios of sameness assume experience is average or above for new parts do same but add fog factor for each component estimate complexity personnel quality determines size of fog factor
Aggregation (AG)	The combination of forecasted values associated with the system pieces produced by decomposition. Key vocabulary words are: add-up SRM (JPL resource management tool)	a up effort for all modules add all sub-tasks to get effort add them up add up pieces add up SLOC add up SLOC SRM to get dollars bottom up from component level