Session 4: Software Quality

Gerry Heller, CSC, Discussant

Rex D. Pendley, Computer Sciences Corporation

Philip A. Hausler, IBM Corporation

Norman F. Schneidewind, Naval Postgraduate School

Development and Application of an Acceptance Testing Model

Rex D. Pendley
Caroline H. Noonan
Kenneth R. Hall

Computer Sciences Corporation
10110 Aerospace Road
Lanham-Seabrook, MD 20706

Phone: (301)794-1375
FAX: (301)794-4377

The process of acceptance testing large software systems for NASA has been analyzed, and an empirical planning model of the process constructed. This model gives managers accurate predictions of the staffing needed, the productivity of a test team, and the rate at which the system will pass. Applying the model to a new system shows a high level of agreement between the model and actual performance. The model also gives managers an an objective measure of process improvement.

INTRODUCTION

Acceptance testing is the process whereby users of a system satisfy themselves that the system meets their requirements. The Flight Dynamics Technology Group (FDTG) of Computer Sciences Corporation (CSC), under the Systems, Engineering, and Analysis Support (SEAS) contract, has long supported Goddard Space Flight Center's (GSFC's) Flight Dynamics Division (FDD) in acceptance testing systems developed or modified for satellite mission support. This paper reports an analysis of FDD acceptance testing developed in the FDTG over 4 years, starting with the Cosmic Background Explorer (COBE) attitude ground support system (AGSS) in 1988.

In our discussion we cover the collection and evaluation of metric data, describe the testing model we have developed, and discuss the organizational constructs needed to manage this measurement-based approach to testing. Finally, by discussing the application of this model to software recently developed for the Wind and Polar spacecraft of the International Solar and Terrestrial Physics (ISTP) mission, we show that we have made acceptance testing predictable and that improvements in the testing process have a measurable effect.

Our model permits a testing manager to predict the resources needed for testing as a function of the system size and to plan

for a fixed team or a fixed duration. The capacity of a trained team of testers is quantitatively characterized with a learning curve that gives productivity chronologically, arming the manager with the information needed to construct a realistic schedule. Finally, the model uses the schedule to generate curves showing how the test reporting will proceed and how the software will perform (pass rate) as the testing progresses. Put together, all these projections give the manager a realistic plan as well as a schedule and performance profile by which to gauge the progress of the testing process.

DEVELOPMENT OF THE MODEL

Organization and Roles

Reflecting the organization of the FDD, the FDTG divided itself into an analysis group, responsible for specifying requirements, and a development group, responsible for developing the software to meet those requirements. The analysis group was responsible for acceptance testing this software. This division of labor facilitated communications between FDTG managers and FDD managers, used the skills and interests of analysts and developers to maximum advantage, and ensured that acceptance testing was independent. The FDD's Software Engineering Laboratory (SEL) has long supported development teams with quantitative process analysis, giving development teams a model for planning their side of the acceptance testing process (Reference 1). Analysis teams lacked such a resource and therefore sought to understand and improve their part of the testing process as described here.

While the manager of a test team can follow that team's progress, assembling data from multiple teams requires a way to get all the teams to collect the same data and some way to gather and analyze these data. This synthesis was accomplished by using the Total Quality Management (TQM) concept of a process-evaluation group made up of persons performing the work. In the FDTG this group was called the acceptance testing (AT) process group. The AT process group could not function without a means of access to data and a way to implement its findings. The necessary element was commitment at a management level that spanned the test teams. In the FDTG, test teams were generally led by an analysis section manager (SM) and task leader (TL); the Launch Support department manager (DM) was in the position to provide management leverage to the AT process group. Because some test efforts were in other departments, the Flight Dynamics Analysis Operation (FDAO) manager gave additional management authority for AT process group needs and recommendations.

Another important benefit of this approach is the heightened communication among the principal participants. Figure 1 shows the relationships among the various groups and managers of testing efforts, clearly depicting the lines of communication.

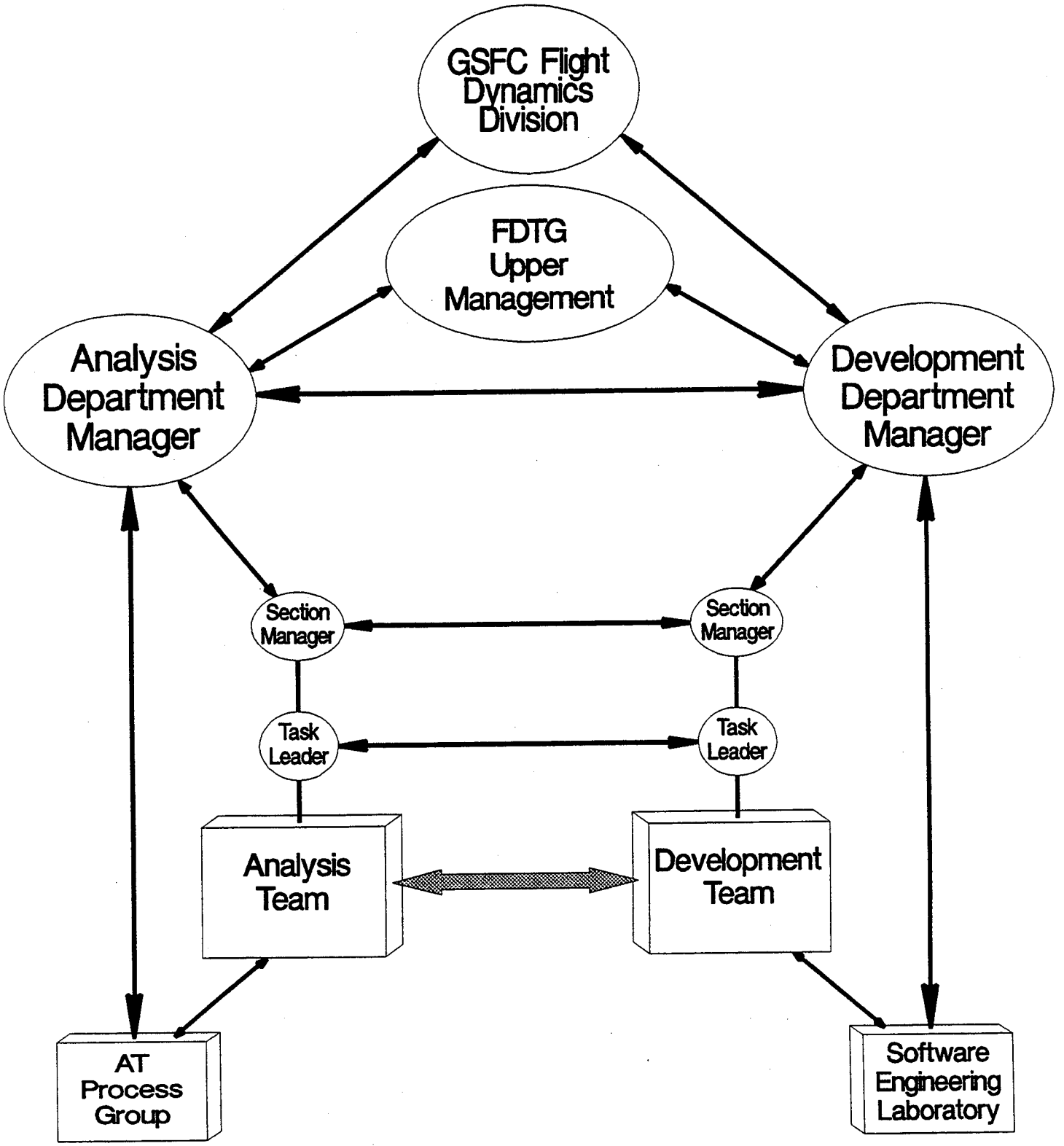


Figure 1 - The well-defined process facilitates communications at all levels among the participants in testing. Within the analysis team, the three-way interaction among the test team, the AT process group, and the DM optimizes the development of testing technology and its application to FDD projects.

One management practice that contributed significantly to the success of this approach was a weekly review of all testing efforts, based on a common status report that compared each group's progress with its model-based schedule and pass curve. The DMs, SMs, TLs, and AT process group facilitator all attended, and they were able to exchange information freely. The result was that the analysis and development teams had a common view of testing status and were able to deal directly with points of contention.

Nature of Systems Under Test

The software systems under consideration in this paper are attitude ground support systems (AGSSs), developed as part of the ground support software for scientific satellites flown by GSFC. An AGSS comprises 4 to 10 individual programs used to calculate spacecraft attitude, calibrate the attitude sensors, predict view geometries and periods for various sensors, and perform utility functions such as data management. Some of the programs are interactive graphics programs, and others are batch systems; all run on IBM mainframes computers under the multiple virtual storage (MVS) operating system. The systems discussed here are largely new development in FORTRAN with no more the 30-percent reuse.

The most salient characteristic of an AGSS for the purposes of test modeling is its size. The systems from which the model was constructed (Reference 2) are summarized in Table 1, with the sizes given in thousands of source lines of code (KSLOC).

Table 1 - Sizes of Systems Tested

Satellite	AGSS Size (KSLOC)
COBE	178.7
GRO	266.9
GOES	128.8
UARS	319.5
EUVE	268.5

No distinction is made between new and reused lines of code, since for the purpose of acceptance testing, the entire system had to be verified against its mission requirements. The spacecraft for which AGSSs were developed were the Cosmic Background Explorer (COBE), launched in November 1989; the Gamma Ray Observatory (GRO), deployed from the space shuttle in May 1991; the Geostationary Operational Environmental Satellite (GOES), scheduled for launch in March 1994; the Upper Atmosphere Research Satellite (UARS), deployed from the space shuttle in September 1991; and the Extreme Ultraviolet Explorer (EUVE), launched in June 1992. The UARS and EUVE AGSSs were developed in parallel efforts and tested together in a large combined test

effort. In all cases acceptance testing was performed only on the final build, meaning that errors had to be corrected before testing was finished, as corrections could not be made in future builds.

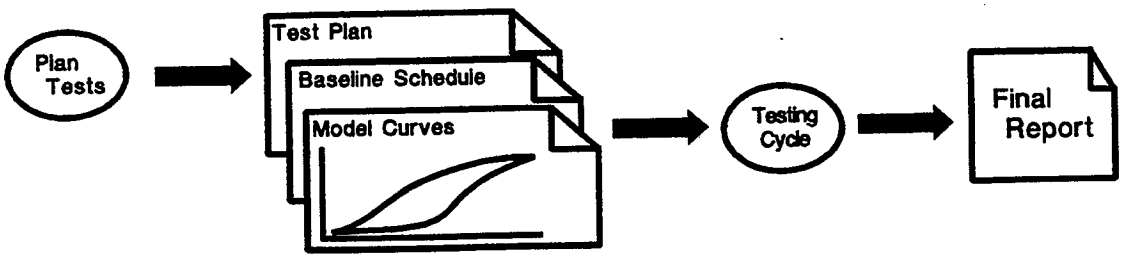
Process of Acceptance Testing

Figure 2a shows the context of the FDTG acceptance testing process in schematic form, highlighting the three necessary prerequisites for testing: a test plan, a baseline schedule, and a predicted performance curve. The last two items are supplied by the model discussed below. Figure 2b amplifies the cycle of the testing process, with the appropriate division into analysis and development activities. Data items on the dashed vertical line are shared between the groups. Note that the bubble containing assessing the status and recommending continuation or completion is a joint activity of the analysis and development teams. Each decision to continue implies that a corrected load module will be placed under test; in the FDD a single cycle is called a round of testing. This diagram represents the AT process group baseline process as of January 1992.

In keeping with the preceding definition of acceptance testing, the test (or analysis) team sought to demonstrate that the system met their requirements. In defining the requirements, the analysis team generally first defined mission requirements: what the FDD needed to do to support a particular satellite mission. Recognizing that some part of the mission requirements would be met by people performing procedures, they identified the subset of requirements levied on software, the software requirements. The development team developed or modified a system according to the software requirements, but the analysis team tested with the prerogative of evaluating system acceptability in light of mission requirements. The minimum criterion for acceptance was a single demonstration that a particular mission requirement was met. Testing against this larger set of requirements captured flaws in requirements definition and in the allocation of software requirements, and thus led to a larger number of corrections than would testing against software requirements alone. On the other hand, the practice gave added assurance that the FDD team would have the necessary support software.

This process was difficult to plan accurately and hard to monitor. Planning, in terms of staffing, schedule, and expected performance, was impeded by a lack of historical records of sufficient detail to permit constructing a predictive model. The lack of accurate histories thus led to unrealistic plans and unmet expectations. Monitoring the test process was difficult because of the large volume of test reports that required review and summarization by hand. The analysis manager also needed to direct the team daily to keep test evaluation moving and to reassign testers as software availability changed.

(a) Testing Context



(b) Testing Cycle

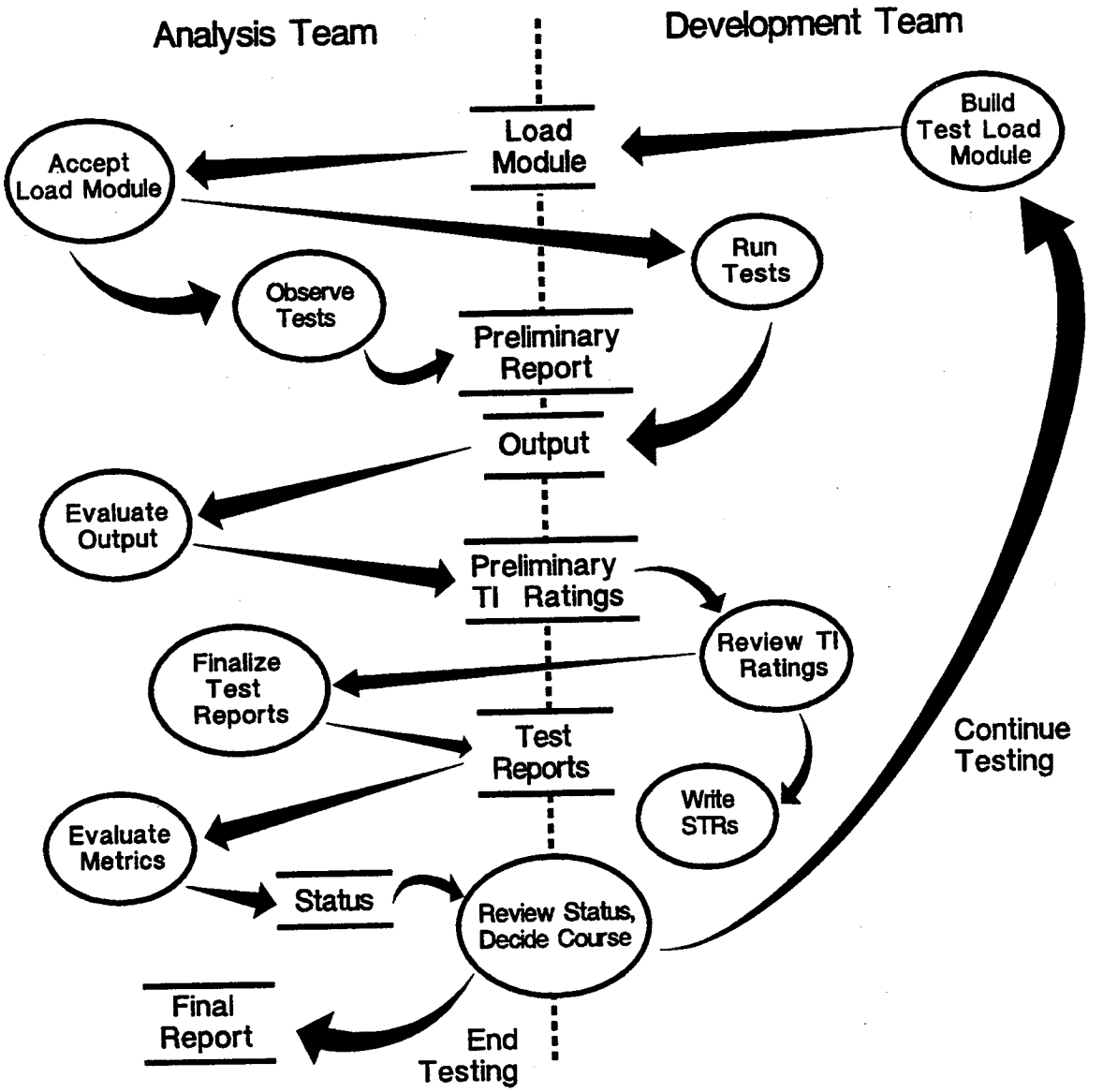


Figure 2 - The AT process group defined all stages of the acceptance testing process.

In spite of these difficulties, a feature of FDTG test planning offered a basis on which to construct a model. In a typical FDTG acceptance test plan, tests were constructed on the basis of operational scenarios. Each test was mapped to a set of requirements whose fulfillment it demonstrated; in turn, the results for the requirements were reported as individual test items (TIs) for that test. Each TI was given a ranking on the five point scale in Table 2.

Table 2 - Test Item Rating Scale

Level	Meaning
1	Item cannot be evaluated
2	Item fails
3	Item fails but a workaround exists
4	Cosmetic failure
5	Item passes

For a typical AGSS, there were 40 to 50 tests, each with an average of 30 to 40 TIs, for a total of 1200 to 2000 TIs. These numbers are statistically significant, so by analyzing the status of the TIs as a function of time, it should be possible to construct an empirical model of the testing process. By correlating the labor hours, it should be equally possible to extend the model to predict schedule and staffing.

Metric Data and Analysis

The COBE analysis manager and test team began the first data analysis during COBE AGSS testing in 1988. The AT process group was formed shortly after that, initially as a pilot group trying out the Oregon Objectives Matrix (OMX) technique. The OMX approach was unsuccessful, and the team was soon re-chartered as a process analysis group. They made the first applications of their findings with acceptance testing of the GRO AGSS later that year. They have supplemented test teams for all AGSS test efforts in the FDTG since then.

Table 3 summarizes the testing data collected by the AT process group through September 1990 (Reference 2). For each AGSS, the table lists the total cumulative test labor in staff-hours, the system size in KSLOC, the number of TIs in the test plan, and the total number of TIs evaluated, including all those re-evaluated for retesting and benchmarking purposes. This last quantity is the best basis for modeling acceptance testing because the total test effort includes repeated testing after error correction and regression (benchmark) testing. Note that the size of the combined UARS/EUVE AGSSs is less than the sum of individual AGSSs (cf. Table 1). This difference arises because the two systems, jointly developed and tested, share a significant amount of code.

Table 3 - Summary of Testing Metric Data

AGSS	Staff-Hr	KSLOC	TIs	TIs Evaluated
COBE	7124	178.7	1771	5677
GRO	8771	266.9	925	3626
GOES	5004	128.8	1053	2816
UARS/EUVE	18884	390.5	2723	6054

The entry of 6054 TIs evaluated in Table 3 is the sum of the shared code and the unique code from the UARS and EUVE AGSSs.

The AT process group combined these figures into ratios in analyzing the data and constructing their model. Table 4 lists three ratios of interest. The first two, hours per TI evaluated and hours per KSLOC, are reflections of testing productivity,

Table 4 - Analysis of Testing Metric Data

AGSS	Hr/TI	Hr/KSLOC	TI/KSLOC
COBE	1.3	40	32
GRO	2.4	39	16
GOES	1.8	39	22
UARS/EUVE	3.1	48	16

and are the key to predicting the size and duration of a test effort. Examining these ratios reveals that, while hours per TI varies by over 100 percent from the lowest to highest value, the hours per KSLOC varies by only 20 percent. The third ratio, called the TI density by the AT process group, shows variation similar to that of the hours per TI. Clearly, hours per KSLOC affords the best basis for making a size and duration prediction.

Plotting total staff-hours against KSLOC, as shown in Figure 3, suggests that the relationship is linear. Fitting these data points with a linear regression gives Equation 1:

$$y = 48.85x - 1005 \quad (1)$$

where y = total test effort in staff-hours
x = system size in KSLOC

Obviously the negative intercept is not realistic. Constraining the intercept to zero, on the grounds that zero KSLOC implies zero test time, yields a slope of 43.1 staff-hours per KSLOC. Alternatively, the explanation could be that for small systems the test effort depends less linearly on the total system size, making the effort resemble the logistic curve characteristic of resource use by processes that carry a certain minimum overhead

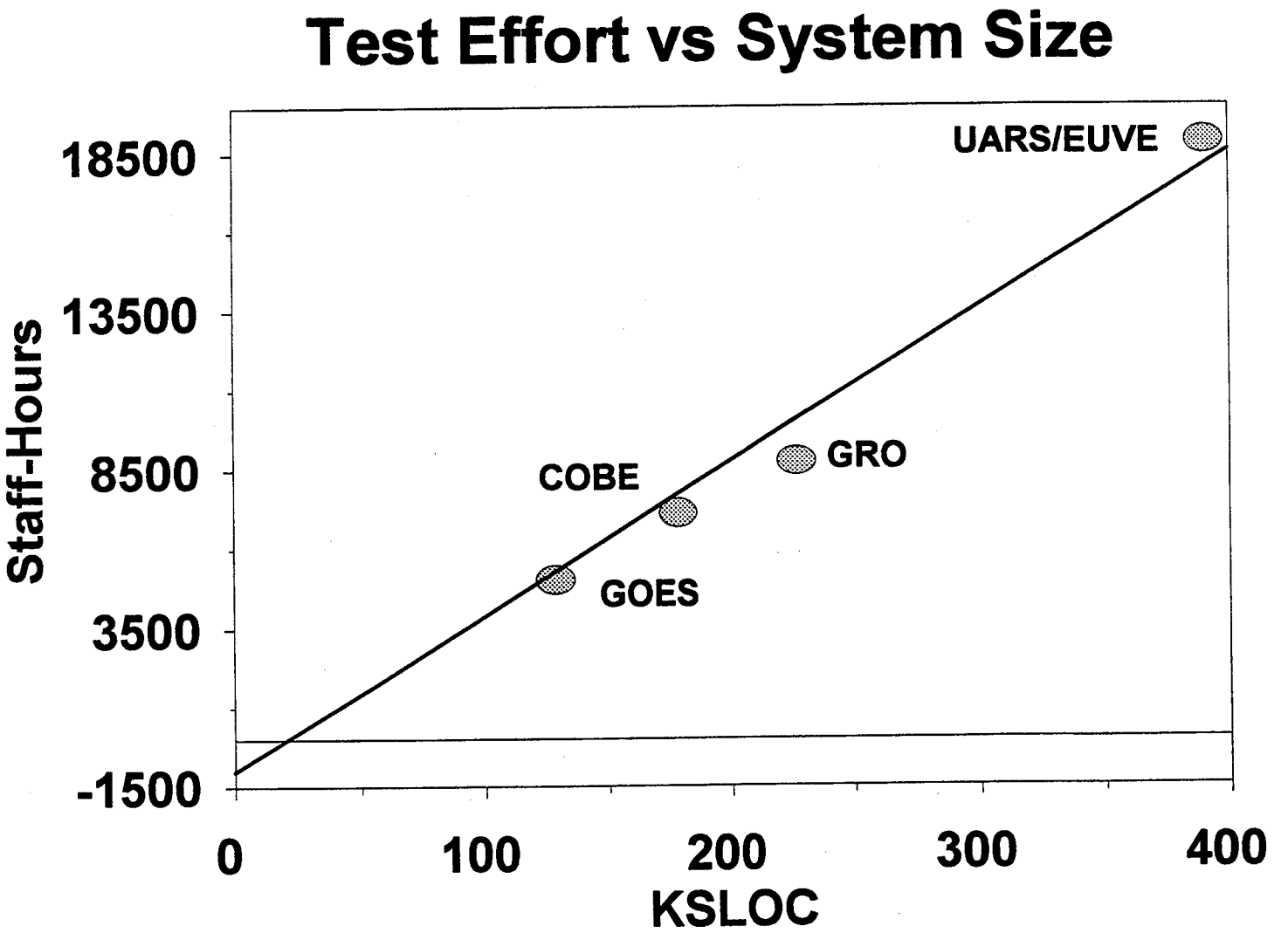


Figure 3 - The number of staff-hours to acceptance test a system fits a straight line as a function of system size.

regardless of total size. The data reported here are then explained as falling in the central or linear part of such a curve, and therefore contain little information about small system behavior.

Another implication of Equation 1 is that there is no cost in trading staff for schedule. In other words, if acceptance testing costs are accurately modeled by this expression, twice the staff will accomplish the work in half the time. This tradeoff is clearly not plausible at the extremes, but within the range of observed data, the AT process group found that time and staff were highly interchangeable. For example, the combined UARS/EUVE testing effort had a hard completion deadline that was met by expanding the staff according to the model. The enlarged group finished 1 week (out of 33 weeks) after the target deadline (which was 4 weeks in advance of the hard deadline as a management buffer).

Through further analysis of the productivity data, the AT process group found that the productivity of a test team varied over time in a predictable way. Figure 4 shows the hours per TI as function of the completeness of testing. This graph contains data from GRO testing and from testing of two previously undiscussed AGSSs: the Solar, Anomalous, and Magnetic Particle Explorer (SAMPEX) AGSS, and the International Solar and Terrestrial Physics (ISTP) mission AGSS. The SAMPEX and ISTP data are discussed below. All the curves show a characteristic peak, meaning lower productivity, at the beginning of a test effort, an example of a learning curve. The peaks are significantly higher than the steady-state values that appear further into the testing period, but are relatively narrow, lasting roughly 10 to 15 percent of the testing period. The significance to model construction is that a sharp, narrow peak will have little effect on the overall average productivity but will be very significant during the peaks's duration. Planning a test effort without allowing for reduced productivity at the beginning will put the work behind schedule from the outset, catching up only slowly as the team works at the average rate for the rest of the test effort.

The preceding analysis addresses productivity, but a successful model should also address the performance of the software as well. That is to say, how many rounds of testing can be expected before all the TIs are passed? The AT process group found that the data could be fit with two models, one assuming five rounds, called the 5 Round model (for five testing rounds), and one assuming seven rounds, called the 7 Round model (for seven testing rounds). Figure 5 depicts these models graphically.

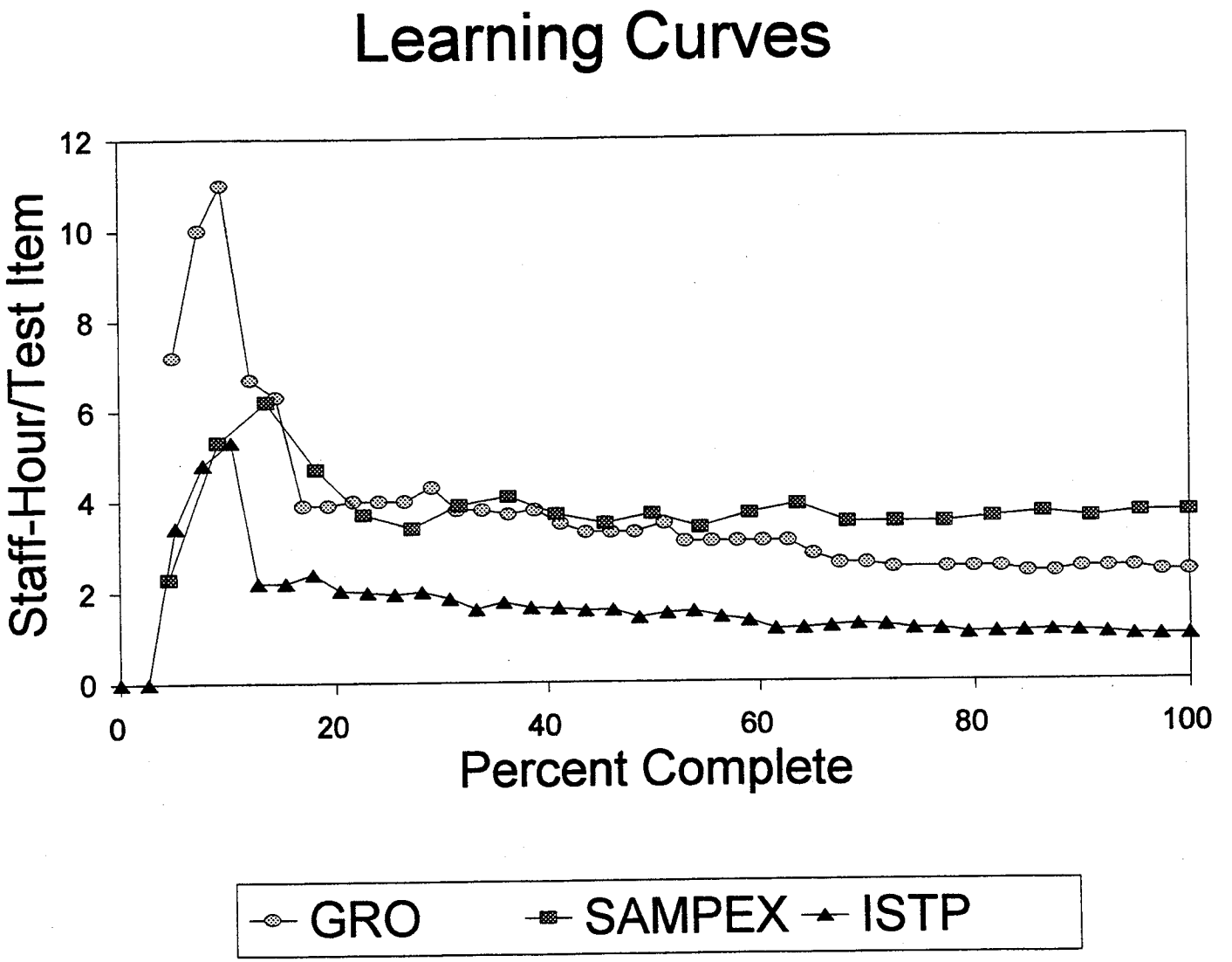


Figure 4 - The plot of hours to evaluate a test item shows a startup peak characteristic of a classic learning curve.

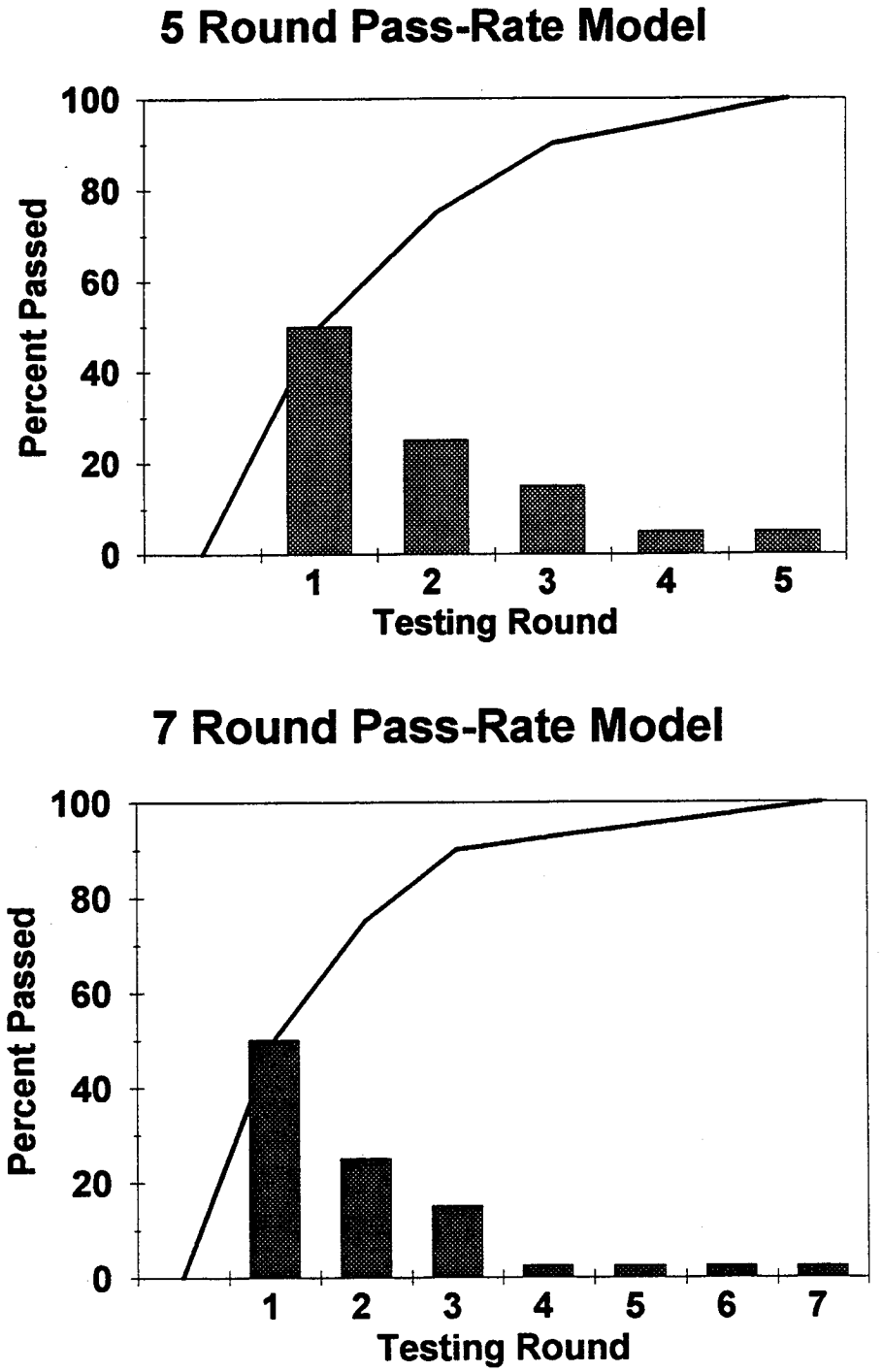


Figure 5 - Two pass-rate models can be used to fit the data. The upper graph shows the cumulative (line) and incremental (bars) percentages of items passed over five rounds of testing. The lower graph gives the same information over seven rounds of testing.

Table 5 gives the incremental pass rates - that is, the additional number of TIs passed in each round - and the cumulative pass rates for the two models.

Table 5 - Incremental and Cumulative Pass Rates (Percentage)

Round	1	2	3	4	5	6	7
5 Round Model							
Incremental Rate	50	25	15	5	5		
Cumulative Rate	50	75	90	95	100		
7 Round Model							
Incremental Rate	50	25	15	2.5	2.5	2.5	2.5
Cumulative Rate	50	75	90	92.5	95	97.5	100

Statement of the Model

By combining the three pieces of analysis described in the previous section, the AT process group was able to derive the planning paradigm shown in Table 6.

Table 6 - A Paradigm for Acceptance Test Planning

1. Size the total effort via method (a) or (b).
   (a) Total hours = KSLOC * 48.85 hours/kSLOC - 1005 hours
   (KSLOC = 1000 source lines of code)
   (b) Total hours = number of TIs * 3.2 hours/TI

   Convert hours to staff-months

2. For a fixed team size, divide staff-months by team size to determine duration.

   For a fixed duration, divide staff-months by duration to determine team size.

3. Based on anticipated software availability, the learning curve, and the average test turnaround time of 6 days, construct a schedule based on available test capacity in terms of TIs reported per week.

4. Use the schedule to plot the number of TIs to be reported each week.

5. Use the pass-rate model to estimate and plot the number of TIs expected to pass each week.

This planning paradigm yields accurate cost and schedule predictions, as well as realistic expectations for technical performance at each stage of testing. Step 1 of this paradigm gives two simple expressions for estimating total effort. Derived from measurement data for five AGSSs, the two methods should give the same number of hours within 15 percent. Step 2 implies that there is no penalty for trading off level of effort and duration, which is probably not true in general. However, for the cases studied, this simple linear relationship holds. The learning curve referred to in step 3 simply gives the average number of hours needed to evaluate a test item as a function of time. As expected, this curve shows that more hours per item are needed at first, leveling off into a fairly constant value. The test manager thus has the number of TIs expected to be reported each week and can project that performance curve in step 4. Finally, the AT process group determined the percentage of reported items that will pass (level 5) as function of testing round, so that a pass-rate curve can be generated from the report and testing schedule. The analysis manager uses a standardized spreadsheet to set up these model curves and updates it with actual performance data weekly. Examples of the model curves are given below in the discussion of applying the model to the ISTP AGSS.

AN APPLICATION OF THE MODEL

ISTP AGSS acceptance testing was planned and executed taking full advantage of the AT process group's model, training, and advice. Planning took place in August 1991, with AGSS testing beginning in November 1991 and ending in August 1992. This AGSS is large, with 174 kSLOC and 2117 test items, and consisted mostly of new development. Applying Equation 1 yields a predicted expenditure of 7495 staff-hours and the actual expenditure was 7852 staff-hours, an agreement within 5 percent.

Figure 6 shows the predicted TI reporting and pass curves based on the 5 Round model. Because the various programs constituting this AGSS were available for testing on different dates, each was projected separately. The curves represent the sum of curves for all the individual programs, and they are typical of projected performance curves.

Figure 7 adds the actual performance curves to the predicted curves. Note the accuracy of the predicted curves. The only large deviation in the pass-rate curve comes about midway through the test period, and it largely reflects a negotiated delay in receiving a new load module to test, as can be seen in the flat spot in both the reporting and pass-rate curves. At the direction of the FDD, the test effort was extended to an additional load module beyond the planned five in order to correct some additional errors. The final pass rate was 99.3

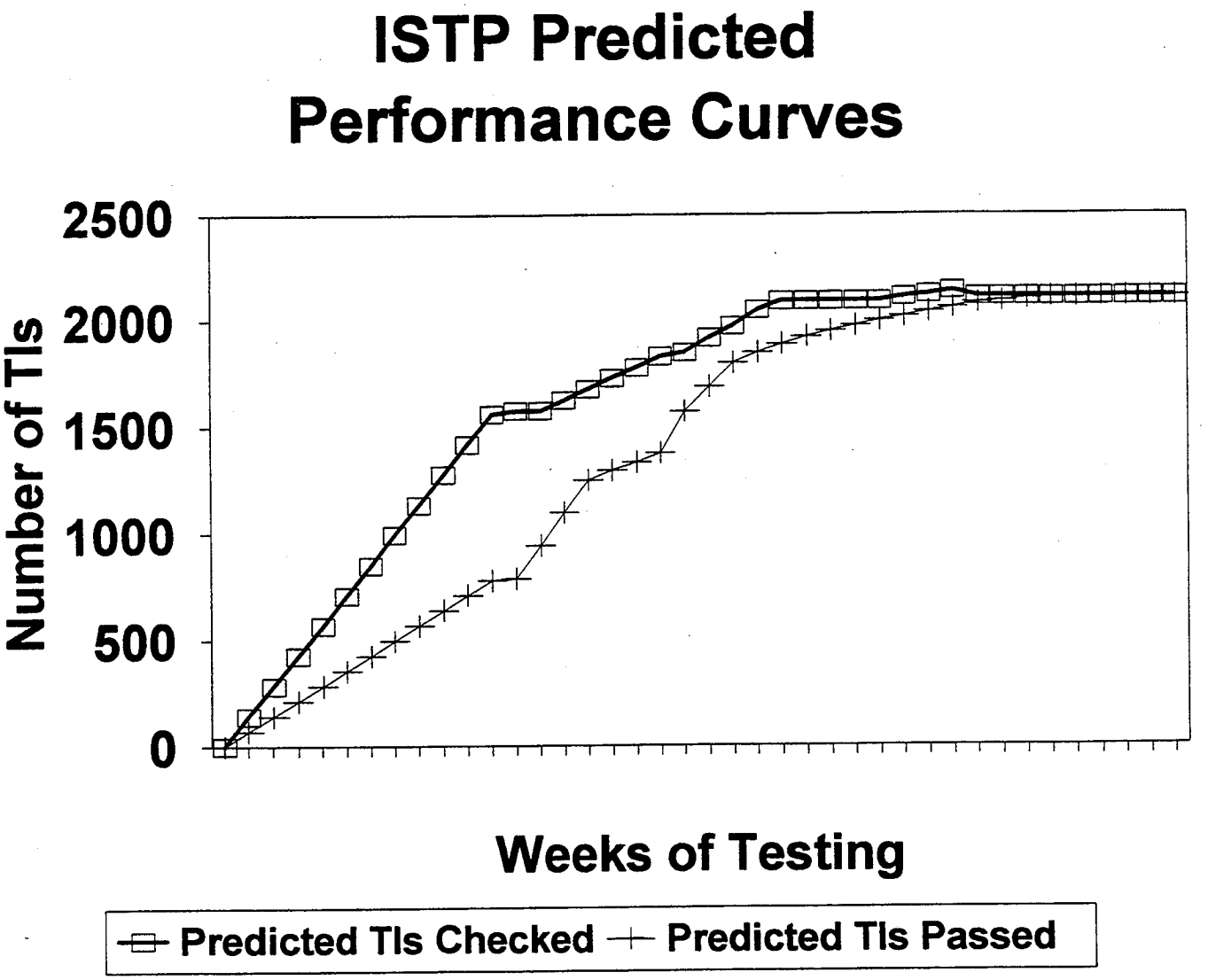


Figure 6 - Using the AT process group model to plan ISTP AGSS acceptance testing yielded a pair of curves projecting TIs checked and passed over time.

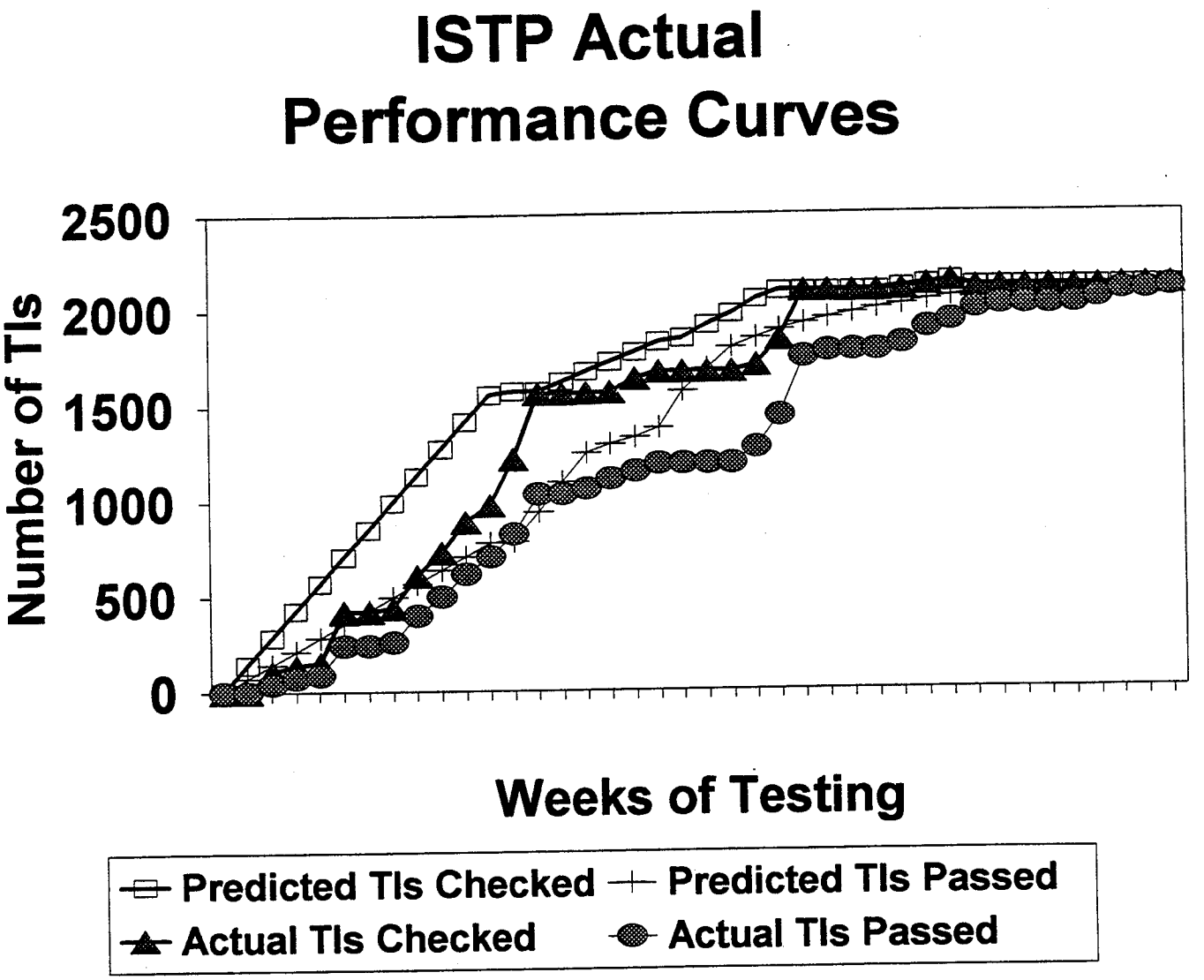


Figure 7 - Testing of the ISTP AGSS closely followed the predicted behavior.

percent, well above the 98 percent at which other recent AGSSs were accepted.

The ISTP team made an interesting innovation in attacking a subtle problem in the computation-intensive differential correction (DC) subsystem. This subsystem calculates attitudes and other parameters based on observation vectors. It involves complex matrix arithmetic, and it was found to yield subtle errors. The analysis manager organized a team of developers and analysts dedicated to the problems of this subsystem. They used a PC-based symbolic mathematical tool to model the DC algorithm and calculate intermediate values for comparison to the mainframe output. This approach allowed the testers to isolate the faults precisely and avoid extensive code reading.

CONCLUSIONS

The FDTG's AT process group has defined an important process in the mission lifecycle. In doing so, they have improved the management of testing by making it predictable and easily monitored. By setting realistic expectations, the analysis team better serves the FDD and has an improved relationship with the development team. Within the analysis team, the testers feel more empowered and thus able to affect their work.

Not only was the AT process group able to define an accurate, quantitative approach to acceptance testing, but they were also able to propose and act on improvements to it. One of their biggest successes came from analyzing the learning curves: the time needed to evaluate a test item over the course of a test effort. Figure 4 shows these values for three test efforts: the GRO AGSS, the SAMPEX AGSS, and the ISTP AGSS. The GRO values are the earliest, and show the characteristic initial peak of a learning curve. The AT process group devised training materials and courses so that new test teams were better able to start a new test effort; the SAMPEX and ISTP curves reflect the impact. Table 7 shows that for SAMPEX the ratio of the initial peak to

Table 7 - Effect of Training on Learning Curves

	Peak/Average	Average
GRO	3.5	3.8 hr/TI
SAMPEX	1.7	3.8 hr/TI
ISTP	3.6	1.6 hr/TI

the sustained rate decreased by 50 percent compared to GRO, but the average levels are comparable. Direct comparison of the peaks is misleading because GRO TIs were defined differently from SAMPEX TIs. The AT process group trained the ISTP test team more extensively and coached the testers throughout the ISTP effort.

SAMPEX and ISTP test items were similar, and comparing those two curves shows not only a 15-percent reduction in the startup peak, but much more significantly a 58-percent reduction in the average hours per TI.

Acceptance testing remains labor intensive, which results from the need to verify extensive computations with hand calculation, and from the number of rounds of testing needed to bring a program up to an acceptable level of compliance with requirements. Quantitative process analysis offers a medium through which this problem can be attacked. Furthermore, because a data-based performance baseline is established, it is possible to determine objectively that an innovation has made an improvement. Combining process analysis with a data base of measurements positions the FDTG and the FDD to work together to refine acceptance testing continually into the future.

ACKNOWLEDGMENTS

The authors wish to thank Mr. Theodore Z. Willard of CSC for his analysis of COBE testing data. They would also like to thank Dr. Gerald T. Page for his support and encouragement of this paper.

REFERENCES

1. Landis, L., McGarry, F. E., Waligora, S., et al., Recommended Approach to Software Development (Revision 3), SEL-81-305, June 1992.

2. Hall, K. R. et al., A Study of Acceptance Testing in the Flight Dynamics Technology Group, CSC/TM-92/6043R0UD0, September 1992.

Development and Application of an Acceptance Testing Model

Rex Pendley

December 2, 1992

Computer Sciences Corporation
System Sciences Division

AGENDA

- Definitions and Assumptions
- Systems Tested
- Construction of the Model
- Application to ISTP Testing
- Conclusions

Computer Sciences Corporation
System Sciences Division

DEFINITIONS . . .

- Mission Requirements–what the Flight Dynamics Division must do to support a mission
- Software Requirements–the subset of mission requirements met by software
- Acceptance Testing–verification that a system meets mission requirements
- Test Plan–a series of test cases reflecting operational scenarios, each verifying a subset of the requirements
- Test Item (TI)–a single requirement verified in a test

. . . and ASSUMPTIONS

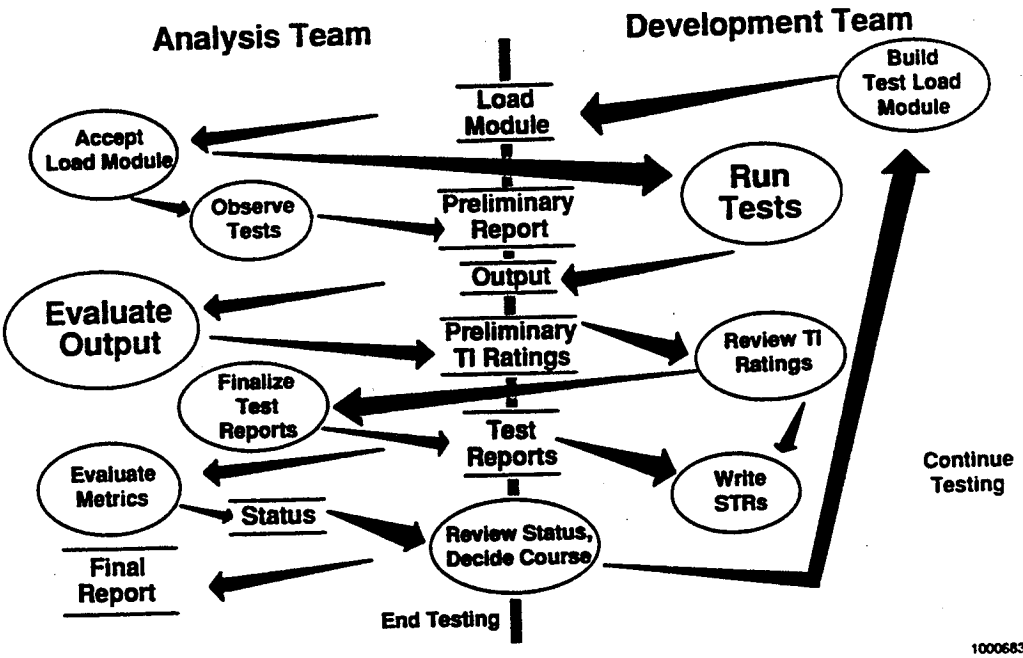
- Effort is divided between an analysis team and a development team
- System is acceptance tested only at the last build

SYSTEMS TESTED

Attitude Ground Support Systems (AGSSs) consist of an attitude determination system and three to nine utilities.

SATELLITE	AGSS SIZE (KSLOC)
COBE	178.7
GRO	266.9
GOES	128.8
UARS	319.5
EUVE	268.5

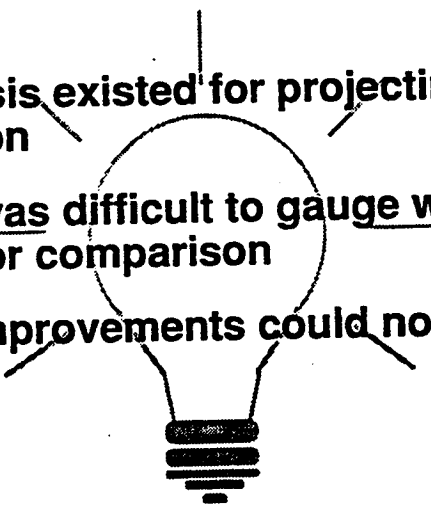
THE ACCEPTANCE TESTING PROCESS



10006830-PRES 5

Computer Sciences Corporation
System Sciences Division

Motivation for a TESTING MODEL

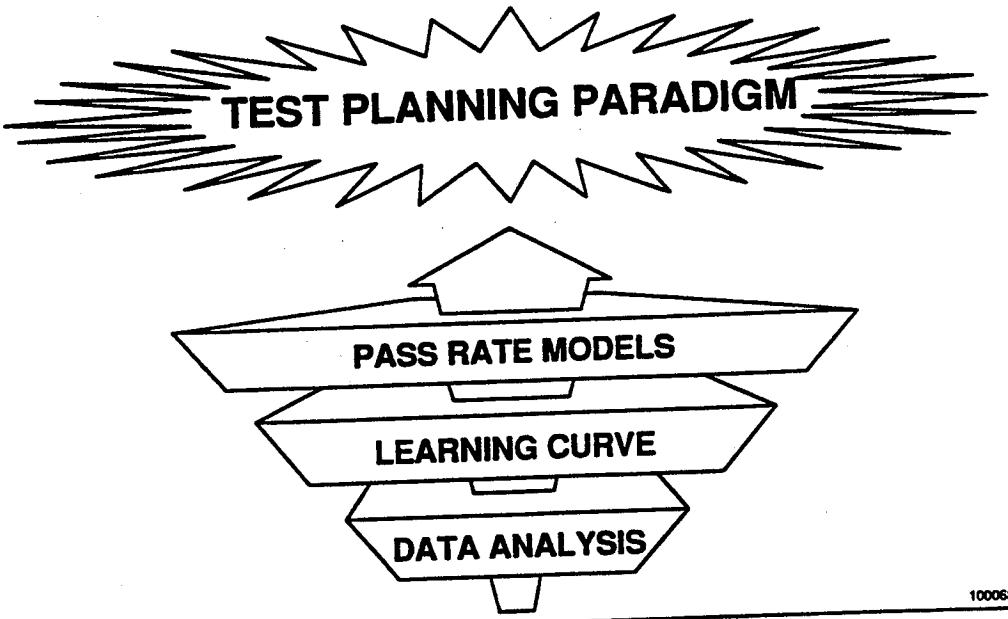


- No firm basis existed for projecting resources and duration
- Progress was difficult to gauge without a standard for comparison
- Effect of improvements could not be assessed

10006830-PRES 6

Computer Sciences Corporation
System Sciences Division

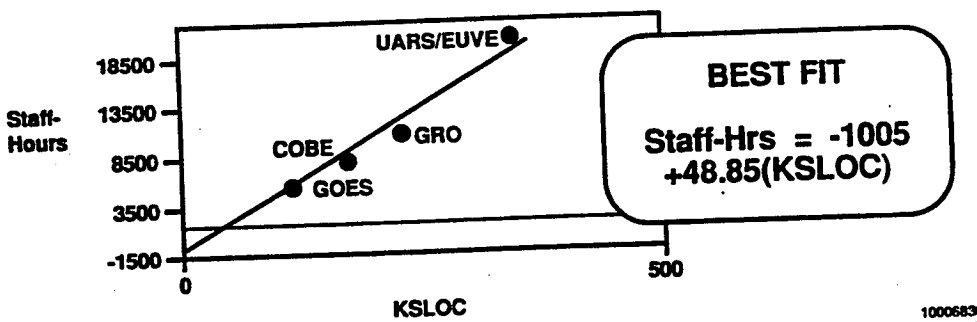
ELEMENTS OF THE MODEL



10006830-PRES 7

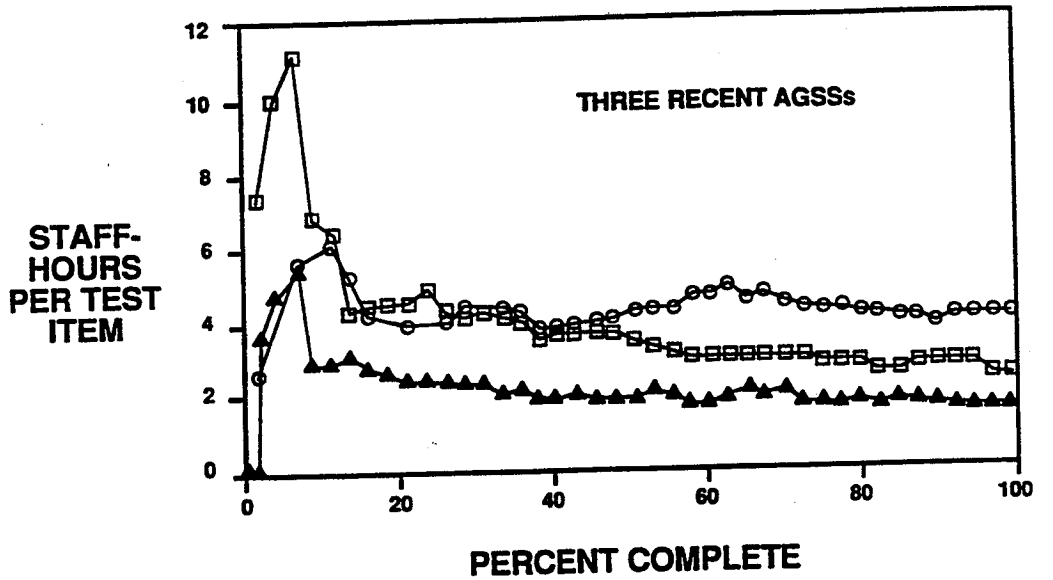
SUMMARY OF AGSS DATA

AGSS	Staff-Hrs	TIs	HR/TI	KSLOC	Hr/KSLOC	TI/KSLOC
COBE	7124	5677	1.3	179	40	32
GRO	8771	3626	2.4	227	39	16
GOES	5004	2816	1.8	129	39	22
UARS/EUVE	18884	6054	3.1	391	48	16



10006830-PRES 8

LEARNING CURVES

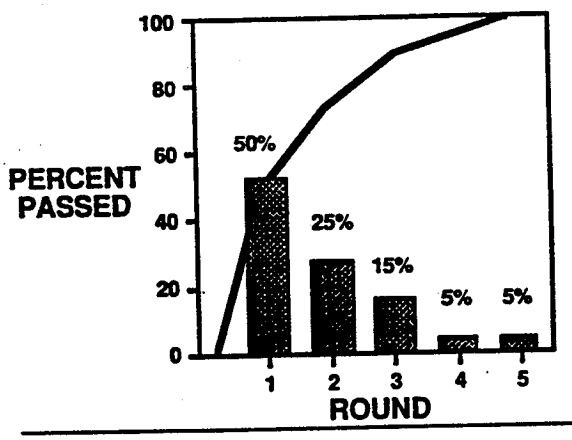


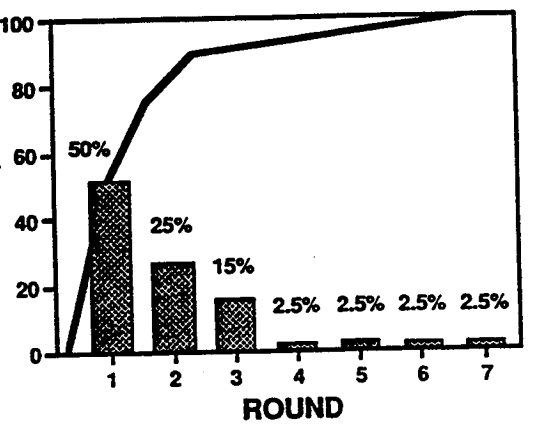
10006830-PRES 9

Computer Sciences Corporation
System Sciences Division

PASS RATE MODELS

SEVEN LOAD MODULE (7LM) MODEL





FIVE LOAD MODULE (5LM) MODEL

10006830-PRES 10

Computer Sciences Corporation
System Sciences Division

TEST PLANNING PARADIGM (1 of 2)

1. Size the total effort via the best fit expression:

   Staff-hours = -1005 + 48.85 x KSLOC
   (KSLOC = 1000 source lines of code)

   Convert hours to staff-months.

2. For a fixed team size, divide staff-months by team size to determine duration.

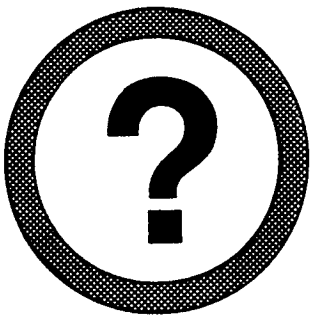
   For a fixed duration, divide staff-months by duration to determine team size.

TEST PLANNING PARADIGM (2 of 2)

3. Based on anticipated software availability, the learning curve, and the average test turnaround time of 6 days, construct a schedule based on available test capacity in terms of TIs reported per week.

4. Use the schedule to plot the number of TIs to be reported each week.

5. Use the pass rate model to estimate and plot the number of TIs expected to pass each week.



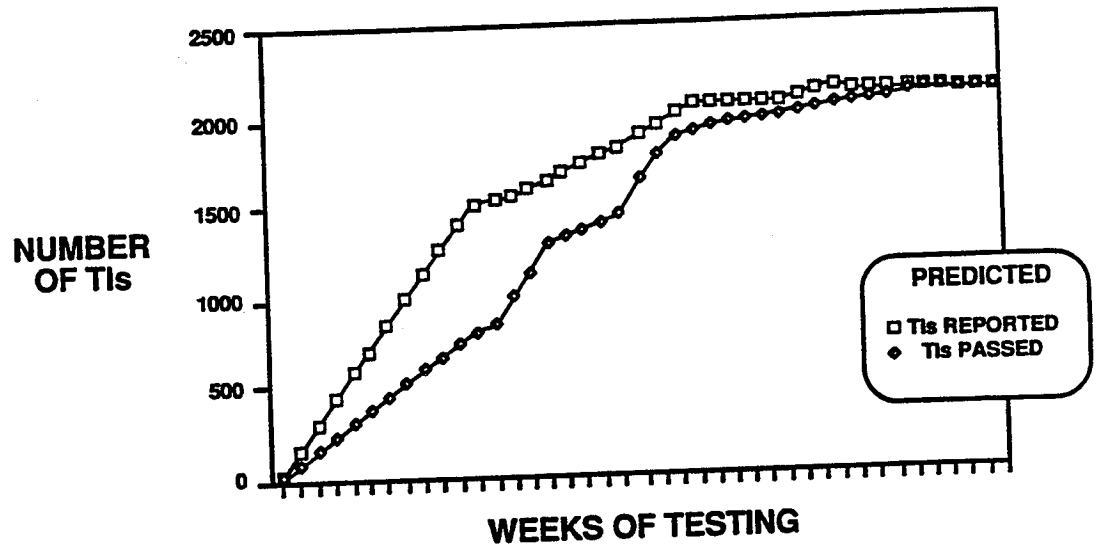
How well did the model work applied to the

ISTP AGSS?

ISTP AGSS TOTAL RESOURCE PREDICTION

- 174 KSLOC
- Substantially new development
- 2117 test items
- Model predicts 7495 staff-hours
- Actual expenditure was 7852 staff-hours, difference of 5%

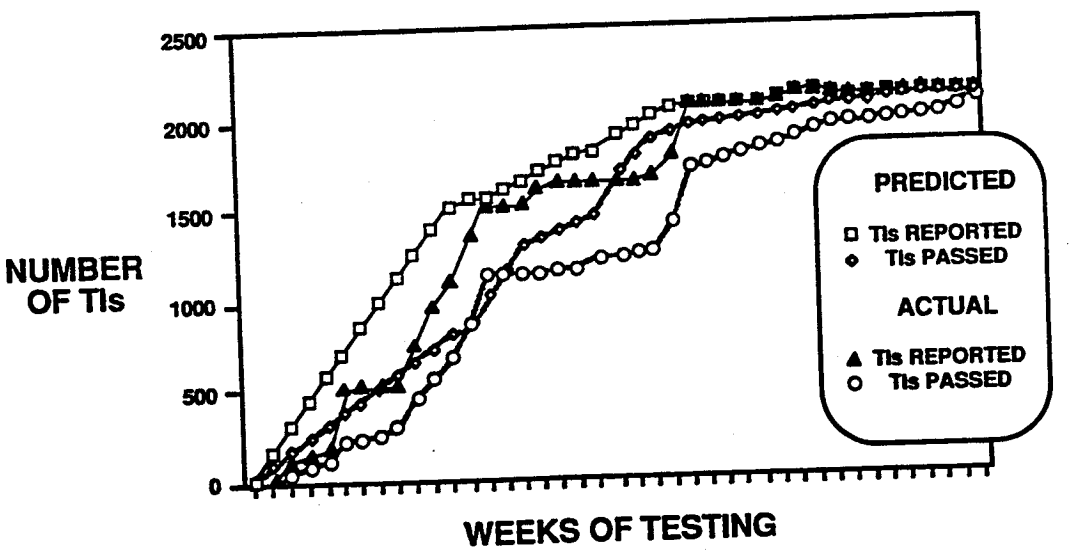
ISTP AGSS PREDICTED PERFORMANCE CURVES



10006830-PRES 15

Computer Sciences Corporation
System Sciences Division

ISTP AGSS ACTUAL PERFORMANCE CURVES



10006830-PRES 16

Computer Sciences Corporation
System Sciences Division

CONCLUSIONS

- Reliable model for predicting overall resources (staff-hours)
- Credible gauge for judging reporting rate and pass rate
- Metric should show effects of process improvement
- Derived from a "knowledge factory"
- Currently being adapted to massive re-use systems

CONTRIBUTORS

- Caroline H. Noonan
- Kenneth R. Hall
- Ted Z. Willard
- The Flight Dynamics Acceptance Test Process Group