LA-13716-C
Conference

Approved for public release;
distribution is unlimited.

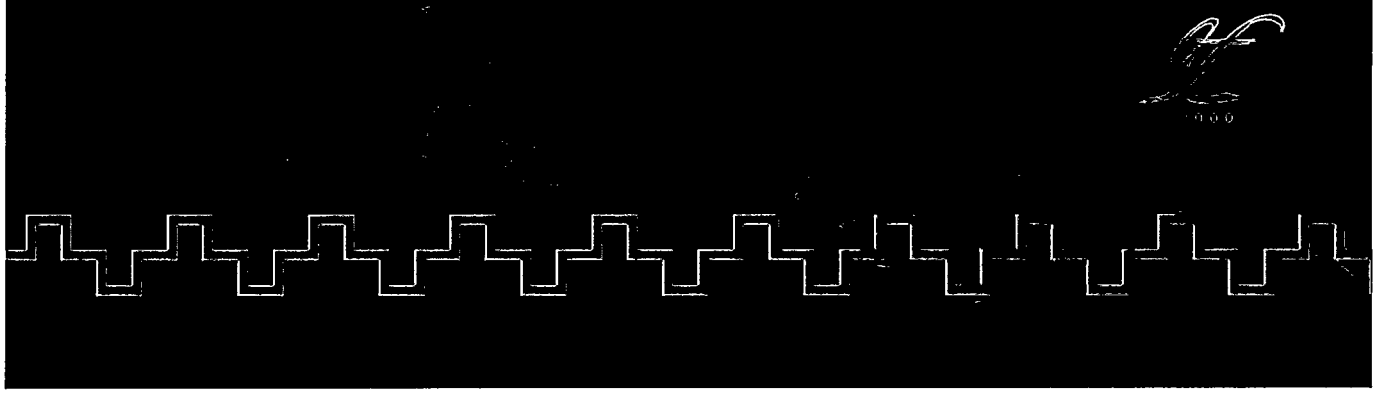
Proceedings of the Fifth Triennial Software

Quality Forum 2000

"Software for the Next Millennium"

Software Quality Forum

Eldorado Hotel, Santa Fe, NM
April 18 & 19, 2000



Los Alamos
NATIONAL LABORATORY

Los Alamos National Laboratory is operated by the University of California
for the United States Department of Energy under contract W-7405-ENG-36.

ABSTRACT:
The Software Quality Forum is a triennial conference held by the Software Quality Assurance Subcommittee for the Department of Energy's Quality Managers. The forum centers on key issues, information, and technology important in software development for the Nuclear Weapons Complex. This year it will be opened up to include local information technology companies and software vendors presenting their solutions, ideas, and lessons learned. The Software Quality Forum 2000 will take on a more hands-on, instructional tone than those previously held. There will be an emphasis on providing information, tools, and resources to assist developers in their goal of producing next generation software.

An Affirmative Action/Equal Opportunity Employer

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither The Regents of the University of California, the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by The Regents of the University of California, the United States Government, or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of The Regents of the University of California, the United States Government, or any agency thereof. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

LA-13716-C
Conference

Issued: April 2000

Proceedings of the Fifth Triennial Software Quality Forum 2000
Software for the Next Millennium

Software Quality Forum

Eldorado Hotel, Santa Fe, NM

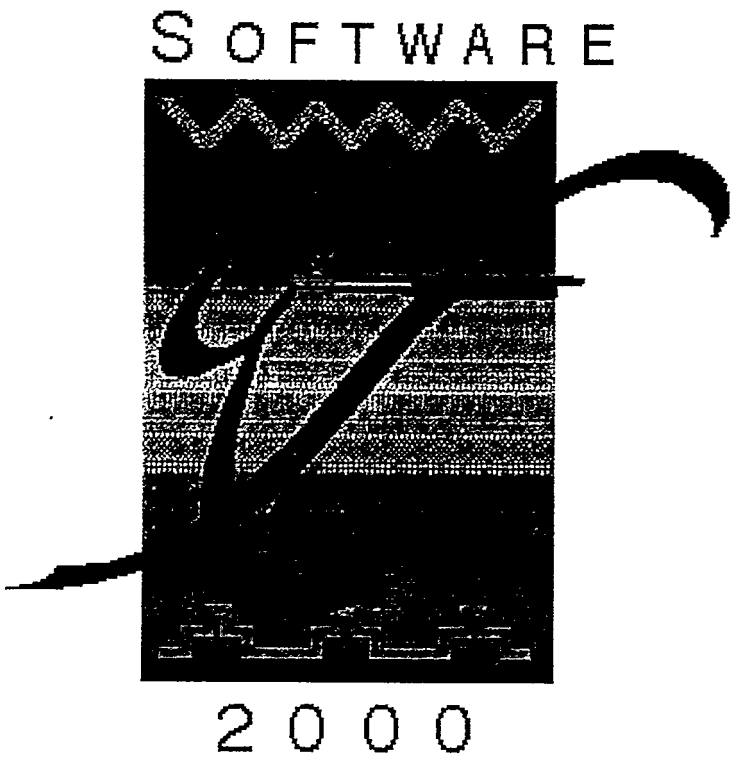
April 18 & 19, 2000

Cosponsored by
U.S. Department of Energy
Nuclear Weapons Complex Quality Managers
Software Quality Assurance Subcommittee

Hosted by
Scientific Software Engineering Group, CIC-12
Los Alamos National Laboratory

Los Alamos
NATIONAL LABORATORY

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36.



General Information

- Forum Committee and Track Chairs
- History of the Software Quality Forum
- Software Quality Assurance Subcommittee
- Forum Awards
- Forum Proceedings
- Banquet – Eldorado Hotel

Forum Committee

General Chair:

Maysa Peterson, Los Alamos National Laboratory, mpeterson@lanl.gov

Program Chair/Publicity Chair:

Jeanette LaGrange, Los Alamos National Laboratory, jvl@lanl.gov

Facilities Chair:

Barb Addessio, Los Alamos National Laboratory

Database Design:

Gerald Reisz, Los Alamos National Laboratory

SQAS Representative:

Kathy Burris, Los Alamos National Laboratory

Forum Assistant:

Gary Clark, Los Alamos National Laboratory

Track Chairs

Track 1 – Testing Architectures and Strategies:

Brenda Coblentz, DOE Headquarters
Nancy Storch, Lawrence Livermore National Laboratory

Track 2 – Project Management and Team Building Strategies:

Cathy Kuhn, AlliedSignal Federal Manufacturing and Technologies, Kansas City Plant

Track 3 – Configuration Management and Defect Tracking:

Kathleen Canal, DOE Headquarters

Track 4 – Software Design Architectures, Methodologies, and Requirements:

Kathy Burris, Los Alamos National Laboratory

Track 5 – Software Verification and Validation:

Dennis Adams, Idaho National Engineering and Environmental Laboratory
Nancy Storch, Lawrence Livermore National Laboratory

Track 6 – Software Management Tools:

Carolyn Owens, Warren Persons, Debra Sparkman, Lawrence Livermore National Laboratory

Track 7 – Software Quality Assurance and Engineering:

Larry Rodin, Idaho National Engineering and Environmental Laboratory

Track 8 – Software Risk Management and Metrics:

Dave Peercy, Patty Trellue, Sandia National Laboratory

Special Thanks to the Following People

Nicholas Degidio
provided us with the server for the website

John Allen Hopkins
did all the beautiful artwork

Carolyn Kuranz and Anita Quintana
helping with mailings and paperwork

Davita Martinez
administrative assistance

Tina Skinner
web page development and SQ 2000 logo design

Gerrick Snider
financial advisor

Dwayne Knirk
presentation evaluations

Linda Milligan-Duran, Anita Sutton, and the Eldorado Hotel
for the beautiful facilities and great help

History of the Software Quality Forum

The Software Quality (SQ) Forum was established by the Software Quality Assurance Subcommittee as an opportunity for all those involved in implementing SQA programs to meet and share ideas and concerns. The SQ Forum is open to the public. Participation from managers, quality engineers, and software professionals provides an ideal environment for identifying and discussing the many issues and concerns raised by the Forum attendees and speakers. The interaction provided by the Forum contributes to the realization of a shared goal—high-quality software product.

Topics presented at the SQ Forum generally include: testing, software measurement, software surety, software reliability, SQA practices, assessments, software process improvement, certification and licensing of software professionals, CASE tools, software project management, inspections, and management's role in ensuring SQA.

The Software Quality Forum is held every three years; past Forums are identified below.

Date	Site
Spring 1988	Sandia National Laboratories
Spring 1991	AlliedSignal Aerospace Kansas City Division
Spring 1994	Lawrence Livermore National Laboratory
Spring 1997	Kirtland Airforce Base

Software Quality Assurance Subcommittee

The Software Quality Assurance Subcommittee (SQAS) serves as a technical advisory group on software engineering and quality initiatives and issues for the Department of Energy's quality managers. The quality manager at each DOE site has the opportunity to select one primary and one alternate representative to the SQAS.

The Subcommittee grew out of a Software Quality Assurance Information Exchange Forum, which was held in March of 1988 at Sandia National Laboratories. The Subcommittee provides a continuing forum for the exchange of information and work issues in the area of software quality engineering.

For additional information about the SQAS, visit our web site at:

http://cio.doe.gov/sqas/index.htm

Forum Awards

The Forum Program Committee would like to recognize those presenters who, through their tutorial or technical presentation, have made a significant contribution to the success of the Forum. A Best Presentation award will be presented at the Forum Wrap-up session on Thursday, April 19.

Forum Proceedings

The Forum Proceedings will include abstracts and presentation materials for all technical presentations, presenter biographies, and final Forum program information. The Forum Proceedings will be distributed at the Forum with the registration packets.

Banquet—Eldorado Hotel

A banquet has been planned for Tuesday Evening at the Eldorado Hotel in the Anasazi Ballroom. There will be a variety of dinner selections offered that should accommodate all tastes. The cost of the banquet is $50.

Track List

Tuesday, April 18th

Track 1 – Testing Architectures and Strategies

Track 2 – Project Management and Team Building Strategies

Track 3 – Configuration Management and Defect Tracking

Track 4 – Design Architectures, Methodologies, and Requirements

Wednesday, April 19th

Track 5 – Verification and Validation

Track 6 – Management Tools

Track 7 – Quality Assurance and Engineering

Track 8 – Risk Management and Metrics

Monday, April 17, 2000

Check in time for hotel is 4:00 p.m.

8:00 a.m.	CIC-2 and Hotel ***Setup***			Forum Staff ***Setup***
9:00 a.m.				
10:00 a.m.				
11:00 a.m.		Registration – Forum and SQAS		
12:00 p.m.			Exhibitor ***Setup***	
1:00 p.m.		SQAS Meeting – De Vargas		
2:00 p.m.				
3:00 p.m.				
4:00 p.m.				
5:00 p.m.				
6:00 p.m.		Registration – Forum	Exhibitors Available	
7:00 p.m.				
8:00 p.m.				
9:00 p.m.				
10:00 p.m.				
11:00 p.m.				

Tuesday, April 18, 2000

7:00 – 8:30 a.m.	Registration/Buffet Breakfast – Anasazi Ballroom Vendors will be available.			
8:30 – 9:00 a.m.	Welcome Address – Anasazi Ballroom			
9:00 – 10:15 a.m.	Keynote Address – Anasazi Ballroom – **John Gilligan,** Chief Information Officer of the DOE			
10:15 – 10:30 a.m.	Break – Drinks will be provided in the Anasazi Ballroom.			
10:30 a.m. – 12:15 p.m. **Session 1**	**TRACK 1** → **Gary Mogyorodi:** Requirements-Based Testing: An Overview	**TRACK 2** → **Greg Schottland:** Object-Oriented Project Management (1 h) **H. W. Egdorf and Laurie Waters:** Process Development Experiences with the Accelerator Production of Tritium Project (45 min)	**TRACK 3** → **Susan Dart:** The Urgent Need for Configuration Management	**TRACK 4** → **Mike McKay:** Case Study: Effective Software Design Architecture of the UPS Project
12:15 – 2:15 p.m.	Lunch (on your own) – Vendors available 1:00–2:00			
2:15 – 4:00 p.m. **Session 2**	**Carolyn Owens:** Testing an Object-Oriented Controls System (45 min)	**Kathleen Byle:** Software Strategies Used for WIPP (1 h) **Rebecca Winston, R. E. Heard, and Brett Rassmussen:** Developing a Cohesive Team; Focused, Facile, and Friendly (45 min)	**Marjorie McCornack:** So I Believe CM Is a Good Idea. Now What?	**Patty Trellue:** Overview of the Software Engineering Institute's Capability Maturity Model for Software
4:00 – 4:15 p.m.	Break – Drinks will be provided in the Anasazi Ballroom			
4:15 – 6:00 p.m. **Session 3**	**Shawn Dawson:** Experiences Implementing Automated Regression Testing	**Panel:** Software Product Processes Web-Based Training—Overview Training	**Gary Froehlich and Harvey Ogden:** Managing Software in the Highly Regulatory Environment of the WIPP Project	**Walter Viali:** Front-End Loading with Facilitated Joint Application Development Techniques
7:00 p.m.	Banquet – Anasazi Ballroom Vendors will be available until 9:00 p.m.			

Wednesday, April 19, 2000

7:00 – 8:30 a.m.	Registration/Buffet Breakfast – Anasazi Ballroom Vendors will be available.			
8:30 – 8:45 a.m.	Announcements – Anasazi Ballroom			
8:45 – 10:00 a.m.	Keynote Address – Anasazi Ballroom – **Dr. Norm Brown,** Executive Director of the Software Program Managers Network			
10:00 – 10:15 a.m.	Break – Drinks will be provided in the Anasazi Ballroom.			
10:15 a.m. – 12:00 p.m. **Session 4**	**TRACK 5** → **Mike Williamson:** Verification and Validation of Modeling Codes	**TRACK 6** → **Jane Booker:** PREDICT, A New Approach to Product Development	**TRACK 7** → **Panel:** Perspectives on Application of Software Process Assessments to Scientific Codes	**TRACK 8** → **Bill Trebilcock:** Setting Up a Metrics Program for a Large Software Development Effort (45 min)
12:00 – 1:45 p.m.	Lunch (on your own) – Vendors available 1:00–2:00			
1:45 – 3:30 p.m. **Session 5**	**Tim Trucano:** Aspects of ASCI Code Verification and Validation	**Evi Dube:** Project Planning for an ASCI Simulation System	**Rick Brenner and Esther Derby:** Anticipating the Effects of Change	**Ruth Stapko:** Using Object-Oriented Metrics to Support Reliability: A General Discussion
3:30 – 3:45 p.m.	Break – Drinks will be provided in the Anasazi Ballroom			
3:45 – 5:30 p.m. **Session 6**	**Ann Hodges:** A Pragmatic Approach to Verification and Validation	**Cathy Kuhn and Don Rathbun:** Project Management Automation	**Michael Tiemann:** DOE Information Architecture and Software Quality Assurance	**Julie Bouchard:** The Adversaries' View (45 min) **Joe Schofield:** Practical Metrics for CMM KPA's (45 min)
5:45 p.m.	Wrap-Up Awards			
7:00 p.m.	Vendor Teardown			

Thursday, April 20, 2000

7:00 a.m.	Continental Breakfast Anasazi Ballroom	Checkout for hotel is 11:30 a.m.
8:00 a.m.	SQAS Meeting – De Vargas	
9:00 a.m.		
10:00 a.m.		
11:00 a.m.		
12:00 p.m.	SQAS Lunch	
1:00 p.m.	SQAS Meeting – De Vargas	
2:00 p.m.		
3:00 p.m.		
4:00 p.m.		
5:00 p.m.		
6:00 p.m.		

The Adversaries' View

Julie F. Bouchard

Track 8: Software Risk Management and Metrics

Wednesday, April 19, Session 6: 3:45-5:30 pm, Zia A

The Sophisticated Adversary (e.g., Cyber Terrorist) is a serious threat to modern information systems, especially those systems trusted to control the nation's critical defense and infrastructure systems. Recent stories in the press indicate that the sophisticated adversary exists. Red Teaming is an advanced form of information surety assessment. This approach is based on the premise that an analyst who attempts to model an adversary can find systemic vulnerabilities in an information system that would otherwise go undetected. Ideally, red teaming is applied during design so that risk mitigation options can be explored prior to system deployment. Sometimes red teaming is applied after deployment, commonly prior to a significant upgrade or after detection of hacking attempts. Sandia's IDART (Information Design Assurance Red Team) has conducted, and/or is currently involved in, assessments for DOE, DOD, DOI, the Treasury Department and industry. These assessments include

- Detailed surety assessments of a system design (prior to implementation)
- Detailed surety assessment of the system design and implementation (after implementation).
- Experiments, exercises, or demonstrations on implemented or prototype systems
- Assessment followed by mitigation techniques—recommend improvements in security design

This presentation will discuss the threats and adversaries that IDART models (all based on unclassified information available to the public). The *sophisticated adversary model* will be presented (and contrasted to other adversary models):

- Adversary Goals and Objectives—this adversary has specific goals for attacking a system.
- Funding—this adversary is well funded and can afford to buy necessary technology and inside information.
- Risk Aversion—this adversary will make every effort to avoid detection.
- Other Behavioral Expectations—this adversary is creative, clever, and unconventional.

IDART assessment and reporting methodologies will be presented. The emphasis will be on the views an assessment takes of an information system. These views include:

- Functional—but instead of "How does it work?" the question is "How can the functionality be broken?"
- Physical and Spatial—where are system components located and how does that make them vulnerable?
- Temporal—the sequences of events are explored to attempt to identify times when vulnerabilities are introduced.
- Lifecycle—the system lifecycle (including design, build, enhancement, modification, and upgrade) is analyzed for lifecycle vulnerabilities.
- Consequences—a consequence-based view is used to determine which vulnerabilities would be of most benefit to an adversary.

In conclusion, actual red team experiences will be shared, with the names omitted to protect our customers.

Anticipating the Effects of Change
Esther Derby and Rick Brenner
Track 7: Software Quality Assurance and Engineering
Wednesday, April 19, Session 5: 1:45-3:30 pm, Devargas

Planning and implementing organizational change and software process improvement requires an array of widely recognized skills that include planning, design, communication, change management and interpersonal relations.

Planned changes affect both the *people* in the organization and the organizational *system* itself. A change effort that builds upon intentional improvement of the organizational change process has a better chance of success than one that relies on a mandated or directed change process.

This presentation covers methods to model system impacts, to understand how change affects people in organizations, and to develop more effective change processes.

Satir Change Model	The Satir change model illustrates the predictable stages that people and systems experience when they experience a change. Knowledge of this model helps managers navigate the turbulence that sometimes comes with implementing planned changes.
Modeling the Effects of Change	Using a modeling technique called a diagram of effects, you can study the effects of changes before implementing them.
Changing the Change Process	By creating a shared view of how organizational change works, based on the Satir Change Model, you can improve your organization's change process, and increase the success of organizational change efforts.

You will probably recognize the "symptoms" of change: understanding the signs and stages provides you with the knowledge to help the people in your organization get through turbulent periods and achieve the desired change outcomes.

Aspects of ASCI Code Verification and Validation

Timothy G. Trucano and Jaime L. Moya

Track 5: Software Verification and Validation

Wednesday, April 19, Session 5: 1:45-3:30 pm, Zia C

The DOE Stockpile Stewardship Program (SSP) calls for higher consequence computational modeling to support management of the U. S. nuclear stockpile in the first decades of the 21st century. The Accelerated Strategic Computing Initiative (ASCI) has specific responsibilities for developing and implementing scientific software (called "codes") on a new generation of massively parallel computers having unprecedented capabilities in order to achieve stockpile computational modeling requirements. A critical element, perhaps the most critical element, of computational modeling as a technology component in the SSP is confidence in the computational results. The needed confidence in the results of elaborate computational models will, in fact, have to transcend the purely technical aspects of the modeling. The required confidence will intertwine with a complex risk and an uncertainty-driven decision process that has traditionally had far less contribution from computational science. How best to achieve a sufficient level of confidence to enable high-consequence decision making for SSP with attendant risk and uncertainty is a major challenge for DOE computational science. Beginning in Fiscal Year 1998, a new DOE program—the ASCI Verification and Validation Program—began to guide the verification and validation of the emerging key DOE computational codes that are critical to the SSP. The stated vision of this program is to "establish confidence in the simulations supporting the Stockpile Stewardship Program through systematic demonstration and documentation of the predictive capability of the codes and their underlying models." In this talk we will discuss three different perspectives on the application of rigorous approaches for verification and validation to scientific codes to achieve the levels of confidence that are required for the DOE SSP. First, we will summarize the assumptions and goals of the overall ASCI Verification and Validation (V&V) program. Second, we will discuss in greater detail the implementation of the ASCI V&V program at Sandia National Laboratories. Third, we will illustrate the intersection of certain features of the Sandia V&V program with a particular ASCI code being developed at Sandia, the ALEGRA shock wave physics code. The scientific codes that are being developed by Los Alamos, Lawrence Livermore, and Sandia under the ASCI program are very complex. We believe that effective strategies for pursuing verification and validation (V&V) of these codes will require a mix of past experience and novel concepts. In this talk, we will emphasize several components of V&V that we believe resonate with new opportunities provided by the ASCI program. For example, we will discuss how the V&V program is defining and emphasizing the role of software quality engineering (SQE) as a verification technology. we will also discuss problems and opportunities associated with the development and execution of test suites for V&V. Finally, we will present our views on the role of "Uncertainty Quantification" or "Uncertainty Estimation" in scientific code validation, as well as in high consequence applications of such codes. Our illustrations of the challenges associated with advanced approaches to scientific code V&V will be centered on ALEGRA, but probably characterize issues that influence much of the software being developed by ASCI. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin

Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Case Study: Effective Software Design Architecture of the UPS Project
Mike McKay
Track 4: Software Design Architectures, Methodologies, and Requirements
Tuesday, April 18, Session 1: 10:30 am - 12:15 pm, Zia C

The Unified Parallel Software project (UPS) is tasked with building a software library that will be used by Accelerated Strategic Computing Initiative (ASCI) codes at Los Alamos National Laboratory (LANL). The discussion will cover the Software Design Architecture of the UPS Project. The presentation will be divided into two areas: user and developer perspectives. Users have certain requirements that must be satisfied before they are willing to program with any software library. The library must serve a purpose, be easy to use, be replaceable, and be portable to different computer architectures. Discussed will be how the UPS project satisfies those requirements.
For those actually working on the UPS team, there are requirements as well. Programmers must operate within an environment that promotes good software practices. The project must be designed to allow for automation of tasks, modification of components, and ease of understanding. Discussed will be how the UPS software design contributes to a successful project.

COSMOS, A Software Project Estimation Tool
Michael Bell
Track 8: Software Risk Management and Metrics
Wednesday, April 19, Session 4: 10:15 am - 12:00 pm, Zia A

COSMOS, a PC-based software estimation tool, is discussed. Also, two industry-standard software cost estimation approaches are described, characterized, and compared: Function Point Analysis and COCOMO. The COSMOS tool combines function point analysis, the COCOMO model, and Putnam's adaptation of the Rayleigh curve to produce project estimates. Use of the tool illustrates how function point counting can be effectively employed in parametric cost estimation. COSMOS uses the backfire technique to produce a source lines-of-code (SLOC) estimate from a function point count. The SLOC estimate is then used as input to COCOMO and Rayleigh models to produce estimates of effort, schedule, and staffing. Since function points can be accurately counted early in the software development cycle, COSMOS' use of function points as input allows for earlier, more accurate estimates than tools that require SLOC as the primary input, such as REVIC. COSMOS was produced as a laboratory project by graduate students in East Tennessee State University's Software Engineering program.
How does it relate to the conference theme?
Two of the most important activities of software project management are task planning and budgeting. Both require effort and schedule estimates. Various parametric estimation models have been proposed that can be brought to bear from the limited

number of broad project characteristics that are known (or easily guessed) from the outset. There is a problem: The most important effort- or schedule-determining characteristic is project size. Most parametric cost models express size in source lines of code (SLOC), which is next to impossible to determine in advance of coding. On the other hand, function point counts can be determined well before detailed design, early enough to be of use in advance planning and budgeting. Function point analysis has been utilized for development project estimation, based on the notion of an average "function points per month" productivity rate. But this approach does not take into account several relevant cost drivers that are reflected in the COCOMO model. The backfire technique can be used to convert function point counts to SLOC counts. This allows for utilization of a number of parametric models, and for enjoying their chief benefit—the fact that they have been calibrated with measurements from a number of software projects. The use of such models can serve as primary sources of effort and schedule estimates or as independent validation of task-based estimates. The COSMOS application illustrates the use of function points in this manner.

COSMOS supports three separate industry models (function point, COCOMO, and Rayleigh), a project model that combines the three, and a system model that combines projects. COSMOS also supports a variety of "what if" scenarios, and incorporates a number of sensitivity analysis and reverse engineering options.

More information:

COSMOS is freely available to the software engineering community. COSMOS may be downloaded from the COSMOS web site: .

http://www-cs.etsu.edu/softeng/cosmos/index.html.

Additional information on COSMOS, including a comprehensive technical reference, is available there as well.

DOE Information Architecture and Software Quality Assurance

Mike Tiemann

Track 7: Software Quality Assurance and Engineering

Wednesday, April 19, Session 6: 3:45-5:30 pm, Devargas

The presentation will focus on DOE's information architecture program and its relationship to software quality assurance initiatives. The purpose is to discuss and highlight the way IT architectures can impact and facilitate software quality assurance activities.

Part I.

The first part of the presentation is designed to inform the participants about DOE's information architecture program. This part will cover three subjects:

- A brief history of IA activities (prior to 1999)
- The DOE-wide project to define a "corporate" IA
- The approach for implementing IA throughout DOE (Headquarters and Field components)

The IA history component will serve to introduce the participants to DOE's overall approach to IA, as articulated in various documents such as "Information Architecture Volume IV – Vision," published in March 1998. It will also describe efforts by DOE

programs—notably the Office of Science—to establish effective IA programs. The pilot IA project (completed by the Office of Science in July 1997) and subsequent implementation actions will be described.

The "corporate" IA project component will describe the conduct and results of the DOE Information Architecture Project (or DIAP). DIAP's origin, scope, and objectives will be presented and its relationship to previous DOE architecture activities will be highlighted. Some time will be spent on describing how DIAP was conducted to ensure that participants gain a clear understanding of the practical, logical steps that were taken to produce the DOE "corporate" architecture and what specific products were produced. The steps of the IA planning methodology (Enterprise Architecture Planning by EAI, Inc.) that were used will be described as will the specific IA products that resulted from DIAP (IM principles, business model, data architecture, application architecture, and technology architecture). The actions necessary to transition DOE corporate IT system and technology projects will also be covered. The important role played by DOE business representatives in defining the "corporate" IT architecture will be stressed.

The implementation approach component will inform the participants about DOE's plans for rolling out the architecture approach and results to DOE headquarters and field organizations. This component will describe how DOE plans to deepen and substantiate the "corporate" architecture by conducting follow-on IA projects. The component will also outline the activities such as an application development methodology and a data management program needed to strengthen DOE's IT processes to support an architected environment.

Part II.

The second part of the presentation is designed to explore the relationship between architecture and software quality assurance. The potential relationship between the major architecture products and SQA initiatives will be identified. The following represent some of the areas that will be covered:

- The use of an IM principle on "Quality" as a basis for a comprehensive IT quality program to include SQA
- The utility of a business model to help define the business functions to be handled by an application (requirements definition)
- The utility of a data architecture as a framework for a corporate data management program to include a corporate data dictionary (data standardization)
- The employment of an applications architecture to structure application requirements (requirements definition)
- The use of a technology architecture to help define IT standards

Specific examples of these types of relationships will be used whenever possible. The examples will be taken from DOE IT projects or activities that exemplify or clarify the relationship.

Process Development Experiences with the Accelerator Production of Tritium Project

H. W. Egdorf and Dr. Laurie S. Waters

Track 2: Project Management and Team Building Strategies

Tuesday, April 18, Session 1: 10:30 am-12:15 pm, Devargas

The Accelerator Production of Tritium (APT) project is designing a particle accelerator and target to produce tritium using neutrons generated from a Tungsten spallation source. The project has a variety of participants with various levels of software engineering experience and maturity. Different software efforts are being pursued with different processes and degrees of formality. The resulting products are required to be of high validity and quality.

The APT project is divided into three project offices responsible (roughly) for operation of the final facility (Operations Project Office, OPO), construction of the final facility (Plant Project Office, PPO), and research and development of the facility (Technology Project Office, TPO). The OPO has a fairly mature set of documented processes for software development designed for local operation at the sponsoring organization. The PPO has responsibility for quality assurance and configuration management of the engineering products, but with little process support for research and development activities. The TPO has the least software development formality, but the richest responsibility and capability for the research required for the advanced technology needed by the facility. The overall engineering quality and maturity of the project is excellent, but software processes used for the design and research software required by the project (particularly the TPO) lag considerably behind the other engineering disciplines.

The efforts underway to increase the level of software validity for the design of the APT facility are concentrated in two areas. The Software Engineering Institute Capability Maturity Model (CMM) is the framework being used as a basis for these efforts.

At the top level, support for standard software processes is being developed at the project director's office (above OPO, PPO, and TPO). This will provide the management support needed as a basis for eventual operation with processes at a CMM level 3 across the project. A major challenge is to produce a management plan that can embrace and coordinate the very wide range of existing processes and formality (or lack thereof) across the different organizations that make up the various project offices.

At the lower levels, individual projects are being moved toward greater process maturity on a project-by-project basis. The goal of these efforts is to produce examples of the development of CMM level-2 processes. Efforts are being concentrated on a few projects, with special emphasis on a large design code development activity.

This paper describes experiences with this two-pronged approach towards software project maturity. The MCNPX code project, which is developing an enhanced Monte-Carlo-based design tool for the APT accelerator, is used as an example for discussion of these experiences.

Experiences Implementing Automated Regression Testing

Shawn Dawson

Track 1: Testing Architectures and Strategies

Tuesday, April 18, Session 3: 4:15-6:00 pm, Zia A

Regression testing of parallel, portable programs presents challenges not inherent in the testing of sequential programs. These challenges arise primarily from the following factors:

1. Parallel programs must be written in a portable manner to be run on multiple machines. Reasons for this are
   - The parallel machine on which the code will be run may not be the best parallel machine on which to develop the code.
   - Programs typically outlive a specific parallel machine. This allows programs to be ported to new parallel machines when they become available.
2. The method of starting a parallel program varies across parallel computers. That is, the method of requesting multiple processors and starting a parallel program is not standard.
3. There are multiple parallel options when running the program. This results in a myriad of ways to run a single problem.
   - Sequential or parallel
   - Number of processors on which to run—1, 8, 1000, ...
   - Method of decomposing the problem
   - Manner of obtaining parallelism—MPI, Threads, or both

Therefore, when running a single regression test, it isn't sufficient to say the test succeeded. What can be said is that the test successfully ran on 8 processors, with the problem decomposed into 16 blocks, using MPI, on an IBM-SP2 parallel computer. This is no guarantee that the same would run successfully on a DEC SMP computer, or with threads, or on 16 processors. Each option change results in a different test.

Given the large number of tests required to perform a regression test suite, this procedure must be automated. In the past, testing has been a manual process, with the result that testing wasn't performed as often as desired.

I implemented an automated, portable method of performing the large number of tests necessary to perform a thorough regression test across multiple platforms. This method uses Perl and other portable graphic and file comparison tools. The results of the tests are summarized for the user in simple reports. The details of regression tests' failures are presented in detailed reports and in postscript graphic charts.

My presentation will detail the challenges encountered, the successes we had, and the outstanding obstacles yet to be overcome.

Front-End Loading with Facilitated Joint Application Development Techniques

Walter Viali, CQA, PMP

Track 4: Software Design Architectures, Methodologies, and Requirements

Tuesday, April 18, Session 3: 4:15-6:00 pm, Zia C

Projects continue to fail at an alarming rate and the industry continues to do very little to aggressively change this trend. The models, techniques and tools for improving the way business is done in I.T. are available to everyone, yet very little progress has been made in the past several years, with the exception of a few enlightened organizations. As the size of projects increases, so do the risk and cost factors. Huge cost overruns and questionable implementation approaches are encountered everyday. Yet, all the emphasis is on getting systems up and running faster and faster, cutting all conceivable corners and frustrating everyone involved, from the customer to the developers, to I.T. management, who are busy fighting fires from all directions. We need to take a serious look at the way we do business in I.T. and realize that we can no longer postpone getting our house in order. The most effective approach for the continuous improvement of internal practices is provided by the Software Engineering Institute's Capability Maturity Model (SEI CMM). The effective use of the SEI CMM and of benchmarking techniques can make the difference between a viable internal I.T. infrastructure for project delivery and the inevitable outsourcing of I.T. functions which are viewed as too expensive and of questionable value. We cannot, however, implement these methods effectively unless there is a program to deal with the lack of awareness of these methods at all levels of the I.T. organization. It is impossible to create commitment and dedication to improving our practices if the majority of the players have not gained an appreciation for the way it should be, as opposed to the way it really is! Many of these players, even at the highest levels of the I.T. hierarchy, lack the necessary knowledge to make more inspired decisions as to how an I.T. organization should really be managed. Finally, projects continue to fail because of poor front-end loading techniques. At a time when increased customer involvement in I.T. projects has become synonymous with greater probabilities of project success, we seem to drift into an ever more impersonal world of e-mail, phone mail, paged mail and electronic forums, which lead to more misunderstandings and lack of communication. Joint Application Development (JAD) is the most effective technique for dealing with project communication issues. Before the quality movement turned everyone into a problem-solving facilitator, JAD was used with enormous success in all aspects of project development, especially in the front-end loading phase of a new project. JAD has also been used to drive I.T. strategies from business strategies and achieve the Business/I.T. alignment, which has been so elusive for many companies. JAD is not only the catalyst for improving our front-end loading activities in a project, but it is also the best mechanism to satisfy a primary business need: People talking to people!

Managing Software in the Highly Regulatory Environment of the WIPP Project
Gary Froehlich and Harvey Ogden
Track 3: Configuration Management and Defect Tracking
Tuesday, April 18, Session 3: 4:15-6:00 pm, Zia B

The U.S. Department of Energy (DOE) Waste Isolation Pilot Plant (WIPP), located in southeast New Mexico, is a deep geologic repository for the permanent disposal of transuranic waste generated by DOE defense-related activities. Sandia National Laboratories (SNL), in its role as scientific advisor to the DOE, is responsible for evaluating the long-term performance of the WIPP. This risk-based performance assessment (PA) is accomplished in part through the use of numerous scientific modeling codes, which rely for some of their inputs on data gathered during characterization of the site. The PA is subject to formal requirements set forth in federal regulations. In particular, the components of the calculation fall under the configuration management and software quality assurance aegis of the American Society of Mechanical Engineers (ASME) Nuclear Quality Assurance (NQA) requirements. This paper describes SNL's implementation of the NQA requirements regarding software quality assurance (SQA) and configuration management. The description of SQA for a WIPP PA calculation addresses not only the interpretation of the NQA requirements, it also discusses roles, deliverables, and the resources necessary for effective implementation. Examples are given which illustrate the effectiveness of SNL's WIPP SQA program, followed by a detailed discussion of lessons learned. The complexity of the PA calculation is described, and the rationale for developing a flexible, robust run-control process is discussed. The run-control implementation is described, and its integration with the configuration-management system is then explained, to show how a calculation requiring 37,000 CPU-hours and involving 225,000 output files totaling 95 Gigabytes, was accomplished in 5 months by 2 individuals, with full traceability and reproducibility.

Object-Oriented Project Management
Greg Schottland
Track 2: Project Management and Team Building Strategies
Tuesday, April 18, Session 1: 10:30 am - 12:15 pm, Devargas

This talk presents the current approach to object-oriented project management using the Iterative (Unified) process and the Unified Modeling Language (UML). It compares the process to the more traditional structure analysis and design process. It also provides detailed, step-by-step guidance toward implementing the Iterative (Unified) process with the UML on real-world projects.

Overview of the Software Engineering Institute's Capability Maturity Model for Software
Patty Trellue
Track 4: Software Design Architectures, Methodologies, and Requirements
Tuesday, April 18, Session 2: 2:15-4:00 pm, Zia C

This presentation will give an overview of the Software Engineering Institute's Capability Maturity Model for Software (CMM-SW), Version 1.1. It will discuss the five levels of maturity and the key process areas (KPAs) that make up each level. Insight into what is required to fully satisfy a KPA will be provided. Finally, guidelines on how to map your organization's current software practices to the CMM/SW will be discussed.

Perspectives on Application of Software Process Assessments to Scientific Codes
Dave Peercy, Patty Trellue, Dennis Adams, Nancy Storch, Gerald Reisz, Evi Dube, Jaime Moya, Bill Reed
Track 7: Software Quality Assurance and Engineering
Wednesday, April 19, Session 4: 10:15am - 12:00 pm, Devargas

The software process assessment approach applied to the Tri-Lab Accelerated Strategic Computing Initiative (ASCI) code teams at SNL, LANL, and LLNL will be presented along with perspectives of the benefits and problems as viewed by persons who participated in the various roles. Each participant will provide a brief interpreted viewpoint after which the panel will provide some general recommendations and lessons learned. Audience questions and discussion will also be solicited. The various viewpoints of the survey roles include process methodology developer and assessors, site coordinator, code team participant, independent process observer, and management (DOE Headquarters and site-specific).

A Pragmatic Approach to V&V
Ann Hodges
Track 5: Software Verification and Validation
Wednesday, April 19, Session 6: 3:45-5:30 pm, Zia C

I will present my experiences and lessons learned in leading and developing V&V/test programs during the last 10 years for simulation and modeling and decision support systems. The presentation will focus on areas that are applicable to the current work I'm involved in for ASCI applications V&V and CPLANT production: infrastructure to support V&V (configuration and release management, processes, tools, guidelines/standards, metrics, training), V&V strategy (prioritization, tailoring, V&V performed in parallel with development, models of V&V, serial->parallel), pragmatic software engineering/reverse engineering to support V&V, V&V of code usage (problem simplifications, environment used, files, accredited use of code), answering the question "Does science stop when V&V starts?", V&V of COTS, and design for testability.

PREDICT, A New Approach to Product Development

Jane M. Booker, Thomas Bement, Mary Meyer, Sallie Keller-McNulty

Track 6: Software Management Tools

Wednesday, April 19, Session 4: 10:15 am-12:00 pm, Zia B

The 1999 R&D 100 award winning methodology called PREDICT (Performance and Reliability Evaluation with Diverse Information Combination and Tracking) will be presented. R&D 100 Awards are given annually to the top 100 technological innovations. PREDICT is statistically based methodology. It is a set of formal, structured techniques for estimating the performance and/or reliability of a product, when test data are scarce or unavailable. PREDICT accomplishes this task by documenting and exploiting the expert knowledge of a company's designers, engineers, and scientists and statistically combining expert knowledge and uncertainty with a wide variety of existing data while folding uncertainties about the product's expected performance into its calculations.

This approach has been applied to two application areas: 1) the lifetime prediction of an automotive system (from concept, where test data are absent, to in-use, where data are available) and 2) the estimation of the performance of a physics package of nuclear weapon systems undergoing aging in the stockpile. The interesting differences discovered in problem formulations and methodological implementations between these two settings will be discussed.

The steps involved in implementing the PREDICT methodology will be presented and illustrated with examples. The presentation will also illustrate how the methods developed in these two applications are applicable to other problems where test data are sparse or lacking in systems in a dynamic environment, such as for software reliability. Specifically, the formal use of expert judgment will be emphasized with an interactive example demonstrating some of the structured methods for eliciting or analyzing this information. Analysis issues such as handling uncertainties, quantification of qualitative information, and combining diverse sources of information and data are presented along with some practical solutions. Knowledge management of the diverse information sources and analysis is another important aspect of PREDICT, and this area will also be discussed and illustrated.

While this methodology has been successfully applied in these two applications, many analysis issues remain for continued research. These will also be presented, including an update on progress in certain areas such as automation of the methods. PREDICT was developed jointly by the nuclear weapons program at Los Alamos National Laboratory and Delphi Automotive Systems.

Project Management Automation
Cathy Kuhn and Don Rathbun
Track 6: Software Management Tools
Wednesday, April 19, Session 6: 3:45-5:30 pm, Zia B

Project management improvements, including processes, support tools, and the merging of the two have been a big focus at AlliedSignal Inc. Federal Manufacturing and Technologies. In 1999, two software applications were implemented to help automate the management of projects and the associated documentation. The purpose of this talk is to introduce the two applications and to provide the listener with an overview of their use. The hope is that the listener will come away with some ideas as to how they might improve their processes through the use of automated tools. The first application was developed to automate the service request process for our Information Systems organization. Developed as a Lotus Notes® application, it directs workflow, provides automatic mail notification, traceability of changes, as well as being a repository for project information. The second application, also a Lotus Notes® application, is the Project File Cabinet. The objective of this application was to improve the engineering project management process by providing an on-line shareable project management system. The system needed to be flexible (provide minimum required features to a project file, but allow a project team to customize their project file) and scaleable (suitable for small to very large projects) based upon end-user project management needs. Funding for the work was provided through the ADAPT Program. Both applications achieved their goals of easily capturing information while providing security features to control access to those who have a "need to know" on individual "unclassified" projects. The presentation approach will be to provide PowerPoint slides of information related to the project development on each application and a "live" demonstration of each application. The "live" demonstration will promote a "hands-on" environment, show actual projects that are managed in the applications, and how to set-up new requests or projects in each.

Project Planning for an ASCI simulation system
Evi Dube
Track 6: Software Management Tools
Wednesday, April 19, Session 5: 1:45-3:30 pm, Zia B

What does it take to control the continually changing environment of an ASCI simulation project? See what tools, strategies and processes an ASCI code team is using to handle the challenges of managing this large code system. Hear about our successful strategies, and our not-so-successful attempts that we have used over the past 5 years.

Requirements-Based Testing: An Overview
Gary Mogyodrodi
Track 1: Testing Architectures and Strategies
Tuesday, April 18, Session 1: 10:30 am-12:15 pm, Zia A

In many organizations, testing is viewed as slowing down the delivery of the system, partly because it is the last step in the development process. Ironically, when testing is properly deployed, with heavy emphasis on requirements-based testing (RBT), it can have a major impact on overall project productivity as well as product quality. Many organizations also have discovered that capture or playback tools require a significant investment in building and maintaining test scripts. They also discover that the scripts cannot keep up with rapidly changing specifications. This presentation will address how an RBT process provides major productivity gains, especially when used in conjunction with a tool to support it. The RBT process stabilizes the completeness of the specifications early in the development process. An RBT tool then designs an optimized set of test cases that are then automatically fed into all of the major capture/playback tools. The results are fewer tests with greater functional coverage, shortened time to delivery, reduced costs of development, and significantly improved quality.
Presentation deliverables are as follows:

- How to manage the effects that quality programs will have on the overall planning and management of software development.
- Understanding the problems encountered in software development, opportunities and objectives for improving the process.
- Knowing the importance of good functional specifications.
- The 12-point RBT process.
- Techniques for writing testable requirements.
- Ambiguity review process for functional specifications.
- A rigorous approach to testing, contrasting and comparing it to other testing techniques used in industry.
- How to use quantitative completion criteria for organizing a testing effort.
- A survey of the categories of automated tools for supporting testing techniques, including Requirements-Based Testing and the relationship between Caliber-RBT and other tools.
- The important issues of implementing the quality program and the organization required to support it.
- Real life examples of the use of the RBT techniques.

A description of Caliber-RBT, the tool that supports the RBT Process.

- Identifies logical inconsistencies in the requirements specification.
- Translates the input Cause-Effect Graph statements into logical test cases.
- Evaluates the functional coverage achieved using previously existing test libraries.
- Provides a quantifiable yardstick for measuring the status of the testing effort.

Setting Up a Metrics Program for a Large Software Development Effort
Bill Trebilcock
Track 8: Software Risk Management and Metrics
Wednesday, April 19, Session 4: 10:15 am - 12:00 pm, Zia A

Sandia started work on the latest version of ground station software about four years ago. The Rational Software suite of tools (Rose, Apex, Requisite, etc.) was chosen because of a desire for an integrated tool set. While previous versions of the software used metrics, a formal metrics program was desired for the new version. The PSM (Practical Software Measurement) approach was chosen because of the Air Force customer and its library of suggested metrics. Project risks led to a prioritization of software issues which in turn led to a selection of metrics. The chosen metrics were analyzed in a metrics plan as to what data were needed, the source for each piece of data, when it would be available, what questions the metrics would answer and how the metric would be presented. Some of the metrics have been collected for over two years now. Examples will be shown and discussed from the following status and progress metrics: earned value and lines of code schedule metrics, schedule variance and planning reports development environment metrics, trouble tickets reports requirements metrics, system specification history. Several rules-of-thumb-type conclusions will conclude the presentation.

So I Believe Configuration Management Is a Good Idea. Now What?
Marjorie McCornack
Track 3: Configuration Management and Defect Tracking
Tuesday, April 18, Session 2: 2:15-4:00 pm, Zia B

Configuration Management (CM) is foundational to developing quality software. Classic CM functions are identification, control, auditing, and status accounting. The activities that result from these functions result in controlled, reproducible, and traceable software. On the other hand, CM introduces traits into software development such as discipline, structure, and accountability to which developers are resistant. Developers often claim that CM will stifle their creativity.

In my talk, I'll start by defining configuration management from the perspective of one who has learned CM through the trenches. Then I'll discuss different practical activities encompassed by CM, how to start establishing a CM infrastructure, and what are some factors to be considered when first establishing a CM infrastructure. Finally, I will address ways of possibly making CM more palatable to reluctant developers, and we'll develop the argument that we are not stifling the developers' creativity, but providing an infrastructure in which their creativity can flourish.

The audience will be asked to contribute insights from their experiences as we look at establishing a CM infrastructure palatable to reluctant developers.

Software Product Processes Web-Based Training—Overview
Philip Huffman, Cathy Kuhn, Orval Hart, Dave Peercy, Patty Trellue
Track 2: Project Management and Team Building Strategies
Tuesday, April 18, Session 3: 4:15-6:00 pm, Devargas

It is DOE/AL policy that a Nuclear Weapons Complex (NWC) wide Technical Business Practices (TBP) system be established for implementing DOE requirements and to provide a mechanism for disseminating technical policies and procedures within the NWC. All NWC agencies are participants in this system (reference DOE/AL Appendix 56XB, Development and Production Manual). As of July 16, 1999, the TBP system was released for use. One of the TBPs, TBP-306, specifically applies to Software Product Processes. TBP-306 provides information specifically applicable to software product program management, concurrent qualification, identification and traceability, definition, design control, records management, off-the-shelf and interagency transfers, acceptance, support, and safety. Design Guide DG10235 provides supplementary details with many examples to support the general guidance. To assist implementation of TBP-306 while minimizing the cost of training, the NWC Software Quality Assurance Subcommittee has initiated a work item to develop Web-based training materials. This session will provide an on-line overview of the TBP's and some of the training material being developed in support of TBP-306. A panel discussion will follow to answer audience questions concerning software product processes and web-based training within the TBP system.

Software Team Strategies used for WIPP
Kathleen Byle, Co-Author: Jeanne Evans
Track 2: Project Management and Team Building Strategies
Tuesday, April 18, Session 2: 2:15-4:00 pm, Devargas

Background: The Waste Isolation Pilot Plant, Located in Southeastern New Mexico, received its first shipment of Nuclear Waste March 26, 1999. The decision to certify the WIPP was based on Sandia National Laboratory's (SNL) evaluation that the site could safely contain transuranic waste. The evaluation performed by SNL was accomplished using software to simulate the overall performance of the WIPP as a geologic repository. The time frame from identification of regulatory requirements, to the use of qualified software to evaluate the site in terms of public health and safety was extremely short. The success of the project depended on the construction of high-performance software teams.
Focus of Presentation: This presentation first describes the successful project management strategy and methods used for building high-performance software teams and keeping the teams on track with customer requirements. The presentation then describes how these teams have been improved in response to the post-certifcation environment of WIPP. The change management process has used a standard business model of defined roles and responsibilities for a Champion, Team Lead, and Core members, adjusted for a technical environment. This improved team strategy addresses the following new challenges: 1) a significantly reduced budget, 2) improving efficiency

of site evaluation, 4) upgrading the simulation models, 5) testing and requalifing the software, and 5) improving the user interface for use by the regulator.

Testing an Object-Oriented Controls System
Carolyn Owens
Track 1: Testing Architectures and Strategies
Tuesday, April 18, Session 2: 2:15-4:00 pm, Zia A

The Open Modular Architecture Controller (OMAC) will provide a PC-based, open architecture, software controller operating environment that can be applied to a variety of equipment, including machine tools, robots, and general manufacturing. A generic controller, such as the OMAC, will enable manufacturers to quickly reconfigure machine tools for producing various parts. The OMAC software controller is an object-oriented system written in Java that will initially control a 3-axis numerical control (NC) milling machine.
Due to the complexity of this application, the real-time aspects, and the highly iterative characteristics of the object-oriented environment, testing the OMAC software presents many challenges. Therefore, the OMAC project philosophy, approach, and process of testing the software throughout the project life-cycle will be discussed. Additionally, testing tools, test plans, test cases, and testing results will be presented and discussed.

Using Object-Oriented Metrics to Support Reliability: A General Discussion
Ruth Stapko, Dr. Linda Rosenberg, Mike Parizer
Track 8: Software Risk Management and Metrics
Wednesday, April 19, Session 5: 1:45-3:30 pm, Zia A

Object technology is being utilized more and more on projects of all applications across industry, and NASA is no exception. The Software Assurance Technology Center (SATC) at NASA's Goddard Space Flight Center outside Washington, DC has spent over two years identifying appropriate object-oriented metrics. Based on this research, the SATC has begun compiling a metrics database from over 20,000 object-oriented classes in both C++ and Java. A ranking and reporting process was then developed to evaluate the attributes of high-quality, object-oriented design and coding. Based on initial findings, guidelines for expected reliability have been developed. These guidelines were applied to project data and presented to project personnel at all levels, from senior managers to project managers, quality assurance professionals, software engineers and experienced as well as entry level programmers. Post-release feedback indicates that SATC's method for ranking object-oriented classes does indeed do a good job of predicting the future reliability of the software. This talk will discuss the metrics used, a historical database of metrics information, and the ranking of code for reliability.

The Urgent Need for Configuration Management

Susan Dart

Track 3: Configuration Management and Defect Tracking

Tuesday, April 18, Session 1: 10:30 am - 12:15 pm, Zia B

Configuration management (CM)—the disciplined approach to managing the evolution of software development and maintenance practices and their products—has matured to the point where organizations now cannot thrive, let alone survive, without CM capabilities. E-commerce and the software industry's quest for process improvement (via the Capability Maturity Model or ISO 9000) have created an urgent need for CM. Organizations are being driven into an automated CM solution from a variety of causes: their CM pain level (such as inability to deliver software changes on time), mergers, new development tools, the failure to pass audits, the overwhelming pace and volume of web changes, the need to leverage competitive advantage, the desire to achieve standards certification (such as the various CMM levels) by the sheer complexity of being successful (the web variant explosion problem), and the challenges of web maintenance in order to avoid a "web crisis." Industry has learned many lessons from its software crisis: namely, that CM is a mission-critical, foundational practice. Organizations have seen the value of a full-featured process-oriented CM solution: improved release cycle times, 50% reduction in bugs, automatic quality control, minimal change complexity, variant or concurrent release management, visibility into all work status, better forecasting of release dates, 15 minute audits, repeatability of processes and product creation, more responsiveness to customers, and faster standards certification (such as CMM level 2) among other benefits. Full-featured, process-oriented CM solutions provide quality automation by operationally supporting version control, parallel and distributed development, build-and-release management, workspace and repository management, change tracking, status reporting and audit control. This presentation discusses why CM is urgently needed today; what a best practice, fully automated CM solution looks like, how the CM tool industry has matured, and the role CM plays in process improvement goals.

John M. Gilligan, Chief Information Officer, U. S. Department of Energy



The chief information officer (CIO) reports to the deputy secretary of energy and is responsible for the development and oversight of department-wide information technology management strategies, policies, and practices. These responsibilities include year 2000 readiness; protection of information; strategic planning; standards in computing, networking, and security; establishment of system and information architectures; and corporate information technology, capital planning, and investment.

Before becoming CIO, John M. Gilligan was the Air Force program executive officer for battle management, stationed at the Pentagon, Washington, D.C. He was responsible and accountable to the Air Force acquisition executive for the cost, schedule, and performance of a portfolio of assigned major and selected acquisition programs.

Mr. Gilligan earned a bachelor's degree in mathematics from Duquesne University in 1973, a master of science degree in computer engineering from Case Western Reserve University in 1976, and a master's degree in business administration from Virginia Polytechnic Institute in 1984. Before assuming the position of Air Force program executive officer, he was director of resources for the deputy chief of staff, Command, Control, Communications and Computers at U.S. Air Force Headquarters, Washington, D.C.

EDUCATION:

1973 Bachelor's degree in mathematics, Duquesne University, Pittsburgh, PA
1976 Master of science degree in computer engineering, Case Western Reserve University, Cleveland, OH
1984 Master's degree in business administration, Virginia Polytechnic Institute, Blacksburg, VA
1989 Federal Executive Institute Program, VA
1990 Systems Acquisition Management Course for General and Flag Officers, Defense Systems Management College, Fort Belvoir, VA
1991 Program for Technical Managers, Carnegie Mellon University, Pittsburgh, PA
1992 Program for Executives in National and International Security, Harvard University, MA

ASSIGNMENTS:

July 1975 - September 1977, Systems Analyst, System Development Corp., McLean, VA

September 1977 - October 1978, technical staff, Defense Communications Division of International Telephone and Telegraph, Nutley, NJ

November 1978 - December 1981, Branch Manager for computer security projects, System Development Corp., McLean, VA

December 1981 - December 1984, Computer Scientist, Worldwide Military Command and Control System (WWMCSS) Information System (WIS) Joint Program Office, McLean, VA

December 1984 - September 1987, Deputy Director, WIS System Program Office, Hanscom Air Force Base, MA

October 1987 - October 1990, SES, Director of Studies and Analysis, Air Force Communications Command, Scott Air Force Base, IL

November 1990 - August 1991, Director of C4 Software Architecture and Policy, Deputy Chief of Staff, Command, Control, Communications and Computers (C4), U.S. Air Force Headquarters, the Pentagon, Washington, DC

August 1991 - October 1992, Director of Resources, Deputy Chief of Staff, Command, Control, Communications and Computers (C4), U.S. Air Force Headquarters, the Pentagon, Washington, DC

November 1992 - October 1998, Air Force Program Executive Officer for Battle Management, the Pentagon, Washington, DC

MAJOR AWARDS AND DECORATIONS:

Joint Chiefs of Staff Distinguished Civilian Service Medal

Air Force Meritorious Civilian Service Medal

Senior Executive Bonus Awards

SES Meritorious Executive Presidential Rank Award

PUBLISHED ARTICLES:

"A Generic Security Architecture for Distributed Systems," Proceedings of the 7th Conference on Local Computer Networks, IEEE, October 1982

"Security Design Issues of Microprocessor-Based Systems," Proceedings of 1979 Fall COMPCON Conference, September 1979

"Structured Specification of a Security Kernel," Proceedings of the International Conference on Reliable Software, June 1975

"Modernization of the WWMCCS Information System (WIS)," WIS JPMO, July 1982

"Technology Assessment: Methods for Measuring the Level of Computer Security," December 1980

Norm Brown, Executive Director of the Software Program Managers Network

Norm is the change agent for over two hundred large-scale, software-intensive Army, Navy, Air Force, and other defense programs. He founded the Software Program Managers Network, which provides technical and management consulting support to these programs and has some 10,000 members. Norm founded the Airlie Software Council; led the Defense Department's Software Acquisition Best Practices Initiative; was a member of the DoD Software Management Review Board; chaired the Joint Logistics Commanders Group on Technical Data and Computer Software and the group that drafted the federal acquisition regulations on technical data and computer software; and served on numerous other groups and committees. He was responsible for the review of over 300 defense software acquisition projects. Norm served on numerous acquisition selection boards and in advisory positions to the secretary of the navy, deputy secretary of defense, and other defense officials; and served as an advisor to the Navy and Defense Science boards. He led extensive benchmarking of commercial best management and technical practices of large-scale software development projects. He was the software manager for a number of real-time Navy weapons systems and was a commercial software developer.

EDUCATION:

BSEE, MSEE, JD (LAW)

BACKGROUND:

IEEE & ACM

Programmer & Systems Analyst, downstate Medical Center

Software Manager, BQQ-5 submarine sonar system

Special Assistant for Software to DoD's Senior Official for Mission Critical Computer Resources (DUSD(R&T))

Drafted Defense Science Board report section on instruction set architectures

Reviewed 255 programs/year for ASN (S&L) for approximately 6 years

Chaired Enhanced Modular Signal Processor review for ASN (S&L)

Member, SECNAV ADP Acquisition Review Board

Chaired SECNAV ADP Project Acquisition Approval Committee

Member, DoN ADP Strategic Planning Board

Member of various source selection boards

Advisor to SSAs on major software acquisitions

Advisor to Navy and Defense Science boards

Chaired drafting of FAR section on tech data & computer software

Chaired JLC group on technical data & computer software

Director, Assessment & Support Management ASN(RDA)/NISMC

Member, DoD Software Acquisition Education Review Team (USD(A&T)/AR)

CURRENTLY:

Director, DoD Software Acquisition Best Practices Initiative

Founder and Executive Director, Software Program Managers Network

Member (ex officio), DoD Software Management Review Board

Special Assistant for Software Management (ASN(RDA)/C4I)

Dennis Adams,
Idaho National Engineering and Environments Laboratory

Dennis Adams is currently a Consulting Program Manager at the Idaho National Engineering and Environmental Laboratory. During his 32-year career he was responsible for the development of mission critical systems for NASA, the U. S. Air Force, the DOE, and commercial nuclear power. These systems included man-rated flight hardware, computer-controlled nuclear refueling systems, emergency response systems, air war planning, and secure communication capabilities. He has provided Verification and Validation services for the Army, DOE, and commercial industry. He has a B.S. in Physics from New Mexico Highlands University with graduate work at Washington State University and the University of Colorado in Physics, EE, and Computer Science.

Michael Bell,
Lockheed Martin Energy Systems

Michael Bell is a software engineer with Lockheed Martin Energy Systems at the Y-12 Plant. He is a principal investigator on the Classification Resource System project, as well as a member of the LMES software metrics team. He has worked in the Oak Ridge area for nineteen years, at both Y-12 and Oak Ridge National Laboratory. His experience includes research- and production-oriented software, in areas such as language analysis, document workflow, imaging, manufacturing automation, access control, plasma physics, meteorological modeling, and econometrics. In this capacity, he has performed user interface and database design, application migration (cross-platform and mainframe-to-workstation), real-time device control, modeling, statistical and graphical analysis, and all aspects of structured and object-oriented software development. Mike holds a bachelor's degree in mathematics and a master's degree in computer science, and is an IFPUG Certified Function Point Specialist.

Jane M. Booker,
Los Alamos National Laboratory

Dr. Booker joined the Statistics Group at Los Alamos National Laboratory in 1980 and was the Group Leader from 1992-1998. She received her Bachelor of Science degree in Meteorology/Engineering, her Master of Statistics and Ph.D. in Statistics, all from Texas A&M University. She has 2 years of university teaching experience, 10 years of technical management experience, and over 20 years of consulting experience in scientific and engineering application areas. Her recent research and development efforts have been in the areas of statistical reliability, information integration methods, and analysis of expert opinion. During her career, she has published in various journals and coauthored a book with Mary Meyer, *Eliciting and Analyzing Expert Judgment: A Practical Guide*, published by Academic Press, London, 1991. She is a member of the Institute of Mathematical Statistics, a Fellow of the American Statistical Association, a 1995

recipient of the H.O. Hartley award for service to the statistics profession, and current associate editor of *The American Statistician.* In 1999, she received the prestigious R&D 100 Award presented by R&D Magazine for the reliability methodology, PREDICT, in conjunction with the Delphi Automotive Systems.

Julie F. Bouchard,
Sandia National Laboratories

Julie has been with Sandia National Laboratories for 10 years. She is a project lead and applications analyst for IDART (Information Design Analysis Red Team). IDART provides vulnerability assessments for a variety of government and commercial customers. Julie has a Computer Information Systems degree, eight years experience in system and network administration, and seven years experience in development and testing of military applications.

Kathleen Byle,
Sandia National Laboratories

Kathleen is a software quality engineer at Sandia National Laboratories, in Department 12326. She has over 10 years experience in software engineering and quality assurance. She is a Certified Nuclear Quality Assurance Lead Auditor. Prior to joining the quality department, she was responsible for writing and implementing the Waste Isolation Pilot Plant software program. Previously Kathleen was a Research Scientist for the University of New Mexico, contracted to NORAD to design and program their environmental control system. Before working for UNM, Kathleen was a software engineer on a team developing a computer-assisted stereotactic system. She holds a degree in Computer Science from the University of New Mexico.

Susan Dart,
Dart Technology Strategies, Inc.

Susan Dart is President of Dart Technology Strategies, Inc., a four-year old, independent consulting firm that helps companies attain the best possible configuration management (CM) solution. This is achieved via the proper evaluation and deployment of technology and processes using simple, risk-based techniques. CM is the foundation for any software development and maintenance environment. and is assuming a key role in enabling organizations to control their distributed objects, particularly for Web content-management and electronic commerce. Ms. Dart has 24 years of experience in industry and academia, focusing on software tools, development environments, and technology adoption. She has over 60 international publications and seminars to her credit, including coauthor of the 1996 Ovum book on Evaluating Configuration Management Tools. Her forthcoming book "CM: The Missing Link in Web Engineering" will be available July, 2000. Ms. Dart was a member. of the United States' Federal Aviation Authority

committee on developing CM for bomb detection systems. She is also a member of the Program Committees for the International Conferences on Web Engineering and on Configuration Management. Before starting her own company, Ms. Dart was Vice President of Process Technology at Continuus Software Corporation, where she implemented deployment services to assist strategic customers in achieving the best possible, enterprise-wide CM solution. Previous to that, she spent seven years at the Software Engineering Institute (SEI) of Carnegie Mellon University (CMU), creating models for CM and evaluating software development environments. Prior to the SEI, Ms Dart developed compilers at Tartan, Inc., and telecommunications software and international standards for Telstra, Australia. Ms. Dart has a Master's Degree in Software Engineering from CMU and a Bachelor's Degree with Distinction in Computer Science from RMIT.

Shawn Dawson,
Lawrence Livermore National Laboratory

Shawn Dawson is a computer scientist at Lawrence Livermore National Laboratory (LLNL). He currently works with a team of computer scientists and physicists developing a large-scale, massively parallel, physics simulation program. This program can run either sequentially or in parallel on greater than 1000 processors. He is primarily responsible for computer-science aspects of the code. In 1999, he implemented a method for performing regression testing of the code. Mr. Dawson has worked with his current group since 1995. From 1992-1995, Mr Dawson worked half-time at LLNL on a parallel global climate modeling code while he attended graduate school. From 1988-1992, he worked as a computer scientist with Electronic Data Systems (EDS) developing business applications. Mr. Dawson has a Master's Degree in Computer Science from the University of California, Davis (1995) and a Bachelor's from Western Oregon State University (1988).

Evi Dube,
Lawrence Livermore National Laboratory

Evi Dube is an administrative group leader for 6 computer scientists, and a co-project leader for a large ASCI simulation code project, ALE3D, with 22 engineers, physicists, chemists, and computer scientists. Her professional background started in 1984 at LLNL, and is in scientific computing, emphasizing working on super computers in parallel, distributed environments. Besides working full time, she completed her Ph.D. (1996) in Applied Science/Computational Science from UC Davis. Her thesis work has been applied to the ALE3D project.

H. W. Egdorf,
Los Alamos National Laboratory

Mr. Egdorf is a Technical Staff Member in the Computing Division of Los Alamos National Laboratory where he serves as an architect and designer for large software systems. Mr. Egdorf joined Los Alamos in 1980 after work with the U.S. Geologic Survey. He is a member of the ACM and was a recipient of the USENIX lifetime achievement award in 1996.
Mr. Egdorf has a Bachelor of Science degree in Engineering Mathematics from the Colorado School of Mines in Golden Colorado, and a Master of Science in Computer Science from the University of New Mexico.

Gary Froehlich,
Sandia National Laboratories

Gary Froehlich joined Sandia National Laboratories in 1981, after graduating from Texas Tech University with degrees in Electrical Engineering. He spent a dozen years as an analyst and modeler in radar systems and antenna systems, and authored numerous codes for simulations of system behavior, including the characterization of scattering cross sections for various radar targets. In 1993, he moved to the WIPP project, where his duties included co-leading the team responsible both for negotiating the software QA and configuration management requirements imposed by the regulator, as well as for implementing those requirements across the project. In December 1999, he joined the Distributed Decision Support Systems department, which assists other groups at Sandia with all aspects of software engineering.

ROBERT E. HEARD, PMP,
Idaho National Engineering and Environmental Laboratory

Mr. Heard received his Bachelor of Science degree in Electrical Engineering in 1977 from the University of Arizona. Mr. Heard has over thirty years of professional experience in nuclear facility operation or management, complex programs, projects management, and resource management. Mr. Heard is currently the Deputy Program Manager for the Idaho National Engineering and Environmental Laboratory (INEEL) Defense Programs (DP) Environmental Surety Program.
Mr. Heard has worked on a number of environmental and research and development programs at the INEEL including the Buried Waste Integrated Demonstration program, a program whose mission was to demonstrate new and innovative ways of locating, catagorizing, and retrieving buried waste. Mr. Heard served as the Manager of the Project Management organization for EG&G Idaho, Inc. During this time, Mr. Heard was instrumental in establishing a formal professional development program in Project Management for the company. Mr. Heard is a Nationally certified Project Manager and a member in good standing in the National Project Management Institute.

Ann Hodges,
Sandia National Laboratories

Received a Masters in Computer Science. Has 17 years software development experience for CAD/CAM applications, mobile command and control subsystem, distributed resource management system for high performance distributed computing community. Has 11 years project leader experience as well as 9 years V&V/test management experience for mobile command and control subsystem, seismic treaty verification and monitoring system, WIPP
simulation codes for site and geophysical processes, the FAA Safety Performance Analysis System for aircraft inspectors, simulation of aircraft vulnerabilities to a threat encounter, enterprise-wide applications, a control strategy simulation code for the Hanford waste storage facility, ASCI Apps V&V PI, and CPLANT productionization.

Philip Huffman,
Mason & Hanger, Pantex Plant

Philip Huffman is a Professional Engineer who is a Section Engineer for Special Nuclear Materials Technology at the NWC Pantex Plant. He has four years experience in automated process equipment design and ten years experience in automated test system design at Pantex. He specializes in RF systems and software design. Philip is fluent in five operating systems (Unix, CPM, DOS, Win 3.1/Win 95/Win 98, NT) and seven programming languages (Fortran, PL/1, Forth, Basic, Pascal, C/C++, Intel & Motorola Assemblers). Philip received a B.S. in Chemical Engineering in 1983 and B.S. in Electrical Engineering, in 1985, both from Texas Tech University.

Marie-Elena C. Kidd,
Sandia National Laboratories

Marie-Elena C. Kidd is a computer scientist and Distinguished Member of the Technical Staff at Sandia National Laboratories. During her thirteen years at Sandia, she has worked as a software engineer on embedded, real-time software systems for such applications as robotics, nuclear weapon components, and control systems. She has been involved in software engineering process improvement activities at local and lab-wide levels for her entire career. She has also worked on lab-wide information sharing systems and initiatives. She has a B.S. in Computing and Information Sciences from Trinity University in San Antonio, Texas and an M.S. in Computer Science from Purdue University in West Lafayette, Indiana.

Cathy Kuhn, Honeywell,
Federal Manufacturing & Technologies

Cathy Kuhn is a Staff Information Technology Systems Engineer with Honeywell International, Inc. Federal Manufacturing and Technologies, Department of Energy, Kansas City Plant (KCP). She has over 23 years of experience as a programmer, project leader and software quality assurance analyst across a variety of projects. She spent 12 years as a project lead on automated test equipment software and 10 years defining and maintaining the Kansas City plant's software quality assurance program. Currently Cathy is assigned to the Information Technology Organization as process improvement and software quality assurance lead. Cathy holds a B.S. in Computer Science from the University of Missouri-Rolla.

Marjorie McCornack,
Sandia National Laboratories

Marjorie has a Bachelors in Math and Masters in Computer Science. She has worked at Sandia National Labs for over 16 years on numerous projects in varying roles. Projects: satellite ground systems, the seismic ground system, nonproliferation information system, FAA information modeling, endgaming simulation and modeling, Integrated Information Systems configuration management, and project productionization. Roles: project leader, database administrator, tester, configuration manager or configuration management support, and software developer. She started doing configuration management approximately 13 years ago and has contributed, whether explicitly or implicitly, to CM infrastructure and process definitions throughout those 13 years.

Mike McKay,
Los Alamos National Laboratory

Mike McKay came to Los Alamos National Laboratory in 1995. While originally knowing nothing about Makefiles, Perl, or CVS, he quickly learned how useful these tools can be when building software. It was knowledge of these tools that made him attractive to the Unified Parallel Software Project. He has been on the UPS team since 1997. While his main task is being in charge of Software Quality Assurance, he has also written several components of the UPS library. Mike received his B.S. in Physics from the University of New Mexico in 1992. He then received his M.A. in Mathematics from Eastern New Mexico University in 1995.

Gary Mogyorodi,
Technology Builders, Inc. (TBI)

Gary Mogyorodi has over 26 years of experience in the computing industry. Currently as a senior technology engineer for Technology Builders, Inc. (TBI), Mr. Mogyorodi consults, trains and mentors in software testing, specifically in test case design. Mr. Mogyorodi worked with Bender & Associates, which was acquired by TBI in the spring of 1999. Prior to his tenure with TBI, the majority of Mr. Mogyorodi's career was with Dofasco Inc. From 1992, Mr. Mogyorodi was a Quality Assurance and Software Testing specialist, managing testing efforts, developing testing methodologies, and creating standards and procedures for quality assurance and testing. Prior to that, he worked at Dofasco as a Programmer, Systems Analyst and Manager of Software Development. Gary Mogyorodi obtained a Bachelors of Math degree from the University of Waterloo, and an M.B.A. from McMaster University. He has delivered several presentations, including SQA User's Conference, CIPS Monthly Meetings (Toronto and Hamilton Chapters), TassQ (Toronto Association for System and Software Quality) Monthly Meeting, and the Star '99 West Conference.

Jaime L. Moya,
Sandia National Laboratories

Jaime L. Moya is currently a department manager of the Nuclear Weapons Support Group in the Engineering Science Center at Sandia National Laboratories (SNL). In addition, he is the program manager for the Experimental Validation Program (a $9.2 million/year program). Until recently, he had the responsibility of managing SNL's Accelerated Strategic Computing Initiative (ASCI) Verification and Validation Program. Prior to becoming a manager in 1995, he was a Member of the Technical Staff in the Thermal Sciences organization where his interests included the investigation of fire physics, the fire-induced response of materials, and safety issues concerning nuclear weapons. Jaime received his undergraduate degree in Mechanical Engineering from New State University and his graduate degree, also in Mechanical Engineering, from the University of California at Berkeley.

Harvey Ogden,
Sandia National Laboratories

Harvey Ogden joined Sandia National Laboratories in 1969 after graduating from the University of Montana with his Masters degree in Mathematics. He has worked his entire career at Sandia as a software developer. For twenty of those years, he specialized in real-time data acquisitions systems. Harvey joined the Waste Isolation Pilot Plant (WIPP) Performance Assessment (PA) department in 1994 and served as a co-lead of the Software Quality Assurance team. His primary responsibility has been the verification and validation of the WIPP PA software. Harvey is currently a member of the Distributed Decision Support Systems department, which assists other groups at Sandia with all aspects of software engineering.

David Peercy,
Sandia National Laboratories

David Peercy is a DMTS software quality engineer with responsibilities for ensuring the quality of weapon/weapon-related software products. Primary projects include the Code Management System use control software products and the Software Quality Engineering software practice improvement projects for the SNL ASCI code teams. He is also the primary investigator on an internal research project to define an improved Requirements Assessment Framework for high consequence systems. David completed his Ph.D. in Mathematics at New Mexico State University in 1971. He has almost 30 years of experience in software development and quality engineering and has numerous publications in software supportability, maintainability, reliability, and process improvement.

Brett D. Rasmussen,
Idaho National Environmental and Engineering Laboratory

Mr. Rasmussen received his Bachelor of Science in Computer Science in 1987 from Utah State University. Mr. Rasmussen has over 13 years of professional experience designing, developing, and testing software applications. Mr. Rasmussen is a Principal Engineer at the Idaho National Environmental and Engineering Laboratory (INEEL) and is currently developing applications in support of the Defense Programs (DP) Environmental Surety Program. Mr. Rasmussen has worked on many different software projects throughout his career ranging from development of large Computer Program Components (CPCs) to control custom hardware devices, to development of aviation logistics systems for the U.S. Army. Mr. Rasmussen has also worked as a private software consultant and developed commercial applications for the financial management industry. Mr. Rasmussen is currently developing simulation and web applications using Java, HTML, and the Extend simulation tool set.

Don Rathbun,
Honeywell International, Inc.

Mr. Rathbun is a Staff Engineer with Honeywell International, Inc. Federal Manufacturing & Technologies Kansas City Plant (KCP). He has thirty-eight years of experience as an electrical engineer in process and product engineering including 15 years of project lead positions on radars. Mr. Rathbun also had lead positions on two major information systems initiatives at the KCP and on the ISO 9001 Certification Team at the KCP. Currently Don is assigned to the Science-Based Design & Information Management Organization at Honeywell FM&T and has the Industry Lead position on the CAM-I (Consortium for Advanced Manufacturing International) Model Based Engineering Project. Don holds a B.S.E.E. from Kansas State University and an M.S.E.E. from the University of Missouri and is a Licensed Professional Engineer in Missouri.
In addition, Don is active in volunteer organizations in Kansas City with board of directors/leadership team positions on three such organizations. Don was also a presenter at the 1997 SQForum held in Albuquerque.

Gerald Reisz,
Los Alamos National Laboratory

Mr. Reisz worked at Chemical Abstracts Service (CAS) from 1974 to 1984 in the Chemical Information Science department. During this time he participated in the design and implementation of many of the early on-line search systems developed for the abstracting and indexing services. He was also responsible for the production of the 1971 to 1980 decennial issue of the CAS Source Index. Mr. Reisz has worked at Los Alamos National Laboratory (LANL) from 1984 to the present. During this time he has worked on various information systems ranging from Tribology for the National Institutes of Standards and Technology to mission profiles for the armed services to the programs used by the emergency response team at a nuclear weapon accident. Mr. Reisz has been the Deputy Group Leader of the Scientific Software Engineering Group since 1995.

Joseph R. Schofield, Jr., CQA
Sandia National Laboratories

Joe has been active in the application of emerging technology for business and engineering solutions at Sandia National Labs since receiving his MS/MIS from the University of Arizona in 1980. As a principal member of the technical staff, he has been involved in the specification, selection, and application of software methodologies; served on a corporate-wide software quality improvement team, and facilitated teams in their quest for new business processes and customer-driven software. For the past ten years, Joe has developed and taught IS courseware in the MBA program at the College of Santa Fe. In 1990 he received his CQA accreditation.

After speaking at USE, SHARE, GUIDE, and DOE-sponsored conferences, Joe delivered a keynote address at the Structured Development Forum in San Francisco in 1988. Subsequently he spoke on CASE at the National Conference on Information Systems Quality Assurance, and at CASEWorld in Los Angeles. Articles on CASE were then published by the Journal of Quality Data Processing and System Builder including "*Considering CASE: Write the Fine Print.*" The publishers of System Development approached Joe to author an article for them; "*CASE Users Bill of Rights*" resulted. This effort was followed with three more thus far: "*CASE: Not a Joke, Now a Threat,*" "*The Next Silver Bullet*" in 1995 and then "*The Year 2000 - Finally a Reality Check*" which was one of the first articles to downplay the hysteria on Y2K and warn of the impending consulting infestation.

Published Articles

Lighten Up (The Year 2000 - Finally a Reality Check); System Development, June 1997

The Next Silver Bullet; System Development, August 1995

CASE: Not a Joke, Now a Threat; System Development; October 1994

CASE: A User's Bill of Rights; System Development; January 1992

Considering CASE: Write the Fine Print; System Builder; August 1990

Emerging Technologies Enrich Opportunities for Software Quality Improvements; Quality Data Processing; April 1990

4GLs Make Great Strides, But At What Sacrifice; Government Computer News; July 8, 1988

Electronic Forms Should Be Central To Future OA; Government Computer News; December 4, 1987

Presentations (materials and/or articles published in proceedings)

Applying Use Cases for Process Transformation, Framework Planning, Testing & Measurement, Annual International Information Technology Quality Conference, Orlando, Florida, April, 1999

The Next Silver Bullet – Or Just Another Shot in the Foot?, Software Quality Forum, Alb., NM, April, 1997

I-Case Success Stories; Piedmont CASE User's Group; August 1993

CASE: Avoiding the Myths, Applying the Technology; CASE World; March 1991

Standardizing Complexity Characteristics in Function Points—A Process Improvement; Eighth International Conference on Measuring Data Processing Quality; Orlando, Florida; March, 1990

CASE and AI: Influencing Productivity in Software Engineering; National Conference on Information Systems Quality Assurance; Orlando, Florida; April, 1989

The New Report Writer: More Than Just a Migration Path; GUIDE; Chicago, Illinois; July, 1988

The New Report Writer; SHARE; Anaheim, California; March, 1988

Electronic Forms - How Close the Value-Added Processing; Automated Office Systems Conference; Pittsburgh, Pennsylvania; 1986

Altering the Application of the Systems Development Life Cycle; Univac Scientific Exchange; Orlando, Florida; April, 1984

Altering the Application of the Systems Development Life Cycle; Univac Scientific Exchange; Dallas, Texas; October, 1983

Greg Schottland,
Advanced Software Technologies, Inc.

Greg Schottland founded Advanced Software Technologies, Inc., and is responsible for the initial design of its flagship product, GD*Pro*. He oversees all activities of the company and is the primary driver behind new business development and partnering relationships. Additionally, he provides sales and marketing guidance as well as high-level consulting in the evolution of the company's products. Schottland's dual technical/business background provides him with many of the necessary abilities to effectively manage Advanced Software Technologies and to assure the successful sales, marketing and development of the company's products.

Schottland began his career at Martin Marietta where he was involved in the Smart Weapons program, an advanced research project that involved placing artificial intelligence software inside a SCUD-seeking "smart" missile. He was responsible for development of the systems "planner," which allowed the missile to search for SCUDs in an intelligent and effective manner. The contract was highly successful and achieved notoriety among the DOD research community. Following his work on the Smart Weapons program, Schottland joined the Flight Telerobotic Servicer (FTS) project, a $300 million NASA space robotics project devoted to developing the first space-based robot capable of assisting astronauts in the performance of many tasks. He was responsible for the design and development of the FTS Operator Interface, which controlled robot operation, and for coordination of various groups both within FTS and NASA to assure proper evolution of this component of the FTS system.

Schottland received his Bachelor of Science in Business from the University of Colorado in 1986, graduating Magna Cum Laude. In 1988, he received his Masters of Science in Computer Science from the University of Illinois where his studies and thesis focused on Artificial Intelligence research—specifically learning systems and automatic program development.

Debra Sparkman,
Lawrence Livermore National Laboratory

Debra Sparkman is a staff member for the Lawrence Livermore National Laboratory Software Technology Center. She has a B.S. in Computer Science from the University of the Pacific. Ms. Sparkman has over 15 years of experience in software engineering, and 8 years in the software quality assurance and testing fields. She is an American Society for Quality, certified software quality engineer.

Ruth K. Stapko,
Unisys

Ruth K. Stapko is a principal software engineer, who has been with Unisys in the Software Assurance Technology Center at the Goddard Space Flight Center for three years and leads the project metrics group. Ms. Stapko has 15 years of software quality assurance experience. She is expert in the collection, analysis, and interpretation of software quality metrics. She provides metrics reports and tutorials to software developers and project managers and has coauthored and presented papers about Object Oriented Metrics and Risk Based Testing. As a statistician, she also designs tools for SATC's reliability research work. Ms. Stapko has a B.S. in Mathematics and M.S. Degrees in Statistics and Information Systems Management.

Michael "Mike" Tiemann,
Division of Architecture and Standards, Office of Architecture, Standards and Planning, Office of the Chief Information Officer, U.S. Department of Energy

Mike Tiemann has been in government service for 27 years. His Federal career started in 1972 at Army Material Command Headquarters, as a lieutenant working in environmental program management. The next 13 years or so were spent at the Federal Energy Regulatory Commission, as an Environmental Protection Specialist and as a Computer Systems Analyst, supporting the licensing of nonfederal hydroelectric power projects.

In 1987, he came to Headquarters, Department of Energy (DOE) as a Project Manager in the Office of Computers and Telecommunications Services. He was responsible for information technology services and support for 5 offices, including the Office of the General Counsel, the Office of the Inspector General, the Office of Hearings and Appeals, the Economic Regulatory Administration and the Board of Contract Appeals. Two years later, he was assigned the primary responsibilities for Information Resources Management Planning, Headquarters wide. In 1994, he was assigned the primary responsibilities for developing an Information Architecture for DOE. He assembled a nationwide team, representing programs, offices, sites and laboratories to develop a comprehensive Information Architecture Program. Recently, he was appointed the Director, Division of Architecture and Standards within the Office of DOE's Chief Information Officer (CIO). He is currently responsible for the continuing implementation of the Departmental Information Architecture.

Mike has led and been a member of many Department-wide teams, strategic planning efforts and interagency panels. He has been the Department's representative to the Federal CIO Council's Interoperability Committee since its beginning. He is the former Chair of the Federal Agencies Information Architecture Working Group which last September received the CIO Council's endorsement of the Federal Enterprise Architecture Conceptual Framework. He has been a speaker at Federal information management conferences, including the Government CIO Summit and the Interagency Resources Management Conference. Mike has made the natural progression to

information architecture educator, presenting lectures at the Federal Imaging Conference and at the IRM College of the National Defense University.
He has received numerous performance awards and special citations, including the Secretary of Energy's Pride Award, and Certificates of Appreciation from Armed Forces Communications and Electronics Assc. (AFCEA) and the Federal CIO Council. Mike Tiemann holds degrees in Architecture (Bachelors of Environmental Design, Texas A&M, 1972) and Systems Management (Masters of Science in Systems Management, U.S.C., 1977). He is a member of AFCEA and Association for Federal Information Resource Management (AFFIRM). He lives in Northern Virginia, is married and has two children.

Bill Trebilcock
Sandia National Laboratories

Bill is a PMTS software systems engineer at Sandia National Laboratories with thirty-seven years of experience in software engineering activities. He has worked for the National Aeronautics and Space Administration, Dikewood Corporation, Albuquerque Public Schools and Sandia. At Sandia he has worked in reactor safety, command and control of nuclear weapons, military liaison, and ICADS (Integrated Correlation and Display System). He has worked on the collection and presentation of metrics for the past seven years. Bill has a B.S. in Mathematics from Northern Michigan University (1962) and an M.S. in Applied Mathematics from Catholic University (1969). He has been working at Sandia since 1981 and on ICADS since 1996.

Patty Trellue,
Sandia National Laboratories

Patty is a PMTS software quality engineer at Sandia National Laboratories with twenty-five years of experience in software engineering activities covering all phases of the software life cycle. Her primary interests are in project management, risk management, quality management, and process improvement. She served as the Lead Assessor for the Tri-Lab Survey of Software Practices recently completed on Accelerated Strategic Computing Initiative code teams at Sandia, Los Alamos, and Lawrence Livermore National Laboratories. She is an active member of the DOE Quality Managers' Software Quality Assurance Subcommittee. In her position as a Software Quality Engineer, she applies the Technical Business Practices to the development and production of software. Patty has a B.S. in Mathematics from the University of Wisconsin (1971) and an M.S. in Electrical Engineering (Computer Science Option) from the University of New Mexico (1975). She has been working at Sandia since 1976 and in the Quality Engineering Department since 1997.

Timothy G. Trucano,
Sandia National Laboratories

Timothy G. Trucano is currently a Distinguished Member of the Technical Staff at Sandia National Laboratories in the Optimization and Uncertainty Estimation Department. He received his Ph.D. in mathematical physics from the University of New Mexico. Since 1980, his work has focused on research, development, and applications of computational shock wave physics. He has worked on problems of equation of state and constitutive modeling, hypervelocity impact phenomena, nuclear weapons effects, and radiation-hydrodynamics in high-energy density physics applications. Beginning in 1995, his work began to focus on issues of computational model verification and validation, as well as the application of uncertainty quantification in computational physics. Currently, he works on technical and programmatic challenges for the Accelerated Strategic Computing Initiative Verification and Validation Program at Sandia.

Panel Presenters:
Perspectives on Application of Software Process Assessments to Scientific Codes
The Software Quality Assurance Subcommittee (SQAS) operates under the DOE Nuclear Weapons Complex (NWC) Quality Managers to identify and resolve Software Quality issues and problems common to all DOE sites and facilities. This panel presentation is the result of an effort that was initiated through the NWC SQAS work item #24 on promoting and facilitating Accelerated Strategic Computing Initiative (ASCI) software quality. As a result, an extensive Tri-Lab Survey of Software Practices was conducted at each of the LANL, LLNL, and SNL NWC sites. The Nuclear Weapons Complex-wide participants and presenters of this panel include:

Dave Peercy/Patty Trellue, SNL
Dennis Adams, INEEL
Nancy Storch, LLNL / Gerald Reisz LANL
Evi Dube LLNL
Jaime Moya , SNL
Bill Reed, DOE HQ ASCI V&V Executive Council

David Peercy, Chair WI#24, ASCI Software Quality
Sandia National Laboratories
P.O. Box 5800, MS-0638
Albuquerque, NM 87185-0638
Voice: 505-844-7965
Fax: 505-844=3920
E-mail: depeerc@sandia.gov

Panel Presenters:

Software Product Processes Web-Based Training – Overview

The Software Quality Assurance Subcommittee (SQAS) operates under the DOE Nuclear Weapons Complex (NWC) Quality Managers to identify and resolve Software Quality issues and problems common to all DOE sites and facilities. This tutorial is the result of an NWC SQAS work item to develop web-based training materials for software as product based on the Technical Business Practices, especially TBP-306 on Software Product Processes. The work item was established to satisfy a need to bring greater awareness to the NWC concerning the process for handling product software. The Nuclear Weapons Complex-wide participants and presenters of this tutorial are tentatively scheduled to include:

David Peercy, Sandia National Laboratories
Patricia Trellue, Sandia National Laboratories
Philip Huffman, Mason & Hanger, Pantex Plant
Catherine Kuhn, Honeywell, Federal Manufacturing & Technologies
Orval Hart, Los Alamos National Laboratory

David Peercy, Chair WI#25, Software Product Processes Web-Based Training
Sandia National Laboratories
P.O. Box 5800, MS-0638
Albuquerque, NM 87185-0638
Voice: 505-844-7965
Fax: 505-844=3920
E-mail: depeerc@sandia.gov

McCabe & Associates, Inc.

McCabe & Associates...because some things just have to work right. In today's competitive business environment software is no longer a productivity tool, it has become the business and it must work right. McCabe & Associates enables IT to deliver better applications, by providing products and processes that implement a relevant, repeatable, and measurable approach to managing software changes and their effects on the testing and quality of applications. McCabe Integrated Quality (McCabe IQ), our proven methodology and product suite, tightly integrates change management with software quality analysis and increased test effectiveness to promote and assure the delivery of high-quality software systems. McCabe IQ has been implemented by global corporations and government agencies in their efforts to

- Manage and thoroughly test software changes,
- Quickly and easily merge software changes into released code,
- Objectively assure the quality of released software,
- Lower the maintenance time and costs of strategic systems,
- Assure the stability of enterprise software.

Mercury Interactive

Mercury Interactive is the leading provider of Web performance management solutions that help e-businesses ensure a positive user experience. Mercury Interactive solutions turn Web application performance, scalability, and user experience into competitive advantage. The company's performance management products and hosted services are open and integrated to best test and monitor business-critical Web applications.

Founded: August 1989
IPO: October 1993
NASDAQ Symbol: MERQ
Headquarters: Sunnyvale, CA
Worldwide Presence: 40 Offices
Employees: Approximately 900 Worldwide
Executives: Amnon Landan, President and CEO
Kenneth Klein, Chief Operating Officer
Sharlene Abrams, Chief Financial Officer
Moshe Egert, Vice President, European Operations
Jayaram Bhat, Vice President, Marketing
Ori Danieli, Vice President, R&D
Yuval Scarlet, Vice President, Technical Services
Zohar Gilad, Vice President, Product Marketing

Products:

Mercury Interactive's Web performance management solutions address the full range of e-commerce application testing and performance monitoring challenges facing e-businesses. The Company provides:

- Functional Testing Products: ensuring e-commerce applications work as expected;

• Load-Testing Products: Stress-testing e-commerce applications under real-world conditions to predict systems behavior and performance and to identify and isolate problems.
• Testing Process Management Products: Organize and manage the testing process to determine application readiness.
• Web Performance Monitoring Products: Monitoring Web applications in real time and alerting operations groups of performance problems before users experience them.
• Hosting Web Performance Monitoring Service: Proactively monitor sites in real time.
• Hosting Load Testing Service: Identify bottlenecks and capacity constraints before your site goes live.
Mercury Interactive's performance management solutions help business-to-business and business-to-consumer organizations enhance the scalability, reliability and availability of their mission-critical applications. We have established relationships with major e-business industry leaders such as Oracle, BroadVision, Vignette, SilverStream, Allaire, BEA, IBM, Ariba, Bluestone, Calico Commerce, CommerceQuest, DoubleClick, GemStone Systems, HAHT Software, IONA Technologies, OneSoft, Selectica, Sun-Netscape Alliance and Vision Software to ensure that our performance management solutions complement their unique products and reflect the unique needs of their customers.

Technology Builders, Inc.
Technology Builders, Inc., is an Atlanta-based software and services company providing integrated solutions for enterprise and e-business application development within the Automated Software Quality, Business Information Management and Internet and Client/Server markets. Through the TBI-Caliber® suite of software products and strategic vendor partnerships, the company provides best-of-class technologies in requirements management, requirements-based testing, test case design, configuration management, test planning and management, and defect tracking. Ranked fifteenth on the Deloitte & Touche Fast 500 list, TBI maintains a client base of over 500 companies, many among the Fortune 1000.

Products:
TBI markets and sells Caliber-RM®, a collaborative, Internet-based requirements management system that enables project teams to deliver higher quality applications. In an effort to detect and eliminate requirement errors and deficiencies earlier in the development cycle, Caliber-RM allows teams to fully define, manage, and communicate changing requirements. Changes to requirement data such as traceability, status, priority and more are recorded and stored in Caliber-RM's central repository, providing reliable, up-to-date information for effective requirements-driven development and testing.
Caliber-RM allows team members to compare project baselines to easily manage scope creep and identify other factors that may affect schedules and budgets. Caliber-RM's support for reusable requirements ensures that project teams can build from previous experience and applications, enabling more rapid development and better use of resources. With Caliber-RM, organizations can instill discipline in their development cycle through a structured requirements management process, minimizing the cost of

reworks due to requirement errors, focusing team members on the project scope and decreasing the likelihood of project failures and overruns.

TBI has integrated Caliber-RM with the following tools:

- Caliber-RBT™, the company's functional test case design tool
- TestDirector®, a test planning and management tool from Mercury Interactive®
- SELECT Enterprise®, an object modeling tool from Princeton Softech
- Estimate Professional™, a project estimation tool from Software Productivity Center
- MS Project®, a project management tool from Microsoft®

Another TBI product, Caliber-RBT, formerly SoftTest, is a requirements-based functional test case design system that drives clarification of the application requirements and streamlines the testing process. Caliber-RBT enables project teams to analyze and refine requirements to eliminate ambiguities and conflicts, then uses those requirements to design a minimum set of test cases for functionally complete test coverage.

Currently, Caliber-RBT is integrated with the following tools:

- TestDirector, a test planning and management tool from Mercury Interactive
- WinRunner®, a functional automated testing tool from Mercury Interactive

TBI also plans to develop integration for the TBI-Caliber® suite with other market-leading test management, project estimation and object modeling tools, as well as additional third-party tools such as source code management (with MERANT PVCS) in the near future.

Requirements-Based Testing Overview

Gary Mogyorodi, Senior Technology Engineer

Technology Builders, Inc.

400 Interstate North Parkway
Suite 1090
Atlanta, Georgia 30339

tel 770-937-7900
fax 770-937-7898
http://www.tbi.com

TBI

Requirements-Based Testing Overview

Copyright © 1977, 1985, 1987, 1991, 1992, 1994, 1995, 1996, 2000 Technology Builders, Inc. All rights reserved. No part of this manual (including interior design, cover design, and illustrations) may be reproduced or transmitted in any form, by any means, (electronic, photocopying, recording, or otherwise) without the prior written permission of the publisher.

For authorization to photocopy items for internal corporate use, personal use, or for educational and/or classroom use, please contact:

Educational Services, TBI, 400 Interstate North Parkway, Suite 1090 Atlanta, GA 30339 USA. Phone: 770-937-7900.

RM / RBT Integration Overview

- Caliber-RM
  - Manages the requirements objects
- RBT / Caliber-RBT
  - Addresses the quality of those objects
    - Correct
    - Complete
    - Logically consistent
    - Unambiguous

TBI

Caliber-RBT/Mercury Interactive Integration Overview

Test Activity		
Define Test Completion Criteria	CALIBER RBT	
Design Test Cases	CALIBER RBT	
Build Tests		M
Execute Tests		M
Verify Test Results		M
Verify Test Coverage	CALIBER RBT	
Manage Test Library		M

- Stabilize interface definition early
- Maximum coverage with minimum number of tests

TBI

Agenda

- The Business Case
- The RBT Process
- Management Considerations
- Summary

TBI

The Business Case – Goals

- ✓ Deliver MORE Function,
- ✓ In LESS Time,
- ✓ With FEWER Resources,

  AND

- ✓ With HIGHER QUALITY.

TBI

Areas Of Opportunity

Reduced Development Schedules And Budgets

- Minimize Scrap and Rework
- Increased Parallel Efforts

Reduce Support Costs

- Fewer Support Calls
- Eliminate Emergency Product Releases

Quality Moves Market Share

- Example: dBASE IV Release 1.0

Reduced Liability Exposure

TBI

What Are Software Errors Costing You?



American Airlines

Outages in SABRE cost $20,000 per minute!

1989 – 12 hour outage

1994 – 5 hour outage



System error incorrectly showed flights full – $50,000,000 loss!



Denver Airport

$1,100,000 per day lost due to defects in baggage handling system

TBI

Tester Job Security

- Over 15,000,000,000 Lines of Code Written Per Year
- At 5 Errors Per Thousand Lines (After Product Release): 75 Million New Bugs Per Year

TBI

The Test Process – Make It:

- **Timely:**
  - Integrate Throughout the Life Cycle
- **Effective:**
  - Rigor in Test Definition
- **Efficient:**
  - Heavily Automated
  - Minimum Number of Tests
- **Manageable:**
  - Measurable
  - Predictable

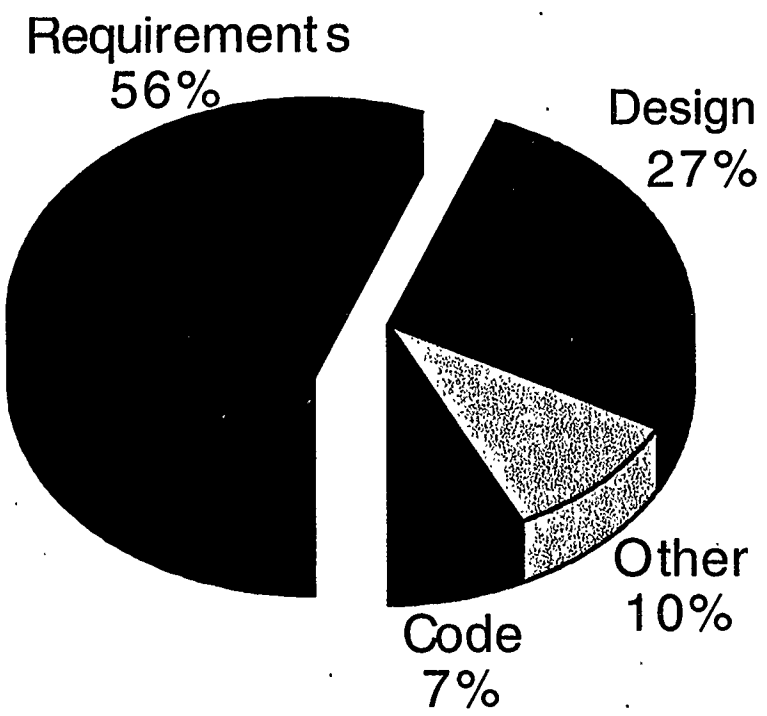
Relative Cost To Fix An Error

Phase In Which Found	Cost Ratio
Requirements	1
Design	3–6
Coding	10
Unit/Integration Testing	15–40
System/Acceptance Testing	30–70
Production	40–1000

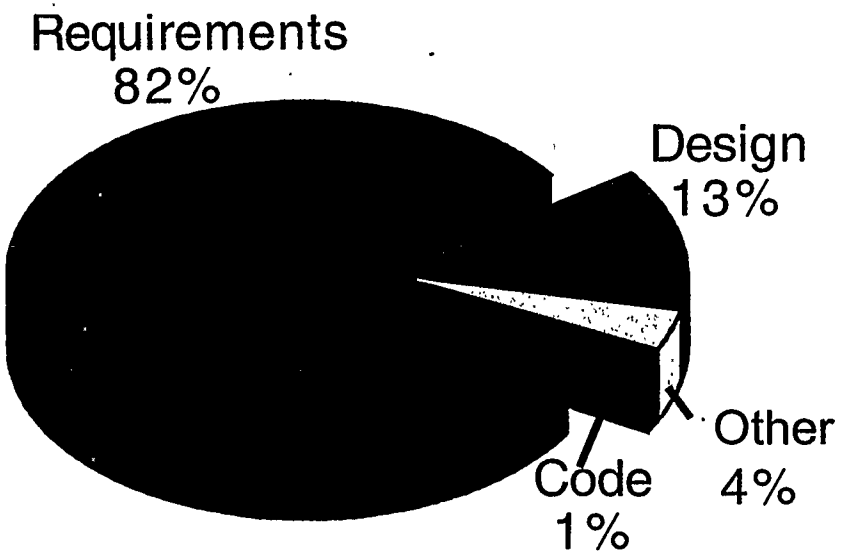
(IBM, GTE, et. al.)

TBI

Distribution of Bugs



Distribution of Effort To Fix Bugs



(James Martin)

RBT20180

TBI

Why Good Requirements Are Critical

In 1997 American Companies Spent $100 BILLION For Canceled Software Projects

$45 BILLION Spent For Projects Which Significantly Exceeded Time And Budget Estimates

(Standish Group)

TBI

Why Good Requirements Are Critical

Top Reasons For Failure

Incomplete Requirements And Specifications

Changing Requirements And Specifications

Lack Of User Input

(Standish Group and Other Studies)

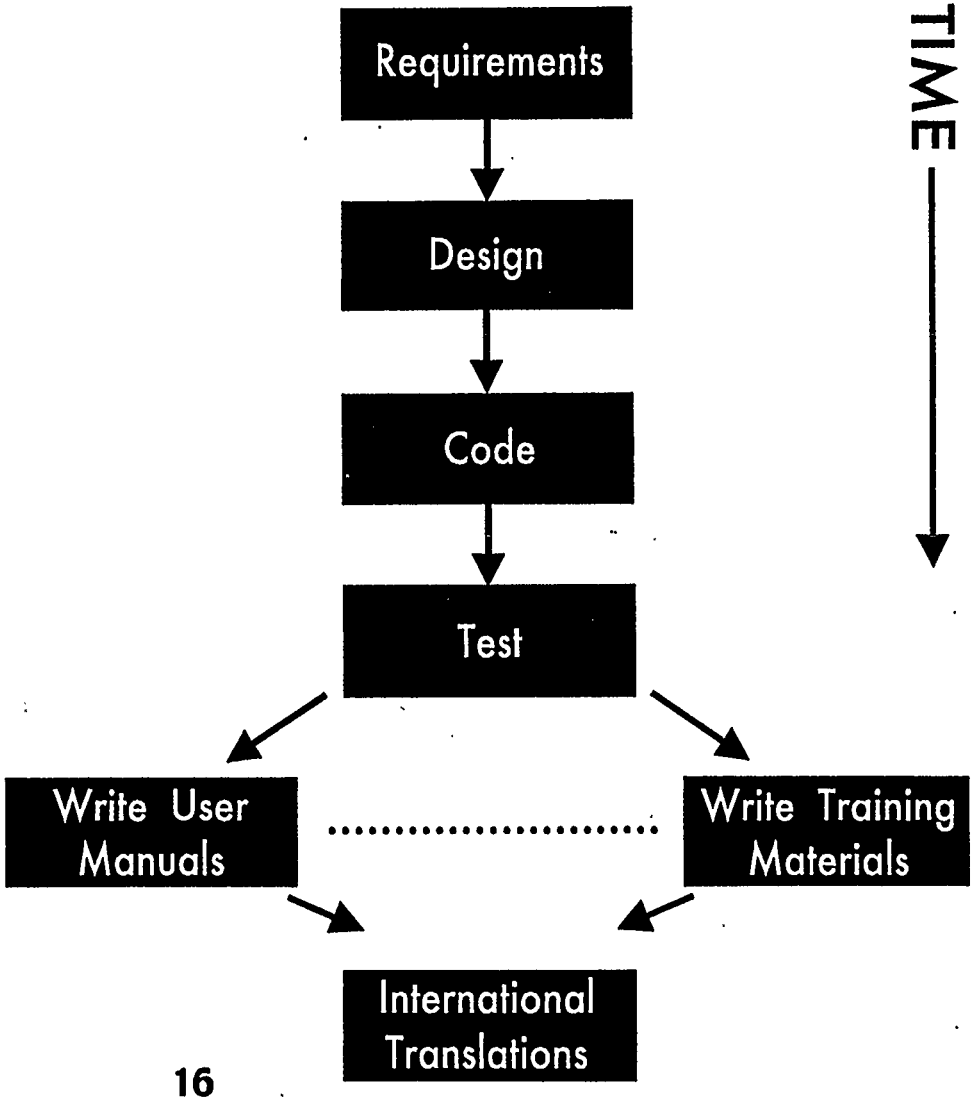
TBI

Test Strategy

- → Focus On Requirements–Based Testing
- → Supplement for Design / Code–Based Issues
- → Use Quantitative Completion Criteria

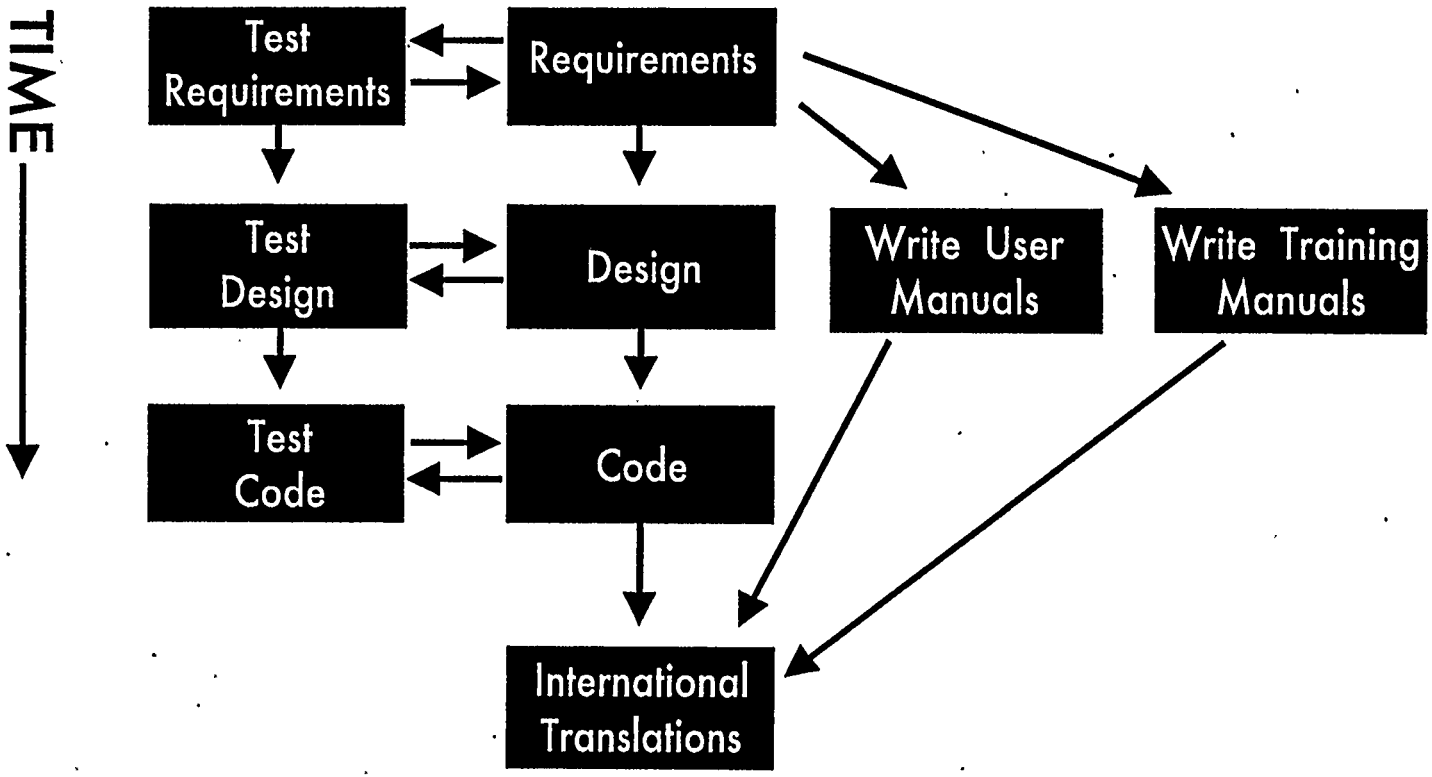
TBI

Standard Development Lifecycle



TBI

Lifecycle With Testable Requirements and Integrated Testing



TBI

Continuous Integration & Process Modeling

- Perform Ambiguity Reviews Of Each Process
- Design Test Cases For Each Individual Process
- String The Tests Together Into End-To-End Test Suites
- Sequence The Specification Writing To Allow Early Full Thread Test Definitions
- Sequence Design And Coding To Continue To Exploit Early Integration Opportunities

TBI

Continuous Integration & Process Modeling

- Identifies Inconsistent Interfaces
- Identifies Process Gaps / Redundancies
- If You Can Not Integrate The Specifications, How Will You Integrate The Code?

TBI

The Requirements-Based Testing Methodology

Quality Filters

1. Validate Requirements Against Objectives
2. Apply Use-Cases Against Requirements
3. Perform Initial Ambiguity Review
4. Perform Domain Expert Reviews
5. Create Cause-Effect Graph
6. Logical Consistency Check and Test Cases Designed By Caliber-RBT
7. Review Of Test Cases By Specification Writer
8. Review Of Test Cases By Users
9. Review Of Test Cases By Developers
10. Walk Test Cases Through Design
11. Walk Test Cases Through Code
12. Execute Test Cases Against Code

Characteristics Of Testable Requirements

1. Deterministic
2. Unambiguous
3. Correct
4. Complete
5. Non-Redundant
6. Lends Itself To Change Control
7. Traceable
8. Readable By Users / Customers, Etc.
9. Written In A Consistent Style
10. Processing Rules Reflect Consistent Standards
11. Explicit
12. Logically Consistent
13. Lends Itself To Re-Usability
14. Terse
15. Annotated For Criticality
16. Feasible

TBI

Definition Of Testable

→ Measurable / Deterministic Results

→ Given An Initial System State And A Set Of Inputs Can Predict Exactly What The Outputs Will Be

→ Testing:
Comparing An Expected Result To An Observed Result

TBI

Inputs to RBT

RBT Does Not Assume That Good Requirements Specifications Exist

Inputs:

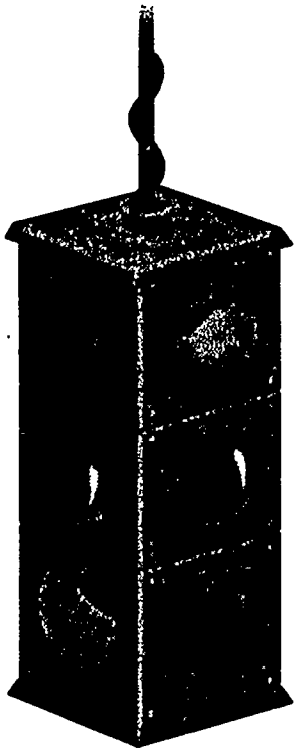
- High Level Requirements
- Somewhat "Detailed" Design Documents Written In "Technicaleze"
- Screen Prototypes
- Supplemented By Memos, E-mails, Conversations, Rumors

RBT Drives Down The Level Of Detail

TBI

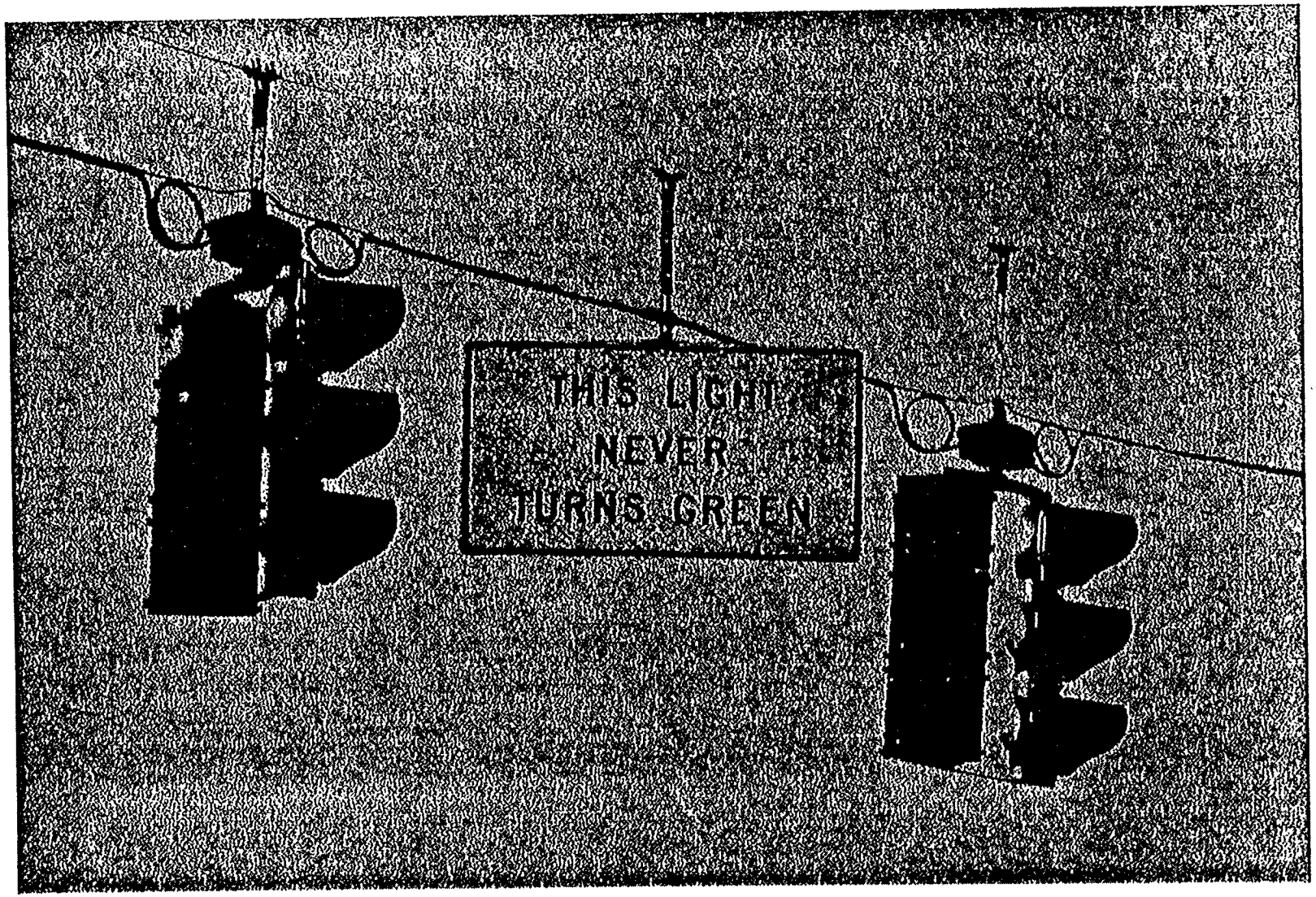
Testable Specifications

EXAMPLE:



If the light is red, then stop.

TBI



Photograph is © by **Car and Driver** Magazine

Ambiguous Specifications And Signoffs

A DIFFERENCE BETWEEN VERSION I AND VERSION II EXISTS ONLY WHEN MIXED DATA TYPES ARE USED, AND THEN ONLY WHEN OPERAND LENGTHS DIFFER, AND THEN ONLY SOMETIMES.

TBI

Ambiguity Review Checklist

- Dangling Else
- Ambiguity Of Reference
- Scope Of Action
- Omissions
  - Causes Without Effects
  - Missing Effects
  - Effects Without Causes
  - Complete Omissions
  - Missing Causes
- Ambiguous Logical Operators
  - Or, And, Nor, Nand
  - Implicit Connectors
  - Compound Operators
- Negation
  - Scope Of Negation
  - Unnecessary Negation
  - Double Negation
- Ambiguous Statements
  - Verbs, Adverbs, Adjectives
  - Variables, Unnecessary Aliases
- Random Organization
  - Mixed Causes And Effects
  - Random Case Sequence
- Built-In Assumptions
  - Functional/Environmental Knowledge
- Ambiguous Precedence Relationships
- Implicit Cases
- Etc.
- I.E. Versus E.G.
- Temporal Ambiguity
- Boundary Ambiguity

TBI

Dangling Else

MUST BE, WILL BE, IS ONE OF, SHOULD BE, COULD BE, CAN BE, SHALL.

EXAMPLE:

"The Marriage Status must be either Married, Single, or Divorced."

Else? An error condition?

UNIX 5.4 – Ambiguous Logical Operators

[EPERM]

nice will fail and not change the nice value if *incr* is negative or greater than 39 and the effective user ID of the calling process is not super-user.

TBI

UNIX 5.4 – Ambiguous Logical Operators

[EPERM]

nice will fail and not change the nice value if *incr* is (negative) or (greater than 39 and the effective user ID of the calling process is not super-user.)

TBI

UNIX 5.4 – Ambiguous Logical Operators

[EPERM]

nice will fail and not change the nice value if[*incr* is(negative)or (greater than 39]and[the effective user ID of the calling process is not super-user.)]

TBI

Savings Via Early Testing

Ambiguity Reviews of Specifications
(source: Bender & Associates)

- **Defects found**
  - **Mutual Funds project**
    - # Functions 180
    - # Issues 1,713
  - **Imbedded code project**
    - # Functions 65
    - # Issues 595

TBI

Savings Via Early Testing

Ambiguity Reviews of Specifications
(source: Bender & Associates)

- Costs per defects found
  - .85 hour/defect
  - $75 hour fully burdened rate ($150K year)
  - $63.75 per defect
- Costs if found in integration test/system test
  - $750 to $3,000 per defect (SEI)
- Cost if found in production
  - $10,000 per defect (HP)
  - $140,000 per defect (IBM)

TBI

Necessity For Deterministic Specifications

- Key To Time To Market, Cost To Market, Quality At Delivery
- Increasing Split Between Application Expertise And Technical Expertise
- Impact Of Staff Turnover
- Impact Of Multi-Platform Ports
- Impact Of Re-Engineering Projects
- Impact Of Off-The-Shelf
- Impact Of Outsourcing

TBI

Test Case Design Approaches

Goal:

Design a necessary and sufficient set of test cases to ensure system integrity.

- **Production files**
- **Gut feel**
- **Combinatorics machines**
- **Rigorous algorithms**

TBI

Testing With Production Files

- Covers Less Than 30% Of The Code
- Exception Cases Not Covered Since Data Is Already Scrubbed
- Time Dependent Functions Not Covered
- Expected Results Not Determined For Every Output Field
- Might Find Some Missing Cases
- Have Value In Performance Testing
- Have Value In Helping Build Test Cases

TBI

Testing By Gut Feel

Totally Dependent On Who Is Doing The Testing:

- **How Experienced They Are At Testing**
- **How Experienced They Are In The Application**
- **How Experienced They Are In The Technology That The Application Runs On**
- **How They Are Feeling Today**

Even If All The Tests Run Successfully, All You Know Is That *Those* Tests Run -- Not That The System Runs Successfully

TBI

Testing By Brute Force Combinatorics

Amount Of Time To Arrange 15 Books In Every Possible Way:

2,487,996 YEARS At One Change Per Minute

Length Of Paper Required To Write All The Possible Sequences Of The 26 Letters Of The Alphabet:

160 Million Light Years

QUESTION: How Many Variables Are In <u>Your</u> Application?

TBI

CALIBER RBT

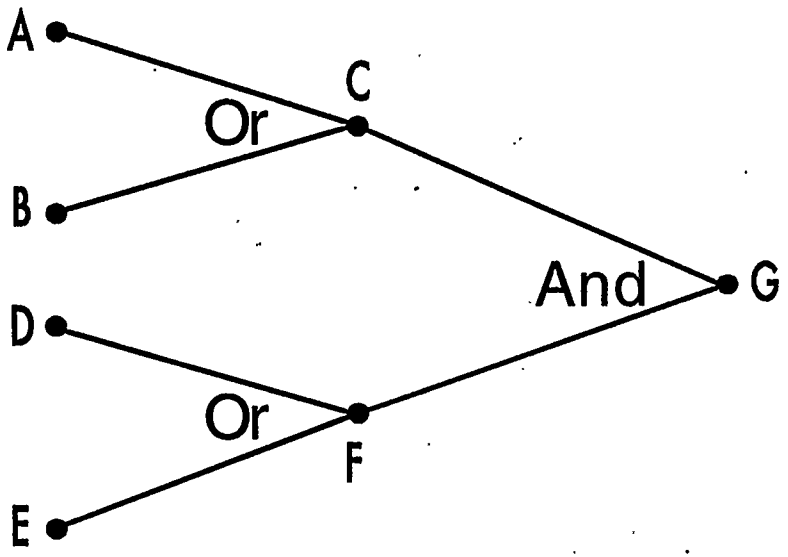
- Validates Functional Specifications
- Provides maximum coverage with the minimum number of tests
- Automates Test Case Design
- Uses a rigorous algorithm

TBI

Rigorous Testing

Cause-Effect Graphing

If A or B, Then C.
If D or E, Then F.
If C and F, Then G.

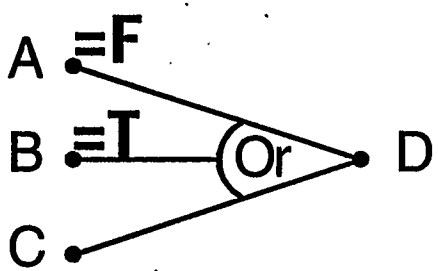


- **Resolve Aliases**
- **Clarify Precedence Rules**
- **Begin Integration Testing**

TBI

Rigorous Testing

Masking of Errors



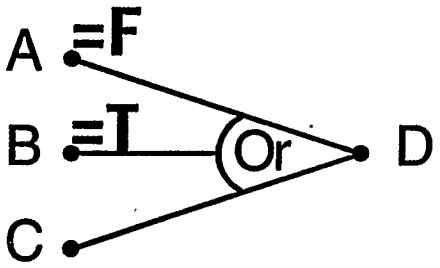
1.	A	—	—	D
2.	—	B	—	D
3.	—	—	C	D
4.	—	—	—	—

RBT24160

TBI

Rigorous Testing

Masking of Errors (continued)



1.	A	—	—	D
2.	—	B	—	D
3.	—	—	C	D
4.	—	—	—	—

Assume A is stuck at FALSE and B is stuck at TRUE.
The machine would interpret:

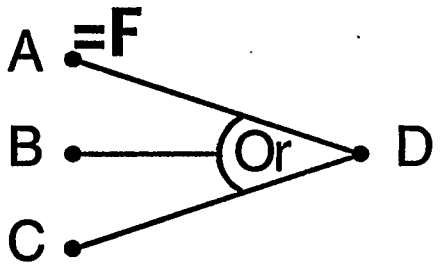
1.	A	—	—	as	—	B	—	D
2.	—	B	—	as	—	B	—	D
3.	—	—	C	as	—	B	C	D
~~4.~~	—	—	—	as	—	B	—	(D)

RBT24160

TBI

Rigorous Testing

Masking of Errors (continued)



1.	A	—	—	D
2.	—	B	—	D
3.	—	—	C	D
4.	—	—	—	—

Assume A is stuck at FALSE and B is stuck at TRUE.
The machine would interpret:

X.	A	—	—	as	—	—	—	⊖
2.	—	B	—	as	—	B	—	D
3.	—	—	C	as	—	—	C	D
4.	—	—	—	as	—	—	—	—

Fix the bug found by #4 and #1 fails.

Must rerun ALL tests until ALL pass!

RBT24160

TBI

Test Design Challenge

Challenge:

Design A Set Of Test Cases, Factoring In:

- The Relations Between The Variables
- Constraints Between The Data Attributes
- Functional Variations Required to Test
- Node Observability

... Such That If Any Logical Defect Or Any Combination Of Defects Are Present, At Least One Test Case Will Fail At An Observable Point.

Rigorous Testing

Required Functional Variations

No. of Inputs	No. of Required Functional Variations	No. of Possible Combination	No. of Possible Faults
n	n+1	2^n	$\sum_{x=1}^{n} \binom{n}{x} 2^x$
2	3	4	8
3	4	8	26
4	5	16	80
5	6	32	242
6	7	64	728

Note: $\sum_{x=1}^{n} \binom{n}{x} 2^x = 3^n - 1$

TBI

Designing Test Cases – An Example

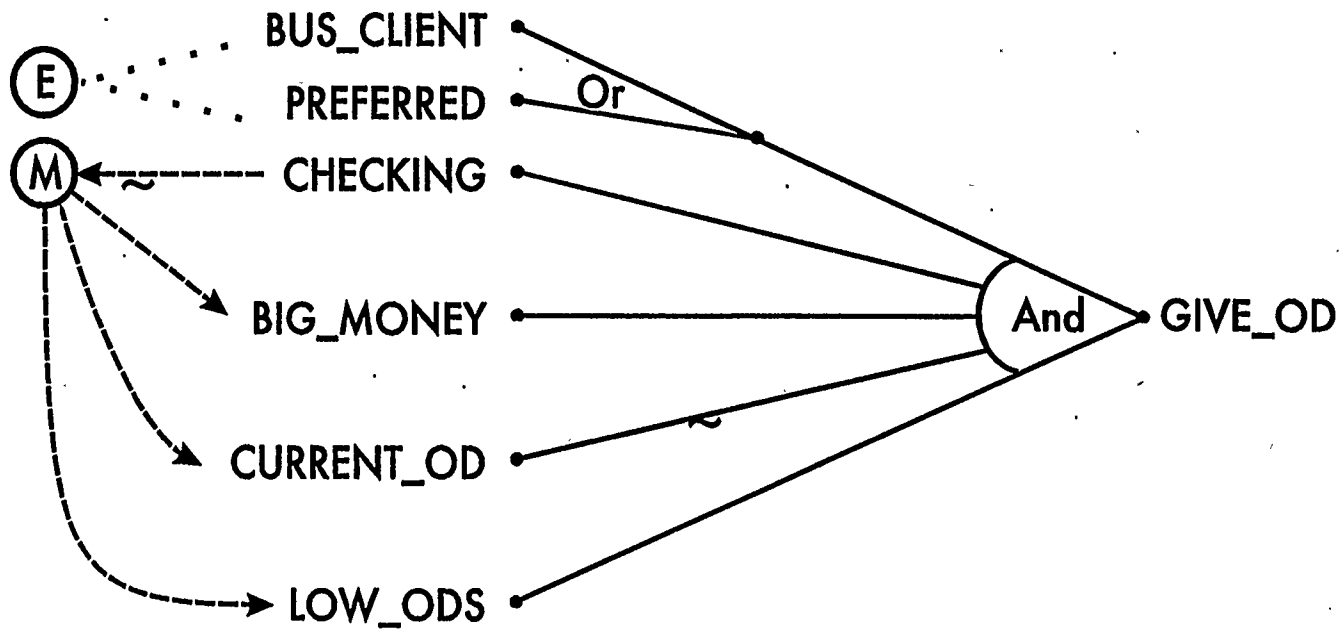
This Function Has Sixty-Four Possible Combinations Of Input From Which To Select Test Cases:

If the customer is a business client or a preferred personal client and they have a checking account, $100,000 or more in deposits, no overdraft protection and fewer than 5 overdrafts in the last 12 months, set up free overdraft protection. Else do not set up free overdraft protection.

How Many Test Cases Are Required To Confirm That The Function Works?

Answer: Seven

Cause-Effect Graphing



TBI

```
TITLE 'Automatic Check For Overdraft Protection'.

NODES
Bus_Client = 'The customer is a business client' | /b.
Preferred  = 'The customer is a preferred personal client' |/b.
Checking   = 'The customer has a checking account' |
             'The customer does not have a checking account'.
Big_Money  = 'The customer has $100,000 or more in deposits' |
             'The customer has less than $100,000 in deposits'.
Current_OD = 'The customer currently has overdraft protection' |
             'The customer does not have overdraft protection'.
Low_ODs    = 'Overdrawn less than five times in last 12 months'|
             'Overdrawn more than 4 times in last 12 months'.
Give_OD    = 'Set up free overdraft protection' |
             'Do not set up free overdraft protection'.

CONSTRAINTS
excl(Bus_Client, Preferred).
mask(not Checking, Big_Money, Current_OD, Low_ODs).

RELATIONS
Give_OD :- (Bus_Client or Preferred) and Checking and Big_Money
                  and not Current_OD and Low_ODs.
```

Tell *Caliber-RBT* what the inputs and outputs are, plus the function's logic.

TBI

TEST#1 -- Automatic Check For Overdraft Protection

Cause states:

The customer is a business client
The customer has a checking account
The customer has $100,000 or more in deposits
The customer does not have overdraft protection
Overdrawn less than five times in last 12 months

Effect states:

Set up free overdraft protection

Next, *Caliber-RBT* takes the FUNCTIONAL VARIATIONS and packages them into a complete set of TEST CASES.

TBI

Automatic Check For Overdraft Protection
STATISTICS

Synthesis of NEW Tests was specified.

There were 14 input statements processed.

There were 9 Functional Variations generated.
    There were NO Infeasible Variations
    There were NO Untestable Variations

There were 7 NEW Test Case definitions generated by Caliber-RBT.
    There were 9 Tested Variations, and
    9 FEASIBLE Variations, which yields
    100% Functional COVERAGE of FEASIBLE variations.

There were 9 Tested variations, and
    9 TESTABLE variations, which yields
    100% Functional Coverage of TESTABLE variations.

For n = 6 Primary causes, then
    2^n = 64. THEORETICAL Maximum Number of Test Cases.

Caliber-RBT generated 7 Test Cases, which yields a
    9 to 1 Test Case Compression Ratio.

Caliber-RBT processed 9 Testable Variations, which yields a
    1 to 1 Testable Variations to Test Case Compression Ratio.

Caliber-RBT Elapsed Time: 00:00:01 (hh:mm:ss)

Summary statistics are also produced to aid in project estimating and tracking.

TBI

Test Statistics – for a Large Problem

CHP_PG5_26/TOBACCO USE STATISTICS

Synthesis of NEW Tests was specified.

There were 112 input statements processed.

There were 141 Functional Variations generated.
    There were NO Infeasible Variations
    There was 1 Untestable Variation

There were 22 NEW Test Case definitions generated by Caliber-RBT.
    There were 140 Tested Variations, and
    141 FEASIBLE Variations, which yields
    99% Functional COVERAGE of FEASIBLE variations.

There were 140 Tested variations, and
    140 TESTABLE variations, which yields
    100% Functional Coverage of TESTABLE variations.

For n = 37 Primary causes, then
**   2^n = [a little more than] 137,438,953,472 THEORETICAL Maximum Number of Test Cases.**

Caliber-RBT generated 22 Test Cases, which yields a
**   6,247,225,157 to 1 Test Case Compression Ratio.**

Caliber-RBT processed 140 Testable Variations, which yields a
    6 to 1 Testable Variations to Test Case Compression Ratio.

Caliber-RBT Elapsed Time: 00:00:02 (hh:mm:ss)

TBI

Test Statistics



Thought Experiment

- Put 137,438,953,450 RED Balls In A Giant Barrel
- Add 22 GREEN Balls To The Barrel And Mix Well
- Turn Out The Lights
- Pull Out 22 Balls



What Is The Probability That You Have Selected The 22 Green Ones?

- Pull Out 1,000 Balls

What Is The Probability That You Have The 22 Green Ones Now?

- Pull Out 1,000,000 Balls

What Is The Probability That You Have The 22 Green Ones Now?

**** This Is What "GUT FEEL" Testing Really Is.****

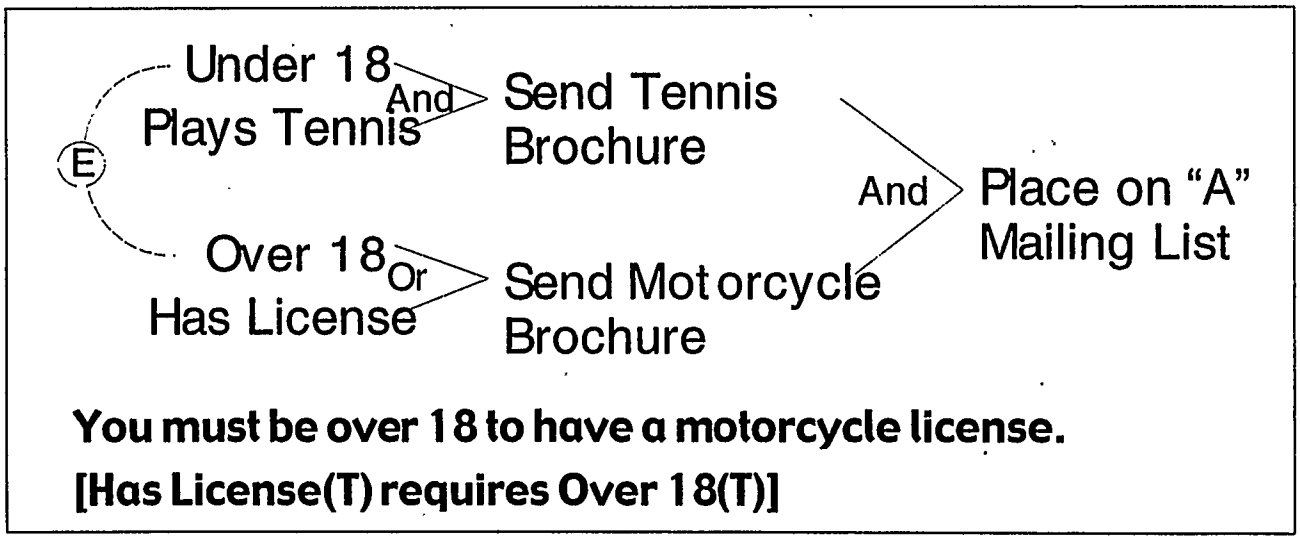
TBI

Caliber-RBT also aids in validating the consistency of FUNCTIONAL SPECIFICATIONS.

If the person is under 18, and plays tennis,
then send them a tennis club brochure.

If the person is 18 or older, or has a motorcycle license,
then send them a motorcycle club brochure.

If the person was sent both brochures, then put them on the "A" mailing list.



You must be over 18 to have a motorcycle license.
[Has License(T) requires Over 18(T)]

TBI

Caliber-RBT Identifies Logic Errors.

```
 Functional Variations for:
A_list:-T_brochure AND M_brochure.
<INFEASIBLE> T01--Due to constraint(s) ACROSS relationships (or
   faulty logic)
       7. If T_brochure and M_brochure
          then A_list.

       8. If not T_brochure
            (and M_brochure)
          then not A_list.

       9. If not M_brochure
            (and T_brochure)
          then not A_list.

T11--Possible graph logic error.  TRUE state of A_list always
   Infeasible
T12--Note:  TRUE state of A_list not covered in any test case
```


TBI

Would You Rather Validate A Specification That Looks Like This:

James Joycian Novel Style

Dental Insurance Claims Payment Specification

Dentists with membership codes of 2, 3, or 9 are member dentists. For claims referencing a non-member dentist or for procedures not within the referenced dentist's record, a system table is used to calculate the amount paid. Otherwise, the amount submitted is paid. However, an override code of 1 or 9 allows the amount submitted to be paid for non-member dentists or for procedures not within the referenced dentist's record. When an override code is used an entry is made on the paid claims report.

TBI

Or Validate Test Cases That Look Like This:

TEST 1

Cause States:

- The Dentist is a Member Dentist
- The procedure was not preauthorized
- An override code was entered

Effect States:

- This is a potential partial payment situation
- Override the partial payment
- Pay the full amount of the claim
- Make an entry on the paid claims report

TEST 2

Cause States:

- The Dentist is a Member Dentist
- The procedure was preauthorized

Effect States:

- It is a valid procedure for the member dentist
- Pay the full amount of the claim
- Do not make an entry on the paid claims report

TBI

Test Case Reviews

Test Cases More Readable Than The Original Specification

- Detailed Rules Often In "Technicaleze"
- Caliber-RBT Local Dictionary Cleans Up Terminology
- Set Of Tests Mathematically Equal To Original Specification(s)

Review Test Cases With:

- Specification Writer To Verify Correct Understanding Of Rules
- Users / Customers To Verify Completeness / Correctness
- Designers / Coders To Ensure Compliance

Spec Rewritten By *Caliber-RBT.*

Functional Requirements Specification for:
DENTAL PLAN -- CLAIMS PAYMENT RULES

1. IF Membership code = 2
   OR Membership code = 3
   OR Membership code = 9
   THEN Member dentist.

2. IF NOT Member dentist
   THEN Non-member dentist.

3. IF Member dentist
   AND The procedure is preauthorized
   THEN Valid procedure for member dentist.

4. IF Member dentist
   AND The procedure is not preauthorized
   THEN Invalid procedure for member dentist.

5. IF Non-member dentist
   OR Invalid procedure for member dentist
   THEN Potential partial payment.

6. IF Override code = 1
   OR Override code = 9
   THEN NOT No override code is entered
   ELSE No override code is entered.

7. IF Potential partial payment
   AND NOT No override code is entered
   THEN Override partial payment.

8. IF Valid procedure for member dentist
   OR Override partial payment
   THEN Pay full amount of claim.

9. IF Potential partial payment
   AND No override code is entered
   THEN Pay claim from system table.

10. IF Override partial payment
    THEN Make an entry on the paid claims report
    ELSE Do not make an entry on the paid claims report.

TBI

Tracking Project Status

Project =
(Time, Resources, Function, Quality)

- **Time – Calendar**
- **Resources – Spreadsheet**
- **Function – ???**
- **Quality – ???**

TBI

RBT Process
Test Case Design/Execution Status Report

Function Priority	Function Name	Spec Written	Ambiguity Review Complete	Spec Complete	Cause-Effect Graphs	Caliber-RBT Entry	Test Cases Designed	Test Cases Reviewed	# of Functional Variations	# of Test Cases	Modules Coded	Test Cases Built	Test Cases Executed	# of Defects Identified	# of Defects Open	% Code Coverage
1																
2																
3																
4																
5																
6																
7																
8																
9																
10																
11																
12																
13																
14																
15																
16																
17																
18																
19																
20																
	TOTALS															

TBI

Synergy With Other Tools

- Caliber-RM (Requirements Management)
  - Derivative links between requirements and test cases
  - Change notification
  - Functional coverage notification
- Capture / Playback Tools
  - RBT stabilizes functional definition and user interface allowing timely implementation of stable test scripts
  - Minimizes the number of scripts required
- Code Coverage Monitors
  - Industry average code coverage at test "completion" is under 50%
  - RBT results in 70% to 90% code coverage

TBI

Optimizing Test Automation

Test Case Design
- Done Once For A Given Version Of The System

Capture/Playback
- Scripts Built Once For A Given Version Of The System
- Effort Is Three To Five Times Effort To Design Test Cases
- Scripts Run Many Times

Test Results Validation
- Requires Expected Results To Be Known
- Done Every Time Test Cases Executed

Code Coverage Validation
- Done Periodically
- Finding That You Have Very Low Coverage At This Stage Leaves Few Options

Test Management
- The Smaller The Library The Easier It Is To Manage
- The Better The Coverage The More Effective It Is

Doing A Good Job On Test Case Design Pays Major Dividends In Every Testing Step That Follows

TBI

Management Considerations

Staffing Curve Peaks Earlier

- Requirements Written In More Detail
- Design Concurrent With Requirements
- Implementation Preparation Concurrent With Design
- Testers Involved From The Beginning
- Technical Writers Involved Earlier

Total Resources Reduced

- Minimize Scrap And Rework
- Plans Have Better Focus On Scope And Priorities

TBI

Contract Management

DELIVERABLES

- Contract Defines The Deliverables
- If Requirements Are Ambiguous The Deliverable Is Ambiguous
- Do Ambiguity Review Of Contract

QUALITY

"The Vendor Will Thoroughly Test The Deliverables"

OR

"The Vendor Will Test 100% Of All Functional Variations Fully Sensitized For Observability Of Defects And Test 100% Of All Code Statements And Branch Vectors."

Lawyers Are Like Coders – You Must Give Them Good Requirements To Get A Good Contract.

TBI

Summary
What RBT/Caliber-RBT Delivers:

- *Maximum* Coverage With *Minimum* Tests
  - 100% functional coverage
  - 70-90% code coverage
- Quantitative Test Progress Metrics
- Testing No Longer A Bottleneck
- Highly Portable Test Scripts
- Tests Any Application Written In Any Language Running On Any Machine

TBI

Results

- ✓ **Shortened Schedules Due to INCREASED PARALLEL EFFORTS**
- ✓ **Reduced Resources Due to MINIMIZING SCRAP & REWORK**
- ✓ **Improved Quality Due to REQUIREMENTS-BASED TESTING and INTEGRATED TESTING**

TBI

Are there any Questions?

TBI

Experiences Implementing Automated Regression Testing

Presented at

Quality Forum 2000

April 18-19, 2000

Shawn Dawson
Lawrence Livermore National Laboratory

Work performed under the auspices of the U.S. Department of Energy by
Lawrence Livermore National Laboratory under Contract W-7405-Eng-48



Introduction

Regression testing of portable, parallel programs presents challenges not inherent in the testing of sequential programs.



Challenge: Parallel programs must be portable

- Parallel programs must run on multiple machines
- The target parallel machine may not be the best development machine
- Parallel programs out-live parallel machines -- Programs will be ported to new parallel machines!



Challenge: Methods of starting parallel programs vary across machines

- How does the OS allocate CPU's to the program?
- Different rules for batch vs. interactive running of parallel programs.
- Mpirun is used on some SMP machines (DEC, Linux)
- Environment variables and psub are used on some MPP machines (IBM SP2)



Challenge: There are many run time parallel options

- Sequential or parallel
- Number of processors on which to run -- 1, 8, 1000, ...
- Method of decomposing the problem varies
- Manner of obtaining parallelism varies -- MPI, threads, or both



Challenge: What does a successful test mean?

- A single test may be run on 8 processors of an IBM SP2, using MPI, with the problem decomposed into 16 blocks.
- A change in the variables is a different test. The above test does not guarantee that the same problem will run on a DEC SMP, or if MPI and threads are used, or if run on 16 processors.



Hardware

- IBM SP2 – Massively Parallel
  - 1000 + CPUS : 320 nodes with 4 CPUS per node
  - Distributed Memory (across nodes) + Shared Memory (on-node)
  - Production, large scale parallel applications



Hardware (cont)

- SGI SMP – Parallel
  - 63 CPUS
  - Shared Memory
  - Testing, sequential and medium-scale parallel applications



Hardware (cont)

- DEC SMP – Parallel & Sequential
  - 10 CPUS
  - Shared Memory
  - Development, testing, debugging, sequential and medium-scale parallel applications



Hardware (cont)

- Linux SMP – Parallel & Sequential
  - 2 CPUS
  - Shared Memory
  - Development, testing, debugging, sequential and small-scale parallel applications



Hardware Picture

(Insert picture showing dif't hardware platforms here)



Software Parallelism Modes

MPI Only

IBM SP2, DEC SMP, SGI SMP, Linux SMP

Threads Only

DEC SMP, SGI SMP

MPI + Threads

IBM SP2, DEC SMP



MPI + Threads Mode

Necessary to access all the CPUS on the IBM SP2.

Threads across 4 CPUS per Node

MPI Between Nodes



MPI + Threads Picture

(insert pictures showing this type of parallelism here)



Testing Goal

On all platforms, developer should be able to test code as simply as

"test code"

Where 'test' is a script and 'code' is an executable

Should perform the complete suite of tests



Testing Goal (cont)

For a complete test – the 'test code' must run a suite of tests

scores of sequential runs

scores of parallel runs – due to many parallel run-time variances, many more parallel runs than sequential runs



File Compare : Verification Option 1

- Codes produce files at end of run which contains entire state of problem.
- Compare files



File Compare Pros

Thorough, the entire state of the problem is checked



DNT-00-0XXU. 18

File Compare Cons

- Dependent on file not changing -- any more or less data, or restructure of data produces mis-compare
- Only shows final problem state -- does not show when runs started to diverge.
- Time-step changes also cause file mis-compares.



Curve Compare : Verification Option 2

Codes produce 'curves' as they run

A curve is simply a plot of a variable (e.g. energy) vs. time

Each run can produce 100's of curves



Curve Compare Pros

- Does not depend on file format
- Time-step changes do not cause curve mis-compares



Curve Compare Cons

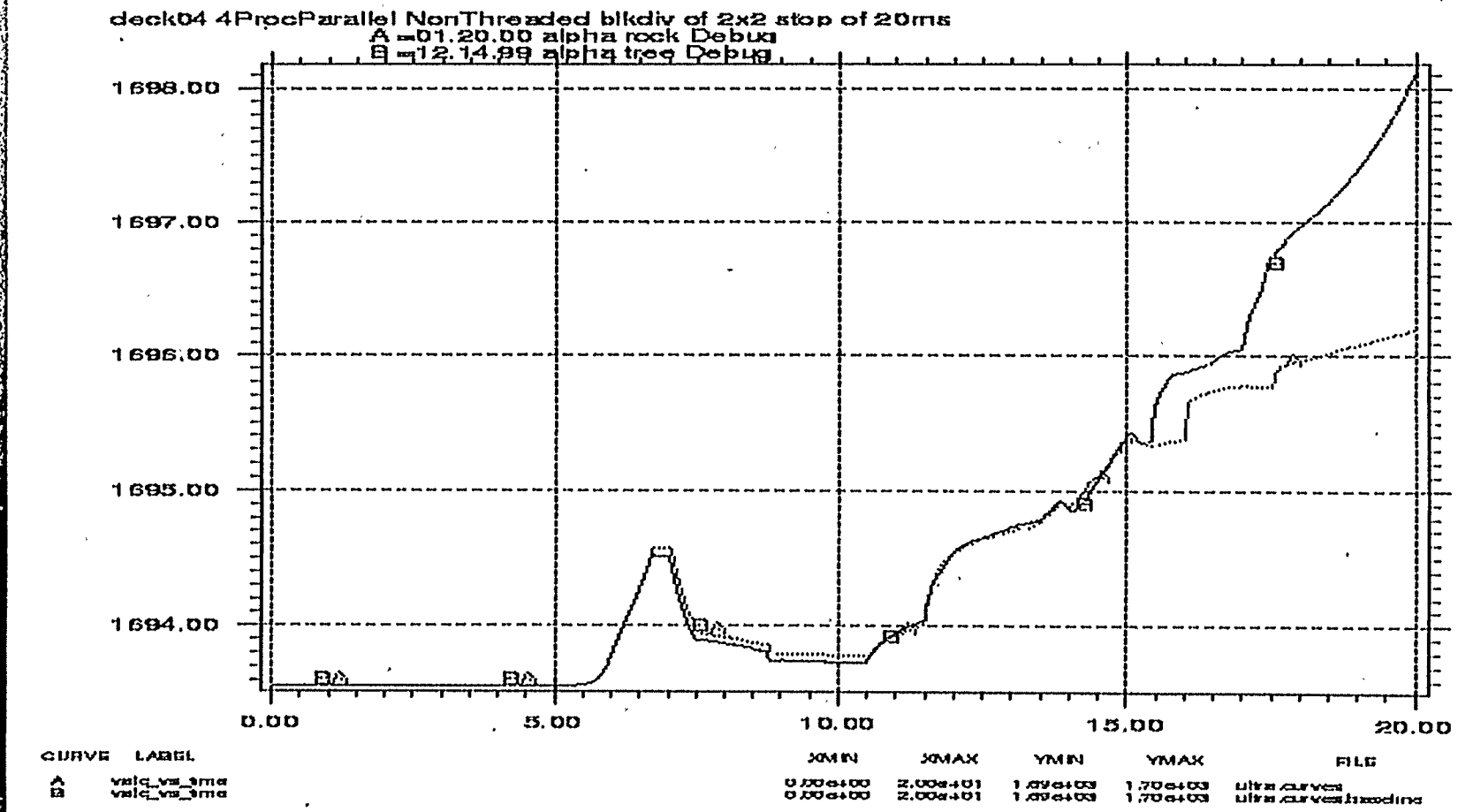
Codes produce 'curves' as they run

A curve is simply a plot of a variable (e.g. energy) vs. time

Each code run can produce 100's of curves.



Curve Compare



Verification Decisions

Default is Curve Compare

User can choose to do a file compare

File compare was due to users of the code having a strong desire for this feature in spite of its limitations.

File compare was user derived specification.



Verification Tools Chosen

Perl

Portable scripting language

SILO – file compares

Portable file format already used by our code

Ultra – curve compares

Portable graphic tool extendable for our needs using 'scheme' a lisp-like language



Implementation -- 2 Major Scripts

Two scripts were developed

'buildtest'

Used by small group of people to build the baseline regression test suite

'test'

used by many people to test a new version of the code. These are primarily programmers



Buildtest Script

buildtest code deck1 deck2 deck3 ... deckn

Runs 'code' n times, once for each deck

Stores all resultant files in a unique directory--one for each deck



Buildtest (cont)

- Need for options for buildtest was quickly apparent
- Options let a single deck be run in several ways
- Number of decks to maintain is dramatically reduced



Buildtest Options

-batch : Needed to submit run to batch system, which allows access to more processors and longer run times

-tm : Used to override default time limit for run. Needed both prevent runaway jobs and to allow long jobs



Buildtest Options (cont)

- -force : Used to run parallel code on a non-parallel file system. Discouraged, but sometime necessary.
- -blkdiv : Used to specify different domain decompositions - e.g. -blkdiv 4x4 results in 16 domains which can be divided among the parallel processors.



Buildtest Options (cont)

-np : Used to specify how many processors on which to run in parallel – e.g. -np 8 specifies a run on 8 processors

-threads : Default is MPI or Sequential. This turns on threading in the code

-tstop : Specify a stop time

-cstop : Specify a stop cycle



Buildtest Options (cont)

-tv: Code can run with a graphical window. Normally disabled during testing, but may be turned on so user can monitor test run.



Buildtest Options Examples

buildtest code deck1

buildtest -blkdiv 16 -np 16 code deck1

buildtest -blkdiv 16 -np 4 -threads code deck1



Test Script

test code

Tests code on ALL tests created by buildtest

test code deck1 deck2 ... deckn

Tests code on a subset of the test cases



Test Script Options

-batch, -tm -force

same use as in 'buildtest' script

-np

only necessary on IBM SP2 – discussion on this later



Test Script Options (cont)

-silodiff

Perform file comparsion rather than default curve comparison

-ultradiff

If specified with -silodiff, then perform both types of compares



Test Script Options (cont)

-suite

We found it useful to break up the scores of test cases into suites.

1 is short suite -- finished in under 2 hours

2 is medium suite -- overnight run

3 is complete suite -- weekend run



Problem Generation Options

- Sequential or Parallel generation
  - It is common for a parallel run to start from a file that was generated sequentially
- Sequential and Parallel generation of problem exercise different code paths
- Challenging on IBM SP2 – more later



IBM SP2 Challenge – Node allocation

Nodes must be allocated before running the code.

Once started, a program must use all the allocated notes

Submitted job may wait in queue a long time before starting



IBM SP2 Challenge - Multiple runs

Each test run is fairly short, so scripts must run all the tests they can once they get the nodes.

Complete test suite cannot be run in one shot.

Best that can be done is to run all 1 node tests, then all 4 node tests, etc.



IBM SP2 Challenge – Parallel Perl

Code itself is not submitted, rather perl script is submitted

Parallel Perl doesn't exist on IBM SP2

Work around using creation and deletion of files by parallel Perl scripts

Work around doesn't scale and sometimes fails.



IBM SP2 Challenge – Sequential Generation

Allocating 4 CPU's and running on only 1 will hang the system.

Generating sequentially and then running in parallel didn't work.

Code was modified to generate sequentially even when run in parallel

In this mode, one code runs and all the other codes just wait around.



IBM SP2 Challenge – Parallel file compare

If file compare test is being done, then code must be run sequentially at end of run also.

Parallel run results in multiple files. Code must be run sequentially to merge multiple files before SILO utility can compare

Same code modification mentioned above



Regression Test Reports

The scripts produce 4 reports

A summary and a detailed report for silo file compares

A summary and a detailed report for curve compares



File Compare Summary

XXXX-001 S I L O F I L E R E G R E S S I O N T E S T S U M M A R Y XXXX-001

Fri Jan 28 13:50:42 PST 2000

TESTED CODE VERSION : rock Debug 01.20.00
REGRESSION TEST SUITE : ./Results

Baselined Code	Deck	Parallel Mode	Thread Mode	Blkdiv	Stop	Result
tree Debug 12.14.99	deck01	Sequential	NonThreaded	1	2.5ms	Silo files have differences!
tree Debug 12.14.99	deck02	Sequential	NonThreaded	1	0.8ms	Silo files have differences!
tree Debug 12.14.99	deck03	Sequential	NonThreaded	1	70cy	Silo files have differences!
tree Debug 12.14.99	deck04	4ProcParallel	NonThreaded	2x2	20ms	Silo files compared correctly

* * * E N D O F R E P O R T * * *



File Compare Detail

XXXX-002 SILO FILE REGRESSION TEST DETAIL XXXX-002

Fri Jan 28 13:50:42 PST 2000

TESTED CODE VERSION : rock Debug 01.20.00
REGRESSION TEST SUITE : ./Results

Baselined Code	Deck	Parallel Mode	Thread Mode	Blkdiv	Stop	Result
tree Debug 12.14.99	deck01	Sequential	NonThreaded	1	2.5ms	Silo files have differences!

Variables that did not compare

parms[0].totaltime
parms[0].global

tree Debug 12.14.99	deck01	Sequential	NonThreaded	1	2.5ms	Silo files have differences!

Variables that did not compare

domain1.x[55]



Curve Compare Summary

```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+----0--
XXXX-003  U L T R A   E D I T   C U R V E S   R E G R E S S I O N   T E S T   S U M M A R Y   XXXX-003

Fri Jan 28 13:50:42 PST 2000

TESTED CODE VERSION   :  rock Debug 01.20.00
REGRESSION TEST SUITE :  ./Results

   Baselined Code    Deck      Parallel Mode    Thread Mode   Blkdiv   Stop          Result
---+----+----+----+  +----+  --+----+----+  ----+----+--  ---+--  ----+-  ---+----+----+----+----+----+-
tree Debug 12.14.99  deck01      Sequential  NonThreaded        1  2.5ms          4 Curve Differences
tree Debug 12.14.99  deck02      Sequential  NonThreaded        1  0.8ms          3 Curve Differences
tree Debug 12.14.99  deck03      Sequential  NonThreaded        1  70cy           3 Curve Differences
tree Debug 12.14.99  deck04  4ProcParallel  NonThreaded      2x2  20ms          21 Curve Differences

                        * * *  E N D   O F   R E P O R T  * * *
```



Curve Compare Detail

XXXX-004 ULTRA EDIT CURVES REGRESSION TEST DETAIL XXXX-004

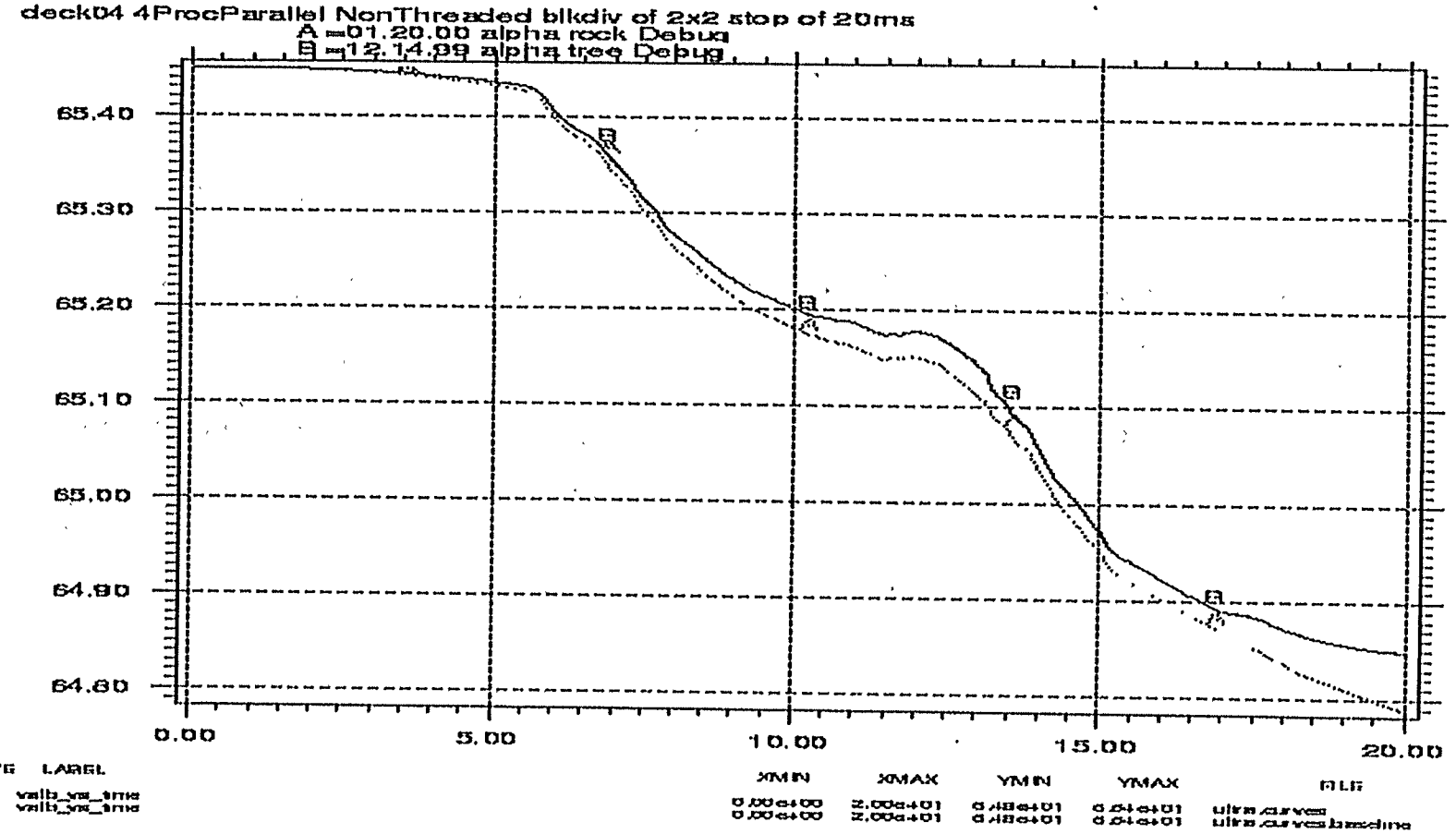
Fri Jan 28 13:50:42 PST 2000

TESTED CODE VERSION : rock Debug 01.20.00
REGRESSION TEST SUITE : ./Results

Baselined Code	Deck	Parallel Mode	Thread Mode	Blkdiv	Stop		Result
tree Debug 12.14.99	deck01	Sequential	NonThreaded	1	2.5ms	vala	1.52395e-05
						valb	1.1755e-05
						valc	9.07422e-06
tree Debug 12.14.99	deck02	Sequential	NonThreaded	1	0.8ms	vala	1.01955e-05
						valb	7.03955e-06
						valc	4.33065e-06
tree Debug 12.14.99	deck03	Sequential	NonThreaded	1	70cy	valx	0.00736064
						valy	0.00683893
						valz	0.000581366
tree Debug 12.14.99	deck04	4ProcParallel	NonThreaded	2x2	20ms	vala	8.9295
						valf	1.93315
						valg	0.0607789

*** END OF REPORT ***

Another Curve Compare



Successes

Goal of automated testing met

Goal of one test method across all platforms met



Side Benefits – Cross Platform Compares

IBM SP2 results are tarred up and dropped onto a DEC SMP

Tests are run on the DEC and the results compared to the IBM results



Side Benefits - Compiler Compares

buildtest generates baseline results with IBM compiler

test run with KAI compiler

Can also test different compiler options



Obstacles not overcome

Large parallel runs not automated due to parallel script limitations

Don't see fix for this in near future without parallel scirpt support

IBM SP2 takes multiple test submittals to run complete suite

Could a front end – script take care of this?



Summary

Testing is more automated than ever before

Is being used by programmers

Has found many errors before code was put into production

Has become a part of the development cycle for our code.



Object Oriented Software Project Management

A Practical Approach to Managing Object Oriented Software Projects

Software Quality Forum

April 18, 2000

Greg A. Schottland

President

Advanced Software Technologies,Inc.

© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.

http:/www.advancedsw.com

Discussion Overview

- Goals of Discussion
- Objectives of Project Management
- Pieces of the Puzzle
- Choosing a Process
- Mechanics - Tools of Project Management
- How to Keep Things On Track

© 2000 Advanced Software Technologies, Inc.

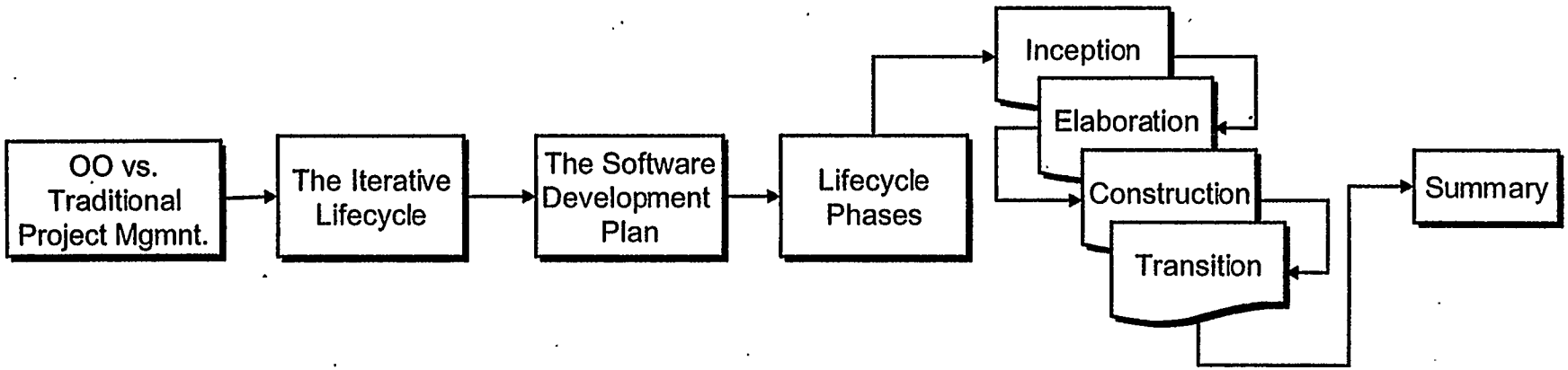
Goals of Discussion

"To outline the essential process of managing object oriented software projects"

- Issues
- Processes
- Challenges
- Guides

© 2000 Advanced Software Technologies, Inc.

Talk Overview



© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
-http:/www.advancedsw.com

Acknowledgements

The author would like to acknowledge Murray Cantor of TASC for his clear, concise and highly practical approach to project management expressed in his book ***Object Oriented Project Management with UML***. This presentation has been modified to more closely conform to the naming conventions and terms presented by Mr. Cantor in his book. I highly recommend this text for anyone involved in software management.

Object Oriented Project Management with UML
Murray R. Cantor
John Wiley and Sons 1998
ISBN: 0-471-25303-0

© 2000 Advanced Software Technologies, Inc.

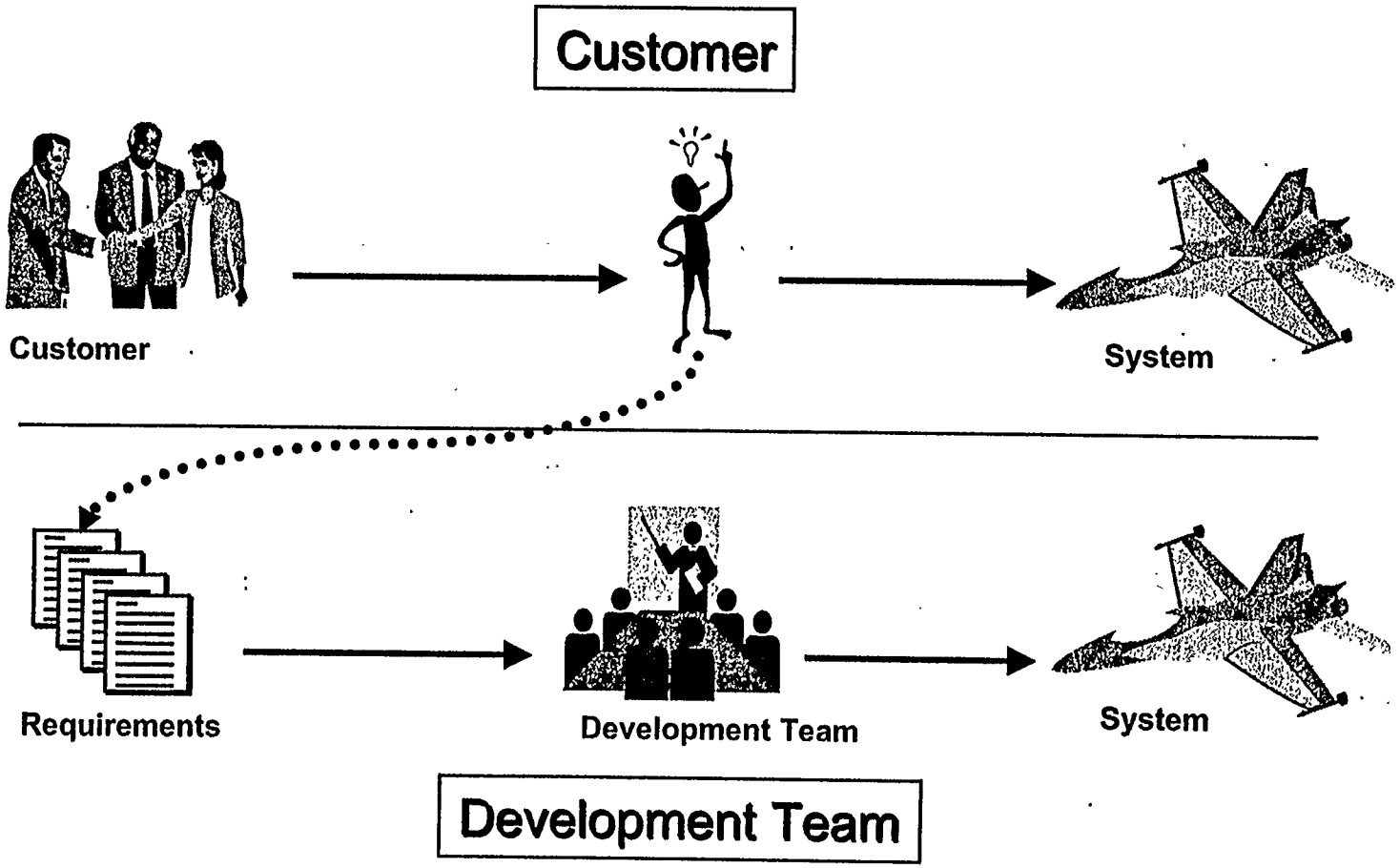
What is Project Management?

Effective project managers provide:

- A **plan** of attack for producing a product
- A **process** to guide the work
- **Tools** to get the work done
- An effective, coordinated **team**
- **Leadership**

© 2000 Advanced Software Technologies, Inc.

Pieces of the Puzzle



© 2000 Advanced Software Technologies, Inc.

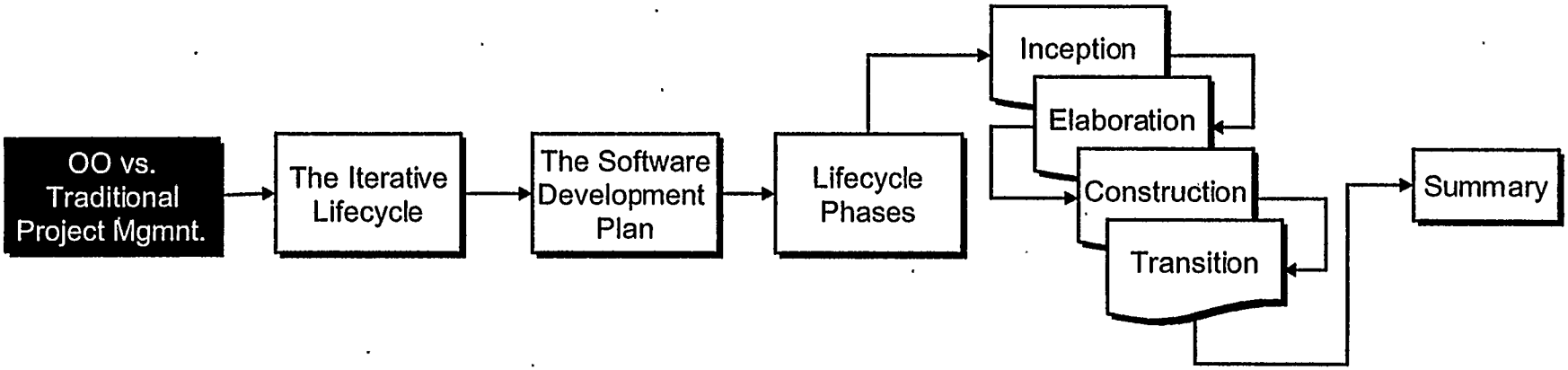
ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Pieces of the Puzzle

- Accurately **capturing user requirements**
- **Evaluating** various **alternatives** to realize requirements
- **Choosing** a particular **design** to implement the product
- **Implementing** the design
- **Testing** and **Delivery** activities

© 2000 Advanced Software Technologies, Inc.

OO vs. Traditional Project Management



© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

OO vs. "Traditional" Project Management

- Many of the overall principles are the same
- Differences are found mainly in process and construction phase
- Evolution of general project management principles over time have been incorporated into OO project management

© 2000 Advanced Software Technologies, Inc.

OO vs. Traditional S/W Project Management

Activity	OO	Traditional
Requirements	Use Cases	Textual descriptions/documents
Process	Iterative	Waterfall
Design Notation	UML	Structured Analysis and Design
Integration	Incremental	All at once at project end
Focus	Process & Activities	Documents

© 2000 Advanced Software Technologies, Inc.

Moving Away From A "Document Centric" Process

Traditional

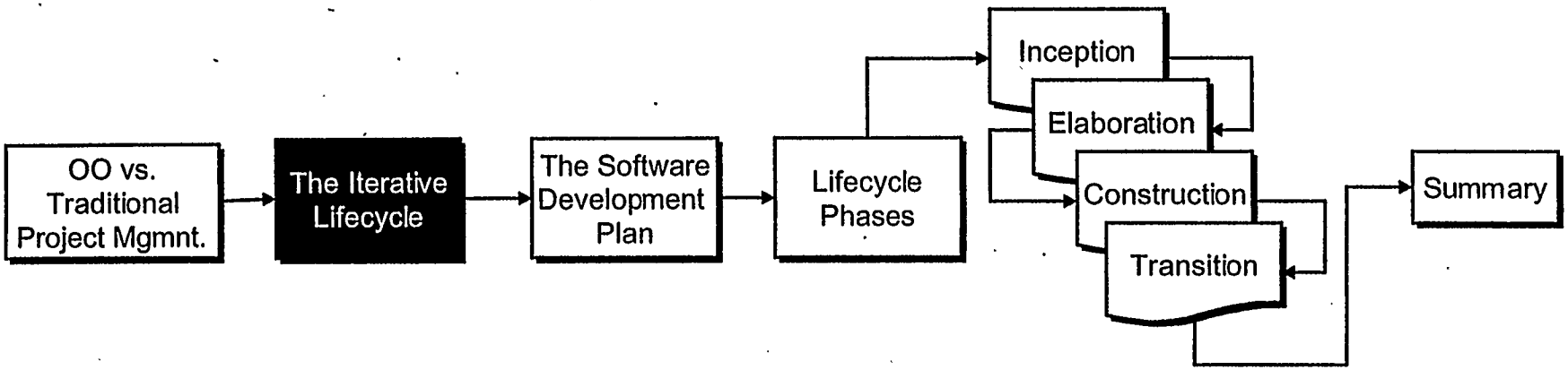
- Document completion marks phase completion
- Documents are highly static
- Documents are quickly outdated

Object Oriented (Iterative)

- Documents should be mostly online and active
- Documented activities change and can be versioned
- Documentation remains up to date

© 2000 Advanced Software Technologies, Inc.

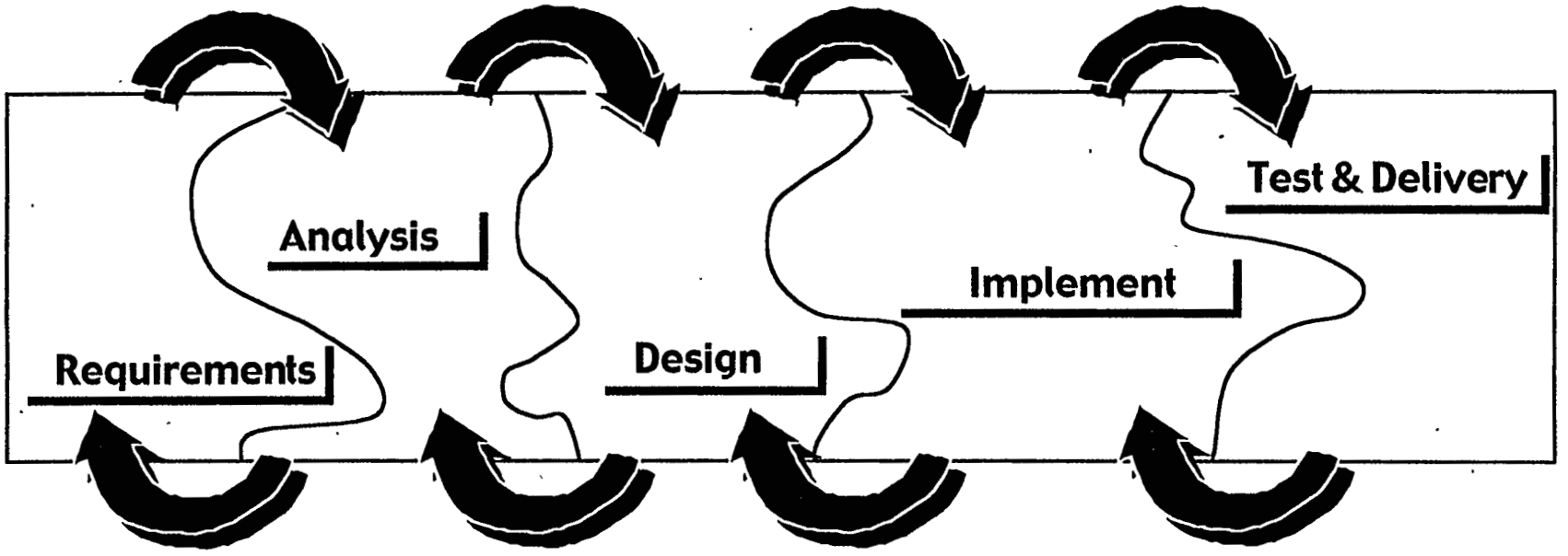
Overview of Iterative Lifecycle



© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Lifecycle Phases

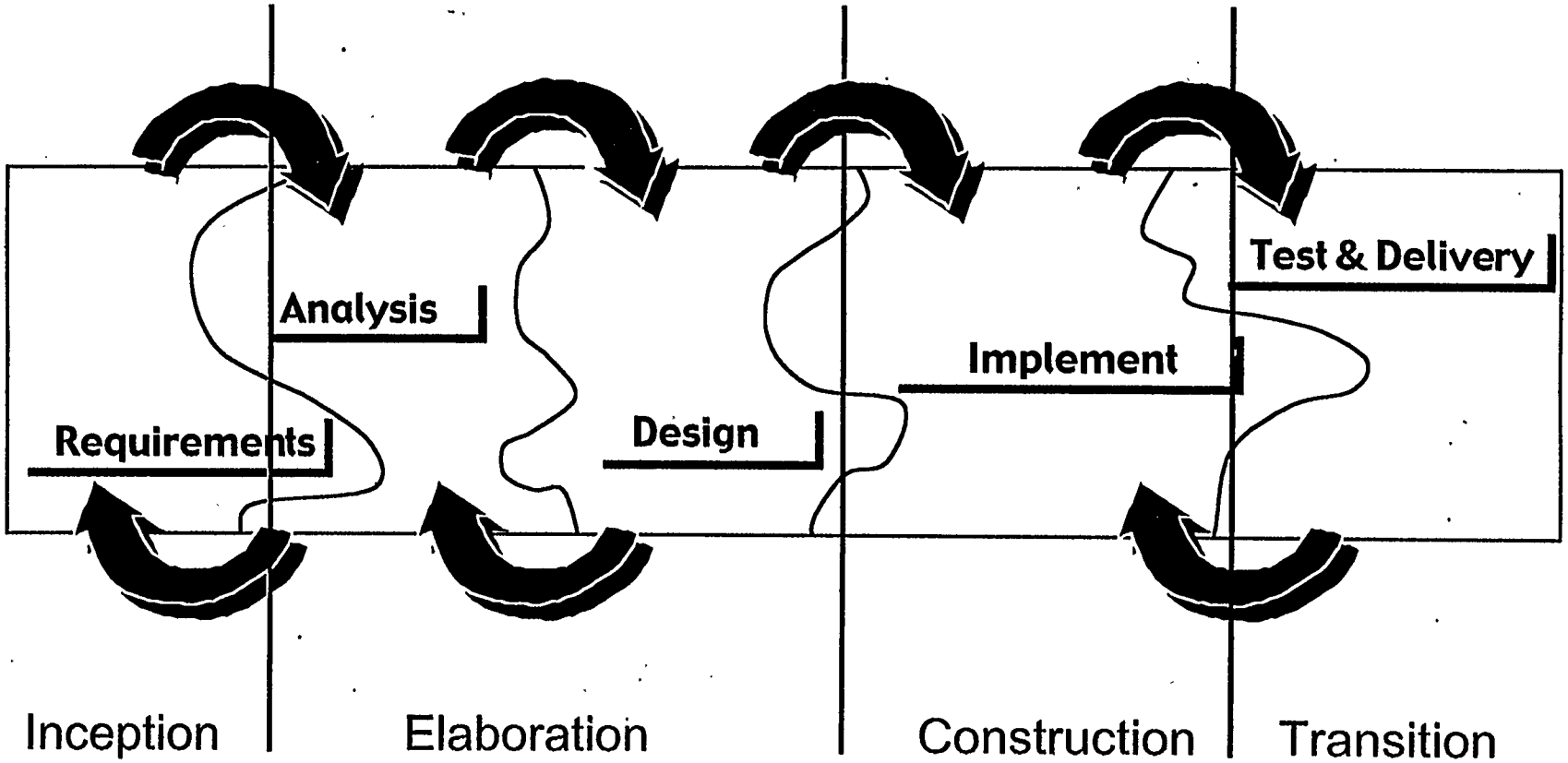


© 2000 Advanced Software Technologies, Inc.

ADVANCED
SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Lifecycle Phases

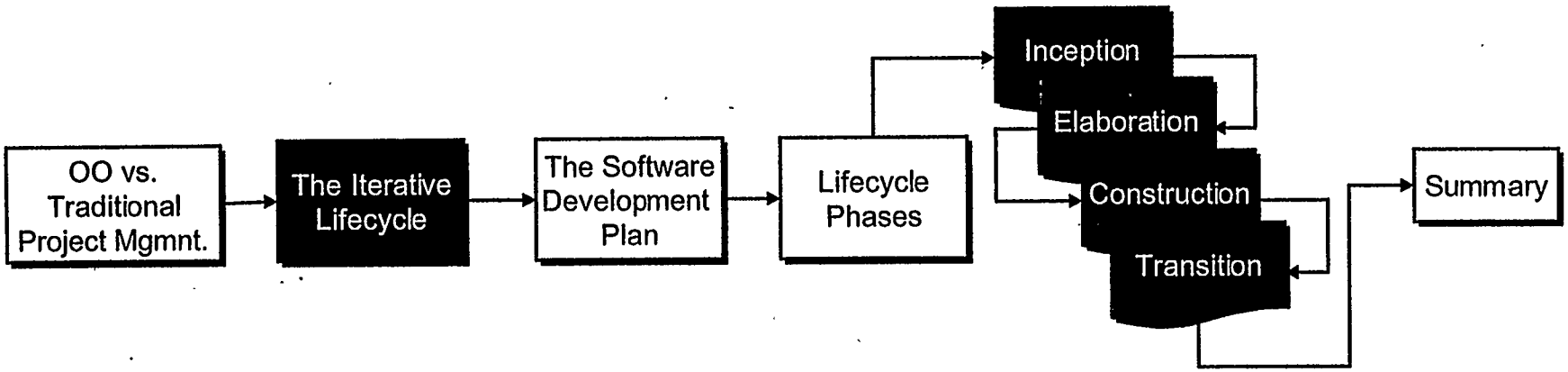
The "Iterative Lifecycle" Approach



© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

The Iterative Lifecycle



© 2000 Advanced Software Technologies, Inc.

Notes About The Iterative Lifecycle

- Phases need not be "complete" before transition to next phase occurs
  - Information from later phases is needed to "complete" earlier phases
- Changes to prior phases are allowed and anticipated but must be clearly managed
- The process is "real world" and should **support you** and not vice versa

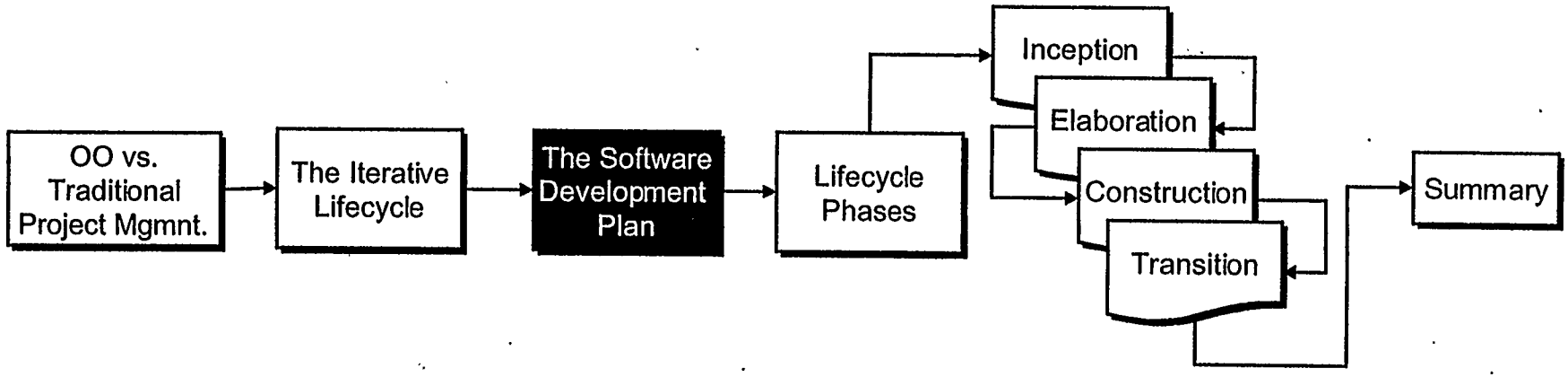
© 2000 Advanced Software Technologies, Inc.

Notes About The Iterative Lifecycle (cont.)

- The process is flexible enough so that it should never be skipped even when late stage changes occur
  - Generally speaking changes to an earlier phase should be limited to the immediately preceding phase
  - Occasionally changes will need to be made to earlier phases and this can occur
  - Whenever an "upstream" change occurs, you must follow it through downstream activities and update documents (This can done on a "fastlane" timeline if required)

© 2000 Advanced Software Technologies, Inc.

The Software Development Plan



© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Getting Started
The Software Development Plan

Creating the plan is important due to the process, activities and discussions it will generate. It is only secondarily important as a static document

- Don't focus on formality, focus on content
- Remember, the SDP is a "plan of attack"
- Include all team members in creating plan
- Get 100% buy in and commitment from the team & customer

© 2000 Advanced Software Technologies, Inc.

What's in the SDP?

Focus on content, not form!

- Deliverables
- Process
- Team Roles and Responsibilities
- Schedule & Milestones
- Tools and Technical Resources
- Budget
- Measurements & Feedback Mechanism

© 2000 Advanced Software Technologies, Inc.

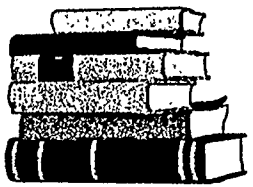
The SDP: Deliverables

*What is the **total package** we will deliver ?*

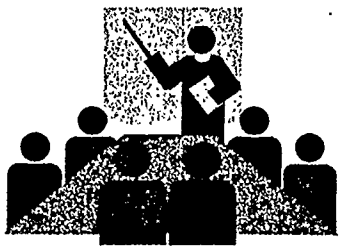
- The System



- Documentation
  - internal
  - external



- Follow-up Training or Support
  - internal
  - external



© 2000 Advanced Software Technologies, Inc.

The SDP: Process

*Focus on **your** implementation of the process*

Outline the process you intend to implement and follow. Include:

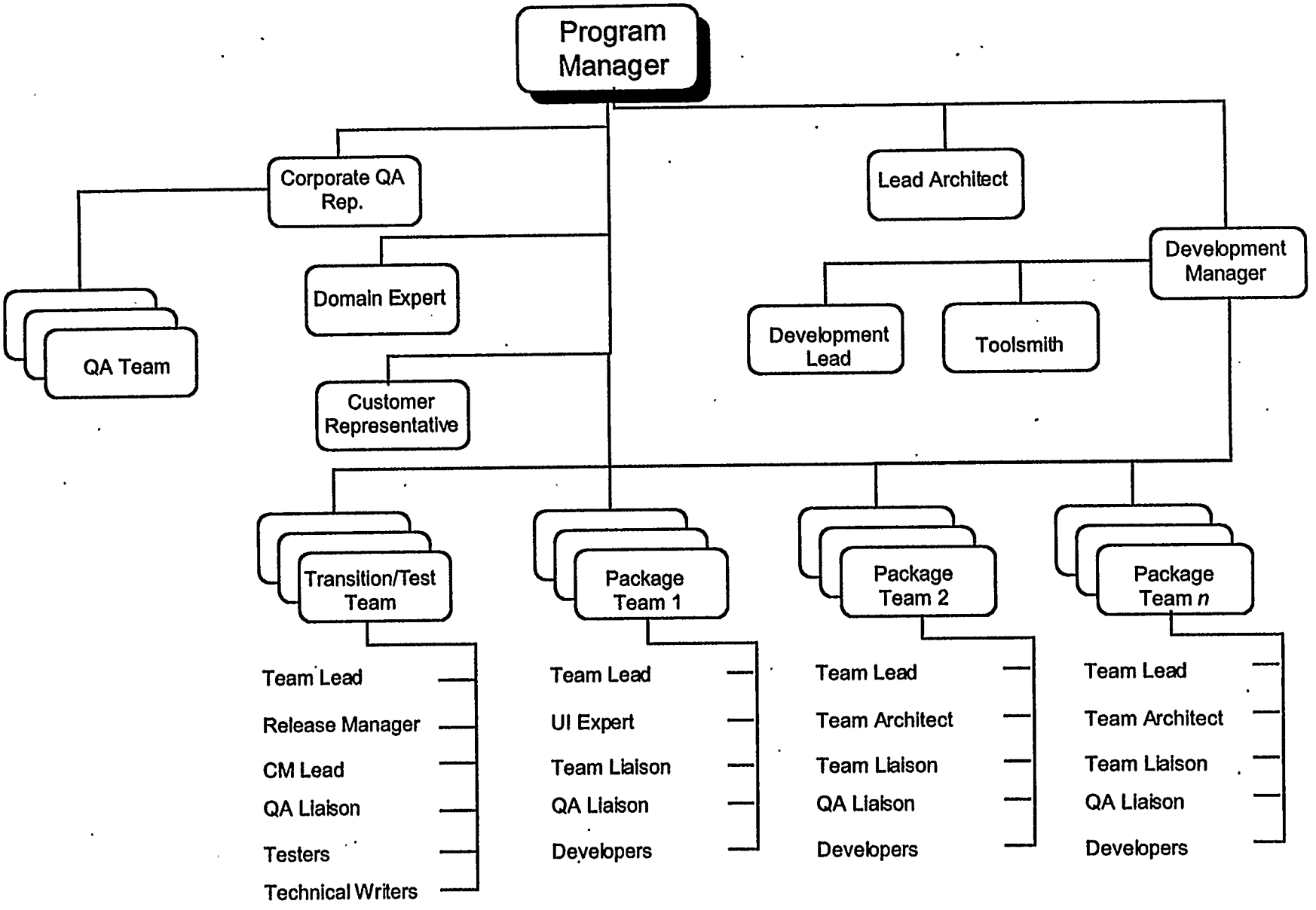
- What are the high level phases?
- What are the "entry" and "exit" criteria for each phase?
- What are the milestones?
- What are your tracking and guide mechanisms?

© 2000 Advanced Software Technologies, Inc.

The SDP: Team Roles and Responsibilities

- Outline roles and responsibilities separate from individuals
- Identify individuals who will fill each role
- Results in a matrix - individuals fill several roles
- Be sure that roles and responsibilities are clearly understood

© 2000 Advanced Software Technologies, Inc.



Source:Object-Oriented Project Management with UML--A program organization chart.

© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

The SDP: Schedule & Milestones

- Act as guides and frame the work
  - Expect them to change
  - Keep them up-to-date
- Manage the change
- Schedules should be realistic
  - Should plan 40 hour work week
  - Should have slack to account for team and project "overhead"

© 2000 Advanced Software Technologies, Inc.

The SDP: Schedule & Milestones (cont.)

- Milestones are imperative
  - Focus team on short-term goals
  - Give team feeling of accomplishment
  - Help you track and manage progress
- Take your schedules seriously
  - You should regularly hit your milestones and dates
  - Avoid "building the perfect beast"
  - Be sure your team is fully committed to its achievement
  - Recognize your team at each milestone

© 2000 Advanced Software Technologies, Inc.

The SDP: Tools and Technical Resources

- Spend time in this critical area
- Development platform / porting issues
- Network setup and maintenance
- Tools
  - Design and Process Management Tools
  - Development and Debugging Tools
  - Configuration Management Tools
  - Requirements Management Tools
  - Testing Tools
  - Documentation Tools

© 2000 Advanced Software Technologies, Inc.

Tool Considerations

- Evaluate all tools prior to purchase
- Test drive tools in your real development environment
- Dry-run a small example process using the tools
- Tool selection is critical and takes time

© 2000 Advanced Software Technologies, Inc.

The SDP: Budget

- Derive budget from schedule, staffing and tools considerations
- Budget process initiates iteration in planning to accommodate budget constraints
  - **First cut plan often exceeds budget**
  - **Trade off project scope or resources**
- Be realistic at all costs
  - **Tell the hard news upfront, not later**

© 2000 Advanced Software Technologies, Inc.

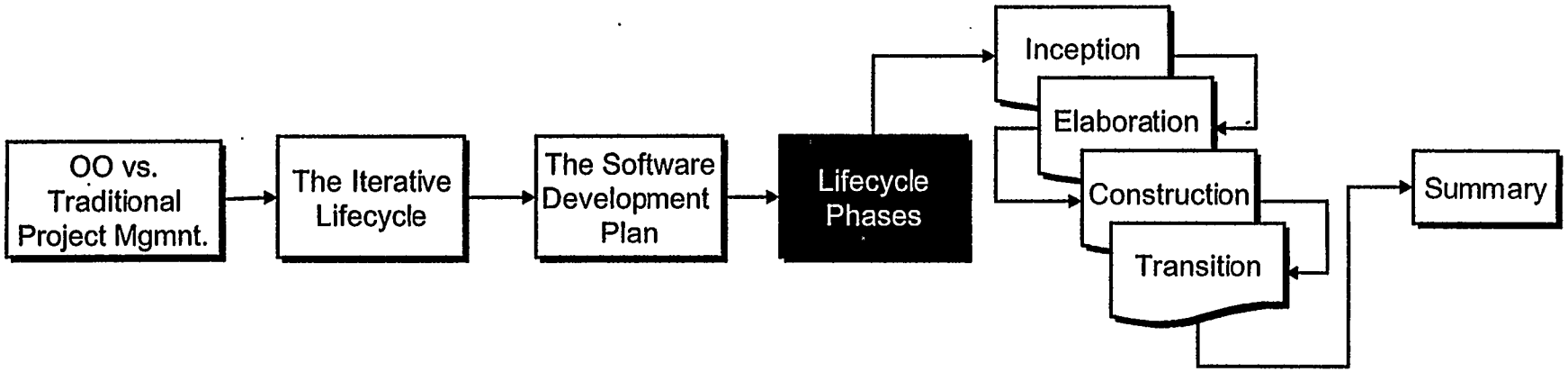
ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

The SDP: Measurements & Feedback Mechanism

- Customer Reviews
  - Get customer approval and buy-in at critical milestones
  - Avoids or minimizes late-game changes
- Schedule Reviews
- Budget Reviews
- Process Phase Reviews
  - Requirements, Design, Code, Test, Release

© 2000 Advanced Software Technologies, Inc.

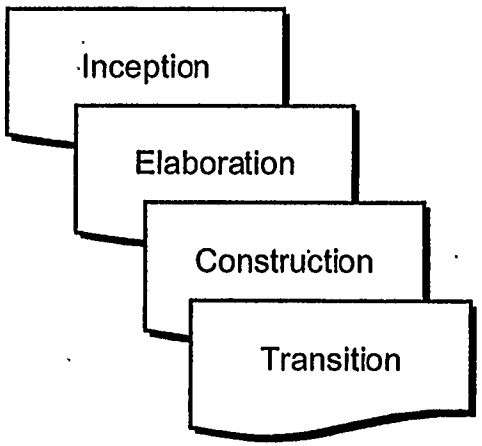
Lifecycle Phases



© 2000 Advanced Software Technologies, Inc.

Managing the Process

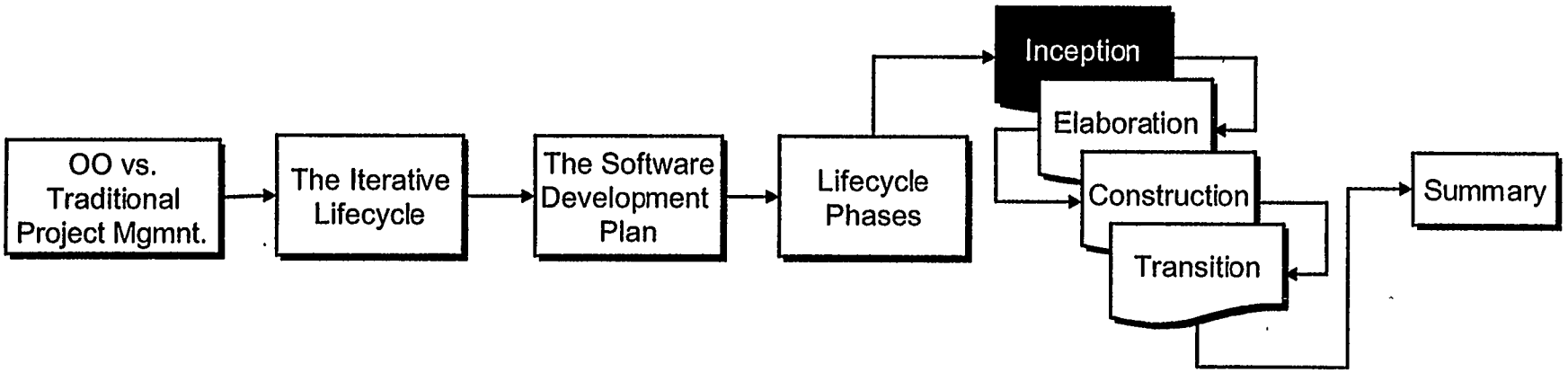
- Inception
- Elaboration
- Construction
- Transition



© 2000 Advanced Software Technologies, Inc.

ADVANCED
SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Inception Phase



© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Inception

- Major Activities
  - Capture all user requirements
- Deliverables
  - Use Case database
  - Use Case models
  - Requirements document
  - User interface prototypes
- Completion Criteria
  - Domain experts and requirements team have recorded all Use Cases
  - User sign-off on requirements

© 2000 Advanced Software Technologies, Inc.

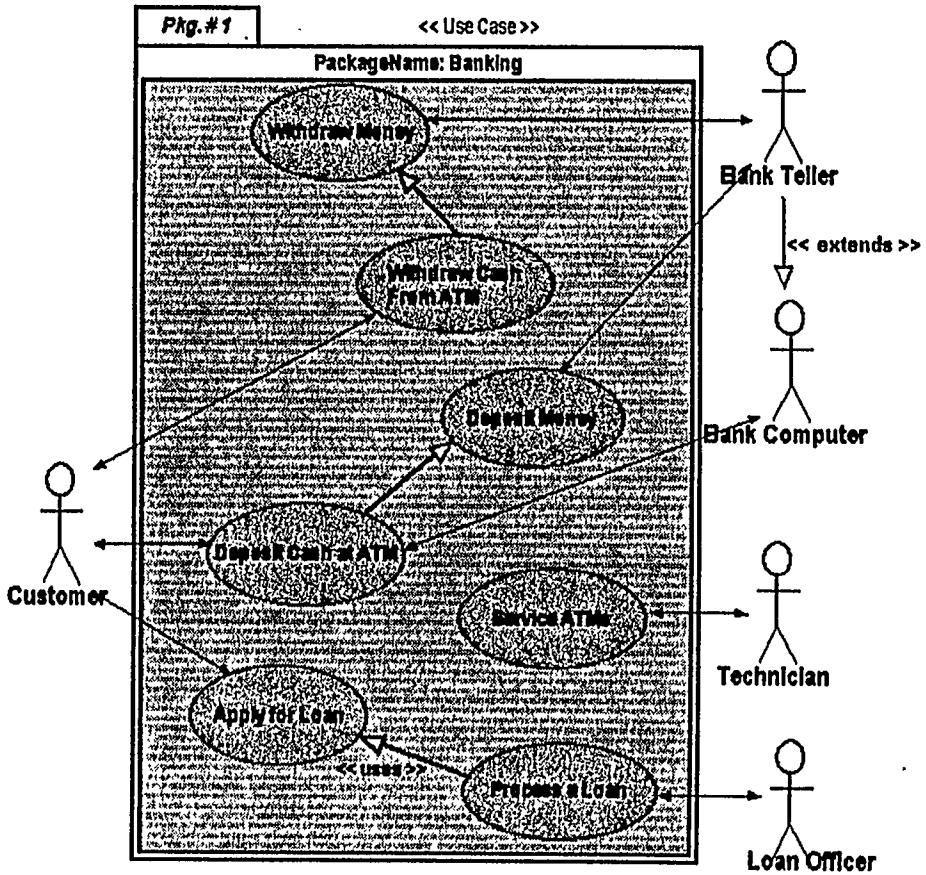
ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Development Use-Case Database Structure

FIELD	USE CASE 1	USE CASE 2	USE CASE 3	...
OSE Case Name				
Identifier				
Identifier of Underlying User-level Use Case				
Description				
Owner				
Date				

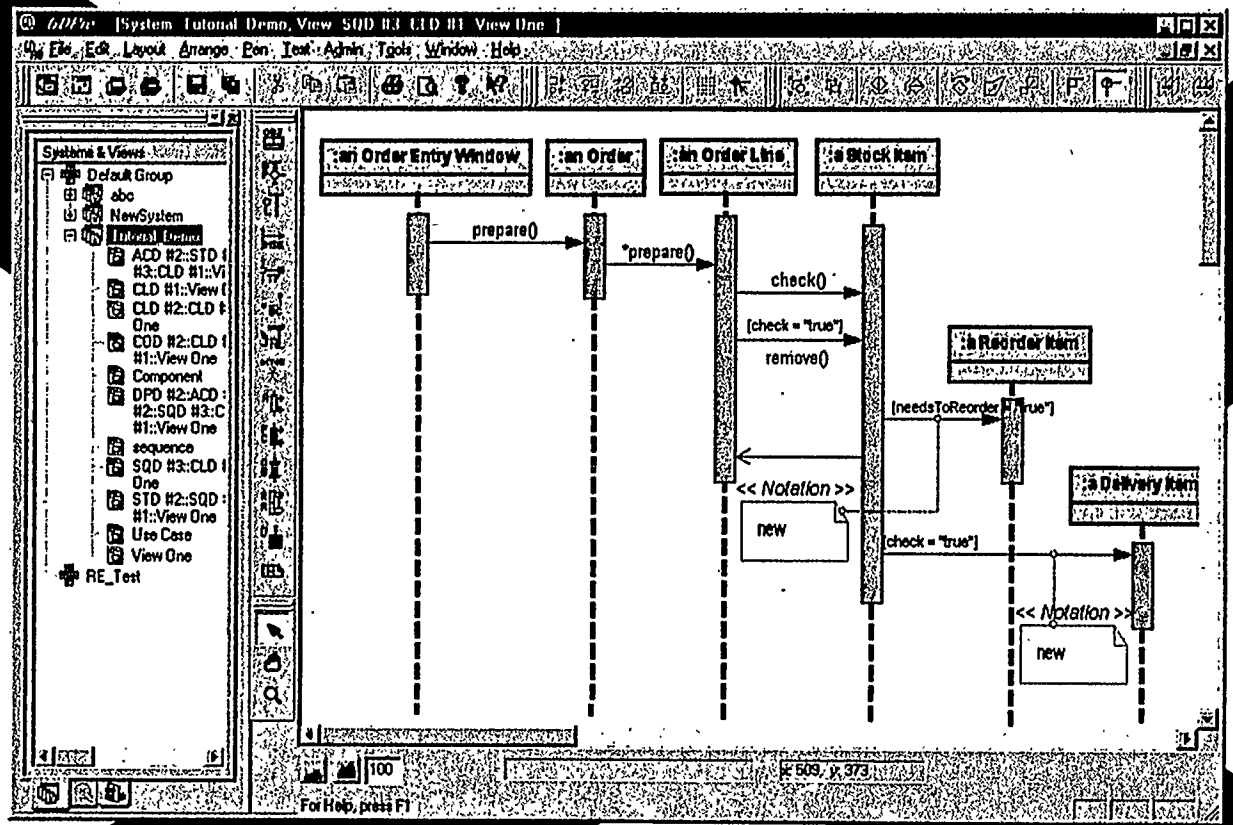
© 2000 Advanced Software Technologies, Inc.

GD*Pro* Use-Case Diagram

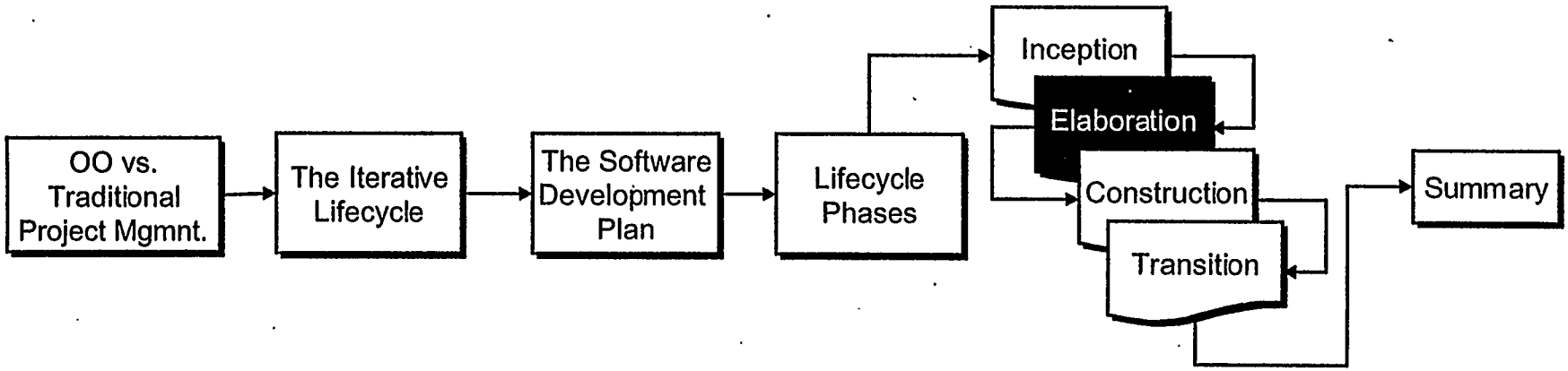


© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com


File Edit Layout Arrange Pen Text Admin Tools Window Help
Systems & Views
Default Group
abc
NewSystem
RE_Test
Use Case
View One
:an Order Entry Window
:an Order
:an Order Line
:a Stock Item
:a Reorder Item
:a Delivery Item
prepare()
*prepare()
check()
[check = "true"]
remove()
[needsToReorder
<< Notation >>
new
[check = "true"]
<< Notation >>
new
100
For Help, press F1

Elaboration Phase



© 2000 Advanced Software Technologies, Inc.

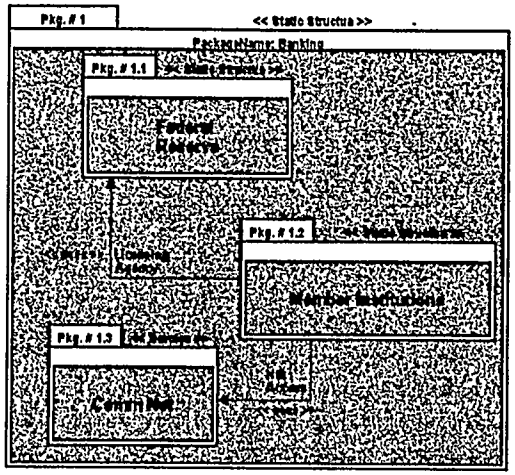
ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Elaboration

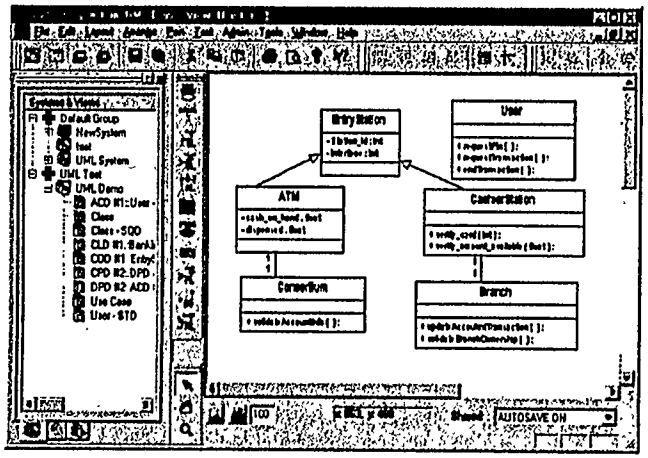
- Major Activities
  - Detailed System Design
  - Preparation for Construction
- Deliverables
  - Development Use Cases
  - Package Diagrams / Class Diagrams / Sequence Diagrams
  - Initial Test Matrix
- Completion Criteria
  - Design is complete - all Use Cases have been accounted for
  - Complete dynamic and static view of system has been designed

© 2000 Advanced Software Technologies, Inc.

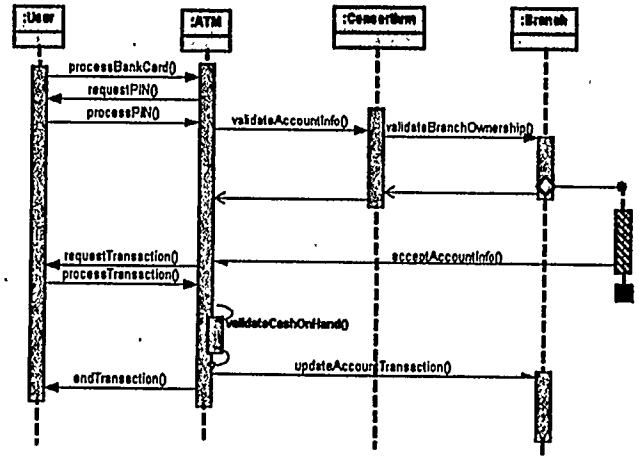
Package Diagram



Class Diagram



Sequence Diagram

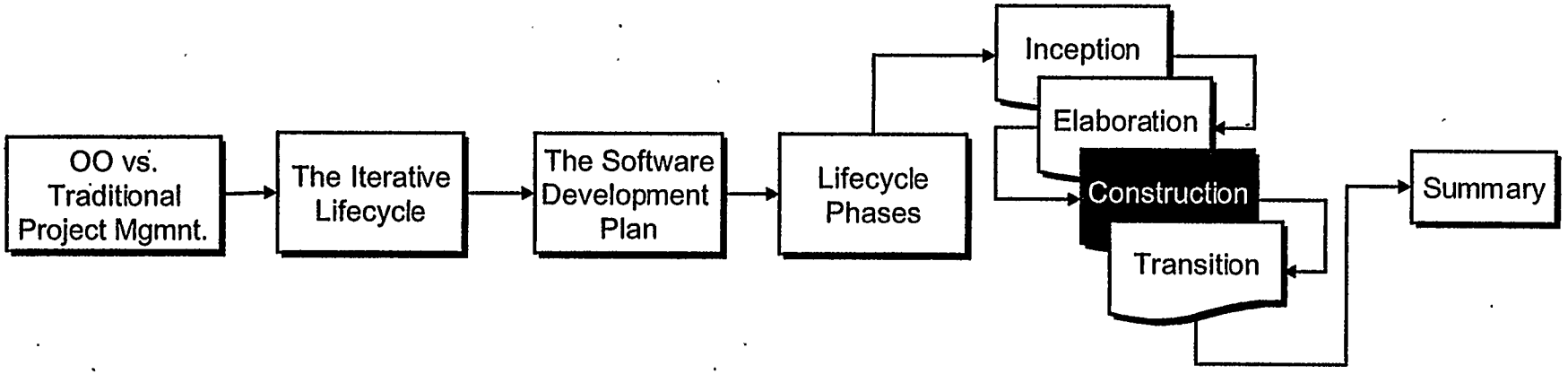


© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.

http:/www.advancedsw.com

Construction Phase



© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

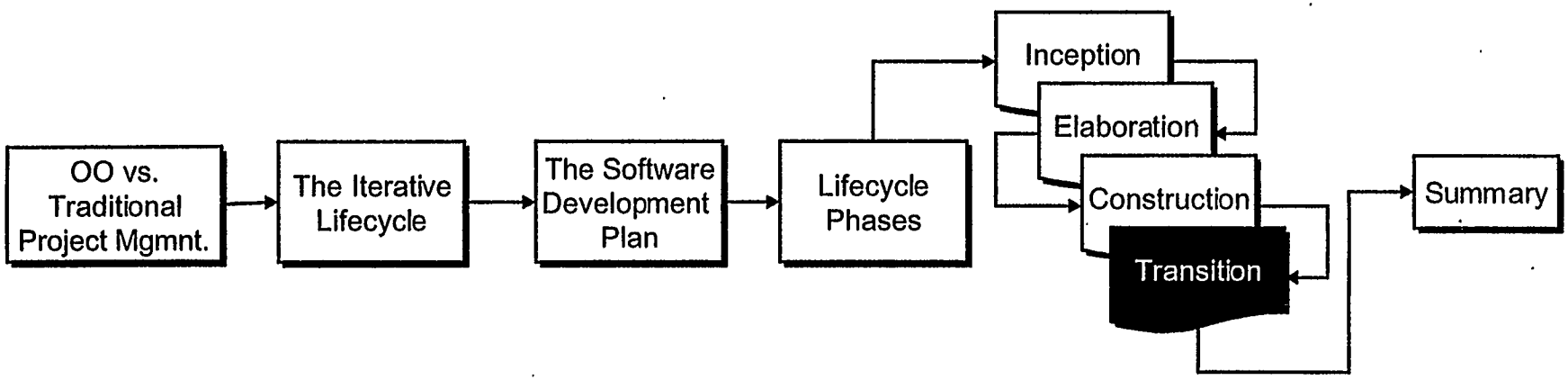
Construction

- Major Activities
  - System coding
  - System integration
  - Alpha testing
- Deliverables
  - Completed initial release of system
  - User and internal system documentation
  - Beta release plan
- Completion Criteria
  - Stable release exists that meets all agreed upon requirements
  - Complete documentation is ready to release with initial system

© 2000 Advanced Software Technologies, Inc.

ADVANCED
SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Transition Phase



© 2000 Advanced Software Technologies, Inc.

ADVANCED SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

Transition

- Major Activities
  - Refining system performance
  - Correcting bugs
  - Clarification of documentation
- Deliverables
  - Updated builds of system
  - Updated user and internal documentation
- Completion Criteria
  - Improved builds address specific identified issues
  - New release is ready for next iteration

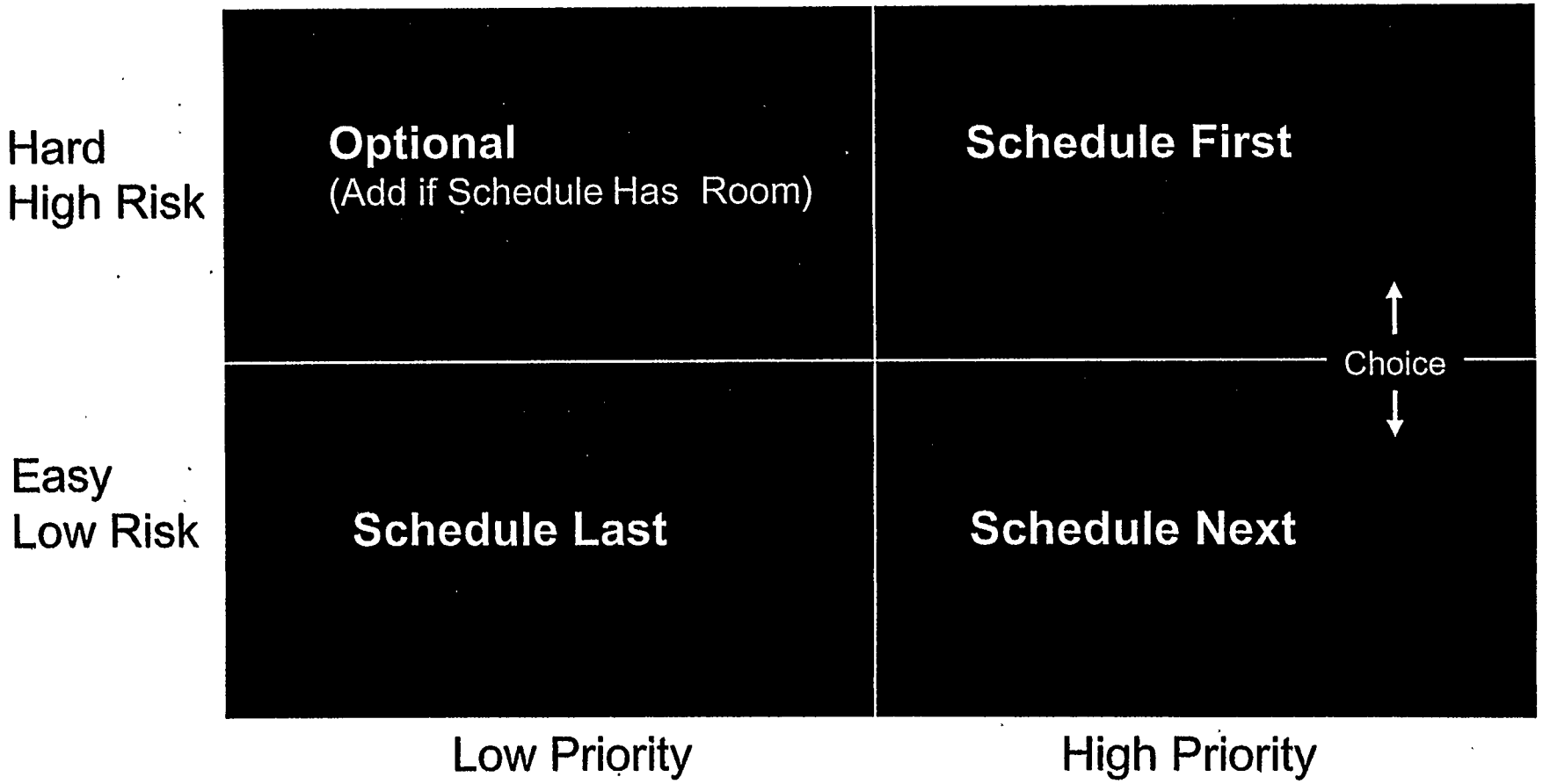
© 2000 Advanced Software Technologies, Inc.

How to Keep Things On Track

- Take your schedules seriously
  - Regularly modify your schedules so that they reflect reality
  - Support your team in staying on schedule
- Carefully manage requirements creep
  - Use Feature/Cost Tradeoff chart
  - Expect to add requirements throughout project
  - Well managed project should minimize additions as time goes on

© 2000 Advanced Software Technologies, Inc.

Feature vs. Cost Tradeoff



© 2000 Advanced Software Technologies, Inc.

ADVANCED
SOFTWARE TECHNOLOGIES INC.
http:/www.advancedsw.com

How to Keep Things On Track (cont.)

- Keep your artifacts up to date
  - At each new phase completion, be sure previous stage's artifacts are updated
  - May need to update earlier phase artifacts
- Stay flexible. Plans are most powerful as guides
  - Reality will impact your plans - count on it!
  - Avoid a reactionary mentality when things don't go as planned

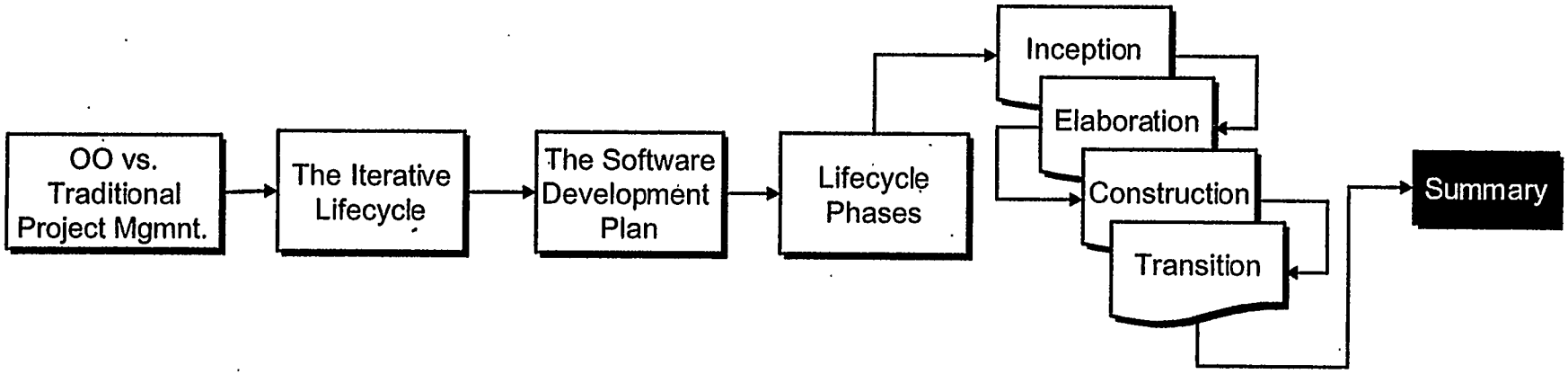
© 2000 Advanced Software Technologies, Inc.

How to Keep Things On Track (cont.)

- Keep your team fully involved and motivated
  - Be sure to acknowledge each and every team member
  - Use public celebrations and events to recognize your team
  - "Weed and Feed" as necessary
    - ✔There's no challenge a cohesive team can't overcome

© 2000 Advanced Software Technologies, Inc.

Summary



© 2000 Advanced Software Technologies, Inc.

Summary

- Create a software development plan
- Develop a schedule and budget
- Follow the Iterative Development Process
- Manage each phase carefully
- Keep schedules and artifacts up to date
- Keep the customer closely involved

© 2000 Advanced Software Technologies, Inc.

Process Development Experiences

The Accelerator Production of Tritium Project

The APT Project

The Accelerator Production of Tritium (APT) project is designing a particle accelerator and target to produce Tritium using Neutrons generated from a spallation source.

APT Project Roles

ED&D and Design Authority at Los Alamos

- LEDA, Area A
- Material Studies and Irradiation

General AtomicsPlant Design

- Systems Engineering

Savannah River

- Construction Site
- Balance of Plant

Burns & Roe

- Systems Engineering

APT Project Structure

- Project Director's Office (PDO)
- Operations Project Office (OPO) Savannah River Site
  - User of control and instrumentation software
- Plant Project Office (PPO) General Atomics, Burns and Roe
  - User of design software, Develop some software
- Technology Project Office (TPO) LANL
  - Develop design software

Engineering at APT

- Mature and Competent
  - Defined Lifecycle
  - Technical Reviews at all levels
  - Configuration Management
  - Emphasis on Quality and Safety

Software at APT

- Design Software
  - Radiation Transport, Transmutation, Thermal-Mechanics, Hydraulics
  - Nuclear Data, Engineering Calculations
- Engineering Software
  - CAD, Systems Engineering
- Control Systems
  - Access Systems, Beam Control
- Test, Measurement, Instrumentation

Software at APT

- Safety
  - Accident Analysis (MACCS)
- Project Controls
  - Primavera
  - EPCS
  - Office Support, Microsoft, Framemaker
  - IMS, Documentum

Software Engineering at APT

- Variety of software practices
- OPO has mature process definition
- PDO has process definitions derived from other engineering areas
- TPO has few processes and a research tradition

Process Improvement Strategy

How to provide a software engineering strategy that produces a final system which is:

- Acceptable to the nuclear engineering regulatory agencies
- Consistent with SMPs at collaborating organizations
- Capable of long term maintenance

Software Strategy Choices

- Use Mature processes from Savannah River/OPO
- Develop processes by extending PPO general engineering processes
- Develop processes specific to research and development needs of TPO

Two-Pronged Approach

- Top Down
  - PDO approval of project-wide software management plan based on CMM
    - Graded Approach
    - Common Terminology
  - Form Software Engineering Process Group
- Bottom Up
  - Insert processes into existing operation
  - Educate on need for process improvement
  - Provide experience base to feed SEPG

Specific Progress

What progress has been made in CMM level 2 and 3 key process areas?

We are currently focussing on 3 specific examples:

- The MCNPX code project
- Excel Worksheets for plant design
- LANSCE Area A data acquisition software

Requirements Management

- Software requirements derived from Project-wide Engineering requirement documents.
- Weekly meetings assess and revise as needed.
- More formal documentation needed.

Software Project Planning

- Overall project planning well supported
  - Derived from project requirements documents
  - Revisions well reviewed and planned
- Software planning more ad-hoc
  - Shows lack of written requirements
  - Derived by informal consensus

Software Tracking and Oversight

- Weekly meetings provide good informal tracking
- No process measurement or history
- Not well integrated with overall project tracking system

Software Subcontract Management

- Good process in place
- Based on existing Los Alamos contracting process

Software Quality Assurance

- SQA function defined
- Initial SQA group formed
- Still in introduction and acceptance stage
- Little measurable history

Software Configuration Management

- Existing project-wide CM not geared for software development
- Two tool suites made available (Razor and CVS/GNATS)
- Code being maintained with CVS/GNATS
- User acceptance growing

Organization Process Focus

- Software Management Plan developed
- Specifies CMM as the measurable basis for process development
- Instituted at PDO level
- Supports OPO, PDO, and TPO existing processes

Organization Process Definition

- Software Engineering Process Group established
- SEPG based on PDO Software Management Plan
- Initial projects providing process candidates identified

Training Program

Little is done for planning of training now. This will naturally grow as SPP and OPD expand.

Integrated Software Management

This is a goal. The very different levels of development maturity of the three main organizations makes this a future goal.

Software Product Engineering

Engineering processes are actually quite good, though ad-hoc and based on experts. A major goal of the SEPG will be to maintain and extend these practices.

- Good test plans
- Good analysis and design support

Intergroup Coordination

- Good, though ad-hoc
- Weekly status meetings often include other engineering representatives
- High level management participates in software activities

Peer Reviews

- So ad-hoc that most members don't realize they regularly happen
- Not universal
- Not well documented
- Small groups

Summary and Results

- A good basis for process improvement is underway
- The two-pronged approach is necessary
- Establishing a program among organizations with different levels of maturity is hard

Software Team Strategies Used For WIPP

Kathleen Byle

Software Quality Engineer

Quality Engineering Dept (12326)

kabyle@sandia.gov

Jeanne Evans

Quality Engineer

Weapons Quality Dept (09821)

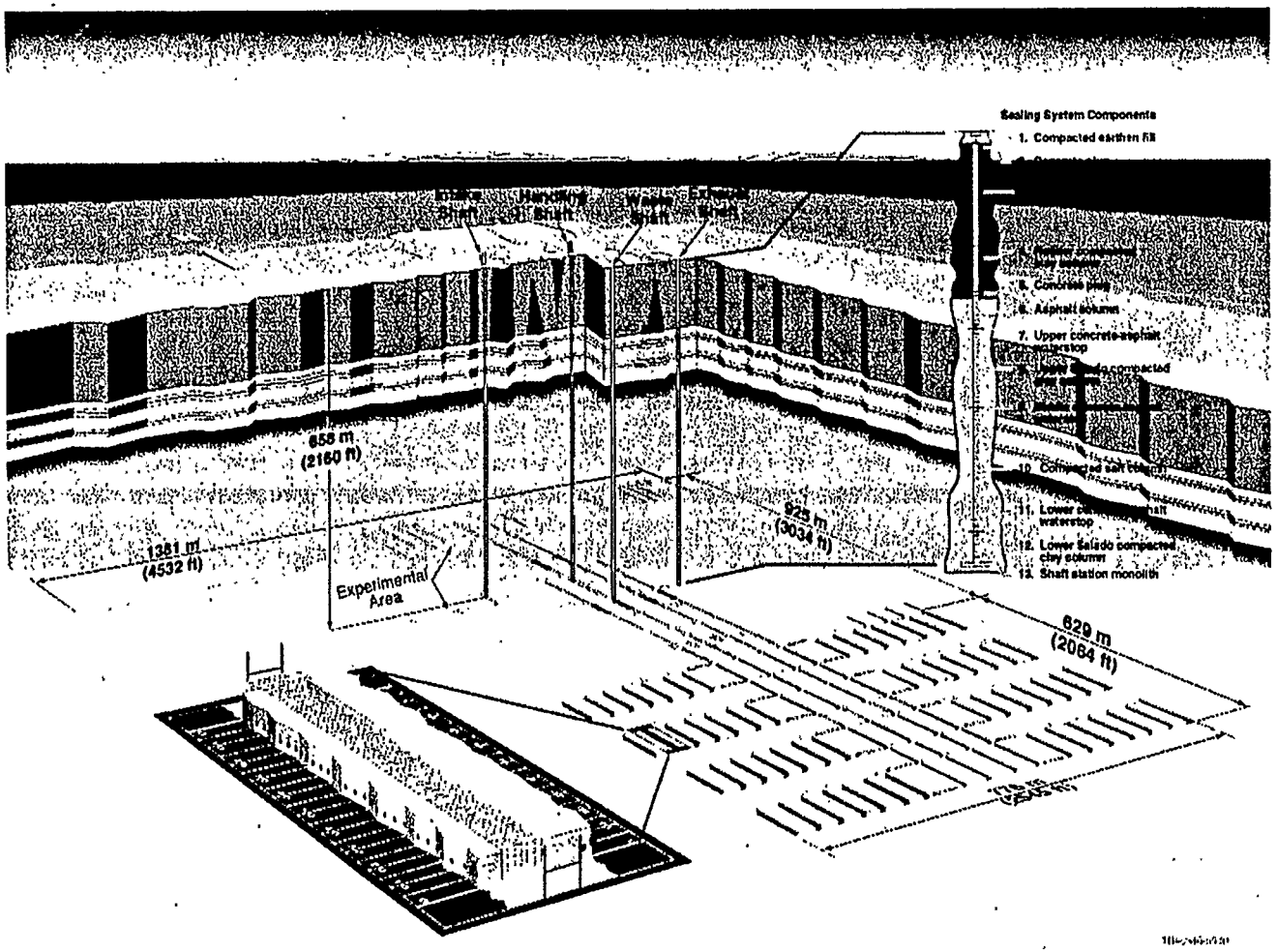
jlevans@sandia.gov

Sandia National Laboratories

Presentation Content

- Introduction to the Waste Isolation Pilot Project (WIPP)
- Team 1 Structure
- Team 2 Modified Structure
- Teaming Lessons Learned

Cutaway Schematic of the WIPP

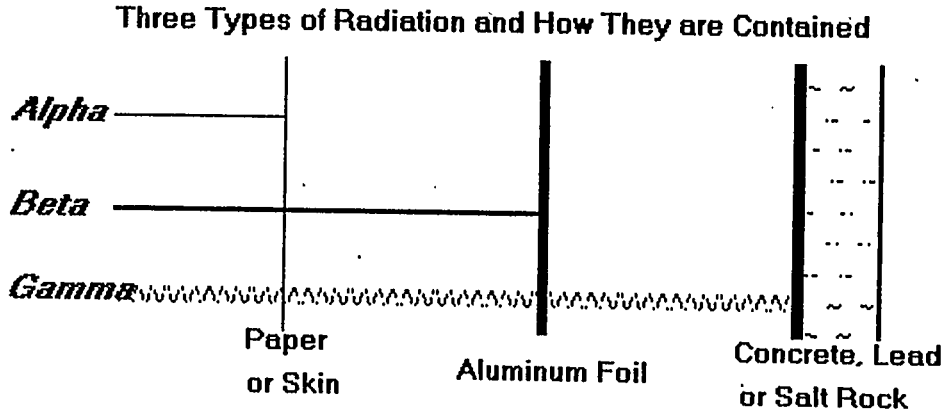


Sandia National Laboratories

WIPP Purpose

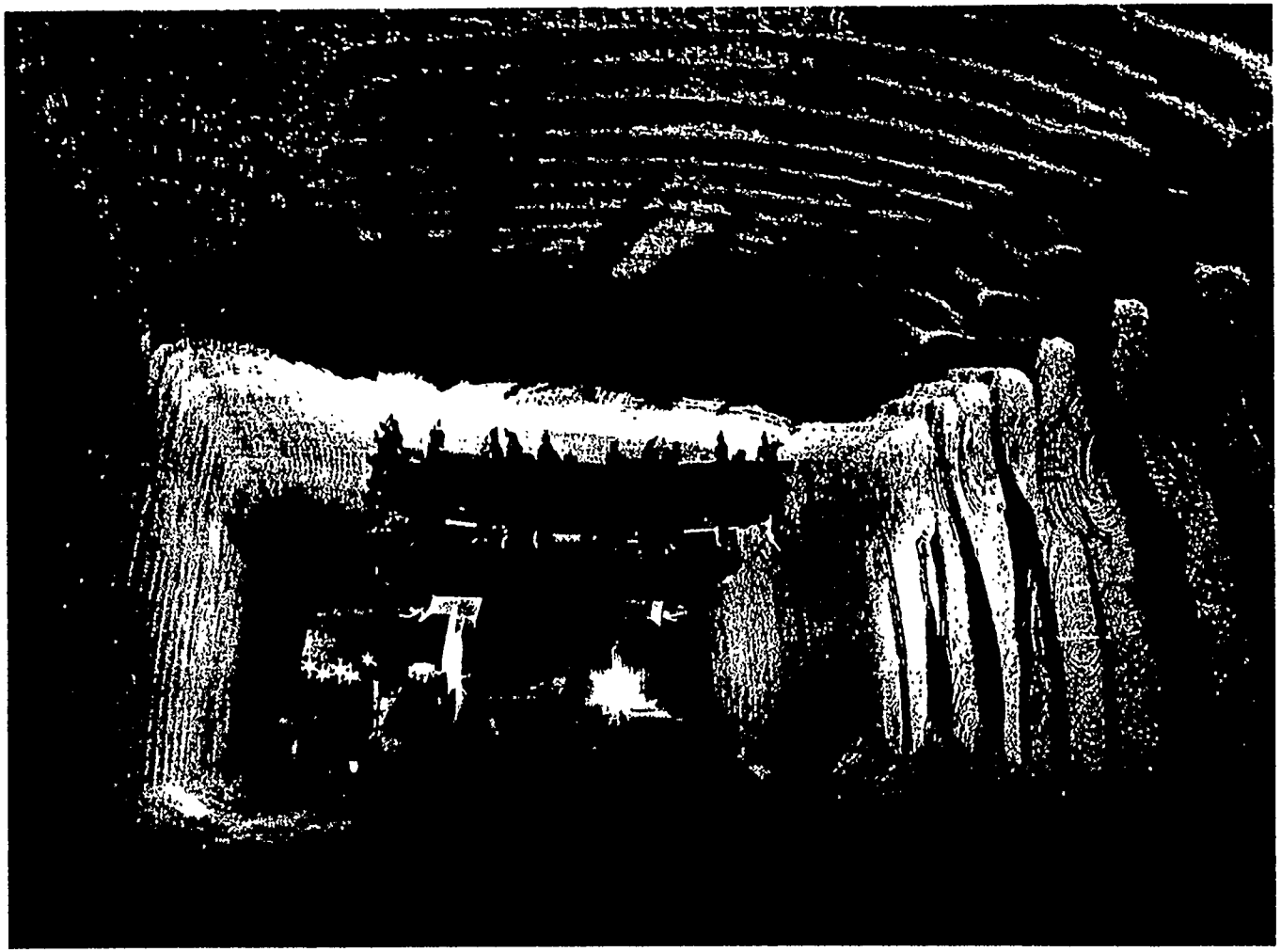
For disposal of transuranic (alpha) materials only

Transuranic Waste: consists of clothing, tools, rags, residues, debris and other such items contaminated with small amounts of radioactive elements.



Sandia National Laboratories

Inside WIPP



Sandia National Laboratories

Team 1 Mission

As Science Advisor for WIPP, develop a methodology (experiments and code simulations) for the evaluation of the waste repositories to determine if safe for disposal of Transuranic Waste.

Customers : Regulator and the Public

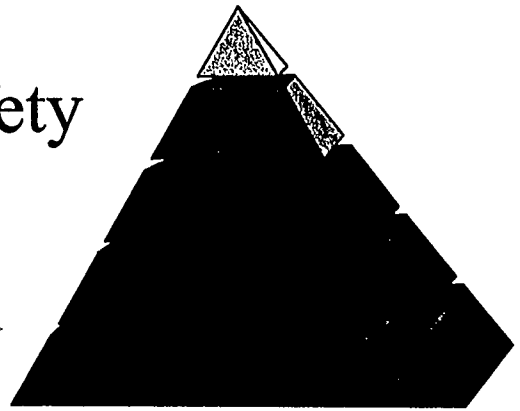
Documentation Produced and QA Methods used "... shall be sufficiently detailed as to purpose, method, assumptions, design input, references, and units such that a person technically qualified in the subject can review and understand the analyses and verify the adequacy of the results without recourse to the originator."

Team 1: Constraints

Quality

Public Health & Safety

DOE & EPA



Resources

Budget Negotiable

Physical

Time

Fixed

Sandia National Laboratories

Team 1: Detailed Constraints

Time: Criteria defined Feb. 96, application (CCA) due Oct 96

Resources

Code qualification (50 numerical and utility codes)
1965 FORTRAN files, 373 .inc or .par files, 624,336 lines

Complexity of simulation:
5,500 code configuration input files
37,000 cpu-hours (over 4.2 cpu-years!) of computation
200,000 files (source, input, script), 95 Gigabytes output

Legacy codes (testing in quality vs building in quality)

Quality: Congressional mandate, NQA, DOE & EPA
Product vs process

Team 1 Solution: Brute Force Code Teams

Code Sponsor
Domain Expert

Technical Reviewer:
Inspections



Support Staff : Tester,
Documentation preparer

Software Configuration
Management Coordinator:
Configuration management

SQA Coordinator:
Procedures, Verified
compliance with customer
requirements

Management Team:
Resource allocation,
Artifact tracking

All roles and responsibilities proceduralized to keep on track with customer requirements

Sandia National Laboratories

Team 1 Solution:Weekly Meetings

- One meeting per week per code team (15 minutes - 1 hour)
- Every problem resolved in one week
- All members involved (no finger-pointing)
- All deliverables tracked and status reviewed each week.

50 codes, 50 meetings per week,
Management involvement 1-2 days a week

Sandia National Laboratories

Team 1 Success!

- October 1996: 84,000 page Compliance Certification Application to EPA
- WIPP is Certified by the EPA (after 1 year review and comment period)
- March 26, 1999: First shipment of waste from LANL arrives at WIPP.

Team 2 Mission

- Cheaper, faster, better **L** (Process Improvement **J**)
- A significantly reduced budget (from $13M to $2M)
- Improving efficiency of site evaluation
- Upgrading the simulation models
- Testing and requalifing the software
- Improving the user interface for use by the regulator
- High level of tension between developers and testers

Team 2 : Constraints

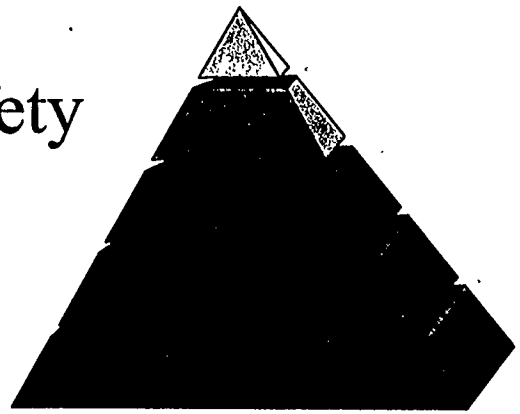
Quality

Public Health & Safety

DOE & EPA

Resources

85% Reduction of SQA Budget



Time

Expanded for Recertification

Sandia National Laboratories

Team 2 Solution: Code Teams Combined with Team Structure

- Constraints that could not be addressed by previous team structure, budget reductions, introduction of a new methodology (OO).

- Reduced hands-on Management participation

Team 2 Roles

- Champion (Technical Manager)
- Team Lead (Technical Lead)
- Core Team Members
- Supplemental Member (Consultants)

Team 2 Responsibilities

- Technical Manager
  - Support team structure, remove identified barriers
- Technical Lead
  - Team process
  - Communicate contribution of individuals
  - Ensure that developer/tester responsibilities clearly defined

Sandia National Laboratories

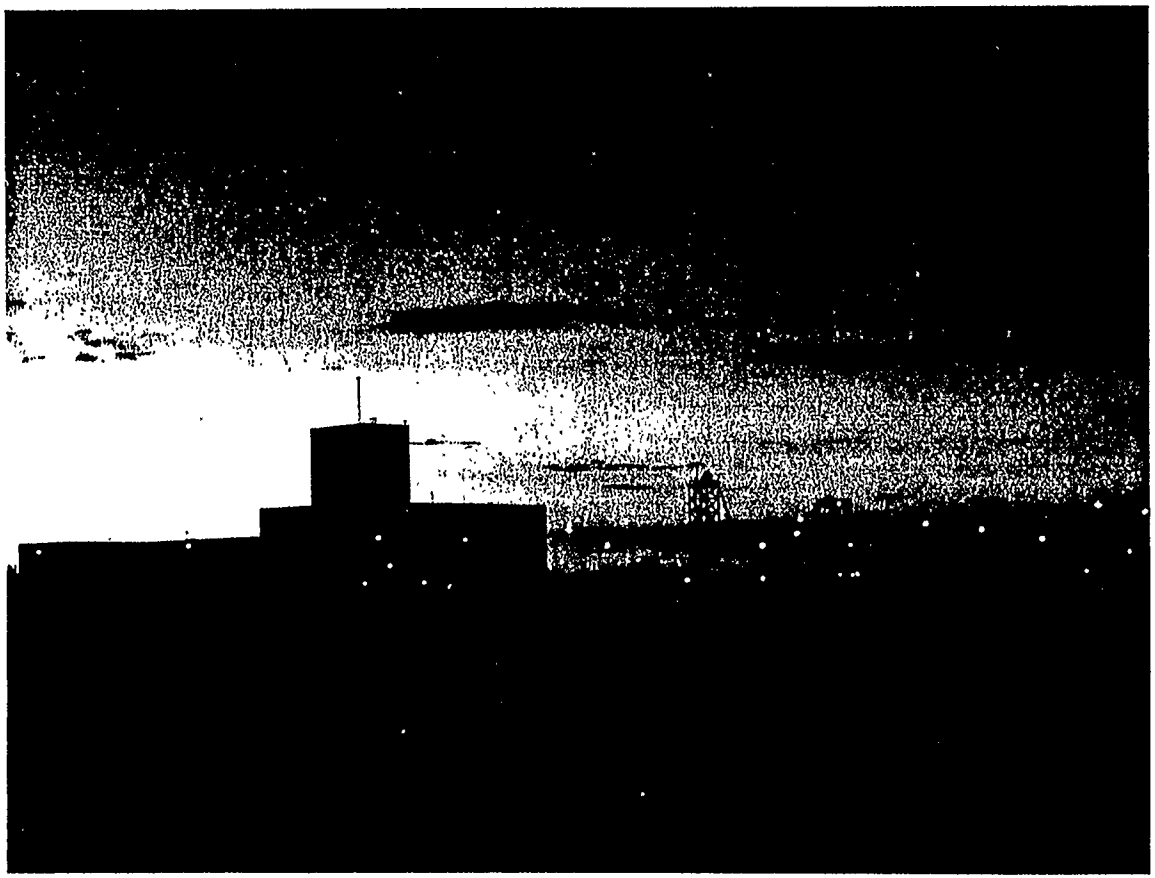
Team Process

- Purpose, scope, schedule, budget for team deliverables clearly defined
- Agendas, minutes, action items tracked
- Use of parking lot for tangential issues
- Use of clear decision-making rules (whose decision is it?)
- Clearly defined communication paths (and included in roles)

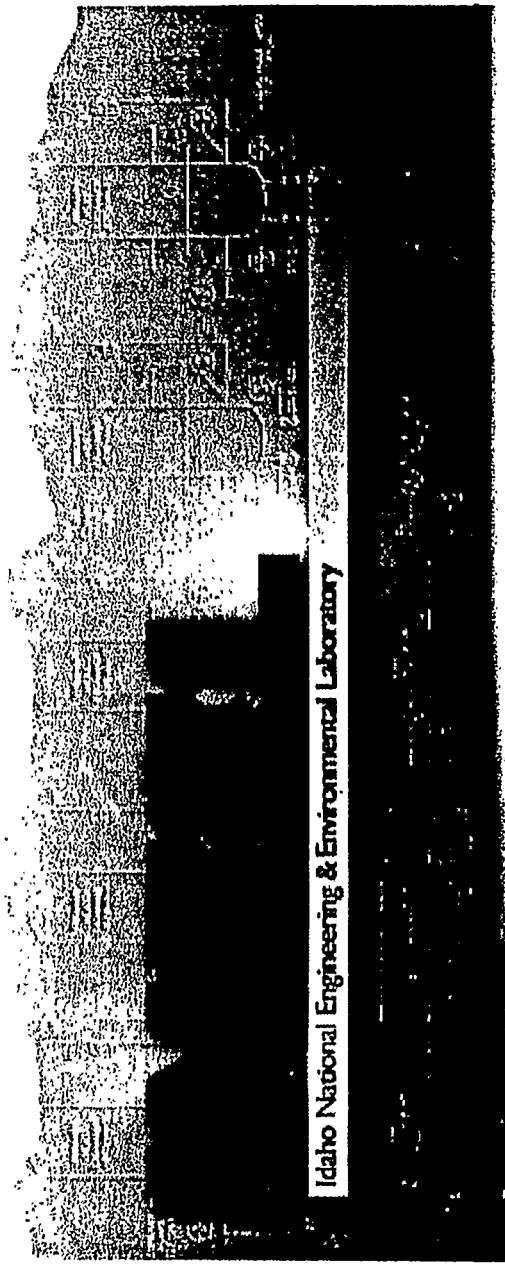
Teaming Lessons Learned

- Team Structure reduces time spent by members on overhead e.g., communication, decision making, ...
- Any "average" structure will succeed with Management support any "excellent" structure will fail without Management support
- If cultural reward system is individually based, individual accomplishments must be recognized and communicated to management.
- 80/20 Rule for Team Lead
  - Team Lead must spend 80% of time on process issues (rather than content contribution), otherwise team will not see a gain in productivity and structure will disintegrate.

?'s

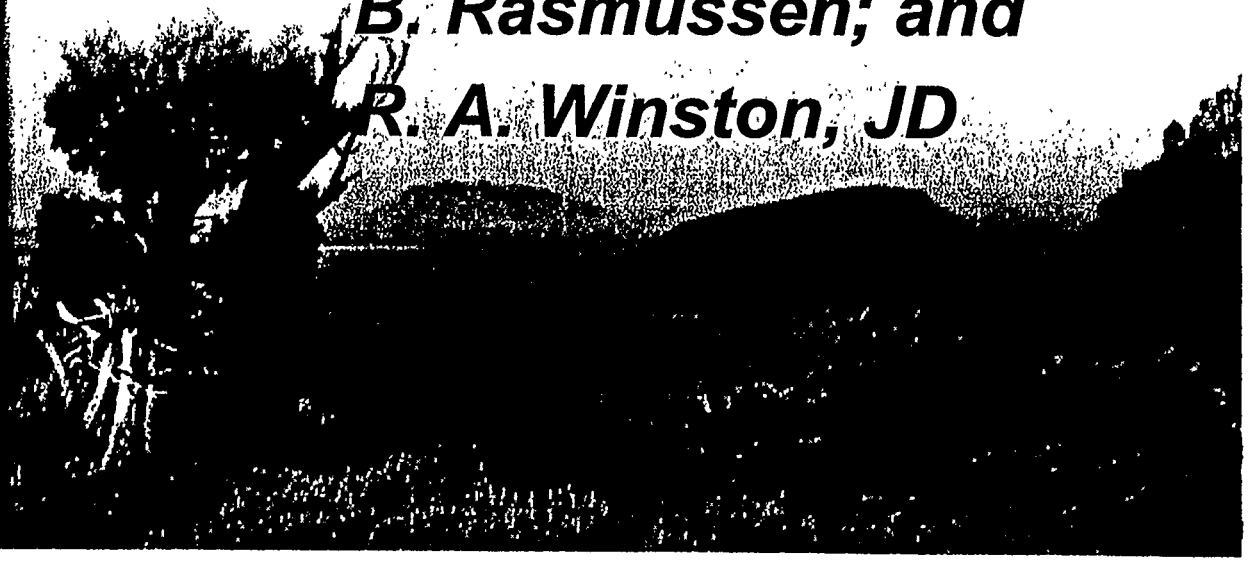


Sandia National Laboratories



DEVELOPING A COHESIVE TEAM: FOCUSED, FACILE, AND FRIENDLY

R. E. Heard, PMP;
G. L. Leatherman, PhD;
B. Rasmussen; and
R. A. Winston, JD



Team

- ***A collection of individuals pulled together to accomplish a task, activity, or project.***



Project Manager's Role

- **Manage:**
  - **Interface with management and customers**
  - **Human resource dynamics on project**
  - **Cost**
  - **Schedule**
  - **Scope**



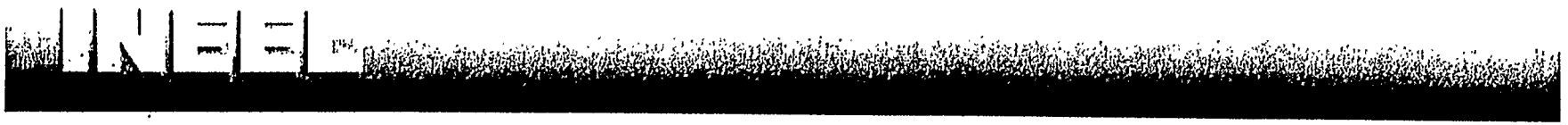
Team Lead's Role

- **Manage:**
  - **Interface of various technical capabilities for accomplishment of team's project**
  - **Interface with Project Manager**
  - **Scope for the accomplishment of technical milestones**
  - **Assign tasks to individual team members**
  - **Prepare technical documentation**



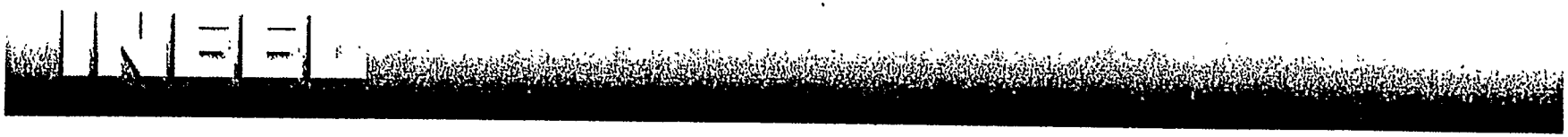
Team Member's Role

- *Interface with Team Lead*
- *Interface with other team members for task accomplishment*
- *Develop individual portion of technical documents*
- *Develop path forward for individual task assigned*
- *Accomplish tasks assigned*



Project Manager's Perspective

- ***Literature defines the role as the application of knowledge, skills, tools, and techniques to project activities in order to meet or exceed stakeholder needs and expectations from a project.***
  - ***A Guide to the Project Management Body of Knowledge, PMI, 1996.***



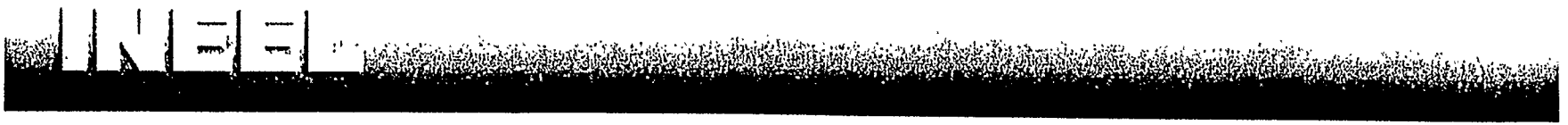
PM's Role by Work Pkgs and Company Literature

- Work packages generally define role by a level of effort solely stating project management will occur
- Company documents seldom refer to the PM; however, when they do, the PM is often referred to as parallel to the line manager with the same roles and responsibilities
  - Usually without the same authority



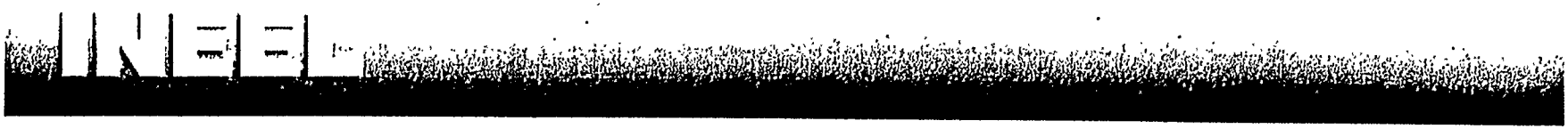
PM's View of Roles Within Team

- Reliant upon the Team Lead for the blending of technical insight into the project
- Reliant upon the Team Lead for updates and communication regarding technical tasks for the project
- Follows-up with Team Lead with spot checks within team to accomplish team building



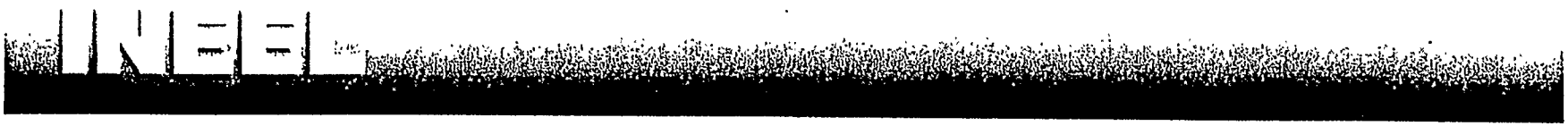
PM Success

- *Trust and communication with management and customer*
- *Trust relationship with Team Lead*
- *Properly educated and trained staff*
- *Good documentation*
- *Access to support personnel; eg., planning and budgets, safety, human resources*
- *Training*
- *Proper PM tools*
- *Ease of corrective actions & change control*



PM's View of Team Success

- Open honest communication
- Clearly defined work scope
- Frequent customer briefings
- Use of documentation
- Proper training
- Proper tools
- Proper blend of human resources: functions and personalities
- Informed team
- Frequent team communications and regular update meetings
- Relationship with Team Lead



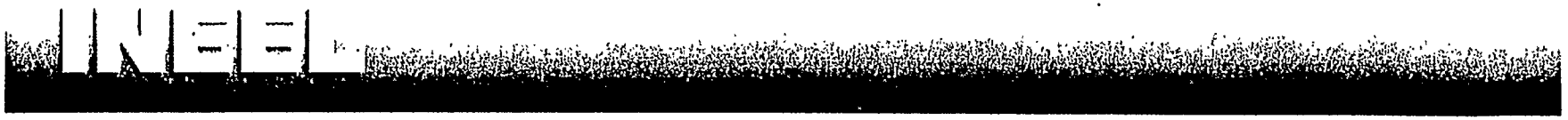
Team Lead's Perspective

- ***Team Lead's role is not defined per se in literature and is often confused with the PM. This confusion is often fostered by the use of the term "principal investigator" where the PI is often endowed with the tasks of the PM as well as the Team Lead.***



TL's Role in Work Pkgs & Company Documents

- Work packages generally show the skill type of the Team Lead rather than the designation of team lead
- Often justification for position is vague or refers to the technical expertise rather than to the duties of the position
- Often no Company defined position



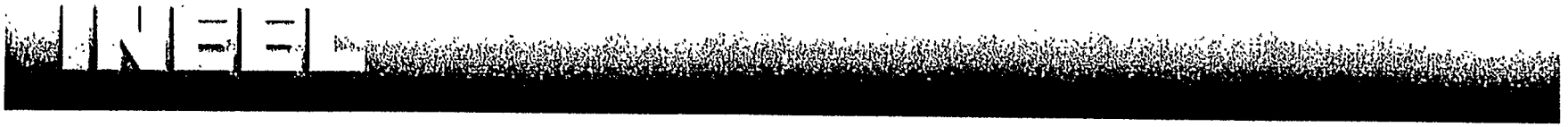
TL's Success

- Broad experience on various projects
- Some education or experience in many of the technical areas of the project
- Open communication with team members
- Open communication with PM
- Mentoring
- Ability to make technical decisions without fear of PM overruling[with limited exceptions]
- Some customer and management interface for broader understanding
- Open to technical judgements of team members



TL's Successful Team

- Open communication
- Necessary and competent technical staff--dedicated
- Trust of Project Manager Clearly Defined scope
- Frequent checks with customer and management
- Necessary tools
- Necessary training and education
- Flexibility
- Free flow of information



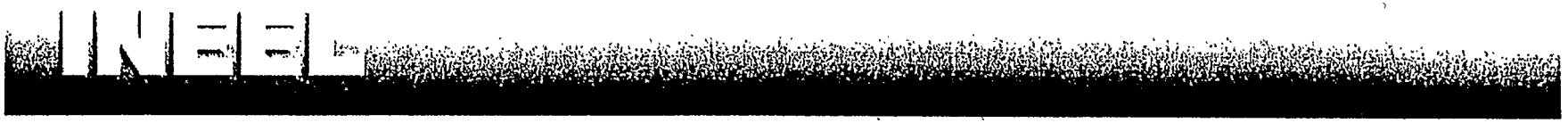
Team Member's Perspective

- ***The team member is not defined in literature* per se; *however, the operable definition is often that of an employee, scientist, or engineer.***



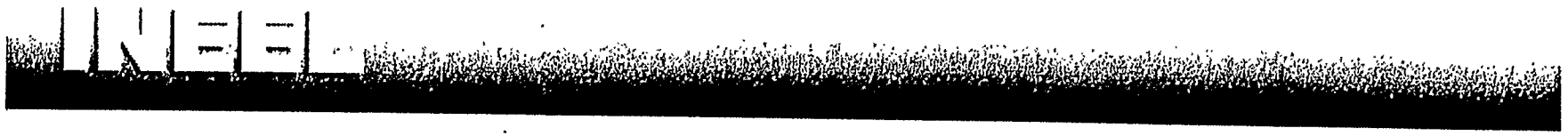
Team Member in Work Pkg. & Company Documents

- Each skill is listed in the work package; often several FTEs with same skill mix are listed on the same line
- No distinction is made in reference to expertise, years of experience, or training other than such is often averaged into the salary base
- Company defines as employee



Individual Team Member Success

- *Work is recognized and rewarded*
- *Ability to negotiate changes in schedule, cost, or scope when necessary*
- *Good working relationship with team lead, other team members and project manager*
- *Open, honest communication*
- *Proper tools and education*
- *Opportunity for technical growth*
- *Define role within the team*
- *Expertise acknowledged by team*



TM's View of Team Success

- *Product realized*
- *Recognition for job well done*
- *Rewarded for extraordinary effort*
- *Customer feedback of satisfaction*
- *TL & PM feedback of good job*
- *Follow-on work*
- *Challenging follow-on work*
- *More skill enhancement*
- *More training*
- *Possible Company recognition*



Communication

- ***As noted, every element of the team highlights communication that is honest and is key to a successful team***

Verbal Communication

- *Most often used channel*
- *Used for informal communication*
- *Used for non-milestone communications*
- *Used for immediacy*
- *Used to communicate general project information*

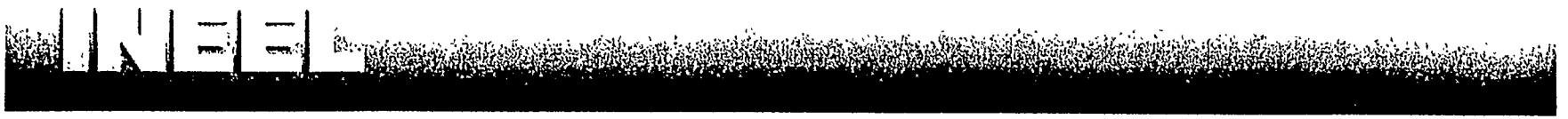


Verbal Con't

- Too often used to by-pass written communication
- Too often not followed-up with written
- Too often used to communicate down a chain w/ information lost or miscommunicated
- Too often incomplete

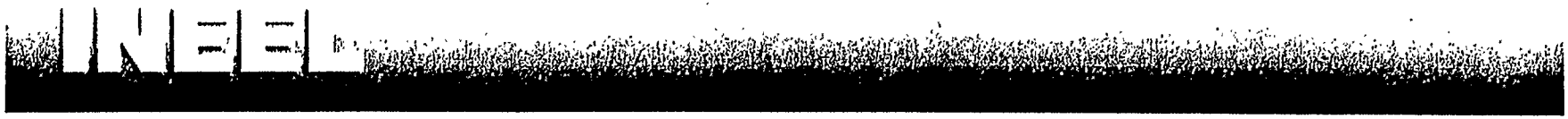
Written Communication

- *Formal - letter, memo, e-mail*
- *Used to follow-up verbal*
- *Used to address changes or status of cost, schedule, and scope*
- *Used to transmit information such as reports*
- *Used to outline steps*
- *Used to capture meeting notes and actions*



Written Con't

- Not used due to formal process
- Company procedures - create burdensome restrictions
- Often not reviewed for completeness or preciseness of communication
- Becomes part of project record so team is hesitant to use



Other Communication

- *Meetings while verbal are another form of communication*
- *Slide presentations*
- *Graphics*
- *Informal side meetings with select members of team*
- *Unspoken perceptions*



Documentation

- Should be reviewed for accuracy with a knowledgeable team member
- Not all should be issued by the PM
- Should be used to reinforce prior verbal communications
- Should be used to capture communication with customers or outside activities



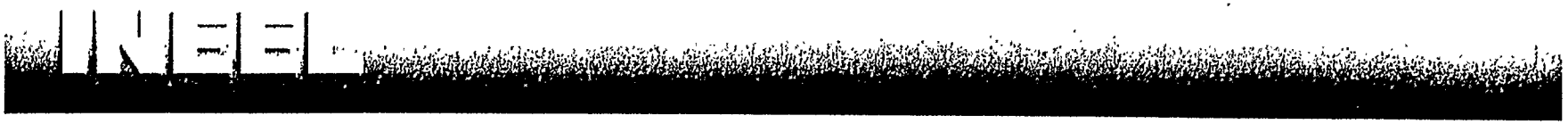
Roles and Responsibilities Definition

- Necessary to establish team member identities
- Avoids miscommunication
- Enhances communication
- Adds to customer and management understanding
- Promotes discussion within team, especially on authority and ownership



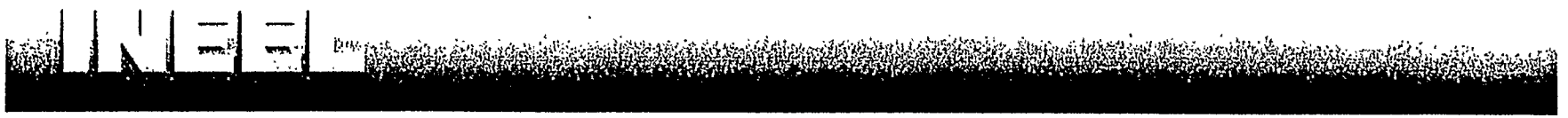
Scope Definition

- ***Must be fully established with customer***
- ***Must be detail sufficient but not detailed bogged***
- ***Must allow for planning***
- ***Must be able to attach budget to discrete activities and/or tasks***
- ***Must identify known constraints***
- ***Must identify assumptions***
- ***Must state end result***



Requirements Definition

- *Further limits scope creep*
- *Requires customer interaction*
- *Enhances change control if necessary*
- *Enables better task breakdown*
- *Allows development of checklists*
- *Allows for software quality assurance steps to be developed*
- *Allows for directed team and team/customer meetings*
- *Alternate techniques "proto-typing"*



Other Necessary Team Elements

- *Ability to recognize need for rework*
- *Ability to recognize communication stove-pipes*
- *Ability to recognize when one does not have requisite skills*
- *Ability to meet honestly over tough issues: human resources, skills, tasks failures, etc.*

Conclusion

- Each team member must have a defined role
- Each team member must be able to interact with other team members including PM and TL
- Each team member must feel and believe he/she is valued
- Each task of the project must be defined and planned to allow for the communication to occur
- Team lead relationship is critical for project success



Software Product Processes
Web-Based TBP Training — Overview

Software Quality Forum 2000

April 18, 2000

presented by

Software Quality Assurance Subcommittee

Work Item #25

Cathy Kuhn, Honeywell, FM&T

Operated for the United States Department of Energy under contract No. DE-AC04-76-DP00613

Honeywell

Orval Hart, LANL

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

Los Alamos National Laboratory

Dave Peercy, Patty Trellue, SNL

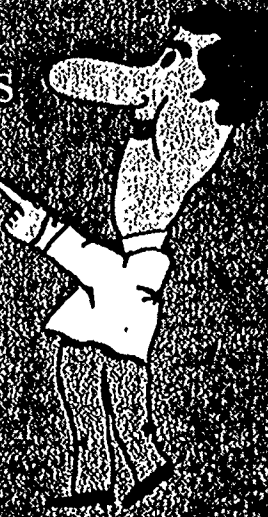
Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Philip Huffman, Mason & Hanger, Pantex

The Pantex Nuclear Weapons Facility is operated by Mason & Hanger Corporation, a subsidiary of Day Zimmerman, for the United States Department of Energy under contract DE-AC04-91AL65030.

Agenda:

- Overview of Session (5 min)
- Training Demo - TBPs (20 min)
  - —A Quick Look at the TBPs for Context
- Training Demo-SW Product Processes (40 min)
  - —Software Product Definitions and Scope
  - —Selective Process Overview with Case Study Examples
    - Program Management
    - Identification/Definition
    - Design/Production
    - Qualification
    - Acceptance
    - Delivery
    - Support
- Specific Concerns & Audience Q&A (25 min)



Session Objectives:

Review Customer Concerns and Expectations

- Why must we consider software as a product?
- QC-1 and D&P Manual guidance?

Provide Overview Training on NWC TBPs

- What are the TBPs and TBP-306 on Software Product Processes?

Illustrate Use of TBP Training Web Site

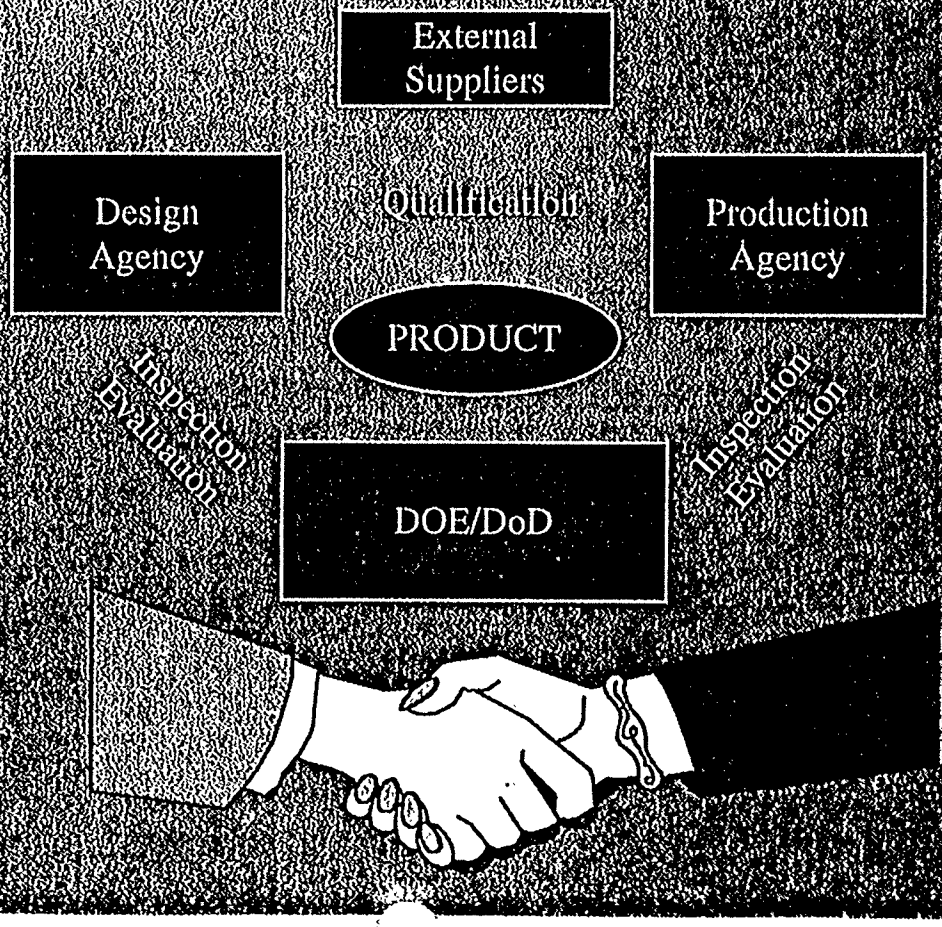
- How can I use the web site to understand the TBPs and TBP-306?

Discuss Your Concerns

- What software should/might be considered product?
- What is a software PRP and how does a software PRT work?
- How do I identify/qualify software product?
- How (& when) do I submit software product to DOE for acceptance?
- How is software product delivered? NWC Sites? External Customers?

Overview: NWC Vision

To Control and Manage Product While Exceeding All Customer Expectations



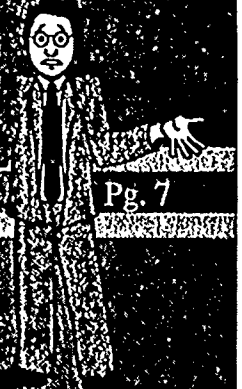
Overview: Customer Expectations

- QC-1 and D&P Manual
  - —Software Is Identified As Product
- NWC Technical Business Practices
  - —Officially Activated July 16, 1999
  - —Training Web Site in Evolution
    - http://prp.lanl.gov:8686/
  - —Training for Software Product Processes
    - Technical Business Practice TBP-306
    - Design Guide DG10235

On-Line Training Demos

- Technicall Business Practices
  - —A cohtext for cohtrolling software as a product
- TBP-306, Software Product Processes
  - —Applying the TBPs to software

Specific Concerns



- Requirement Not Well Known Across [illegible]
- Complex-wide Process NOT Completely Defined
  - —Level 1 and Level 2 TBPs allow room for flexible interpretation and implementation
- Software has a large application scope that presents challenges to the TBP system
  - —Embedded; Test Equipment; Numerical Control
  - —Development; Process Equipment; Logic Device
  - —Inherited; Legacy
  - —Simulation; Scientific Codes; Analysis

Specific Concerns

Application Domains

Test Equipment

Process Control

Calibration Software

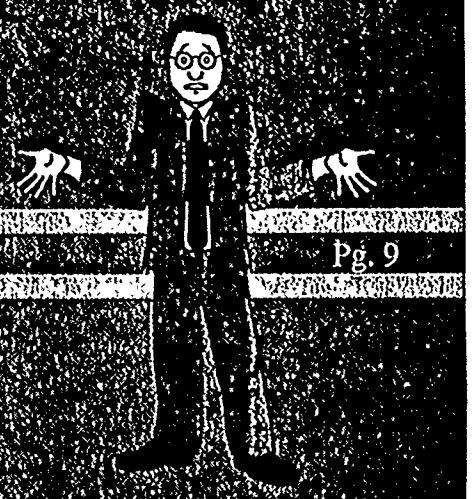
Present ??

Scientific Applications

Stockpile Validation

Logic Devices

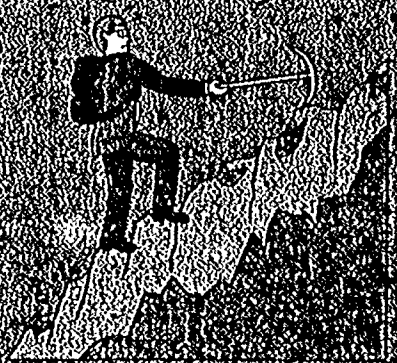
Specific Concerns



- Software Qualification Relationships
  - Quality Engineer (Role?)
  - Product Realization Team (Scope?)
  - Qualification Plan and QER (Format & Content?)
  - Site Quality Assurance Procedures (vs TBPs)
- Software Acceptance Relationships
  - Internal Inspection: Pre QAIP
  - External Inspection: QAIP Interface with DOE

Help is Available From

- Software Quality Assurance Subcommittee
  - —Members of the panel/this tutorial
  - —Other site members not on this panel
- Product Realization Process Online Website
  - —http://prp.lanl.gov:8686/
  - —contacts list
- The local DOE Quality Assurance Agency



Question & Answers



- Panel Q&A Format
  - Use microphone
  - Ask question - may direct to a panel member
  - One or more panel members will [attempt] to answer the question
  - Audience participation (limited but possible)
- Wrapup
  - Product Realization Process Online
  - http://prp.lanl.gov:8686/

The Urgent Need for Configuration Management



Merant presents

Susan Dart

Software Quality Forum 2000

April 18, 2000

In the News Recently

- GM-AOL portal
- Ford-Yahoo portal
- AOL-Time Warner merger
- NorthWest Airlines web problem
- Grainger ERP problems
- SAP problems

© 2000 Susan Dart

Overview

- Why the urgency?
- Why the need for CM?
- Best practices
- The Future of CM tools

© 2000 Susan Dart

Question

Why is there an urgent need for CM these days - more so than in the "old" days?

© 2000 Susan Dart

Why the Urgency?

- Internet drivers
- Success drivers
- Pain drivers
- Process drivers

© 2000 Susan Dart

Internet Drivers

- **Be competitive**
  - Charles Schwab, Amazon won
  - Merrill Lynch, Wal-Mart, Time-Warner failed
- **Join the "gold rush"**
- **Domination of Internet economy**
- **Maintain growth rate**
- **D-commerce (device)**

© 2000 Susan Dart

Right Now, Companies are Busy with Web Site Creation

- Security problems
- Traffic monitoring
- Network infrastructure

But, inevitably ...

© 2000 Susan Dart

Companies Hit a Crisis Point

Proliferation of quickly "hacked together" web-based systems kept running via a continual stream of patches developed without any rigorous or systematic approach

© 2000 Susan Dart

Shock Follows Initial Excitement

- **Limits of usability, scaleability**
- **Unmaintainable web systems**
- **Cannot respond to change**
  - Growth
  - Mergers
  - Redesign
  - Spin offs
- **Mistakes are made**

© 2000 Susan Dart

Question

What mistakes regarding software development and maintenance, or web content management, have you seen?

© 2000 Susan Dart

Well-publicized Problems

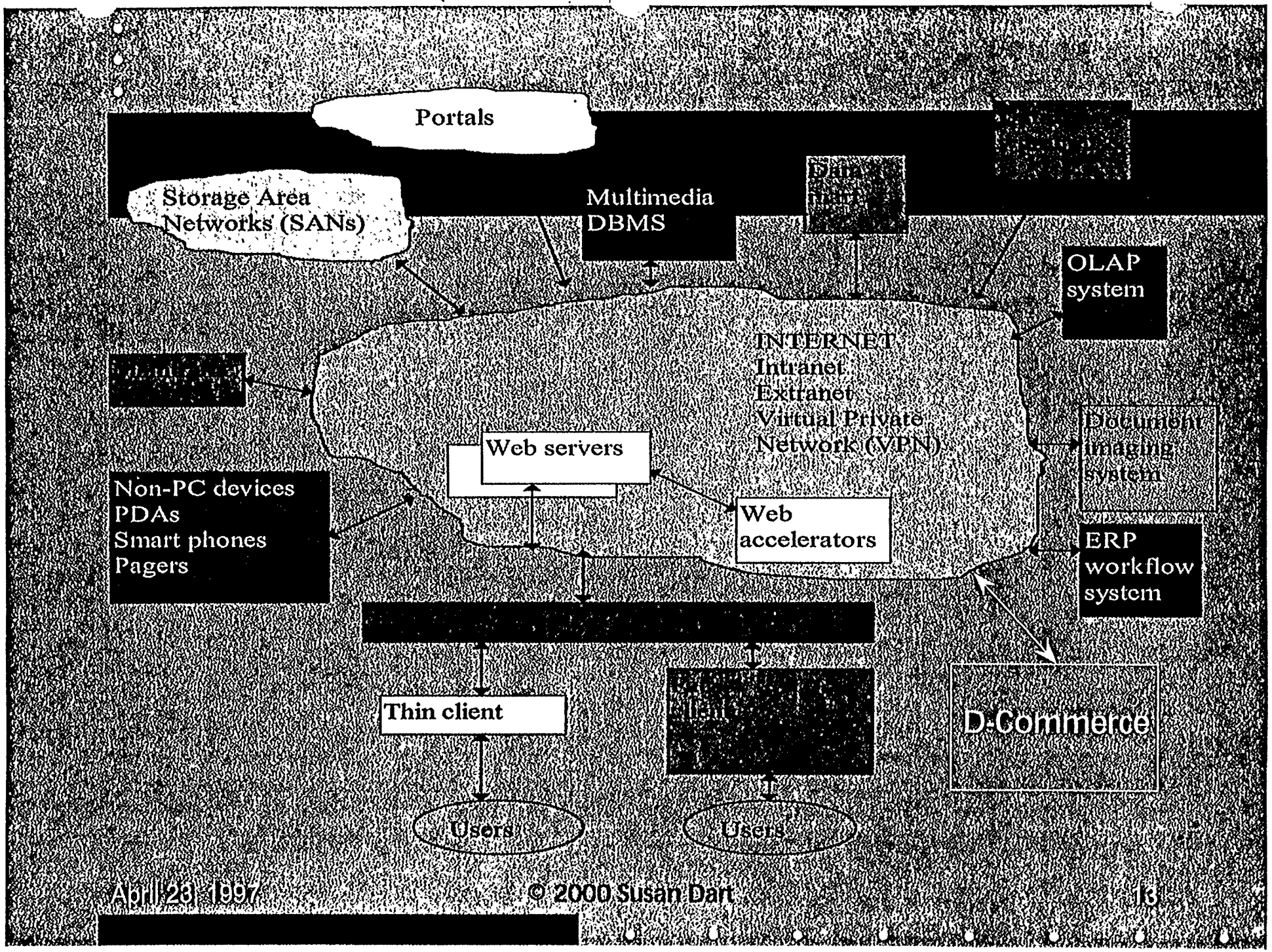
- e-Schwab
- Merrill Lynch
- Encyclopedia Britannica
- e-Bay
- Yahoo
- Election results

© 2000 Susan Dart

Publicly Visible Mistakes

- Inappropriate content
- Wrong timing
- Inaccurate content
- Top secret
- Corrupt
- Unauthorized
- Untested
- Stale
- Inconsistent

© 2000 Susan Dart



© 2000 Susan Dart

Nine Key Challenges

1. Speed of change
2. Variant explosion
3. Dynamic content
4. Process
5. Performance effect
6. Scaleability
7. Outsourcing
8. Politics
9. Immaturity

© 2000 Susan Dart

Speed of Change

Problem:

- Fast change
- Frequent change
- Types of changes
- Re-designs, spin-offs, mergers

CM provides:

- change management
- audit trail

© 2000 Susan Dart

Variant Explosion

- **Problem:**
  - Too many variants
  - Sharing code
  - Propagate changes
- **CM provides:**
  - Parallel development
  - Concurrent baselines

© 2000 Susan Dart

Dynamic Active Content

- Dynamic = Created on-the-fly
- Active = executed
- HTML, ActiveX, Java, XML, VRML
- Component libraries
- CM provides
  - snapshots
  - configuration items
  - build management

© 2000 Susan Dart

Various objects

- **Data objects**
  - Files
  - Documents
  - Images
  - Streaming audio/video
- **Component libraries**
- **Meta-data**
- **CM provides repository**

© 2000 Susan Dart

Free-form Development Process

- Change on-the-fly attitude
- Code and publish
- Lack of release cycle
- CM provides:
  - change cycles
  - release cycles
  - authorizations, notifications

© 2000 Susan Dart

Performance Effect Challenge

- **Response time to user's request**
- **Traffic monitoring**
- **Web accelerators & caching**
- **CM provides**
  - legacy (hybrid) support
  - parallel development

© 2000 Susan Dart

Scaleability Challenge

- **Millions of pages**
- **Thousands added per day**
- **Replication & synchronization**
- **Terabytes of databases**
- **CM provides**
  - baselines
  - repeatable builds
  - repository

© 2000 Susan Dart

Outsourcing Challenge

- Corporate trend
- Web creation
- Distributed management
- CM provides
  - process
  - reports
  - change propagation

© 2000 Susan Dart

Political Challenge

- Role of IT changing
- Business unit independence
- Who is the owner & in control?
- Who can publish to live site?
- CM provides:
  - process
  - audit trails

© 2000 Susan Dart

Immaturity Challenge

- First generation tools
- Non-software authors
- Evolving standards e.g.,, XML
- CM provides
  - version everything
  - query & reports
  - repeatable builds

© 2000 Susan Dart

Question

How can success create problems?

© 2000 Susan Dart

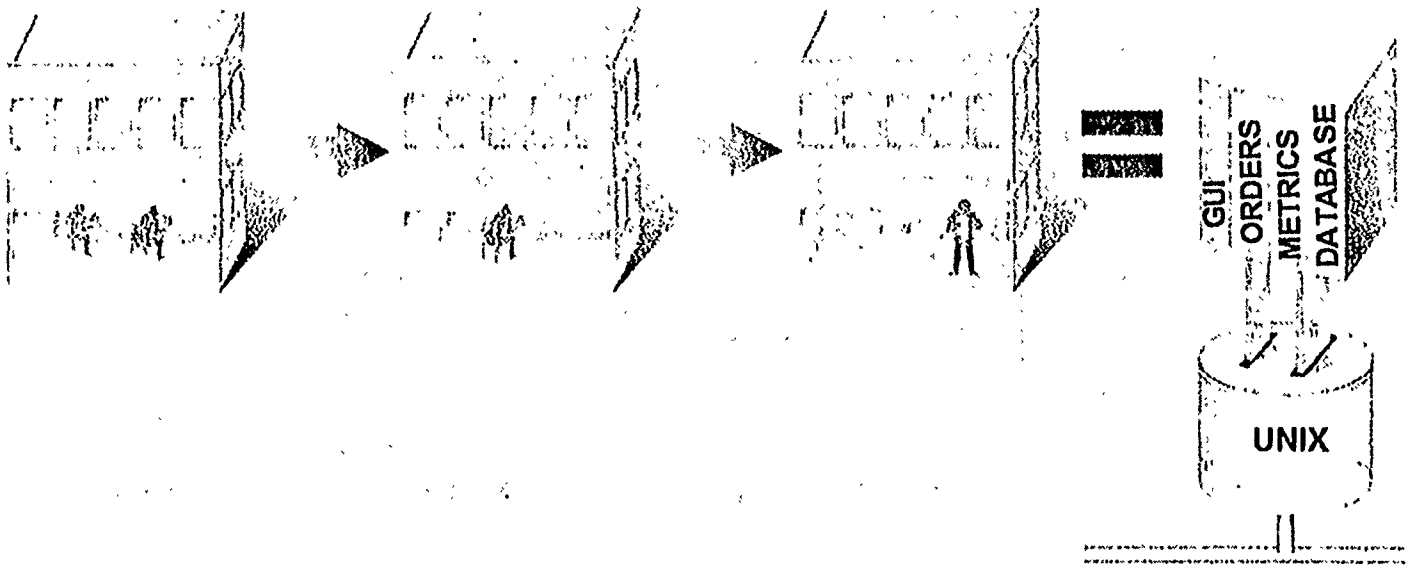
Success Drivers

- Incredibly fast growth
- Too many customers
- Technology evolution
- Legacy management
- Web content management

© 2000 Susan Dart

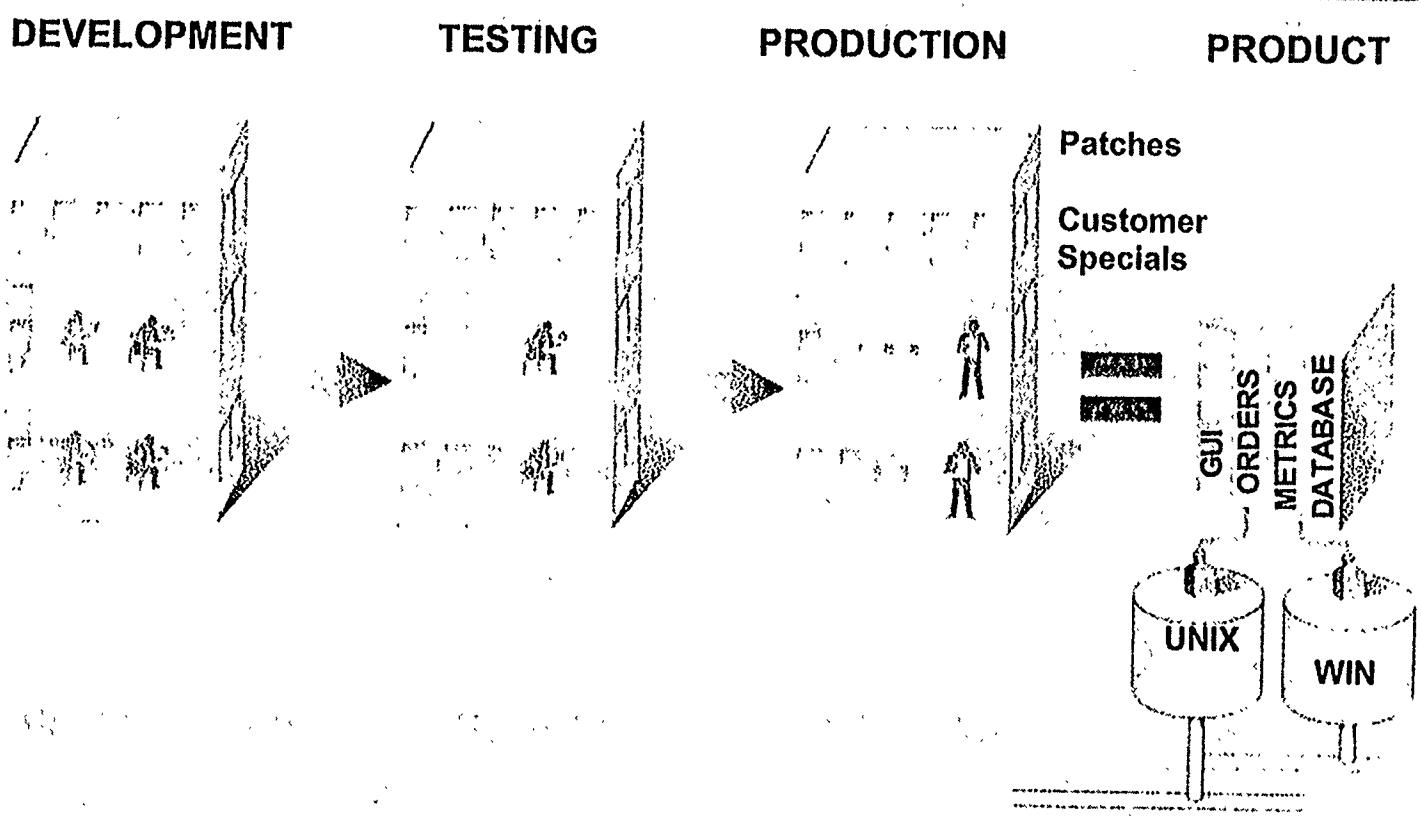
Success Breeds Complexity

DEVELOPMENT TESTING PRODUCTION PRODUCT



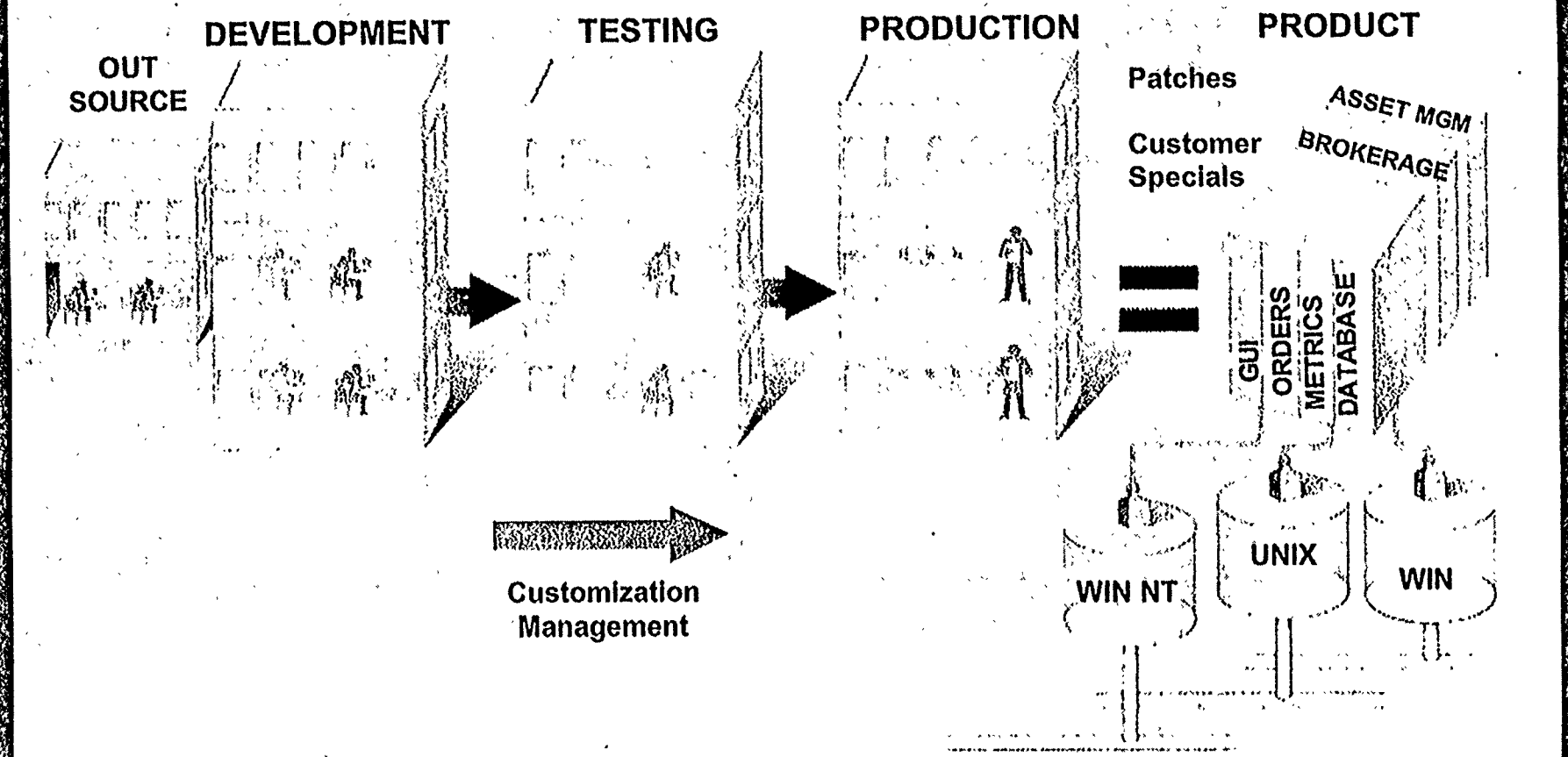
© 2000 Susan Dart

Year Two Being Successful



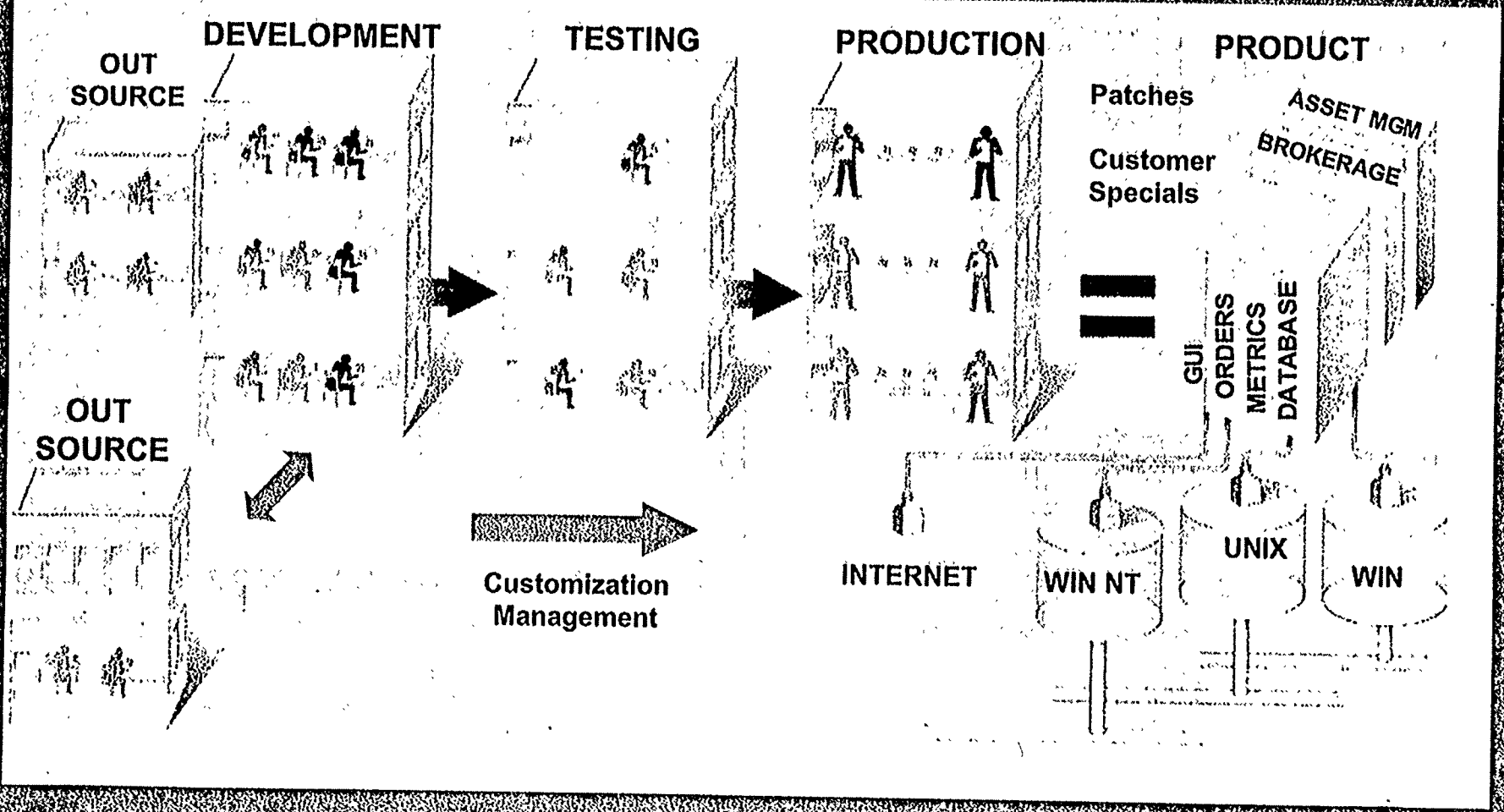
© 2000 Susan Dart

Year Three Being Successful



© 2000 Susan Dart

Year Four Being Successful



© 2000 Susan Dart

Question

How do you know if your company has a CM problem?

© 2000 Susan Dart

Pain Drivers

- Cannot meet deadlines
- Cannot release multiple fixes
- Do not know what went into a release
- Missing features in products
- Developers fix same bug
- Do not know what's been tested
- Have no visibility into work status
- Cannot quantify quality
- Canceled projects

© 2000 Susan Dart

Question

What role does process and process improvement play in CM?

© 2000 Susan Dart

Process Drivers

- **CMM**
  - level 2 Key Process Area
- **CMMI**
  - Data management KPA
  - CM for all processes
- **ISO 9000**
- **Audit failures**

© 2000 Susan Dart

CM Tools Assist Dramatically

- Plans, standards, procedures under CM control
- Tailored processes
- Status of work objects
- Change management, defect tracking
- Notifications, approvals
- Dependency tracking
- Reports and metrics
- 15 minute audits
- Out-of-the-box process

© 2000 Susan Dart

CM Issues

- Respond to drivers/catalysts
- Web programming is different
- What is CM?
- Evolutionary Vs. Full-Process CM
- Business value of CM
- Best practices
- The future of CM tools

© 2000 Susan Dart

Question

Is web programming different from traditional software development?

© 2000 Susan Dart

Web Programming is Different

- Higher volume of users and traffic
- Security needs: access, data
- Distribution of services
- Internationalization
- Asynchronous programming
- Stateless programming
- Component reuse
- Browser, platform transparency

© 2000 Susan Dart

What is CM?

A disciplined approach to managing the evolution of software development and maintenance practices and their products.

© 2000 Susan Dart

Standard Definition

- Identification
- Control
- Status accounting
- Audit and review

© 2000 Susan Dart

Operational Elements

1. Version control
2. Parallel and distributed development
3. Build and release management
4. Workspace and repository management
5. Change tracking
6. Status reporting
7. Audit control
8. Environment support

© 2000 Susan Dart

Evolutionary CM

- Pieces of CM solution
- Easy and quick start
- Example: PVCS Tracker, PVCS Version Manager

© 2000 Susan Dart

Full-process CM

- Process automation
- A comprehensive solution out-of-box
- Example: PVCS Dimensions

© 2000 Susan Dart

CM IS Needed for ...

- Environment
- Content
- Applications
- Legacy code/data
- Third-party code/data
- Databases
- Web crisis

© 2000 Susan Dart

Business value of CM

- Eliminate mistakes
- Recover from crashes
- Cost-effective code building
- Quality automation
- Teamwork optimization
- Change complexity management
- Enables survival and growth

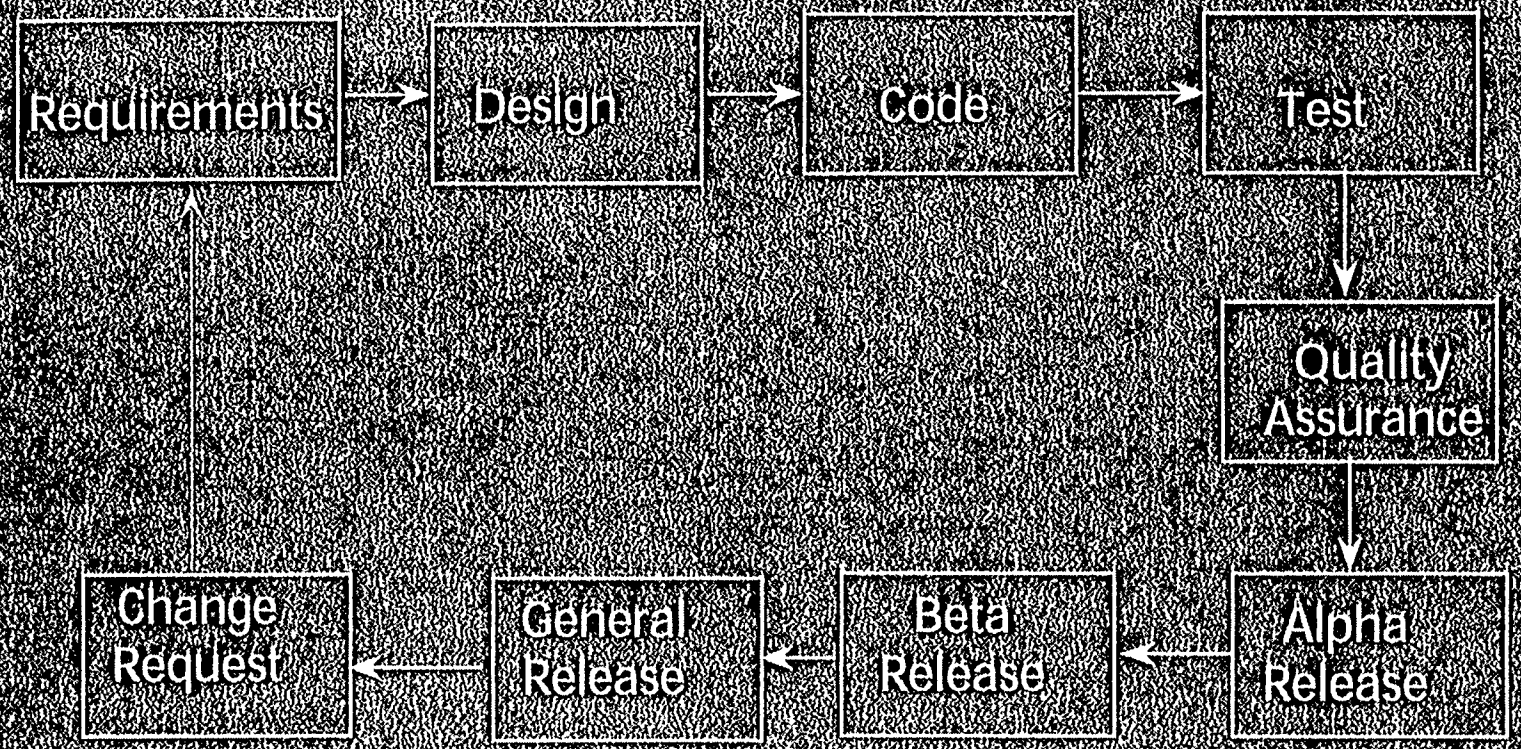
© 2000 Susan Dart

Question

What do we mean by a "best practice"?

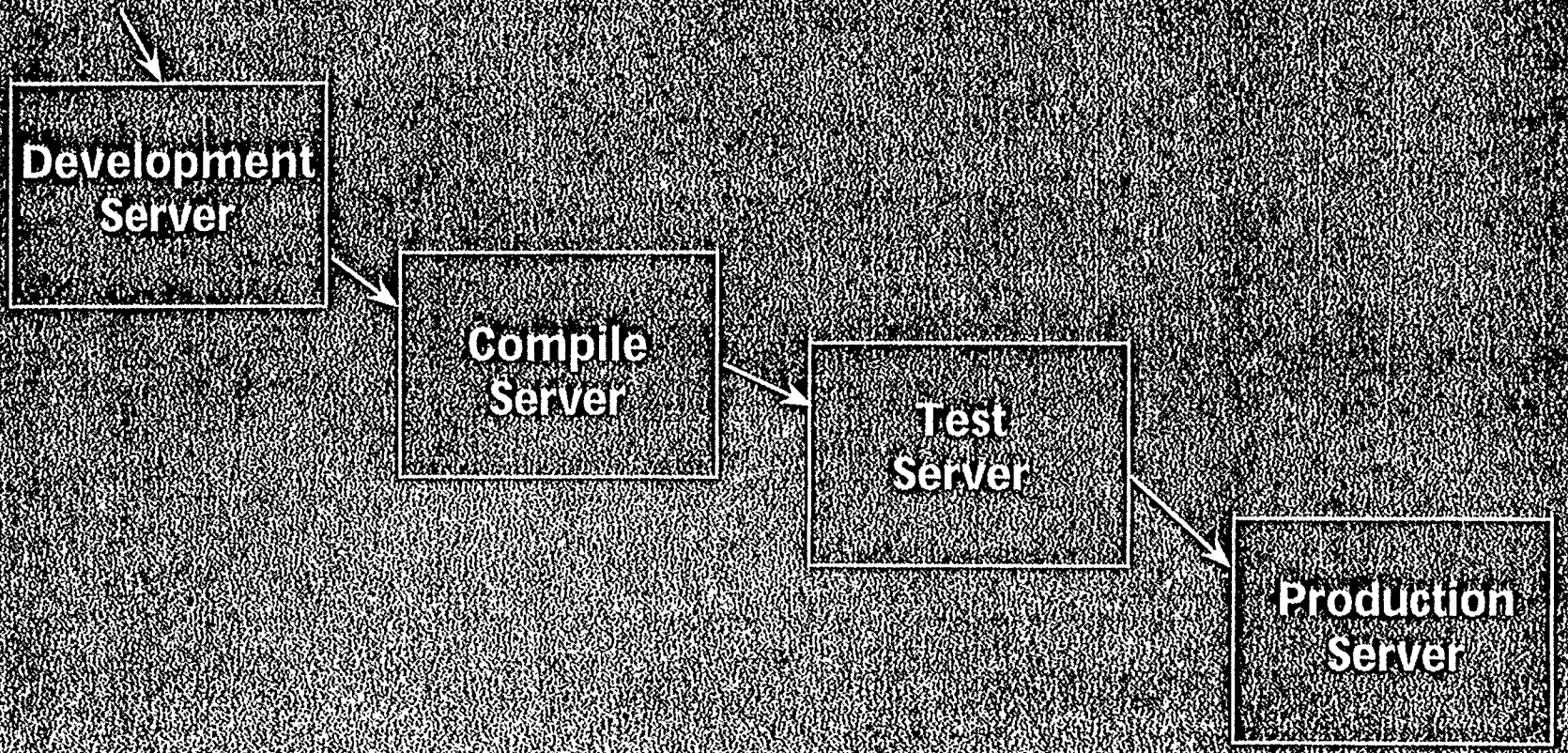
© 2000 Susan Dart

Best Practices: Complete Lifecycle Support



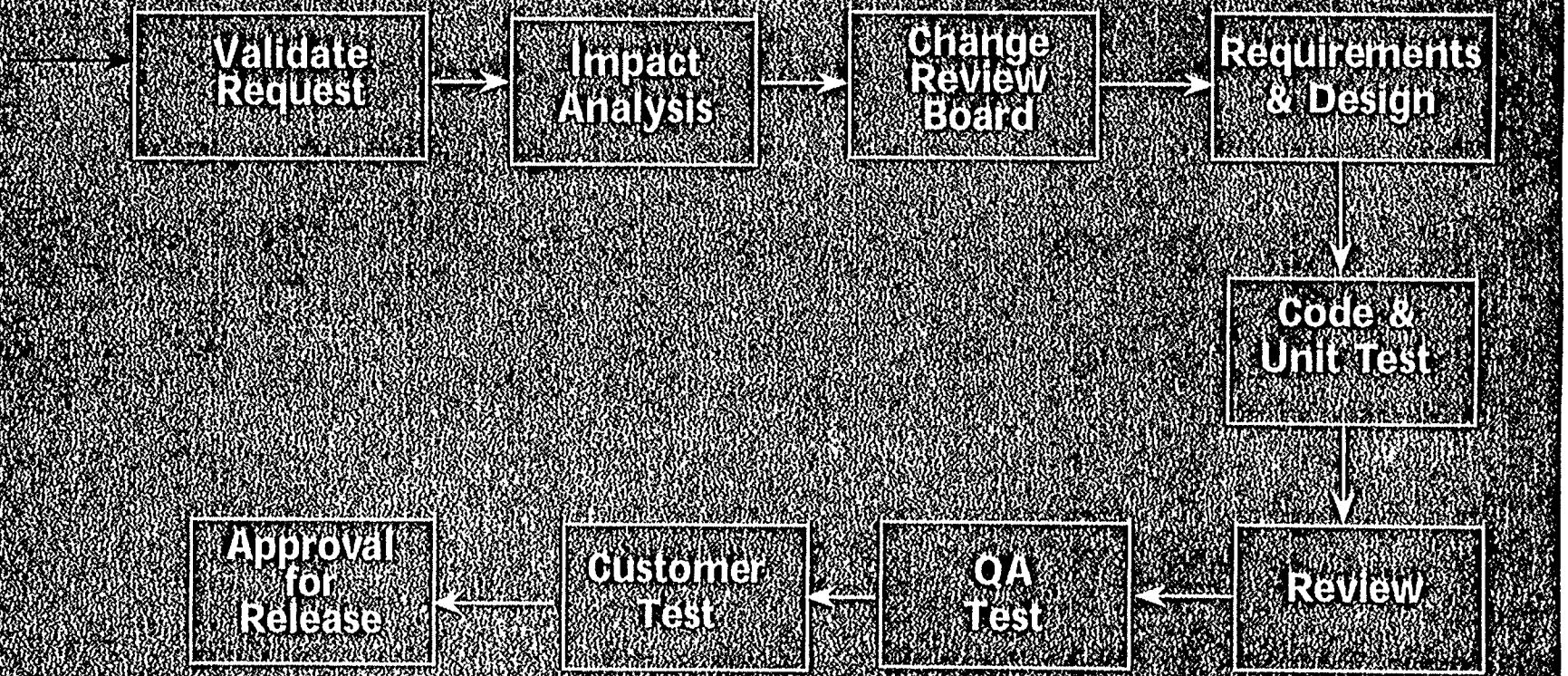
© 2000 Susan Dart

Typical Coding Cycle



© 2000 Susan Dart

Typical Change Request Cycle



© 2000 Susan Dart

Future of CM

- Resurrection of CM industry
- Unified model of CM
- What becomes the differentiator?
- Services
- Standards support
- Pro-active sell
- Solve web problems

© 2000 Susan Dart

Future - 2

- Partnerships with vendors
- CM as the core
- Multiple CM tools in enterprise
- Interoperable CM tools
- Plug-&-play CM modules
- Embedded CM
- Combined legacy & web content support

© 2000 Susan Dart

Conclusions

- "Bricks and clicks" companies cannot survive, let alone thrive without CM
- CM tools support the fast, changing business strategies
- CM works well for web content

© 2000 Susan Dart

Susan Dart's New CM Book

"Configuration Management: The Missing Link in Web Engineering"

1. The Internet is a Fabulous Invention
2. The nature of the web crisis
3. Understanding CM
4. CM Tools
5. Selecting & Deploying a CM tool
6. Case studies
7. Templates and Checklists

© 2000 Susan Dart

Further Contact

Susan Dart
President
Dart Technology Strategies Inc.
1280 Bison, PMB 510
Newport Beach, CA. 92660. USA
Phone: +1 (949) 224-9929
Fax: +1 (949) 515-4442
Email: sdart@susandart.com
Web: www.susandart.com

© 2000 Susan Dart



So I Believe Configuration Management Is a Good Idea Now What?

Marjorie McCornack
Decision Support Systems Business Area
Information Systems Engineering Center
Sandia National Laboratories

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Sandia National Laboratories



What Will be Discussed

- My Background and Current Work
- What I Mean By Configuration Management
- A Roadmap for Establishing a Configuration Management Infrastructure
- Factors To Think About When Establishing a Configuration Management Infrastructure
- Making Configuration Management Palatable to Reluctant Developers



My Background

- BS in Mathematics, MS in Computer Science
- Employed at Sandia National Labs for 16+ years
- Worked on various software development projects in various capacities
  - types of projects: satellite ground system, seismic ground system, non-proliferation info system, FAA info system, endgaming simulation & modeling system
  - types of assignments: DBA, testing, developer, information modeling, configuration management, project lead



Background in Configuration Management

- No formal training in configuration management
- First (implicit) introduction to configuration management was when I was doing testing (14 years ago)
  - we had a testbed but no defined way to get the software from development to the testbed
  - developers kept changing the software on the testbed so that I couldn't reproduce test results
- Have learned and continue to learn by being in the trenches

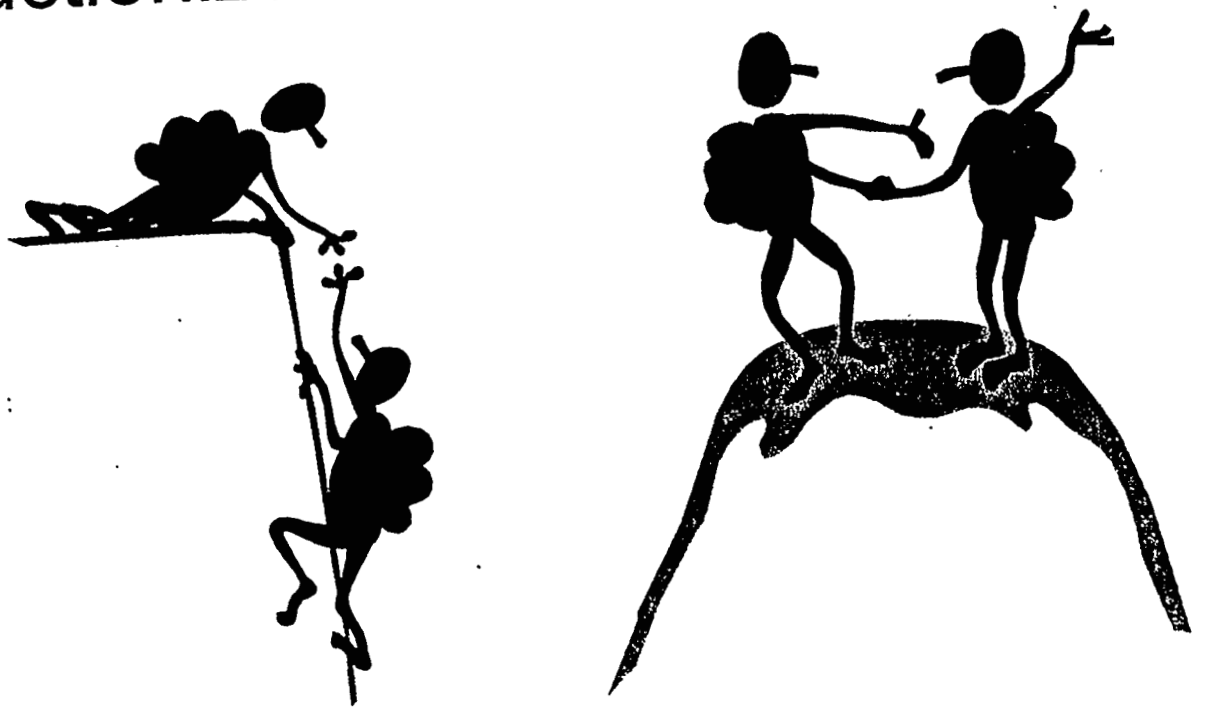


Current Work

- Co-leader of PEARLS team (Productionizing Environments And Releasing Lifecycle Systems)
- My Focus: developing configuration management infrastructures and processes

PEARLS Vision

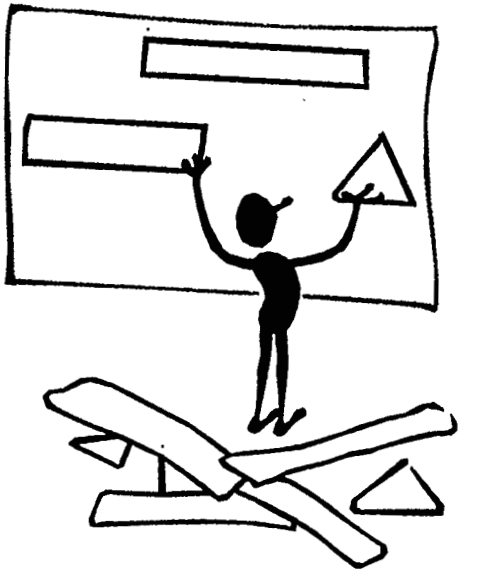
Facilitate internalization of productionization concepts



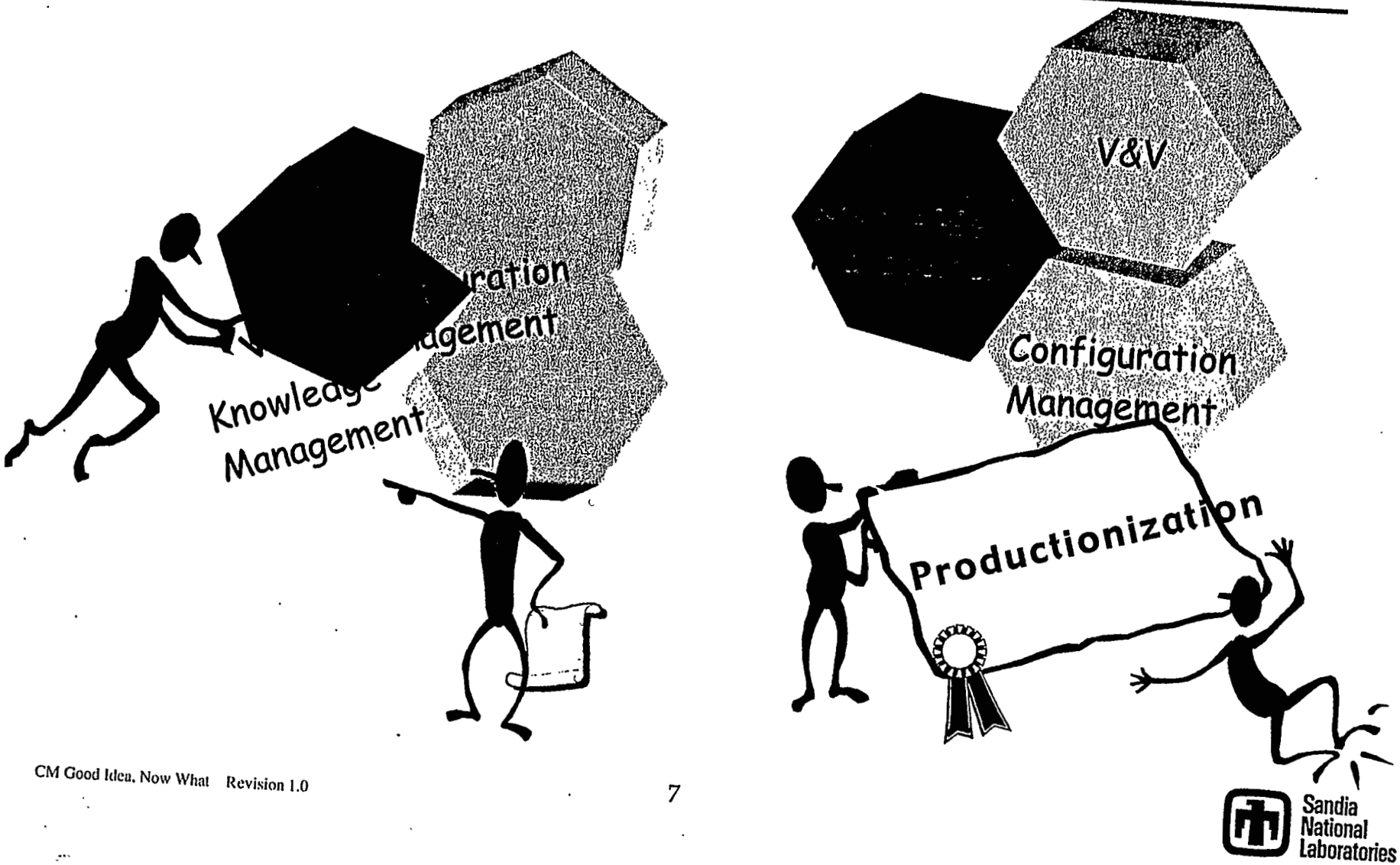
Sandia National Laboratories

Definition of Productionization

Establish an infrastructure identifying the elements necessary to take systems from concept to product

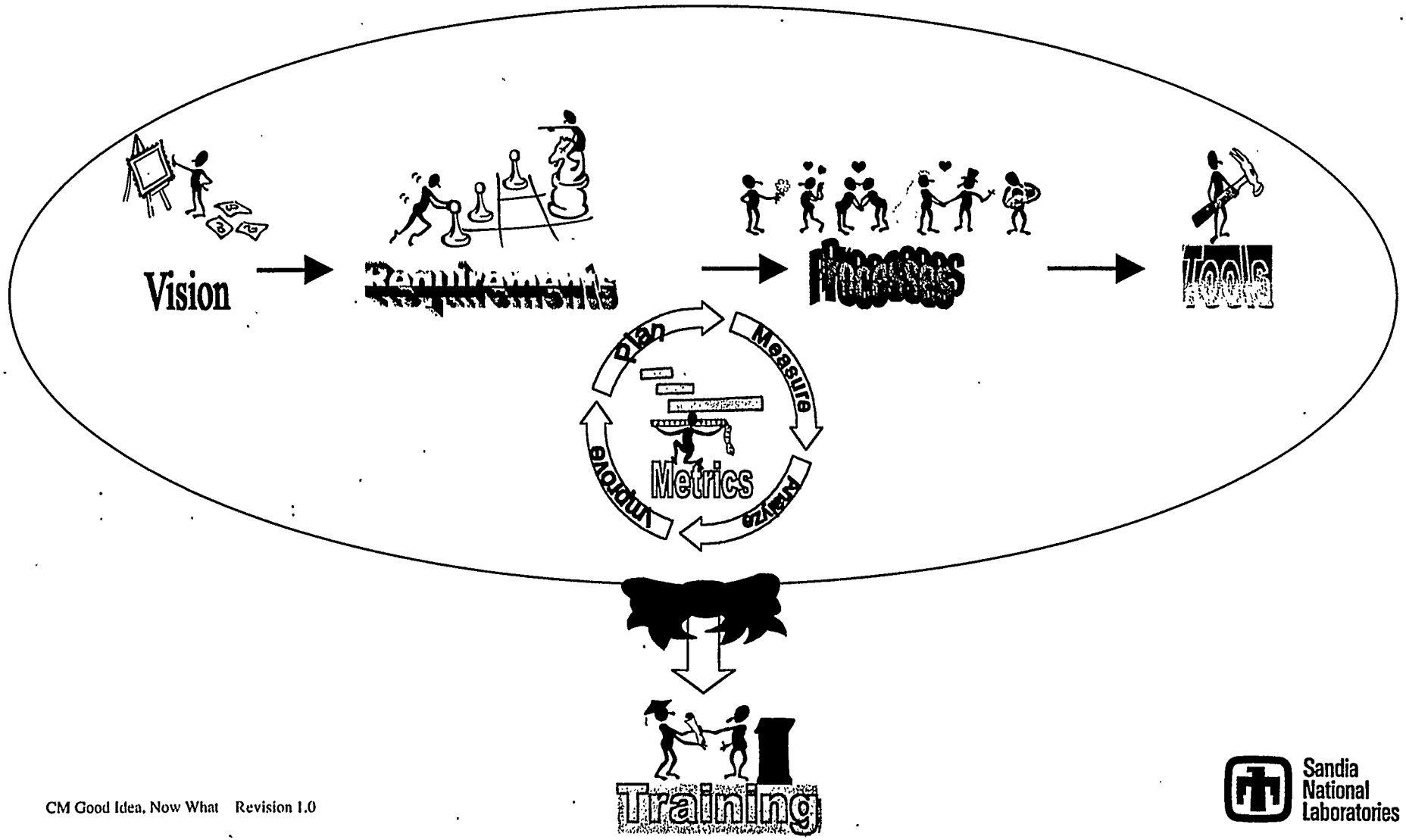


Productionization Facets



Sandia National Laboratories

How We Focus the Customer



Sandia National Laboratories



What Is Configuration Management (CM)?

- **A formal definition of configuration management is**

  the discipline of identifying the configuration of a system at discrete points in time for purposes of systematically controlling changes to this configuration and maintaining the integrity and traceability of this configuration throughout the system life cycle.[1]

[1]Edward H. Bersoff, Vilas D. Henderson, Stanley G. Siegel, *Software Configuration Management An Investment in Product Integrity* (Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1980).



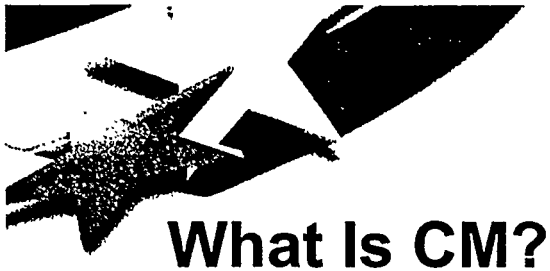
What Is CM?

- As a management support function CM defines procedures to
  - control changes
  - track development
  - report on development



What Is CM?

- As a development support function CM provides the tools and procedures to
  - maintain components
  - keep a history about components
  - support a stable working environment for a changing product
  - build and deliver a product



What Is CM?

- CM gives stability to product development by providing a controlled and managed environment.
- CM infuses discipline, methodicalness, and accountability into the development process.
- CM contributes to the development and delivery of a quality product.



What Is CM?

- **Bottom line: CM provides controls so that there is stability, traceability, and repeatability.**

- **Keywords:**
  - **control**
  - **stability**
  - **traceability**
  - **repeatability**



CM Has Four Classical Functions

- Prevailing literature and books reference 4 classical functions/disciplines of CM
  - Identification
  - Control
  - Auditing
  - Status Accounting

Sandia National Laboratories



Applied CM Encompasses a Variety of Activities

- Some of the various activities of CM are
  - version control
  - change control
  - requirements management*
  - work environment management
  - build process
  - issue tracking*
  - release management
  - product work flow

*Some people might categorize these activities under Knowledge Management.



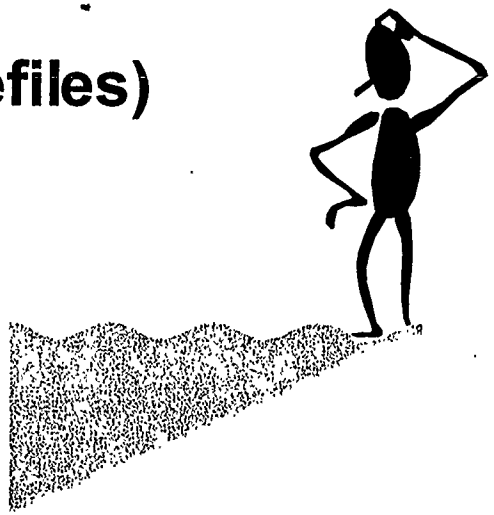
Where Does a Project Start With CM?

- Management/project leadership support and advocacy is the most important first step for implementing CM.
- Management/project leadership support needs to be more than just verbal acknowledgement of the importance of CM.



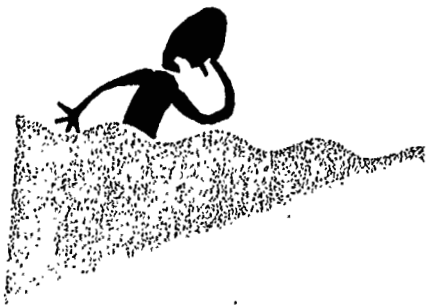
A Roadmap For Introducing CM Elements: Feet Wet

- version control
- directory architecture
  - sandboxes
  - shared areas
- build scripts (for example, Makefiles)



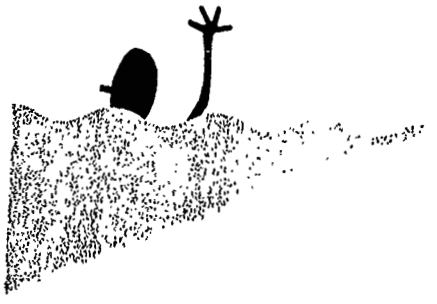
A Roadmap For Introducing CM Elements: Up to the Waist

- release process
  - roles and responsibilities
  - release numbering
- promotion model
- issue tracking
- managing shared components



A Roadmap For Introducing CM Elements: Submerged

- release management
  - implementing release process
  - tracking distributions
  - controlling releases
- branching
- change control/planning
- controlling COTS, open source
- refining different work environments
  - development
  - test or QA





A Couple Things to Keep in Mind

- Start with the activities/elements with which you are comfortable or where you already have details defined.
- Define standards as early as possible and enforce them.
- Realize that you probably will have to re-do or redefine some of the decisions and processes, so schedule for it and don't fret about it when it happens.



Factors To Think About

- The next several slides present different ideas, tips, questions, and concepts that I find myself discussing with the various projects that I support or with which I have consulted. They are presented to demonstrate what kinds of issues need to be considered when designing and implementing a configuration management infrastructure and to generate discussion.

- The factors presented are the ones I have encountered the most or I have found important when starting to establish a CM infrastructure.

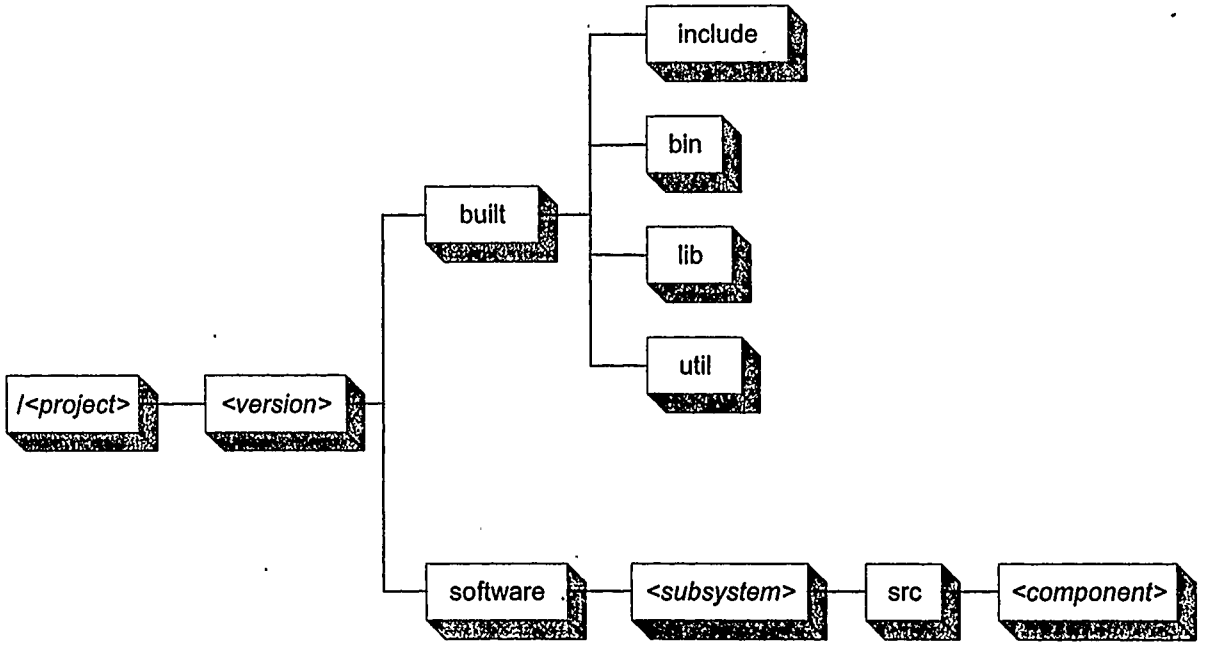


Factors To Think About: Directory Architecture

- Separate directories of "built components" from "develop in" directories
  - build means doing something to source to make it into the file used in the system/application
    - compile
    - preprocess
    - copy
  - files get into built component directories *only* through a build
  - differentiate between the source file and built file for text files (html, scripts)

Factors To Think About: Directory Architecture

- It should be possible to describe the directory architecture generically.





Factors To Think About: Dir Arch & Work Env

- How do you want to separate work?
  - sandboxes via directories or client/server?
  - shared areas on servers or complete duplication?
  - different versions of processes running on servers?
  - separate machines for development, test, production?
    - physical vs. logical separation



Factors To Think About: build process

- Build scripts should have standard targets.
- Files used in a build, such as include files and libraries, should be used from a "built" directory.
  - implies that these files need to be built first in the build process
- Text files (html, scripts) can be pre-processed.



Factors To Think About: version control

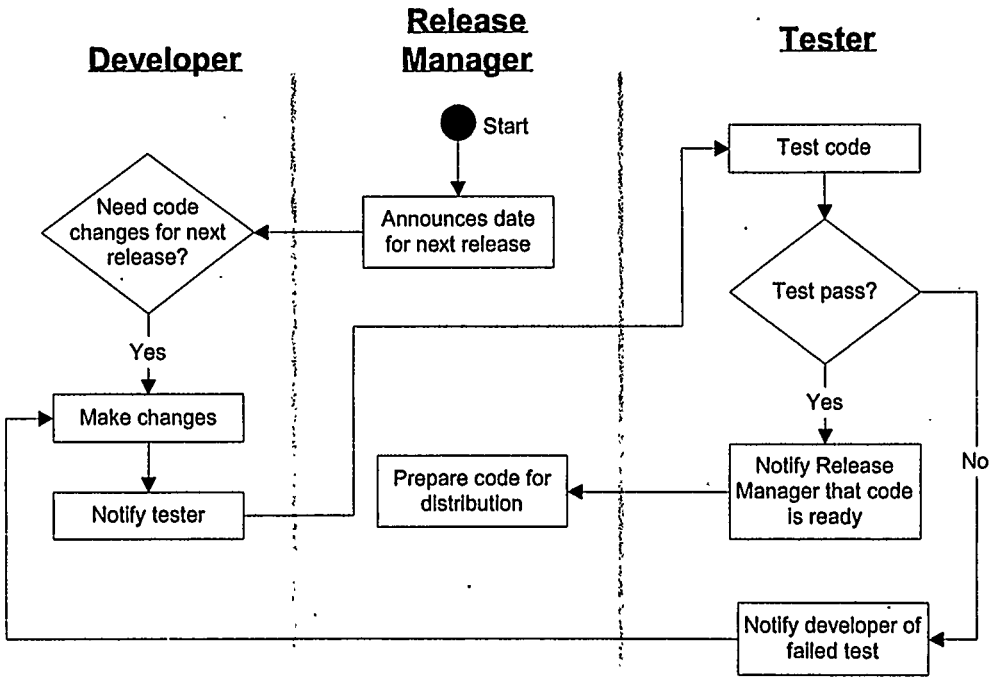
- If you are going to allow multiple locks on a revision of a file, make sure you are not doing it as a work around to a more fundamental problem. For example:
  - poor modularization of the application
  - reluctance to have design reviews
- Design your version control to support incorporating commercial and open source* software updates.
- Design the version control to allow developers to have incremental check ins.

*just because it's open source doesn't mean it doesn't have to be controlled

Sandia National Laboratories

Factors To Think About: release process

- "Swim lane" flowchart diagrams can be a good way of illustrating a process.





Factors To Think About: release process

- Establish a release number format so that releases can be identified consistently in reports, reviews, tracking systems, and so forth.
- Design the release process to include incorporating new releases of commercial and open source* software.

*just because it's open source doesn't mean it doesn't have to be controlled



Factors To Think About: branching

- If you continue to add features to a system/application that has been released, then you probably will need to use branching.
  - to fix a bug in a file, you should modify the revision that was part of the release and not a revision that has new features already added
- Branching implies merging unless it is an experimental branch.
- Watch out for implicit branching outside the version control system.



Making CM Palatable to Reluctant Developers

- Do not oversell
  - CM will add to the developers' responsibilities
  - From a developer's viewpoint, CM does not necessarily make product development "easier"
  - CM will not (and should not) be invisible to the developer
- However, a selling point
  - a CM infrastructure does free the developer from some hassles, for example, the stable work environment makes debugging easier



Making CM Palatable to Reluctant Developers

- If a problem results because a developer does not follow defined CM procedures, make sure that developer "feels the pain".
  - Corollary: don't reward developers as heroes because they had to make heroic fixes to problems caused by circumventing CM
- Have the reluctant developers be responsible for some crucial elements of CM such as building the system/application.

Managing Software in the Highly Regulatory Environment of the WIPP Project

Gary K. Froehlich

(505)284-3930, gkfroeh@Sandia.gov

Harvey C. Ogden

(505)844-1742, hcogden@Sandia.gov

Distributed Decision Support Systems

Department 6535

Sandia National Laboratories

Sandia National Laboratories

Managing Software in the...WIPP Project
Contents (1)

- Description of WIPP
- SNL's role in WIPP
- Performance assessment
- Origins of our QA requirements
- The requirements, summarized
- Our definition of quality
- Application of our definition
- Where we began
- Implementation of software QA
- SQA documentation

Sandia National Laboratories

Managing Software in the...WIPP Project Contents (2)

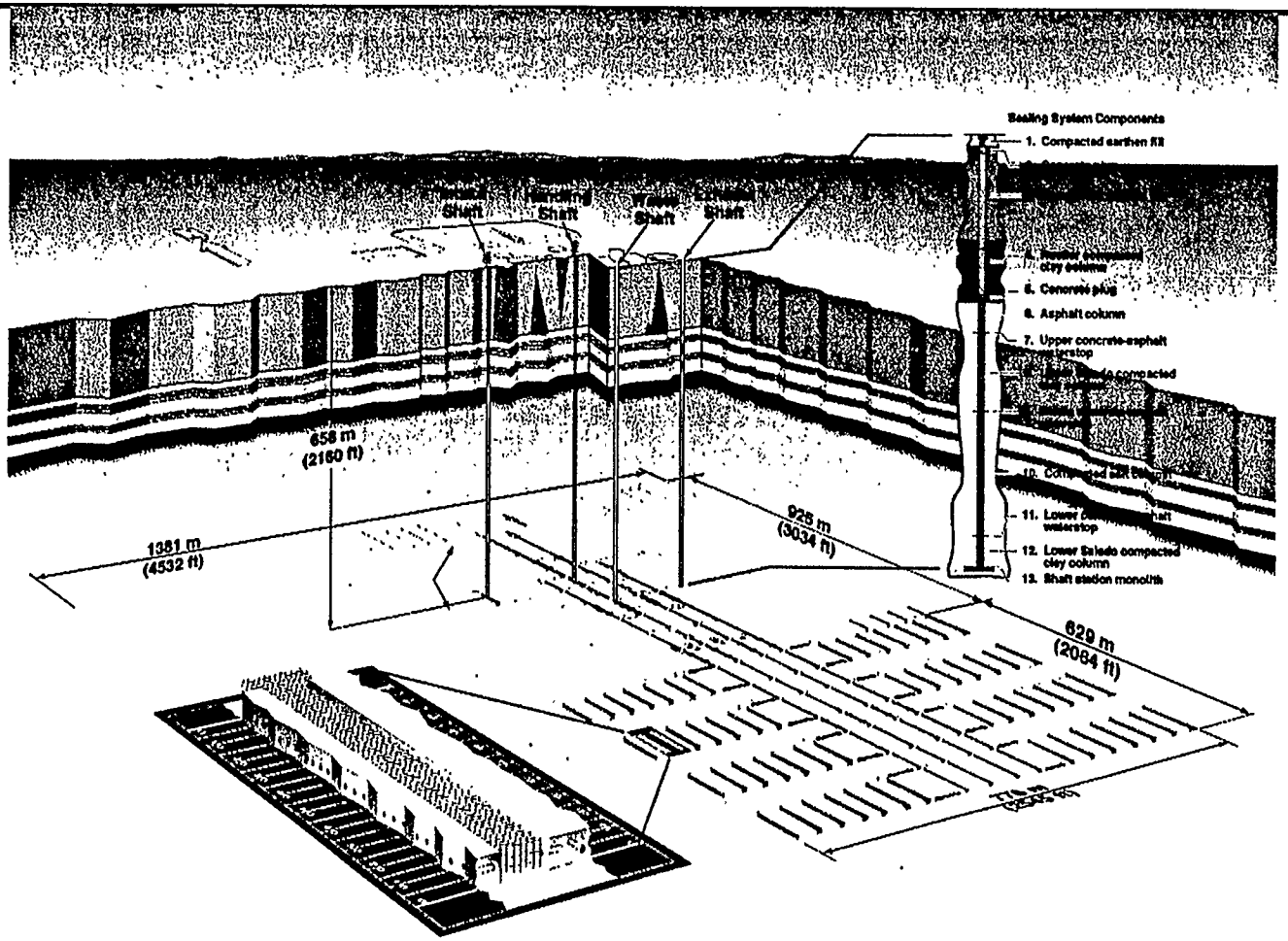
- SQA -- benefits
- Our definition of software configuration management
- Implementation of software CM
- Our definition of run control
- Implementation of run control
- SCM and Run control -- benefits
- How well it all worked
- Lessons learned

Sandia National Laboratories

The Waste Isolation Pilot Plant -- WIPP

- The Department of Energy's (DOE) WIPP is a deep geologic repository for the permanent disposal of spent nuclear fuel, high-level waste, and transuranic (TRU) waste
- It is the first such repository in the United States to have successfully demonstrated compliance with the US Environmental Protection Agency's (EPA) long-term radioactive waste disposal requirements
- First disposal of TRU waste at WIPP occurred on March 26, 1999

Cutaway Schematic of the WIPP



MSOffice\Powerpnt\GKF_work\SQF_paper•3/30/00• slide5

Sandia National Laboratories

SNL's Role in WIPP

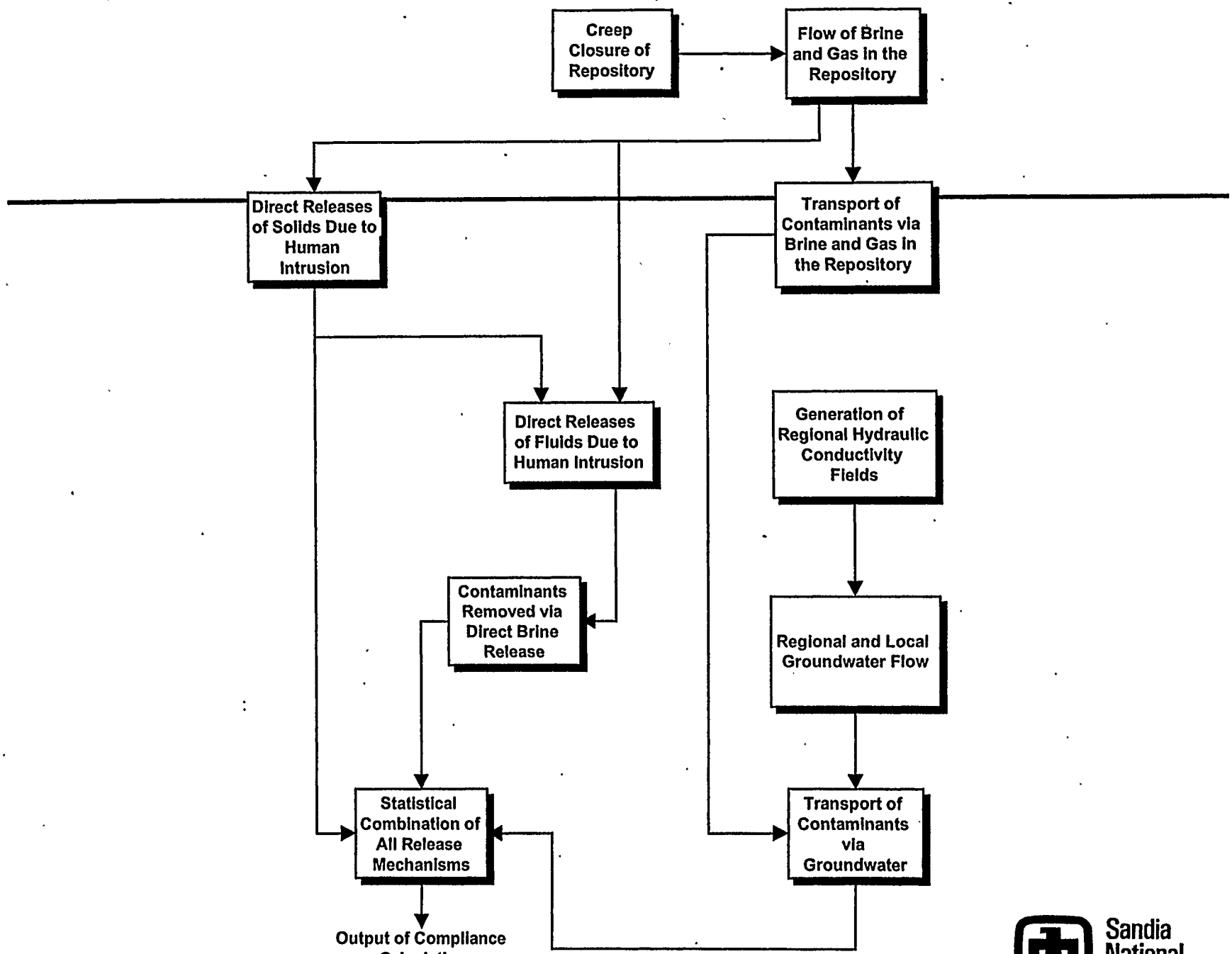
- **Science Advisor to the DOE**
- **Teamed with:**
  - **DOE Carlsbad Area Office (CAO)**
  - **Westinghouse Electric Corp. (WID)**
- **Responsibilities included:**
  - **Data collection, site characterization, model development**
  - **System Prioritization -- a determination of the relative importance of scientific activities with respect to their impact on regulatory compliance**
  - **Performance Assessment (PA) -- a risk-based analysis of whether the repository will perform as expected over its 10,000-year regulatory period**

Performance Assessment for WIPP (1)

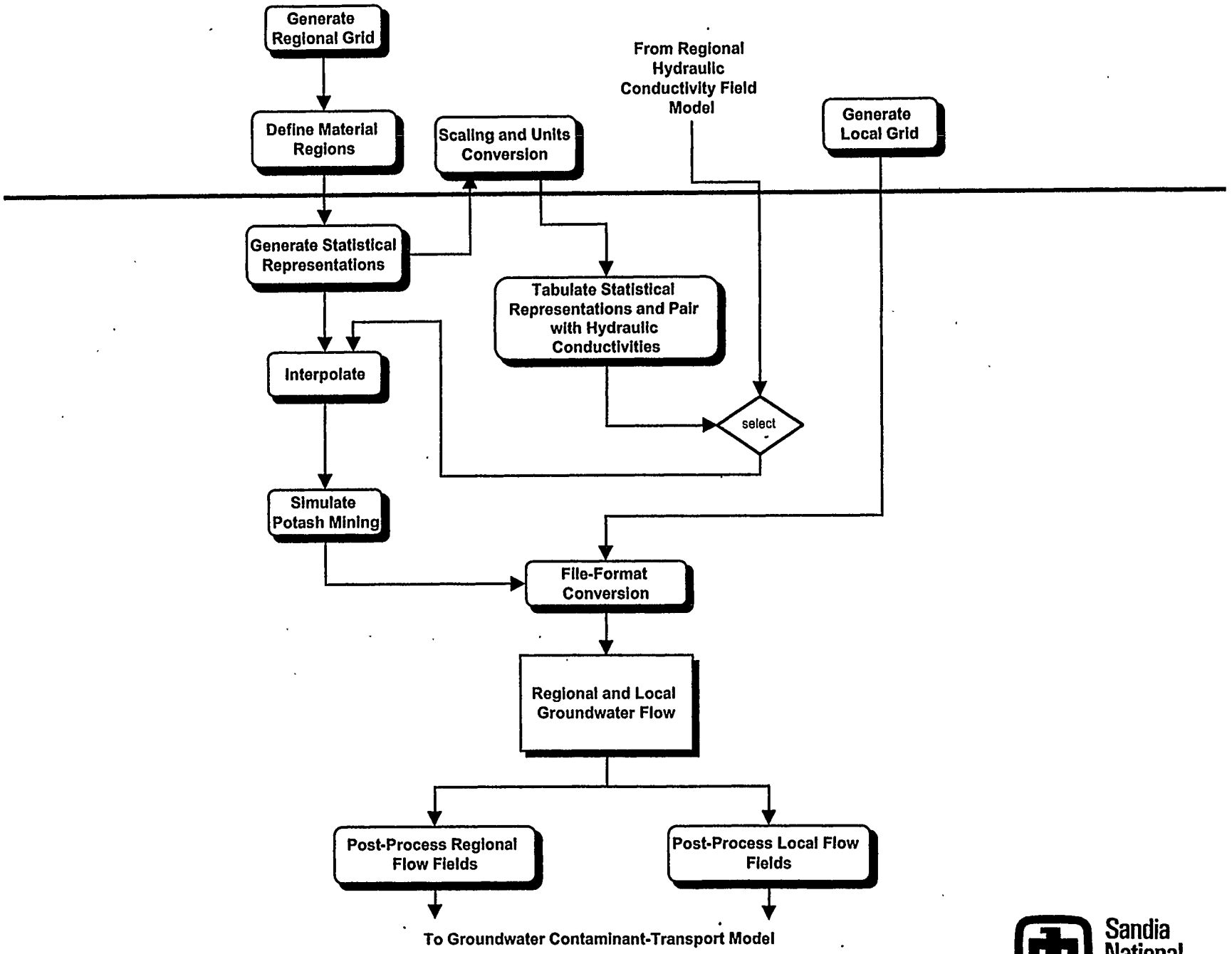
- **Characterization of the repository system**
  - Interpretation of field and lab data to describe site-scale phenomena
  - Modeling of interactions between the waste and repository and surrounding geology
- **Simulation of repository-system performance over a 10,000-year period**
  - Modeling codes: geohydrology, rock mechanics, geochemistry, two-phase flow and transport, etc.
  - Utility codes: coordinate transformations, unit conversions, file-format manipulation, etc.
- **Consideration of possible future scenarios**
  - Undisturbed
  - Human intrusion: mining, drilling

Performance Assessment for WIPP (2)

- **Sources of uncertainty**
  - Modeling uncertainties
  - Variability of physical phenomena (e.g., heterogeneity)
  - Many possible futures
- **Comparison of predicted performance with release limits**
  - Stochastic performance measure
  - Number of simulations dictated by required confidence
- **Large number of complex codes, with complex interactions**
- **Large number of simulations, due to probabilistic nature of problem**



Sandia National Laboratories



Sandia National Laboratories

The Origins of Our QA Requirements

- **1992 -- Land Withdrawal Act**
  - Congress named DOE manager of WIPP
  - EPA named as regulator; made responsible for developing disposal regulations and for certifying long-term safety
- **late 1993 -- 40CFR191**
  - Set forth disposal regulations and release limits
  - Told us what we needed to do
- **early 1996 -- 40CFR194**
  - Established criteria for demonstrating compliance with standard 40CFR191
  - Invoked NQA 1, 2, and 3
  - Told us how we needed to do it

The Requirements, Summarized

- Our testing, reviews, analyses, and documentation "shall be sufficiently detailed as to purpose, method, assumptions, design input, references, and units such that a person technically qualified in the subject can review and understand the analyses and verify the adequacy of the results without recourse to the originator."
- We must also provide for software problem reporting, change control, and change tracking

Our Definition of Quality -- T^2R^3

- **Traceable: all relevant steps and components of a process are unambiguously identified**
- **Transparent: the logic and decisions involved in a process are clearly presented**
- **Retrievable: all relevant components of a process are rapidly and easily recovered**
- **Reproducible: a process can be fully reconstructed based on documentation of relevant steps and components**
- **Reviewed: the credibility of each step and component of a process is established by independent technical evaluation**

Sandia National Laboratories

Application of Our Definition

- Balancing T^2R^3 with cost and schedule
- Application of T^2R^3 principles only to the extent necessary to demonstrate "adequacy of intended use" for the process under consideration
- Recognition that, for WIPP, SNL's product is the analysis of repository performance and assistance to DOE in the preparation of the Compliance Certification Application (CCA), <u>not</u> software

Sandia National Laboratories

Where We Began (1)

- Upper-tier requirements derived from DOE Order 5700.6C, but:

- No life-cycle methodology
- Inadequate documentation
- No project-wide configuration management
- Poor reproducibility
- Poor traceability
- Inadequate retrievability

Sandia National Laboratories

Where We Began (2)

- Official QA standards promulgated by 40CFR194, in early 1996 (but rule was proposed in mid-1995)
- Application due October 1996
- Calculation needed to start at beginning of 1996
- Four-month window to revise existing QA Procedures, train staff, and qualify nearly 60 codes! (Serial nature of calculations and SCM helped here)
- In parallel with software QA, there were related activities in qualification of data, analyses, models...

Implementation of Software QA (1)

- **Negotiation with DOE and EPA on interpretation of standards -- applying reason**
  - Begun after promulgation of 40CFR191, but before 40CFR194 was proposed
  - Negotiated interpretations incorporated into 40CFR194
- **Interpretation of and applicability of the NQA series, in particular NQA 2.7, to WIPP**
  - Vast majority of WIPP codes, developed prior to our use of the standard, exempt from full life cycle (legacy codes -- NQA 2.7, paragraph 10.2)
  - Time-critical problem-response criteria not applicable (e.g., restart capability, unintended function, etc.)
  - Code validation possible only by independent technical review (more aptly regarded as model validation)

Implementation of Software QA (2)

- **Responsibilities divided across teams**
  - Code Sponsor -- responsible for qualification of a given code; developed functional requirements, most test cases, and related acceptance criteria
  - Tester -- performed static and dynamic testing, ran all test cases, and compared results to acceptance criteria
  - Documentation Support -- prepared uniform, consistent software QA documentation, under the direction of the above team members
  - Reviewer(s) -- performed and documented independent technical review of adequacy of testing, and reviewed all software QA documentation
- **SQA management team periodically reviewed status with individual code teams**

Sandia National Laboratories

SQA Documentation (1)

- **User's Manual** -- description of code's capabilities; discussion of theory, models, and numerical methods; instructions for execution; description of input and output files; discussion of limitations and assumptions
- **Requirements Document / Verification and Validation Plan** -- description of code's capabilities and functional requirements; description of test cases needed to test all applicable functional requirements; discussion of acceptance criteria for each test; description of the test procedure to be used, necessary input files, what outputs to expect, etc.

Sandia National Laboratories

SQA Documentation (2)

- **Validation Document** -- description of test results; evaluation of results against acceptance criteria (actually verification)
- **Implementation Document** -- listing of the source code; description of build instructions, compiler directives, libraries linked, and platform; listing of actual build logs; description of code connectivity, e.g., subroutine call tree
- **Design Document** -- only required for non-legacy codes (of which there was only 1 in 1996)

Sandia National Laboratories

SQA -- Benefits

- Fewer errors due to formal test and review
- Greater reliability of the final product
- Greater defensibility of results
- SQA records to "back up" our results (i.e., to provide objective evidence)

Sandia National Laboratories

Our Definition of Software Configuration Management (SCM)

The meticulous identification, storage, and ongoing tracking of computer codes from a baseline version through all subsequent versions, *along with* all relevant inputs, outputs, compilation options, library linkages, and any other information needed to faithfully reproduce the most recent or any previous calculation for which a code has been used (whether for testing or for production)

Implementation of Software CM (1)

- Needs identified, software product selected
- Procedures developed, access-control requirements identified
- Three parallel environments provided
  - Development: code sponsors/developers responsible for check-in, check-out, versioning of codes; optional
  - Testing: codes submitted by sponsors for testing, added to software baseline inventory; all official testing performed here
  - Production: once qualified, codes intended for use in official calculations are installed; check-in by authorized CM librarian only

Sandia National Laboratories

Implementation of Software CM (2)

- Input and output files also under software CM
- Executables built in software CM as well, using build scripts
- Rigorous naming conventions enforced
- SQA documentation also in software CM
- Two additional equivalent, but separate, SCM systems maintained in parallel
  - One for EPA to run the WIPP codes for their own evaluation
  - The other for stakeholders and oversight groups to do likewise
  - Provided T^2R^3 for those groups as well -- proved invaluable!

Sandia National Laboratories

Our Definition of Run Control

The automated execution of a suite of codes by those granted the necessary access, including retrieval of all needed codes and inputs from within configuration management, and appropriate disposition of outputs, as well as distribution of the computational load across appropriate and/or available resources

Implementation of Run Control

- Leveraged investment in SCM
- All official calculations run by scripts
  - Codes and inputs pulled from SCM production area
  - Outputs *automatically* installed in SCM
  - Run logs stored in SCM as well
  - Naming conventions permitted automatic generation of file names
- All official calculations executed by "run masters", using only qualified components*
- Also enabled run distribution, for optimum efficiency and use of available resources

*SCM allowed proceeding, at risk, with unqualified codes, to be qualified later

Sandia National Laboratories

SCM and Run Control -- Benefits (1)

- Scalability -- same run scripts to perform 1 or 1000 runs
- Execution scripts verified only once, and then controlled in SCM
- Reuseability -- never assume a calculation will only be done once!
- Assists in planning and scheduling runs
- Relieves some of the burden from code sponsors (both in conducting calculations and responding to regulator questions about them)

Sandia National Laboratories

SCM and Run Control -- Benefits (2)

- Lessens dependence on specific individuals
- Proper access controls and elimination of manual-interaction steps provides better accuracy and defensibility
- Permits rapid distribution of outputs to analysts

How Well It All Worked (1)

- In May 1998, EPA certified WIPP as being in compliance with the applicable disposal regulations!
- WIPP PA codes were qualified in time
  - Over 40 code sponsors
  - 6-8 testers
  - About 6 documentation support staff
- Massive calculation completed on time, with full T^2R^3
  - Five months end-to-end
  - Conducted by 2 "run masters" (once codes and inputs were qualified and installed in SCM)

How Well It All Worked (2)

- **CCA Example**
  - 37,000 CPU-hours (over 4.2 CPU-years)
  - 225,000 files retained (many times that number of intermediate results temporarily retained)
  - Approximately 100 Gigabytes in permanent storage
- **EPA Example**
  - Similar in scope to CCA calculation
  - Conducted by 3 individuals in 3 months for EPA

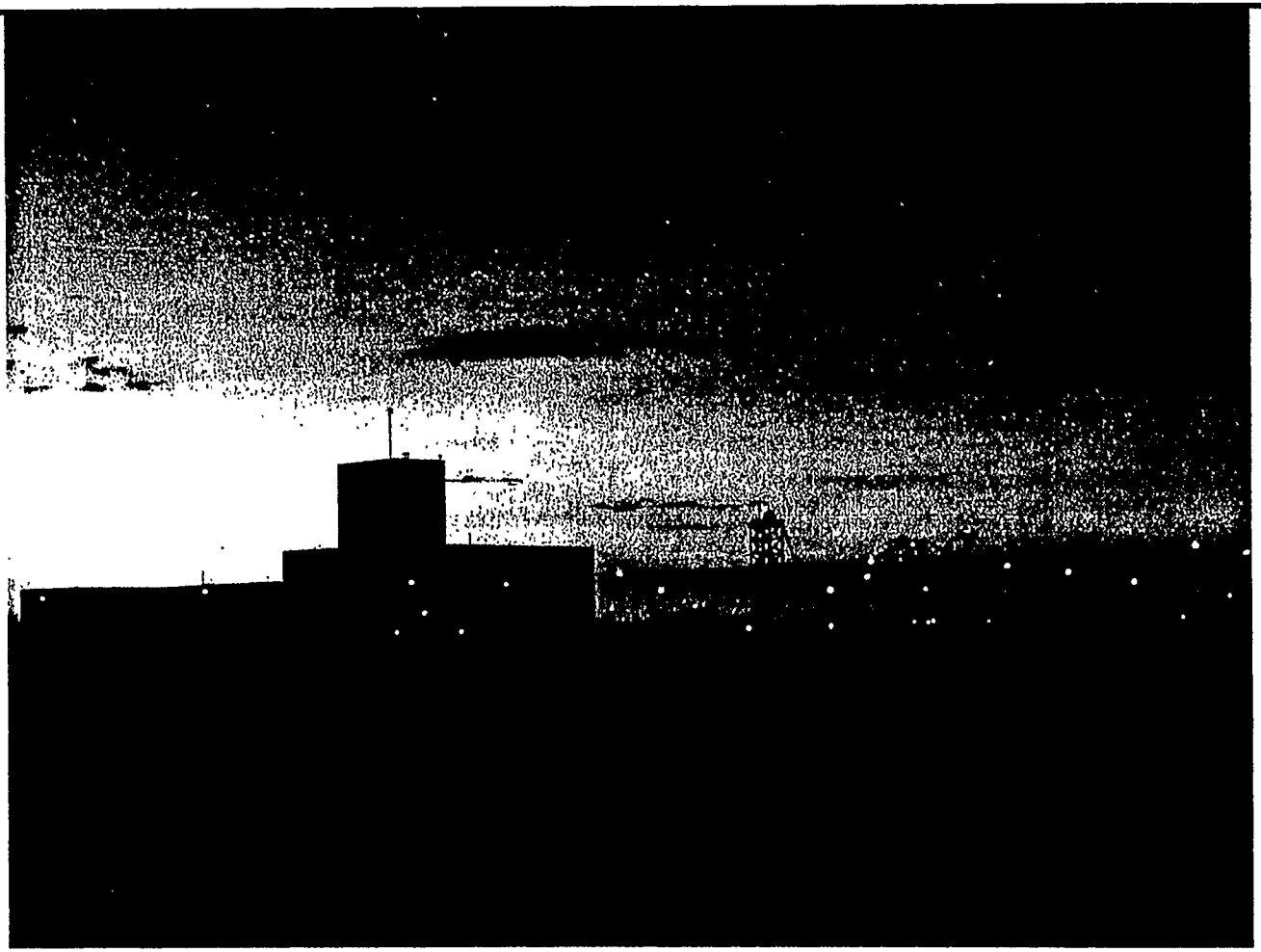
Lessons Learned (1)

- **What worked best**
  - Strong management support
  - QA assessments (audits, surveillances)
- **Reviewer training**
  - Not to criticize coding style, just to review adequacy
  - Responsibility is merely to answer the question "Is the product adequate for its intended use?", not "Can I conceive/design a better product?"
- **Software QA**
  - Implement QA throughout software lifecycle, rather than at end
  - It's cheaper to *build* in quality than to try to *test* it in

Lessons Learned (2)

- **Software CM**
  - Apply SCM early in software development phase
  - Software CM must be ongoing and relentless
- **Engage the customer and regulator early -- develop a mutual understanding of requirements and expectations**

Questions?



Sandia National Laboratories

Case Study: Effective Software Design Architecture of the UPS Project

Mike McKay
X-8
Los Alamos National Laboratory

Some notes...

- Effective Software Design
  - Get done more...keep simple
- Effective Software Design
  - Source
    - Guides
    - Directory structure
    - Robustness

Some notes... (continued)

- Mode of discussion - Directory Structure
  - "...and in this room we have..."
  - some rooms are messy, must be well kept
  - not a formal...

Overview

- Parallel Programming Background
- User Perspective
- Developer Perspective

Parallel Programming

- Mowing the lawn
  - Give everyone a lawn mower
    - Lawn mower = processor
  - Everyone mows the lawn
    - Instruction = program
    - Lawn = data

MPI
Message Passing Interface

- Tool to harness multiple processors
  - Data is passed from processor to processor
  - communicate when data boundaries are
  - communicate how much you have to move
    - MPI_Send(), MPI_Send(), ...
- Interface, not implementations
  - Interfaces exist on many computers

PVM
Parallel Virtual Machine

- Another tool to harness multiple processors
- Similar interface to MPI
- Users program with the tools available
  - Many use MPI (vendors give us that)

UPS
Unified Parallel Software

- Our tool harnesses multiple processors
- Duplication?
  - Built on top of existing tools (MPI/PVM/...)
  - Improved Functionality
    - Optimization where possible
    - Works when stuck
  - Additional Functionality
    - Gather/Scatter, Solvers, Data Parallel, ...

User Perspective: User Needs

- Useful
- Easy to Use
  - well documented
  - simple interface
- Highly reliable
  - system will not crash (no black boxes)
- Portable

Users Learning About UPS

- User Guide
  - MUST be kept accurate/current
- Directory structure
  - How is the product organized?
- Programming with UPC

UserGuide Organization

- Organized by time
  - What the user will do is discussed in the time order in the User Guide
- Programming with UPC
  - Sample Code
  - Algorithmic Issues
    - How to compile
    - How to run

UserGuide Organization (continued)

- Programming with UPS Section (continued)
  - UPS Features
    - Namespace conventions
    - Initialization/termination
    - Routine argument passing
    - Opaque object handles
    - Processor and language issues
    - Communication contexts
    - Conflicts with other products

UserGuide Organization (continued)

- Package Description Section
  - Functions grouped into packages
  - High level overview of these packages
- Constants Section
  - Values and Descriptions
    - Parameters to program with
    - Error values
  - Value for error codes must be known

UserGuide Organization: (continued)

- Function Descriptions (all functions listed)
  - Purpose
  - Usage
  - Argument Descriptions
  - Return Values
  - Discussion
  - Examples
  - See Also

Directory Structure (User): Top Level Directory

```
/usr/projects/ups/
    v-01-01-00/
    v-02-00-00/
    latest->v-02-00-00/
```

- New Minor Version - Small Changes
  - bug fixes
  - transparent enhancements

Directory Structure (User): Top Level Directory (continued)

```
/usr/projects/ups/
  v-01-00-00/
  v-01-00-01/   v-02-00-00/
                latest->v-02-00-00/
```

- New Version - Medium Changes
  - New functions
  - Major transparent enhancements

Directory Structure (User): Top Level Directory (continued)

```
/usr/projects/ups/
v-01-00-00/
v-01-00-01/
v-01-00-02/  latest->v-02-00-00/
```

- New Major Version - Major Changes
  - New Package (many new functions)
  - New Language Interface

Directory Structure (User): Top Level Directory (continued)

```
/usr/projects/ups/
v-01-00-00/   v-01-01-00/
v-01-00-01/   00-00/
v-01-00-02/   v->v-02-00-00/
```

- Symbolic link pointing to latest version

Directory Structure (User): Version Directory

```
/usr/projects/ups/v-02-00-00/
all_changes.txt   lib/
doc/              user_interface_changes.txt
include/
```

- Changes made since last version
  - `all_changes.txt`
    - Changes made to every file
  - `user_interface_changes.txt`
    - Changes made to files users access

Directory Structure (User): Version Directory (continued)

```
/usr/projects/ups/v-02-00-00/
all_changes.txt  lib/
                 user_interface_changes.txt
include/
```

- UserGuide.ps and DeveloperGuide.ps
  - Don't need DeveloperGuide....but good advertisement

Directory Structure (User): Version Directory (continued)

```
/usr/projects/ups/v-02-00-00/
all_changes.txt  lib/
doc/                    user_interface_changes.txt
```

- Include files
  - Compile specific directory
    - Language specific files

Directory Structure (User): Include Directory

```
/usr/projects/ups/v-02-00-00/include/
SGI64_mpi/        SGIn32_mpi/
SGI64_sm_mpi/     SGIn32_sm_mpi/
```

- Compile Specific (invariant on how user accesses UPS)
  - Compiler options
  - Message passing interface
  - …
- 64 bit addressing and MPI with shared memory optimizations

Directory Structure (User): Compile Specific Include Directory

```
/usr/projects/ups/v-02-00-00/include/SGI64_sm_mpi/
                                                  mpif77.h
                  CONST
```

- Same files in other compile specific directories
- C/C++ and Fortran/Fortran77 "include" files

Directory Structure (User): Version Directory (continued)

```
/usr/projects/ups/v-02-00-00/
all_changes.txt
doc/                    user_interface_changes.txt
include/
```

- Library
  - Similar to include directory
  - Compile specific directory

Directory Structure (User): Lib Directory

```
/usr/projects/ups/v-02-00-00/lib/
SGI64_mpi/        SGIn32_mpi/
SGI64_sm_mpi/     SGIn32_sm_mpi/
```

- Compile Specific (invariant on how user accesses UPS)
  - Compiler options
  - Message passing interface
  - …
- 64 bit addressing and MPI with shared memory optimizations

Directory Structure (User): Compile Specific Lib Directory

```
/usr/projects/ups/v-02-00-00/lib/SGI64_sm_mpi/
```

- Same file in other compile specific directories
- One library contains C/C++/Fortran/Fortran77 objects

Programming With UPS

- Name Space Conventions
- Function Argument Order
- Opaque Object Handles
- UPS Example Program
- UPS Example Run

Programming With UPS: Name Space Conventions

- Why?
  - Avoid collisions
  - Mnemonics
  - "What does that mean?"

Name Space Conventions (continued)

- Functions - Format
  - UPC_<PACKAGE>_<OPERATION>()
  - Packages - Grouping of functions
    - GAScommunication - Berg Parallel
    - IO: Input/Output - Shafer
    - Operation - Purpose of this routine
      - Start, Broadcast
      - Sort, Dotproduct, Gather, ...

Name Space Conventions (continued)

- Functions - language differences
  - C/C++
    - int UPS_CA_Bcast()
  - Fortran
    - call UPSF_CA_BCAST(..., ierr)
    - Fortran 77
      - call UPS_CA_BCAST(..., ierr)

Name Space Conventions (continued)

- Constants
  - Error Codes
    - UPS_ERROR_CM_BCAST
  - Data Types
    - UPS_DT_INTEGER
  - Operations
    - UPS_AA_SUM

Programming With UPS: Function Argument Order

```
UPS_AA_SAMPLE(
                                    ,
                    <qualifiers>,
                                    ,
                    <qualifiers>,
                    <qualifiers>,
                    <qualifiers>,
                    Return Value> )
```

Programming With UPS: Opaque Object Handles

- Several routines require setup calls
  - return a "handle" (integer)
- Future applications of routines use this "handle"
- Keeps complexity hidden

Programming With UPS: UPS Example Program (Fortran)

```
program foo
use UPS

call UPSF_AA_INIT( ierr )
if ( ierr /= UPS_OK ) stop

call UPSF_CM_GET_NUMPES( inumpes, ierr )
if ( ierr /= UPS_OK ) stop

call UPSF_CM_GET_PENUM( ipenum, ierr)
if ( ierr /= UPS_OK ) stop

print*,'Hello From ',ipenum,' of ',inumpes

call UPSF_AA_TERMINATE( ierr )
if ( ierr /= UPS_OK ) stop

end program foo
```

Access UPS Variables + Arg Checking

Init UPS

"UPSF" Fortran Format

Error Check

Terminate UPS.

Programming With UPS: UPS Example Run

```
% f90 foo.f
   -I /ups/v-02-02-00/include/SGIn32_mpi_sm  \
   -L /ups/v-02-02-00/lib/SGIn32_mpi_sm      \
   -lups -lmpi -larray

% mpirun a.out

Hello from  1  of 4
Hello from  3  of 4
Hello from  2  of 4
Hello from  0  of 4
```

Include File Location

Library Location

Libraries

Run command

Output

Developers Listening to Users

- "Where can I find out about UPS?"
  - User guides
- "Can you make it simpler?"
  - auto initialization (no need for UPSF_AA_INIT)
  - language interfaces
    - UPS_CM_BCAST, UPS_CM_BCAST, UPS_CM_Bcast

Developers Listening to Users (continued)

- "Can I still use MPI?"
  - UPS plays in its own sandbox
- "What if you go away?"
  - Explicit functionality to interoperate
- "Will you be on this machine?"
  - Portability is a priority

Developer Perspective: Rules to Live By

- Automate everything
  - If something has to be done twice...
    - you'll probably end up doing it again.
- Allow for easy modifications
  - Don't make people afraid of changes
    - (Changes are going to happen)
- Keep things simple
  - Bells and whistles can become nightmares

Right Tools for the Job

- Freeware?
  - Don't need a signature to purchase
  - Without tweaking, can do a lot
- A tool expert on the team
  - Expert willing to learn
  - On the team
    - Does the tool better
    - Knows when it's broken (or should be better)

Tools Worth Mentioning

- Perl
  - Automation of tasks
- GNU Make
  - Build procedures
  - Glue tasks together
- CVS
  - Configuration management
  - Takes tweaking

Developer's Learning UPS

- User Guide and Developer Guide
  - Keep it current (accurate)
- Directory structure
  - Can tell about a project by its directory tree
  - Experimental, test, documentation?
- Tools
  - Makefiles, Makefiles, README's, ...

DeveloperGuide Organization

- Organized by time
  - What the developer will do first discussed in the first order in the Developer Guide
- Getting/Building UPC
- Directory Structure
- Programming Conventions
  - Naming, Comments, Spacing, ...

DeveloperGuide Organization (continued)

- Committing to the Repository
- "How do I...?"
  - Answer miscellaneous questions
    - CVS, Perl, Make, SQA, ...
- Sourcefile Examples
  - Templates

Directory Structure (Developer)

- Component oriented
- Logically organized
- Insulated
- Every directory has a Makefile
- Typing in "make" will build everything needed in the current working directory and all directories below

Directory Structure (Developer): Top Level Directory

```
                    UPS/
[illegible]/    lib/          script/
aux/            papers/       src/
include/        peer_review/  testing/
```

- Administrative Files
  - Project Plan
  - General ToDo lists

Directory Structure (Developer): Top Level Directory (continued)

```
                    UPS/
admin/      lib/            script/
doc/        papers/         src/
include/    peer_review/    testing/
```

- Software UPS team didn't write but placed in libups.a
  - METIS
  - Shared Memory library
    - written for the SGI O2K architecture

Directory Structure (Developer): Top Level Directory (continued)

```
                    UPS/
admin/      lib/              script/
aux/        papers/           src/
            peer_review/      testing/
```

- User level include files for all languages
  - Global Constants
  - Prototypes (C)

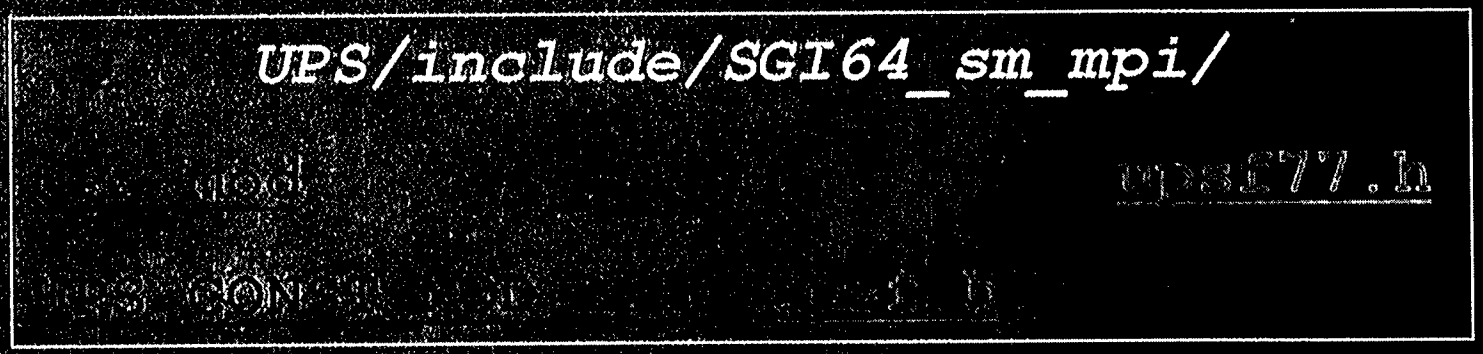
Directory Structure (Developer): Include Directory

UPS/include/

	UPS.F	master_ups.h
sun4sol/	UPS_CONST_MOD.F	sync_f_include.pl

- Similar to user directory
- Compile Specific
- Generated each compile

Directory Structure (Developer): Compile Specific Include Directory



- C/C++/Fortran/Fortran77 "include" files
- Copied over to user area during an install
- Same files in other compile specific directories
- All these files automatically generated

Directory Structure (Developer): Include Directory (continued)

UPS/include/

SGI64_mpi/	UI	master_ups.h
SGI64_sm_mpi/	UI	sync_f_include.pl

- Source that generates Fortran modules
  - Kept in include (not src) because of files they generate
  - simple files…info kept in master_ups.h

Directory Structure (Developer): Include Directory (continued)

UPS/include/

SGI64_mpi/	UPS.F	master_ups.h
SGI64_sm_mpi/	UPS_CONST_MOD.F	sync_f_include.pl

- Primary include file - written in C
  - All global constants and prototypes
  - "master" include file (other include files generated from it)

Directory Structure (Developer): Include Directory (continued)

UPS/include/

SGI64_mpi/	UPS.F	master_ups.h
SGI64_sm_mpi/	UPS_CONST_MOD.F	sync_f_include.pl

- Generates includes from master_ups.h
  - changes in master_ups.h get propagated

Automation of Include Files

upsf.h

(spacing set for fixed or freeform)

```
/* ---------------- */
/* Statistics flags */
/* ---------------- */
enum UPS_Statistics_enum
{
  /* disable statistics (default) */
  UPS_STATISTICS_OFF = 0,
  /* enable full statistics */
  UPS_STATISTICS_ON,
  /* no timings (faster) */
  UPS_STATISTICS_ON_NO_TIMINGS
};
```

```
! ----------------
! Statistics flags
! ----------------
! disable statistics (default)
  integer(KIND=UPS_KIND_INT4), parameter &
 & :: UPS_STATISTICS_OFF           = 0,
! enable full statistics
  integer(KIND=UPS_KIND_INT4), parameter &
 & :: UPS_STATISTICS_ON            = 1
! no timings (faster)
  integer(KIND=UPS_KIND_INT4), parameter &
 & :: UPS_STATISTICS_ON_NO_TIMINGS = 2
```

master ups.h/upsf.h

Directory Structure (Developer): Top Level Directory (continued)

```
                    UPS/

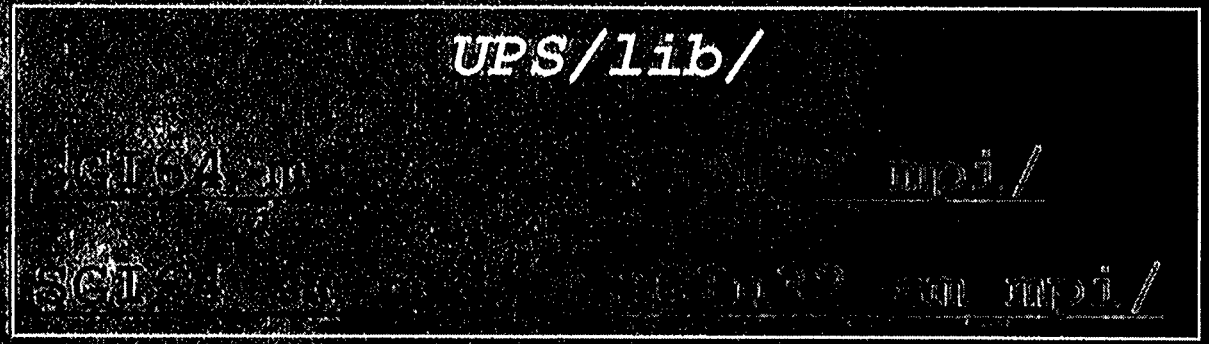
admin/                              script/

aux/          papers/               src/

include/ peer_review/               testing/
```

- Library
  - compile specific directory

Directory Structure (Developer): Lib Directory



- Similar to user directory
- Compile Specific
- Generated each compile

Directory Structure (Developer): Compile Specific Lib Directory

```
UPS/lib/SGI64_sm_mpi/
lib
```

- One library contains C/C++/Fortran/Fortran77 objects
- Copied over to user area during an install
- Same file in other compile specific directories
- This file automatically generated

Directory Structure (Developer): Top Level Directory (continued)

```
                    UPS/

admin/    lib/                script/

aux/                          src/

include/  peer_review/        testing/
```

- Written documents

Directory Structure (Developer): Papers Directory

```
UPS/papers/
DEST        HPCU99/  SC99/
DG/         SC99/    UG/
```

- Design documents

Directory Structure (Developer): Design Directory

`UPS/papers/DESIGN/`

- Contains design documents of major tasks of UPS

Directory Structure (Developer): Papers Directory (continued)

```
UPS/papers/
DESIGN/   HPCU99/  SC99/
          SC99/    UG/
```

- DeveloperGuide and UserGuide

Directory Structure (Developer): UG Directory

```
UPS/papers/UG/
                ref/
header2latex.pl
```

- LaTeX files used to generate UserGuide.ps
- Many of our docs are written in LaTeX

Directory Structure (Developer): UG Directory (continued)

```
UPS/papers/UG/

UserGuide.tex

ref/                    ups_ug_*.tex
```

- `ref/`
  - directory containing files describing each user accessible routine (100 routines|files)
- `header2latex.pl`
  - Perl script that automatically generates these LaTeX files from the C source code

Automation of UserGuide Files

Condensed C file comment header

```
/* ================================================================
   int UPS_CM_Shift(
                     double *x
                     );
   Purpose
   =======
          Shifts data buffer x
   Arguments
   =========
   x      Intent:        input/output
          C type:        const void*
          Fortran type: user choice
          Data buffer
   ierr   Intent:        output
          C type:        (n/a) int function return value
          Fortran type: integer
          Return status.
   Program Flow
   ============
   1) shift buffer
   ================================================================ */
```

Prototype

Purpose

Arguments
(intent)
(types)
(description

Program
Flow

Automation of UserGuide Files

C File Comment Headers



UserGuide Reference Pages

Unified Parallel Software

Directory Structure (Developer): Papers Directory (continued)

```
UPS/papers/
DESIGN/                    SC99/
DG/                        UG/
```

- Presentations/Conferences

Directory Structure (Developer): Top Level Directory (continued)

```
                    UPS/
admin/      lib/            script/
aux/        papers/         src/
include/                    testing/
```

- Files to be peer reviewed
- Personal ToDo lists

Directory Structure (Developer): Peer Review Directory

```
UPS/peer_review/
```

- Every developer has his own directory

Directory Structure (Developer): lmdm Directory (continued)

```
UPS/peer_review/lmdm/
    ups_gs_setup.c ups_gs_scatter.c
```

- Simple list on what the developer is doing and what they are planning to do
- For visibility to other developers

Directory Structure (Developer): lmdm Directory (continued)

```
UPS/peer_review/lmdm/

ToDo.lmdm                                    dp_sort.c
```

- Files I wish to be peer reviewed by a coworker
- Both peer reviewer and developer can modify same file thanks to CVS
  - discussion lines
  - simple fixes
  - CVS ==> log of changes
- When reviewed, copy file back to original place

Directory Structure (Developer): Top Level Directory (continued)

```
                    UPS/
admin/      lib/            script/
aux/        papers/         src/
include/    peer_review/    testing/
```

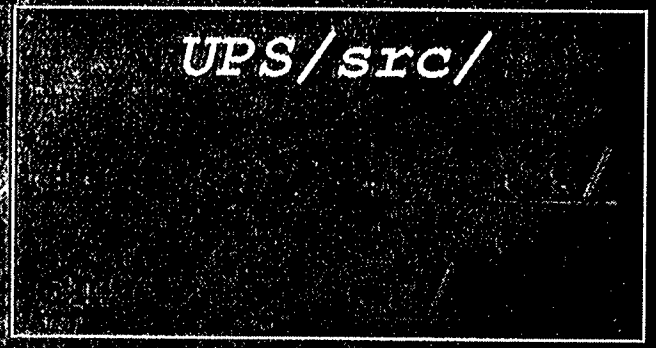
- Useful scripts
  - Use tools again
  - Templates

Directory Structure (Developer): Top Level Directory (continued)

```
                    UPS/
admin/    lib/              script/
aux/      papers/           src/
include/  peer_review/      testing/
```

- Software UPS team writes and places in libups.a

Directory Structure (Developer): src Directory



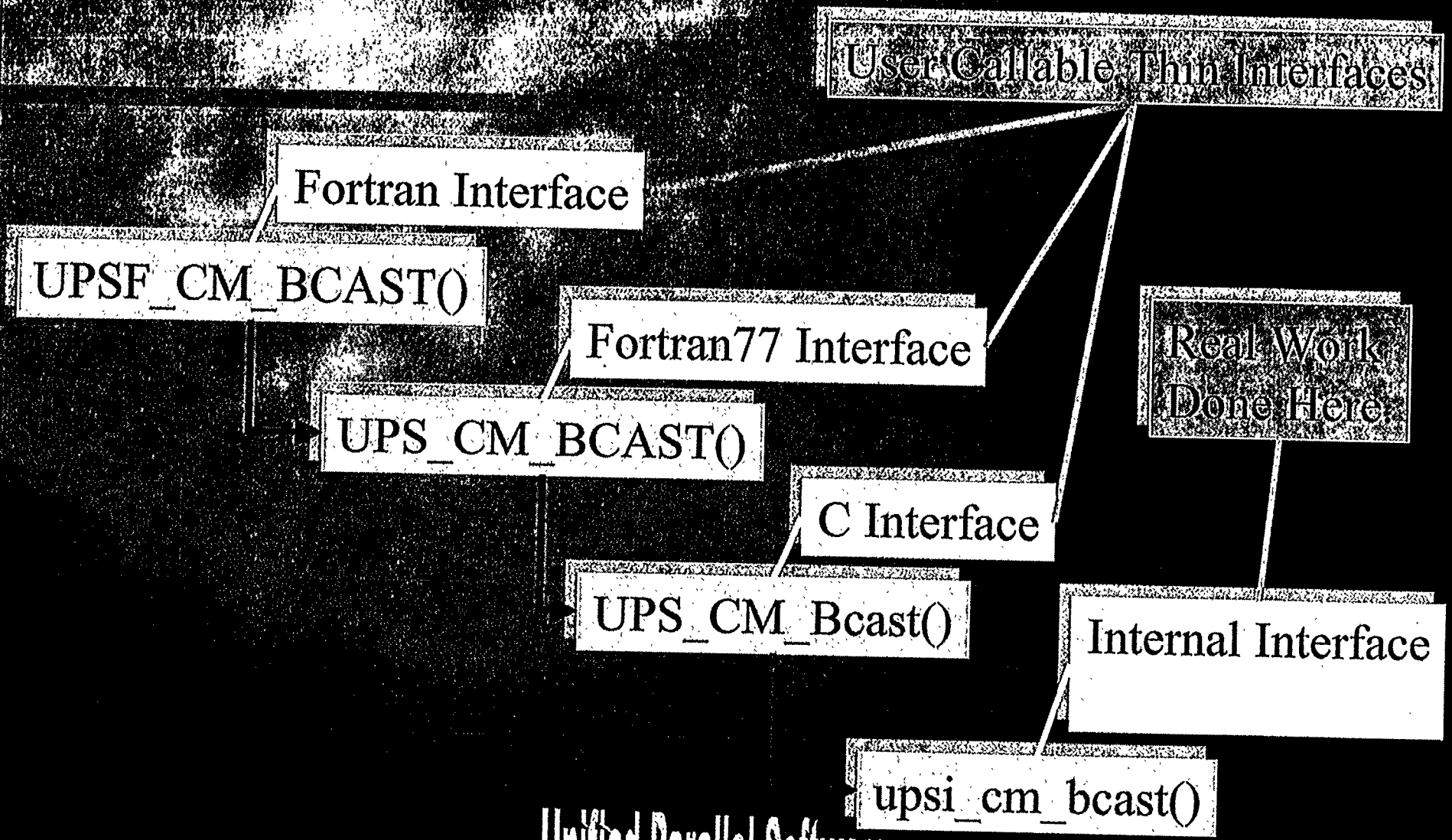
- Split into logical packages
  - Locate related routines together
    - Directory name
    - Routine name

Directory Structure (Developer): src Directory (continued)

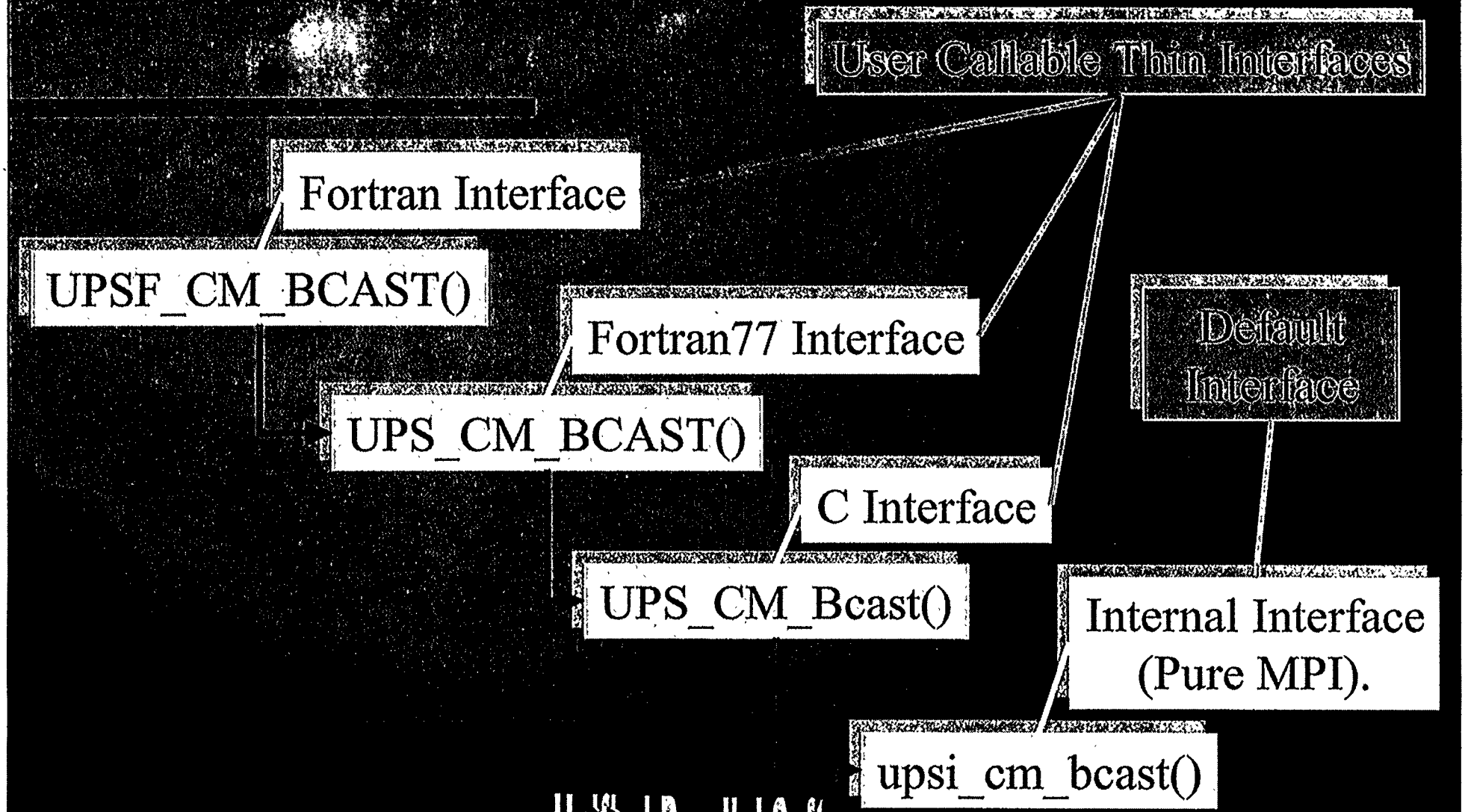
```
UPS/src/
aa/  dp/  er/  ut/
     dt/  gs/
```

- Communication Package
- Will use this package as a design example

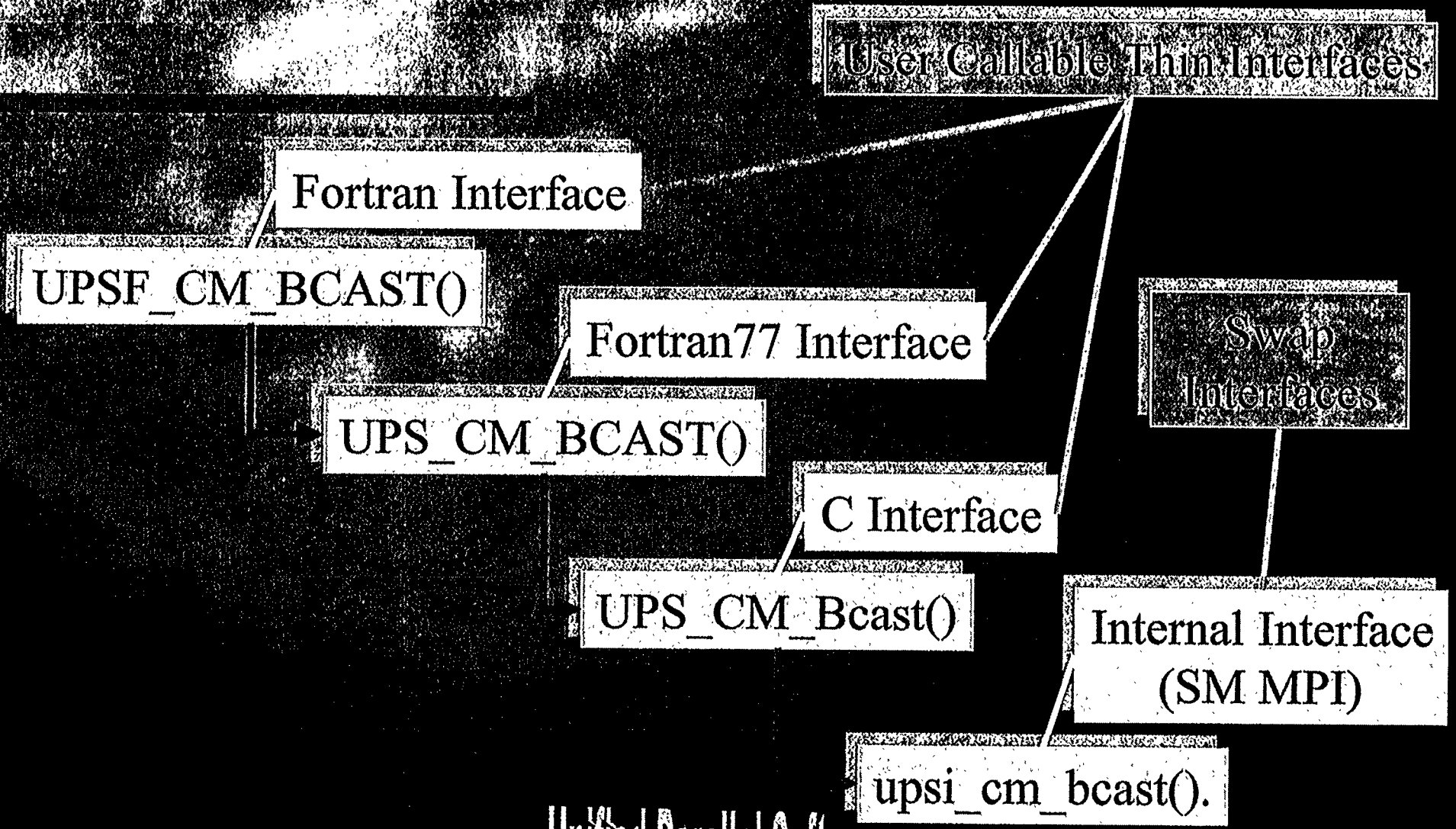
Function Calling Chain



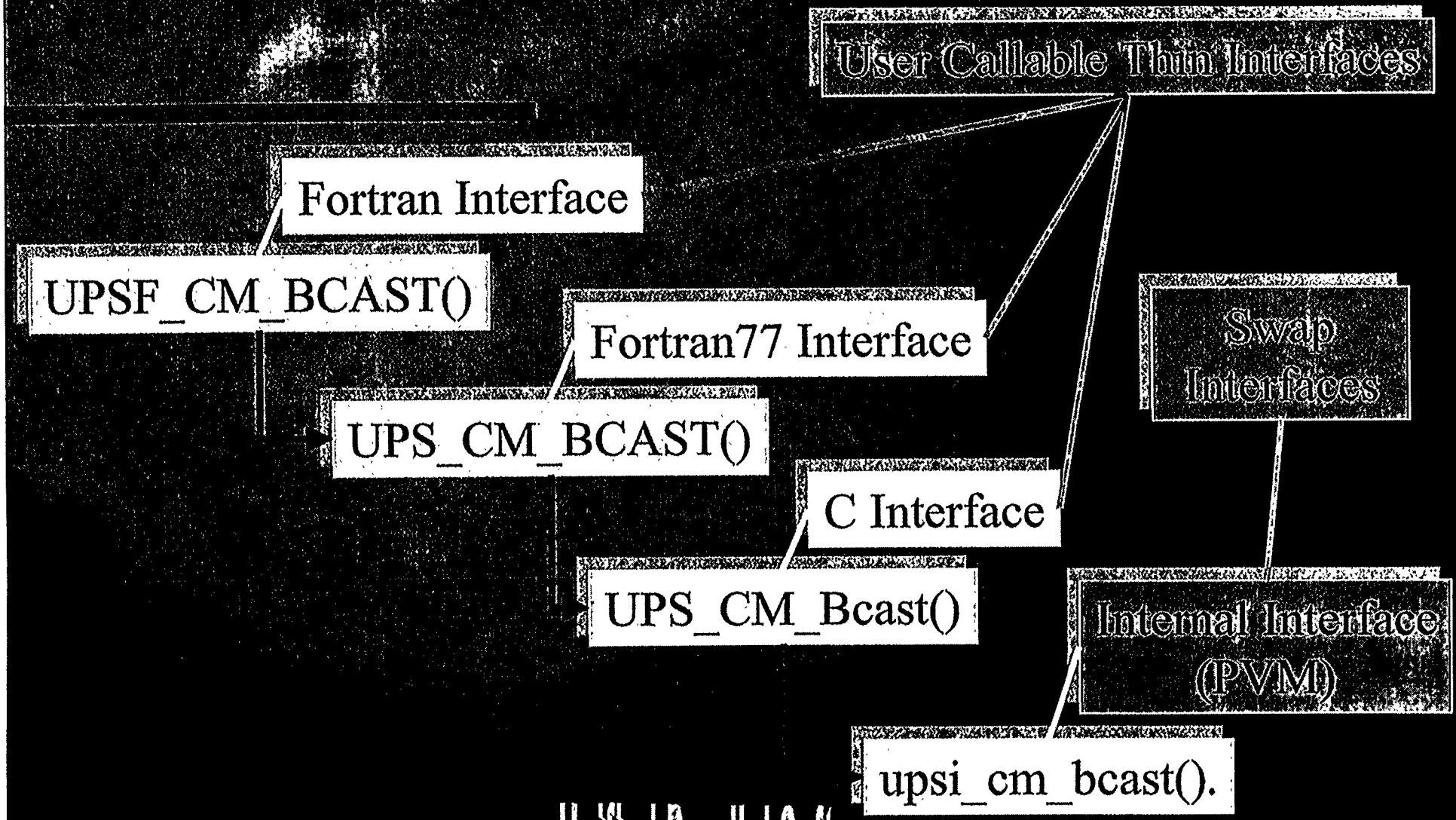
Function Calling Chain



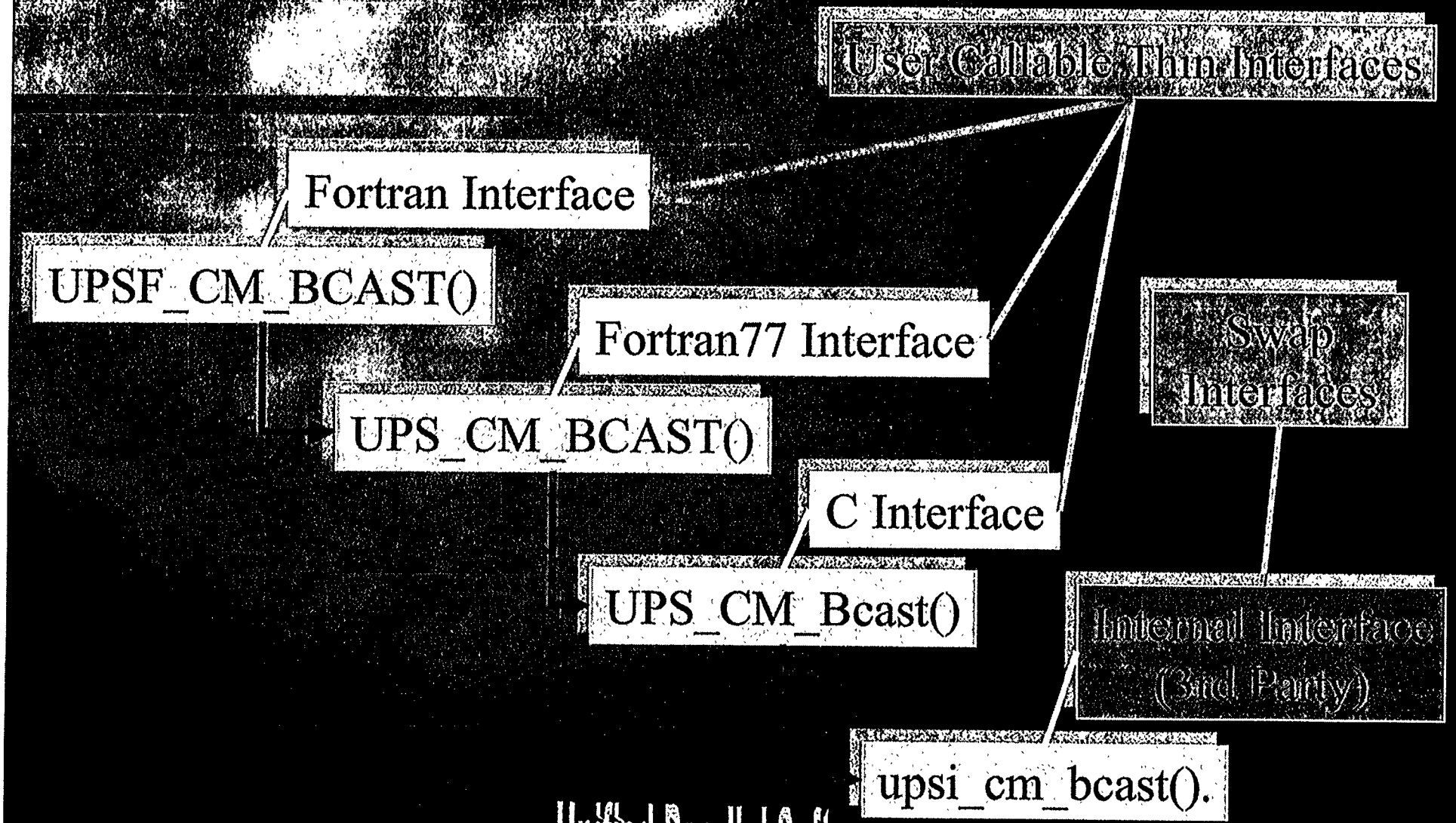
Function Calling Chain



Function Calling Chain



Function Calling Chain



Directory Structure (Developer): src cm Directory

UPS/src/cm/

upsp_cm.h

utils/

- User/Other Routine Interface Directory

Directory Structure (Developer): src cm Directory (continued)

UPS/src/cm/

upsp_cm.h

utils/

internals/

- Function Files
  - One function per file
  - File name reflects function name
- Every function at this level is user callable

Example C Source File Format

White Space

Variable Declarations

{ }

```
Comment Header (previously seen in UserGuide Discussion)
int UPS_CM_Shift( double *x )
{
   /* Local Declarations */
   int
      i,              /* loop variable */
      count = 20;     /* num elements  */

   /* Executable Statements */

   /* shift data */
   for ( i = 0; i < count; i++ )
      {
         x[i] = x[i+1]
      }
   /* DONE: shift data */

} /* End UPS_CM_Shift */
```

Comments

Comment Header

Local Variables

Main Program

Code Block

End.

Directory Structure (Developer): src/cm Directory (continued)

UPS/src/cm/

UPS_CM_BCAST.c	ups_cm_bcast.c	upsp_cm.h
UPS_CM_BCAST.c	ups_cm_init.c	utils/
internals/	ups_cm_terminate.c	

- Fortran Interface : `UPSF_CM_BCAST()`
- Written in Fortran
  - Automatic datatype detection
  - Optional Arguments
  - Strong argument type checking

How Fortran Calls Fortran77

UPSF_CM_SUM.F

```
INTERFACE UPSF_CM_SUM
 MODULE PROCEDURE UPSF_CM_SUM_INT4
 MODULE PROCEDURE UPSF_CM_SUM_REAL8
END INTERFACE
…
SUBROUTINE UPSF_CM_SUM_INT4( in, out, mask, ierr )
   INTEGER … ::              in
   INTEGER … ::              out
   INTEGER … , OPTIONAL :: MASK
   INTEGER … ::              ierr

   IF ( PRESENT( mask ) ) THEN
      CALL UPS_CM_SUM_MASK( in, out, mask, UPS_DT_INT, ierr )
   ELSE
      CALL UPS_CM_SUM( in, out, UPS_DT_INT, ierr )
   ENDIF
RETURN
…
```

Datatype Detection

Optional Arguments

Directory Structure (Developer): src cm Directory (continued)

```
                        UPS/src/cm/

UPSF_CM_BCAST.F  ups_cm_bcast.c        upsp_cm.h
                 ups_cm_init.c         utils/
internals/       ups_cm_terminate.c
```

- Fortran77 Interface : `UPS_CM_BCAST()`
- Written in C so it can call our C interface

How Fortran77 Calls C

Name Mangling

Prototype in master_ups.h

```
#if defined _F2C_UNDERLINE
#define UPS_CM_SET_INTEGER ups_cm_set_integer_
#else
#define UPS_CM_SET_INTEGER ups_cm_set_integer
#endif
  void UPS_CM_SET_INTEGER( int *int_to_set,
                           int *ierr );
```

ierr
C: return value
F: argument

UPS_CM_SET_INTEGER.c

```
void UPS_CM_SET_INTEGER( int *int_to_set,
                         int *ierr )
{
    *ierr = UPS_CM_Set_integer( *int_to_set );
}
```

All variables passed as pointers

Directory Structure (Developer): src cm Directory (continued)

```
                    UPS/src/cm/

UPSF_CM_BCAST.F                          upsp_cm.h
UPS_CM_BCAST.c   ups_cm_init.c           utils/
internals/       ups_cm_terminate.c
```

- C Interface : `UPS_CM_Bcast()`
- UserGuide gets its reference pages from the C file comment headers
- C interface accesses internals
  - UPS routines access the C interface

How C Calls Internals

ups_cm_max.c
comment header used for UserGuide

```
int UPS_CM_Max( void *in,
                void *out,
                UPS_DT_DATATYPE datatype )
{
   int
     ierr = UPS_OK; /* error return variable */

   if ( !upsp_cm.initialized ) UPS_CM_Init();

   ierr = upsi_cm_max( in, out, datatype );
   if ( ierr != UPS_OK )
      {
         UPS_ER_Perror( "UPS_CM_Max (upsi_cm_max) ",
                        ierr );
         ierr = UPS_ERROR_CM_MAX;
         return ierr;
      }
   return ierr;
}
```

Auto-Initialization

Error Check

Call Stack

Function Error Code

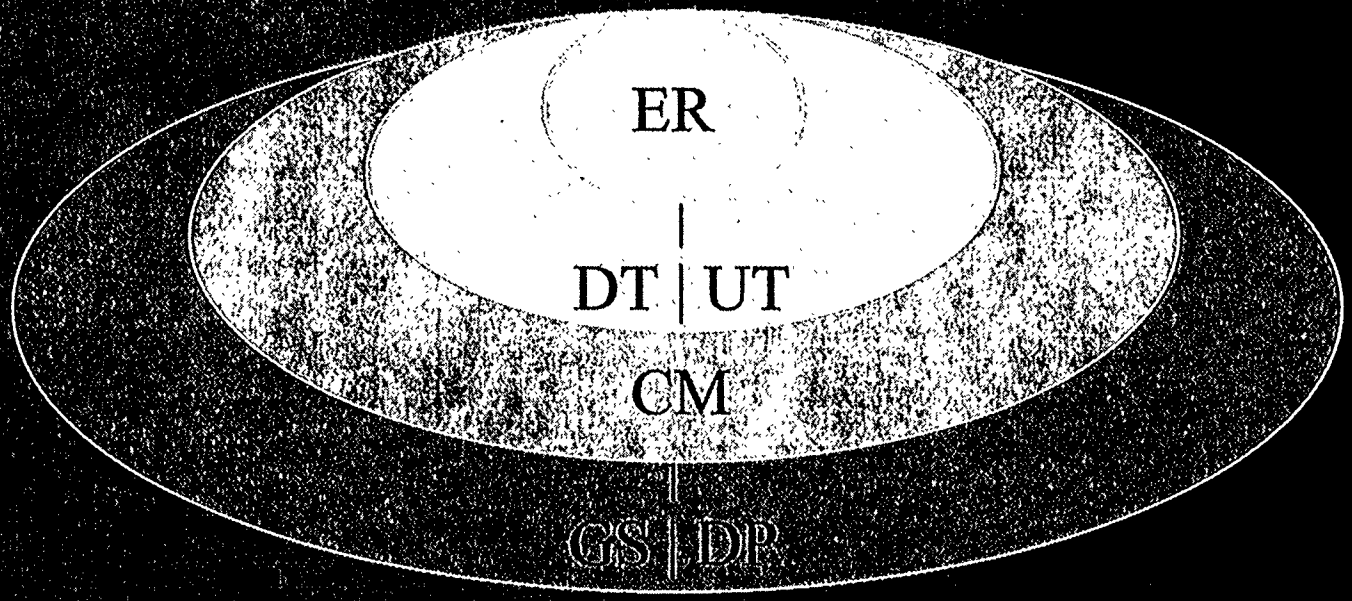
Directory Structure (Developer): src/cm Directory (continued)

UPS/src/cm/

```
UPSF_CM_BCAST_F.c   ups_cm_bcast.c      upsp_cm.h
UPS_CM_BCAST.c      ups_cm_init.c       utils/
internals/          ups_cm_terminate.c
```

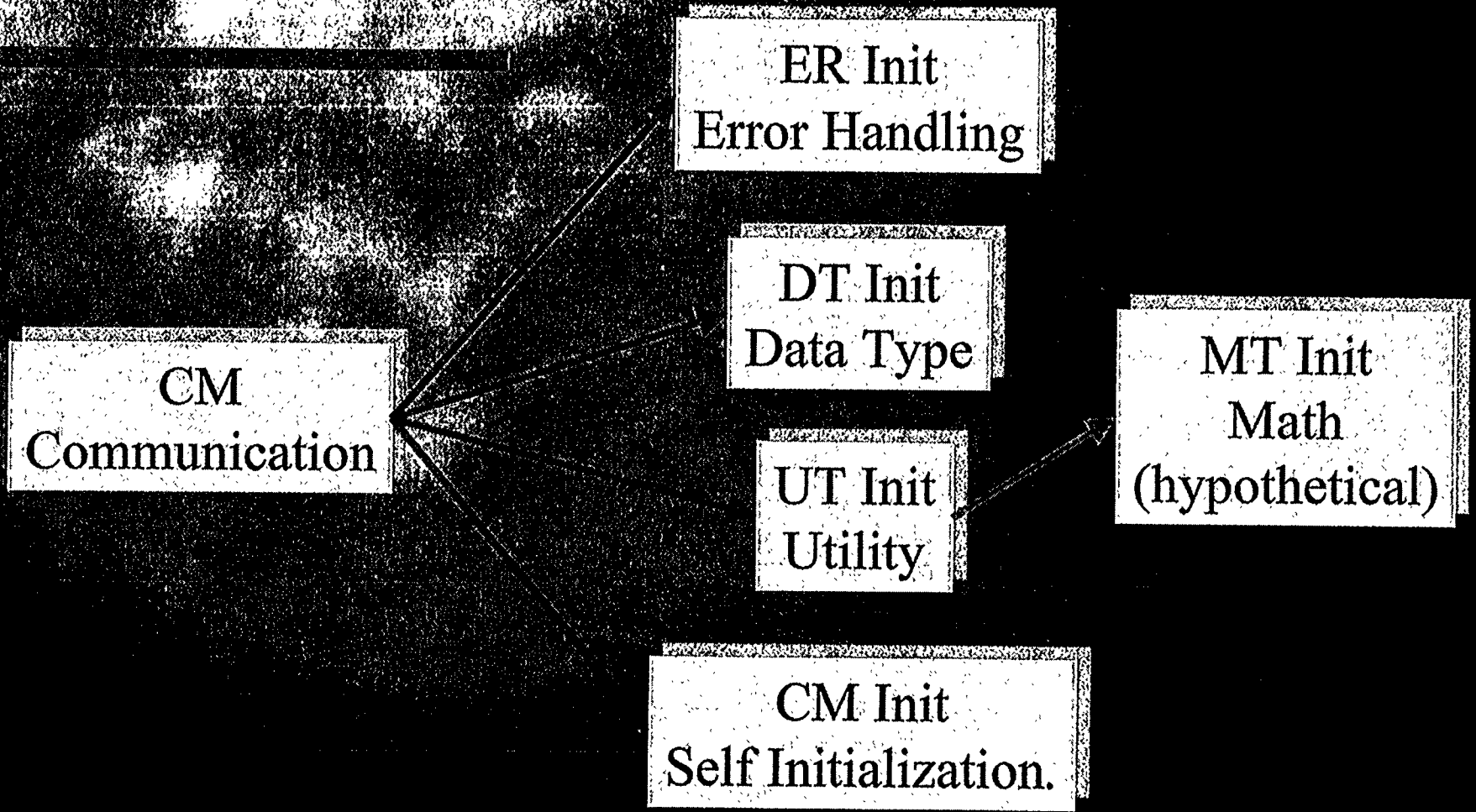
- Every package has init/terminate routines
  - Init only packages you use
  - Terminate only packages you init'd

Package Layering

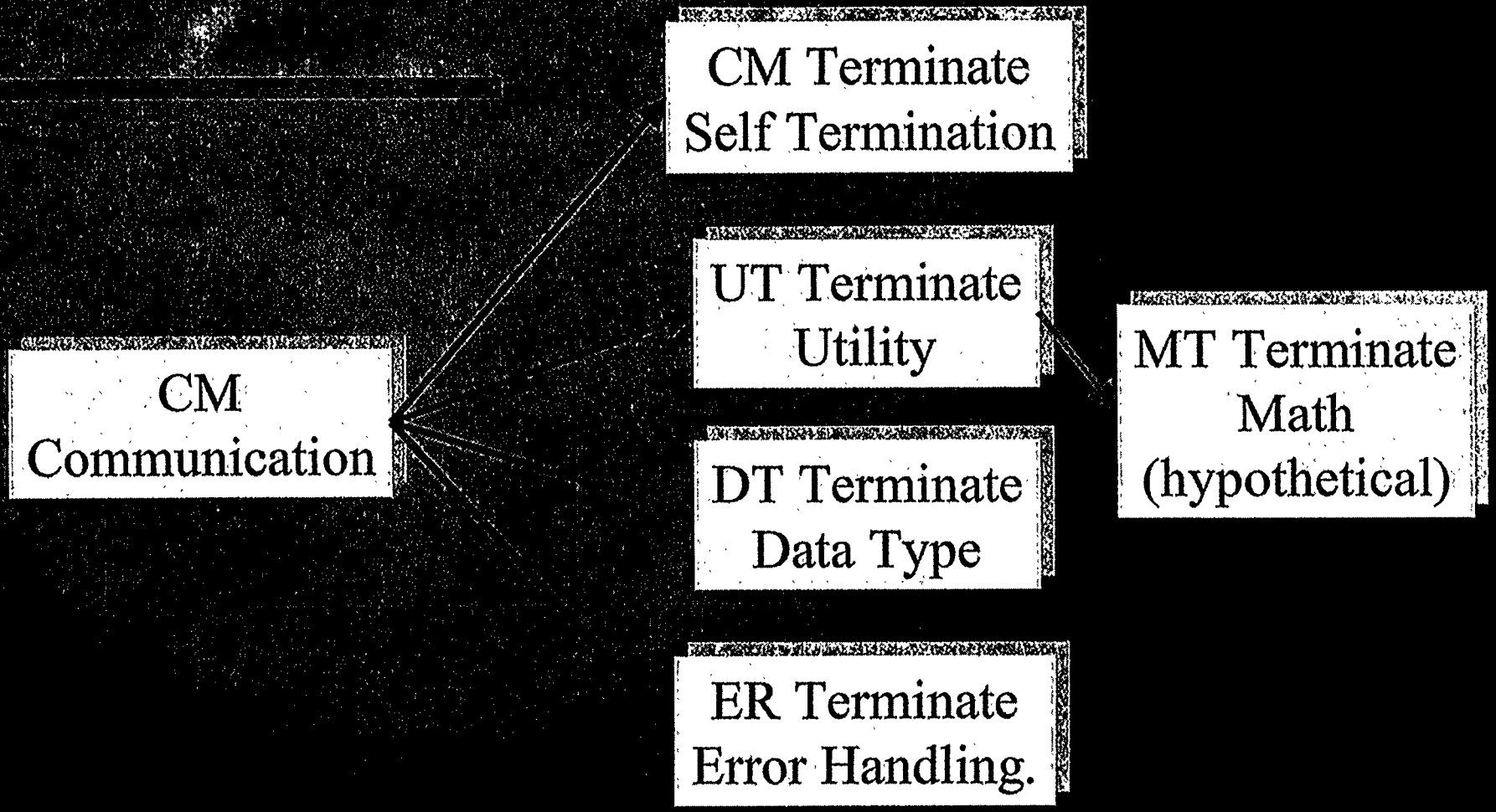


Can call your own package or any package inside yours

CM Package Initialization



CM Package Termination



Directory Structure (Developer): src cm Directory (continued)

UPS/src/cm/

UPSF_CM_BCAST.F	ups_cm_bcast.c	upsp_cm.h
UPS_CM_BCAST.c	ups_cm_init.c	utils/
internals/	ups_cm_terminate.c	

- Package private include file
  - Accessible to other packages
  - Not accessible to user (not directly)

Private Include File

upsp_cm.h

```
typedef struct
{
  /* initialization status */
  int
    init_dt,      /* if dt is initialized  */
    init_er,      /* if er is initialized  */
    init_ut,      /* if ut is initialized  */
    initialized; /* currently initialized */

  /* current context info */
  int
    numpes,       /* number of pe's */
    penum,        /* pe number      */
}
ups_cm_globals;
UPS_EXTERN ups_cm_globals upsp_cm;
```

Initialization Status

Other Variables

Struct Variable Name.

Directory Structure (Developer): src/cm Directory (continued)

UPS/src/cm/

UPSF_CM_BCAST.F	ups_cm_bcast.c	upsp_cm.h
UPS_CM_BCAST.c	ups_cm_init.c	utils/
internals/	ups_cm_terminate.c	

- Utility routines not meant to be called by user

Directory Structure (Developer): src cm utils Directory

```
UPS/src/cm/utils/
                    upsp_cm_print_stats.c
```

- upsp_cm_buffer_init()
  - allocates buffer space

Directory Structure (Developer): src cm Directory (continued)

UPS/src/cm/

UPSF_CM_BCAST.F	ups_cm_bcast.c	upsp_cm.h
UPS_CM_BCAST.c	ups_cm_init.c	utils/
	ups_cm_terminate.c	

- Where the real routines are located
  - Default internal
  - Architecture optimizations

Directory Structure (Developer): Internals Directory

`UPS/src/cm/internals/`

`../ cmal ../ cmi sm mpi/`

- Architecture/Communication Protocol Directories

Directory Structure (Developer): Internals Directory (continued)

UPS/src/cm/internals/

cmi_pvm/ cmi_serial/ cmi_sm_mpi/

- Default Internal - Pure MPI
  - Should work on every architecture

Directory Structure (Developer): Internals Directory (continued)

UPS/src/cm/internals/

`cmi_mpi/` `cmi_serial/` `cmi_sm_mpi/`

- PVM
  - If we wish to use the PVM communication protocol instead of MPI

Directory Structure (Developer): Internals Directory (continued)

```
UPS/src/cm/internals/
cmi_mpi/  cmi_pvm/  ...  cmi_sm_mpi/
```

- Serial Code
  - 1 processor instead of multi-processor
  - Eliminates some unnecessary overhead
    - Data Transfer ===> memcpy()

Directory Structure (Developer): Internals Directory (continued)

UPS/src/cm/internals/

`cmi_mpi/` `cmi_pvm/` `cmi_serial/` `cmi_sm_mpi/`

- MPI that takes advantage of the Shared Memory Architecture on the SGI O2K at Los Alamos National Laboratory

Directory Structure (Developer): cmi_sm_mpi Directory

```
UPS/src/cm/internals/cmi_sm_mpi/
                                  utils/
                  upspi_cm.h
```

- Function Files
  - Every function at this level has its corresponding C language function at the `UPS/src/cm/` level
  - Functions at this level are callable only from the "parent"
    - `upsi_cm_bcast()` <== `UPS_CM_Bcast()`

Directory Structure (Developer): cmi_sm_mpi Directory (continued)

```
UPS/src/cm/internals/cmi_sm_mpi/
                    upsi_cm_terminate.c utils/
upsi_cm_init.c   upspi_cm.h
```

- With `UPS/aux/<shared memory library>`
  - Able to outperform pure SGI MPI

Directory Structure (Developer): cmi_sm_mpi Directory (continued)

```
UPS/src/cm/internals/cmi_sm_mpi/
upsi_cm_bcast.c                    terminate.c utils/
                    upspi_cm.h
```

- Called by their corresponding C language function at the `UPS/src/cm/` level.
- Specific internal init/terminate

Directory Structure (Developer): cmi_sm_mpi Directory (continued)

UPS/src/cm/internals/cmi_sm_mpi/

```
upsi_cm_bcast.c upsi_cm_terminate.c utils/
upsi_cm_init.c
```

- Similar to `UPS/src/cm/upsp_cm.h`
  - `upsp_cm.h` ==> `upspi_cm.h`
  - `upsp_cm.<variable>` ==> `upspi_cm.<variable>`
- One struct contains all internal global variables

Directory Structure (Developer): cmi_sm_mpi Directory (continued)

```
UPS/src/cm/internals/cmi_sm_mpi/
upsi_cm_bcast.c  upsi_cm_terminate.c  utils/
upsi_cm_init.c   upspi_cm.h
```

- Similar to **UPS/src/cm/utils/**
- Utility functions that do not have a corresponding C language function at the **UPS/src/cm/** level

Directory Structure (Developer): cmi_sm_mpi utils Directory

UPS/src/cm/internals/cmi_sm_mpi/utils/

upspi_cm_shared_buf.c

cm_bcast_init.c

- This internals bcast function must be init'd before called

File Overview (main)

File	Location	Accesses	Accessed By
UPSF_CM_BCAST.F (Fortran)	src/cm/	UPS_CM_BCAST.c	User
UPS_CM_BCAST.c (Fortran77)	src/cm/	ups_cm_bcast.c	User + UPSF_CM_BCAST.F
ups_cm_bcast.c (C) (All Packages)	src/cm/	upsi_cm_bcast.c+ src/cm/utils/+ User Level C Funcs	User + UPS_CM_BCAST.c+ User Level C Funcs
upsp_cm.h (Package Vars)	src/cm/	(n/a)	Developer Functions
upsp_cm_buf.c (private)	src/cm/utils/	src/cm/utils/ + src/cm/	src/cm/ + src/cm/utils/.

File Overview (Internals)

File	Location	Accesses	Accessed By
ups_cm_bcast.c (Internal)	/INT/cmi_mpi/	aux/ cmi_mpi/utils/ User Level C Funcs	ups_cm_bcast.c
ups_int_cm.h (/INT/ Package Vars)	/INT/cmi_mpi/	(n/a)	cmi_mpi/
ups_int_cm_bcast.c (Internal Private)	cmi_mpi/utils/	cmi_mpi/utils/ User Level C Funcs	cmi_mpi/* + cmi_mpi/utils/

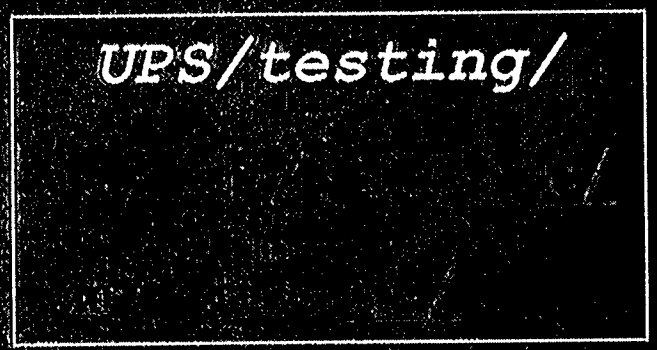
Directory Structure (Developer): Top Level Directory (continued)

UPS/

admin/	lib/	script/
aux/	papers/	src/
include/	peer_review/	testing/

- Tests
  - correctness
  - performance
  - specific applications

Directory Structure (Developer): testing Directory



- Looks just like `UPS/src/`
  - Each package has its own test suite

Directory Structure (Developer): testing Directory (continued)

```
UPS/testing/
aa/  dp/  er/  ut/
cm/  dt/
```

- Testing in the GS (Gather-Scatter) Package
  - other directories similar

Directory Structure (Developer): testing/gs Directory (continued)

```
UPS/testing/gs/
        fortran/
```

- Testing how specific applications use the GS package

Directory Structure (Developer): apps Directory (continued)

```
UPS/testing/gs/apps/
compare_timings/                    lo_sparse/
```

- Replaced an existing GS package with UPS in a user's code
  - Add UPS calls directly to sample code provided by user
  - Test and time results

Directory Structure (Developer): testing gs Directory (continued)

```
UPS/testing/gs/
apps/
```

- Unit test every function in the GS package
- Only need test Fortran because of the calling sequence
  - Fortran ==> Fortran77 ==> C ==> Internals

Directory Structure (Developer): fortran Directory

```
UPS/testing/gs/fortran/
                    test_scatter.F
test_gather.F test_setup_ids.F
```

- Main program
  - Simple pass/fail message printed to standard out
  - Extensive timing results printed to file

Directory Structure (Developer): fortran Directory (continued)

```
UPS/testing/gs/fortran/
prog.F
```

- Tests components of the GS package
  - Functionality tests
  - Stress tests

Things UPS Does Wrong

- Packages shouldn't be able to call themselves
  - Hard to know when a function is not going to call another
    (Already causes problems with Reduce/Bcast)
- Should write access functions for structs instead of directly accessing them.
  - Lot of work...we have some access functions

Things UPS Does Wrong (continued)

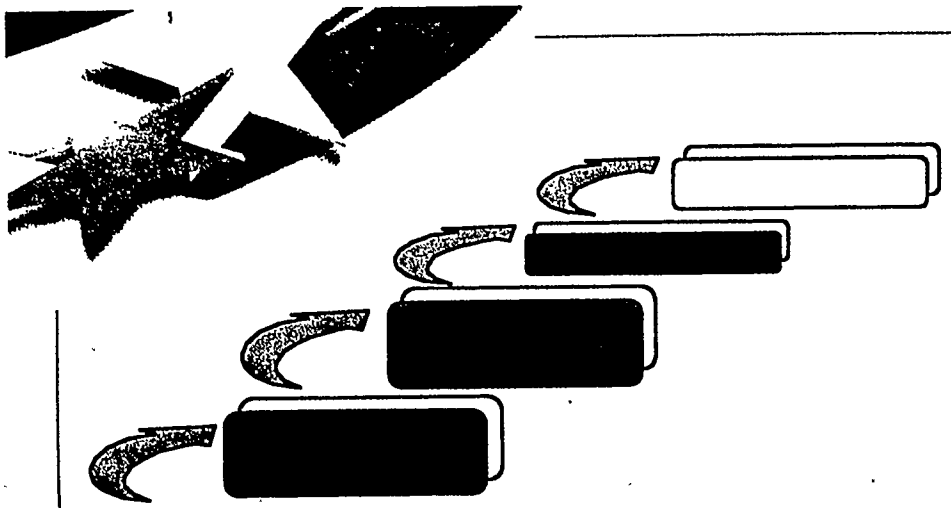
- Copy/Paste of functions with the same functionality (internals directory) is prone to error
- Tedious
  - Must propagate changes to all files
- Error checking code blocks are not tested
- Code coverage tools are more difficult to use

UPS Design

- Modifiable
  - Explicit rules allow developers to "copy/paste"
  - Simplicity: no "Black Boxes"
    - Directories logically break up source
    - Comments reduce "reverse engineering"
- Extensive test suite
  - Makefile integrates the tests

UPS Design (continued)

- Automation
  - Do it correctly once
  - Glue tasks together so nothing is forgotten
- Modifying source
  - "make" builds, installs, builds library, and runs tests
- Installing fixes
  - "make dist" builds all necessary files, copies files to distribution, and creates a CVS snapshot



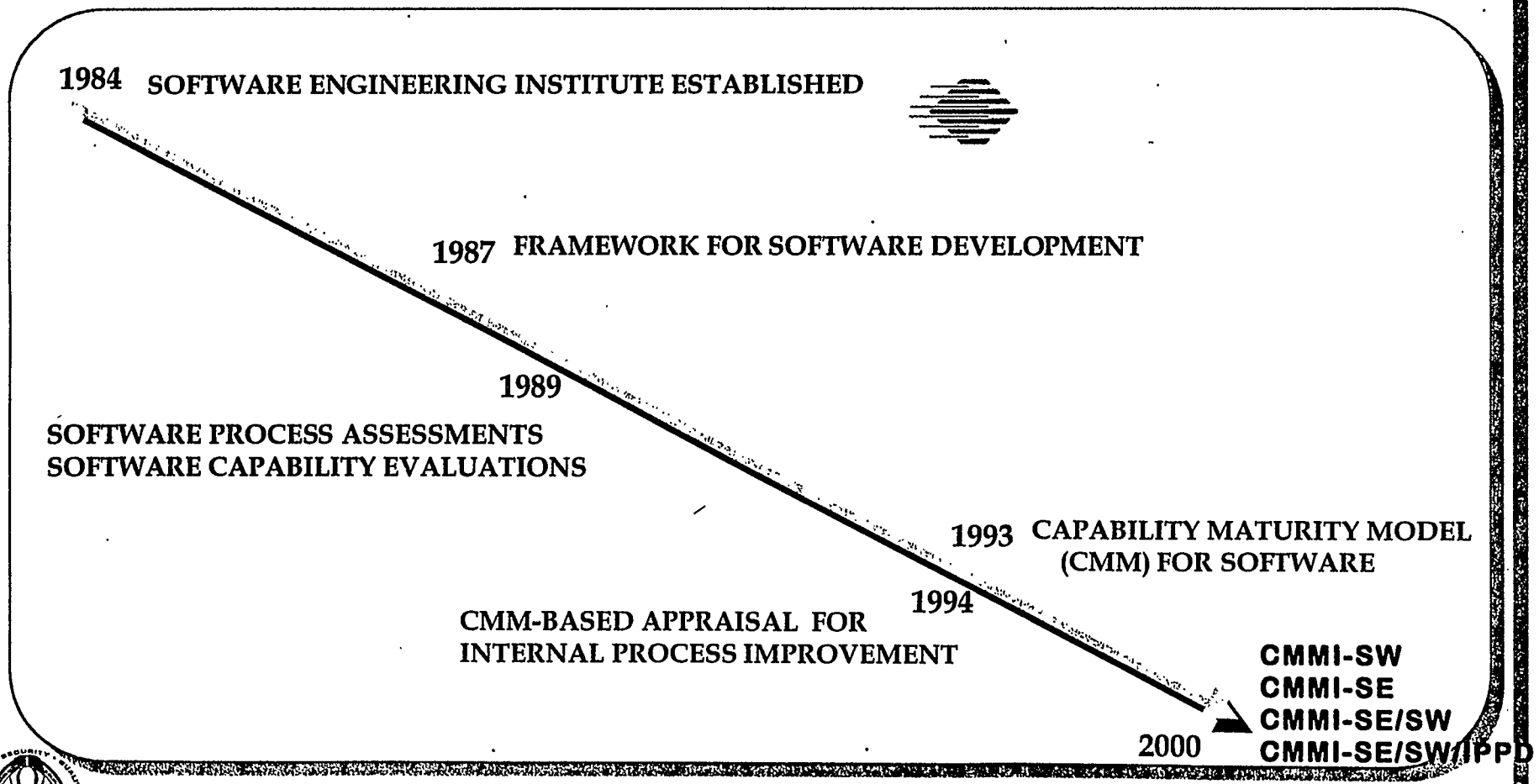
Introduction to the SEI SW-CMM Version 1.1

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Sandia National Laboratories

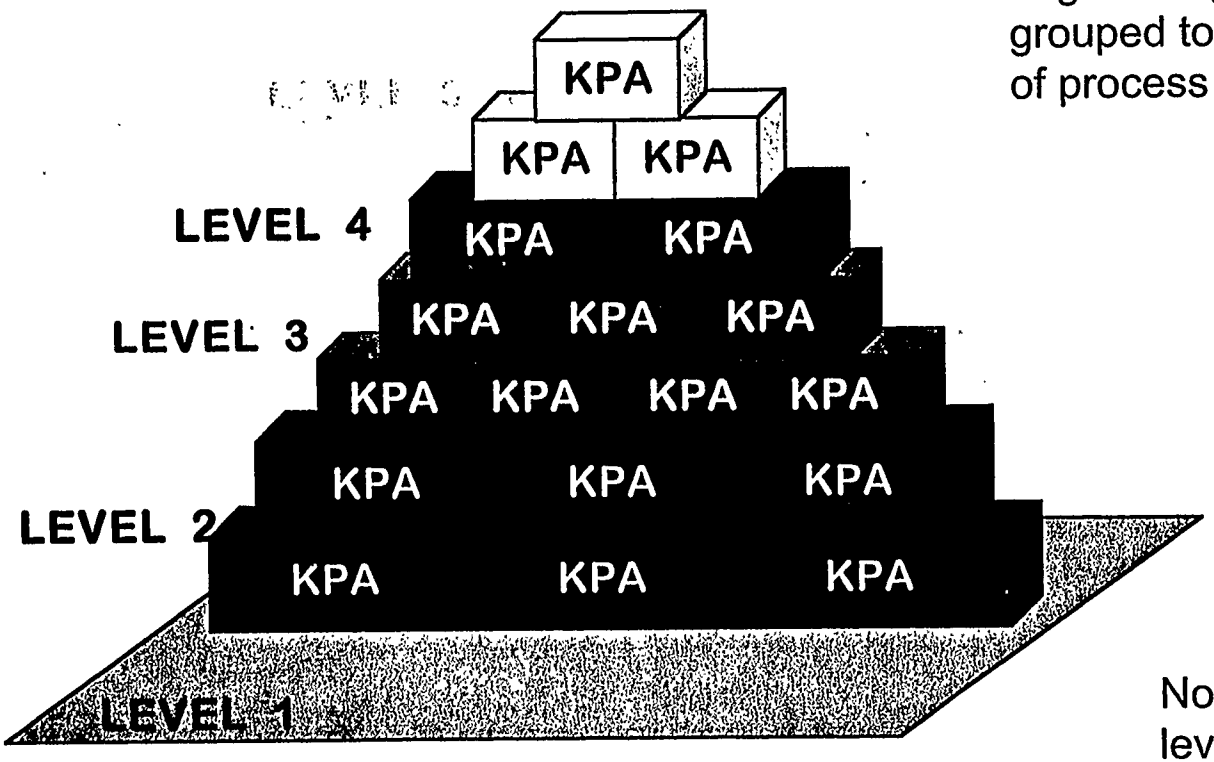
Software Engineering Institute Capability Maturity Model

The SEI first produced a "Framework" document for software development in 1987, which later became the software CMM. Much of this was based on work done in IBM by Watts Humphrey and others. The SEI is currently working on Version 2.0 of the software CMM as part of the Capability Maturity Model Integration (CMMI) effort.



The Structure of the CMM (1)

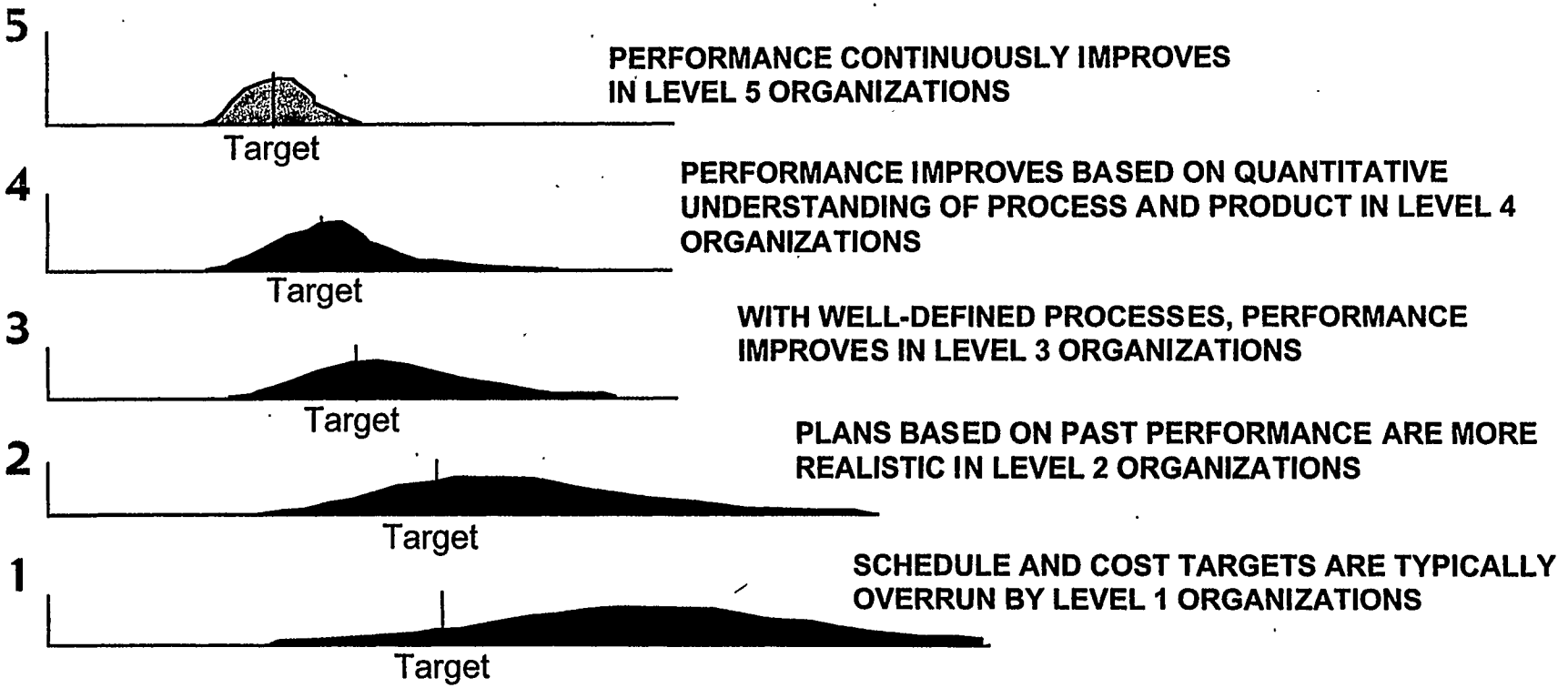
The CMM introduces the concept of "Key Process Areas", or KPAs. Each KPA is considered to be an essential element in achieving a certain level of maturity. Thus, the KPAs are the building blocks for the total software engineering process. The KPAs are grouped together to indicate 'levels' of process maturity



Note that KPAs start at level 2. Level 1 implies a lack of engineering discipline across a project.

The Structure of the CMM (2)

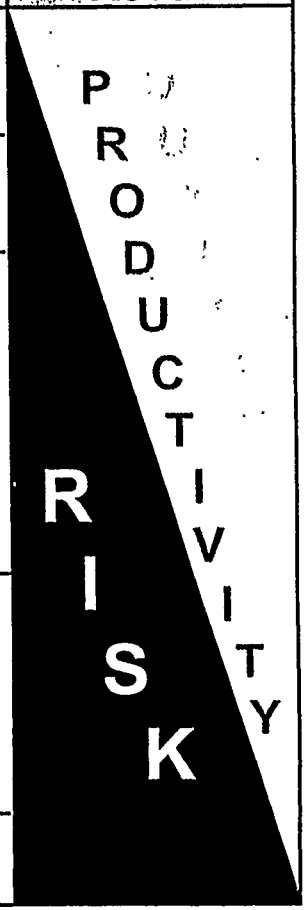
Each Maturity Level provides a foundation for improvements that can be made at higher Maturity Levels.



And with each level of maturity, the probability of meeting cost, schedule, performance, quality targets becomes more stable and predictable.

The Structure of the CMM (3)

In summary, each level reduces risk, while quality and productivity improves.

Level	Focus	Key Process Areas	Result
5 Optimizing	Continuous Improvement	Defect Prevention Process Change Management Technology Change Management	
4 Managed	Product and Process Quality	Quantitative Process Management Software Quality Management	
3 Defined	Engineering Process	Organization Process Focus Organization Process Definition Training Program Integrated Software Management Software Product Engineering Intergroup Coordination Peer Reviews	
2 Repeatable	Project Management	Requirements Management Software Project Planning Software Project Tracking & Oversight Software Subcontract Management Software Quality Assurance Software Configuration Management	
1 Initial	Heroes		

version 1.1

--SOURCE: SOFTWARE PRODUCTIVITY CONSORTIUM

The Structure of the CMM (4)

THE SEI CMM:

- ***Does Not*** **limit the choice of a Life Cycle Model**
- ***Does Not*** **require nor preclude specific software technologies or languages**
- ***Does Not*** **require that documentation conform to any particular set of standards**
- ***Does Not*** **prescribe a tool set**
- ***Does Not*** **impose a specific process**

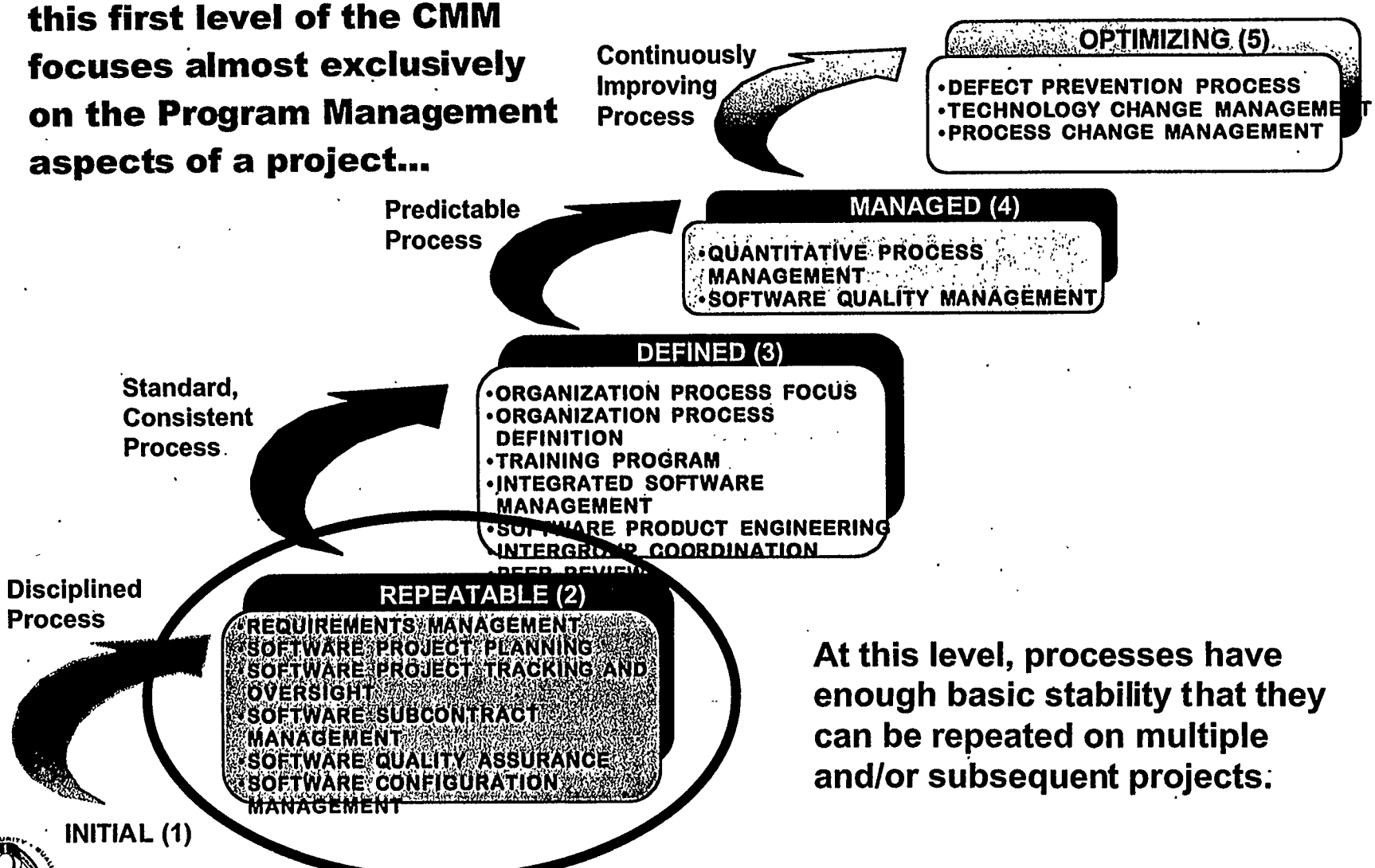
The CMM is simply a MODEL



CMM Level 2

"Repeatable"

this first level of the CMM focuses almost exclusively on the Program Management aspects of a project...



At this level, processes have enough basic stability that they can be repeated on multiple and/or subsequent projects.

CMM Level 2
Key Process Areas

REPEATABLE (2)

- REQUIREMENTS MANAGEMENT
- SOFTWARE PROJECT PLANNING
- SOFTWARE PROJECT TRACKING AND OVERSIGHT
- SOFTWARE SUBCONTRACT MANAGEMENT
- SOFTWARE QUALITY ASSURANCE
- SOFTWARE CONFIGURATION MANAGEMENT



L2 - Requirements Management

PURPOSE:

TO ESTABLISH A COMMON UNDERSTANDING BETWEEN THE CUSTOMER AND THE SOFTWARE PROJECT



GOALS:

- System requirements allocated to software are controlled to establish a baseline for software engineering and management use.
- Software plans, products and activities are kept consistent with the system requirements allocated to software.

L2 - Requirements Management

KEY QUESTIONS

- Are requirements baselined and under configuration control?
- As software requirements change, are adjustments made to software plans and associated work products?
- Does the project follow a written organization policy for managing system requirements?
- Are there measurements taken to determine the status of proposed changes to the requirements?



L2 - Software Project Planning



PURPOSE:

TO ESTABLISH REASONABLE PLANS FOR PERFORMING SOFTWARE ENGINEERING AND FOR MANAGING THE SOFTWARE PROJECT

GOALS:

- **Software estimates are documented for use in planning and tracking the software project.**
- **Software project activities and commitments are planned and documented.**
- **Affected groups and individuals agree to their commitments related to the software project.**

L2 - Software Project Planning

KEY QUESTIONS

- Are size, cost, and schedule estimates documented for use in planning and tracking the software project?
- Does the project have a Technical Management Plan and/or Software Development Plan (or perhaps an Integrated Development Plan)?
- Does the project follow a written organization policy for planning a software project?
- Do all affected groups and individuals agree to their commitments made for the software project?
- Are measurements used (e.g., progress metrics such as schedule milestones, percent completion) to determine the status of activities identified in the project plan?



L2 - Software Project Tracking and Oversight

PURPOSE:

TO ESTABLISH ADEQUATE VISIBILITY INTO ACTUAL PROGRESS SO THAT EFFECTIVE ACTIONS CAN BE TAKEN WHEN THE PROJECT'S PERFORMANCE DEVIATES SIGNIFICANTLY FROM THE SOFTWARE PLANS

GOALS:

- **Actual results and performance are tracked against the project plans.**
- **Corrective actions are taken and managed to closure when actual results and performance deviate significantly from the software plans.**
- **Changes to software commitments are agreed to by the affected groups and individuals.**

L2 - Software Project Tracking and Oversight

KEY QUESTIONS

- Is the project's actual schedule, size, and cost compared to the estimates?
- Is corrective action taken when actual results deviate significantly from the project's software plans?
- Does the project follow a written organization policy for tracking and controlling software development activities?
- Are changes to software commitments agreed to by all affected groups and individuals?
- Are project performance results, open issues, risks, and action items reviewed with senior management on a periodic basis?



L2 - Software Subcontract Management

PURPOSE:

SELECT QUALIFIED SUBCONTRACTORS AND MANAGE THEM EFFECTIVELY



GOALS:

- **The prime contractor selects qualified software subcontractors.**
- **The prime contractor and software subcontractor agree to their commitments to each other.**
- **The prime contractor and software subcontractor maintain ongoing communications.**
- **The prime contractor tracks the software subcontractor's actual results and performance against commitments.**

L2 - Software Subcontract Management

KEY QUESTIONS

- Is a documented procedure used for selecting subcontractors?
- Are periodic technical interchanges held with subcontractors?
- Are results and performance of the subcontractor tracked against their commitments?
- Does the project follow a written organization policy for managing software subcontracts?
- Are software subcontract activities reviewed with the project manager on both a periodic and event-driven basis?



L2 - Software Quality Assurance

PURPOSE:

TO PROVIDE MANAGEMENT WITH THE APPROPRIATE VISIBILITY INTO THE PROCESS USED AND PRODUCTS BEING BUILT BY THE SOFTWARE PROJECT



GOALS:

- **Software QA activities are planned.**
- **Adherence of software products and activities to the applicable standards, procedures, and requirements is verified objectively.**
- **Affected groups and individuals are informed of software quality assurance activities and results.**
- **Non-compliance issues that cannot be resolved within the software project are addressed by senior management.**

L2 - Software Quality Assurance

KEY QUESTIONS

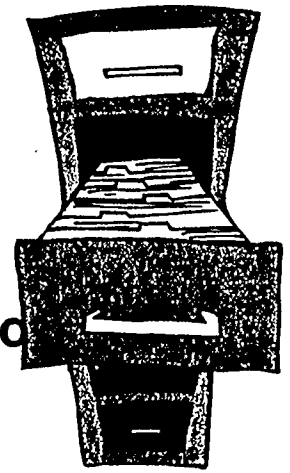
- Is there a documented Software QA Plan which includes activities for software process evaluations? Have adequate resources been identified to support this?
- Is there objective evidence (i.e., records) of Software QA activities?
- Are the results of Software QA reviews and audits provided to affected groups and individuals?
- Does the project follow a written organizational policy for implementing Software QA?
- Are Software QA activities reviewed with senior management on a periodic basis?



L2 - Software Configuration Management

PURPOSE:

ESTABLISH AND MAINTAIN THE INTEGRITY OF THE PRODUCTS OF THE SOFTWARE PROJECT THROUGHOUT THE SOFTWARE LIFE CYCLE



GOALS:

- **Software CM Activities are planned.**
- **Selected software work products are identified, contro and available.**
- **Changes to identified software work products are controlled.**
- **Affected groups and individuals are informed of the status and content of software baselines.**

L2 - Software Configuration Management

KEY QUESTIONS

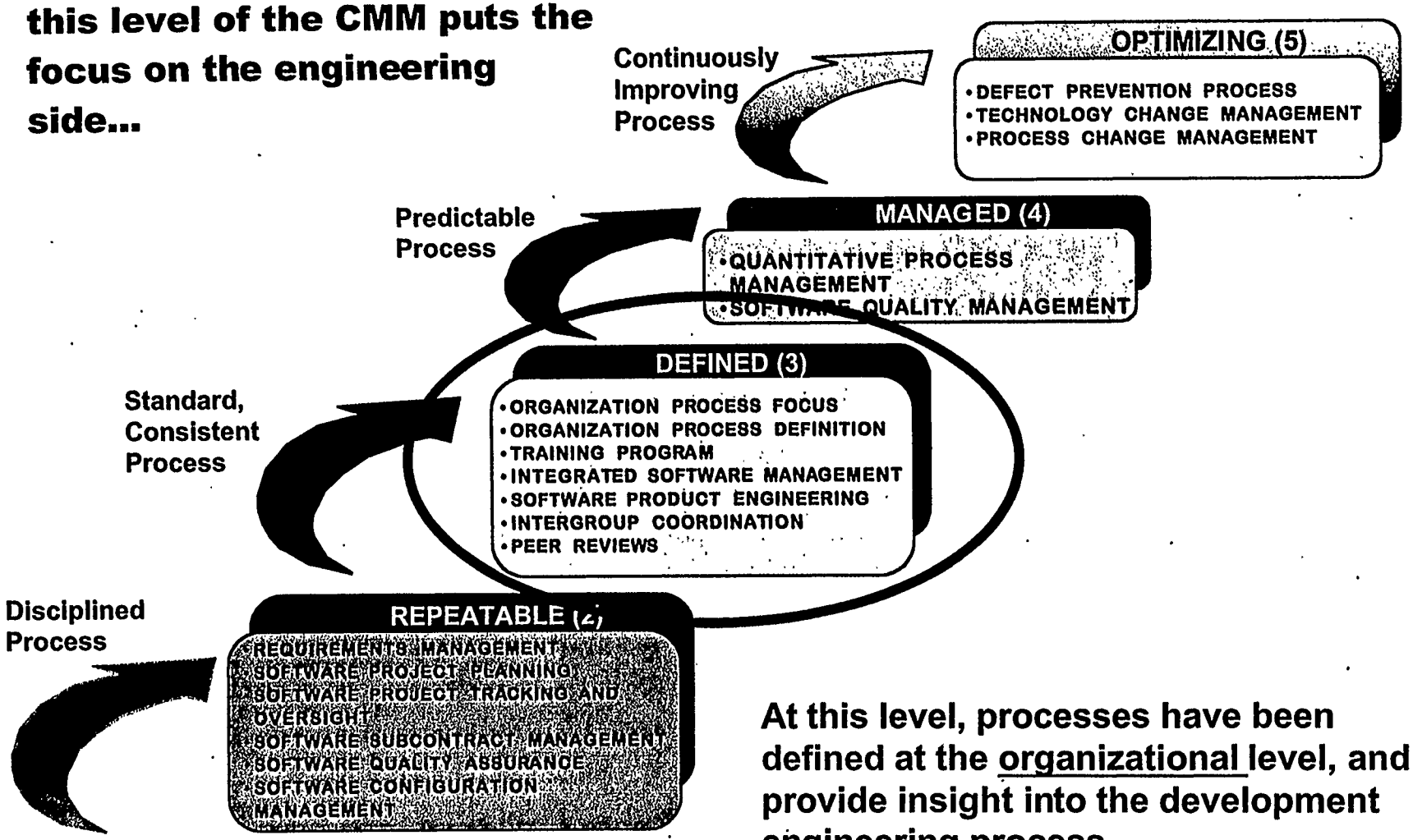
- Is there a documented Software CM Plan?
- Are software work products (e.g., plans/procedures, requirements, code) under CM control?
- Does the project follow a documented procedure to control changes to software work products?
- Does the project follow a written organization policy for Software CM?
- Are standard reports on software baselines and change requests distributed to affected groups and individuals (e.g., CCB minutes)?



CMM Level 3

"Defined"

this level of the CMM puts the focus on the engineering side...



At this level, processes have been defined at the organizational level, and provide insight into the development engineering process.

CMM Level 3
Key Process Areas

DEFINED (3)

- ORGANIZATION PROCESS FOCUS
- ORGANIZATION PROCESS DEFINITION
- TRAINING PROGRAM
- INTEGRATED SOFTWARE MANAGEMENT
- SOFTWARE PRODUCT ENGINEERING
- INTERGROUP COORDINATION
- PEER REVIEWS



L3 - Organization Process Focus

PURPOSE:

TO ESTABLISH ORGANIZATIONAL RESPONSIBILITY FOR SOFTWARE PROCESS ACTIVITIES THAT IMPROVE SOFTWARE PROCESS CAPABILITY

GOALS:

- **Software process development and improvement activities are coordinated across the organization.**
- **The strengths and weaknesses of the software processes used are identified relative to a process standard.**
- **Organization-level process development and improvement activities are planned.**

L3 - Organization Process Focus

KEY QUESTIONS

- Is there a Process Group which coordinates software process and development activities across the organization (e.g., SEPG)? Have adequate resources been provided for this?
- Is your organization's software process assessed periodically?
- Do one or more individuals have responsibility for the organization's software process activities?
- Does senior management review activities associated with developing and improving the software process?



L3 - Organization Process Definition

PURPOSE:

DEVELOP AND MAINTAIN A USABLE SET OF SOFTWARE PROCESS ASSETS THAT IMPROVE PROCESS PERFORMANCE ACROSS THE PROJECTS

GOALS:

- **A standard software process for the organization is developed and maintained.**
- **Information related to the use of the organization's standard software process by the software projects is collected, reviewed, and made available.**



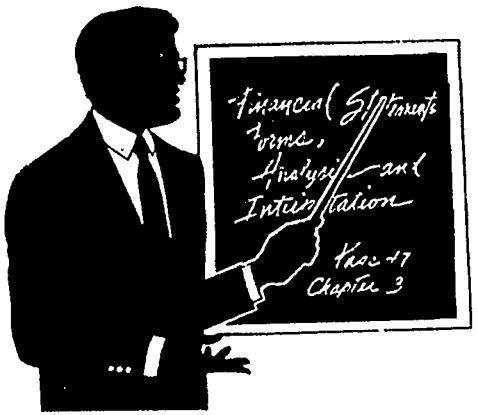
L3 - Organization Process Definition

KEY QUESTIONS

- Has your organization developed, and does it maintain, a standard software process (from which a project can derive its own SDP)?
- Does the organization follow a written policy for developing and maintaining its standard software process (e.g., the Process Library, Standards & Conventions)?
- Are Software QA reviews and audits performed on the software activities and resulting work products?



L3 - Training Program



PURPOSE:

DEVELOP THE SKILLS AND KNOWLEDGE OF INDIVIDUALS SO THEY CAN PERFORM THEIR ROLES EFFECTIVELY AND EFFICIENTLY

GOALS:

- **Training activities are planned.**
- **Training for developing the skills and knowledge needed to perform software management and technical roles is provided.**
- **Individuals in the software engineering group and software-related groups receive the training necessary to perform their roles.**

L3 - Training Program

KEY QUESTIONS

- Is there a training plan for the project? For you?
- Do software engineering members receive the training necessary to perform their roles?
- Does your organization follow a written organization policy to meet its training needs?
- Are measurements used to determine the quality of the training program?

L3 - Integrated Software Management

PURPOSE:

INTEGRATE SOFTWARE ENGINEERING AND MANAGEMENT ACTIVITIES INTO A COHERENT, DEFINED PROCESS THAT IS TAILORED FROM THE ORGANIZATION'S STANDARD PROCESS

GOALS:

- **The project's defined software process is a tailored version of the organization's standard software process.**
- **The project is planned and managed according to the project's defined software process.**

ISM evolves from SPP and PTO at Level-2. At Level 3, there are organization-wide processes and metrics that are tailored for individual programs.

L3 - Integrated Software Management

KEY QUESTIONS

- Was the project's process developed by tailoring the organization's standard process?
- Is the project managed in accordance with the project's defined process?
- Does the project follow a written organizational policy requiring that the software project be planned and managed using the organization's standard software process?



L3 - Software Product Engineering

PURPOSE:

CONSISTENTLY PERFORM A WELL-DEFINED ENGINEERING PROCESS THAT INTEGRATES ALL THE SOFTWARE ENGINEERING ACTIVITIES



GOALS:

- **The software engineering tasks are defined, integrated, and consistently performed to produce the software.**
- **Software work products are kept consistent with each other.**

L3 - Software Product Engineering

KEY QUESTIONS

- Are software work products (e.g., plans, requirements, code) produced according to the project's defined process?
- Is there traceability from software requirements to design, code, and test cases?
- Does the project follow a written organizational policy for performing software engineering activities?
- Are there Software QA reviews for traceability of requirements, design, code, and test cases?



L3 - Intergroup Coordination

PURPOSE:

ESTABLISH A MEANS FOR THE SOFTWARE ENGINEERING GROUP TO PARTICIPATE ACTIVELY WITH OTHER ENGINEERING GROUPS



GOALS:

- **The customer's requirements are agreed to by all affected groups.**
- **The commitments between the engineering groups are agreed to by the affected groups.**
- **The engineering groups identify, track, and resolve intergroup issues.**

L3 - Intergroup Coordination

KEY QUESTIONS

- Does the software engineering group and other groups (e.g., IPT, PM) collaborate with the customer to establish system requirements?
- Do the engineering groups agree to the commitments as represented in the overall project plan?
- Is there a written organizational policy that guides the establishment of interdisciplinary engineering teams?
- Do support tools used by different engineering groups enable effective communication and coordination (e.g., compatible word processing and scheduling systems)?



L3 - Peer Reviews



PURPOSE:

REMOVE DEFECTS FROM THE SOFTWARE WORK PRODUCTS EARLY AND EFFICIENTLY

GOALS:

- **Peer review activities are planned.**
- **Defects in the software work products are identified and removed.**

L3 - Peer Reviews

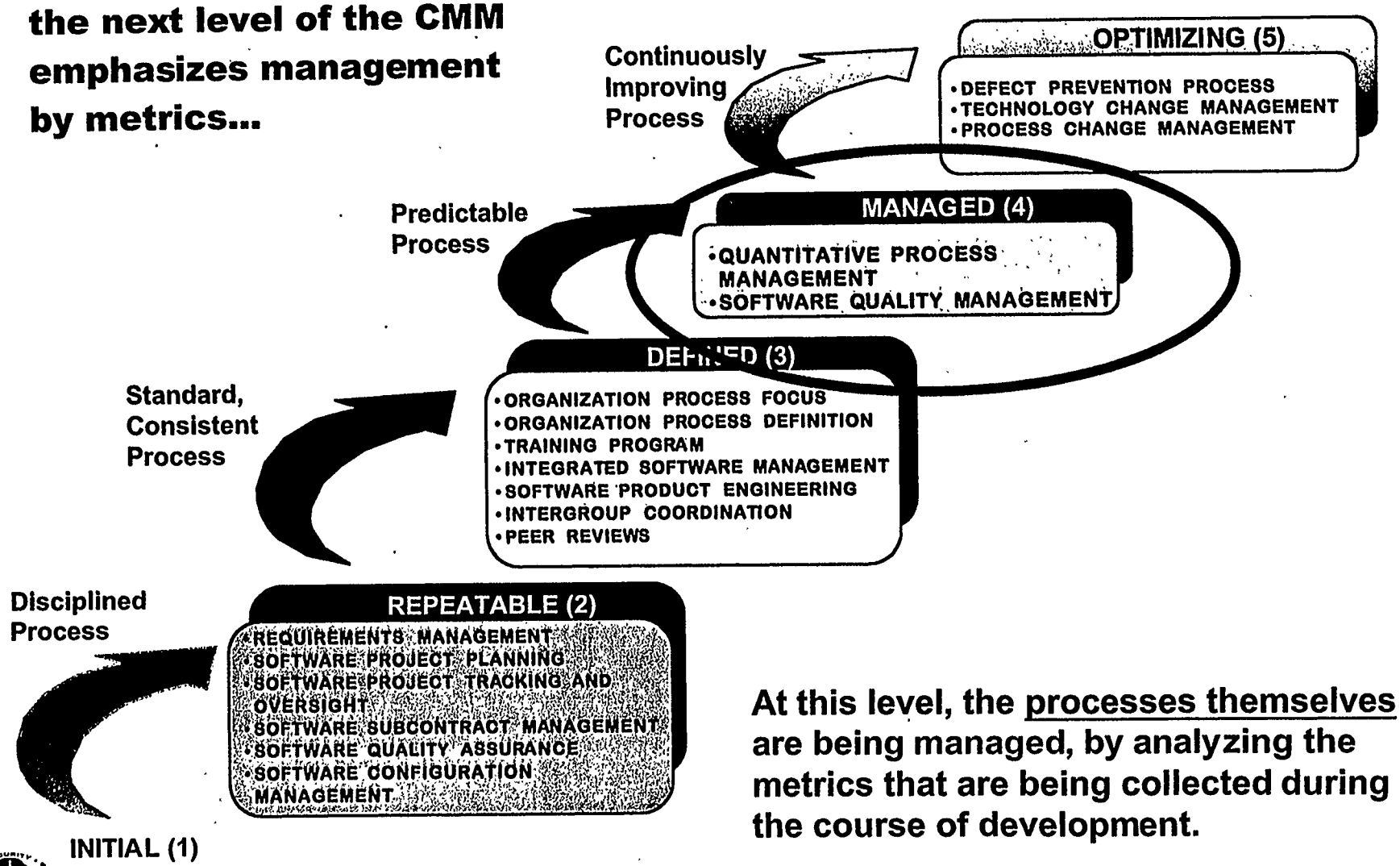
KEY QUESTIONS

- Are peer reviews planned?
- Are action items from peer reviews documented and tracked until closure?
- Does the project follow a written organizational policy for performing peer reviews?
- Are peer review activities and work products subjected to Software QA review?



CMM Level 4

"Managed"
the next level of the CMM emphasizes management by metrics...



At this level, the <u>processes themselves</u> are being managed, by analyzing the metrics that are being collected during the course of development.

CMM Level 4
Key Process Areas

MANAGED (4)

- QUANTITATIVE PROCESS MANAGEMENT
- SOFTWARE QUALITY MANAGEMENT



L4 - Software Quality Management

PURPOSE:

DEFINING QUALITY GOALS FOR THE SOFTWARE PRODUCTS AND ESTABLISHING PLANS TO ACHIEVE THESE GOALS

GOALS

- **The project's software quality management activities are planned.**
- **Measurable goals for software product quality and their priorities are defined.**
- **Actual progress toward achieving the quality goals for the software products is quantified and managed.**

L4 - Software Quality Management

KEY QUESTIONS

- Is there a documented Software Quality Management Plan?
- Does the project use measurable and prioritized goals for managing the quality of its software product (e.g., functionality, reliability, and maintainability)?
- Does the project follow a written organizational policy for managing software quality?



L4 - Quantitative Process Management

PURPOSE:

CONTROL THE PROCESS PERFORMANCE OF THE SOFTWARE PROJECT QUANTITATIVELY

GOALS:

- **The quantitative process management activities are planned.**
- **The process performance of the project's defined software process is controlled quantitatively.**
- **The process capability of the organization's standard software process is known in quantitative terms.**



L4 - Quantitative Process Management

KEY QUESTIONS

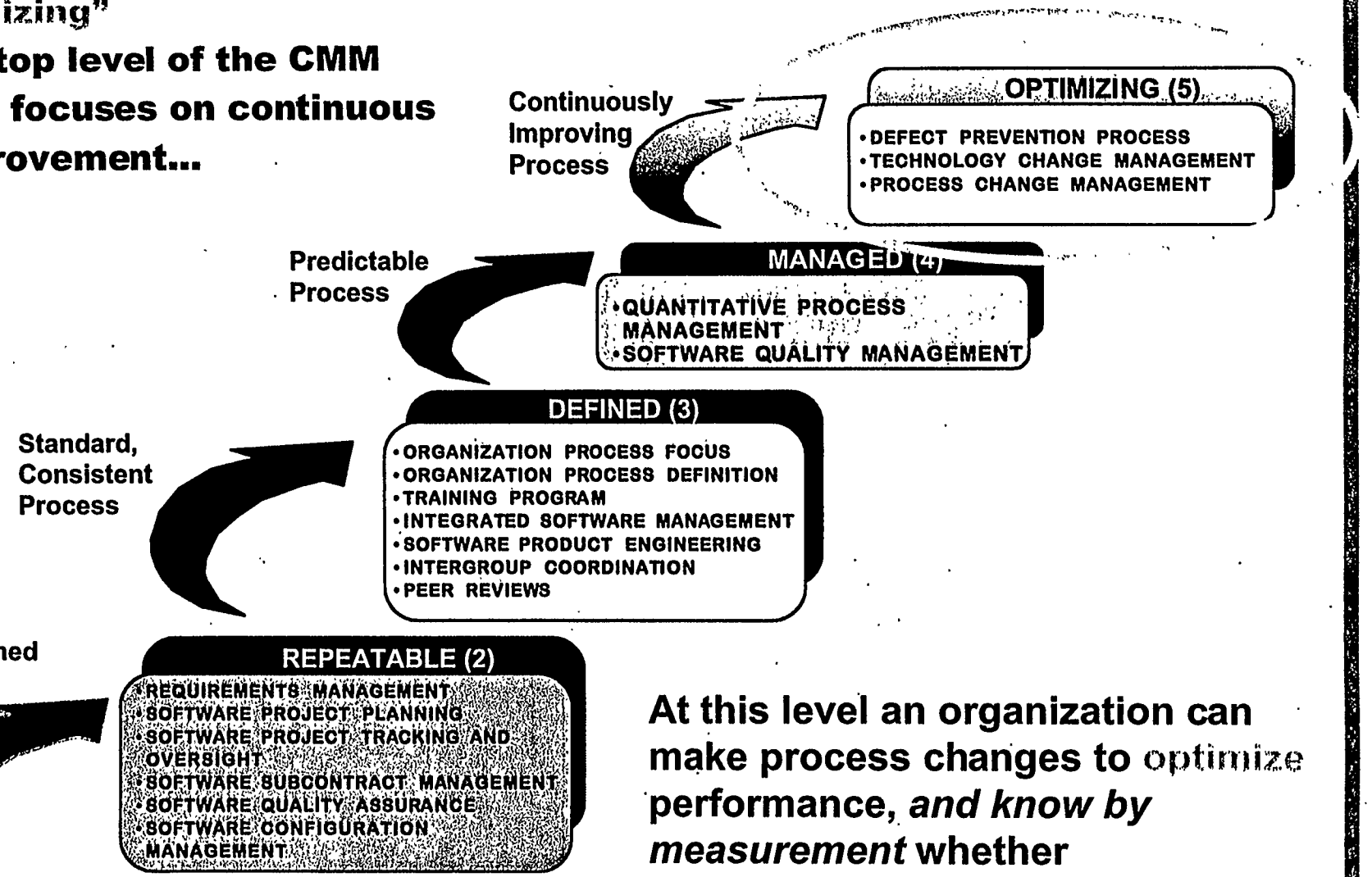
- Does the project follow a documented plan for conducting quantitative process management?
- Is the performance of the project controlled using quantitative analytical methods (e.g., Statistical Process Control)?
- Does the project follow a written organizational policy for measuring and controlling the performance of the project's defined software process (e.g., project plans for how to identify, analyze, and control special causes for variation)?



CMM Level 5

"Optimizing"

the top level of the CMM now focuses on continuous improvement...



At this level an organization can make process changes to optimize performance, *and know by measurement* whether improvement has occurred.

CMM Level 5
Key Process Areas

- DEFECT PREVENTION
- TECHNOLOGY CHANGE MANAGEMENT
- PROCESS CHANGE MANAGEMENT



Defect Prevention

PURPOSE:

ANALYZING DEFECTS AND TAKING SPECIFIC ACTIONS TO PREVENT THE OCCURRENCE OF THOSE TYPES OF DEFECTS IN THE FUTURE

GOALS

- **Defect prevention activities are planned.**
- **Common causes of defects are sought out and identified.**
- **Common causes of defects are prioritized and systematically eliminated.**

KEY QUESTIONS

- Is there a documented plan for defect prevention?
- Does the project hold causal analysis meetings to identify common causes of defects?
- Once identified, are common causes of defects prioritized and systematically eliminated?
- Does the project follow a written organizational policy for defect prevention activities?



Technology Change Management

PURPOSE:

IDENTIFYING, SELECTING, AND EVALUATING NEW TECHNOLOGIES, AND INCORPORATING EFFECTIVE TECHNOLOGIES INTO THE ORGANIZATION

GOALS:

- **Incorporation of technology changes are planned.**
- **New technologies are evaluated to determine their effect on quality and productivity.**
- **Appropriate new technologies are transferred into normal practice across the organization.**

KEY QUESTIONS

- Is there a documented plan for technology change management?
- Are new technologies evaluated to determine their effect on quality and productivity?
- Does the organization follow a documented procedure for incorporating new technologies into the organization's standard software process?
- Does process data exist in the selection of new technology?



Change Management

PURPOSE:

DEFINE PROCESS IMPROVEMENT GOALS AND SYSTEMATICALLY IDENTIFY, EVALUATE, AND IMPLEMENT IMPROVEMENTS TO THE ORGANIZATION'S STANDARD SOFTWARE PROCESS

GOALS

- **Continuous process improvement is planned.**
- **Participation in the organization's software process improvement activities is organization wide.**
- **The organization's standard software process and the projects' defined software processes are improved continuously.**

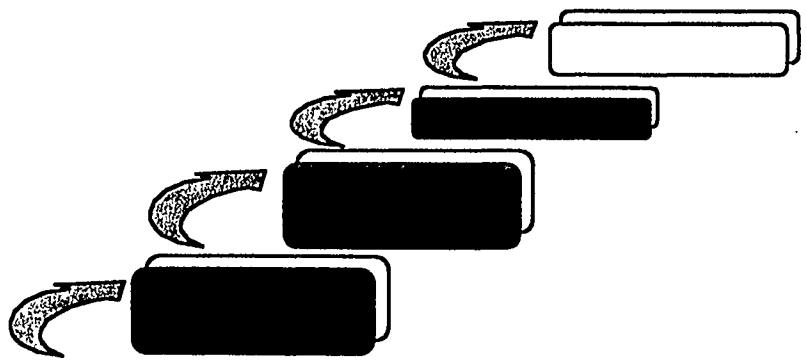
KEY QUESTIONS

- Does the organization follow a documented procedure for developing and maintaining plans for software process improvement?
- Do people throughout your organization participate in software process improvement activities?
- Does the organization follow a written policy for implementing software process improvement?

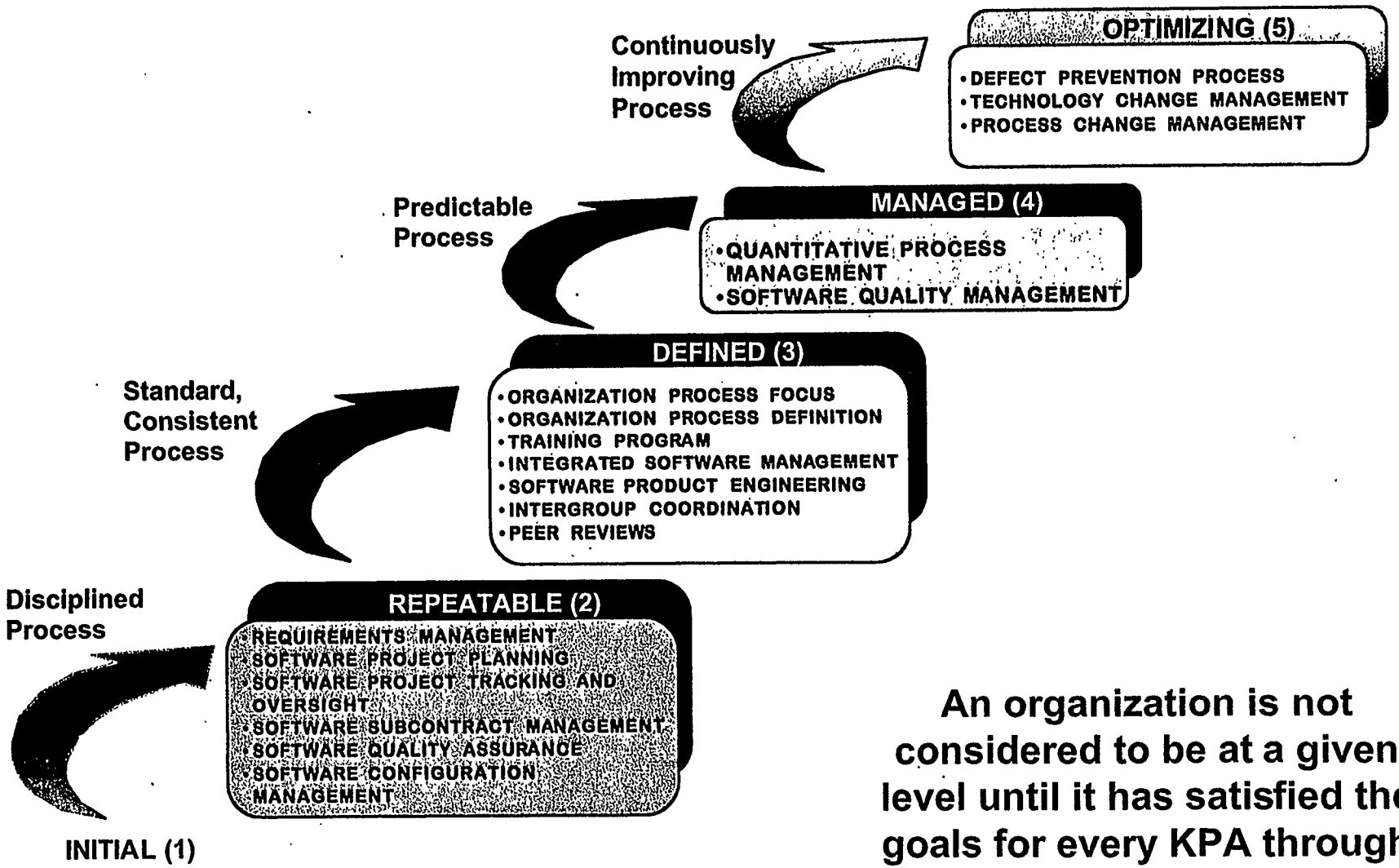


Presentation Objectives

- **An overview of the SEI SW-CMM Version 1.1**
- **Achieving an SEI Capability Maturity Level**
- **An introduction to Software Process Improvement**
- **Interpreting the CMM for Software Process Improvement**



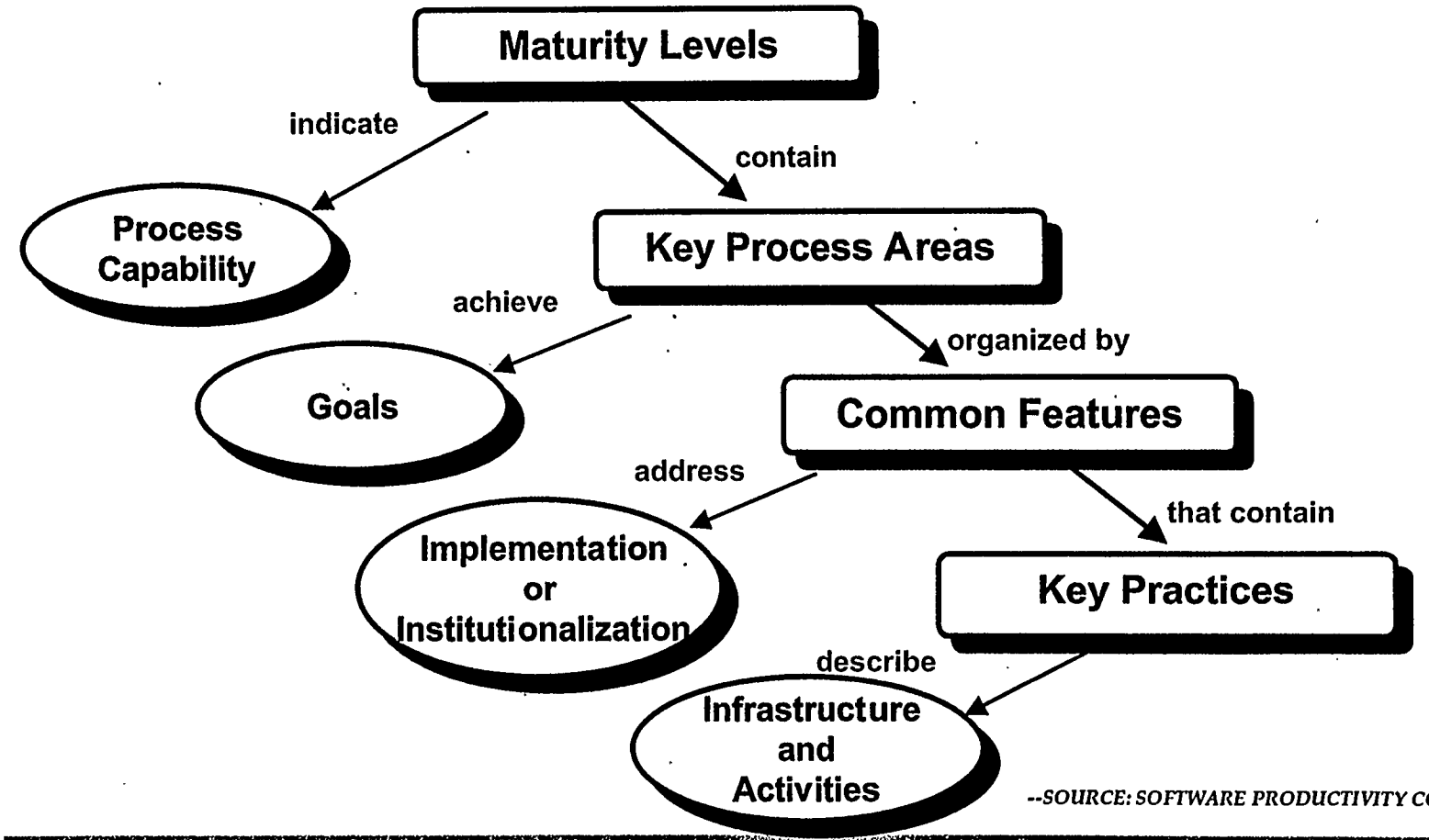
Achieving an SEI CMM Maturity Level (1)



An organization is not considered to be at a given level until it has satisfied the goals for every KPA through that level.

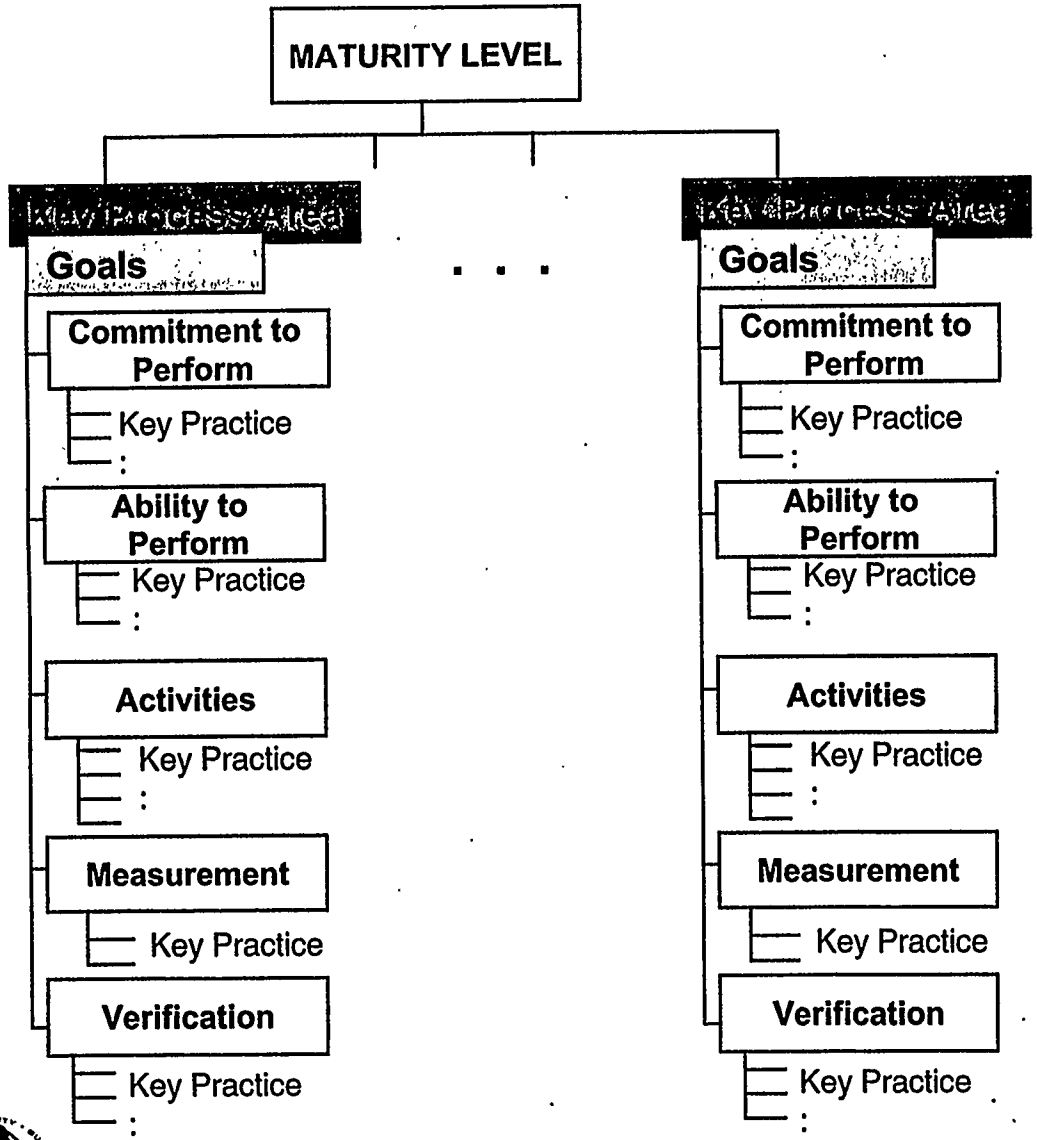
Achieving an SEI CMM Maturity Level (2)

- **Each KPA has Goals and Practices considered important for establishing process capability in that Area.**
- **The Goals summarize the scope, boundaries, and intent of each KPA.**
- **The Practices describe the infrastructure and activities that contribute to the effective implementation and institutionalization of the KPA.**



--SOURCE: SOFTWARE PRODUCTIVITY CONSORTIUM

Achieving an SEI CMM Maturity Level (3)



Each CMM Maturity Level is organized like this

- **When an organization is assessed against the CMM, objective evidence is reviewed to see that the goals are satisfied at each level.**
- **If a goal is not satisfied, the KPA is not satisfied, and the organization cannot claim that level or any higher levels.**
- **The key practices supporting each goal are used to determine if the goal has been satisfied.**

Example: Satisfying Requirements Management KPA (1)

PURPOSE:

TO ESTABLISH A COMMON UNDERSTANDING BETWEEN THE CUSTOMER AND THE SOFTWARE PROJECT



To Satisfy This KPA, Two Goals Must Be Met:

- System requirements allocated to software are controlled to establish a baseline for software engineering and management use.
- Software plans, products and activities are kept consistent with the system requirements allocated to the software.

To Satisfy These Two Goals, 12 Key Practices are reviewed:

- Commitment 1: The project follows a written organizational policy for managing the system requirements allocated to software.

Example: Satisfying Requirements Management KPA (2)

- Ability 1: For each project, responsibility is established for analyzing the system requirements and allocating them to hardware, software and other system components.
- Ability 2: The allocated requirements are documented.
- Ability 3: Adequate resources and funding are provided for managing the allocated requirements.
- Ability 4: Members of the software engineering group and other software-related groups are trained to perform their requirements management activities.
- Activity 1: The software engineering group reviews the allocated requirements before they are incorporated into the software project.
- Activity 2: The software engineering group uses the allocated requirements as the basis for software plans, work products, and activities.
- Activity 3: Changes to the allocated requirements are reviewed and incorporated into the software project.

Example: Satisfying Requirements Management KPA (3)

- Measurement 1: Measurements are made and used to determine the status of the activities for managing the allocated requirements.
- Verification 1: The activities for managing the allocated requirements are reviewed with senior management on a periodic basis.
- Verification 2: The activities for managing the allocated requirements are reviewed with the project manager on both a periodic and event-driven basis.
- Verification 3: The software quality assurance group reviews and/or audits the activities and work products for managing the allocated requirements and reports the results.

SEI's Appraisal Method

CBA IPI = CMM-Based Appraisal for Internal Process Improvement

- Uses the Software Capability Maturity Model Version 1.1 as a process reference model
- Uses a formalized assessment process that complies with the CMM Appraisal Framework
- Involves senior management as the assessment sponsor
- Bases the assessment on the sponsor's business goals and needs
- Observes strict confidentiality by guaranteeing that no information will be attributed to an individual or project
- Approaches the assessment as a collaboration between the assessment team and the organizational participants





From CMM-Based SPI - SEI 94-TR13

- The KPA profiles from CBA IPIs for organizations assessed at Level 1 indicate that **Software Quality Assurance, Software Project Planning** and **Software Project Tracking and Oversight** are the least frequently satisfied KPAs

- The KPA profiles for organizations assessed at Level 2 indicate that **Integrated Software Management, Organization Process Definition** and **Training Program** are the least frequently satisfied KPAs





Carnegie Mellon University
Software Engineering Institute

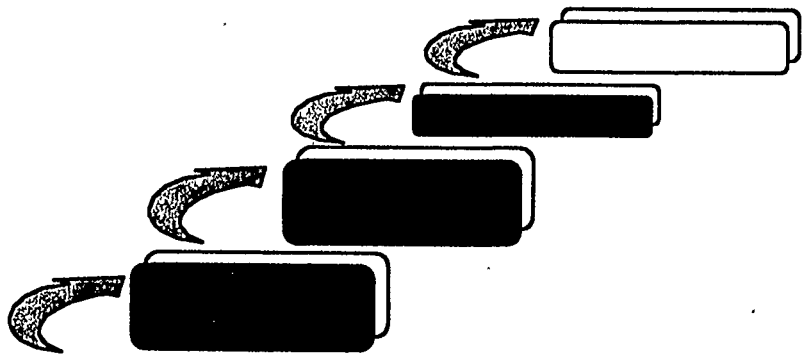
From SEI Process Maturity Profile, August 1999

- Median time to move from maturity level 1 to 2 is 25.5 months for organizations that began their CMM-based SPI effort in 1992 or later
- Median time to move from maturity level 2 to 3 is 19 months for organizations that began their CMM-based SPI effort in 1992 or later
- All groupings exhibit a similar pattern for moving from maturity level 1 to 2 and level 2 to 3: level 2 to 3 tends to be faster and have less variance



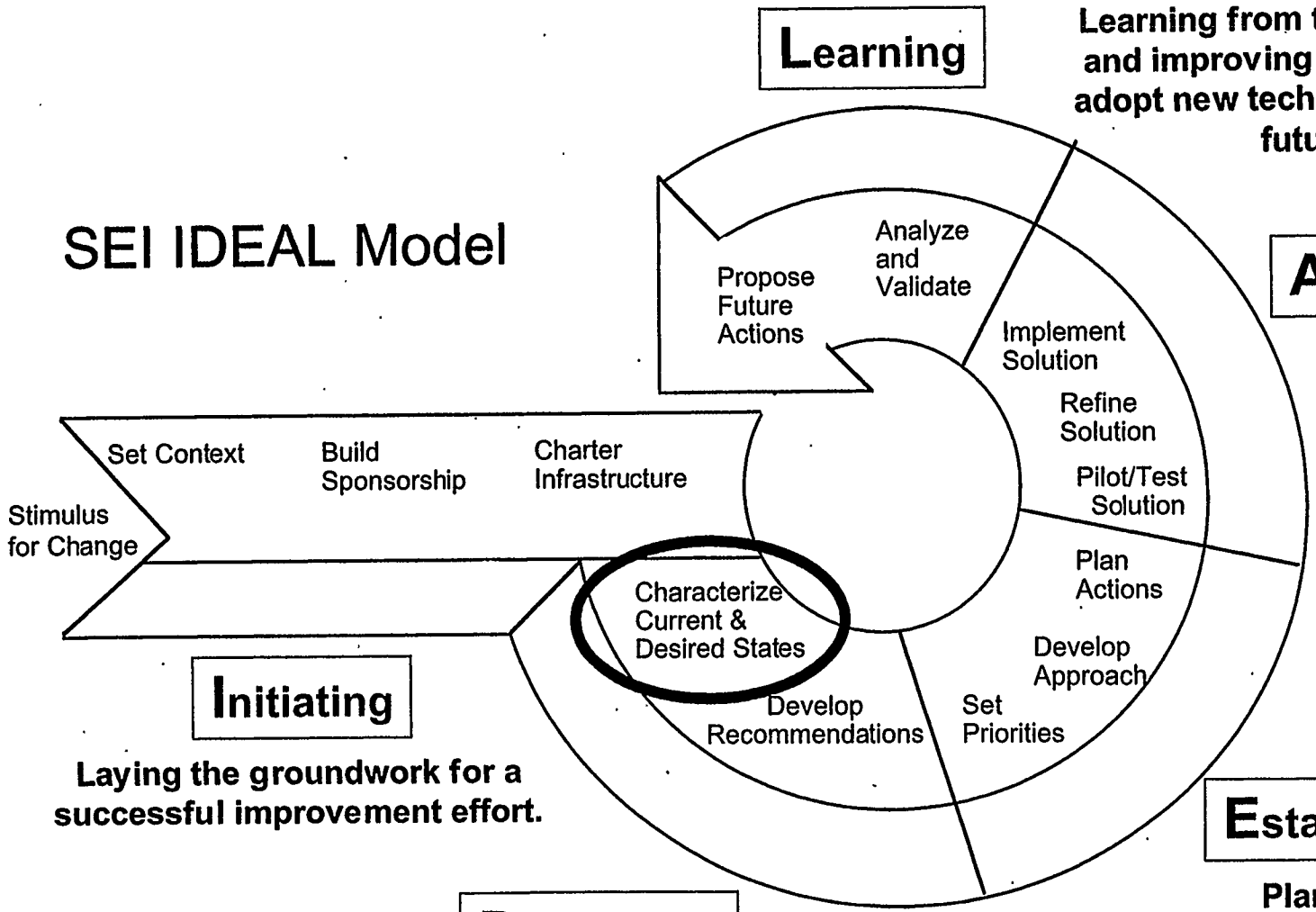
Presentation Objectives

- **An overview of the SEI SW-CMM Version 1.1**
- **Achieving an SEI Capability Maturity Level**
- **An introduction to Software Process Improvement**
- **Interpreting the CMM for Software Process Improvement**



Life Cycle Model for Software Process Improvement

SEI IDEAL Model





Carnegie Mellon University
Software Engineering Institute

Benefits of Software Process Improvement From CMM-Based SPI - SEI 94-TR13

Initial Study of Benefits

Participating Organizations:

Bull HN

GTE Government Systems

Hewlett Packard

Hughes Aircraft Co.

Loral Federal Systems

Lockheed Sanders

Motorola

Northrop

Schlumberger

Siemens Stromberg-Carlson

Texas Instruments

U. S. Air Force Tinker AFB Air Logistics Center

U. S. Navy Fleet Combat Direction Systems Support Activity

Source. "CMM-based software process improvement: initial results." (SEI 94-TR-13)

© 1999 by Carnegie Mellon University

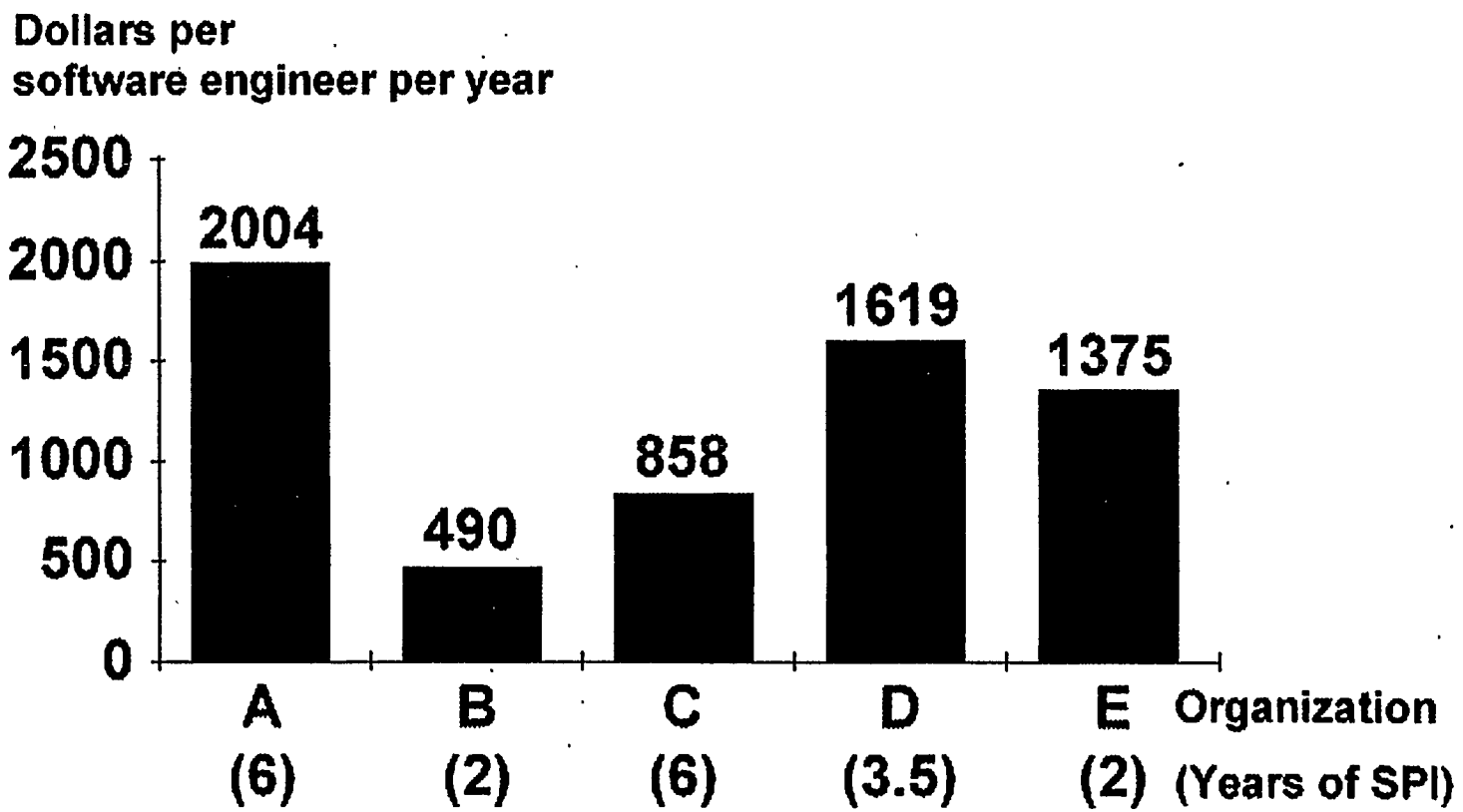




Carnegie Mellon University
Software Engineering Institute

Benefits of Software Process Improvement
From CMM-Based SPI - SEI 94-TR13

Yearly Expenditure on SPI



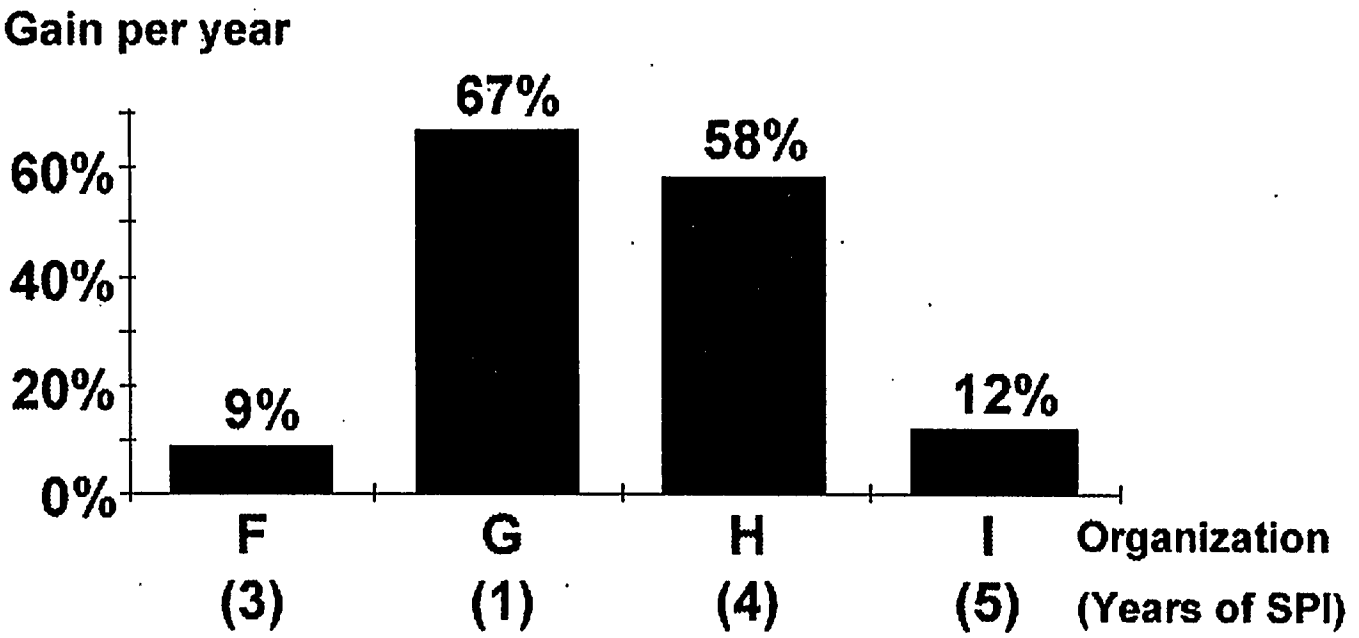




Carnegie Mellon University
Software Engineering Institute

Benefits of Software Process Improvement From CMM-Based SPI - SEI 94-TR13

Productivity

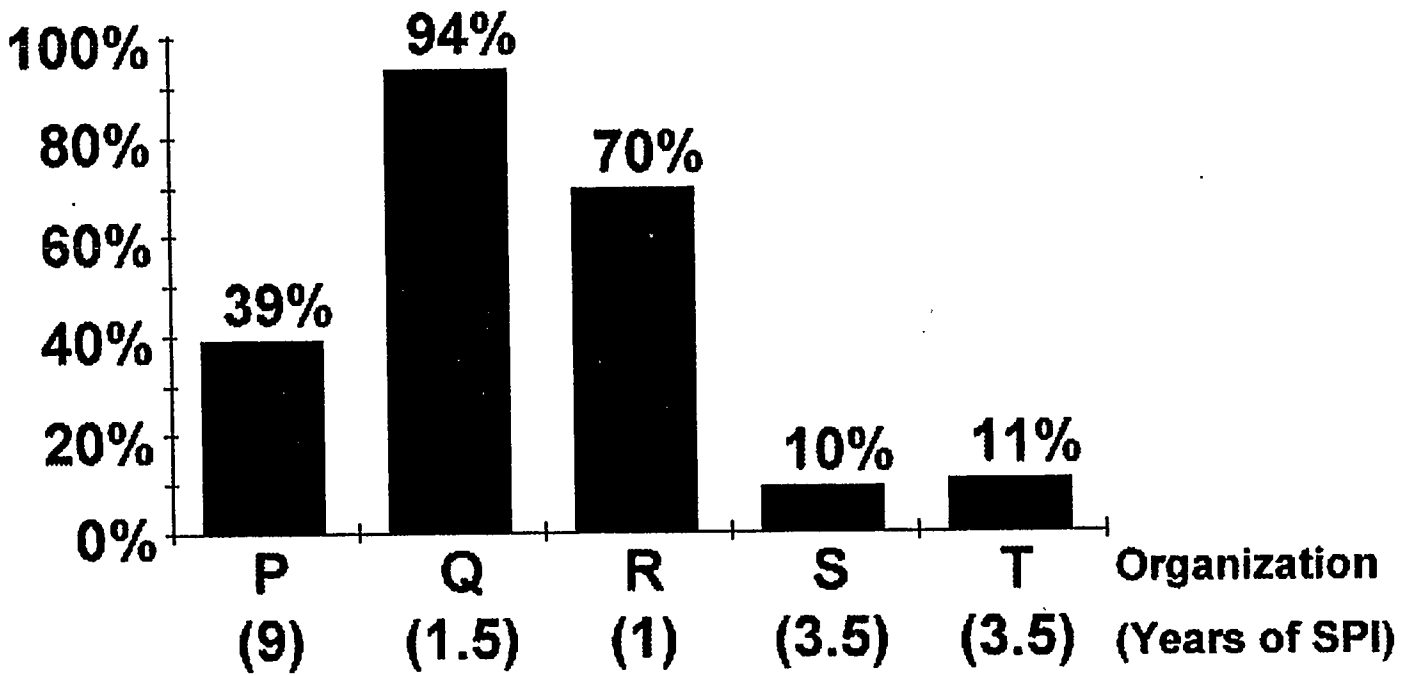




Benefits of Software Process Improvement From CMM-Based SPI - SEI 94TR-13

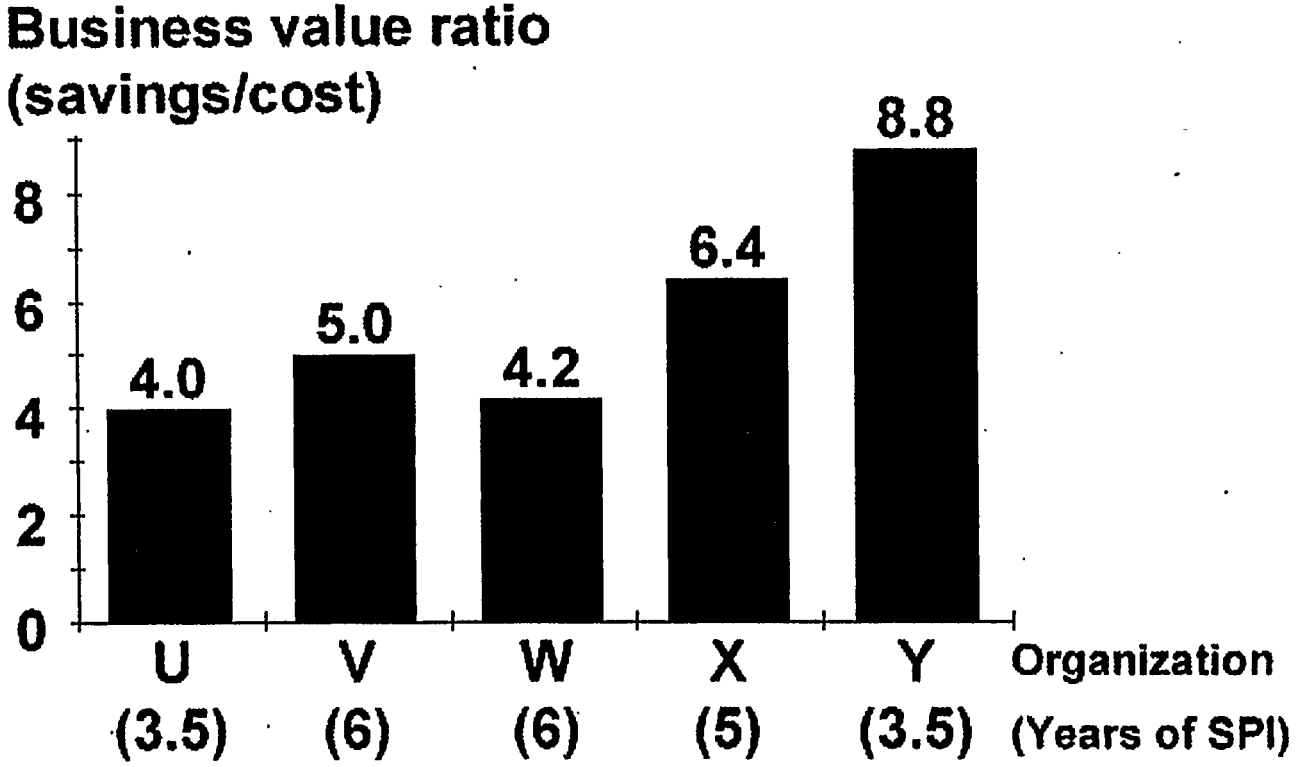
Defects

Reduction per year



Benefits of Software Process Improvement From CMM-Based SPI - SEI 94-TR13

Business Value

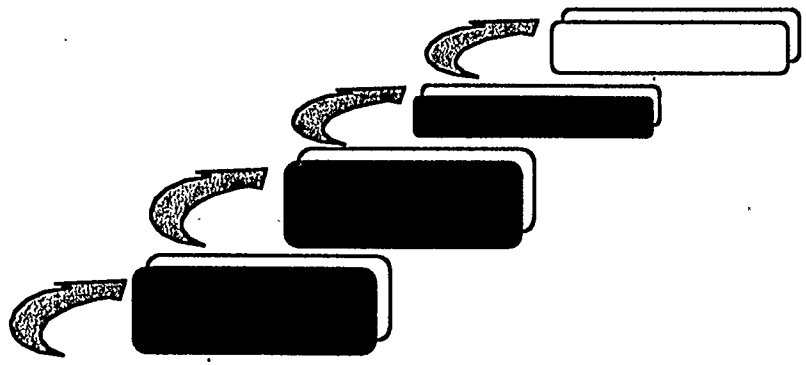


Summary of SEI-94-TR13

Category	Range	Median	Data pts
cost of SPI per software engineer/yr.	$490-2004	$1375	5
gain in productivity per year	9%-67%	35%	4
reduction in time to market per year	15%-23%	----	2
reduction in post-release defects per year	10%-94%	39%	5
business value ratio (benefit/cost)	4.0-8.8:1	5.0:1	5

Presentation Objectives

- **An overview of the SEI SW-CMM Version 1.1**
- **Achieving an SEI Capability Maturity Level**
- **An introduction to Software Process Improvement**
- **Interpreting the CMM for Software Process Improvement**



Interpreting the CMM

PURPOSE:

To adapt and express the CMM in a way that makes it more relevant and recognizable to your specific software development environment.

RULES for the CMM Interpretation:

- The KPAs must be present in the CMM interpretation, with the exception of Software Subcontract Management if it is not applicable
- The goals of the KPAs must be present in the CMM interpretation.
- The goals must be supported by the key practices in the CMM interpretation.
- The CMM interpretation must be documented along with rationale for significant changes to the key practices.

Source: Software Productivity Consortium

Interpreting the CMM
Interpreting Techniques (1)

- **Adopting: the decision to accept, without modification, a CMM key practice because your organization is performing this practice already or plans to perform this practice.**

- **Translating: the process of mapping the names of roles and work products in your organization to those identified in the CMM**
  - **Example: engineering group, SQA group**
  - **Example: allocated requirements, software development plan**

- **Defining: the process of defining CMM terms that have a global meaning in your organization's interpretation of the CMM.**
  - **Example: organization, senior management, customer**

Source: Software Productivity Consortium



Interpreting the CMM
Interpreting Techniques (2)

- **Customizing: matching the scale of the key practice to the scale of the project or organization.**
  - **Example: level of formality that is appropriate for your organization**
  - **Example: proper frequency and triggering events for certain activities**

- **Substituting: process of defining an alternative key practice to be used instead of the one called for by the CMM.**
  - **Note: not the same as customizing - substituting provides an entirely different practice**
  - **Caution: substituted practice must still meet the goals of the KPA**

- **Omitting: process of determining that a key practice is not applicable to your particular environment**
  - **Note: first try finding a suitable alternative and substitute**
  - **Caution: be sure to document the rationale for omitting**

Source: Software Productivity Consortium

Interpreting the CMM
Interpreting Techniques (3)

- **Highlighting: calling attention to certain key practices that are especially valuable or potentially risky for your environment.**
  - **Example: issues identified for process improvement**
  - **Example: key practices that carry a lot of weight in terms of supporting CMM goals**

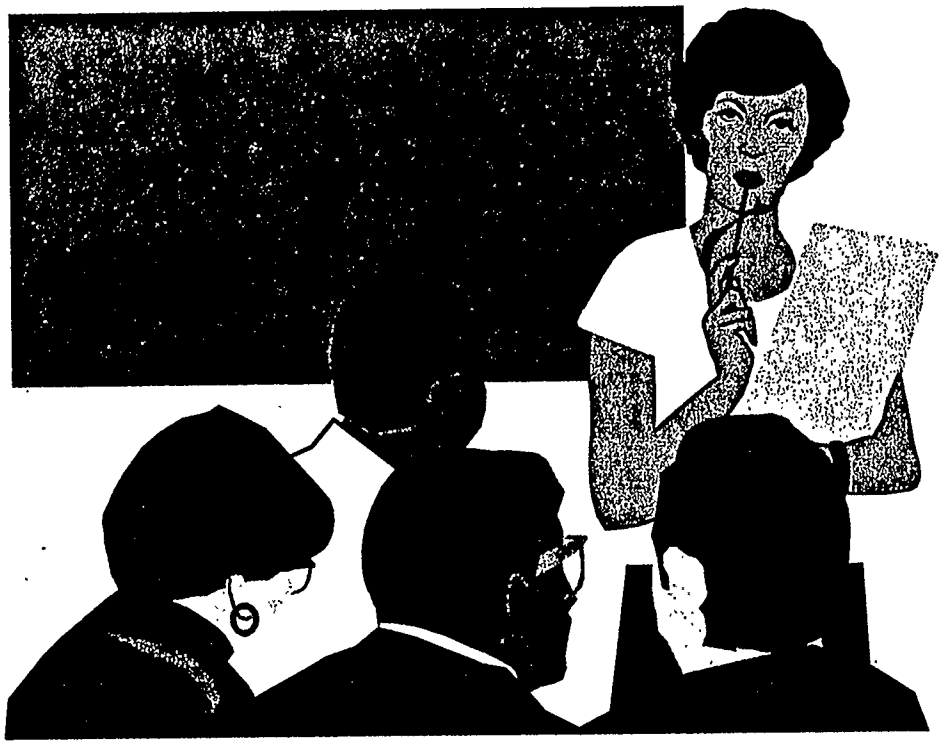
- **Abstracting: process of applying a CMM key practice to any type of product or service - not just software.**

- **Augmenting: identifying and including issues that are critical to your environment but are not covered in the Software CMM.**
  - **Example: Communication KPA from People CMM**
  - **Example: Manage Risk PA from System CMM**

- **Determining Goal Support: process of determining if your interpretation of the key practices is reasonable and meets the goals of the CMM.**

Source: Software Productivity Consortium

Questions



Software Quality Forum 2000

Front-End Loading with Facilitated Joint Application Development Techniques

Santa Fe, NM - April 18-19, 2000

What is Joint Application Development (JAD)?

JAD is a structured meeting, conducted by a neutral facilitator, designed to extract high-quality information from the meeting participants, using a compressed timeframe and a workshop environment to enhance the process.

JAD strongly complements, but does not replace, analytical methodologies.

What's the big deal?

From the Standish Group...

- In the United States, we spend more than $250 billion each year on IT application development of approximately 175,000 projects.
- The average cost of a development project for a large company is $2,322,000; for a medium company, it is $1,331,000; and for a small company, it is $434,000.

http://www.standishgroup.com/chaos.html

From the Standish Group...

- 31.1% of projects will be canceled before they ever get completed.
- 52.7% of projects will cost 189% of their original estimates.
- In the larger companies, only 9% of projects come in on-time and on-budget.
- Projects completed by the largest American companies have only approximately 42% of the originally-proposed features and functions.

SUCCESS CRITERIA	POINTS
1. User Involvement	19
2. Executive Management Support	16
3. Clear Statement of Requirements	15
4. Proper Planning	11
5. Realistic Expectations	10
6. Smaller Project Milestones	9
7. Competent Staff	8
8. Ownership	6
9. Clear Vision & Objectives	3
10. Hard-Working, Focused Staff	3

TOTAL 100

"We know why projects fail, we know how to prevent their failure -- so why do they still fail?"

Martin Cobb
Treasury Board of Canada Secretariat
Ottawa, Canada

What can we do about all this?

They go hand in hand...

1. Adopt the Software Engineering Institute's Capability Maturity Model (SEI CMM) to improve the organization's processes!

2. Benchmark internally and externally

3. Improve people skills in Process and Project Management!

4. Use Joint Application Development!

CIO Priorities
Top 10 Management Issues
(1999-2003)

1. Business/IT Fusion
2. Demonstrating Business Value
3. IT Skills (Recruit, Retain, Reskill)
4. Y2K Clean-up/Contingencies
5. "Sourcing" Management
6. IT Governance
7. **Process/Project Management**
8. M&A IT Integration
9. Knowledge Management
10. IT Organization

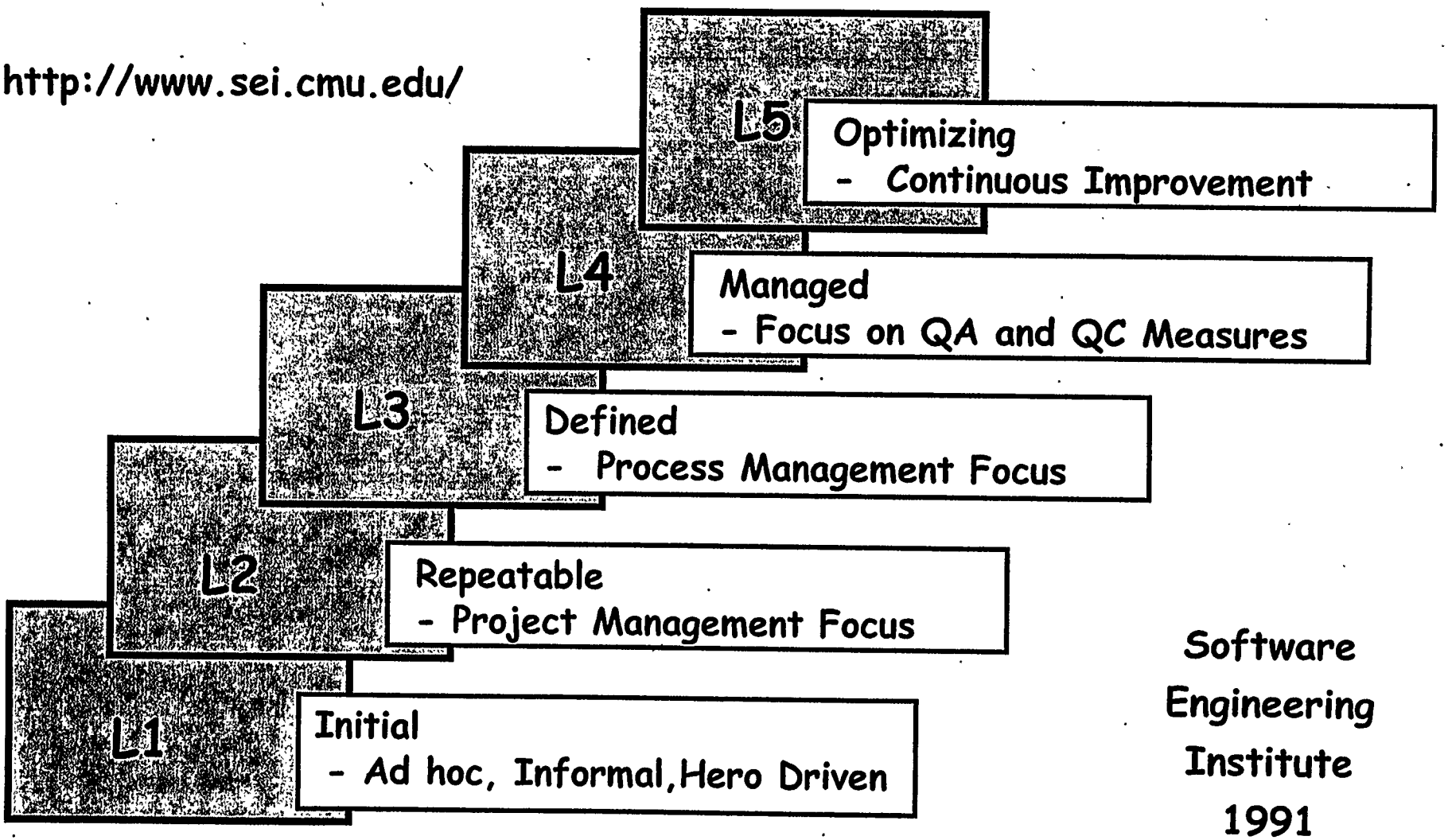
Gartner Group - November 16, 1998

Why the SEI CMM?

Software Engineering Institute's Capability Maturity Model

CAPABILITY MATURITY MODEL

http://www.sei.cmu.edu/



Software Engineering Institute 1991

OBSERVED SEI CMM BENEFITS

Maturity Level	Calendar Months	Person Months	Defects Found	Defects Shipped	Total Cost
1	29.8	593.5	1348	61	$ 5,440,000
2	18.5	143	328	12	$1,311,000
3	15.2	79.5	182	7	$728,000
4	12.5	42.8	97	5	$392,000
5	9.0	16	37	1	$146,000

Estimated impact for 200,000 LOC Software Project
(from Sematech publication 'Communique")

Application Development Practices

"Software Capability Evaluations (based on SEI's Capability Maturity Model) will be used to qualify all IT contractors on U.S. Federal Government projects by year end 2001 (0.8 Probability); however, widespread year 2000 damage will push regulated AD into the private sector, starting with applications that have public health and safety implications."

Gartner Group - November 16, 1998

Benchmark internally and externally

- Establish a Project Office to avoid "industrial tourism"
- Share Best Practices within the multitude of I.T. groups in a single company
- Identify and reward the most effective set of Best Practices
- Benchmark with external organizations
- Improve the I.T. processes with the SEI CMM

Project Office Models

- **Repository Model**
  - source of information on project methodology and standards
- **Repository-Coach Model**
  - coordinates sharing of best practices across business functions
- **Repository-Coach-Manager Model**
  - direct management of projects and project managers

People Skills:

What about Certification?

Some of the available programs...

1. The Quality Assurance Institute

   - Certified I.T. Quality Analyst (CQA)

   - Certified Software Test Engineer (CSTE)

2. The Project Management Institute

   - Certified Project Management Professional (PMP)

Can create the necessary awareness, at all levels, for improving Process and Project Management practices

Is Project Management Certification Worth It?

- By 2004, 70 percent of successful projects will have certified project managers, while 90 percent of failed projects will not (0.7 probability).
- By 2002, employees with Project Management Professional (PMP) certification will receive salaries 20 percent higher than those of project managers without PMP certification (0.7 probability).
- By 2002, leading-edge project management outsource vendors will have at least one out of five contracted staff team members who are PMP-certified (0.7 probability).

Gartner Group - November 16, 1999

Benefits of Project Management Certification

- Consistent Project Management approach within the organization and improved interface mechanisms with all customers
- Certification requires continuing education in the Project Management field, which leads to more effective Project Managers over time
- Project Management Certification thoroughly supports the findings and recommendations of most benchmarking efforts
- Combination of Project and Process Management has the potential of reducing project development costs by 30% a year (Gartner Group)

The overall objective

- Improve I.T. Process and Project Management.
- Generate in people a desire to improve their environment.
- Get the job done faster and cheaper!

Get the job done faster and cheaper!

Since 1980, facilitation has been used for every aspect of business and community development, from strategic planning to detailed process design.

Why JAD works!

- Communication tools
- Structured techniques
- Defined roles
- Structured agendas
- An effective process
- Focus on people, not technology

Communication tools

- Group Dynamics
- Facilitation
- Visual Aids
- Concentrated Exchanges
- Consensus-based Decisions

Structured Techniques

- Brainstorming
- Basic Iteration
- Data Flow Diagramming
- Data and Object Modeling
- Process Design
- Planning Models

Defined Roles

- Session Leader
- Scribe
- Executive Sponsor
- Project Manager
- Participants
- Observers

Structured Agendas

The Standard JAD Agenda

- Introduction
- Review Purpose, Scope, Objectives
- Ground Rules
- Executive Sponsor Kickoff
- I.T. Perspective
- [Session Approach]
- Review Issues
- Review Design
- Evaluate Workshop

Structured Agendas

A few JAD Session Approaches

- Strategic Business Planning Approach
- Business Reengineering Approach
- Business Systems Planning Approach
- Project Scope Definition
- Project Requirements Definition
- Information Modeling
- Project Management
- Acceptance Testing
- Walkthrus/Peer Reviews

Structured Agendas

Strategic Business Planning

- Define Vision and Mission of activity
- Define Guiding Principles
- Analyze current situation
- Analyze competitive situation
- Define Objectives and Goals
- Develop Business Strategies
- Define Critical Success Factors
- Design Programs and Resources
- Describe support Organization
- Define follow-on activities

Structured Agendas

Business Reengineering Approach

- **Phase I - Vision**
  - Customer requirements
  - Management Vision
- **Phase II - "What" Phase**
  - Functional Model
  - Information Model
- **Phase III - "How" Phase**
  - Conceptual Design
  - Organization Design
  - Detailed Design

Structured Agendas

Business Systems Planning

- Review Vision and Mission of the activity
- Review business strategies and CSFs
- For each business strategy:
  - Define new automation requirements
    - Entities and Processes
  - Describe constraints to the new requirements
    - Schedules, cost, resources
- Prioritize the new requirements

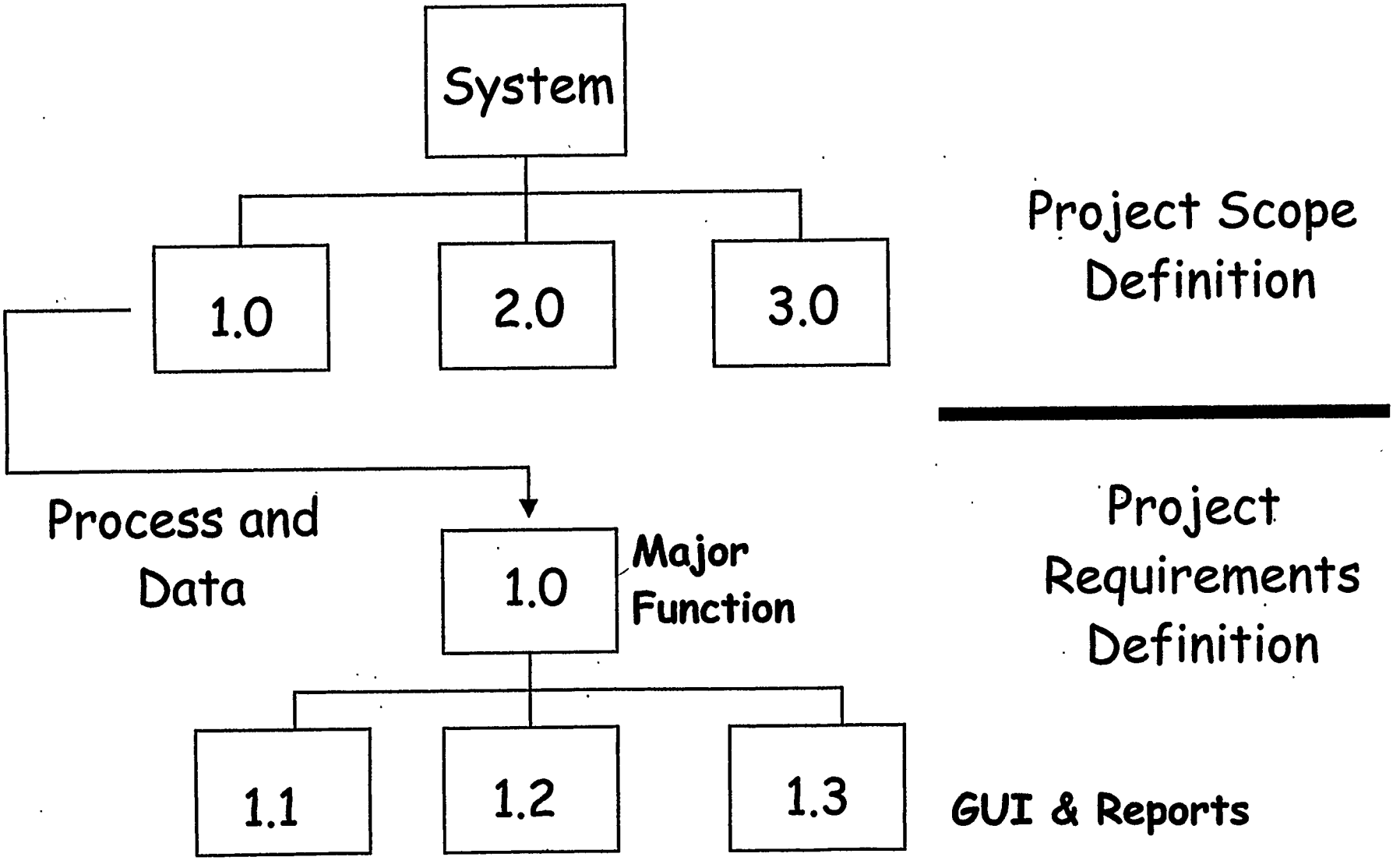
Structured Agendas

Project Scope Definition

- Define System Objectives
- Define Major Outputs
- Define Major Inputs
- Identify System Events
  - Context Diagram
- Information Analysis
  - Entity Relationship Model
- Major System Functions
  - Essential or Zero-level Diagram
- Confirm System Objectives

Structured Agendas

Project Scope & Requirements Definition



Structured Agendas

Acceptance Testing

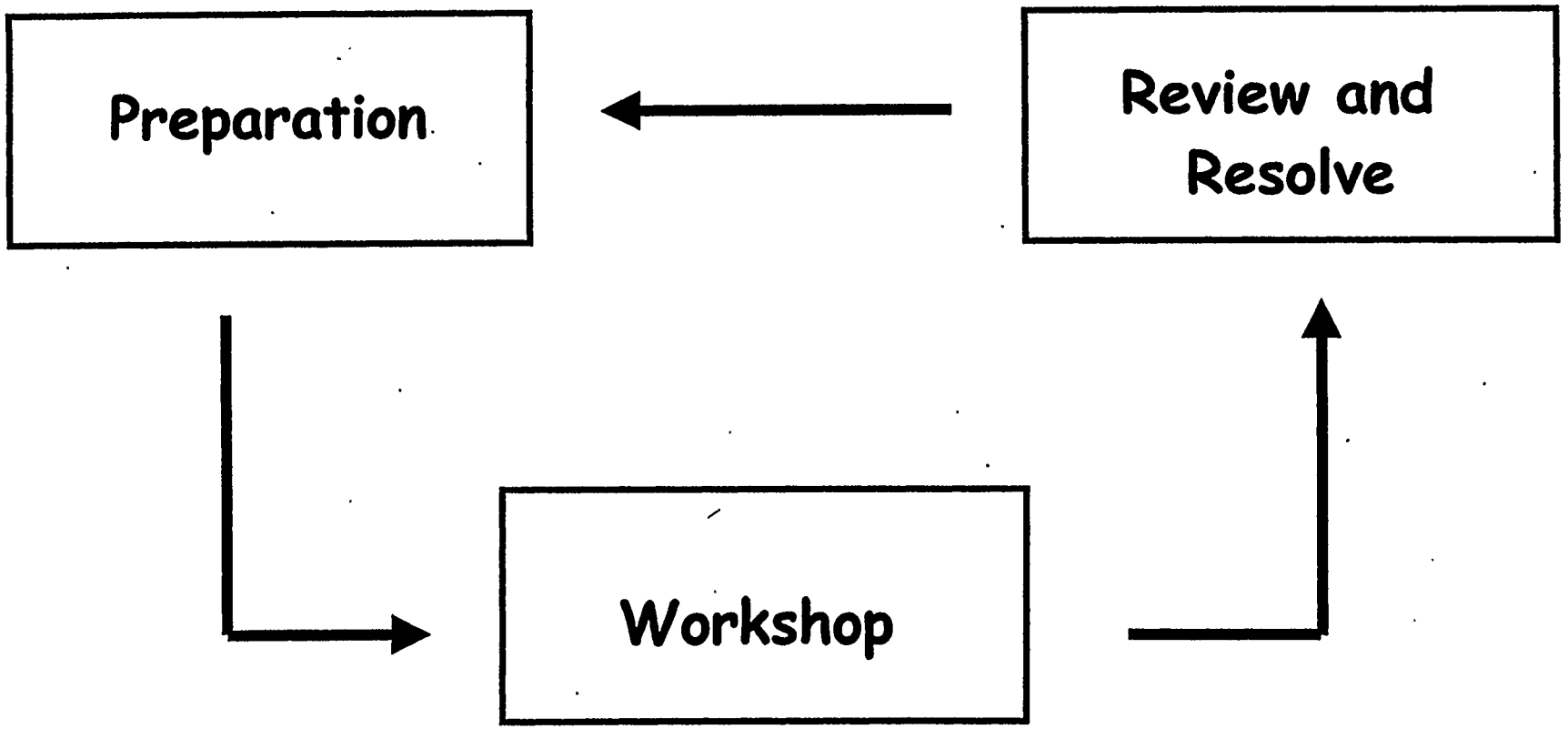
- Establish Baseline
  - Review existing process and procedures
  - Define testing requirements
  - Apply lessons learned
- Testing Process
  - Goals and objectives
  - Review/define testing levels
    - unit, systems, acceptance criteria
  - Review/define testing workflow and guidelines
- Develop Test Plan
  - who, what, when
- Define Roles and Responsibilities
  - Develop RASI chart
- Test Environment
  - Review physical environment

Structured Agendas

Walkthrus/Peer Reviews

- Objectives and Requirements
  - establish baseline for evaluation of system components
- Step through the System
  - For each system function/sets of requirements:
    - verify data requirements
    - verify functional requirements
- Next Steps
  - document results/action items

An effective process



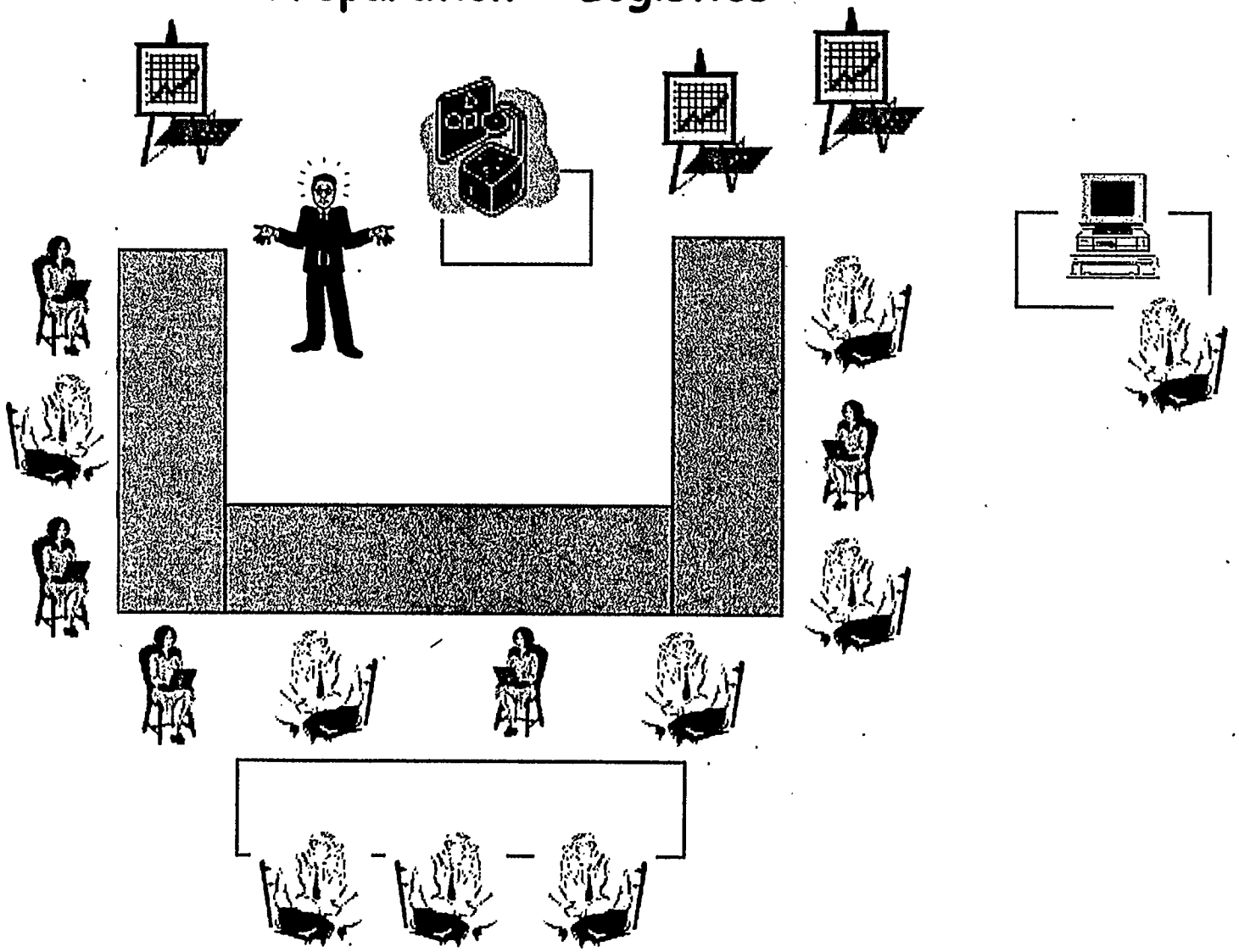
An effective process

Preparation

- Scope Management
- Interview
- Commitment
- Risk Assessment
- JAD Sessions Plan
- Prepare Users
- Logistics

An effective process

Preparation - Logistics



An effective process

Workshop

- Unique environment
- Combines skills of Session Leader and Participants
- Produces complete information

An effective process

Review and Resolve

- Review session documentation
- Resolve issues
- Post-process evaluations
- Follow-up surveys

Focus on people, not technology

- Participants have a stake in the development project or effort at hand.

- JAD brings people together in an effort to eliminate barriers and make the participants work as a team.

Requirements for Success

- Commitment from customer and I.T. management
- Well-trained facilitator with technical skills and lots of facilitation experience
- Availability and commitment of proper resources
- Proper application of the concepts and structure of the process

So, why do we need JAD?

JAD is a breakthrough technique essential for:

- Improving communication throughout the company
- Improving the image of Information Technology
- Integrating the customers in automation decisions
- Disseminating the use of enhanced process and project management approaches

Improving communication throughout the company

- People talking to other people!
- No misunderstood user requirements
- Executives welcome an effective and quick solution to Strategic Planning (as opposed to lengthy studies)
- We still need the business strategies to model the information needs and actually achieve business alignment

Improving the image of Information Technology

Using JAD for strategic business planning and business reengineering is viewed by customers as an endeavor which goes beyond the perceived mission of I.T.

Improving the image of Information Technology

Gaining credibility and visibility

- JAD session leaders are viewed as impartial solution providers
- Business customers and I.T. professionals will typically accept the recommendations of the JAD session leaders
- "For each additional foot of credibility, introduce another inch of more disciplined practices"

Integrating the customers in automation decisions

Customers are asked to make their own decisions about information system strategies and application development issues. I.T. is viewed as a problem solver.

Disseminating the use of enhanced process and project management approaches

JAD session leaders:

- understand process and project management
- are trained in group dynamics, structured analysis and design, data modeling
- can become effective facilitators for strategic planning and business reengineering efforts
- should be primary candidates in Project Office staffing endeavors

A final word on real benefits

- On a major tax project, JAD sessions generated six months worth of project documentation in 14.5 days
- Major refinery identified and prioritized 24 new information systems in 2 days
- JAD brought together participants from several locations to develop specifications for a project that had been on the drawing board for 10+ years
- Alignment of business strategies and I.T. strategies identified need for a major reporting project tied to a new business initiative about to be launched

Software Quality Forum 2000

Front-End Loading with Facilitated Joint Application Development Techniques

Santa Fe, NM - April 18-19, 2000

Speaker Introduction

- Mike Williamson, Compaq Federal, LLC

What Is A Product?

As it relates to our environment:

- Software
- Analysis
- Knowledge based conclusions (professional judgment)

The Common Refrain

In a the LAB and university environment.

SW is not my product, analysis is my product.

Lets Examine the Implications of This Statement

What Is Modeling

- Definition
- As relates to physical systems
- Star movement example

What Makes Modeling SW Different in Terms of V&V?

- Your doing modeling because the systems are so complex that they defy solution.
- That very complexity is the issue, how do you validate it.
- You compare predicted results with observed results.
- But for WIPP (or stockpile stewardship) how do you validate?

What is V&V in a modeling context

Lets Compare /Contrast V&V in the SW Lifecycle and for Modeling Codes

- Conceptual model
- Mathematical model
- Numerical model
- Computer model
- Code
- If we take the results of the code and it corresponds (roughly) with the expected results we have validated, in some sense the conceptual model

Validation and Verification in the modeling domain

Lists of attributes

- SW as a tool – short list
- SW as a product (deliverable) – longer list

Tools Must be maintained

New requirements, new platforms, (technology)

What Facilitates Maintainability

- Design architecture
- Extensibility
- Reuse
- Documentation
- Requirements management
- CM

If It Looks Like a Duck

These attributes, when added to the attributes of a modeling code begin to look very much like a product.

Aspects of ASCI Code Verification and Validation

Timothy G. Trucano

Optimization and Uncertainty Estimation Department
Sandia National Laboratories
Albuquerque, NM 87185

Phone: 844-8812, FAX: 844-0918
Email: tgtruca@sandia.gov

http://sherpa.sandia.gov/9231home/compphys-frame.html

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.



Sandia National Laboratories

Outline of presentation:

1. General purpose scientific codes in the Accelerated Strategic Computing Initiative (ASCI) mold

2. Strategic Computing & Simulation Verification & Validation Program

3. Aspects of the ASCI V&V Program

4. Aspects of the Sandia ASCI V&V Program

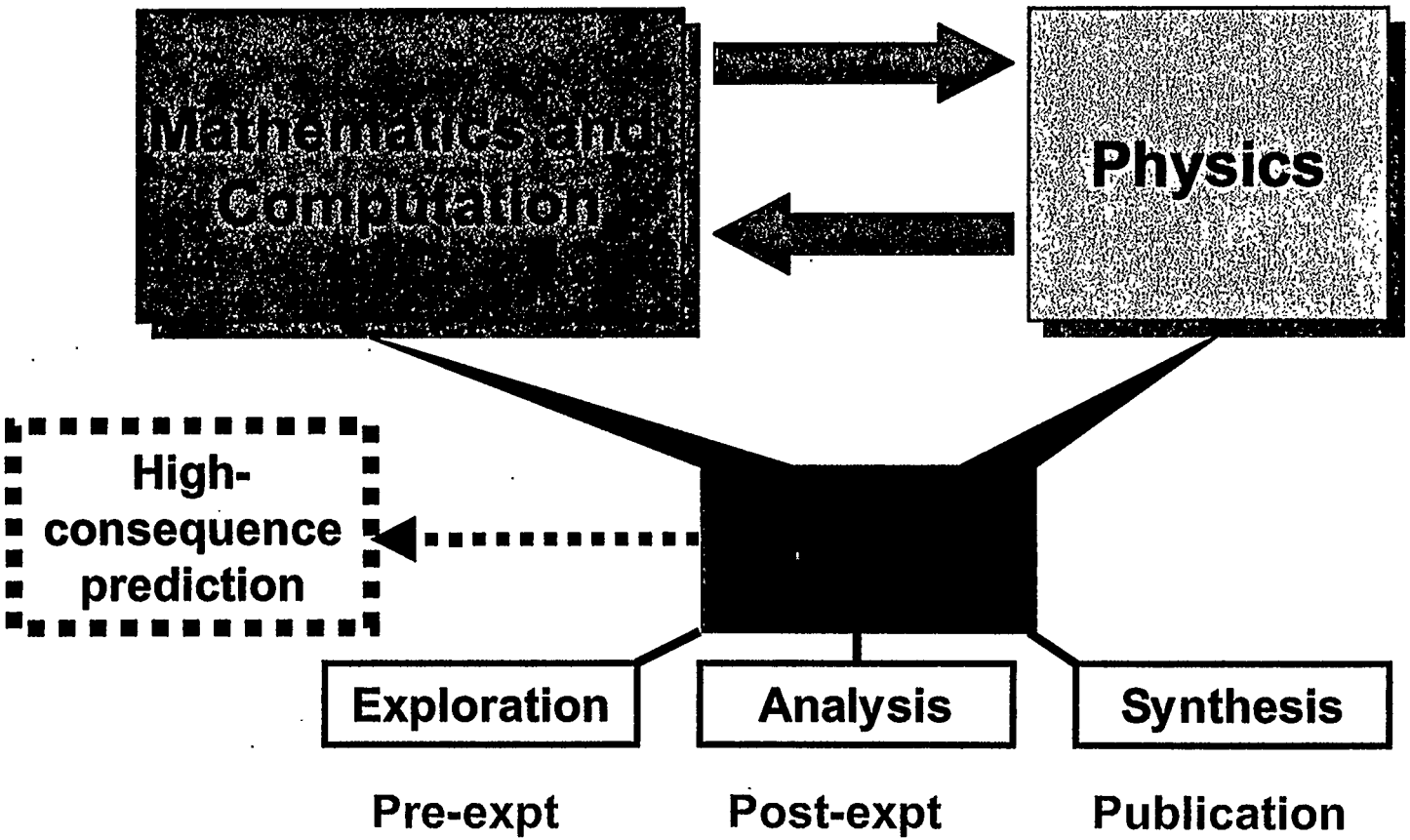
5. Future Challenges

Sandia National Laboratories

Discussion of ASCI-type codes:

Sandia National Laboratories

The simplest picture of scientific computation does not include high consequence software.

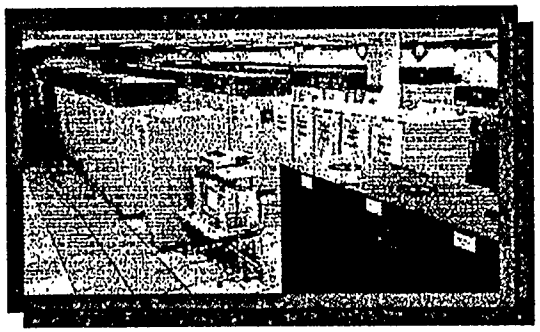


Sandia National Laboratories

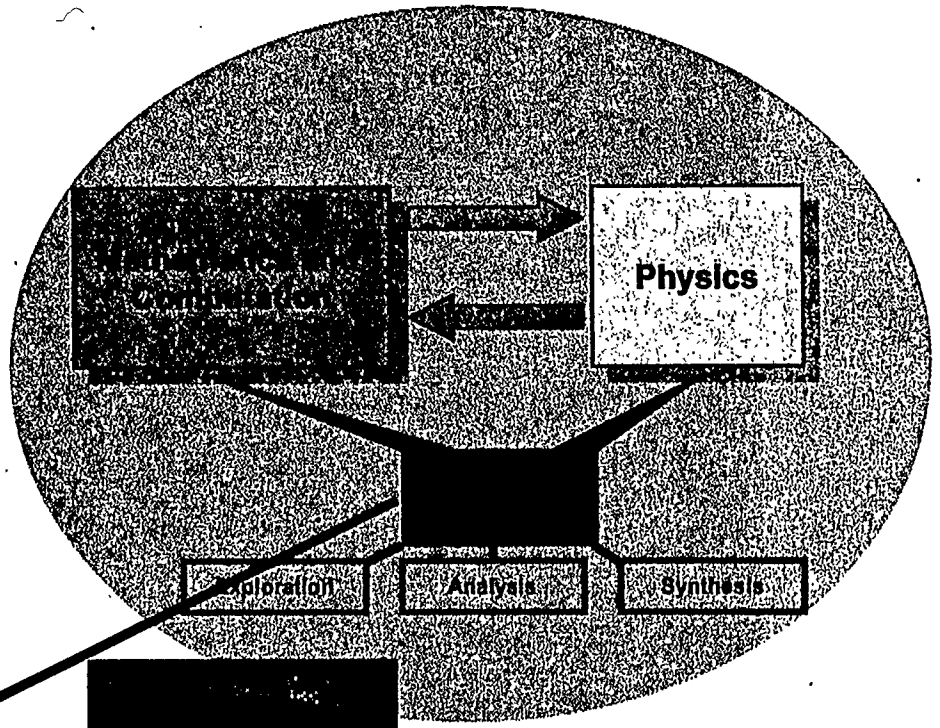
Science Based Stockpile Stewardship (SBSS) is changing this situation as we speak.

"ASCI will create the leading-edge modeling and simulation capabilities that are essential for maintaining the safety, reliability, and performance of the U. S. nuclear stockpile..."

ASCI Red

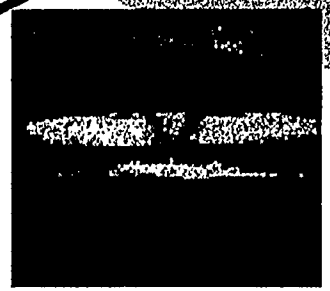


+



=

High Consequence Stockpile Predictions



French Nuclear Test

Sandia National Laboratories

In ASCI, we are concerned with general purpose multi-physics scientific software.

The codes are big. (ALEGRA ~ 1 Mline, ~22 developers, mainly C++)

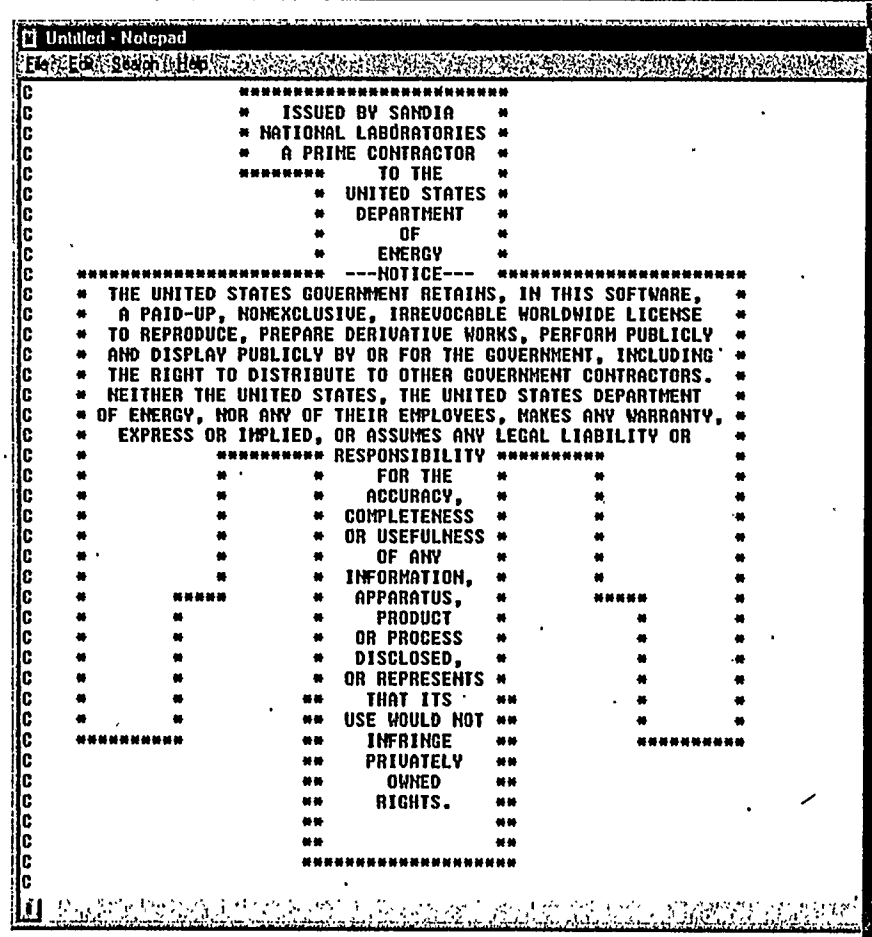
- **Lots of physics**
- **Lots of length and time scales**
- **Strong couplings in the multi-physics**
- **Lots of floating point arithmetic on complex hardware**
- **Effectively infinite number of important usages**

"Federation" of software (preprocessors, postprocessors, and more) is prevalent.

Main users are not the software developers, and there are likely going to be a lot of them! This is a critical metric of success.

Sandia National Laboratories

Are we ready for high consequence scientific computing? Consider this disclaimer in CTH.



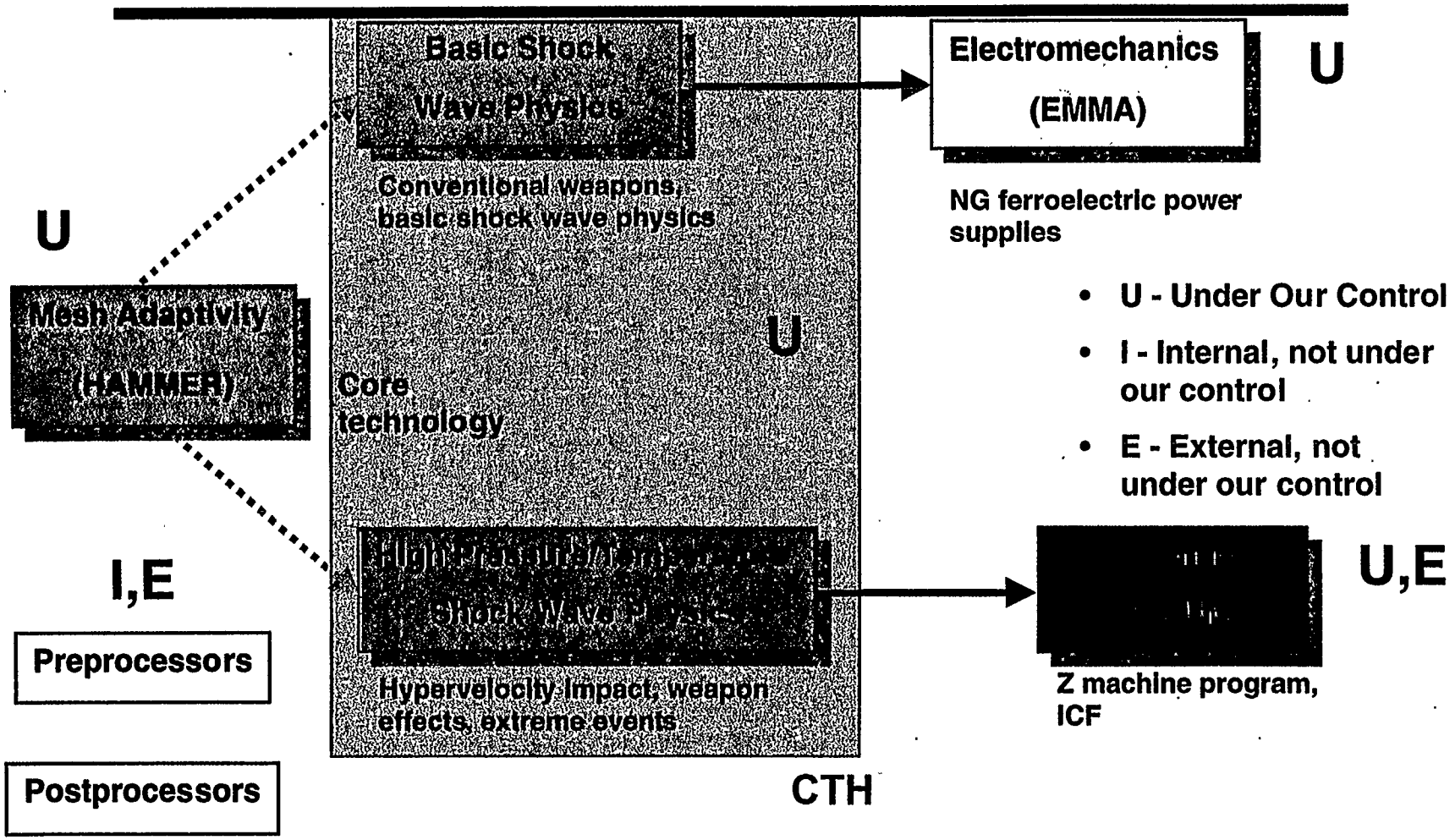
ISSUED BY SANDIA NATIONAL LABORATORIES, A PRIME CONTRACTOR TO THE UNITED STATES DEPARTMENT OF ENERGY

---NOTICE---

THE UNITED STATES GOVERNMENT RETAINS, IN THIS SOFTWARE, A PAID-UP, NONEXCLUSIVE, IRREVOCABLE WORLDWIDE LICENSE TO REPRODUCE, PREPARE DERIVATIVE WORKS, PERFORM PUBLICLY AND DISPLAY PUBLICLY BY OR FOR THE GOVERNMENT, INCLUDING THE RIGHT TO DISTRIBUTE TO OTHER GOVERNMENT CONTRACTORS. NEITHER THE UNITED STATES, THE UNITED STATES DEPARTMENT OF ENERGY, NOR ANY OF THEIR EMPLOYEES, MAKES ANY WARRANTY, EXPRESS OR IMPLIED, OR ASSUMES ANY LEGAL LIABILITY OR RESPONSIBILIT FOR THE ACCURACY, COMPLETENESS OR USEFULNESS OF ANY INFORMATION, APPARATUS, PRODUCT OR PROCESS DISCLOSED, OR REPRESENTS THAT USE WOULD NOT INFRINGE PRIVATELY OWNED RIGHTS.

Apparently Not!

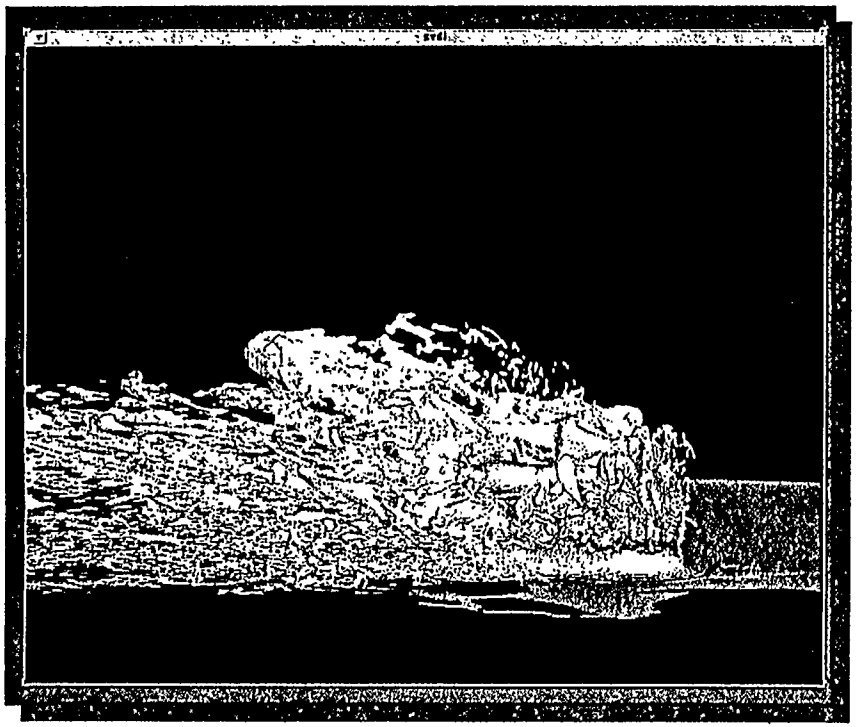
The code I am associated with (ALEGRA) is an example of such a piece of software.



Sandia National Laboratories

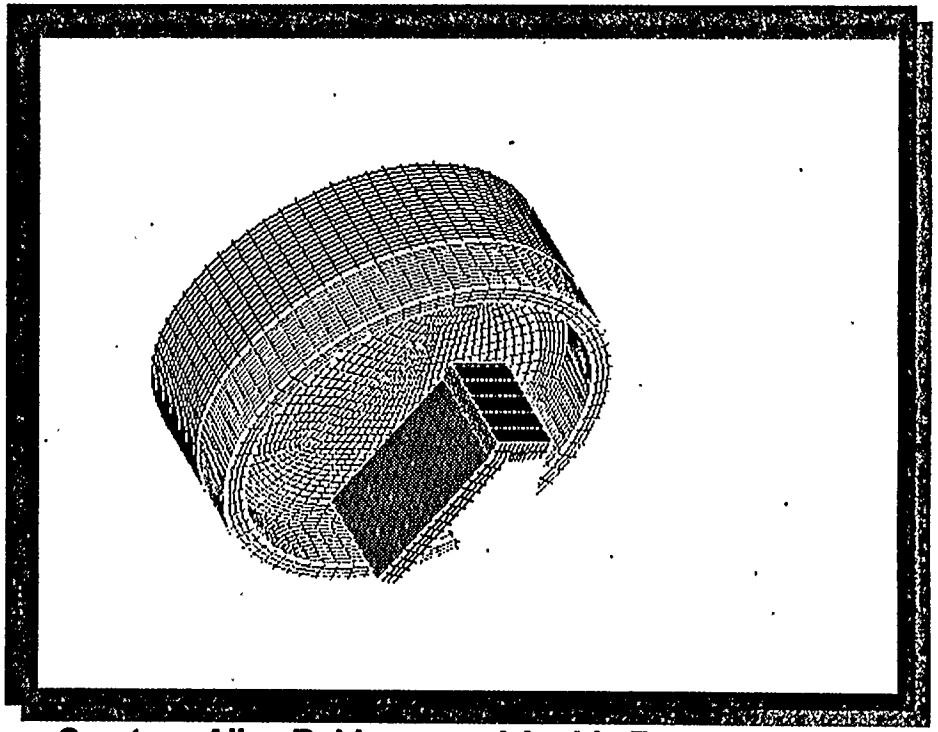
Illustrations of CTH and ALEGRA calculations

CTH calculation of a comet impact burying Long Island - does anybody care?



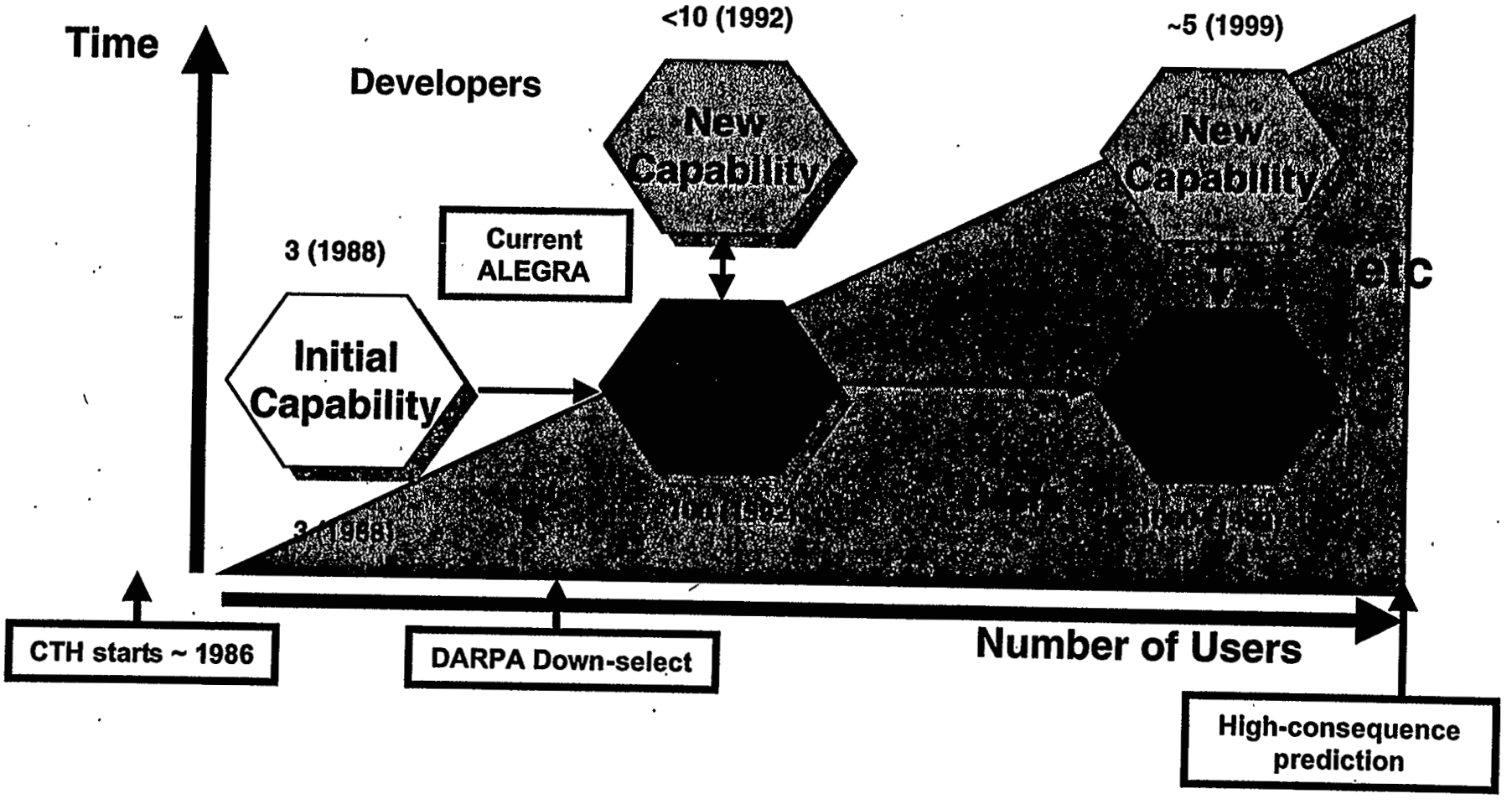
Courtesy David Crawford

Part of an ALEGRA/EMMA neutron generator power supply calculation mesh - there are those who care.



Courtesy Allen Robinson and Archie Farnsworth

Here is my opinion - the lifecycle of a code like this has been *peculiar* up to the era of SBSS. [CTH]



Sandia National Laboratories

We apparently *CAN* do high consequence modeling with CTH:

"Confidence in the [code] modeling approach has reached the point that during the recent Advanced Unitary Penetrator (AUP) Warhead Program at China Lake, the Ordnance Hazard Evaluation Board (OHEB) ruled that actual SD [*sympathetic detonation*] testing was not required and the *results of the SD simulations were adequate for limited fleet introduction of these warheads*. The SD simulations were completed on a part time basis in less than *four weeks* compared to the actual test schedule, which would have been in excess of *one year*." [Emphasis mine]

Sandia National Laboratories

The ASCI V&V program - generic information:

Sandia National Laboratories

What is the "product" lifecycle, anyway?

In addition to vast differences in modeling needs and implementation approaches in ASCI, there are also alternative views of the software lifecycle:

- "We are building codes for 30 years." [TGT]
- "Surely you are joking. We are building codes for five years, at which time we will completely rewrite." [A common historical practice.]
    - Hardware is changing too rapidly
    - Physics is changing too rapidly
    - People are leaving too rapidly

Quality starts with the definition of what you are:

- Is this software science?
- Or is it product?

Sandia National Laboratories

An important task is:

To develop higher confidence in high consequence predictions on shorter code development time scales.

Money is not the main issue ...

We believe that part of the solution is a formal verification and validation program.

The ASCI Verification & Validation Program started in FY99.

Verification - "The *process* of determining that a computer simulation correctly represents the conceptual model and its solution." [Equations solved correctly?]

Validation - "The *process* of determining the degree to which a computer simulation is an accurate representation of the real world." [Solving the correct equations?]

Vision - "Establish *confidence* in the simulations supporting the Stockpile Stewardship Program through systematic demonstration and documentation of the predictive capability of the codes and their underlying models."

There is definitely a process component.

The ASCI V&V program draws on an increasing body of work in formal V&V for scientific software.

The program fits within a continuum of formal work on V&V in scientific computing over the past 20 years.

It specifically draws upon some contact with work by:

- **Defense Modeling and Simulation Organization (1994)**
- **AIAA (1994, 1998)**

Note that the greater scientific community is beginning to apply more formal standards to reported results of scientific calculation:

- **ASME - constraints on journal publication standards**
- **AIAA - promulgated V&V draft standards and constraints on journal publication**
- **Achieving these standards requires formal effort in V&V, hence increase interest in professional organizations**

Further information:

Pat Roache has written the only scientifically oriented book on "code" V&V: P. J. Roache Verification and Validation in Computational Science and Engineering , Hermosa, 1998

Contrast this (and ASCI V&V, for that matter) with the interesting book: P. L. Knepell and D. C. Arangno, Simulation Validation - A Confidence Assessment Methodology, IEEE Computer Society, 1993 [Correct implementation of requirements? Correct requirements?]

Here is a smattering of bibliographies:

W. O. Oberkampf (1998), SAND98-2041, several hundred

O. Balci and R. G. Sargent (1984), Simuletter, Vol. 15, No. 3, several hundred

DMSO, date unknown, bibliography available at the DMSO Web Site

D. S. (Steve) Stevenson (1998), unpublished, ~ 150 references

Sandia National Laboratories

The ASCI V&V program - beyond testing in code verification...

Sandia National Laboratories

The focus of the ASCI V&V program, as defined in the draft plan, happens to be on validation.

Verification is viewed as strongly centered within the code development projects (ASCI Application Program).

Nonetheless, verification is a part of the program (see the March 1998 program plan draft). The issues and opportunities we see are discussed below .

Starting in FY99, under verification the program has:

- Investigated the role of SQA
- Begun sharing information and recommendations about tools and methods for SE and verification methodologies
- Begun prototyping formal V&V plans for selected code projects (at Sandia)

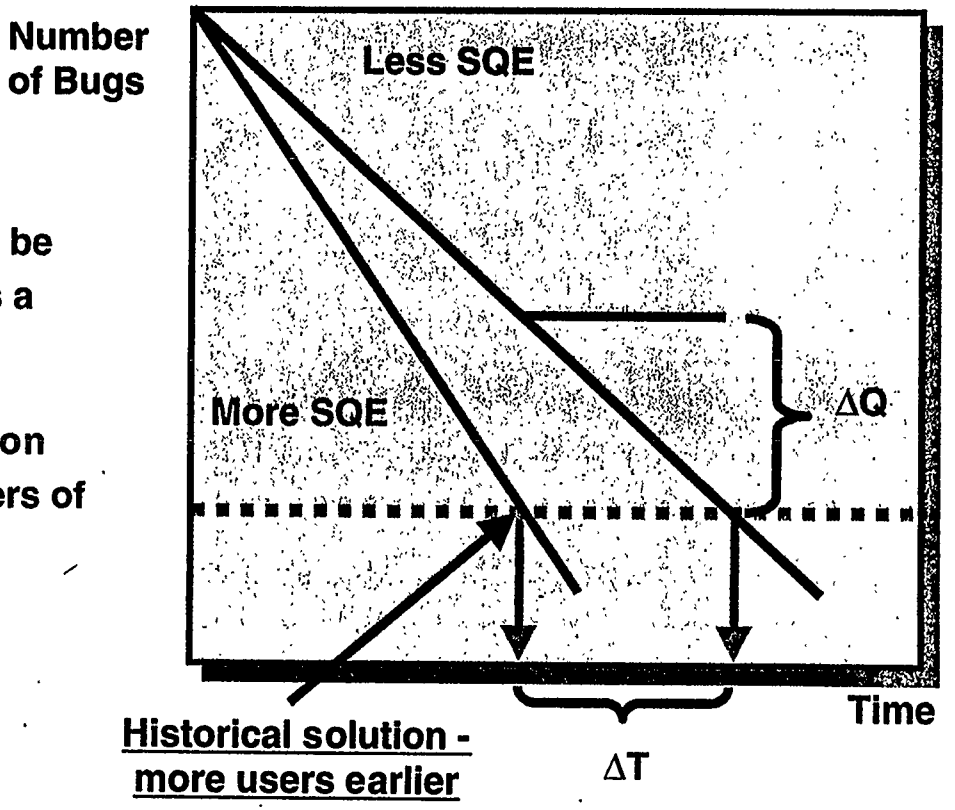
In FY00, focus on process has intensified at Sandia.

Verification should lead to the increase of code quality as a function of time, or cost, or effort. This should increase our confidence.

Software "quality" is a multidimensional entity which evolves with multiple variables.

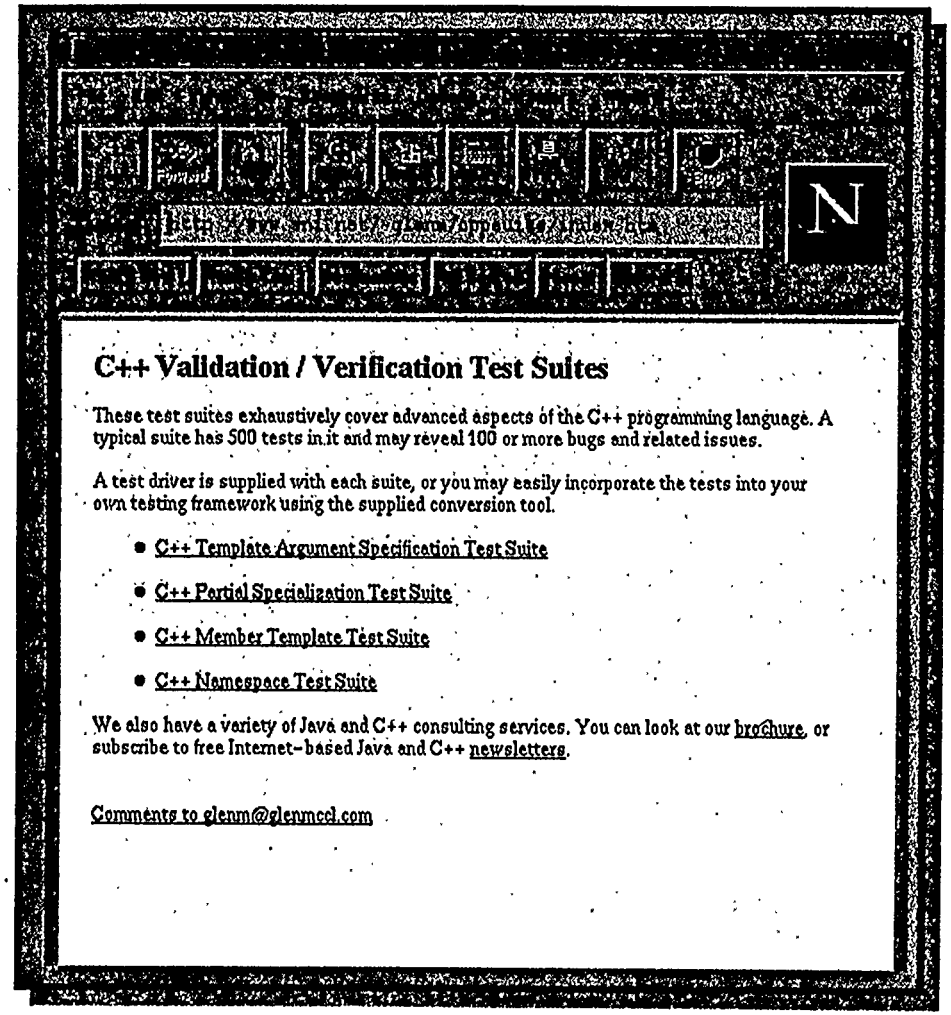
For example, software quality might be measured by the number of bugs as a function of the life cycle time.

The payoff for more formal verification processes would be reduced numbers of bugs at a given time, or less time to achieve a given bug status, or ...



Sandia National Laboratories

Verification = Testing? If only it were true ...



C++ Validation / Verification Test Suites

These test suites exhaustively cover advanced aspects of the C++ programming language. A typical suite has 500 tests in it and may reveal 100 or more bugs and related issues.

A test driver is supplied with each suite, or you may easily incorporate the tests into your own testing framework using the supplied conversion tool.

- C++ Template Argument Specification Test Suite
- C++ Partial Specialization Test Suite
- C++ Member Template Test Suite
- C++ Namespace Test Suite

We also have a variety of Java and C++ consulting services. You can look at our brochure, or subscribe to free Internet-based Java and C++ newsletters.

Comments to glenm@glenmccl.com

Verification logically fits this model relatively well, except...

this still doesn't "prove" that the code implementation is "correct."

Sandia National Laboratories

The historical approaches to testing algorithms are performed in ASCI applications code verification programs.

Test problem classification:

- "Consistency Checks" (spherical blast waves)
- Analytic test problems (Whalen test suite)
- Very precise numerical solutions (Sedov blast waves)
- Code comparisons (ALEGRA vs CTH, etc)

- Mesh generation
- Initial conditions
- Boundary conditions
- Physics algorithm design
- Algorithm accuracy, convergence, stability, consistency
- Algorithm coupling
- Material models (EOS, opacity, strength, fracture)
- Post-processing

Sandia National Laboratories

The "God" of testing speaks:

B. Beizer, Software Testing Techniques, International Thomson Computer Press, 1990 [Blackbox?]

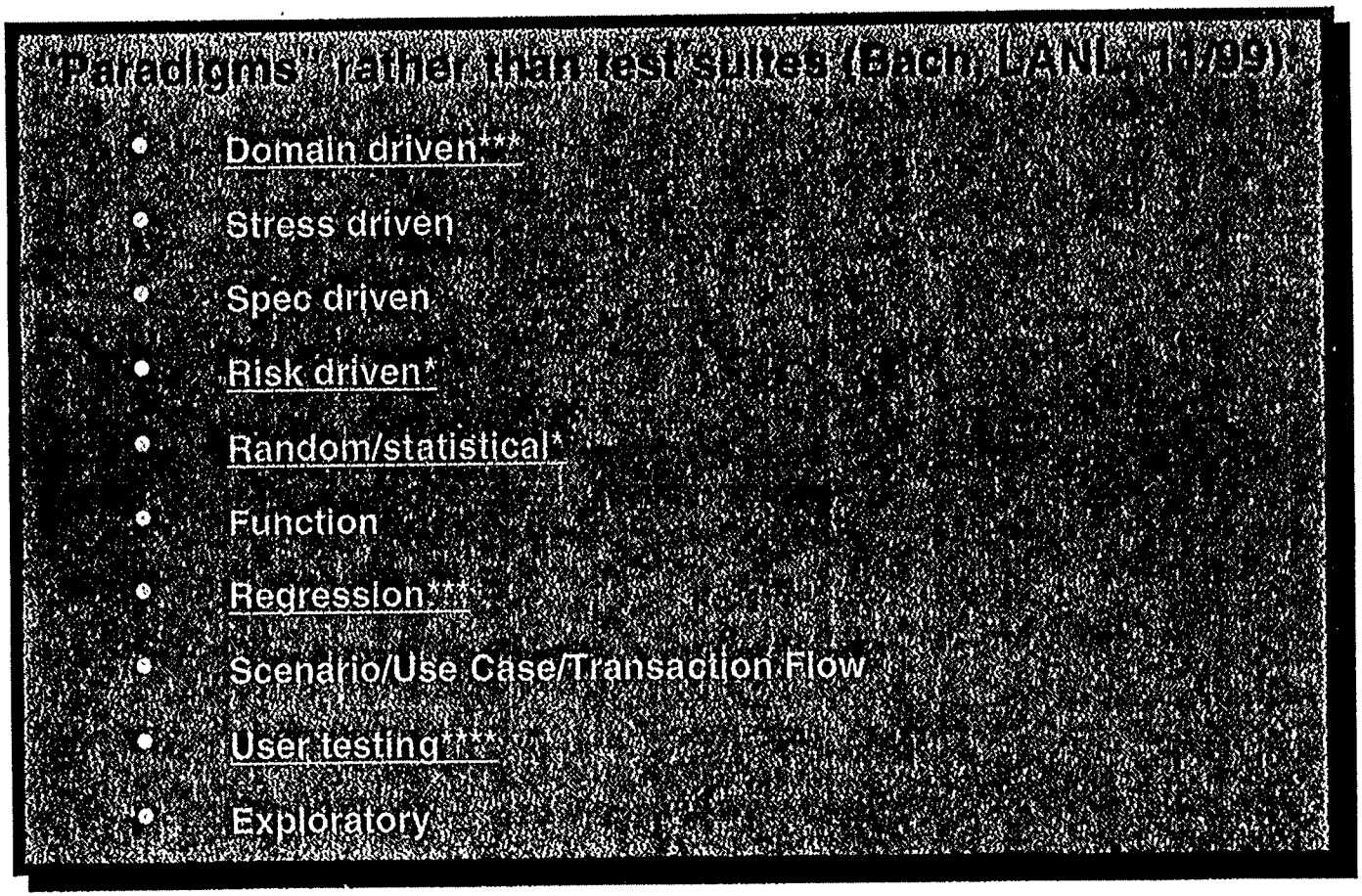
- Flowgraphs and path testing
- Transaction-flow testing
- Data-flow testing
- Domain testing
- Syntax testing
- Logic-based testing
- Transition testing

- Spelling
- Grammar
- Punctuation
- Paragraph structure
- Syntax
- Logic

At the end of all of this you still don't have a novel.

There are also alternative views within the software engineering community:

Kaner, Flak, and Nguyen (1999), Testing Computer Software, John Wiley ["The bestselling software testing book of all time!"]



Sandia National Laboratories

There is a vast amount of information out there suggesting that this is still a research topic.

- For example, consider some of the documents at the NIST Computer Systems Laboratory WEB site
  - W. Peng and D. Wallace (1993), "Software Error Analysis" ["...The study includes statistical process control techniques and relates them to their use as a software quality assurance technique for both product and process improvement..."]
  - M. Zelkowitz and D. Wallace (1996), "Experimental Models for Software Diagnosis;" "Experimental Validation in Software Engineering" ["...there is wide disagreement as to exactly what the term 'experimentation' means in this domain."]
  - J. Poole (unknown), "A Method to Determine a Basis Set of Paths to Perform Program Testing" ["A major problem in unit testing of programs is to determine which test cases to use."]

Verification for a multi-physics code is more than code testing.

What tests?

How many of them?

What is the benefit?

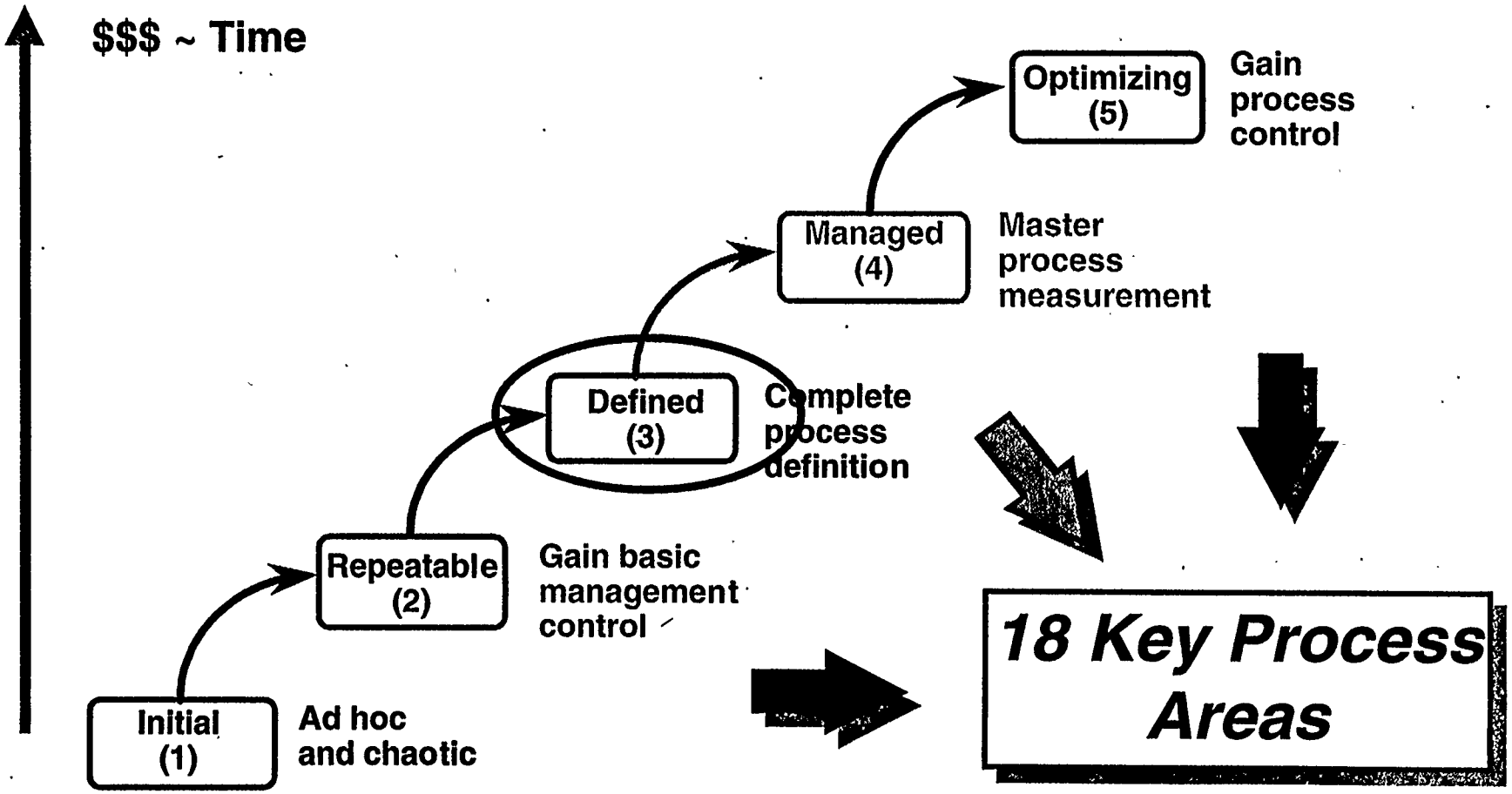
What is the cost?

What are they telling you?

What are they not telling you?

What are the metrics?

SEI CMM Level 2 is at least a useful information focusing methodology.



Sandia National Laboratories

We estimate ASCI code projects are concentrated between Levels 2 and 3.

Transition to Level 2 requires:

- Configuration management s
- SQA T
- Sub-contract management (suppliers) T
- Project tracking and oversight s
- Project planning s
- Requirements management T

Transition to Level 3 requires:

- Peer reviews s
- Intergroup coordination ?
- Product engineering T
- Integrated software management T
- Training program s
- Organization process definition ?
- Organization process focus ?

s - doing it; T - worrying about it; ? - don't know

[A specialized survey has refined this picture in FY99.]

Sandia National Laboratories

Verification in ASCI - some SQE

Sandia National Laboratories

Delivering computational science as product requires software engineering (SE).

What software engineering principles and techniques are applicable to large systems of floating point multi-physics code?

Many recommendations are called but fewer are chosen:

- **Configuration management systems and rules**
- **Program development environments (is the ideal to have no human write any code, but only specs, equations, etc?)**
- **Coding standards**
- **Formal software inspections**
- **Regression testing**
- **Documentation - plans, requirements, theory, implementation, algorithms, process**

The Crux!

> **"...the author cannot find any long term empirical study on the efficacy of the various software engineering techniques...", D. E. Stevenson (1991), "1001 Reasons for not Proving Programs Correct: A Survey," Clemson preprint.**
>
> **Still true as of 1999. Recent comments from NIS study group are similar**

Sandia National Laboratories

Example: Testing Starts with Configuration Management Systems

"CVS" + scripts is probably the widest used systems across ASCI applications code development projects at LLNL, LANL, and Sandia

There is considerable interest in RAZOR at LANL and most or all of their code projects are moving to it

- RAZOR was discarded by the "Codes for the Complex" software engineering support group at Sandia - they are going to use "Dimensions" instead.
- Go figure...

We currently have a selection process for "quality tools" at Sandia - RAZOR is not likely to be chosen (ClearCase has been a significant other); but ...

ALEGRA, CTH - We use CVS plus lightweight scripts; we also have a configuration management "ethic." CTH is supported at over 130 sites external to Sandia as an Export Controlled software system.

- ALEGRA uses the bug tracking tool ClearDDTs (but not completely)

Will it ever be possible to even have configuration management systems that are uniform across a single lab's ASCI program, let alone all three labs?

Sandia National Laboratories

Don't forget static testing - Consider "The T Experiments"

L. Hatton (1997), "The T Experiments: Errors in Scientific Software," Computational Science and Engineering, Vol. 4, No. 2, 1997, 27-38

"The results were horrifying." - Roache

Here are the lessons to me:

- **Well designed *static testing* is critical.**
- **You need *tools* to do this.**
- **You need *metrics* to do this.**
- **The proper language to understand what Hatton is saying is *probabilistic reliability*, not physics.**
  - **Fully discussing the reliability paradigm is beyond the scope of this presentation.**

Sandia National Laboratories

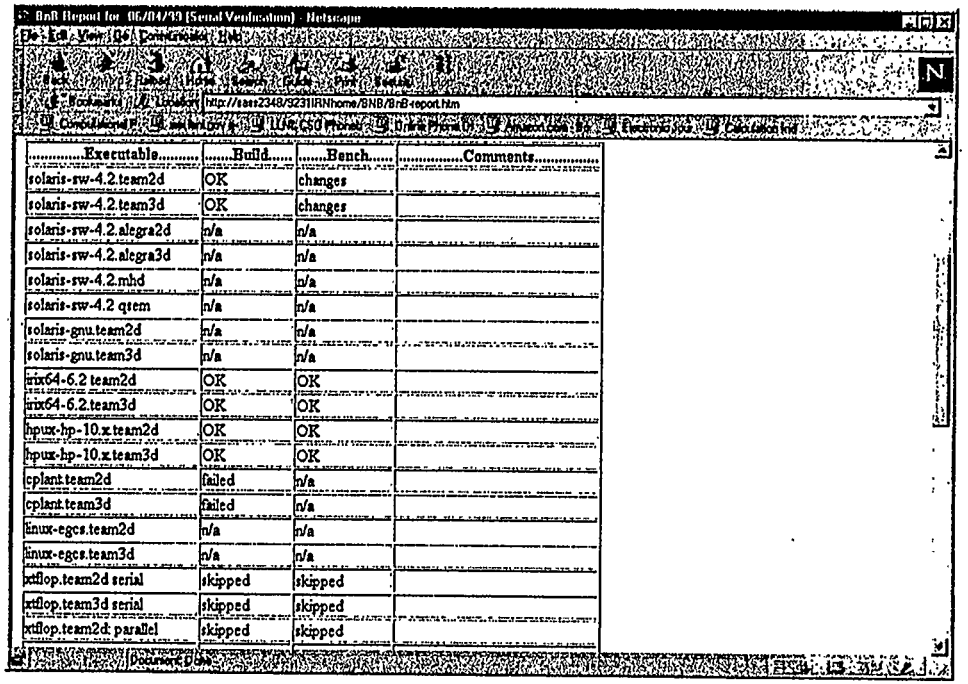
Example: ALEGRA "Static" Testing Tricks

Use of program development "environments" (ObjectCenter, SparcWorks)

Rational Software Inc tools, including Purify, Quantify, PureCoverage

Compilation on multiple platforms EVERY (daily) build

- **An example build summary is shown to the right**



Executable	Build	Bench	Comments
solaris-sw-4.2.team2d	OK	changes	
solaris-sw-4.2.team3d	OK	changes	
solaris-sw-4.2.alegra2d	n/a	n/a	
solaris-sw-4.2.alegra3d	n/a	n/a	
solaris-sw-4.2.mhd	n/a	n/a	
solaris-sw-4.2 qtem	n/a	n/a	
solaris-gnu.team2d	n/a	n/a	
solaris-gnu.team3d	n/a	n/a	
irix64-6.2 team2d	OK	OK	
irix64-6.2.team3d	OK	OK	
hpux-hp-10.x.team2d	OK	OK	
hpux-hp-10.x.team3d	OK	OK	
cplant.team2d	failed	n/a	
cplant.team3d	failed	n/a	
linux-egcs.team2d	n/a	n/a	
linux-egcs.team3d	n/a	n/a	
ptflop.team2d serial	skipped	skipped	
ptflop.team3d serial	skipped	skipped	
ptflop.team2d parallel	skipped	skipped	

Verification in ASCI - verification testing (I don't intend to say much about validation.)

Sandia National Laboratories

If we are talking about "dynamic" testing we are actually talking about two (plus one) different kinds of testing.

- The program plan calls for three different levels of "verification" testing and four different levels of "validation" testing.
- In this talk, I distinguish "regression" testing from these other kinds of testing.
  - Verification testing aims at assessing implementation accuracy
  - Validation aims at assessing conceptual model correctness
  - Regression testing aims at assessing software robustness and/or stability

Sandia National Laboratories

Regression testing - daily testing of the stability of the code performance.

- We (ALEGRA) do it to minimize rework after repository checkins.
- Clearly, there are issues of how to do this and use this process properly.
- Interesting criticisms of James Bach:
  - Danger of the tribe of code developers worshipping at the alter of the regression test suite to learn if their code is "correct:" - is the test suite SUFFICIENT?
  - Takes a lot of time to EXECUTE the regression test suite.
  - Takes a lot of time to UPDATE the regression test suite.

The reality of regression testing in ALEGRA.

- No intent for the test suite to be sufficient - we are testing stability of the code to new checkins.
- We spend a lot of time updating the regression test suite - it changes as we fix/improve the code, hence baselines must be redone quite frequently.
- We force the amount of time to execute the suite to be "livable."
  - Honor system for checkins, not mandatory
- Some information:

  Automatically executed under developer request, plus independent execution of subsets

  ~ 200 different problems

  Run across six different hardware platforms, heading toward seven

  Actual baseline comparison only done on SUN Solaris platform

  Takes roughly "overnight" to run entire set

Sandia National Laboratories

We Emphasize: Regression test suites need not be verification test suites, but they should be.

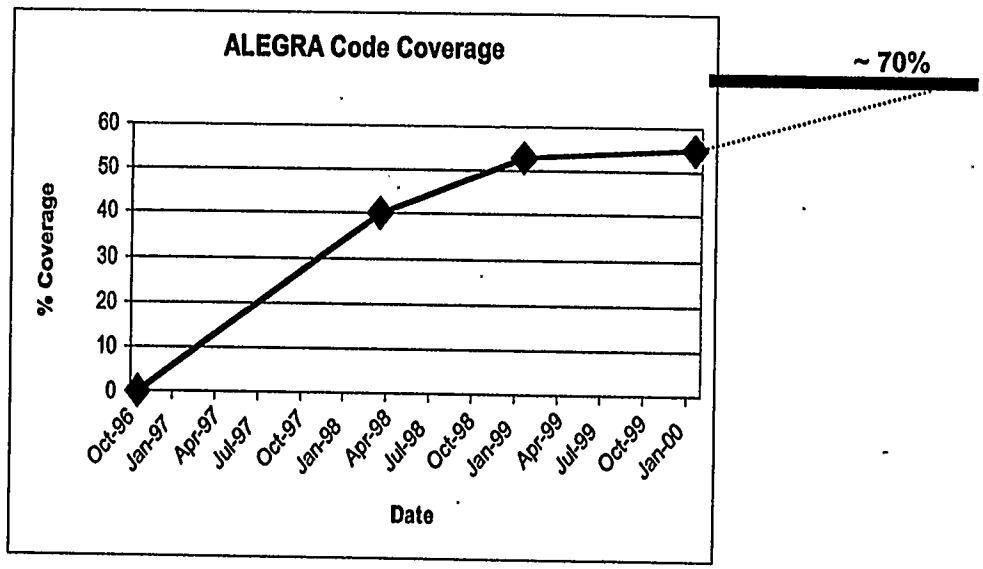
Regression testing measures implementation stability, not correctness. It is a software engineering practice, not a verification activity per se. We are not using regression testing as an oracle.

We do it and it leads to much interesting insight.

Our regression test set currently covers less than 60% of the functioning software (not including pre- and post-processors).

The appropriate metric is: we spend much less time fixing bad checkins than we did before - but they still creep in, even in parts of the code covered by the test suite.

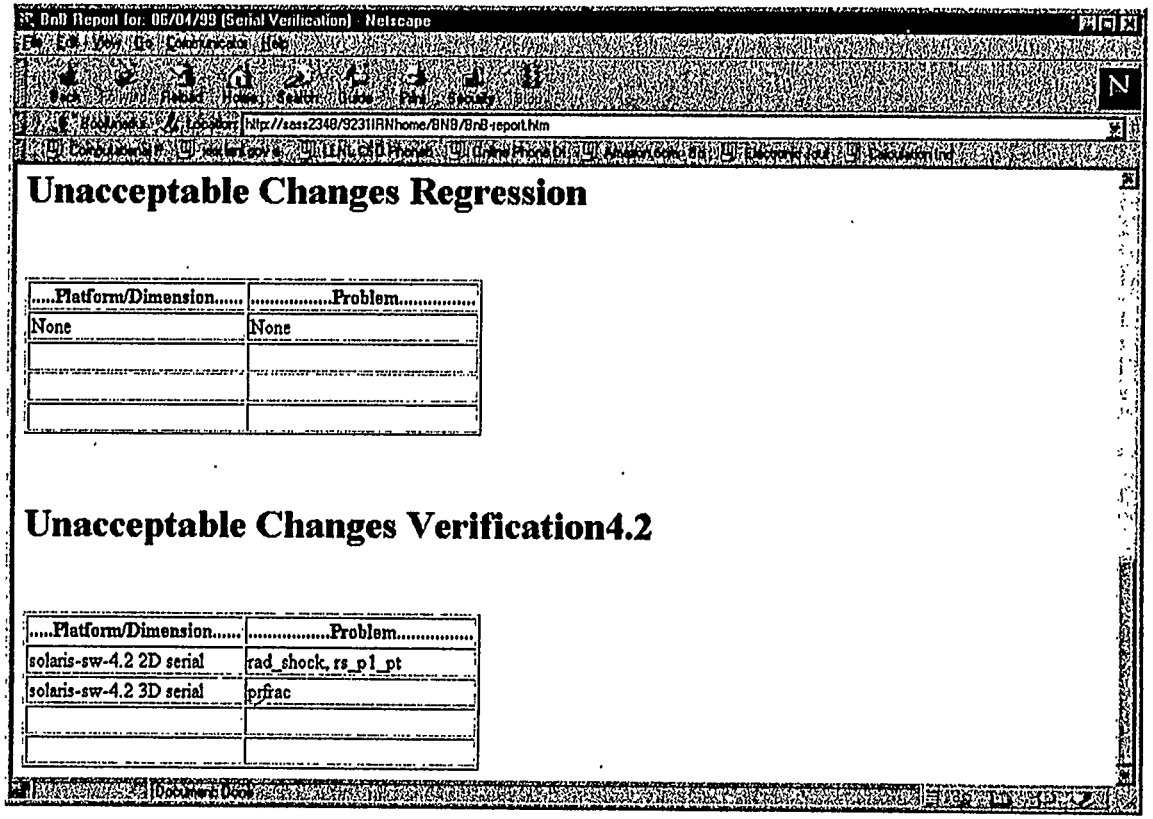
Serves as the kernel for a much larger benchmark (verification) test suite.



Daily reports from the build grind are provided as a Web page.

Unacceptable changes:

- **Regression**
- **Verification**
- **[Validation]**



Sandia National Laboratories

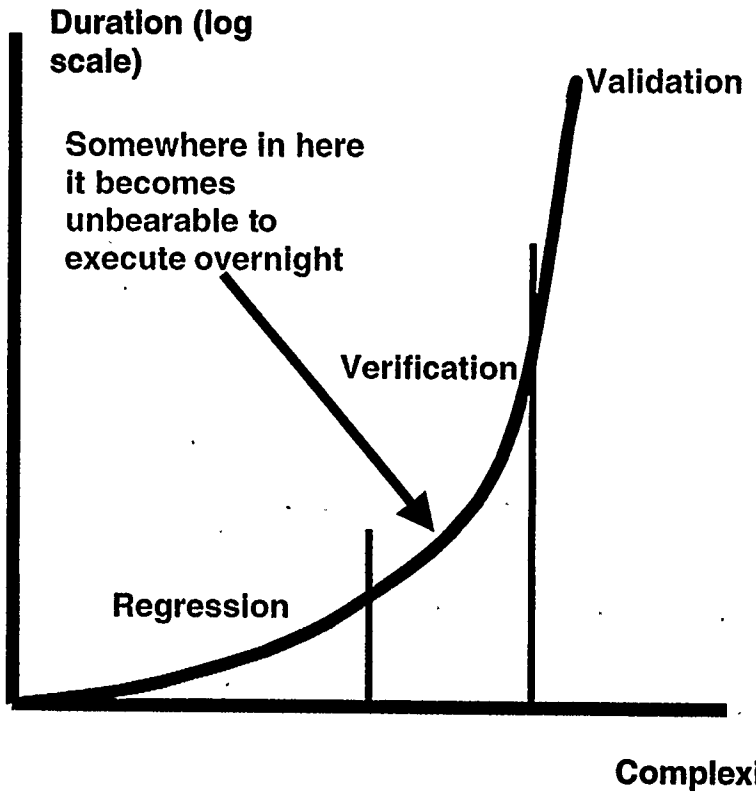
Balancing testing and the time to execute those tests is a very difficult problem.

Testing will never be sufficient to determine the correctness of these implementations.

We seek to test the most important stuff (coverage, sensitivity, and metrics).

We have idiosyncrasies - for example, all problems should be driven by standard code input.

Comparing baselines is an art - we continue to evolve tools that allow us to do this with the most accuracy and flexibility.



Conceptual Basis for Limiting Testing

The taxonomy of verification and validation tests puts a premium on deciding what is important and what is not.

- **Verification (is the math working right?):**
  - Analytic solutions, etc
  - Semi-analytic solutions
  - Code-code comparisons
- **Validation (comparison with physical reality):**
  - Single phenomena
  - Coupled phenomena
  - Integral phenomena
  - Acceptance testing
- **What is important depends on the particular application**
- **We are led to focus testing in particular applications**

Sandia National Laboratories

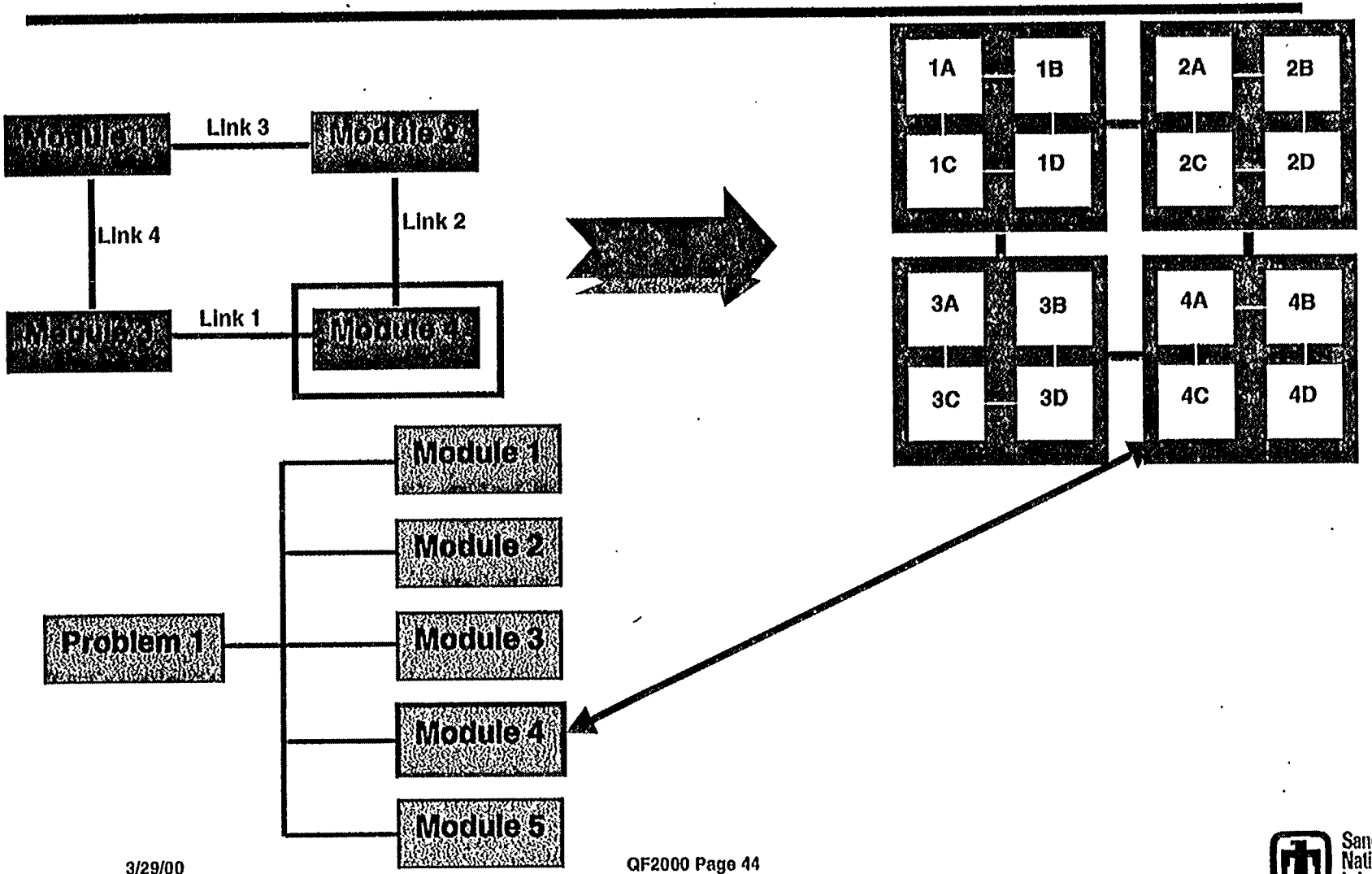
Component testing and coverage metrics must be linked for efficient high granularity testing.

What level of code component granularity is most effective for techniques like regression testing?

- We need to assess both module and path coverage for specific problems. This requires tools.
- We can't test every single "module" and path at the lowest level. We are driven to larger modules, which implicitly contain more paths.
- What is an appropriate module size for efficient testing (so it does not take two weeks of running before you can do a code check in?)
- What is an appropriate measure of module importance? Of path importance?

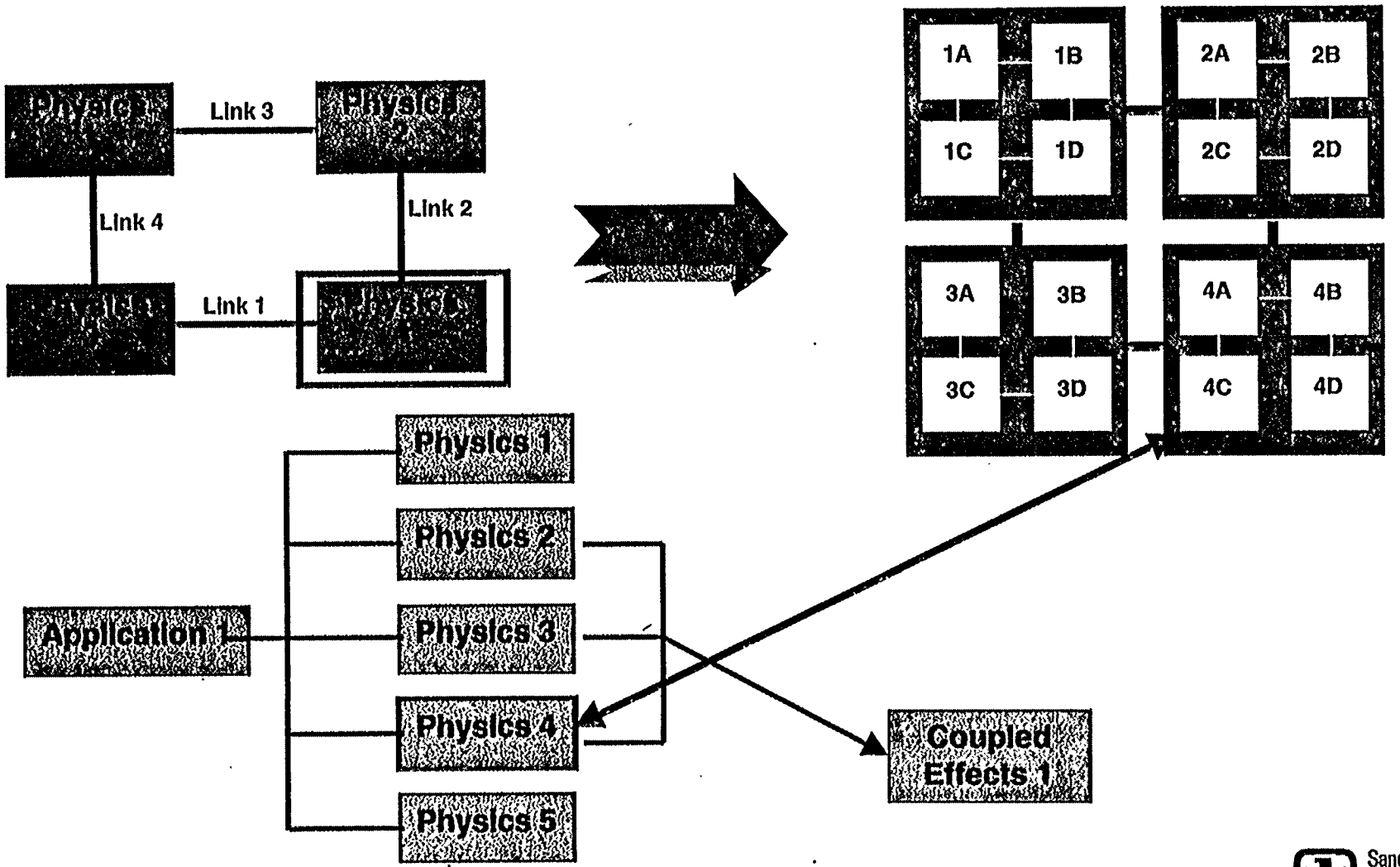
Sandia National Laboratories

Links between test problems and code modules/paths are needed for component testing and deciding granularity.



Sandia National Laboratories

A similar picture is true for validation - but validation is far more complex and time-consuming.



Sandia National Laboratories

In conclusion - we emphasize that there are tensions in the verification process.

- Verification vs **VALIDATION**

  Logically, codes should be verified before they are validated!

- Implementation correctness versus time
- "Software product" vs "computational science"
- Product cost versus product quality
- Higher quality but frozen product vs lower quality but dynamic product

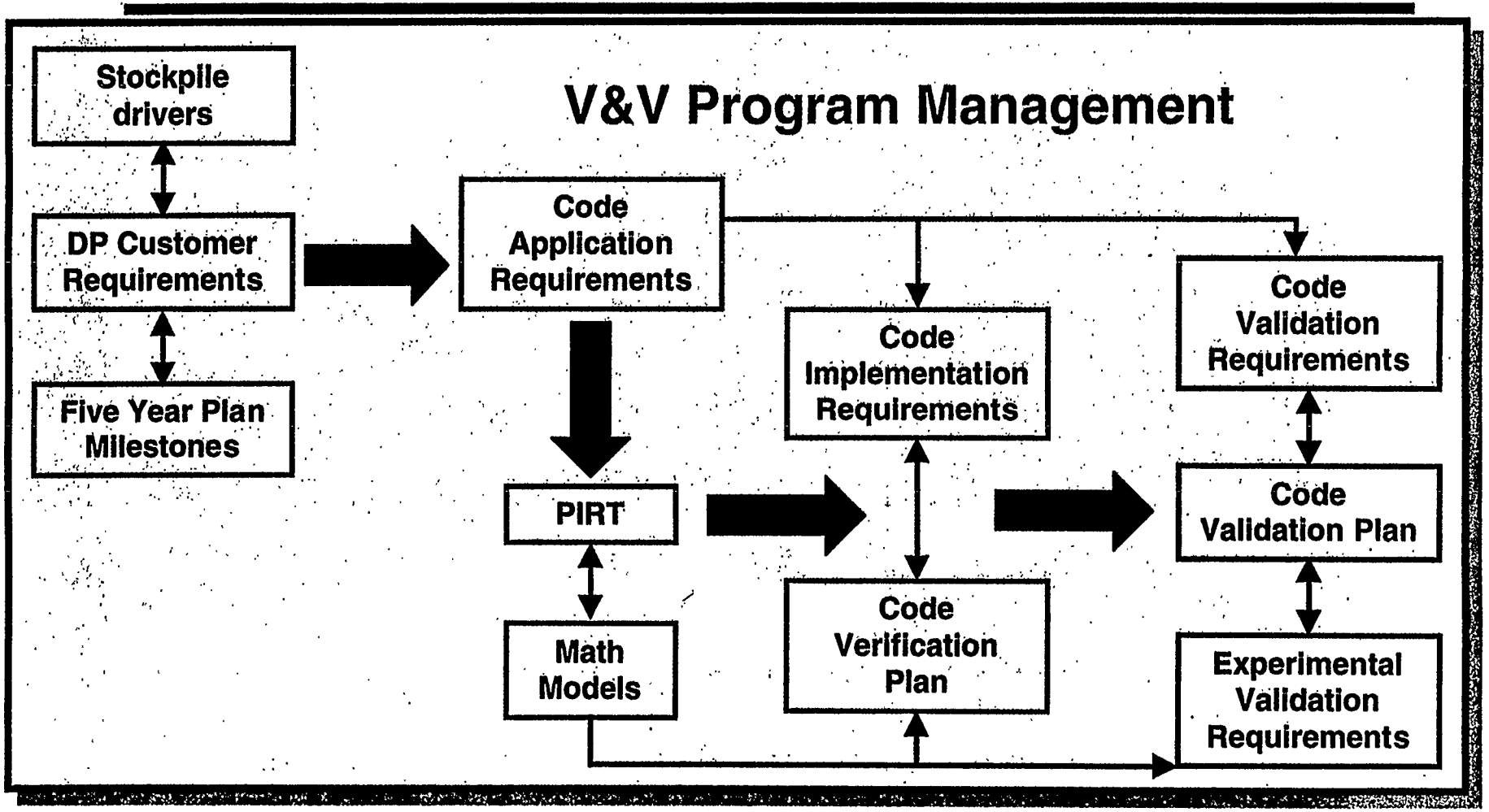
A big problem for this program is the retrofit character. Obviously, if we had it to do over again ...

Every code project must strike the balance that makes sense for them.

Sandia National Laboratories

ASCI V&V - Process at Sandia

The ASCI V&V program at Sandia requires V&V plans from all of the ASCI codes.



Sandia National Laboratories

Table of Contents for requested V&V Plans (evolving)

Table of Contents

1.0 Introduction

2.0 Overview of V&V Plan

3.0 M&S Requirements

- 3.1 Customer Requirements
- 3.2 Performance Requirements
- 3.3 *Identification & Ranking of Physical Phenomena*
- 3.4 Model Capability Assessment
- 3.5 Identification of Deficiencies

4.0 Verification Plan

5.0 Validation Plan

6.0 Protocols for Stockpile Computing

7.0 Software Management

The Sandia program emphasizes requirements based V&V, with special aim at validation requirements.

Requirements should be based on ultimate intended applications of the codes to SBSS.

Priorities are needed to deal with limited money.

Priorities are needed to deal with limited opportunity for experimental validation work (there will never be enough money or time to do all the needed work).

Specific application focus of the requirements lends itself to code "accreditation," thus curing the major problem that V&V is an open ended process.

The Sandia ASCI V&V program is raising the level of awareness of the software life cycle for ASCI scientific software as a fundamental part of verification.

The program is emphasizing:

- Software quality engineering
- Greater formality in testing
- Use of technology
- Focus on requirements-driven software testing

The overarching rationale - you can't prove the codes are correct, so close all of the loopholes that you possibly can.

We are asking that the requested V&V plans discuss what the code development is doing in the Level 2 key process areas.

Feedback from the FY99 SQE survey provided guidance to this part of the program.

Reminder: The processes of interest are:

- Requirements management
- Project planning
- Project tracking and oversight
- Sub-contract management
- SQA
- Configuration management

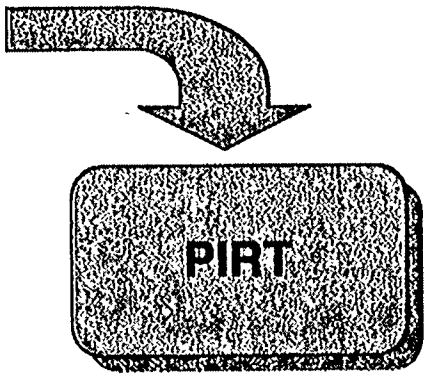
This is a work in progress!!

In the V&V program view at Sandia, validation is driven by *Requirements*.



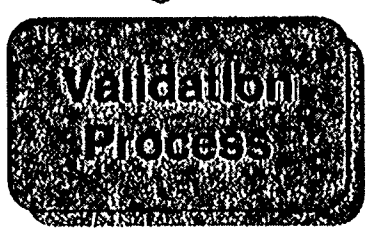
- SLEP drives all, AWLPG 99-0
- Focus on one major application
- Work with some customers
- Define needs, uses, and accuracy requirements
- Focus on a major application leads to sneaky code "certification" or "accreditation"

Phenomena Identification and Ranking Table



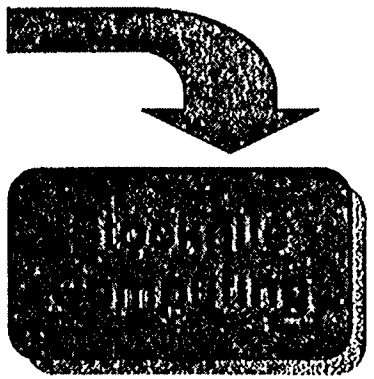
- Translate requirements into phenomena to be modeled
- Assess importance of phenomena and current capability of the modeling

- PIRT leads to experimental and archival data priorities



- PIRT leads to tiered validation test program

- Define how the code should be used to achieve the stockpile requirements
  - Training?
  - Uncertainty quant.?
  - Input lockdown?
  - etc?



Example of a PIRT (from Justine Johannes)

Phenomena Identification and Ranking Table (PIRT)

Neutron Generator Simulations	Importance to NG Performance		Model Adequacy (physics models and/or coefficients)
	Predict Neutron Production	Trends with Design Changes	
PLASMA CUP (C)			
C-1. Inflow Description	High	High	Inadequate
C-2 Surface Interactions: a. Secondary Electron Emission b. Neutralization c. Surface Reactions d. Thermal/Angular Reflection e. Surface Charging	Low Medium Low Low Medium	Medium Medium Low Low Low	Inadequate (3D) Inadequate (C) adequate Inadequate (C) Adequate Inadequate (C)
C-3. Neutral, ion and electron densities and energy distributions	High	High	Adequate(2D) Inadequate (3D)
C-4. Gas Phase Reactions: a. Electron Impact b. Excitation Reactions c. Coulomb Interactions d. Charge Exchange Reactions	High High Medium Medium	High High Medium Medium	Inadequate (3D,FK) Inadequate (3D, FK) Adequate Adequate
C-5. Electric fields	High	High	Inadequate
C-6. Magnetic fields	Low	High	Inadequate
C-7. Sheath	Low	Low	Inadequate
C-8. Outflow Description	Medium	Medium	Adequate
C – transport or interaction parameters, FK – full kinetic simulation			

Validation Experiments

January, 2000 status of the Sandia V&V planning process

Version 1.0 completed for 75% of codes

More emphasis on validation than software verification

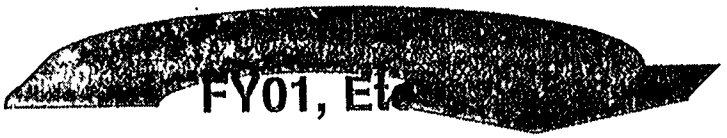
- Software process improvement group

Confirms the limitation of our validation program

- data to support *development* and validation of material models
- development of a material database

Peer review process starts in FY00

A formal peer review process is beginning in FY00



December, 1999

Guidelines Version 1.0

April, 2000

FY00 Peer Review Process

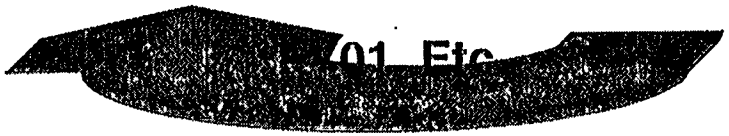
September, 2000

Guidelines Version 2.0

Metrics?

Three levels:

- Programmatic
- Technical and benchmarking
- Independent V&V



[redacted]

Weapons Program

[redacted]

Sandia V&V Program

[redacted]

Experiments

Challenges: Technology support for verification across the program.

Tools:

- Software architecture tools (like architecture visualizers)
- High performance static testing
- High fidelity dynamic testing
- Making tools common

High impact SQE and code applications environments

- Like the Basis environment at LLNL

Additional concepts:

- Advanced design concepts, like "Design by Contract"
- Application of "formal methods" [for what?!]

Challenges: ALEGRA code development.

We would like to build on what is working:

- Regression testing
- Automated V&V testing
- Multi-platform builds

We are also uncomfortable with the "enhanced" V&V retrofit

Defining and applying good metrics is important

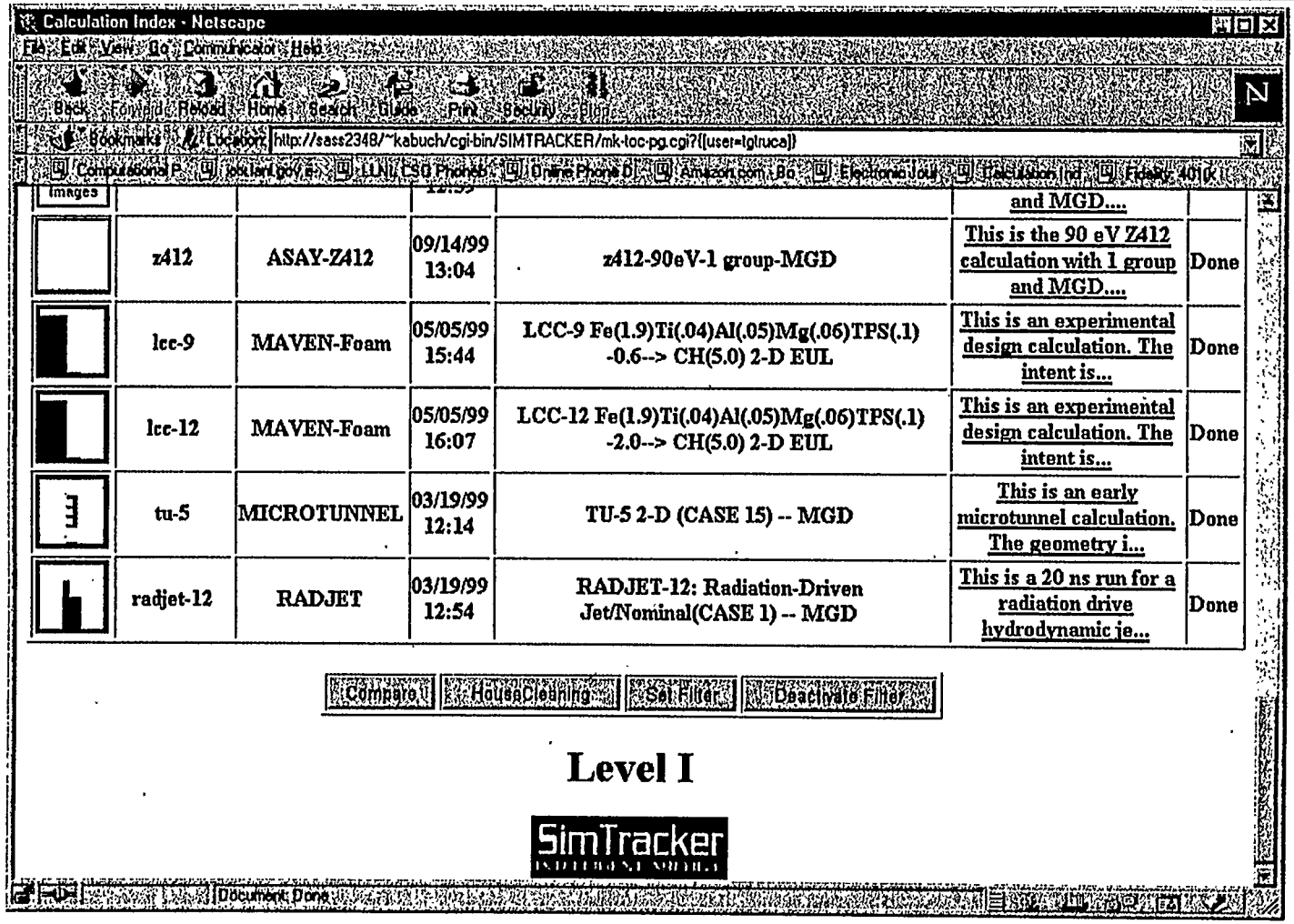
The "information infrastructure" to support V&V is going to have to grow dramatically over the next few years

- High quality archive systems (ALEGRA_Tracker)
- Storage of experimental - calculation comparisons
- Information shared between labs

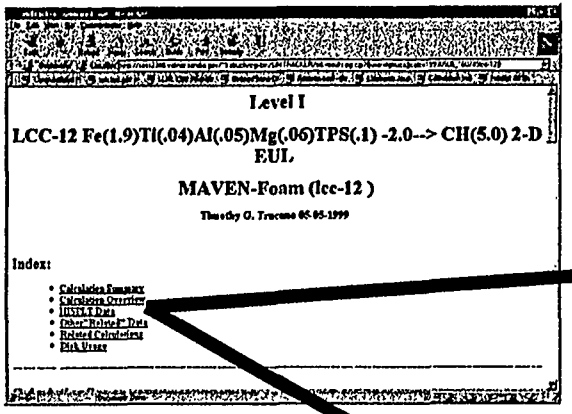
Additional software engineering

- Design by Contract software inspection methodology

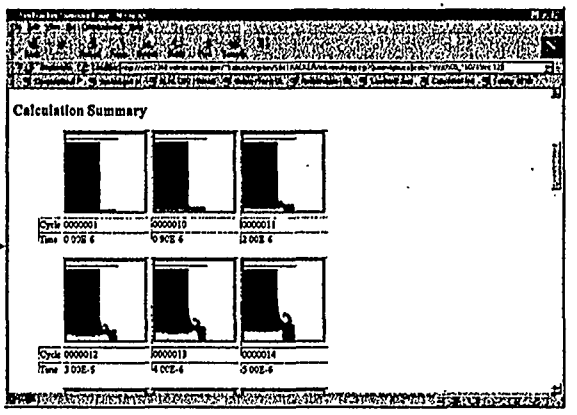
Highly informative archiving systems are REQUIRED for validation. Example: ALEGRA_Tracker

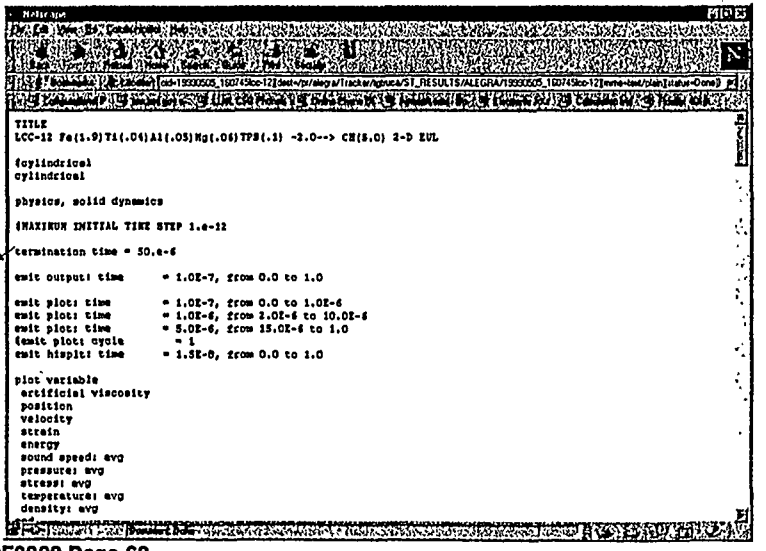


The ideal is easy access to a lot of information, available on a long time scale, over a large computer network



Access computations, data, code-code comparisons





Built-in traceability

Conform to selected V&V processes

Archivability

Lab-lab information sharing

Sandia National Laboratories

Further open questions of interest:

How do you verify and validate proprietary commercial software?

What IS the role of advanced methods like DbC, formal methods, etc.?

What are quantitative measures of our confidence in the ASCI codes?

What IS the appropriate software lifecycle for ASCI codes?

Sandia National Laboratories

In summary:

We are serious about V&V as process.

We are serious about high consequence stockpile computing.

There is an element of research in our approach to V&V, especially in the technology component.

Will it all really work out? See me again in four years.

In conclusion:

"Summary: Computers Are Here To Stay. They Endanger Thought, Language, Science, and the Survival of Man. Like Any Other Dangerous Tool, They Should Be Put Under Strict Controls."

Clifford Truesdell, "The Computer: Ruin of Science, Threat to Man" in An Idiot's Fugitive Essays on Science

Sandia National Laboratories



A Pragmatic Approach to V&V

Software Quality Forum 2000

Ann Hodges
Information Systems Engineering Center 6500

Sandia National Laboratories



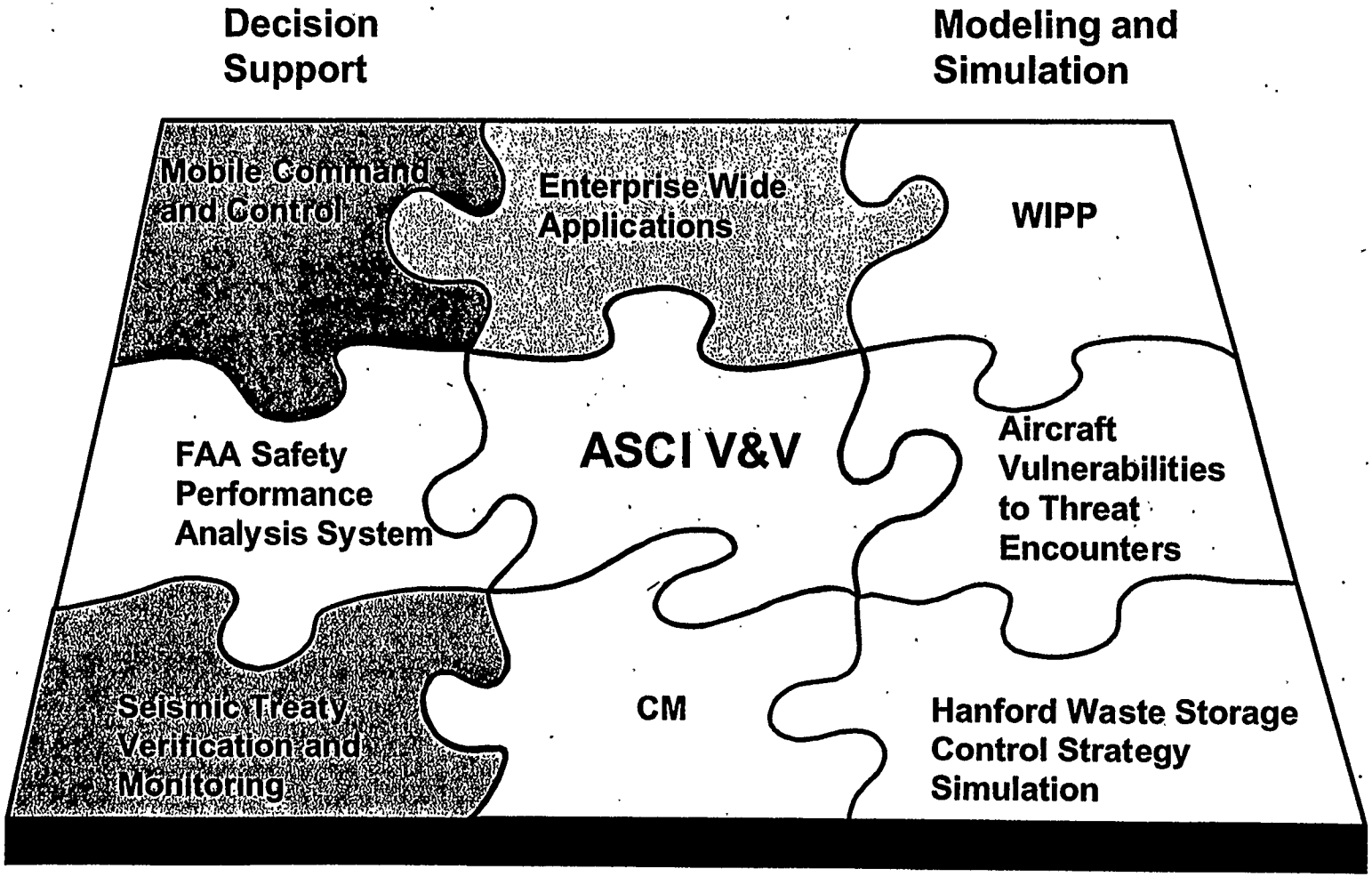
Testing Maturity Phases

The purpose of testing[1] is:

- **Phase 0: No difference between testing and debugging.**
- **Phase 1: Show the software works.**
- **Phase 2: Show the software doesn't work.**
- **Phase 3: Reduce perceived risk of not working to an acceptable level.**
- **Phase 4: Testing is a mental discipline that results in low-risk software *without* much testing effort.**

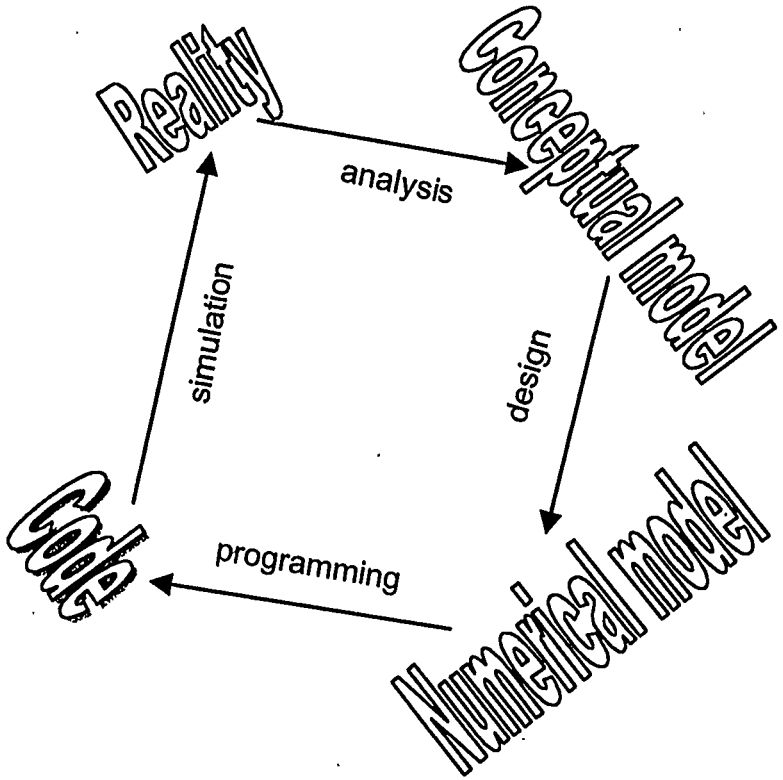
[1]Boris Beizer, Software Testing Techniques, 2nd edition, Van Nostrand Reinhold, 1990.

V&V/Test Programs Experience

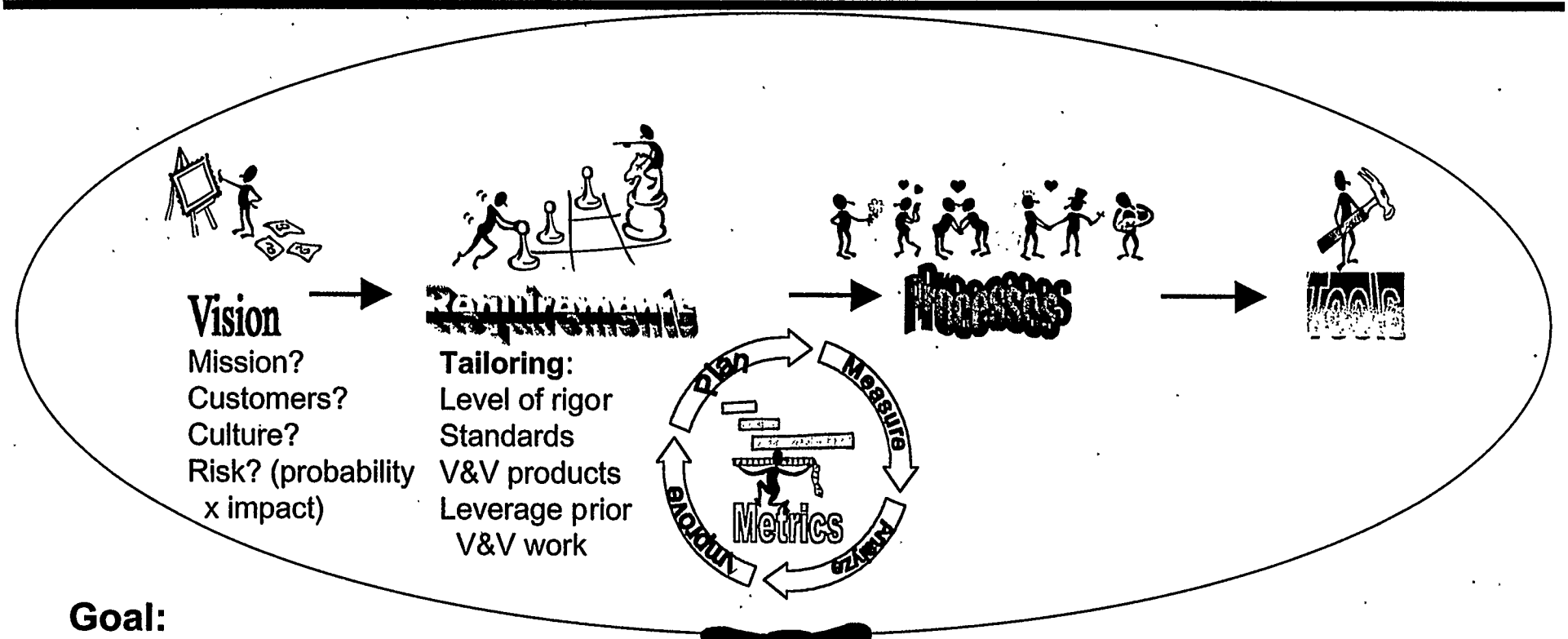


V&V of Scientific Software

Validation: Is it the right model?
Verification: Is the model right?



Process of Developing a V&V Program



Goal:

- **Determine right level of V&V**
- **Most cost/time/resource efficient**
- **Justification for V&V level**

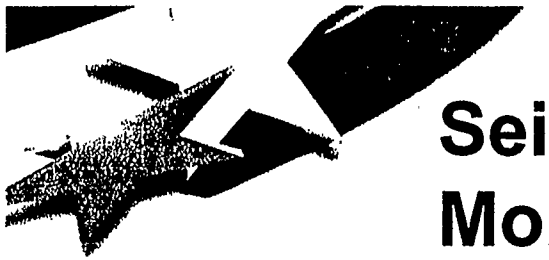


Mobile Command and Control System

- Fault Isolation Built In Test Equipment for MBB system
  - C3 system
  - secure
  - interconnected via broadband LAN
  - FIBITE controlled test equipment to test and report on devices
- Security Test Plan
  - based on security path analysis
  - systematic approach to testing
    - Test procedure for each piece of the path
    - Evaluated by external agency
  - run through tests
- CM
  - different customers, different software - customer build ID
  - different test environments
- Testing maturity phase 0->1

Seismic Treaty Verification and Monitoring

- Validation
  - requirements-based tests for Air Force DT&E
  - test plan maps to test procedures
  - analysis, demo, test procedures
- Operational testing
  - live data experiments
  - OT&E
  - comparison with current operational system
- CM
  - appreciation for promotion model
  - build ID - "tip" version
  - issue tracking
    - home grown
    - DDTS



Seismic Treaty Verification and Monitoring

- V&V strategy and management
  - Industry standard practices - IEEE
  - Usability of system is more than satisfying requirements
  - Testability
    - requirements - ambiguity (state of art signal processing) ultimately addressed in test procedures
    - design - processing pipeline
  - Coverage and results - "battle map"
  - Combine validation tests with integration testing
  - Test summary report
  - Visibility of problem reports - negotiated with Air Force
  - Training Air Force software maintenance staff



WIPP

- Pragmatic software engineering
  - support test activities
  - reverse engineered requirements, design - included in code specific test plan
- CM
  - test team had initial VM tool, participated in WIPP's selection
  - architecture for supporting simulations - input, output, sims
  - used by EPA - at Activity Integration Forum, Gary Froehlich reports key to opening WIPP site
  - CM Plan
  - issue tracking



WIPP

Test Strategy

- Infrastructure
  - Common test plan
  - Code specific test plan
  - Templates
  - General test procedures
    - design for testability
    - numerical stability
    - tool usage
  - Test tools
    - coverage - used to elicit further requirements, judge test suite adequacy
    - flint
    - structural reverse engineering
    - complexity metrics
  - Training
- Public safety, EPA compliance
  - bureaucracy fallout
  - QA requirements
- Standards - IEEE, NQA
- Testing maturity phase 3
- Developer partnership



FAA Safety Performance Analysis System

- Validation
  - Functional, operational, physical requirements
- Verification
  - traceability analyses
  - evaluation of development products
- V&V strategy and management
  - public safety
  - IEEE standards
  - prioritization of requirements validation
    - criticality analysis - risk-based
    - availability of products
  - developer, program office partnership



Aircraft Vulnerabilities to Threat Encounters

- Pragmatic software engineering
  - reverse engineered requirements
  - test cases based on requirements/features
- Validation
  - cross comparing with another agency's version
  - if differences, explanation
- V&V strategy and management
  - position code for wider use
  - Standards
    - DoD Modeling and Simulation Office (DMSO) VV&A
    - IEEE



Enterprise Wide Applications

- Test strategy and management
  - performance/load testing
  - design for testability
    - transactions traced through distributed components
    - monitoring key architectural components during test (I/O, CPU, threads, cache utilization)
  - automated GUI, load test tools
- CM
  - > 50 projects sharing VM, process
  - issue tracking
  - promotion model
  - automated deployment



Hanford Waste Storage Control Strategy Simulation

- Validation
  - tests comparing to field data, experiments, other models
    - explanation of differences
    - increased understanding of model requirements -> *science didn't stop - aided refinement!!*
  - tests based on analyst's expectation of phenomenological behavior
  - data validation
- Verification
  - evaluation of conceptual model
  - rederivation of numerical model
  - code inspection for scientific principals, model implementation
  - traceability between model, requirements, design, implementation (code and data)



Hanford Waste Storage Control Strategy Simulation

V&V strategy and management

- public safety
- ANS, IEEE, Hanford QA standards
- prioritizing V&V test activities for deliveries
  - new models
  - changed models
  - regression tests
- test coverage of all models, software products
- analysts part of V&V team
- issue tracking
  - customer
  - developer
  - V&V
  - reporting
- external review process
- increased reliability of simulations
- increased confidence in simulations
- design for testability - visibility into models (intermediate parameter values)



Pragmatic Approach to ASCI V&V

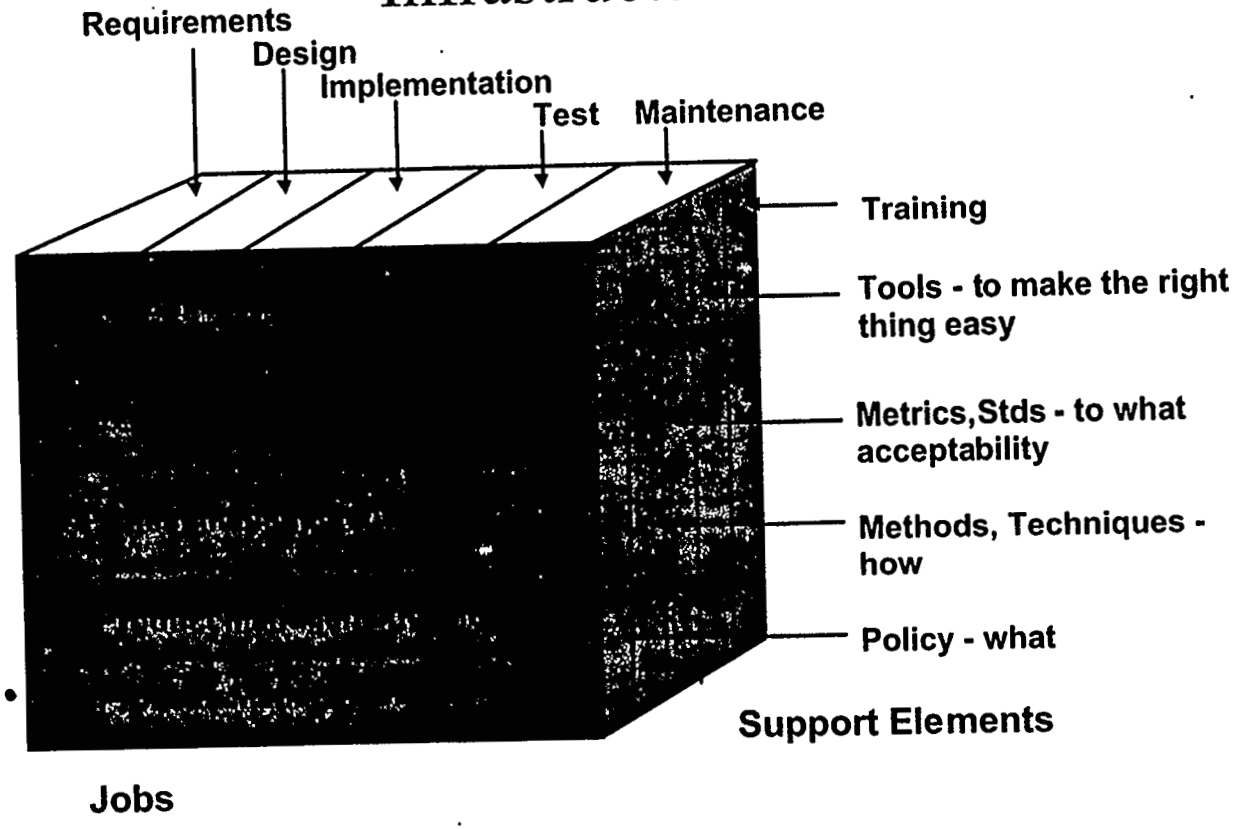
- Vision, requirements
  - Science Based Stockpile Stewardship
  - increase confidence of simulations
  - increase reliability
  - movement of R&D to product
  - support both R&D and production in parallel
  - common understanding of quality - physics, software
  - defensibility in process, rigor
  - qualification for treating "code" as product
  - training, documentation, support
  - software engineering - common language (discussed UML as a possible candidate)
  - metrics - quantity, quality (coverage), experiments related to test cases
  - V&V framework
  - Assist meeting cost and schedule

Pragmatic Approach to ASCI V&V

- How to start with defining this V&V infrastructure?
  - SNL V&V Plans review
  - Mini-software process assessments

ASCI Apps V&V Infrastructure

Software Product Development Infrastructure





Pragmatic Approach to ASCI V&V

V&V strategy and management

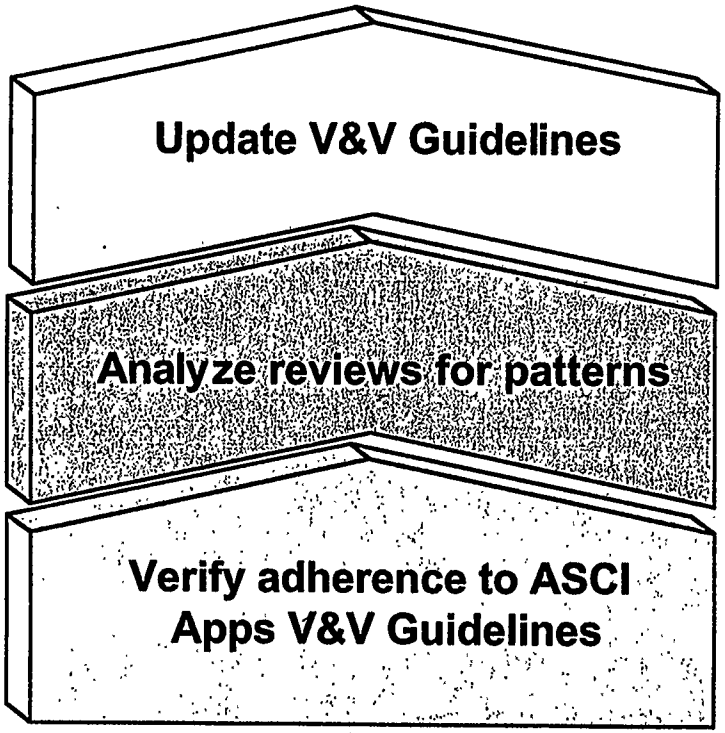
- identify appropriate standard industry practices
  - AIAA, ANS, NQA, IEEE, ...
- establish infrastructure
  - allows code teams to *focus on the science*
  - appropriate level of rigor
  - CM, RM, templates, tailoring guidelines, processes, tools, training, metrics
  - address ASCI-specific areas
- leverage on R&D culture
- reverse engineer to support V&V
- facilitate science
  - model behavior
  - simplifications
  - experiments <-> simulations
- accredited code usage
  - environment
  - simplifications
  - characterize areas of stability
  - training
- design for testability
- COTS
  - acceptance testing - scenarios
  - integration with code

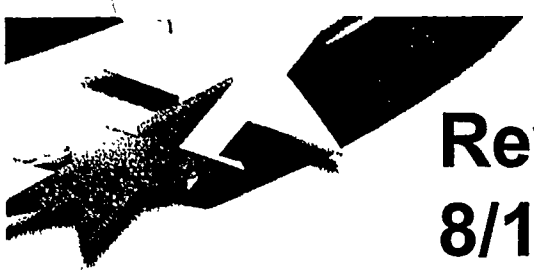
ASCI Apps V&V Plan Review

Phase 3

Phase 2

Phase 1





Review Phase 1: Verify Adherence to 8/1999 V&V Guidelines

- Review template highlights key requirements
  - Programmatic
  - Stockpile Drivers
  - Phenomena Identification and Ranking Table (PIRT)
  - Verification
    - software development process areas
    - software engineering techniques
    - issues and challenges
    - software testing process
  - Validation
    - requirements
    - validation plan
  - Protocols for stockpile computing
- Comments in a requirement entry signifies reviewer could find that aspect addressed
- Empty entry implies information not evident to reviewer
- Multiple reviewers



Review Phase 2: Analyze Plans and Reviews for Patterns, Trends

For each requirement

- how many plans addressed
- may indicate misunderstandings with respect to Guidelines

- Common definitions, perspectives
- Interdependencies between codes

This leads to

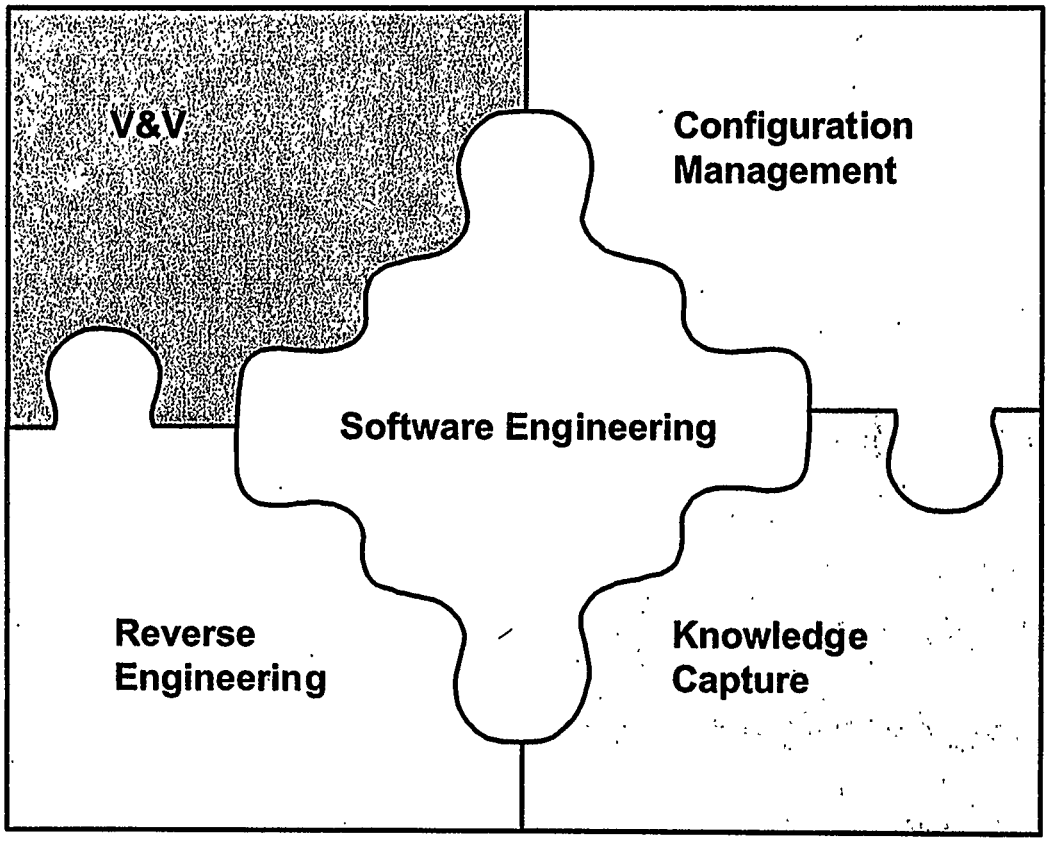
- Better knowledge of current understanding, requirements
- Crisper focus on process and Guidelines improvements



Review Phase 3: Update V&V Guidelines

- Based upon patterns, trends
  - identify areas of clarification
  - areas of commonality - vision, requirements, approaches
  - recommendations for Guidelines at programmatic level
- Becomes a key part of the ASCI Apps V&V Infrastructure

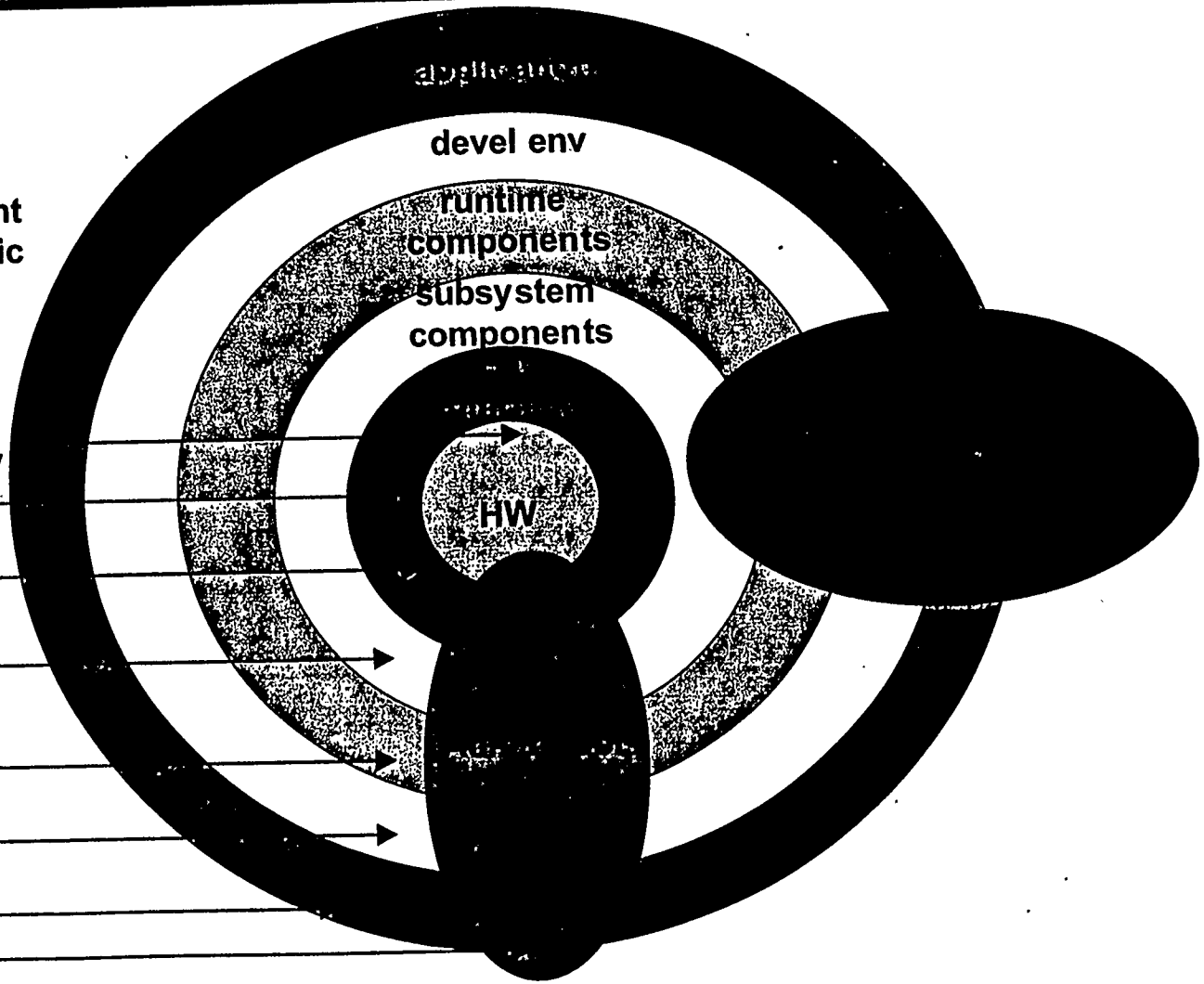
CPLANT Productionization Facets



CPLANT Test Architecture

Examples of architectural layer components to be tested

Red text implies the component is used to test CPLANT-specific components





References

AIAA G-077-1998, Guide for the Verification and Validation of Computational Fluid Dynamics Simulations, 1998.

ANSI/ANS-10.4-1987, American Nuclear Society, Guidelines for the Verification and Validation of Scientific and Engineering Computer Programs for the Nuclear Industry.

B. Beizer, Software Testing Techniques, Van Nostrand Reinhold, 2nd edition, 1990.

R. Black, Managing the Testing Process, Microsoft Press, 1999.

Defense Modeling and Simulation Office, http://www.dmso.mil/dmso/index.msql

IEEE Standard 829-1983, IEEE Standard for Software Test Documentation.

IEEE Standard 1012-1986, IEEE Standard for Software Verification and Validation Plans.

IEEE Standard 1059-1993, IEEE Guide for Software Verification and Validation Plans.

IEEE Standard 1220-1994, IEEE Trial-Use Standard for the Application and Management of the Systems Engineering Process.

Joint Accreditation Support Activity (JASA), http://www.nawcwpns.navy.mil/~jasa/

C. Kaner, et al, Testing Computer Software, 2nd edition, Van Nostrand Reinhold, 1993.

G. Myers, The Art of Software Testing, Wiley Interscience, 1979.

NWC Technical Business Practice 306, 1999.

NQA-1-1994, American Society of Mechanical Engineers, subpart 2.7 QA Requirements of Computer Software for Nuclear Facility Applications, Requirement 11 Test Control, Requirement 11S-2 Supplementary Requirements for Computer Program Testing.

SECNAV Instruction 5200.40, 1999.

PREDICT—A New Approach To Product Development

(What can Analysts do Without Data?)

PREDICT — Performance & Reliability Evaluation with Diverse Information Combination and Tracking

Jane M. Booker
Statistical Sciences Group, Los Alamos National Laboratory
505-667-1479 jane@lanl.gov
http://www.lanl.gov/orgs/tsa/tsa1/projects/predict.html

OUTLINE

I. What is PREDICT?

II. PREDICT Steps

(A flowchart of steps)

III. Analytical Issues in implementing PREDICT

(Examples of analytical issues, including exertions into expert judgment and fuzzy control systems)

IV. Audience participation in setting up a software reliability problem.

I. PREDICT is a process, a methodology

A Statistical Methodology / Process:
An integrated, quantitative assessment of performance with associated uncertainty of a dynamic system

Based on rigorous, mathematical methods (especially for the elicitation and analysis of expert judgment)

PREDICT is a multidisciplinary effort

Los Alamos Developers

Thomas Bement, statistics
Jane Booker, statistics
Mary Meyer, anthropology

Delphi Developer

Bill Kerscher, engineering
PLUS
Teams of experts from
Luxembourg
Mexico
USA

Los Alamos Co-Developers

Ron Smith, engineering
Masha Davidson, computation
Yvonne Martinez, computation
Jerry Parkinson, engineering
Dale Talbott, engineering
Stephen Becker, physics
Lyle Edwards, engineering
Fred Mortensen, physics
Richard Yactor, engineering

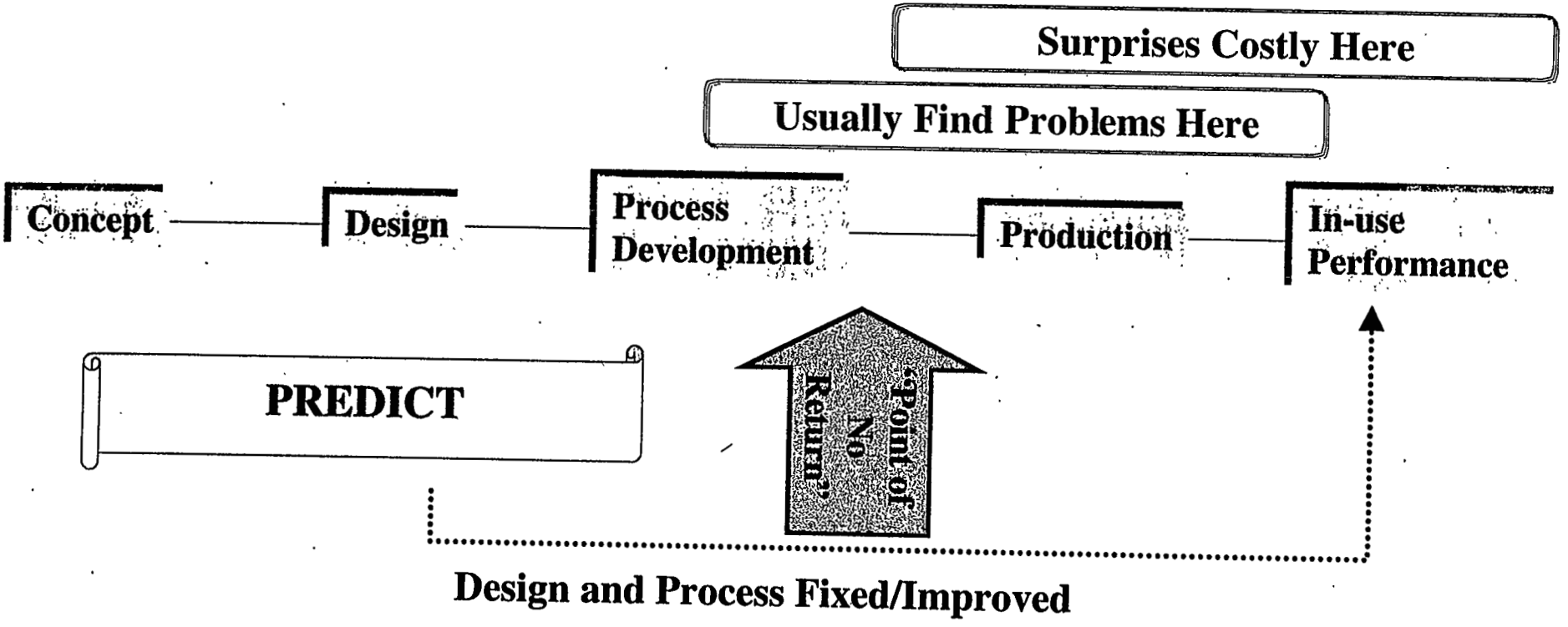
PREDICT system performance in the absence of traditional test data

Methods Developed for Two Parallel Applications

Fuel Systems at Delphi Automotive Systems *before testing* Pre-Prototype/ Pre-Production Concept Designs	Nuclear Weapons Program at Los Alamos *in the absence of testing* Systems Under Change (Aging)

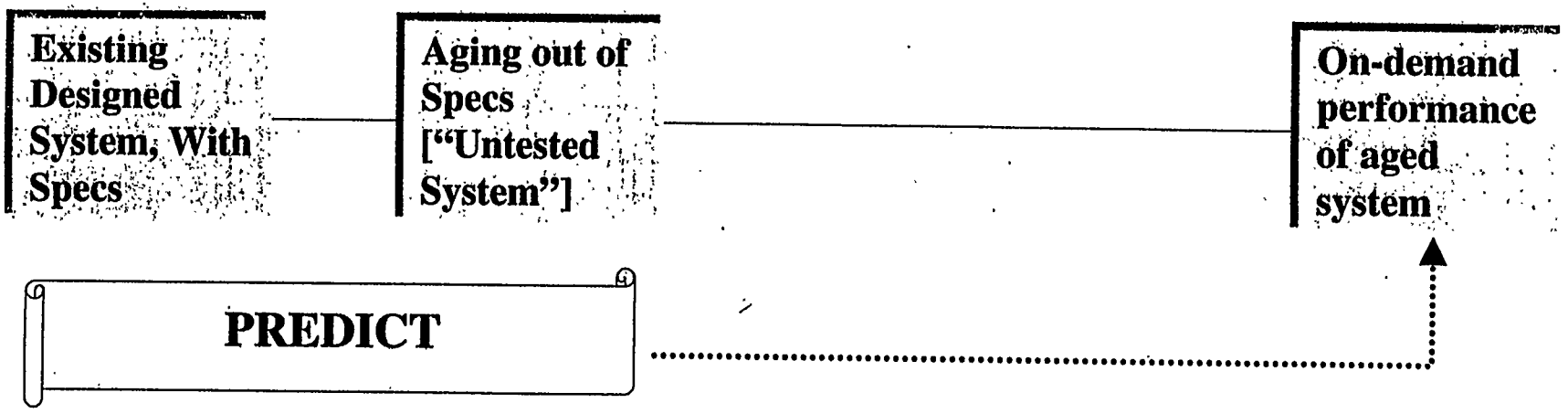
PREDICT avoids costly surprises

Industry Application Process
Predicting Concept System Performance

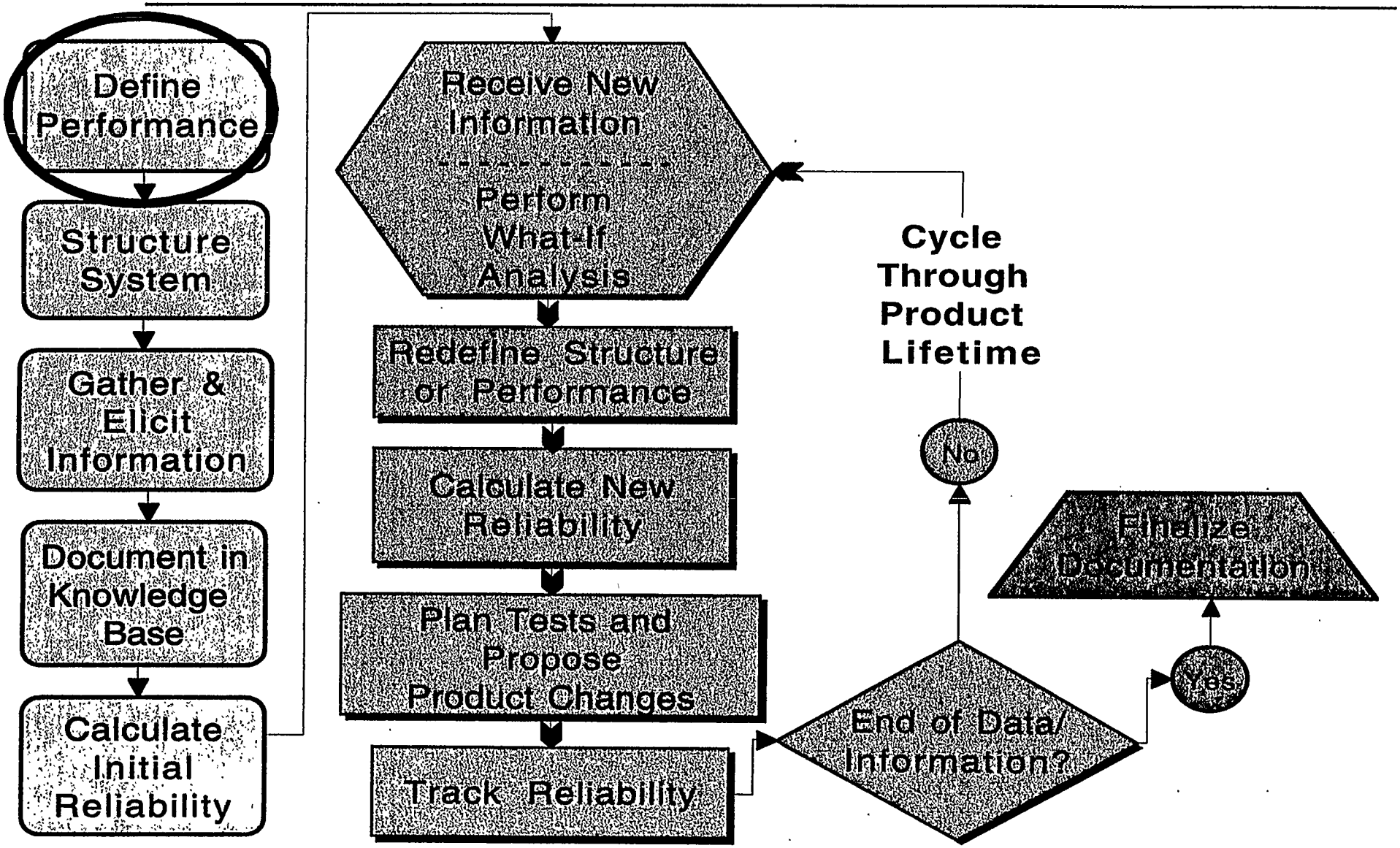


PREDICT monitors performance through system changes

Weapon Application Process
Predicting Aging System Performance

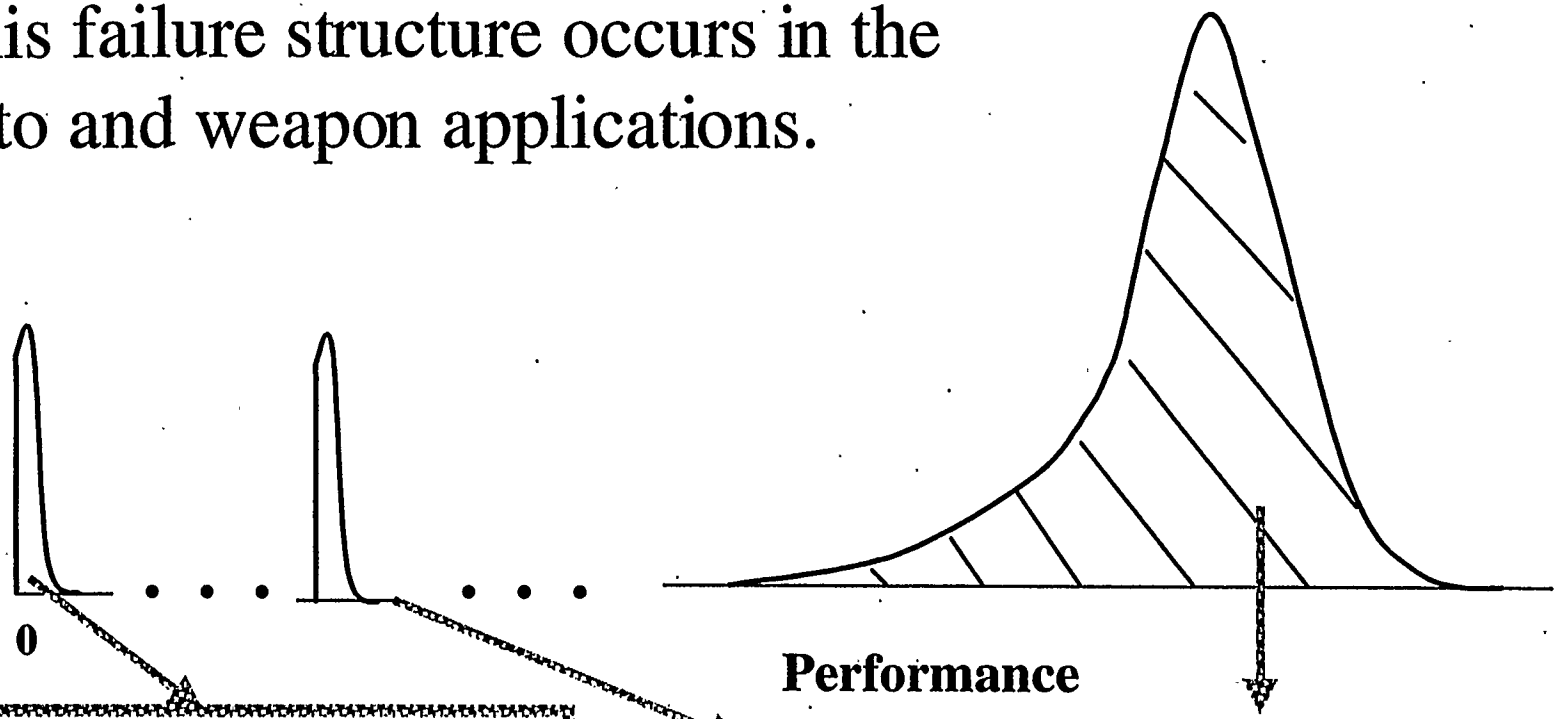


II. PREDICT Steps



Defining Performance / Reliability

This failure structure occurs in the auto and weapon applications.



Catastrophic Failures
Infant Mortality
From manufacturing,
Poor quality control, etc.

Extreme
(rare) cases/
environments

Reductions in
performance
from uncertainties,
new parts, etc.

Defining Performance / Reliability

For example , automotive engineers consider component performance in terms of failure rates and reliability as a traditional Weibull model for aging:

$$R(t) = \exp\left[-\lambda t^{\beta}\right]$$

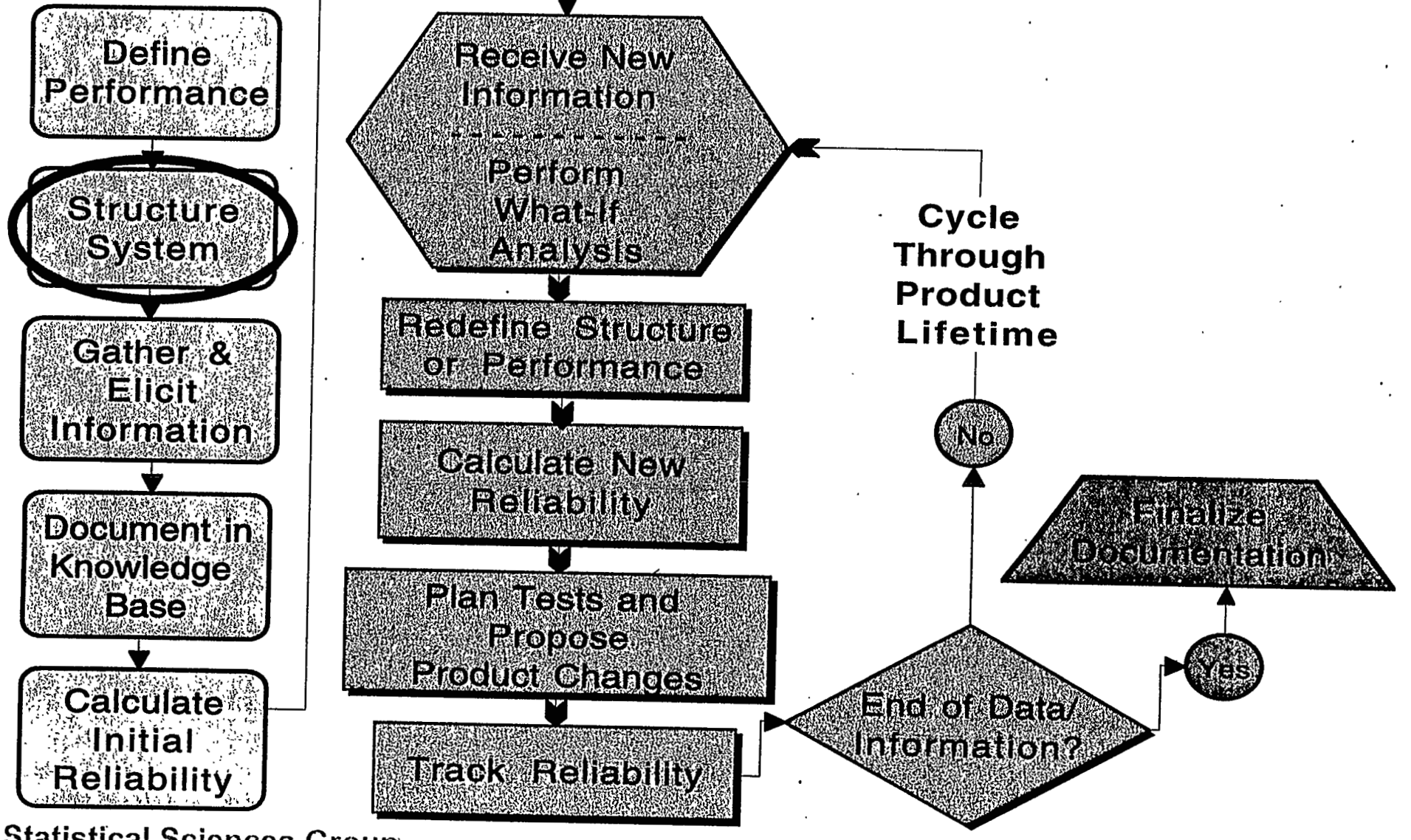
where λ and β have uncertainty distributions.

Uncertainty Distributions

Functions used to represent uncertainties:

- Probability Distribution Functions → From probability (crisp set) theory
- Cumulative Distribution Functions → From probability (crisp set) theory
- Likelihoods → From fuzzy set and possibility theory
- Membership Functions → From fuzzy set and possibility theory
- Possibility Distributions → From fuzzy set and possibility theory

II. PREDICT Steps



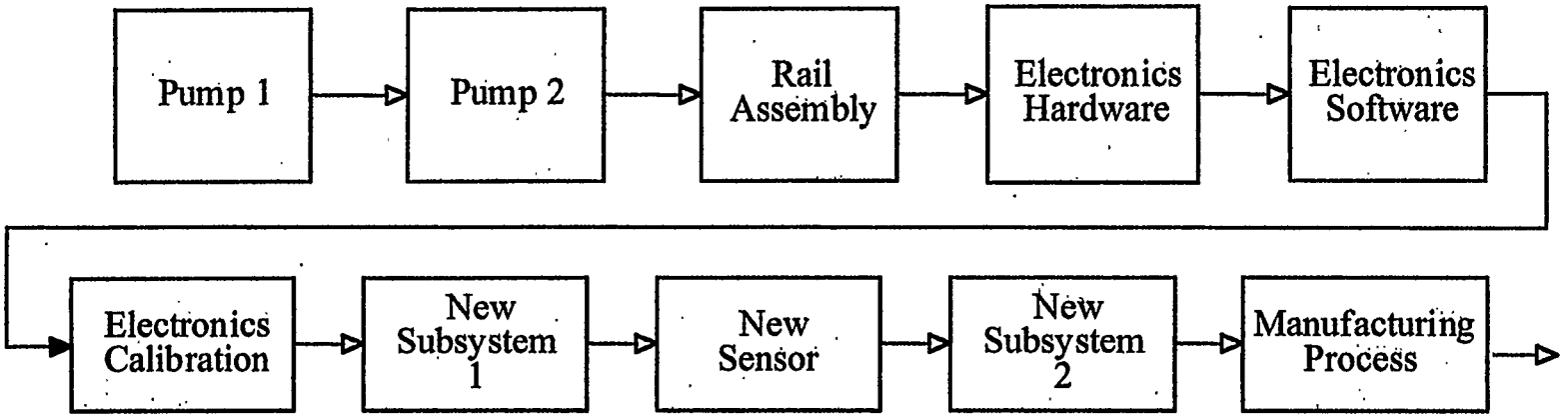
Statistical Sciences Group

Los Alamos National Laboratory

Structuring the System

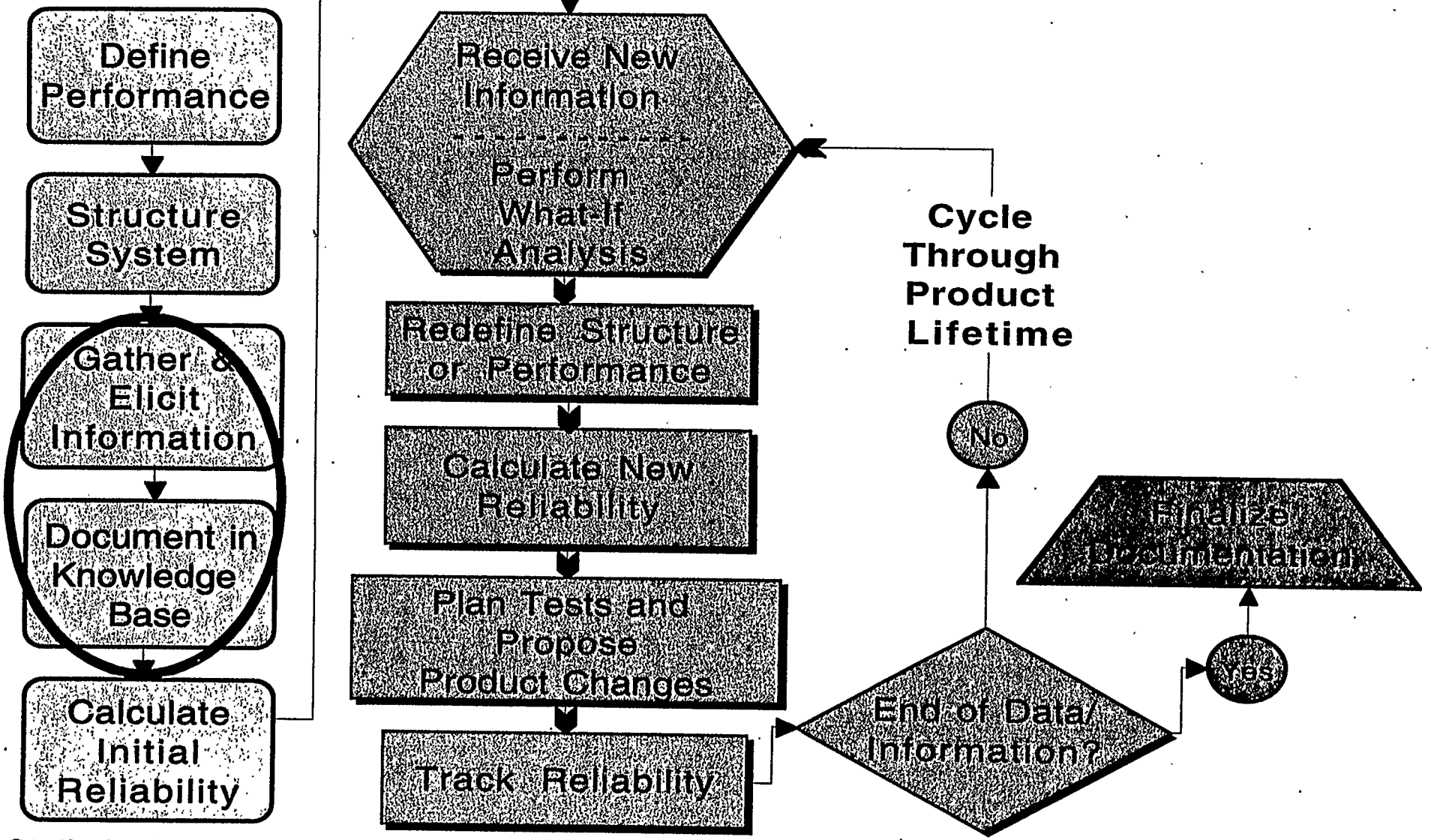
- Logic Models / Diagrams
- Event Trees
- Fault Trees
- Failure Modes and Effects
- Directed Graphs
- Process Trees
- Causal Diagrams
- Reliability Block Diagrams

Reliability Block Diagram at the Subsystem Level for a Fuel System



Each block has a diagrams of components, processes and failure mechanisms beneath it.

II. PREDICT Steps



Sources of Information / Data

- Expert judgment
- Historical test data
- Data / information of similar, relevant systems, parts, processes, etc.
- Design specifications
- Computer simulation model outputs
- Physical model / code outputs
- Prototype data / beta test data
- Test data

Documentation

The entire state of knowledge at a given time is documented in a knowledge base— a GUI in HTML.

Includes:

All knowledge/information/data—

- ***definitions,***
- ***expertise***
- ***assumptions***

All analyses

Tracking / updating

Graphs / diagrams

Results

Uses of Expert Judgment

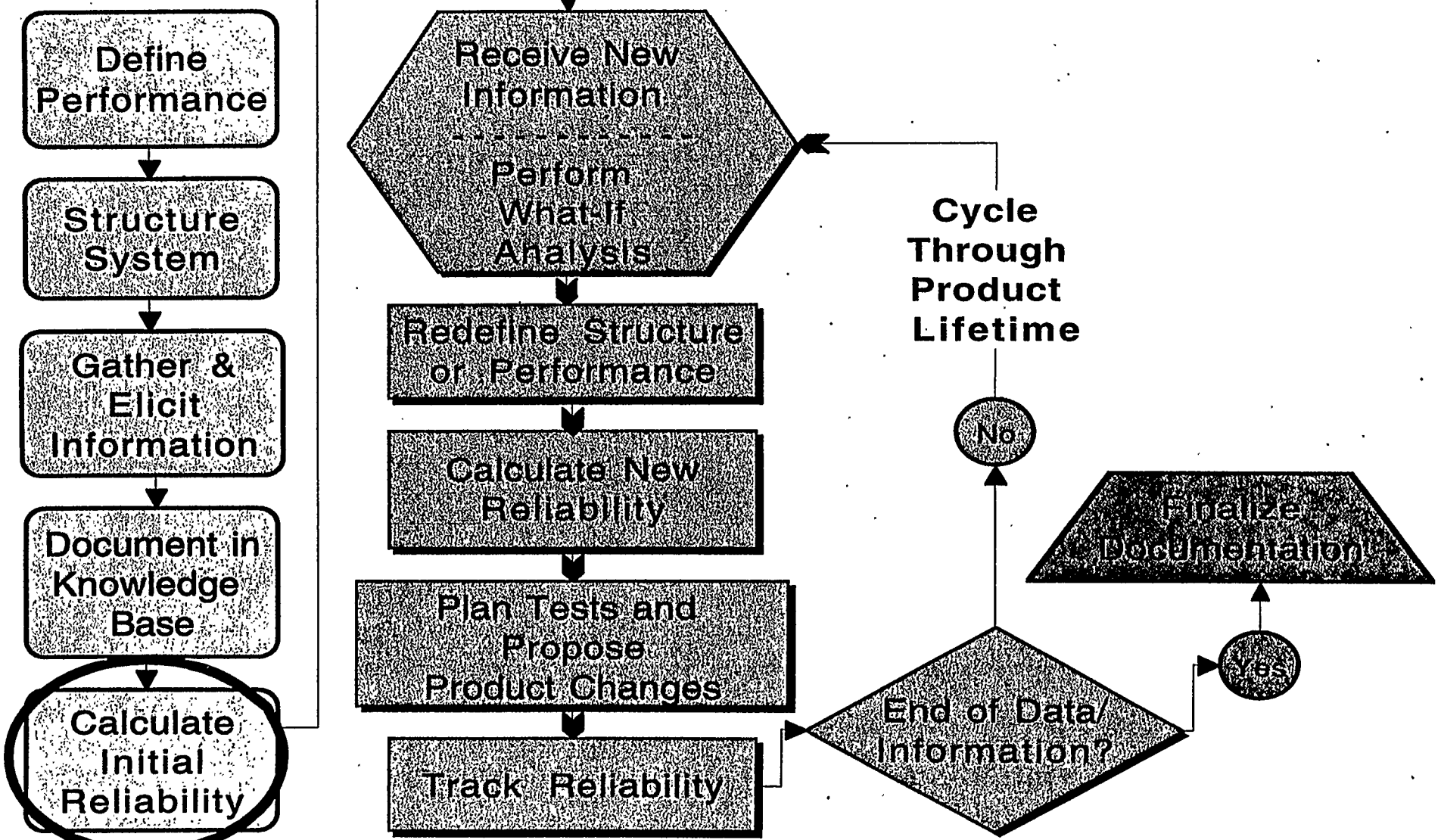
Used Implicitly for

- Defining performance
- Structuring the system
- Deciding / gathering all relevant information
- Model use / selection
- Feedback

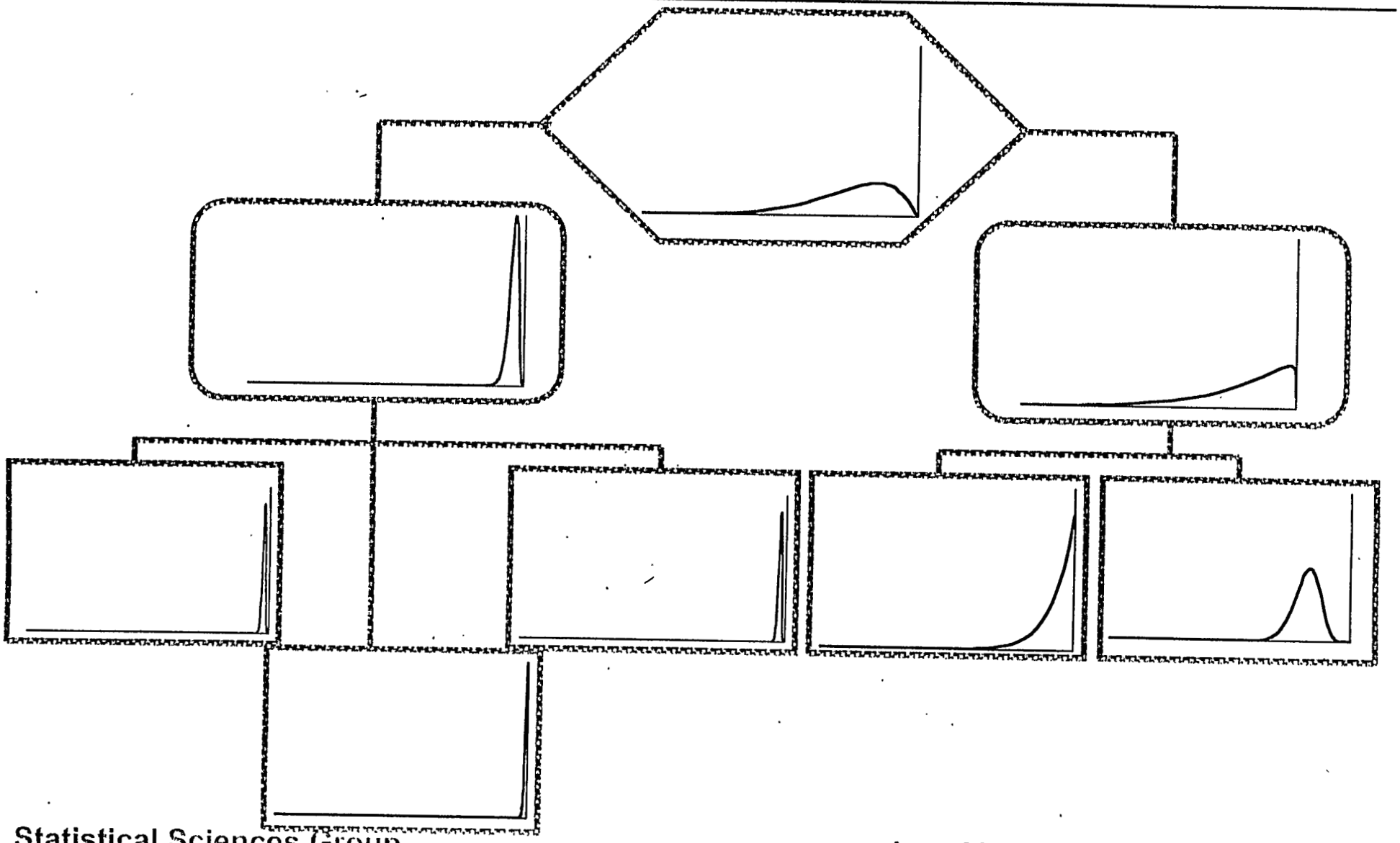
Used Explicitly for

- Estimates (data)
- Weights
- Uncertainties

II. PREDICT Steps



Reliability Distributions at all Levels



Reliability Distributions at all Levels Are Combined

Using

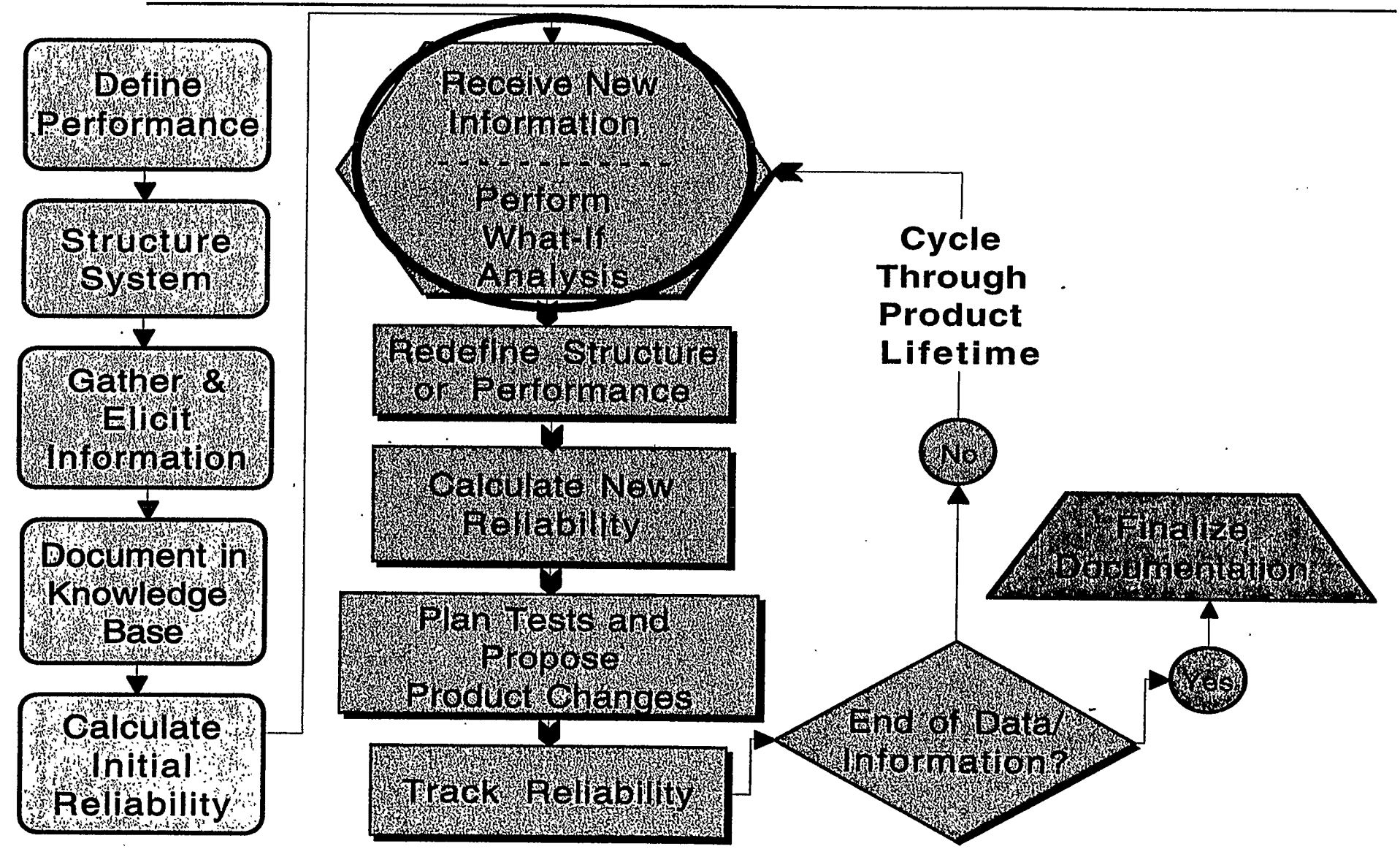
- Boolean expressions:

  Example: reliability blocks in series:

  R(*ith* level) = Π R(*j*) for *j* items below it.
- If-Then rules
- Weighting factors
- Conditional functions
- Expert Judgment
- etc.

Whatever mathematically corresponds to the specified structure of the system and the model for performance/reliability

II. PREDICT Steps



Statistical Sciences Group

Los Alamos National Laboratory

New Information / *What-if* Analysis

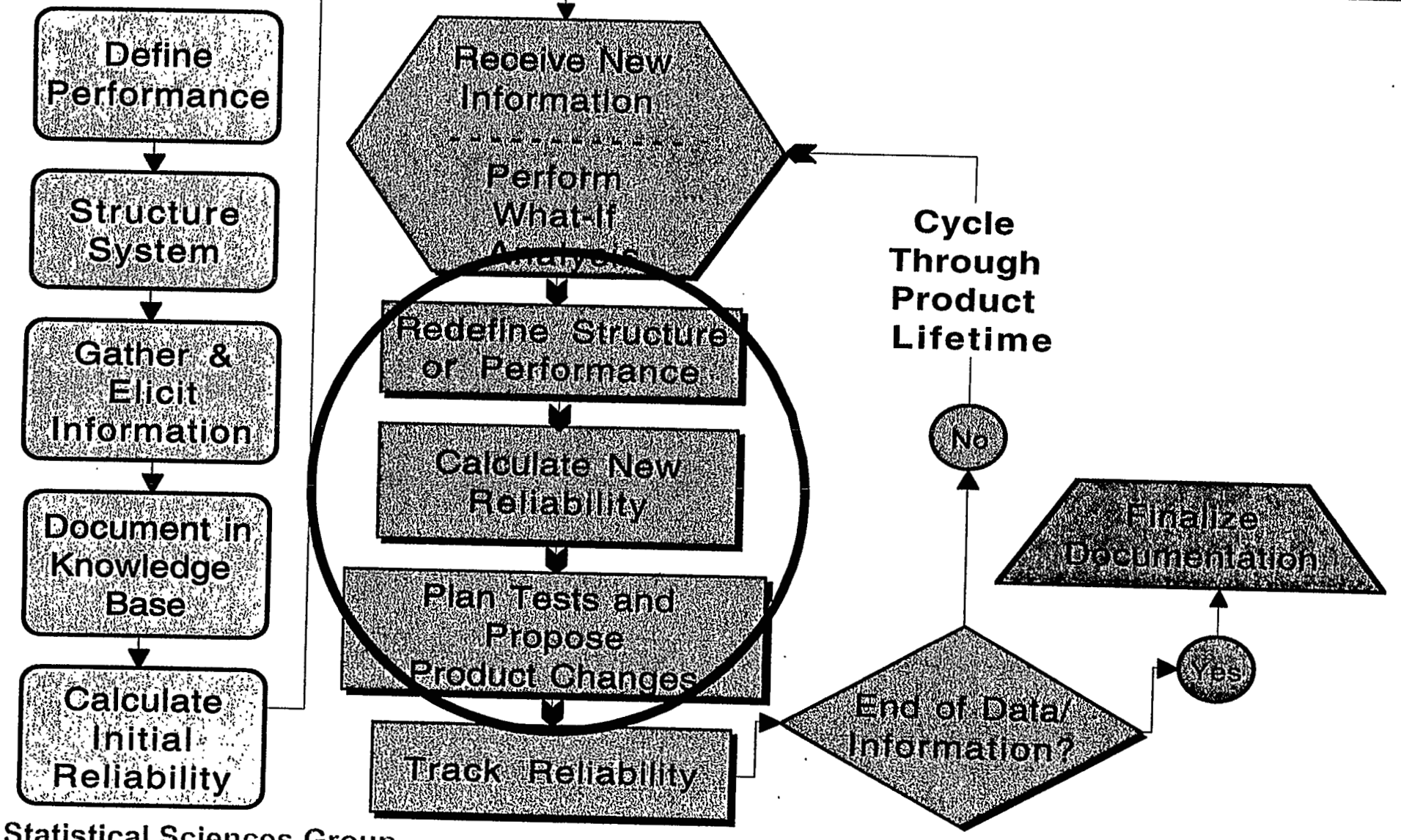
New information / data comes from:

- Design / process changes
- Suppliers
- Specifications
- Prototypes
- Tests (perhaps only on certain failure modes)
- Production
- User surveys / warranty information

*What-if*s:

- ***What if we build 40 prototypes, perform this test for this failure mode, and they all pass?***
- ***What if we change vendor contracts?***
- ***What if we use plastic on this part instead of metal?***

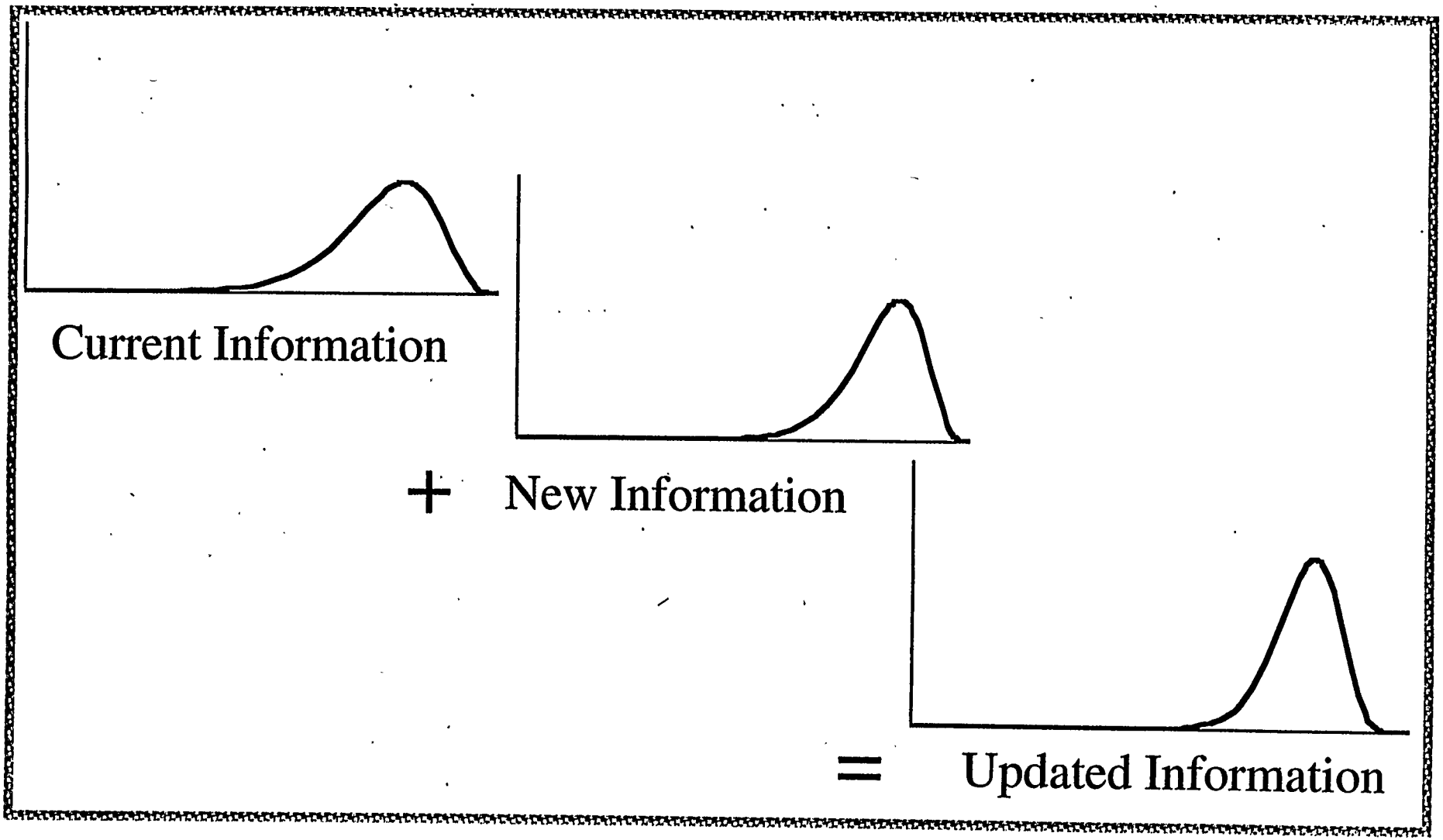
II. PREDICT Steps



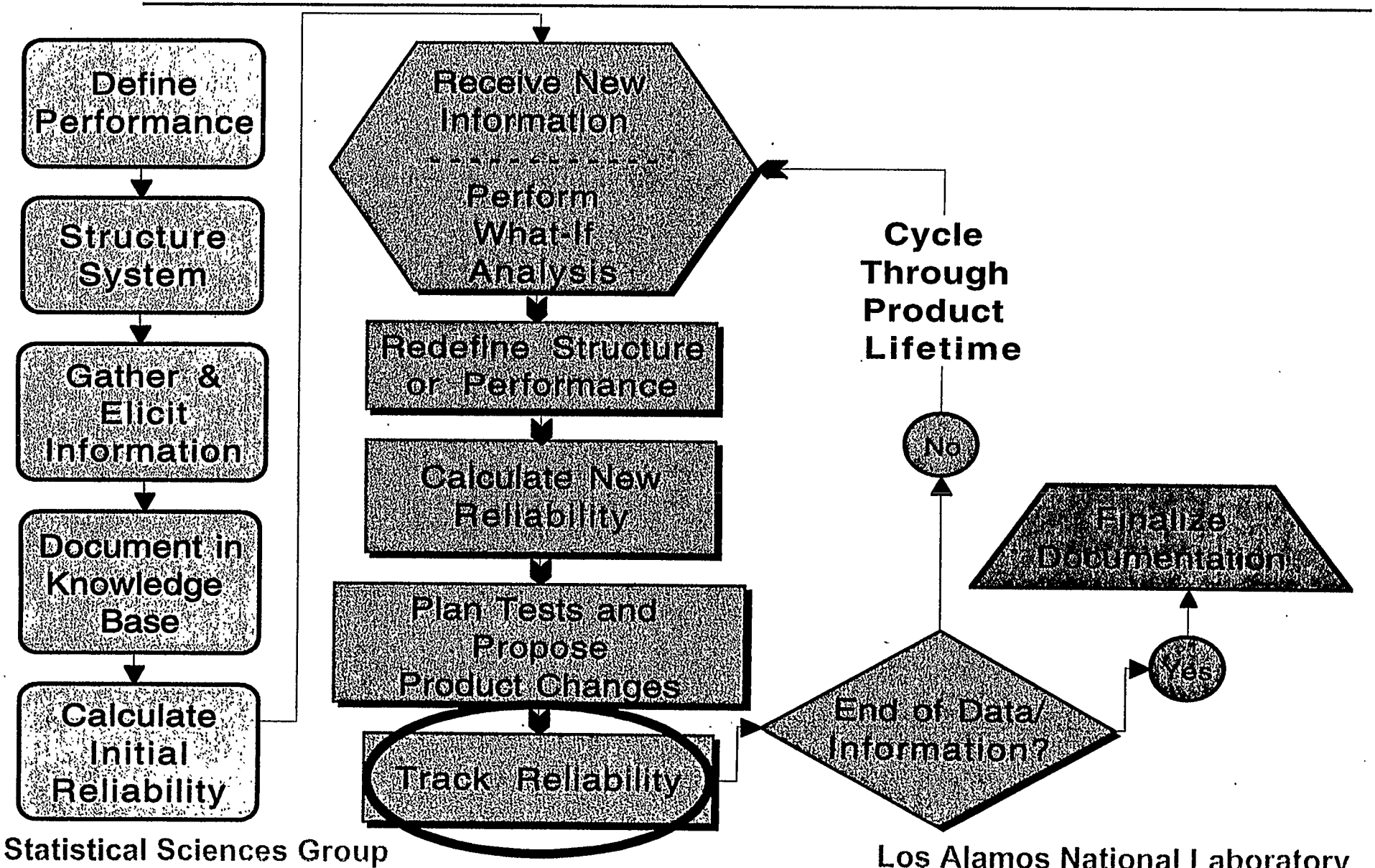
Statistical Sciences Group

Los Alamos National Laboratory

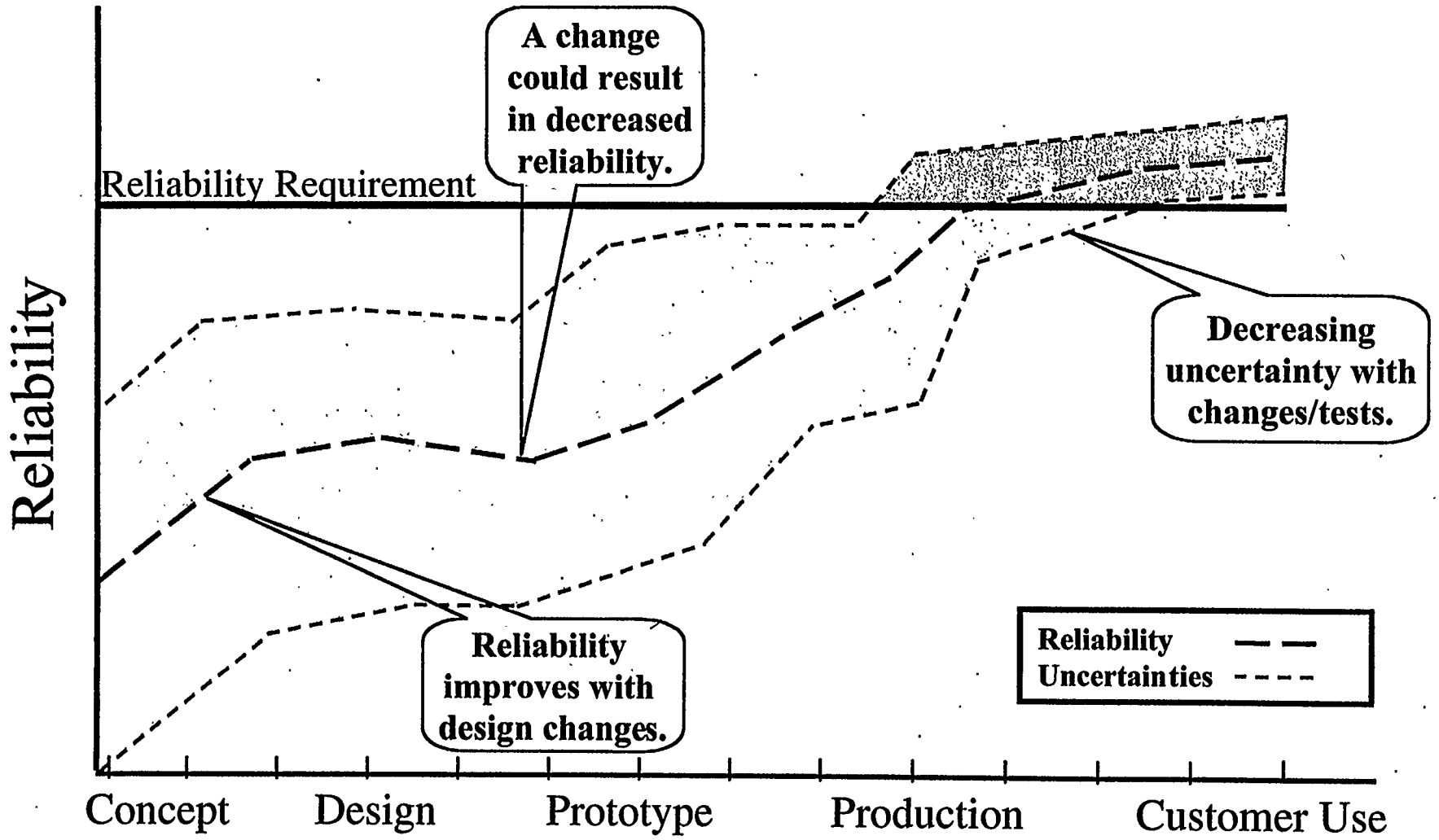
Update with New Information



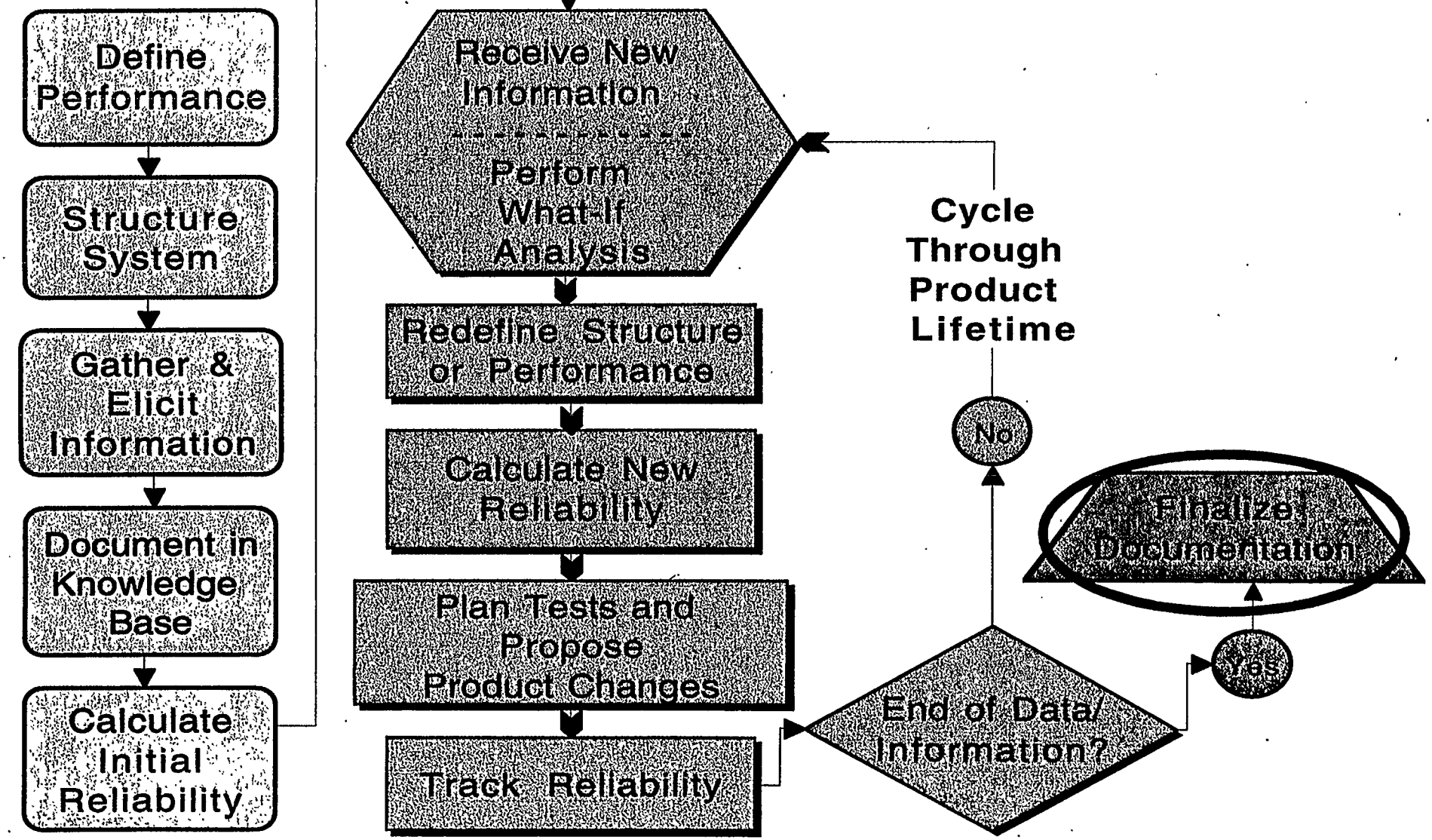
II. PREDICT Steps



Tracking Reliability Through System Lifetime



II. PREDICT Steps



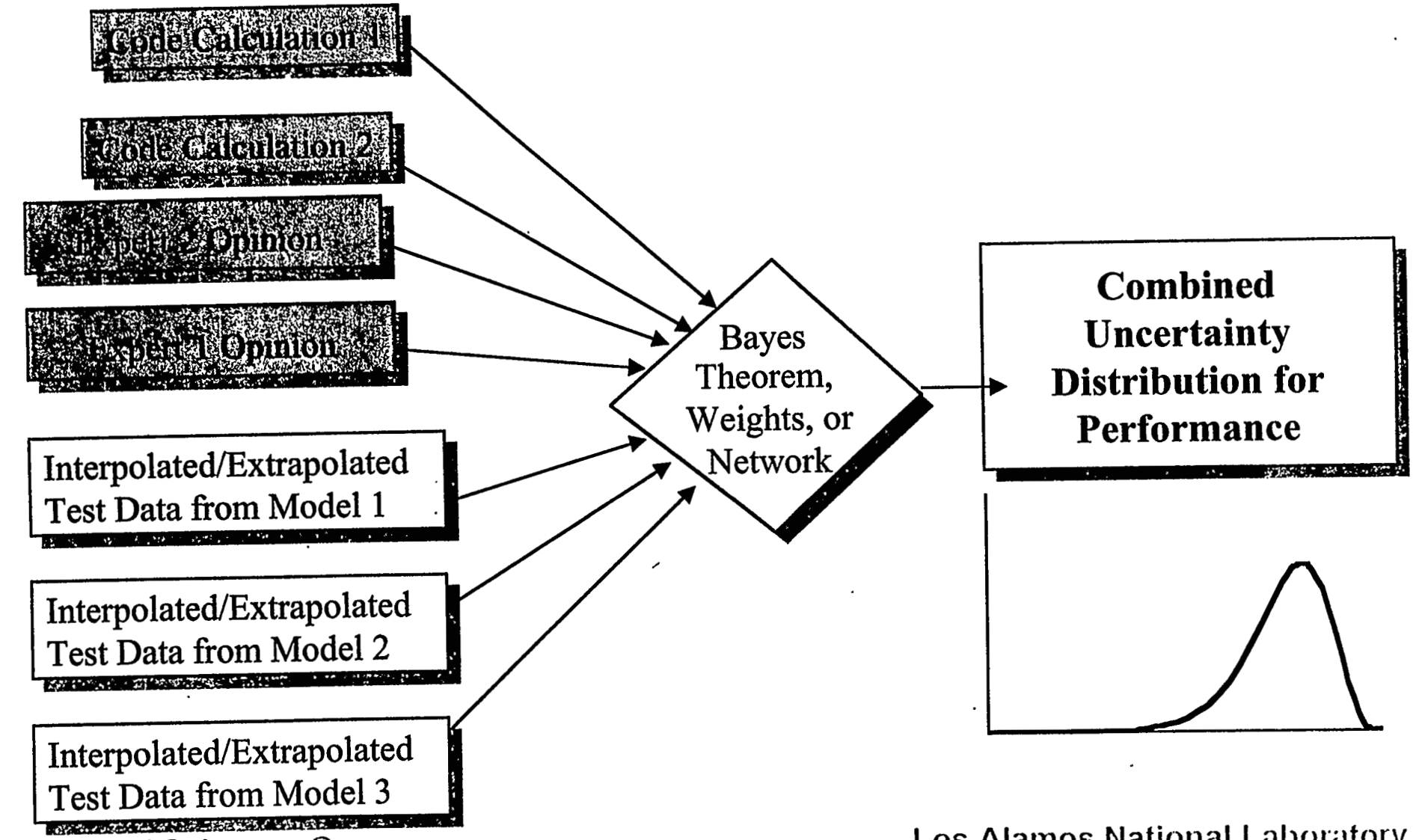
PREDICT Success & Validation

- **Delphi is implementing PREDICT on all its new concepts for their top designs for the future.**
- **Weapons program has adapted our methods.**
- **Success stories from 5 auto systems are providing validation of PREDICT**
  - early lifetime reliance on expert judgment
  - early identification of potential failure modes
  - "*what if*" planning and resource allocation
  - integrating test data as it becomes available
  - learning from studies of the systems
- **National recognition: R&D 100 Award**

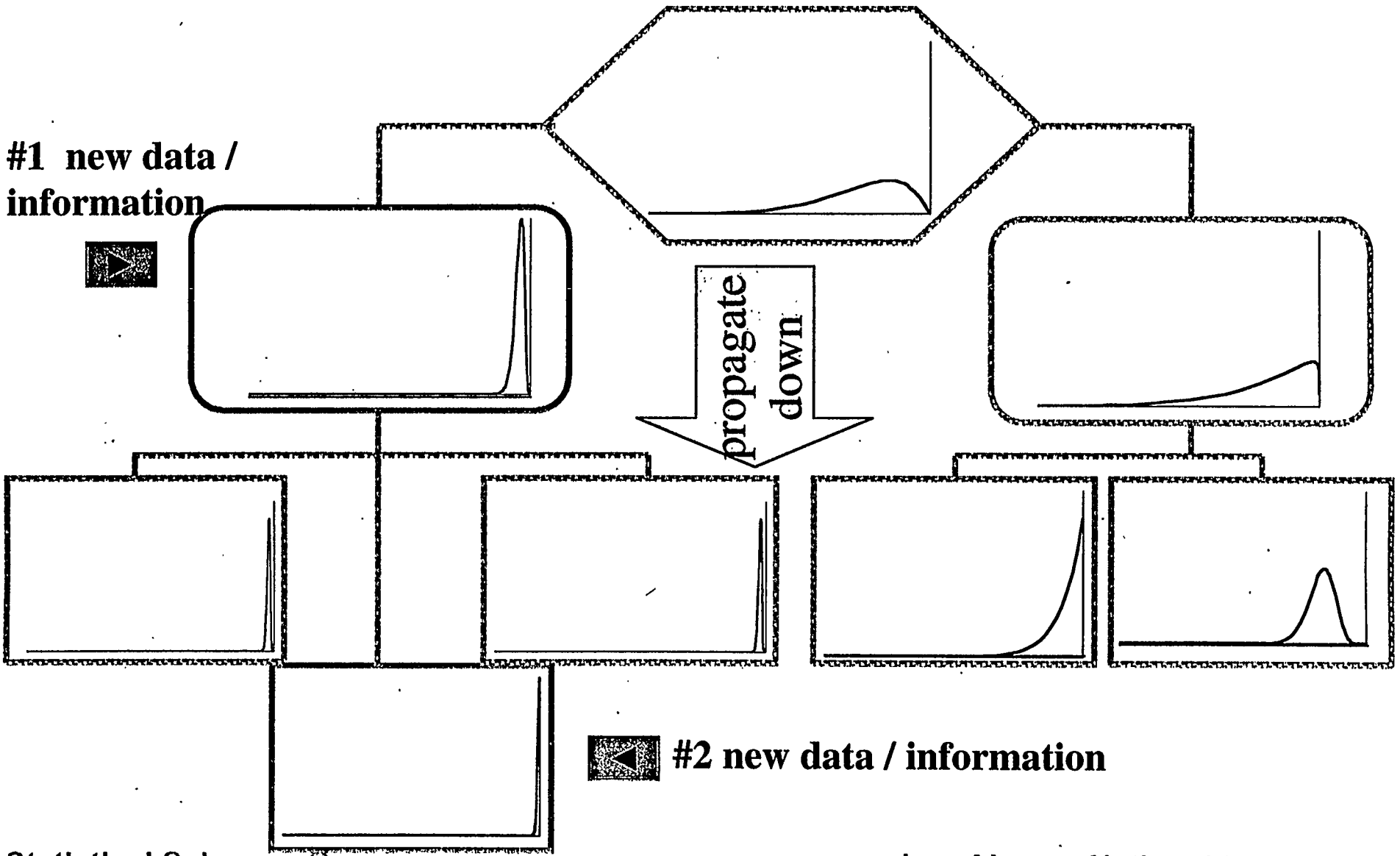
III. Analytical Issues - 1

Combination methods	Weights Rule-based (fuzzy logic) Expert resolution **Networks**
Dependence	Among experts Using same data for different models / simulation
Level of Detail	**Information enters at various levels**

Combining Multiple Information Sources at the Subsystem Level



Information / Data Enters at Any Level



Analytical Issues - 2

Conditioning

- Reliability / Performance definition
- **Expert Judgment**
- Models and their parameters
- Data / Information

These conditions must be understood, documented and become an integral part of the analyses.

Metrics for Comparing

- Models
- Combination methods
- Assumptions
- Data / Information
- **Distributions**

Expert Judgment as Data

Expert judgment shares traits with data from tests, experiments, or physical observations. It:

- Is affected by the process of gathering it (principles of experimental design).
- Has uncertainty, which can be characterized and subsequently analyzed.
- Can be conditioned on various factors, such as
  the phrasing of the question,
  the information the experts considered,
  the experts methods of solving the problem, and
  the experts assumptions.
- Can be combined with other data.

Eliciting and Analyzing Expert Judgment

Formal Elicitation & Documentation Using a Designed Worksheet

Worksheet for Component Performance

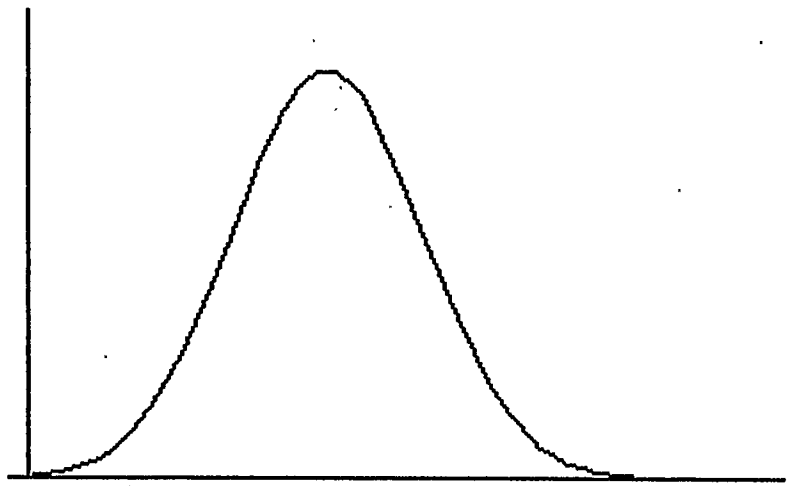
Subject Matter Expert Jane Doe
Component New sensor
Date 11/30/99

To obtain an overall reliability estimate of the fuel system, we are asking you to provide your expert judgment regarding estimates of the reliability of its components and sub systems. See the attached reliability block diagram / fault trees for the components and subsystems that your group will be concemed with. .....

1c) What is the incidence of the key failure modes? That is, if a failure occurs, how would you apportion the occurrence of all these failure modes among the entire component listing (percentages)?

Failure Mode	Incidence (%)
Signal not received	20%
....	
Minimum	0.1 IPTV
Most Likely	0.5IPTV
Worst	1.0 IPTV

Provides Performance Estimates & Uncertainties for Components, Processes, and Subsystems



These results are fed back to the experts for review.

Metrics for Combining & Comparing Distributions

- *Quantiles*
- *Density estimators (kernels)*
- *Relative Distributions*
- *Information content measures*

For example, Jeffreys' J measure— the basis of Kullback-Leibler information:

$$J = \int \log \frac{dP2}{dP1} d(P2 - P1)$$

Analytical Issues - 3

Uncertainties in:

- Weights
- Expert judgment estimates
- Models
- Model parameters
- Data / Information
- Conversions

Quantification using:

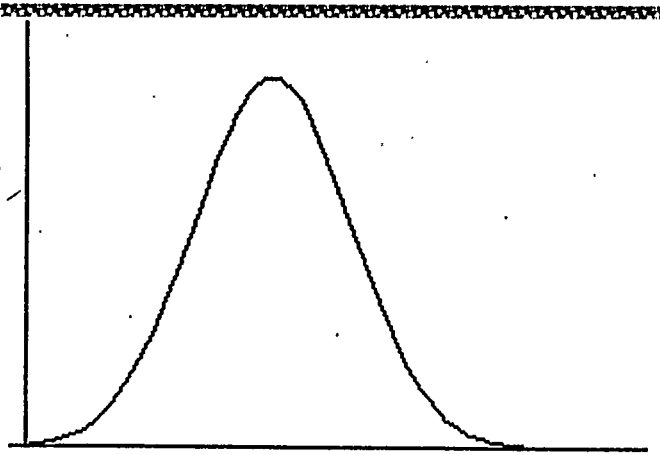
- Scores / Ranks
- Pairwise comparisons
- Distributions
- Transformations
- Expert judgment
- Rules (fuzzy logic)

Uncertainty Distributions

Functions used to represent uncertainties:

- **Probability Distribution Functions**
- **Cumulative Distribution Functions**
- **Likelihoods**
- **Membership Functions**
- **Possibility Distributions**

density, probability, likelihood, possibility, membership, etc. associated with each value



values of the uncertain variable

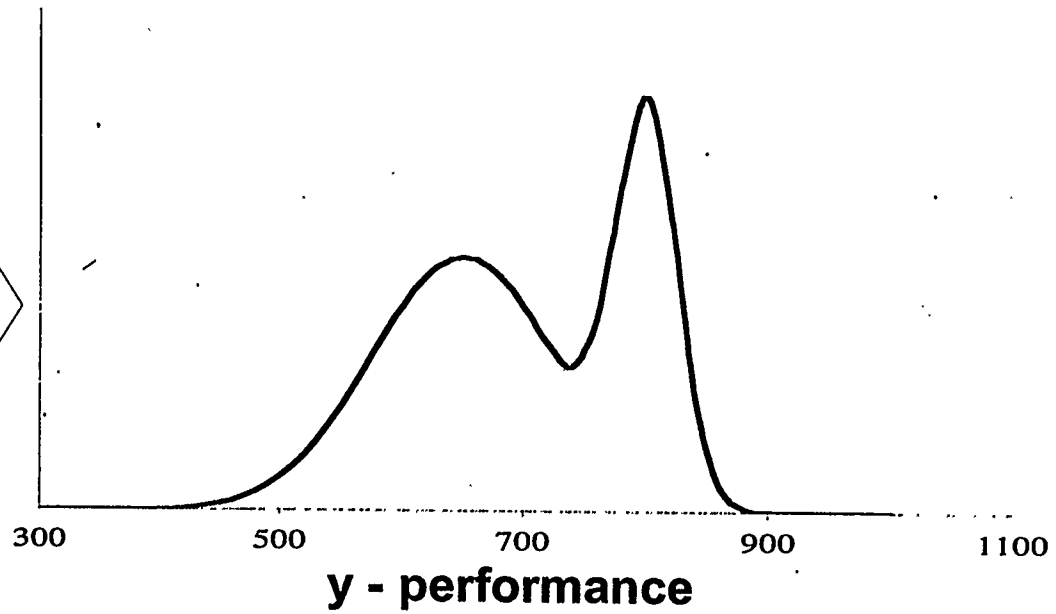
Quantification Using Fuzzy (Logic) Control Systems

Methods that transform qualitative descriptions of system/component condition (x) and performance (y) into numeric, ***fuzzy***, membership functions.

If-then (qualitative) rules map

Condition (x) into **Performance (y)**

The result is an uncertainty distribution of y, for each value of x, in the condition space.



Application of Fuzzy Control Systems

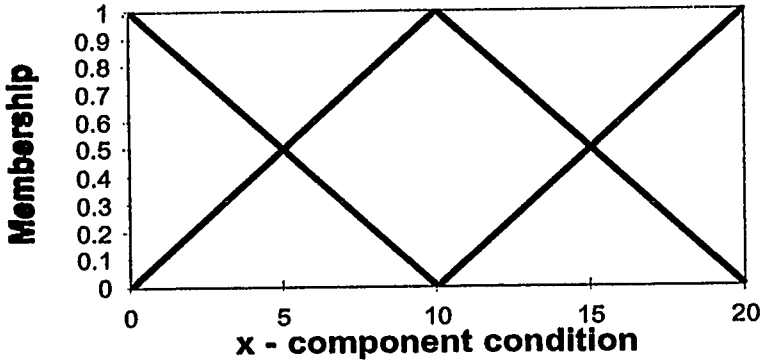
Condition

Performance

Fuzzy condition sets

{good, nominal, poor}

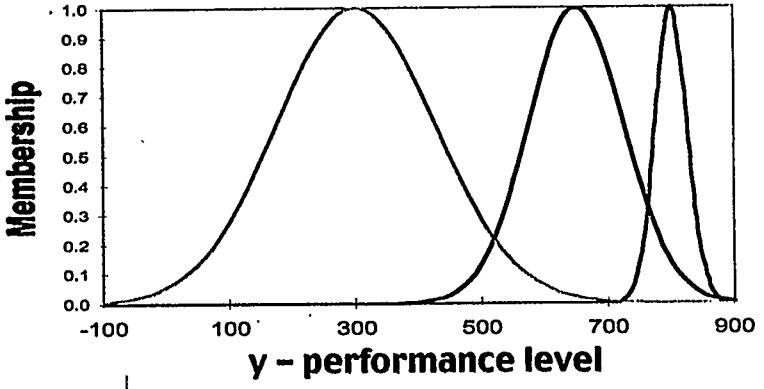
Condition membership functions



Fuzzy performance sets

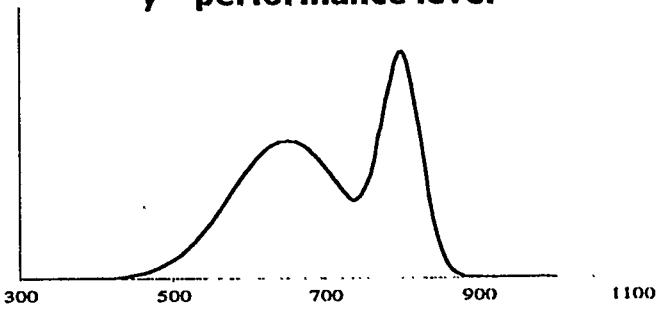
{normal, marginal, bad }

Performance membership functions



For $x = 4.0$

If-Then rules



Condition Membership Functions (MFs)

Simple example—one component, subject to wear, which degrades the system performance.

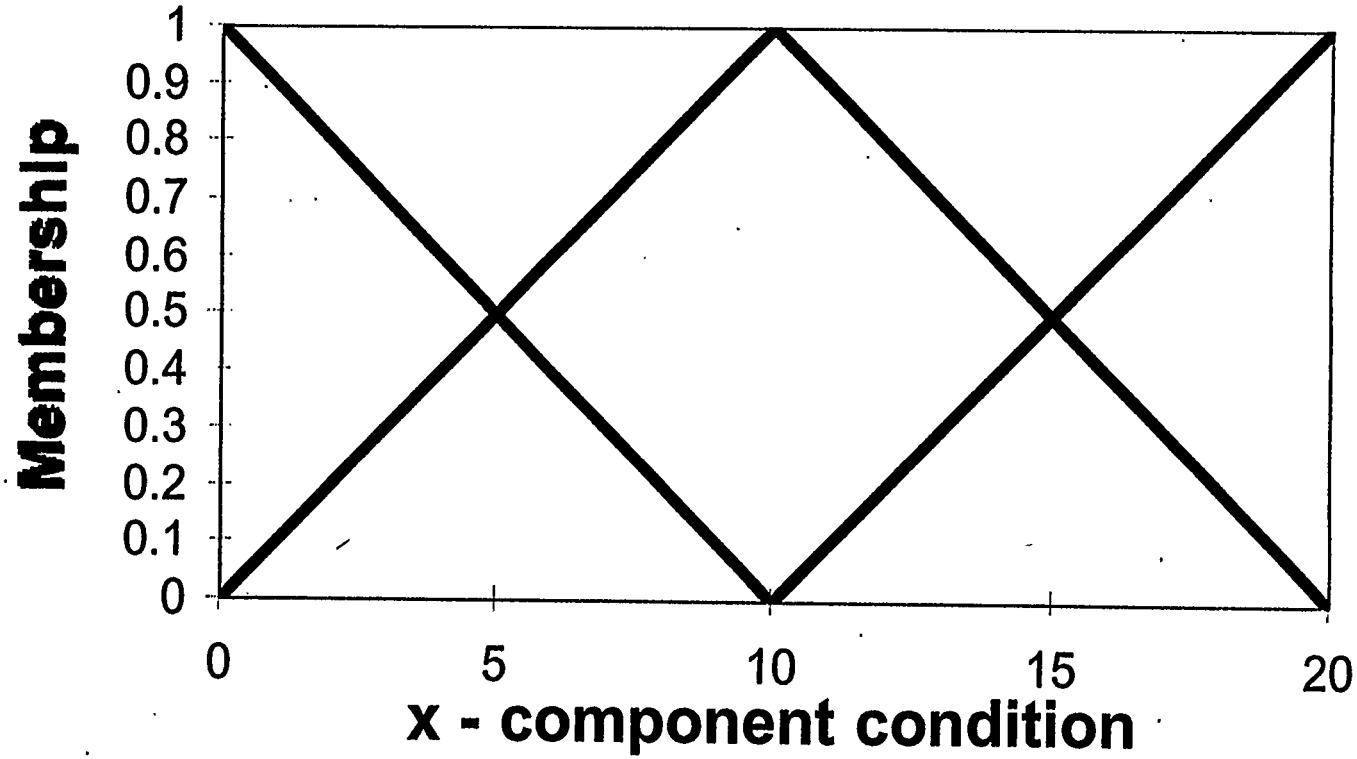
Membership functions (input, condition)

A —> good

B —> nominal

C —> poor

Assume: triangular



Performance Membership Functions (MFs)

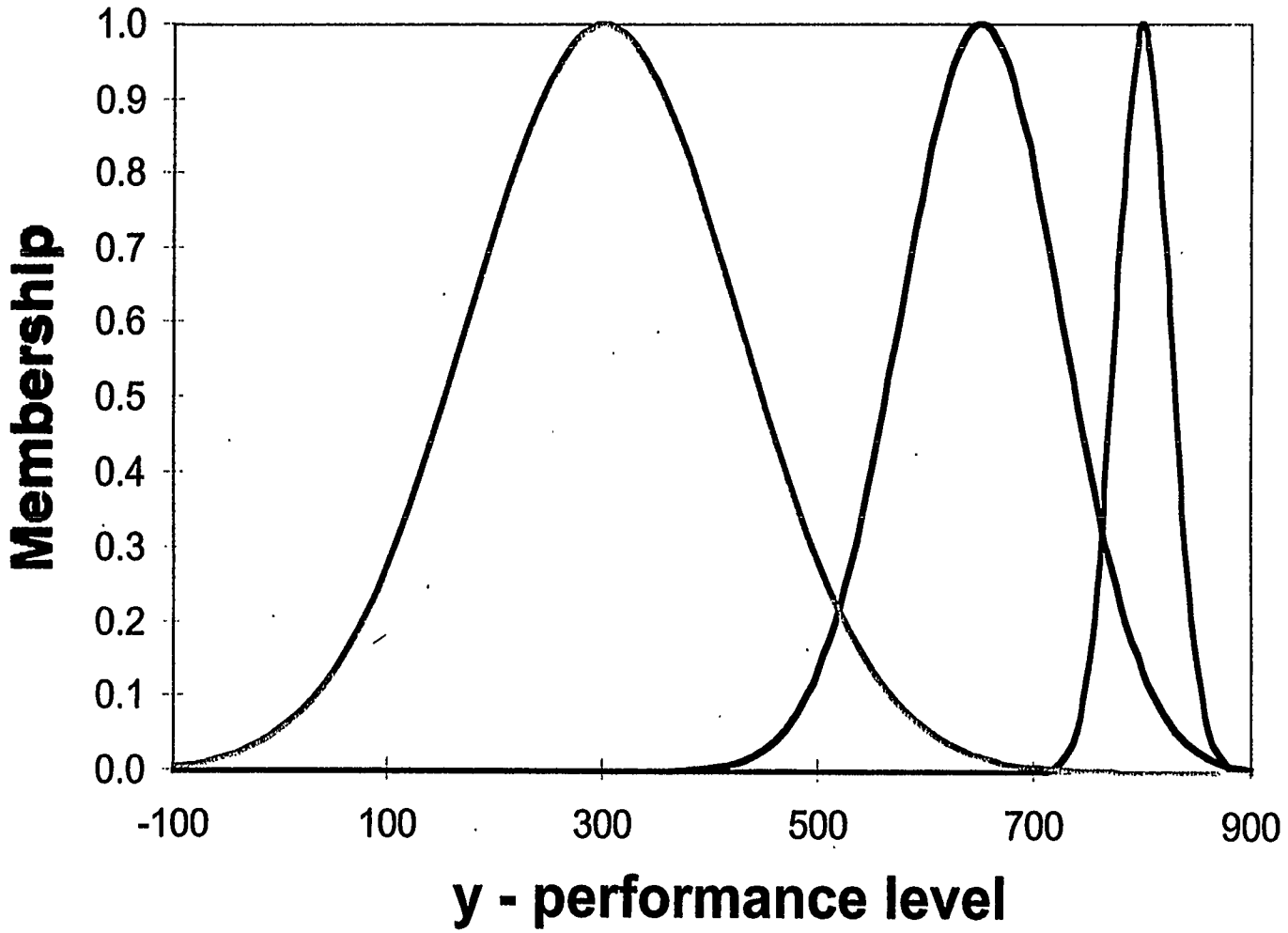
Membership functions (for output, performance)

a—> normal

b —>marginal

c —> bad

Assume: scaled (0 to 1) normals



If-Then Rules Map Condition MFs into Performance MFs

If		***Then***
Condition (x)	⇨	**Performance (y)**

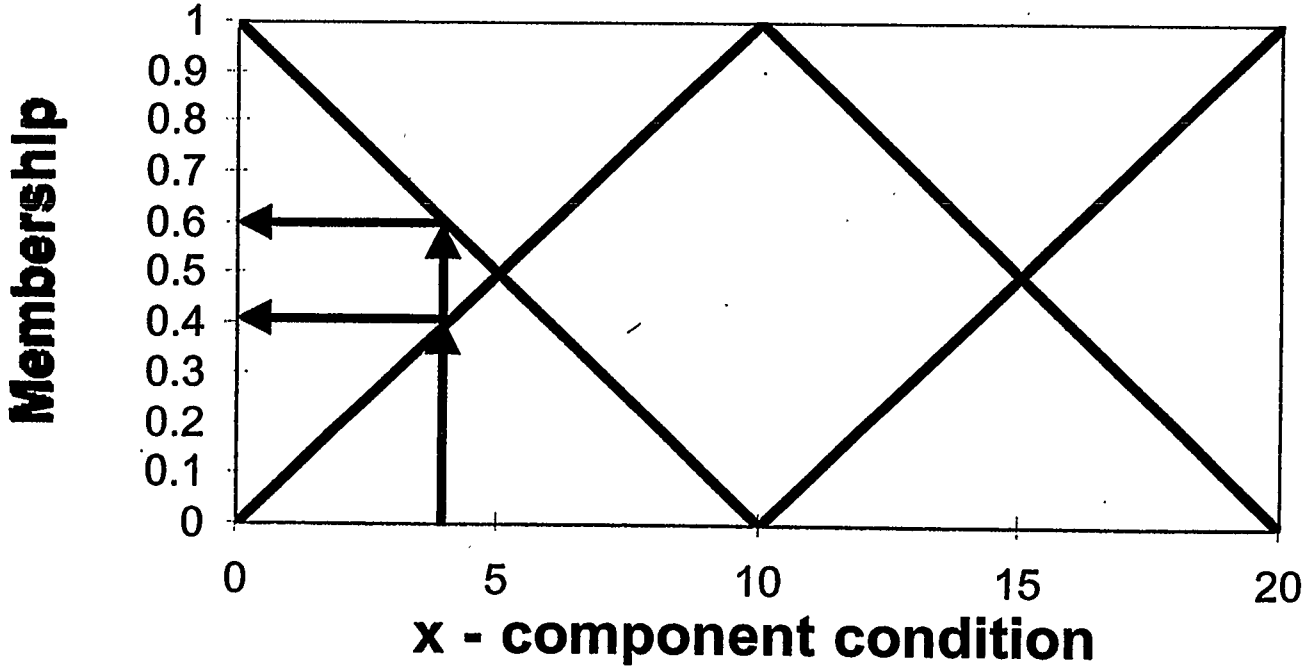
A=good	***a=normal***
B=nominal	***b=marginal***
C=poor	***c=bad***

If component condition (x) is *nominal* (in wear), ***then*** system performance (y) is *marginal.*

Mapping Condition into Performance for 1 value of x

If component condition is x=4.0 , then x has membership of 0.6 in *A* and 0.4 in *B*.

Using the *If-Then* rules, map into performance set *a* with a weight of 0.6 and into performance set *b* with a weight of 0.4 .

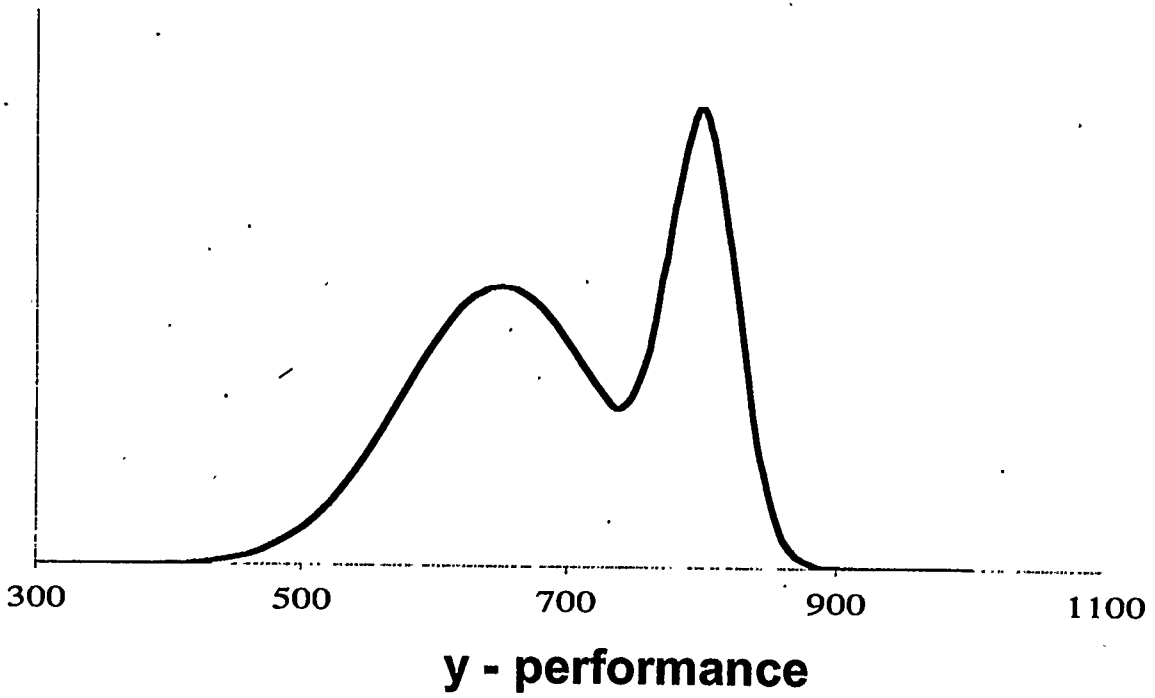


Determining the Uncertainty Distribution for x=4.0

For component condition x=4.0: **The *fuzzy combination* combines performance sets *a* and *b* with weights of 0.6 and 0.4. This combined function can serve as an uncertainty distribution of performance given *x*, *f*(*y*|*x*).**

In fuzzy control systems, this is reduced to a *centroid* value (*defuzzification*).

Problem: can be bimodal.

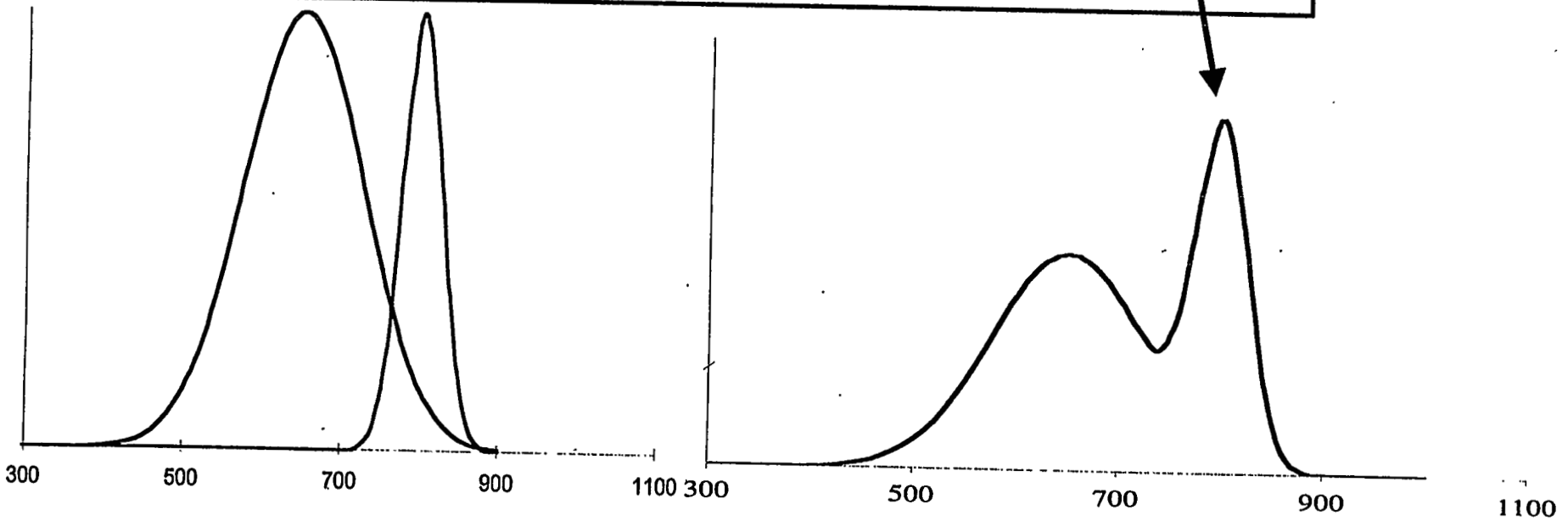


Fuzzy Combination Method for x=4.0

Combine Performance membership functions as:

0.6 (membership function a—> normal) +

0.4 (membership function b—> marginal =

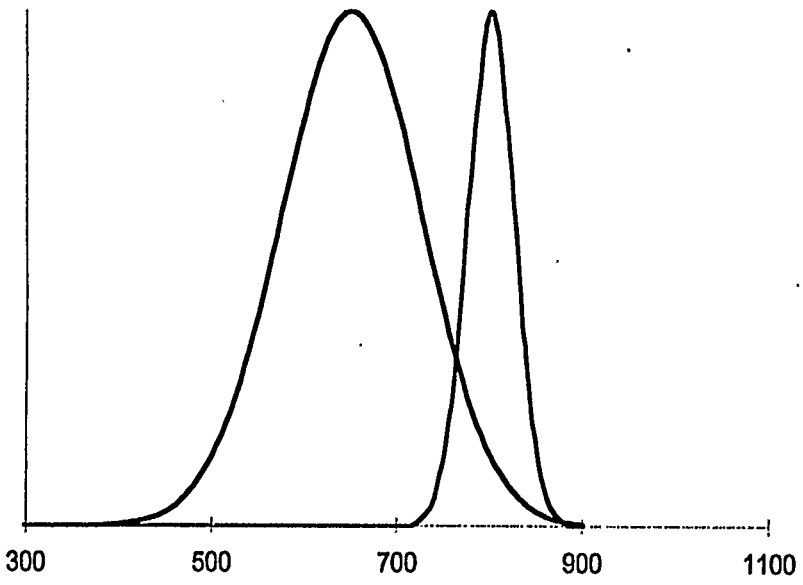


Alternative Method for the Uncertainty Distribution for x=4.0

Performance MFs

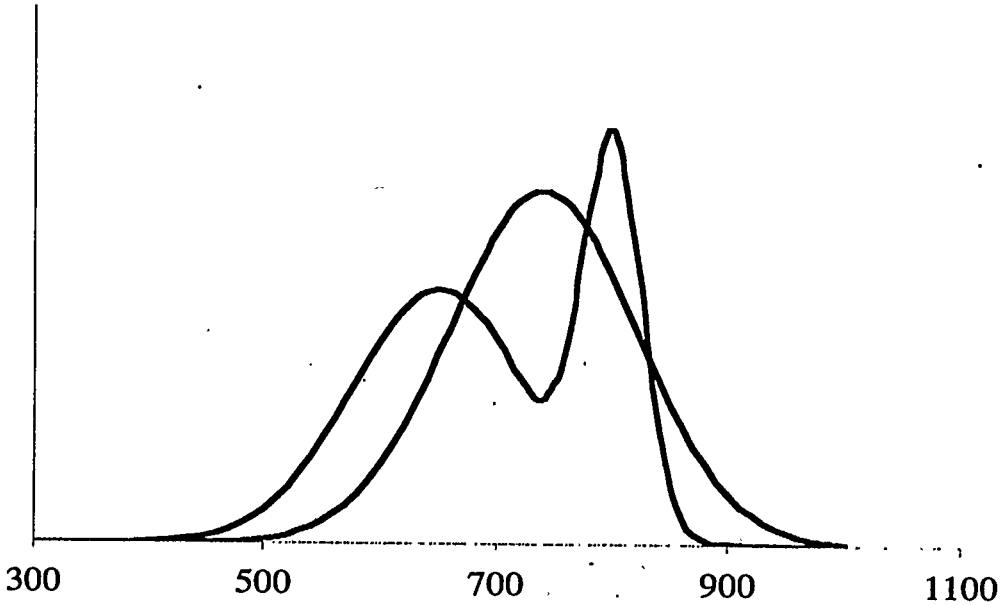
a—> *normal*

b—> *marginal*



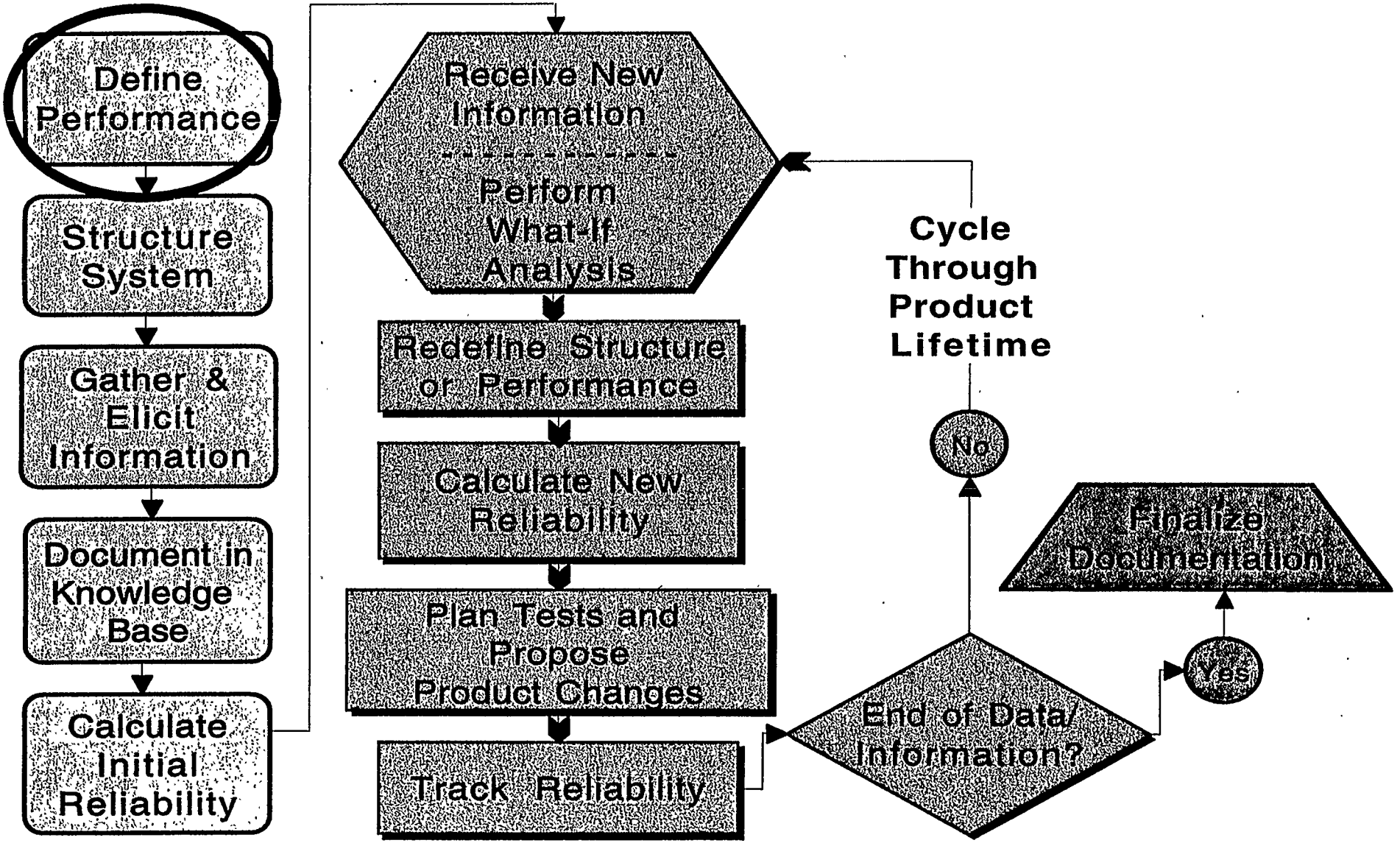
y - performance

Combinations
fuzzy
linear



y - performance

IV. Construct an Example in Software Reliability

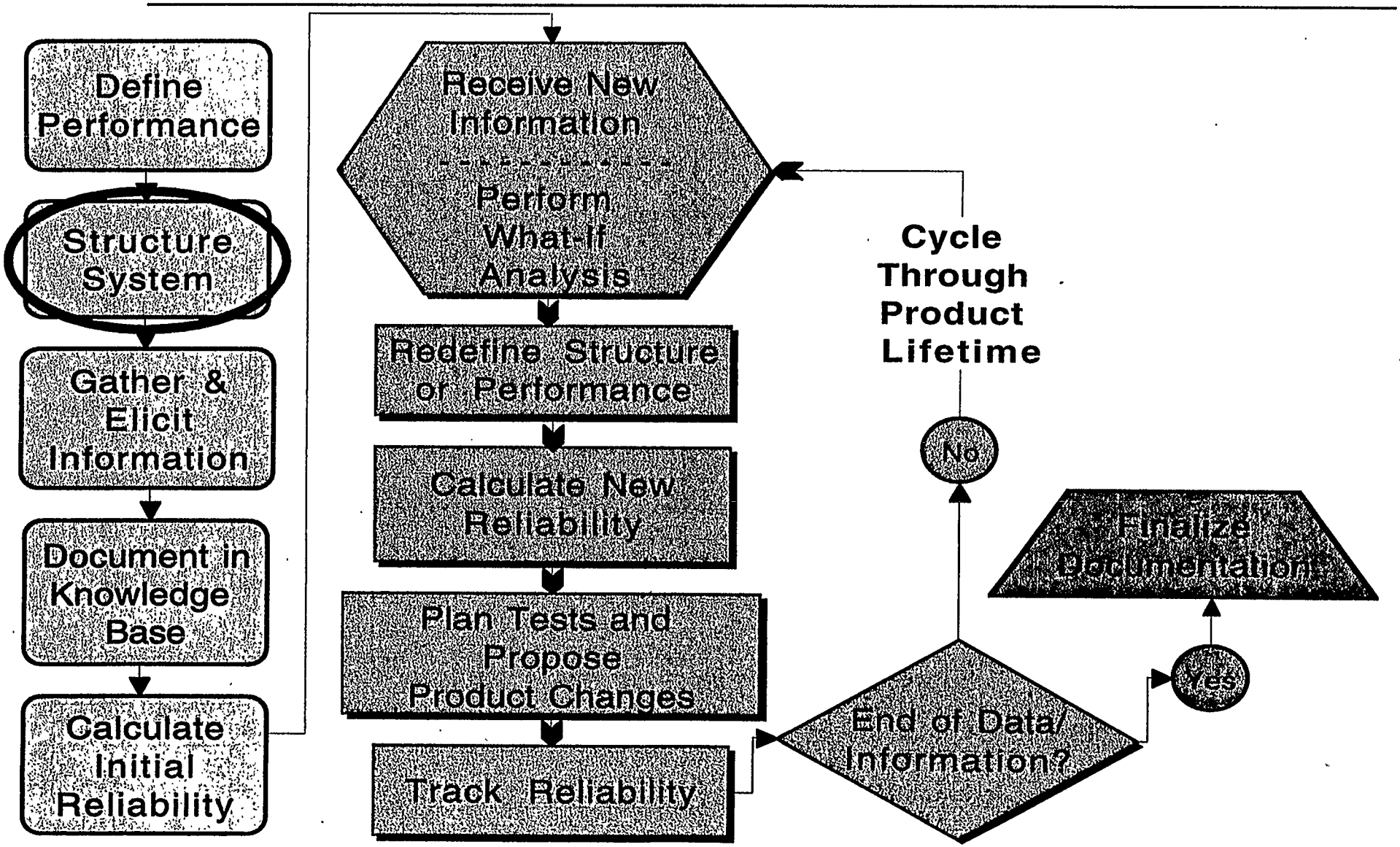


IV. Construct an Example

Define Performance

Is it probability of a failure/success?
What is a model for reliability/performance?
Do experts "think" in terms of probability?
Do we have rare events?
Do we have "bathtub" curve?
Are there failures per unit operating time?
Is it number of failures per total units made?
Operations per unit time?
Errors per LOC?
Is it mean time to failure?
Is it failures for total time on test?
Do we need membership functions because it's too difficult to quantify?

IV. Construct an Example



IV. Construct an Example

Structure System

Causal relationships among parts?
Failure modes?
Dependencies?
Parts? Processes? Other?

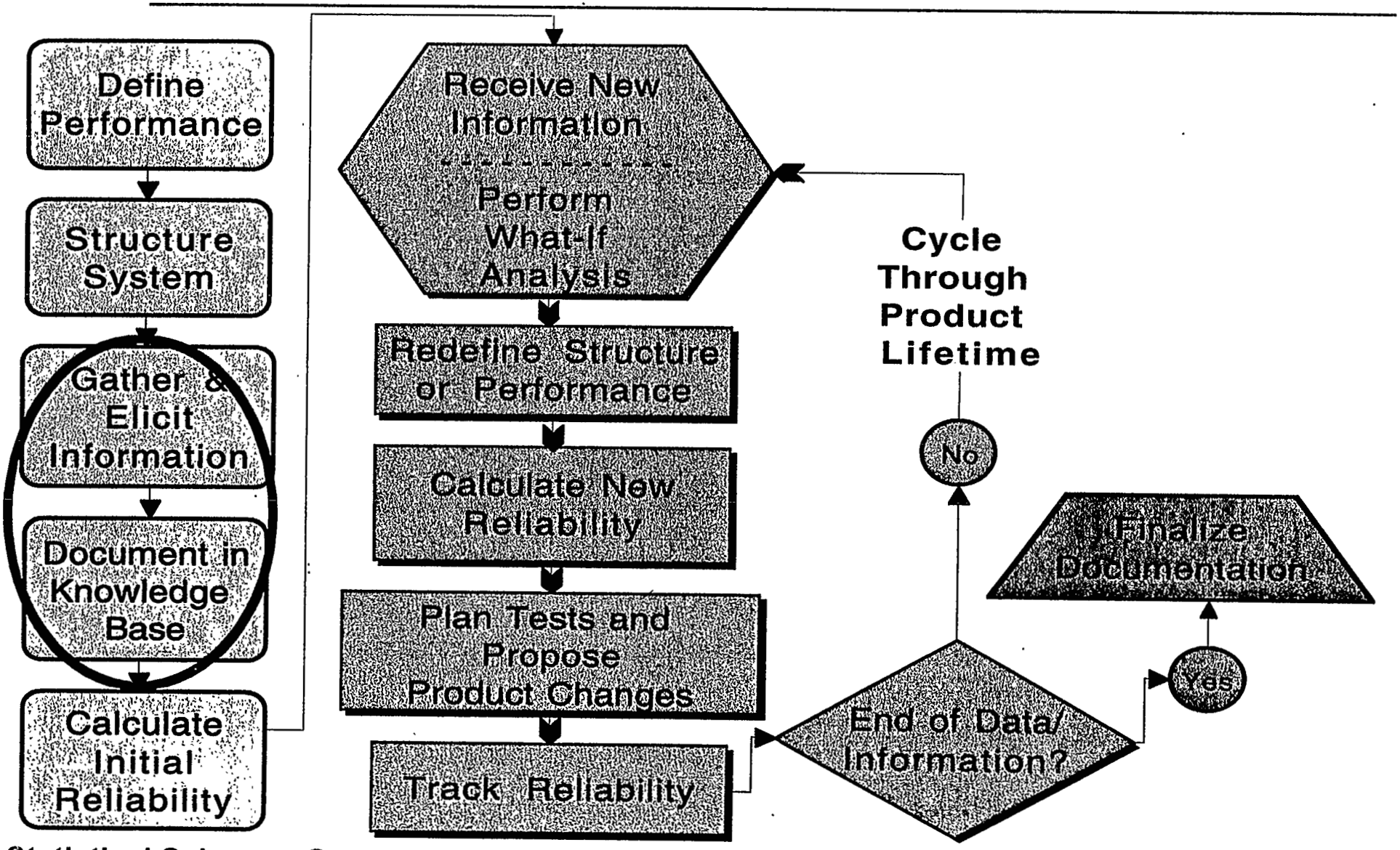
How are pieces/processes put together
—macros, subroutines, cross platforms—
from the past? Is everything newly formed?

What are the operating environments?

Can we diagram the entire "system" logically?

Does the reliability/performance model change?

IV. Construct an Example



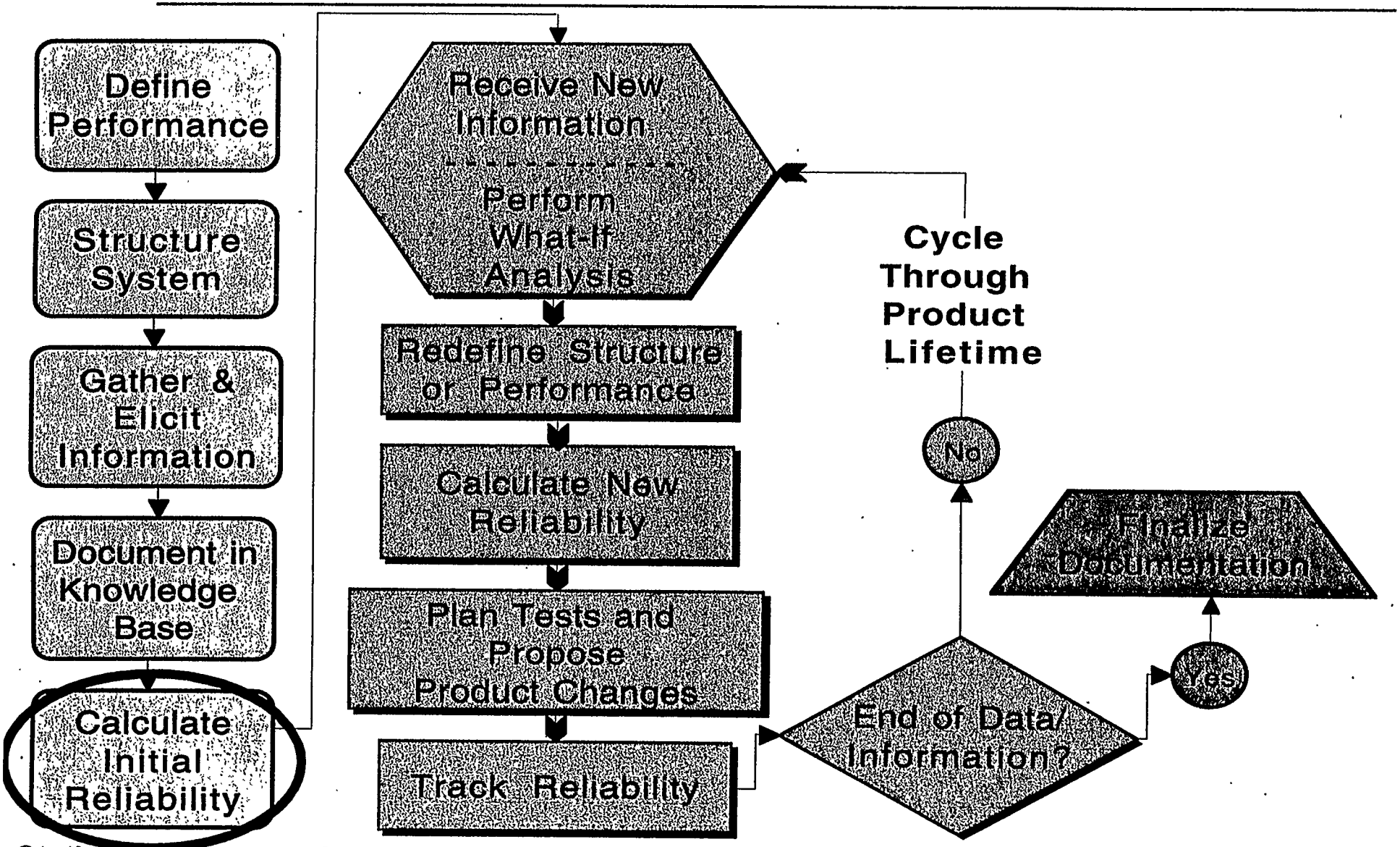
IV. Construct an Example

Gather Information

What data/information is available?

Expert judgment
Beta trials
Old pieces used
Specifications
History, track record
Test data
Other

IV. Construct an Example



IV. Construct an Example

Calculate Initial Reliability

Form uncertainty distributions for all information, model parameters, conversion factors, etc.

Populate these into the system structure.

Combine using model of the structure.

What can Analysts Do Without Data?

Gather all we know,
Determine how well we know it (uncertainties),
and
Combine all that information
to estimate performance,
Using statistical methods and philosophy.

UCRL-00xxx

Scientific Simulation and Software Project Management



Presented at Quality Forum 2000

April 18-19, 2000

Dr. Evi Dube

B Group Leader/Co-Principle Investigator, ALE3D

Lawrence Livermore National Laboratory

Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-Eng-48

Lawrence Livermore National Laboratory, P.O. Box 808, Livermore, CA 94551

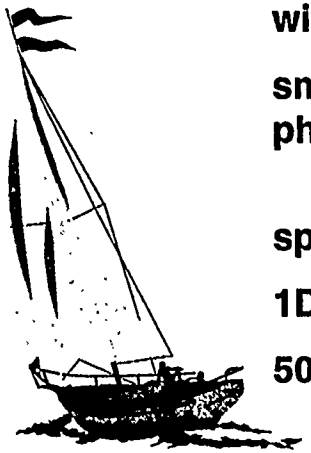
The software systems have changed over time.

- Software systems have evolved from a collection of hero codes to a few large simulation projects that need to be managed in a much more structured and organized manner.
- The errors between the two different eras of software development have a different effect:
  - Previously, software errors were caught with experiments.
  - Today, we do not have that option.

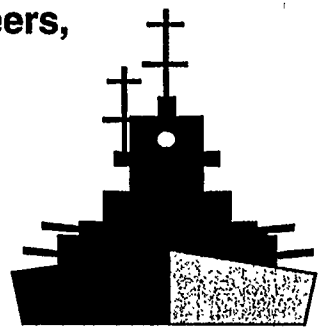


The Simulation teams have risen to the challenge of managing software systems in this new environment.

Let us contrast the past and present simulation development environments.



PAST	PRESENT
Lead code developer familiar with ALL of code	Code leader not able to be knowledgeable of entire code
small teams with a few physicists, and one CS person	Large, multidisciplinary teams - physicists, many CS, engineers, chemists
spaghetti code	structured code systems
1D, 2D	3D
50K elements large problem	10's of Millions of elements routinely
ran on one consistent machine - Cray (equivalent to today's laptop) - in serial	many parallel platforms, portable across machines and time
Systems of linked codes	Full system, high fidelity code systems
small data sets (<1 Mbyte)	lots of data (100's of Gbytes)



A good Software Development Process helps us manage our current complex code development environment.

The Software Development Process defines how we design and build our simulation codes.



- There are two parts to the Software Development Process:
  - Software Engineering
    - how are the pieces put together, or engineered?
  - Software Structure
    - how is the software designed?

We are continually exploring new technologies in the area of Software Development.

The Software Engineering Process defines how we build our simulation codes.



- What are the formal and informal steps we use in
  - requirements management
  - design
  - implementation
  - verification
  - validation

  of our code systems?



- What are the controls we use to ensure the correctness of the codes when making changes or enhancements?

The critical mission of these software packages demands a much more structured and ordered environment then in the past

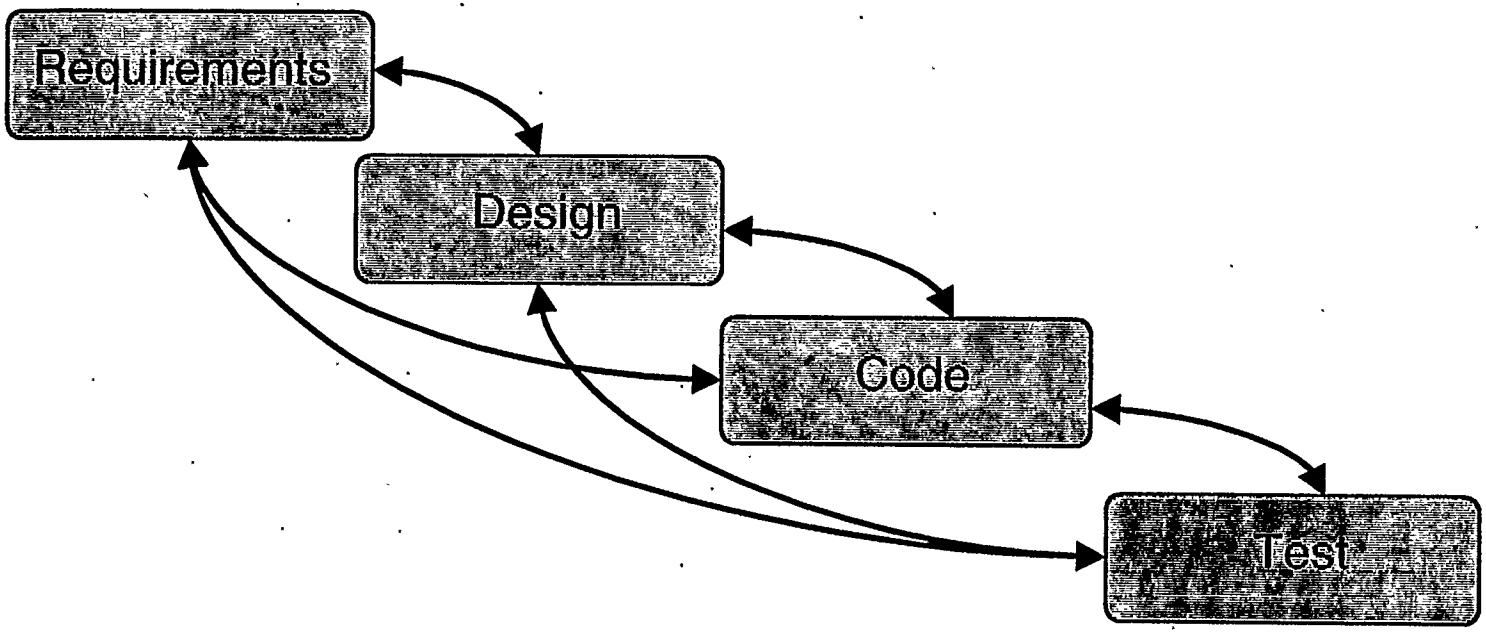
We have a unique Code Development environment.



- The requirements for the simulation codes continually change, and are driven by the types of problems they must solve, and the physical effects they must model.
- The operational nature of the requirements demands extremely tight coupling between the simulation team and the users.
- There is no unique "right answer" by which the simulation codes can be characterized.
- Extensive testing and characterization is required to determine if the simulation package meets its requirements.
- Many packages have a strong R&D element, allowing for an environment of creativity and experimentation.
- The simulation codes are simultaneously undergoing revisions and additions, while being used in a production mode.

We are taking the best parts of the Software Development Process to effectively work in our unique code development environment.

We use the Parallel Development Software Life Cycle.



This Life Cycle is adaptive and responsive to our changing requirements and new technology, allowing for all phases of the life cycle to be occurring in parallel.

We employ more rigorous tools and methodologies to help us manage our complex code systems.

- **Project Management**
  - Gantt Charts
  - Task spreadsheets
  - Frequent Status meetings
- **Software Configuration Management**
  - Don't find the same bug twice.
- **Software Quality Assurance**
  - Verification/Validation
- **Requirements Management**
  - Bug/Enhancement tracking
- **Design Reviews**



Implementing these pieces of the Software Development Process have been instrumental in managing our complex simulation development environment.

Several Project Management tools and processes help us stay focused on our deliverables.

- The Gantt charts are primarily used to list our projects.
- The task spreadsheets are used to track code developer
  - projects,
  - status,
  - start and completion times,
  - quarterly deliverables.



- Regularly scheduled meetings are held to discuss
  - progress on projects,
  - code updates,
  - issues,
  - user needs
  - theory discussions

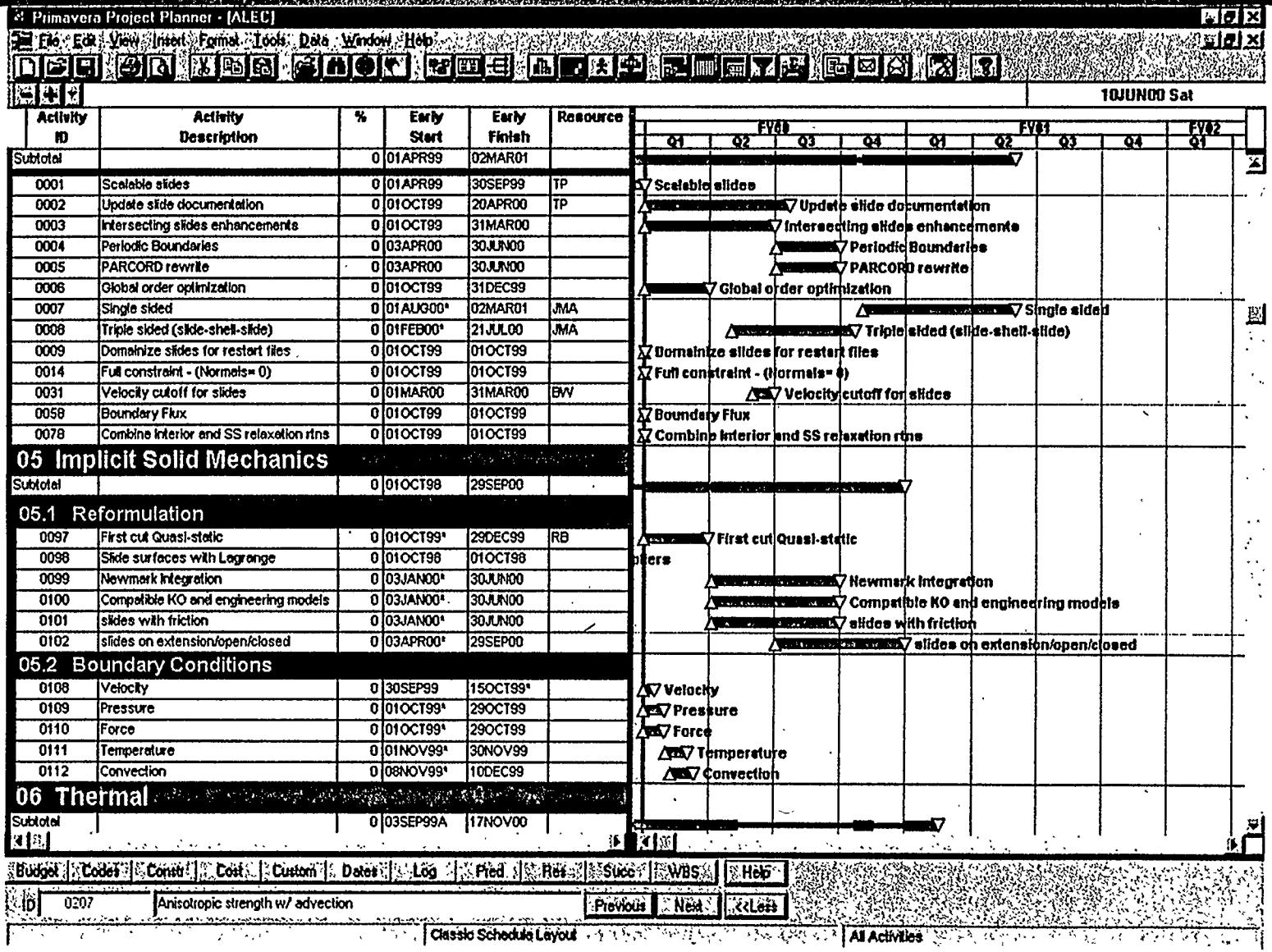
→ Status meetings

→ User meetings

→ Theory meetings

We feel more in control of the project and we have a better understanding of the workload.

We use Primavera Project Planner for managing our tasks.



Activity ID	Activity Description	%	Early Start	Early Finish	Resource
Subtotal		0	01APR99	02MAR01	
0001	Scalable slides	0	01APR99	30SEP99	TP
0002	Update slide documentation	0	01OCT99	20APR00	TP
0003	Intersecting slides enhancements	0	01OCT99	31MAR00	
0004	Periodic Boundaries	0	03APR00	30JUN00	
0005	PARCORD rewrite	0	03APR00	30JUN00	
0006	Global order optimization	0	01OCT99	31DEC99	
0007	Single sided	0	01AUG00*	02MAR01	JMA
0008	Triple sided (slide-shell-slide)	0	01FEB00*	21JUL00	JMA
0009	Domainize slides for restart files	0	01OCT99	01OCT99	
0014	Full constraint - (Normals= 0)	0	01OCT99	01OCT99	
0031	Velocity cutoff for slides	0	01MAR00	31MAR00	BW
0058	Boundary Flux	0	01OCT99	01OCT99	
0078	Combine interior and SS relaxation rtns	0	01OCT99	01OCT99	
05 Implicit Solid Mechanics					
Subtotal		0	01OCT98	29SEP00	
05.1 Reformulation					
0097	First cut Quasi-static	0	01OCT99*	29DEC99	RB
0098	Slide surfaces with Legrange	0	01OCT98	01OCT98	
0099	Newmark Integration	0	03JAN00*	30JUN00	
0100	Compatible KO and engineering models	0	03JAN00*	30JUN00	
0101	slides with friction	0	03JAN00*	30JUN00	
0102	slides on extension/open/closed	0	03APR00*	29SEP00	
05.2 Boundary Conditions					
0108	Velocity	0	30SEP99	15OCT99*	
0109	Pressure	0	01OCT99*	29OCT99	
0110	Force	0	01OCT99*	29OCT99	
0111	Temperature	0	01NOV99*	30NOV99	
0112	Convection	0	08NOV99*	10DEC99	
06 Thermal					
Subtotal		0	03SEP99A	17NOV00	

We use an Excel Spreadsheet for managing our person power and deliverables



Microsoft Excel - People_project_Jan00.xls

File Edit View Insert Format Tools Data Window Help

Arial 12

E90 working - waiting for additional solvers

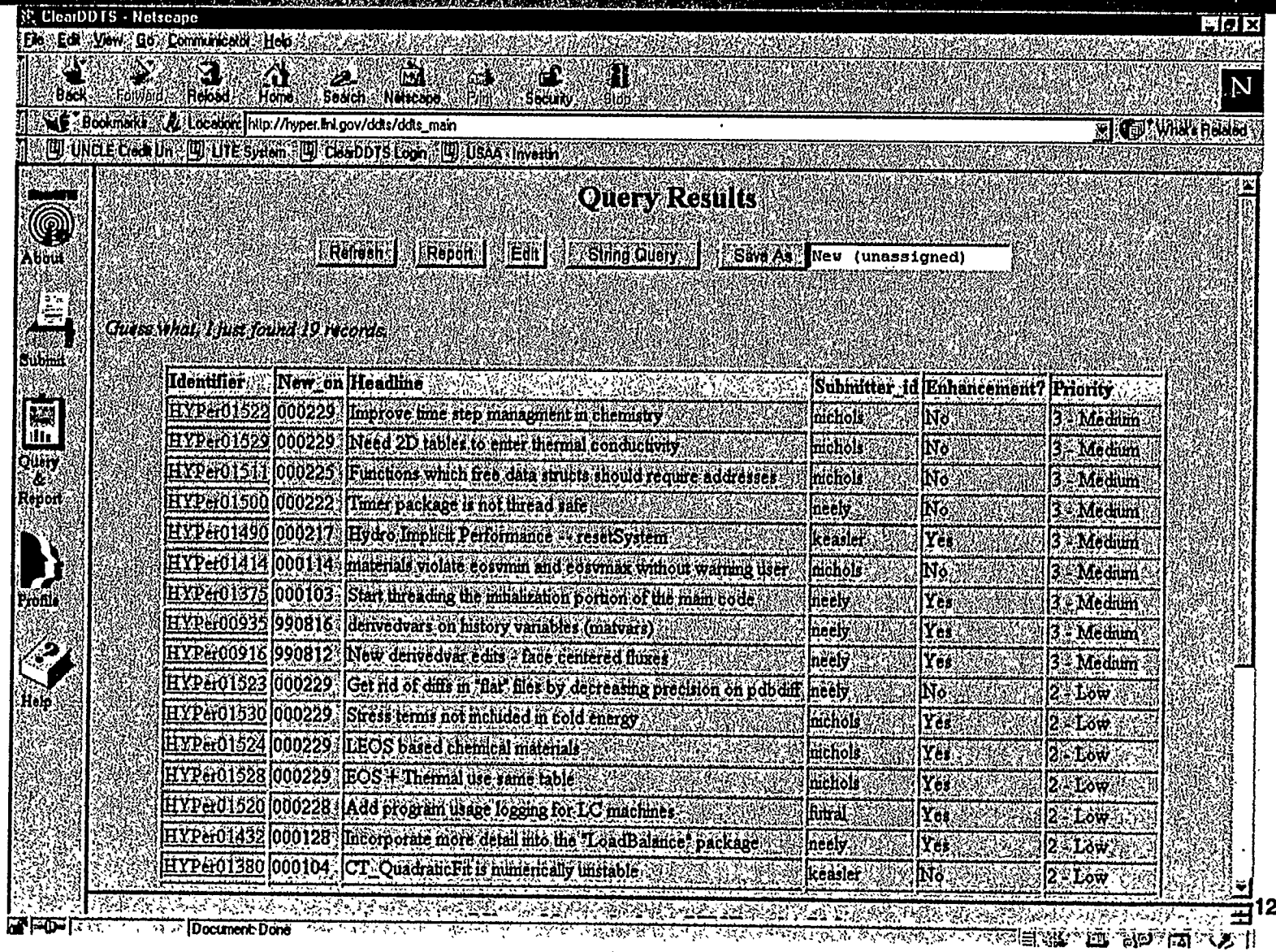
	A	B	C	D	E
80	**Brad Wallin**	**reformulated implicit slide surfaces**	**11/1/99**	**3/31/00**	working without voids
81		**Parallel Reduction with Tong**		**2/29/00**	need to get HYPRE on blue working
82		**parallel implicit hydro**			need history var mods.
83		**shells**		**2/14/00**	need for Jim Maltby end of this week
84		**Verification**		**forever**	meeting with Gregg M. to set up tests
85	**Andy Anderson**	**New thermal Boundary Condition**		**12/31/99**	testing accuracy on 3-D geometry
86		**Thermal/Chemisty V&V plan**			
87		**Write-up thermal test suite**		**1/31/00**	first draft completed - looks fantastic
88		***Big Thermal Problem - scaling study***		***2/28/00***	*debugging 30k parallel job; was 1/31/00*
89	**Rich Becker**	**Implicit hydro - solid elements with static equil**		**12/31/99**	Brad working on and testing
90		**- Newton iteration**			working - waiting for additional solvers
91		**Implicit dynamic capability**		**6/30/00**	
92		**- inertial - Newmark iteration**			
93		**Insertion of Gurson const model for fractures**		**3/31/00**	works with Lagr, doesn't with advect
94		**Insertion of traditional crystal plasticity const.**		**9/30/01**	started on
95	**Dan Nikkel**	**Reorganize core anisotropic plasticity code**		**12/31/99**	ready for new interface
96		**Demonstrate anisotropic effects**		**3/31/00**	
97		**Evaluate/quantify effect of anisotropic model**	**11/1/99**	**4/1/00**	getting problem from Kris Winer
98		**Improve integration meths for anisotropic model**	**1/1/00**	**4/1/00**	started on ; need more robust meths
99		**SRI interface**	**12/1/99**		protocol issues to work out
100		**Improve rate-dependent hardening model**		**Oct-00**	
101		**- implement Leuser model**		**Jan-00**	Deepak working on - starting to test
102	**Rich Couch**	**Shell elements**		**12/31/99**	debugging probs with slide surfaces
103		**- parallelization**			Rob Neely /Brad Wallin
104		**- verification**			
105		**new concrete model - Dyna-45**		**1/30/00**	getting back to

Sheet1 Sheet2 Sheet3

Ready

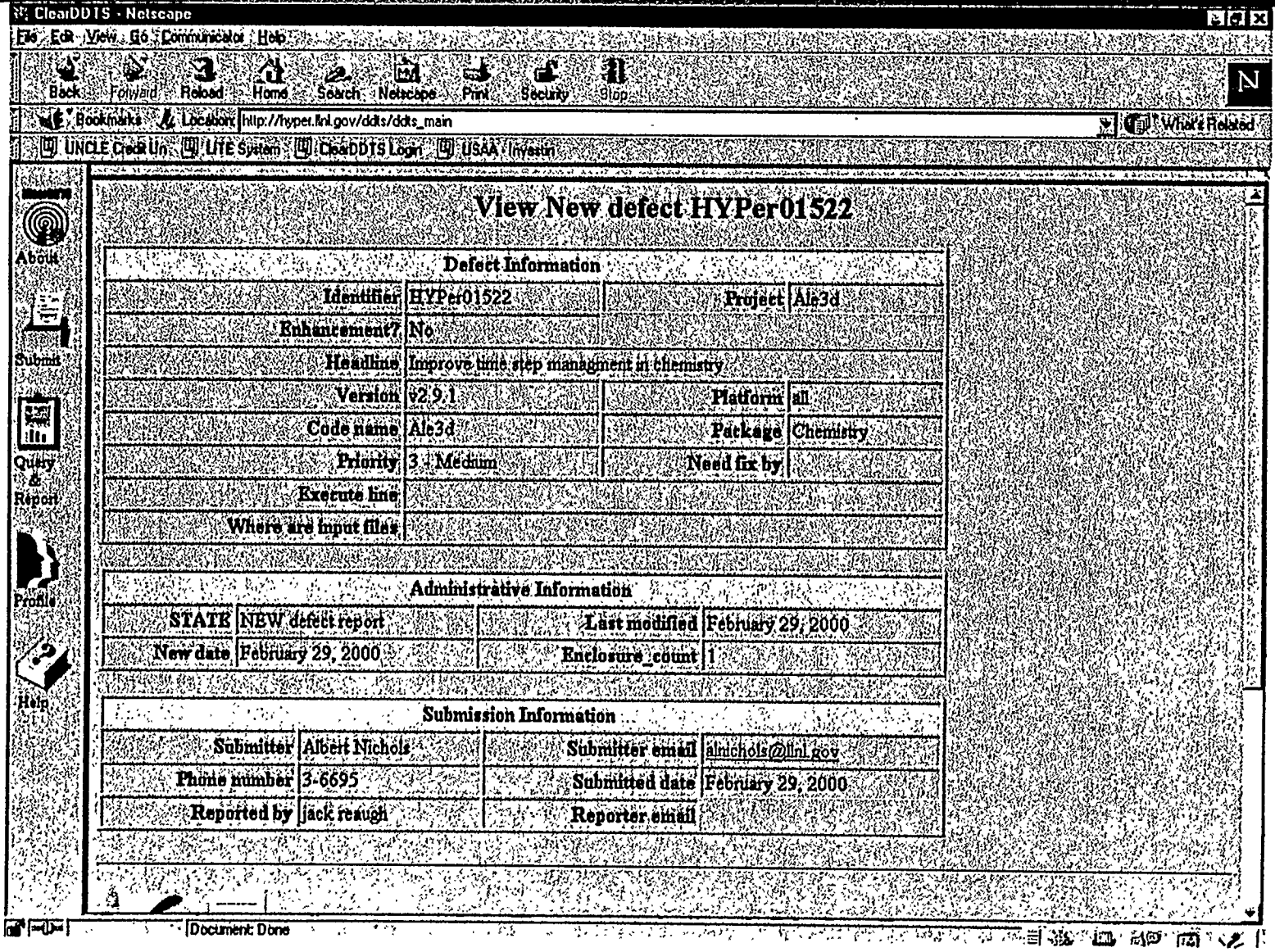
We use ClearDDTs for tracking bug fixes and enhancements.





Identifier	New_on	Headline	Submitter_id	Enhancement?	Priority
HYPer01522	000229	Improve time step managment in chemistry	nichols	No	3 - Medium
HYPer01529	000229	Need 2D tables to enter thermal conductivity	nichols	No	3 - Medium
HYPer01511	000225	Functions which free data structs should require addresses	nichols	No	3 - Medium
HYPer01500	000222	Timer package is not thread safe	neely	No	3 - Medium
HYPer01490	000217	Hydro Implicit Performance -- resetSystem	keasler	Yes	3 - Medium
HYPer01414	000114	materials violate eosvmin and eosvmax without warning user	nichols	No	3 - Medium
HYPer01375	000103	Start threading the initialization portion of the main code	neely	Yes	3 - Medium
HYPer00935	990816	derivedvars on history variables (matvars)	neely	Yes	3 - Medium
HYPer00916	990812	New derivedvar edits - face centered fluxes	neely	Yes	3 - Medium
HYPer01523	000229	Get rid of diffs in "flat" files by decreasing precision on pdbdiff	neely	No	2 - Low
HYPer01530	000229	Stress terms not included in cold energy	nichols	Yes	2 - Low
HYPer01524	000229	LEOS based chemical materials	nichols	Yes	2 - Low
HYPer01528	000229	EOS + Thermal use same table	nichols	Yes	2 - Low
HYPer01520	000228	Add program usage logging for LC machines	futral	Yes	2 - Low
HYPer01432	000128	Incorporate more detail into the 'LoadBalance' package	neely	Yes	2 - Low
HYPer01380	000104	CT_QuadraticFit is numerically unstable	keasler	No	2 - Low

We use ClearDDTs for tracking bug fixes and enhancements.



Defect Information

Identifier	HYPer01522	Project	Ale3d
Enhancement?	No		
Headline	Improve time step managment in chemistry		
Version	v2.9.1	Platform	all
Code name	Ale3d	Package	Chemistry
Priority	3 - Medium	Need fix by	
Execute line			
Where are input files			

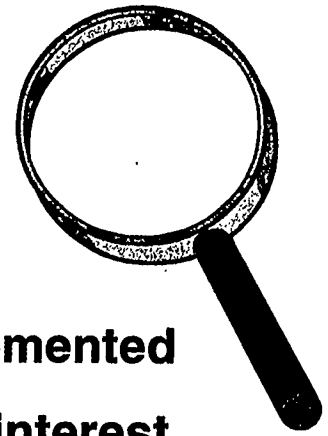
Administrative Information

STATE	NEW defect report	Last modified	February 29, 2000
New date	February 29, 2000	Enclosure_count	1

Submission Information

Submitter	Albert Nichols	Submitter email	alnichols@llnl.gov
Phone number	3-6695	Submitted date	February 29, 2000
Reported by	jack reaugh	Reporter email	

Software Configuration Management (SCM) lets us establish and maintain the integrity of the project.

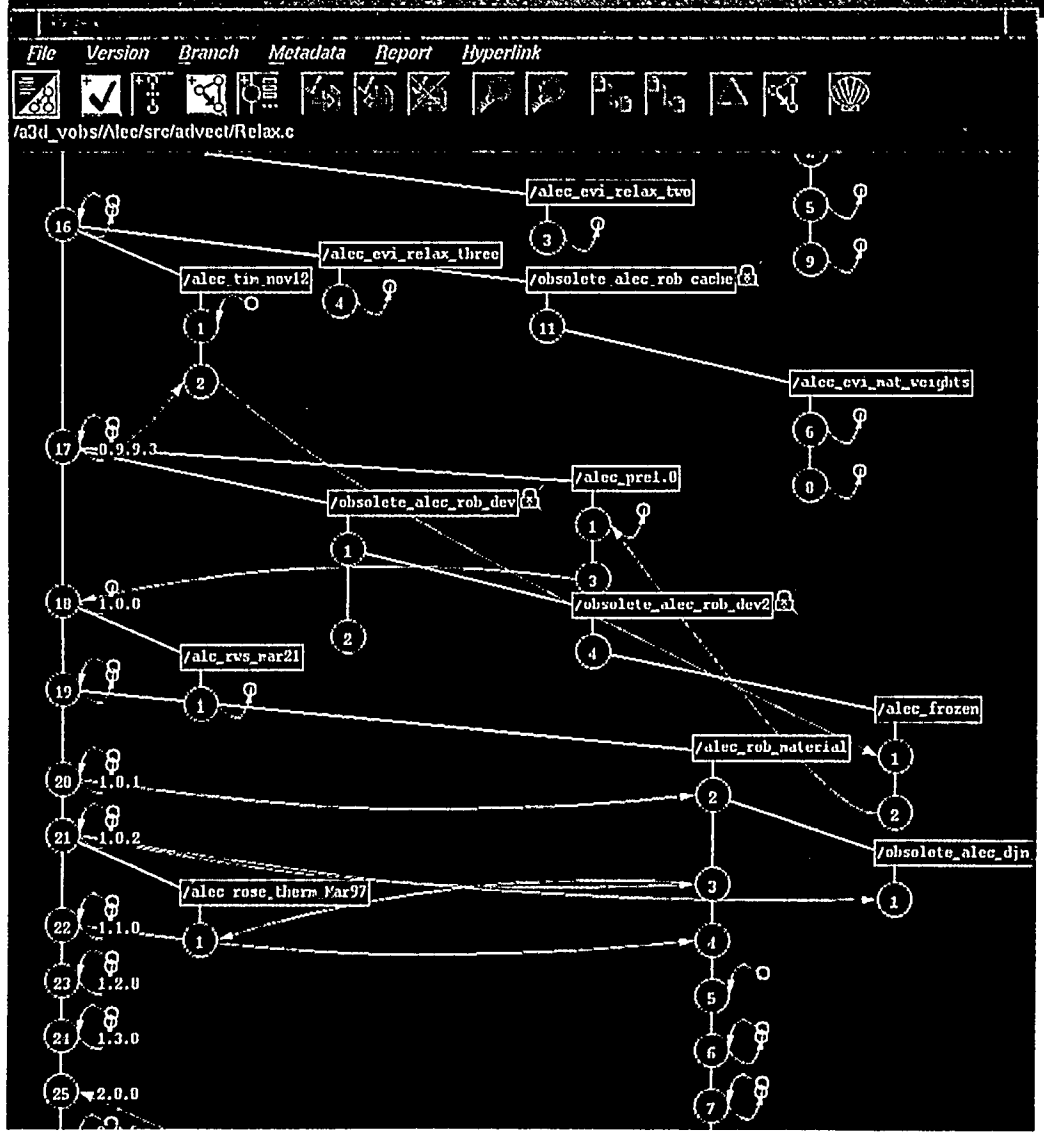


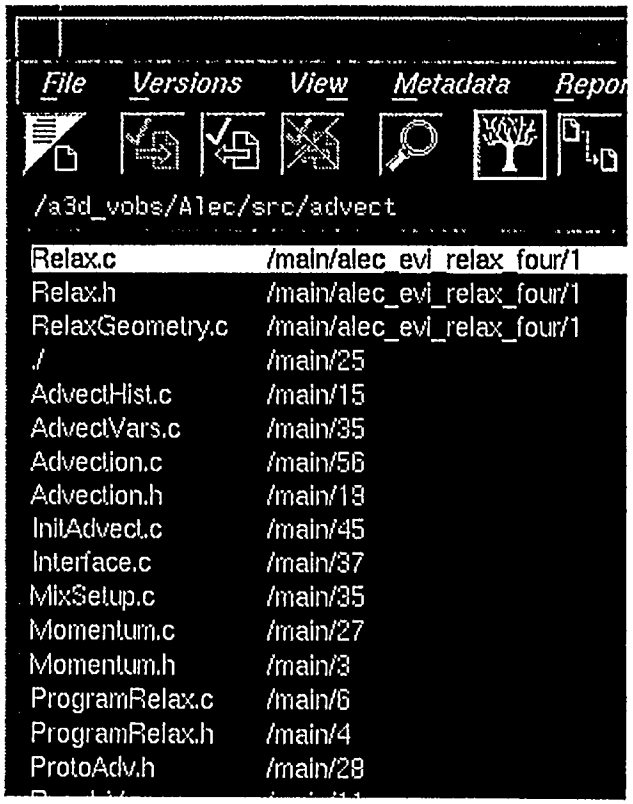
With SCM, we

- identify change
- control change
- ensure that change is being properly implemented
- report change to others who may have an interest.

Every LLNL ASCI simulation project uses SCM, and these tools and practices have allowed us to gracefully recover from major mishaps.

The SCMS tool gives a pictorial view of the simulation codes parts and history.





We incorporate a broad range of Verification procedures in our Software Development Process.



- We use analytic solutions to test against, when available.
- We compare the current simulation results against previous results, when available.
- We compare simulation versus experiments, when available.
- We compare one code's results against another, when available.
- We maintain and execute test suites prior to each code release.
- Our testers come from within the simulation code team and from outside the code team.

Understanding and confidence in the simulation results requires extensive testing and several years of normalization and experience.

The simulation environment at LLNL has changed.



- We have evolved from the collection of "hero" codes to a few large, integrated, and managed simulation projects.
- We have not stuck our heads in the sand - we are quickly learning new techniques to help us manage our critical software projects.
- The Tri-Lab Assessment showed us and DOE management that we are on the right track.

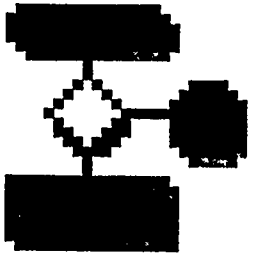
We believe that in using these Software Development Processes, we will effectively meet the SSP and ASCI challenges.

KANSAS CITY PLANT

Honeywell



Project Management Automation



Don Rathbun Cathy Kuhn

Honeywell Federal Manufacturing & Technologies*

DOE, Kansas City Plant

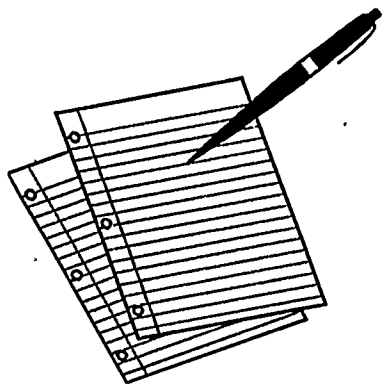
*Operated for the United States Department of Energy under Contract No. DE-ACO4-76-DP00613

Disclaimer Notice

All data prepared, analyzed and presented has been developed in a specific context of work and was prepared for internal evaluation and use pursuant to that work authorized under the referenced contract. Reference herein to any specific commercial product, process or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof or Honeywell International Inc.

Agenda

- Project History
- Information Systems Service Request
- Engineering Project file
- Demo



- Two Projects
- Two Approaches
- Same Goal: *Provide a consistent method for managing projects electronically*
- One Solution: *Lotus Notes¤ Applications*

Presentation Goals

- Explain each project's approach
- Discuss the commonality and differences of the solution
- Discuss how the projects leveraged the use of existing software and processes
- Provide YOU somé thoughts on how you might be able to do something similar

Information Technology's Service Request System

History - Old System

- Paper Requests generated in Electronic Forms System
- Project tracking done in Access database
- Paper Filing System of Project Documentation

Concerns with Current Process



- Paper documentation
- Some incomplete forms
- Different approaches
- Couldn't easily support our SEI Capability Maturity Model (CMM) level 2 initiative

Summary - Despite Documented Processes - Inconsistencies Existed

Solution - Cross-Functional Team

Goals -



- Keep it simple
- Make is user friendly
- Don't forget the customer
- Ensure all security requirements would be met
- Ensure management input
- Ensure that all projects would follow the same process & generated required documentation
- Budget limitations
- **+ a few dozen more**

Team Activities

- Reviewed & mapped documented process
- Reviewed & mapped "real" process(s)
- Interviewed - managers, customers, developers across all platforms
- Developed and validated solution

Results

- A defined process involving key plant-wide functional areas, developers, customers and management
- An automated tool to track and manage the process including all the generated documentation



Improvement Highlights

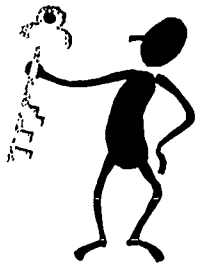
Before

- Divisional Focus
- Managers Coordinate
- Paper Forms
- Access Database
- Paper Project Files

After

- Functional Focus
- Single-Point-of-Contact
- Real Time Status
- Web Application
- Embedded Workflow
- Electronic Notification
- Electronic Sign-offs
- Electronic Project Files

Key Process Steps



Step	Responsibility
• Initiate Request	Requester
• Business Feasibility	Business Unit Coordinator
• Security & Technical Review	Request Coordinator
• **Technical Feasibility**	Manager/**Analyst**
• Resource Availability	Manager
• Project Status	Request Coordinator
• Prioritize Request	Business Unit Coordinator
• Assign Request	Manager
• **Work Request**	**Analyst**
• Close-out Request	Manager

Tool - Lotus Notes® Application

Highlights -

- Submit requests via the Web
- Live status
- Automatic Mail notification
- Controlled work flow
- Controls access rights
- Provides for electrónic sign-offs
- Documentation is part of request
- Requires consistent documentation

Tool - Lotus Notes® Application

Why?

- Familiarity with Lotus Notes®
- Customizable - made it easy to use
- Support structure (and hardware) in place
- Easy linkage to procedures and mail system
- High degree of security

We chose to go with what we had and understood - reduced learning curve for users

Results

- Customers like it
- Business Unit Coordinators can manage & prioritize
- Managers know what projects need to be worked
- Analyst know what is assigned to them
- No lost requests
- No misplaced documentation

Currently 250+ requests since Apr 99

Slide Demo

&

Questions?

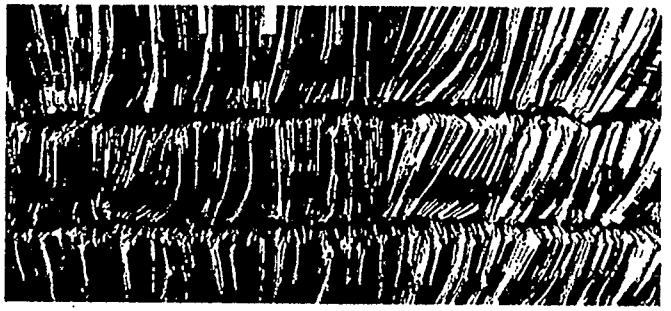
Project File Cabinet

History - Old System

- Paper file system
- Project tracking done many ways
- Paper Filing System of Project Documentation

Typical Problems

- Misplaced documentation
- Inconsistent approaches
- Potential audit findings



Summary - Despite Documented Processes - Confusion existed

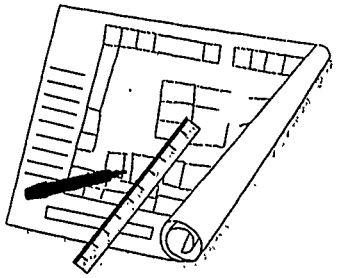
Cross-Functional Team

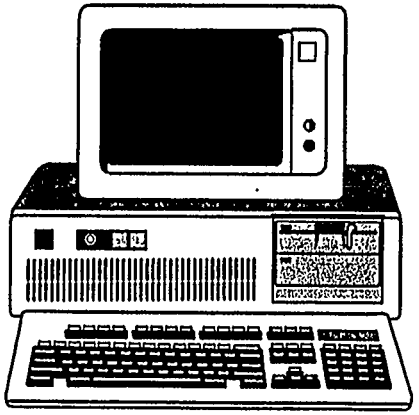
- ADAPT Project
- Six-Sigma Team
- Objective-Improve the Project Management Process by providing an on-line project management system that was:
  - Flexible
  - Shareable
  - Scaleable



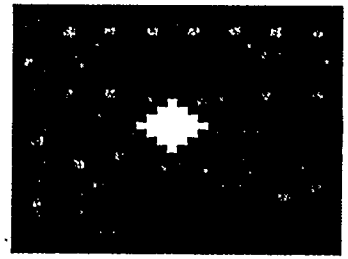
Cross-Functional Team

- ----and would facilitate the capturing of project history as it occurs.





Project File Cabinet



- **IS Service Request used as the starting point for this application**
- **Utilized the Lotus Notes® infrastructure already in place to reduce development and implementation costs**
- **Application was developed on Lotus Notes®, QA tested and approved and released for use October 4, 1999**

Slide Demo

Metrics on Project File Cabinet

- Application released to Production 10/4/99
- 1/31/2000 - 65 Active Projects maintained in the System & 2 Archived Projects
- 36 Different Project Leaders
- Database size = 132 Mb, 232 Documents
- High user acceptance!

Project File Cabinet Enhancements

- Web browser interface
- Print email to PDF Writer for better formatting for storage.
- Integrate with business processes related to ICOs (Integrated Contractor Orders).
- Integrate with Design Processes for "Check-In" of Design Documents.

Questions?



Perspectives on Application of Software Process Assessments to Scientific Codes

Software Quality Forum 2000

April 19, 2000

presented by

Bill Reed, Headquarters DOE
ASCI V&V Executive Council Chair

Lawrence Livermore National Laboratory is operated by the University of California for the United States Department of Energy. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process

Nancy Storch, Evi Dube LLNL

Gerald Reisz, LANL

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

Los Alamos
National
Laboratory

Dennis Adams, INEEL

The work for this project was funded under the ADAPT program (Advanced Design and Production Technologies - DPR6KB142) under contract DE_AC04_91AL65030.

INEEL

Dave Peercy, Patty Trellue, Jaime Moya SNL

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Sandia
National
Laboratories



Survey Request Background

At the request of the Accelerated Strategic Computing Initiative (ASCI) V&V Executive Council, a Tri-Lab Survey of software engineering practices used by ASCI code teams was performed at LANL, LLNL, and SNL. The survey approach used was the Software Engineering Institute's Capability Maturity Model-Based Appraisal for Internal Process Improvement (CBA-IPI).

This survey was conducted during the period from December 1998 through deliver of the final report on August 9, 1999.

Tri-Lab Survey Information is available at the web site:
http://cio.doe.gov/asci/
select: Software Verification for ASCI





Presentation Approach

Objectives

- Present details of the survey process; specific results are not given
- Provide perspectives of the benefits and problems as viewed by persons who participated in the various survey roles
- Discuss some general recommendations and lessons learned
- Solicit audience questions and discussion





Agenda

- **Assessment View** – 35 minutes
  - Process overview - Patty Trellue, SNL
  - Assessor view – Dave Peercy, SNL
  - Site coordinator view – Nancy Storch, LLNL and Gerald Reisz, LANL
  - Code team participant view – Evi Dube, LLNL
  - Independent Observer View – Dennis Adams, INEEL
- **Management View** - 20 minutes
  - Laboratory V&V Manager – Jaime Moya, SNL
  - Headquarters DOE – Bill Reed, HQ DOE
- **Lessons Learned & Recommendations** – 10 minutes
  - Panel Discussion
- **Audience questions and discussion** – 40 minutes





Survey Process Overview

presented by
Patty Trellue, SNL





Background:
Survey Process

- **Background**
  - Request
  - Process
  - Participation
- **Results**
  - Strengths
  - Opportunities for Improvement
  - Questionnaire Analysis
- **Recommendations**
  - Laboratory and Code Team
  - DOE ASCI Program





Background:
Survey Request

- **ASCI DOE Headquarters**
  - Requested: Memo 13 Dec 1998 by Gil Weigand & Bill Reed
  - Focus: Software Engineering Practices
  - Participation: Tri-Lab ASCI Application Code Teams
- **Tri-Lab Survey**
  - Mike Blackledge, SNL, Survey Leader
  - Survey Team participants from SNL, LANL, LLNL
  - INEEL participate as independent observer
  - Implementation Plan for the Tri-Lab Survey process





Objectives of Survey

- **Establish Software Process Baseline**
  - Use SEI Capability Maturity Model (CMM) Version 1.1
  - Establish reference point for process improvement
  - Provide basis for prioritizing improvement efforts
- **Identify Areas of Strength and Opportunities for Improvement**
  - Focus on 13 CMM Key Process Areas (KPAs) plus V&V KPA
  - Appropriate application of KPAs to ASCI environment
- **Support Development of Action Plans**
  - Strategic action plans to guide long-term improvement
  - Tactical action plans for initial improvement





Background:
Training Session:15Dec98

- **Site: DOE's Energy Training Complex, ABQ**
- **Attendees (26):**
  - LLNL(13)
  - LANL(6)
  - SNL(5)
  - INEEL(2)
- **Deliverables:**
  - Training Notebook; Draft Survey Implementation Plan
  - One day session for Assessment Team Training





Background
Implementation Plan: 25 Jan 1999

- **Project Statement**
  - Purpose, Background, Scope
  - Benefits, Deliverables
  - Roles and Responsibilities
- **Plan for the Tri-Lab Survey**
  - Requirements
  - Approach, Issues
  - Activities and Responsibilities
  - Confidentiality and Information Management
- **Plan for Laboratory Assessments**
  - Personnel, Activities, & Generic Schedule

http://cio.doe.gov/asci/rd000032.htm





Background
Participant Roles and Responsibilities

◆Assessment Team

- Team of assessors who perform laboratory assessments.
- Produces Code Team Assessment Reports, Laboratory Assessment Reports, and the Tri-Lab Survey Report.
- Normally, the Lead Assessor and the three Primary Assessors compose the A-Team. Alternate Assessors may substitute for Primary Assessors for a particular laboratory assessment.

◆Process Observer

- INEEL staff member who observes and evaluates each laboratory assessment.
- Evaluates the survey process.
- Produces the Tri-Lab Survey Assessment Report.





Background
Participant Roles and Responsibilities

- **Site Coordinator**
  - Laboratory staff member who works with the Lead Assessor for planning and conducting a laboratory assessment.
  - Responsible for all site logistics, to include scheduling, facilities, equipment, and meals.
  - Works with the laboratory's ASCI V&V Executive Council Representative for selecting the participating code teams.
  - Works with participant code teams for responding to the standard questionnaire, selecting interview group members, and collecting samples of project materials.





Background
Participant Roles and Responsibilities

- **Code Team Lead and Code Team Staff**
  - Technical lead or project manager for a Code Team. Receives the Code Team Assessment Report.
  - Staff members in a laboratory group who collaborate on the development and delivery of software products for a particular ASCI functionality. Participate in interview session for code team.
- **ASCI V&V Executive Council**
  - The group of all laboratory ASCI V&V Program Managers, chaired by the DOE V&V Program Manager
  - Chair: Bill Reed
  - LLNL ASCI V&V Manager: Cynthia Nitta & Richard Klein
  - LANL ASCI V&V Manager: Tom Seed (now Ken Koch)
  - SNL ASCI V&V Manager: Jaime Moya (now Martin Pilch)





Background
Survey Activities

- **Preparation**
  - Training Sessions for Assessment Team (A-Team)
  - Orientation Meetings with Participants and Management
  - Questionnaire & Project Profile Information Completed
  - Process Evidence Gathered/Identified
- **Assessment**
  - In-Briefing
  - Interviews, Analysis, Evidence Review
  - Out-Briefing
- **Follow-Up**
  - Final Report





Background
Survey On-Site Visits

Site	Assessment Dates	Delivery & Review Dates
LANL	Feb 8-10, 16-19, 22-26, 1999	May 26-28, 1999
SNL	Mar 15-19, 22-26, 29, 1999	June 14, 15-16, 1999
LLNL	April 12-16, 19-22, 1999	June 23, 1999



Background
Survey Teams

	LANL Survey Team	SNL Survey Team	LLNL Survey Team
Team Lead	Patty Trellue	Patty Trellue	Patty Trellue
LANL Representative	Skip Egdorf	Jonathan Parker	Maysa Peterson
LLNL Representative	Nancy Storch	Booker Thomas	Nancy Storch
SNL Representative	Dave Peercy	Dwayne Knirk	Dave Peercy
INEEL Observer	Dennis Adams	Dennis Adams	Nyle Brown





Background
Interview Groups

- **LANL** — **94**
  - Management Group - 15 individuals
  - HW/SW Environment Group - 13 individuals
  - V&V - 8 individuals
  - 8 Code Teams - 58 individuals
- **LLNL** — **53**
  - Management Group - 11 individuals
  - V&V Group - 10 individuals
  - 4 Code Teams - 32 individuals
- **SNL** — **57**
  - Management Group - 11 individuals
  - 8 Code Teams - 46 individuals
- **Total Participation in Tri-Lab Survey** — **204**





Background
Typical Survey Team Day

08:30-10:30 - Interview with Code Team

10:30-12:00 - Interview analysis session

12:00-12:45 - Lunch

01:00-02:30 - Code Team evidence review

02:30-03:30 - Evidence analysis session

03:30-05:30 - Code Team analysis - strengths and areas for improvement

05:30-07:30 - Code Team questionnaire rating





Background
Evidence Review

- **Documentation**
  - Project profile sheets, questionnaires
  - Project plans: task/schedule charts, task descriptions, cost data, meeting minutes
  - V&V plans, test problems, test results, defect lists
  - On-line & Web page documentation, processes, procedures
  - Specific software artifacts: requirements, design, programmers manuals, users manuals, style guides, code fragments, build scripts
  - Published papers, presentation materials
- **Demonstrations**
  - Code execution; V&V results; Movie script visualization of experiments





Background
Example Strength Statement

- **Physics & Engineering Verification and Validation**
  - Verification regression test suites exist for most codes
  - Analytic verification results exist for most codes
  - Approach to validation is systematic and scientifically defensible
    - However, V&V is primarily defined at the physics level rather than at the software level
    - There is a still a significant technical challenge in being able to demonstrate the viability and credibility of the project codes.
  - Designers are a critical resource in this process

Conclusion: There is an understanding of what analytic and experimental test problems are needed at the scientific physics and engineering level and some progress among the code teams in demonstrating these test results.





Background
Example Improvement Statement

◆Observations

- Code teams generally do not have enough computer science/software engineering personnel
- Software engineering skills necessary to implement engineering practices and design for requisite supportability and reliability are not adequately represented on the code teams
- Additional personnel with science-based computer science/software engineering skills generally are not available to the code teams
- Several codes can only be supported by a few code domain experts and are difficult to migrate into the larger scope and complexity of ASCI applications

Conclusion: Lack of personnel with adequate computer science skills hinders the establishment of software engineering practices needed by ASCI code teams.

☺ Code teams with better balance demonstrated effective use of better software practices



Background
Example Questionnaire Summary*

* These numbers do not represent any code team or composite rating; example only.

		Average Survey Team Rating			
Level 2	Acronym	Ability to	Performance	Deployment	Results
Requirements Management (KPA.01)	RM	[illegible]	[illegible]	8.0	8.0
Software Project Planning (KPA.02)	PP	[illegible]	[illegible]	8.0	7.0
Software Tracking & Oversight (KPA.03)	TO	[illegible]	[illegible]	7.0	7.0
Software Subcontract Management (KPA.04)	SS	n/a	n/a	n/a *	n/a
Software Quality Assurance (KPA.05)	QA	[illegible]	[illegible]	3.0	5.0
Software Configuration Management (KPA.06)	CM	[illegible]	[illegible]	6.0	6.0
Level 3					
Organization Process Focus (KPA.07)	PF	[illegible]	[illegible]	7.0	6.0
Organization Process Definition (KPA.08)	PD	[illegible]	[illegible]	6.0	5.0
Training Program (KPA.09)	TP	[illegible]	[illegible]	8.0	8.0
Integrated Software Management (KPA.10)	IM	[illegible]	[illegible]	6.0	5.0
Software Product Engineering (KPA.11)	SE	[illegible]	[illegible]	6.0	6.0
Intergroup Coordination (KPA.12)	IC	[illegible]	[illegible]	7.0	7.0
Peer Reviews (KPA.13)	PR	[illegible]	[illegible]	3.0	3.0
ASCI Verification and Validation (extra KPA)	VV	[illegible]	[illegible]	5.0	5.0

Each cell was rated on a scale of 0-10, with 6 being interpreted as marginally passing the indicated cell criteria.





Background
Summary of Deliverables

- **Implementation Plans (4)**
  - 1 Tri-Lab Survey Implementation Plan
  - 3 Laboratory Implementation Plans
- **Final Report Briefings (4)**
  - 3 Laboratory Final Report Briefings
  - 1 DOE Final Survey Report Briefing
- **Final Reports (24)**
  - 20 Code Team Final Reports
  - 3 Laboratory Final Reports
  - 1 DOE Final Survey Report





Background
Confidentiality

- **Survey Team and Independent Observer Sign Confidentiality Agreement**
  - No individual or code team or Laboratory will be identified as the source of any information in the final reports
  - No specific information gathered from interviews or evidence reviews will be discussed outside of the survey team





Background
General Recommendations

- **Develop Action Plans Based on This Survey**
  - Laboratory Level
  - Code Team Project Level
  - Target at least one improvement
- **Establish a Mechanism to Continue Assessment Concepts**
  - Conduct regular internal self-assessments at the code team project level
  - Conduct interval self-assessments of the Laboratory ASCI software processes
- **Share Improvements**
  - Identify how improvements can be shared





ASSESSMENT PERSPECTIVE

Assessor Perspective
presented by
Dave Peercy, SNL





Assessor Perspective
Overcoming Obstacles

- **Early Issues and Concerns**
  - Obtaining Tri-Lab cooperation
  - Ensuring confidentiality of assessment information
  - Training on the survey method
  - Use of survey results
- **Frequently Exchanged Assessment Rumors**
  - The Baseline Survey Report to Bill Reed can be misused and impose non value-added compliance/ standards on code teams.
  - The Survey is weighted too heavily with Sandia participation rather than equal Tri-Lab. Why is Sandia qualified to lead this Survey effort?
  - The Survey will not provide adequate value to the code teams for the effort expended; the Survey will require too much time by the code team members.
  - The Survey process will not adequately handle classified codes and possible classified discussions/information.





Assessor Perspective
Overcoming Obstacles

☺ *Despite initial fears and resistance, all Code Teams were open and cooperative during the interview sessions.*

☺ *Code Teams were open and cooperative in providing evidence for review. There tended to be more evidence than the Code Teams realized.*

☺ *Professional approach by the assessment teams was well-received by the code teams and site management.*

The security stand-down at all three laboratories occurred in the middle of the survey process and created delays in conducting the assessments and in returning results to teams already assessed. Created time crunch for the assessment team.

Assessor Perspective
Time Commitment and Logistics

Assessment team was extremely stressed during the assessment activities at each site due to the number of code teams covered in a short time.

Assessment team was concerned about the lack of adequate time to review evidence presented by each code team during the day allocated to each code team.

The logistics of coordinating the assessment team activities across the geographic separation of the three laboratories was very difficult. Technical and classification reviews of the reports were delayed.

☺ *Despite the performance stress, the assessment team members were cooperative, respectful of each other's position and statements, and were not biased toward the site they represented.*



Assessor Perspective
Overall Survey Results

- *Assessments were fair and accurately represent the state of software practices by the various ASCI code teams. Accumulative results at the laboratory and across all laboratories reflect general practices, but not necessarily any specific team.*
- *Most ASCI code teams and the ASCI V&V laboratory management were responsive to the results. ASCI V&V at each laboratory tended to provide extended funding for improving software practices.*
- *Most ASCI code teams had initiated improvement efforts (or actually had improved) from the time of the assessment to the time when their final report was distributed and reviewed with each code team.*
- *There are still some after effects (read that as comments from some teams) that the results were either already known, hence of little value, or not completely accurate representations. Some teams do not believe the process applies to their software, and that CMM practices are not appropriate for their needs.*





ASSESSMENT PERSPECTIVE

Site Coordinator Perspective
presented by
Nancy Storch, LLNL and Gerald Reisz, LANL





Assessment Resistance

- **Process Assessment was Foreign and Resisted**
  - Control and direction comes from within the projects
  - Considered as diversion away from primary mission and milestones
  - A feeling of being judged by outsiders against criteria not well understood or felt to be irrelevant
  - Needed opportunities to explain their culture and approach
  - Some open hostility; rare instances of interest





Management Support

- **Management Support Down the Line was Essential**
  - Clear statement of participation
  - Responsibility for scheduling put on program management when convergence on dates became impossible
  - Code team leaders willing to put in time and effort for questionnaire, evidence collection and meetings with A-Team





Logistics Coordination

- **Coordination was Complex and Time Consuming**
  - Participants included code teams from different organization, all levels of management and V&V designers
    - ~ 75 people at LLNL (53 direct survey participants)
    - ~110 people at LANL (94 direct survey participants)
    - ~ 75 people at SNL (57 direct survey participants)
  - Significant planning, detailed scheduling, and visit preparation for the A-Team
  - Needed to understand and abide by Site Classification and Security procedures
  - A-Team had a rigorous day schedule and worked after hours
  - Relied on significant administrative help





Results

- **Personal Results**
  - Enjoyable in spite of the pain
  - You really accomplish something
  - Critical to think of questions and consistent answers ahead of time
  - Closure
    - Sit in on some of the interviews
    - Sit in on some of the final assessment sessions





ASSESSMENT PERSPECTIVE

Code Team Participant Perspective
presented by
Evi Dube, LLNL



The code team's initial impressions:

- WHAT!!!!
- There was much concern about the relevance of the survey questions.
- There was much concern of the usage of the results.
- There was much concern of the time necessary to do survey.
- There was much concern of the usefulness of the process.



Uncertainty and skepticism was the major feeling of the code teams.

The code team's view of the actual assessment:

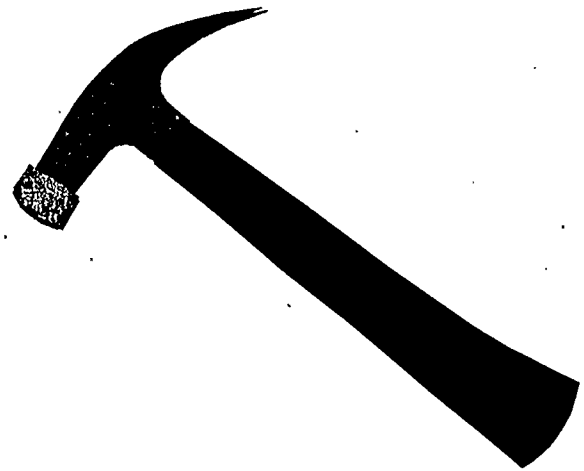


- 70% of survey questions were relevant.
- It was not too time consuming.
- We were given assurances from DOE HQ that the results were baseline, and not to be used against us.
- The survey team was made up of professional, courteous, intelligent folks.

I think the survey team's honesty and professionalism made the experience palatable. I cannot say enough positives to reflect the integrity of this team.

The code team's post-assessment views:

- The assessment was reasonably accurate of the Tri-Lab codes.
- There were no surprises for my code team.
- Have things changed?
  - We still lack staffing necessary to tackle our biggest concerns.
  - We are more cognizant/aware of trying to establish good, standard practices and processes.



The critical mission of these software packages demands a much more structured and ordered environment then in the past.



ASSESSMENT PERSPECTIVE

Independent Observer Participant Perspective
presented by
Dennis Adams, INEEL





INEEL's ROLE

- **Provide Independent Perspective**
  - ◆ INEEL not part of the Weapons Complex
  - ● INEEL is not involved in ASCI code design, development, or testing activities
- **◆ Objectivity**
  - ◆ Provide a separate position if required
  - ◆ Provide assessment feedback, process changes...





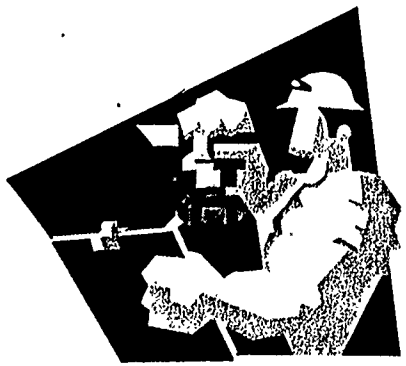
INEEL's ACTIVITIES

- **Attended Survey Process Training**
- **Reviewed Implementation Plan**
- **Observer at All Tri-Lab Surveys**
  - Asked questions
  - Participated in analysis and evidence sessions
  - Helped formulate survey results
  - Participated in code team report debriefings



Observation Summary

- **INEEL's Opinion**
  - Problems encountered are fairly minor
  - Followed the implementation plan
  - Survey questions, interviews, evidence, and analysis provided valid information
  - Baseline established at the code team level





MANAGEMENT PERSPECTIVE

Laboratory V&V Manager Perspective
presented by
Jaime Moya, SNL





ASCI V&V Manager Role

- **Provide Link to the ASCI V&V Executive Council**
  - Extended representative of the council chair at the laboratory
  - Encourage survey focus & commitment of the laboratory code teams
- **Provide Link to the Site Code Teams**
  - Work with site coordinator to resolve code team questions
- **Provide Site Commitment to the Survey**
  - Participate in the survey in-briefing and out-briefing
  - Receive the laboratory final report
- **Provide Follow-Up Incentives**
  - Ask code teams to provide their response to individual reports
  - Provide funding and resource support for follow-on improvement
    - Laboratory level across code teams
    - Code team specific improvements





Survey Issues

Code Team Resistance

- Some code teams needed significant reassurance and debated the value of the assessment
- Some code teams welcomed a look at their practices
- The general attitude of the code teams improved over the course of the survey, but should not be construed as strongly supportive

Survey Value-Added

- Most code teams appreciated the survey effort and results
- Most code teams learned something about the CMM key process areas and their relationship to their software practices
- Many code teams were unsure as to the cost/benefit of the results





Survey Follow-On

Laboratory V&V Commitment

- Funding has been provided at all laboratories through the V&V program to improve software practices of the ASCI code teams
- Software quality engineering is being addressed to some extent in the required ASCI V&V Plans for each code team
- Precise approach and implementation details varies across the laboratories

Tri-Lab Coordination and Commitment

- There is continued effort to coordinate improvements, communication of methods and tools across the three laboratories
- ASCI V&V Executive Council is committed to improving software engineering practices
- ASCI Tri-Lab Web Page: http://cio.doe.gov/asci/





MANAGEMENT PERSPECTIVE

ASCI V&V Executive Council Chair Perspective
presented by
Bill Reed, HQ DOE





LESSONS LEARNED

Panel Discussion





Some Panel Ticklers

Process Issues

- Application to science-based software/R&D software/small teams
- Laboratory cultural influences
- Differences between standard SPA and the Tri-Lab Survey SPA
  - Request from HQ rather than from Code Team/Organization
  - Multiple Laboratory A-Team Participants vs one organization team
  - Multiple A-Teams vs same A-Team
  - Multiple teams in short time vs one organization
  - Multiple team rollup into laboratory report vs none
  - Multiple laboratory rollup into ASCI program report vs none
  - A-Team training of multiple persons vs team members only
  - Follow-up discussions with each code team & laboratory vs none
  - Follow-up reports in one to two months vs couple of weeks
  - Classified security issues vs none





Some Panel Ticklers

- **A-Team**
  - Needed better understanding of terminology and agreement on the assessment rating scale
- **Code Team**
  - Needed to have a better mapping of the questionnaire terminology to organization-specific terminology
  - Value of survey vs impact on technical work
- **Security**
  - Needed to have an authorized derivative classifier available daily to review and declassify the notes, observations, etc.
  - Needed to have a better process for getting final reports reviewed for declassification -- before the start of the survey





Some Panel Ticklers

- **Follow-Up**
  - Need better measures to determine what follow-up is being done and whether there is evidence of commitment to improvement for ASCI code teams, laboratory, and overall program
- **Results**
  - Are code teams addressing the tri-lab survey reports?
  - Is there more commitment to improving software engineering practices?
  - Is there any improvement?
  - How would we know?
- **If we were to do this again, what would we do different?**





Audience Q&A

- Survey Process?
- Survey Results?
- Application to Science Based Software?
- Better Approaches?

Please address your questions to the panel moderator or a specific panel member. Use a microphone if one is available.



Anticipating the Effects of Change

presented to

Software Quality Forum 2000

by

Esther Derby
derby@EstherDerby.com
(612) 724-8114

Esther Derby Associates
www.EstherDerby.com

Rick Brenner
rbrenner@ChacoCanyon.com
(617) 263-1112

Chaco Canyon Consulting
www.ChacoCanyon.com

April 18-19, 2000

"People hate change. . .
And that's because people hate change. . .
I want to be sure that you get my point.
People really hate change.
They really, really do."

Steve McMenamin
The Atlantic Systems Guild
London (1996)

Copyright (c) 2000 Richard Brenner, Esther Derby

Models of Change

- People and organizations go through change in predictable patterns
- Many models of change
- All have some validity
- The Satir Change Model best illustrates how individuals and organizations go through change

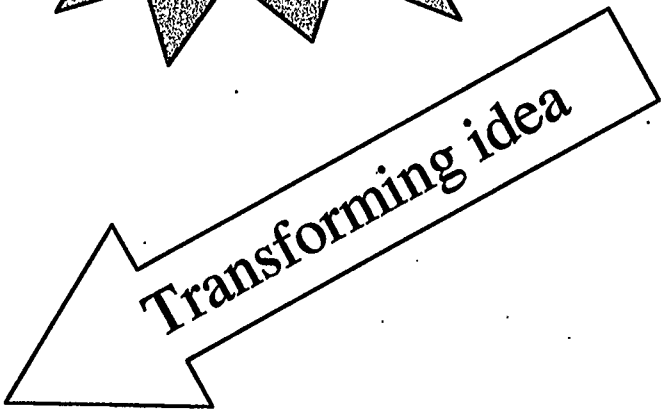
Copyright (c) 2000 Richard Brenner, Esther Derby

Stages of Change

- Late Status Quo



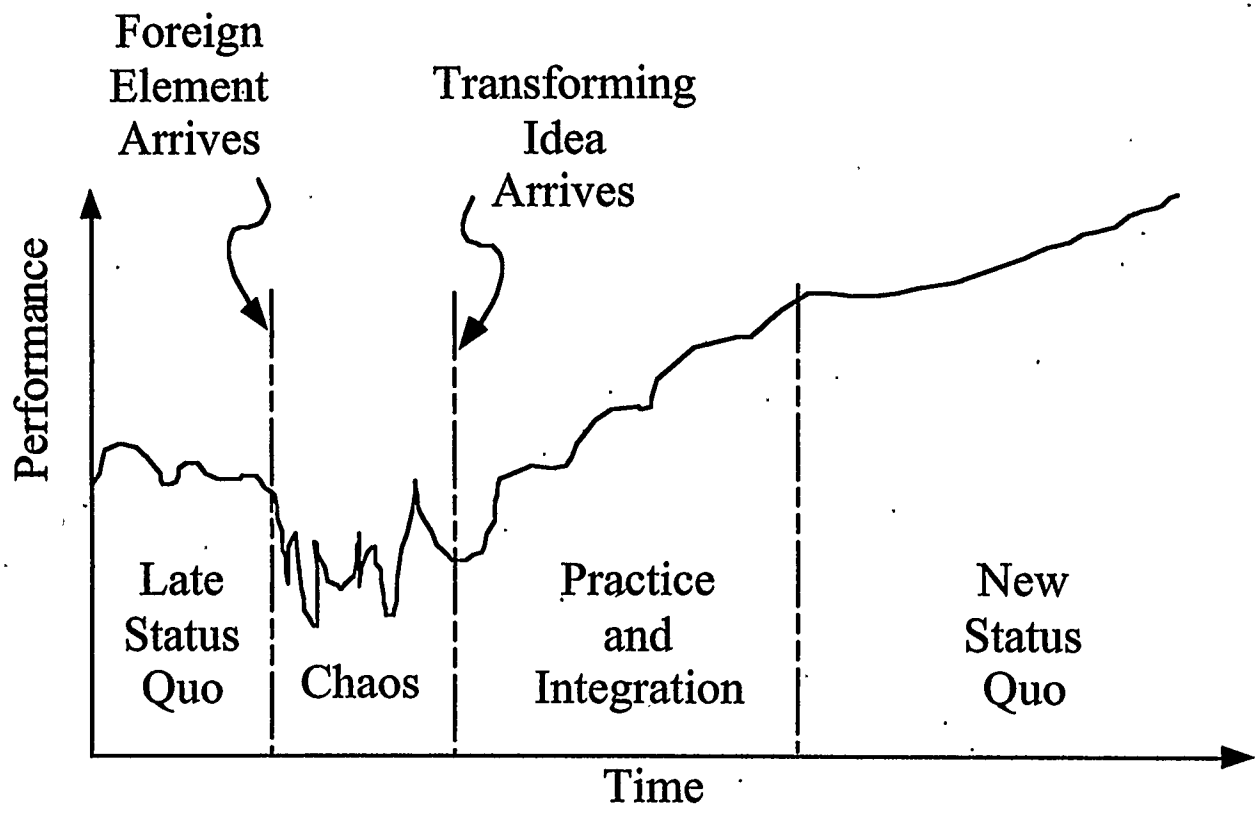
- Chaos



- Practice and Integration
- New Status Quo

Copyright (c) 2000 Richard Brenner, Esther Derby

Performance in the Satir Change Model



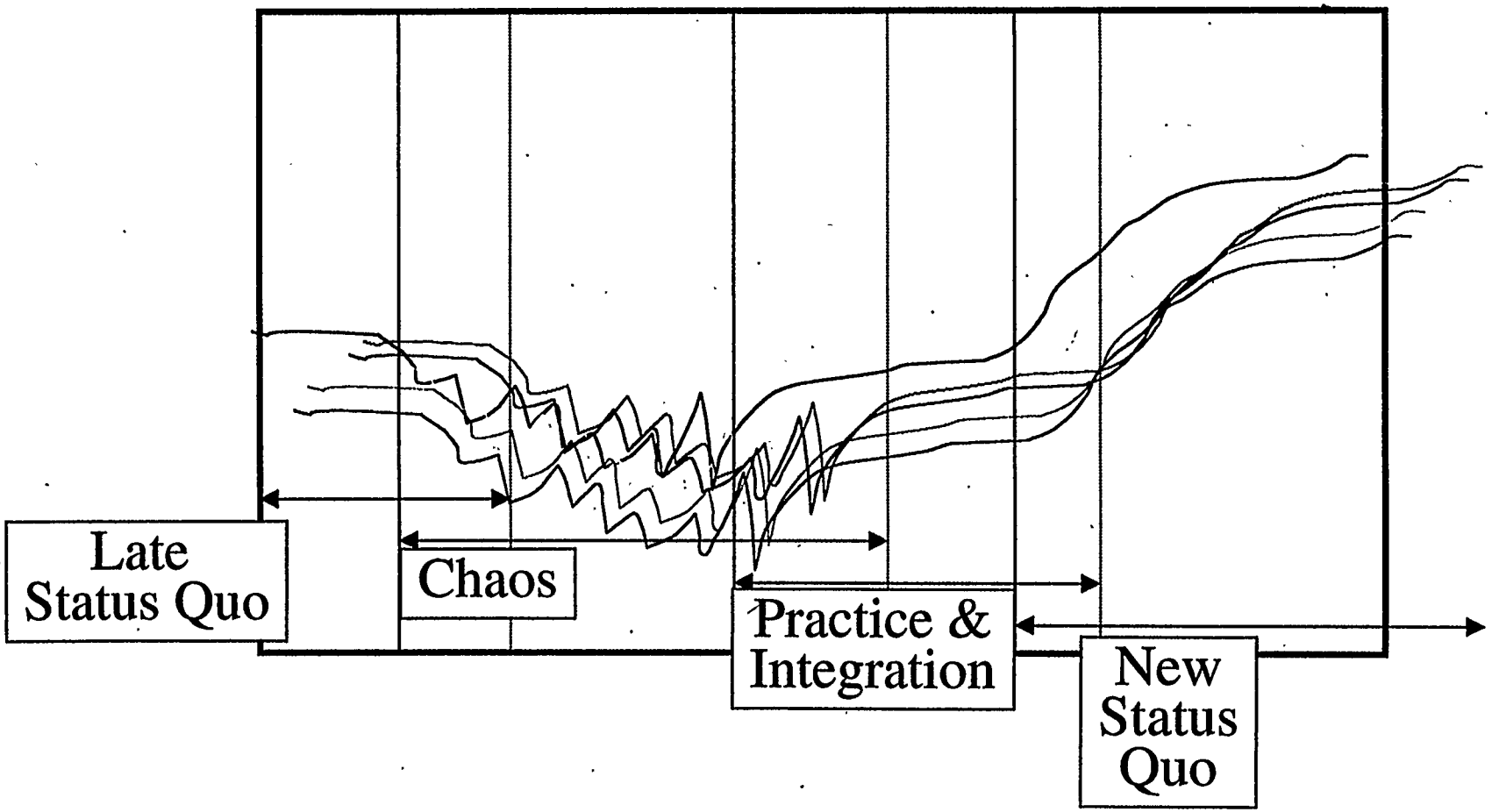
Copyright (c) 2000 Richard Brenner, Esther Derby

Individuals and Organizations

- Organizational change is the accumulation of changes within individuals
- Each of us changes at our own unique rate
- Organizational change is complete only when the majority of members have integrated the change

Copyright (c) 2000 Richard Bremner, Esther Derby

We Each Integrate Change In Our Own Time



Copyright (c) 2000 Richard Bremner, Esther Derby

Change Cycles Can Overlap

- Several changes can be happening at once
- They can start at different times
- They can move ahead at different rates
- Each new foreign element reduces the chance of integrating previous changes

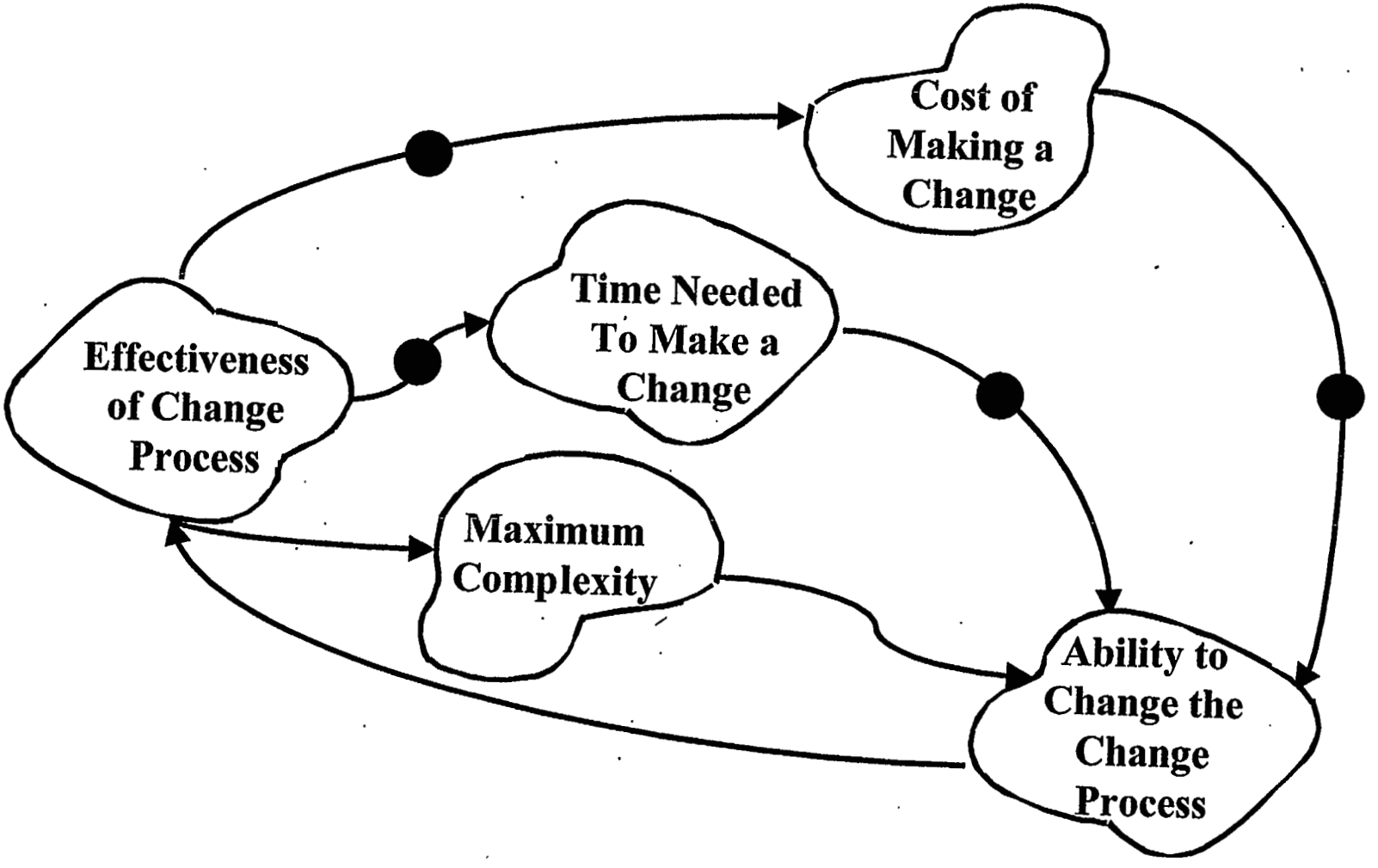
Copyright (c) 2000 Richard Brenner, Esther Derby

Changing the Change Process

- Improving the change process requires understanding Change
- This is Meta-Change
- Meta-Change affects how all other changes happen
- High impact, high return on effort

Copyright (c) 2000 Richard Bremner, Esther Derby

Changing the Change Process



Copyright (c) 2000 Richard Bremner, Esther Derby

How Do You Change The Change Process?

- Educate everyone in the Satir Change model
  - Knowing what to expect makes it easier to manage
  - Knowing that there will be a dip in performance helps manage expectations
- Sequence and space changes
  - Address first those changes that make change easier
  - Make sure that each change is largely integrated before the next foreign element
- Realize that change happens one person at a time
  - Know who is likely to support each change
  - Know who can kill the effort

Copyright (c) 2000 Richard Brenner, Esther Derby

Change Communications

- Communication is two-way
- Let people know what will change and what will remain the same
- Let people know what part they will play
- Give people a vision of what the new situation will be

Copyright (c) 2000 Richard Brenner, Esther Derby

Reframe Your Time Frame

- It takes time for a change to be integrated
- Continue to support it until it is well-established
- Set small, short-term goals
- Celebrate achievements
- Reinforce

Copyright (c) 2000 Richard Brenner, Esther Derby



Using Object - Oriented Metrics to Support Reliability

Software Assurance Technology Center

Goddard Space Flight Center - NASA

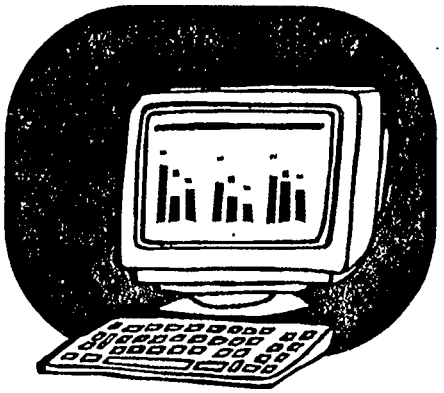
http://satc.gsfc.nasa.gov

Ruth Stapko	**rstapko@pop300.gsfc.nasa.gov**	**301 286 0101**
Dr. Linda Rosenberg	**linda.rosenberg@gsfc.nasa.gov**	**301 286 0087**
Al Gallo	**al.gallo@gsfc.nasa.gov**	**301 286 8012**

Software Assurance Technology Center

Overview

- Introduction
  - Metrics Basics
- Product Metrics
  - Product Quality Attributes
  - Risks to Product Quality
- Object Oriented Metrics
  - OO Quality Attributes
  - SATC's OO Metrics
  - Establishing Baselines
- Metrics & Reliability
  - Developing Standards
- Conclusion

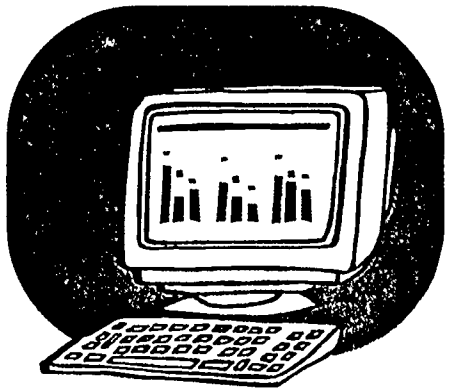




NASA SATC
Software Assurance Technology Center

Introduction

- *Introduction*
  - Metrics Basics
- Product Metrics
  - Product Quality Attributes
  - Risks to Product Quality
- Object Oriented Metrics
  - OO Quality Attributes
  - SATC's OO Metrics
  - Establishing Baselines
- Metrics & Reliability
  - Developing Standards
- Conclusion





Some Characteristics of a Good Metric

1 **Goal oriented**
2 **Precisely defined**
3 **Repeatable**
4 **Reproducible**
5 **Orthogonal**
6 **Cost effective**

1 - Goal Oriented Metrics

Basili s Goal-Question-Metric method:

Three Steps:

1. **Generate a set of *goals* based upon the needs of the organization.**
2. **Derive a set of *questions.***
3. **Develop a set of *metrics* which provide the information needed to answer the questions.**

2 - Precise Definition

Quiz:

What is a Line of Code?

What is a Requirement?

What is Defect Density?

What is a Defect?

What is Person-week of Effort?

What is Comment Percentage?

NASA SATC
Software Assurance Technology Center

3 - Repeatable Metrics

Repeatability: The extent to which repeated measurements of a particular object with a particular instrument produces (or would produce) the same value.

Repeatability risks:

- **Poor tool configuration**
- **Data archiving & retention problems**
- **Inadequate record keeping**

4 - Reproducible Metrics

Reproducibility: The extent to which measurement of an object varies when taken by different people using the same measurement tool.

Reproducibility risks:

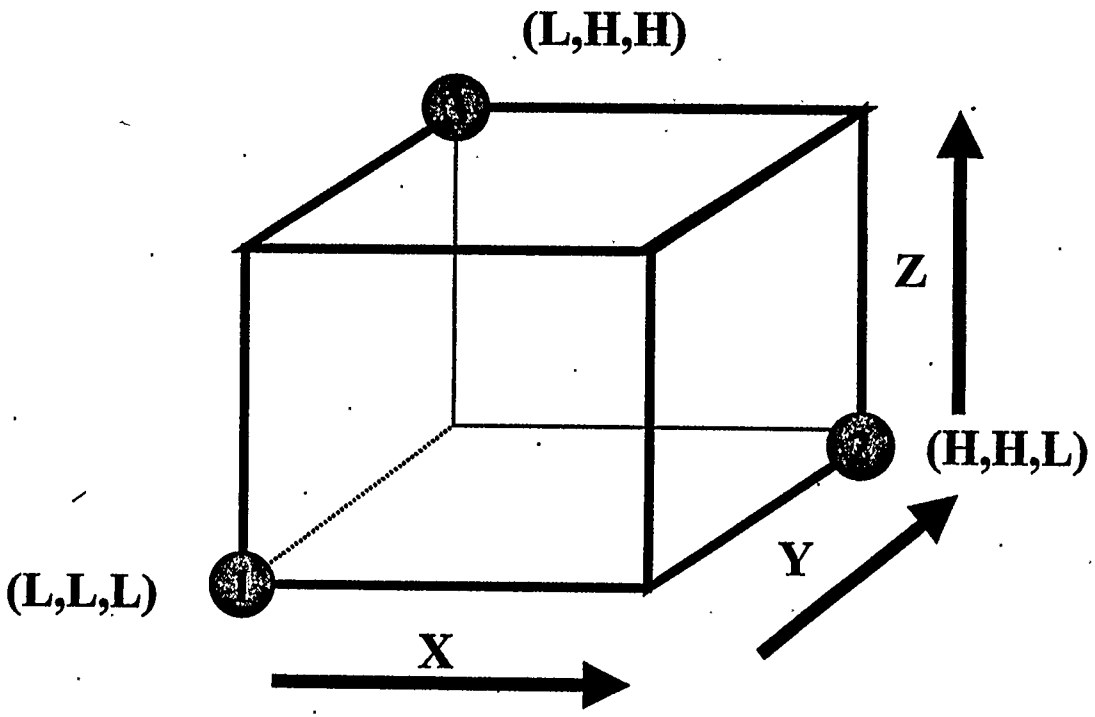
- **Poorly defined metrics (KLOC, defect density)**
- **Metrics that depend on human judgement (Function Points, effort)**

NASA SATC
Software Assurance Technology Center

5 - Orthogonal Metrics

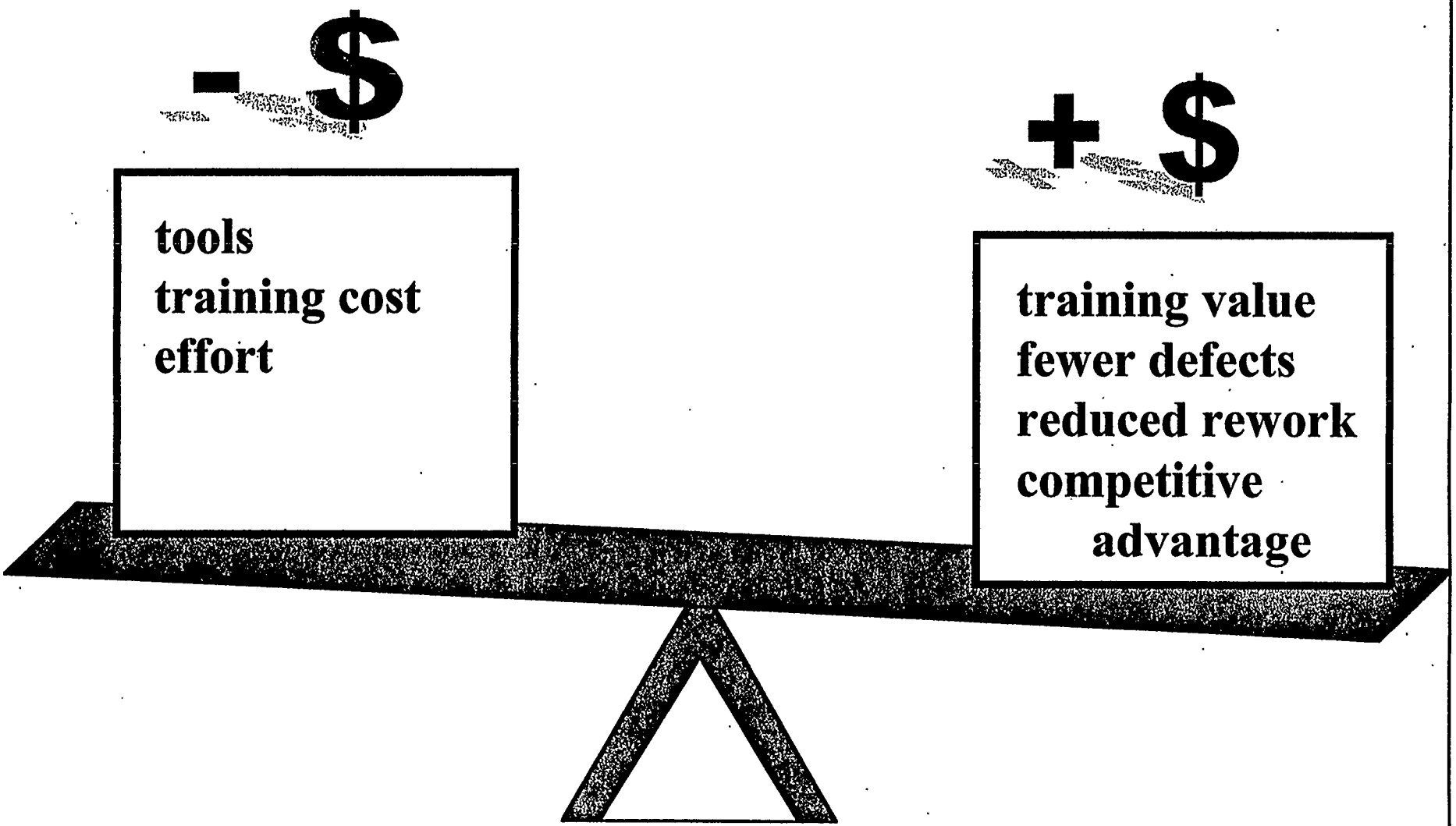
Orthogonality: The extent to which metrics measure *different* attributes.

Collecting correlated metrics wastes time and money.



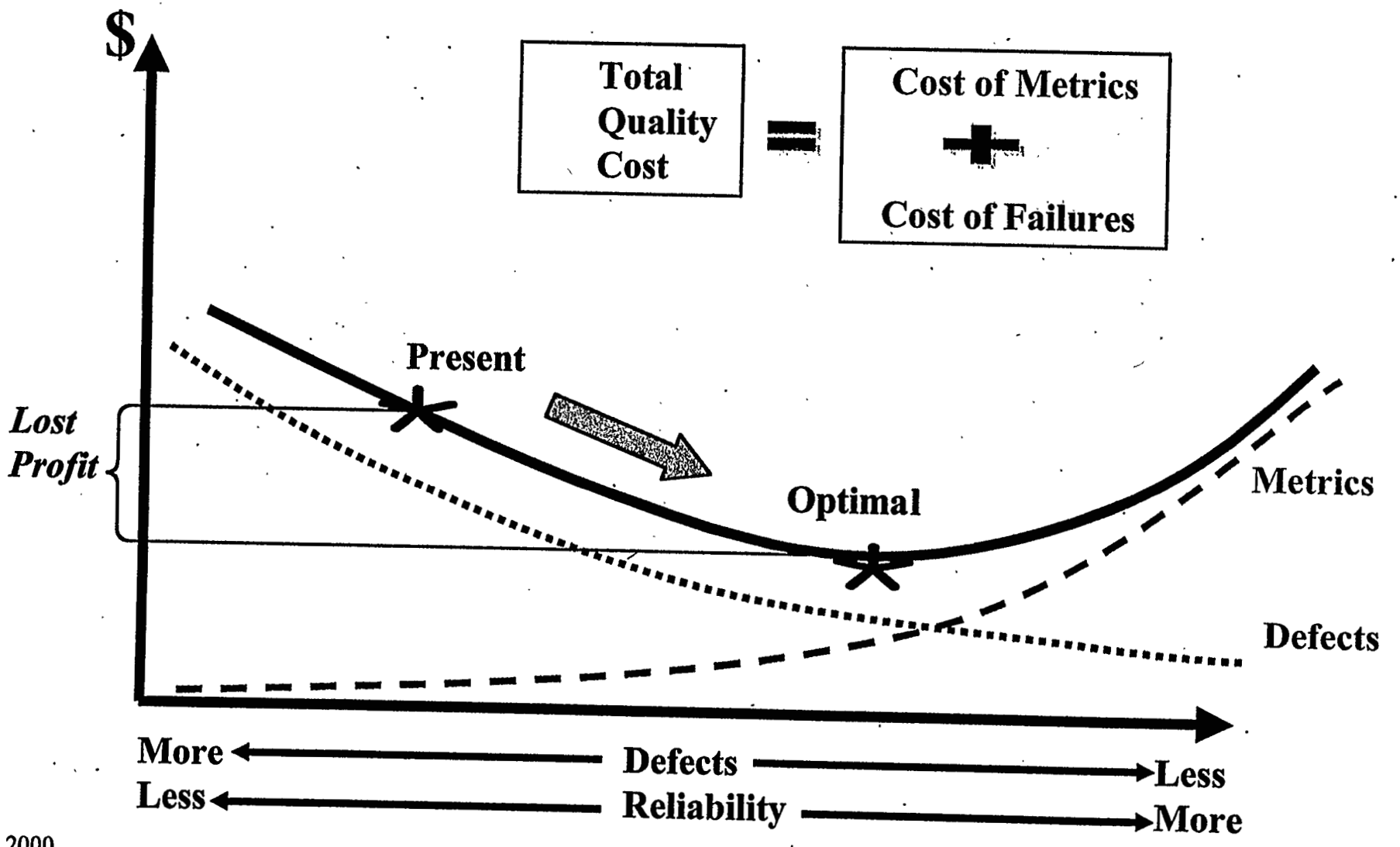
NASA SATC
Software Assurance Technology Center

6 - Cost Effective Metrics



NASA SATC
Software Assurance Technology Center

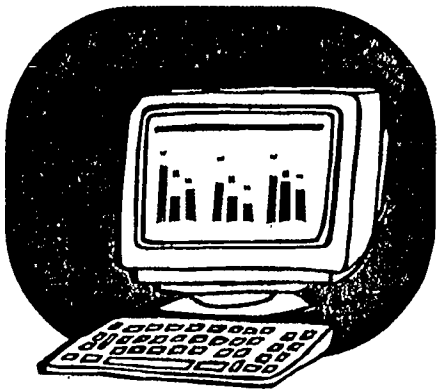
Cost of Quality



NASA SATC
Software Assurance Technology Center

Product Metrics

- Introduction
  - Metrics Basics
- *Product Metrics*
  - Product Quality Attributes
  - Risks to Product Quality
- Object Oriented Metrics
  - OO Quality Attributes
  - SATC's OO Metrics
  - Establishing Baselines
- Metrics & Reliability
  - Developing Standards
- Conclusion





NASA SATC
Software Assurance Technology Center

Product Quality Goals

Maintainability - *Can I fix it?*

Flexibility - *Can I change it?*

Testability - *Can I test it?*

Reusability - *Will I be able to reuse some of the software?*

Reliability - *Can I expect it to work for a long time?*

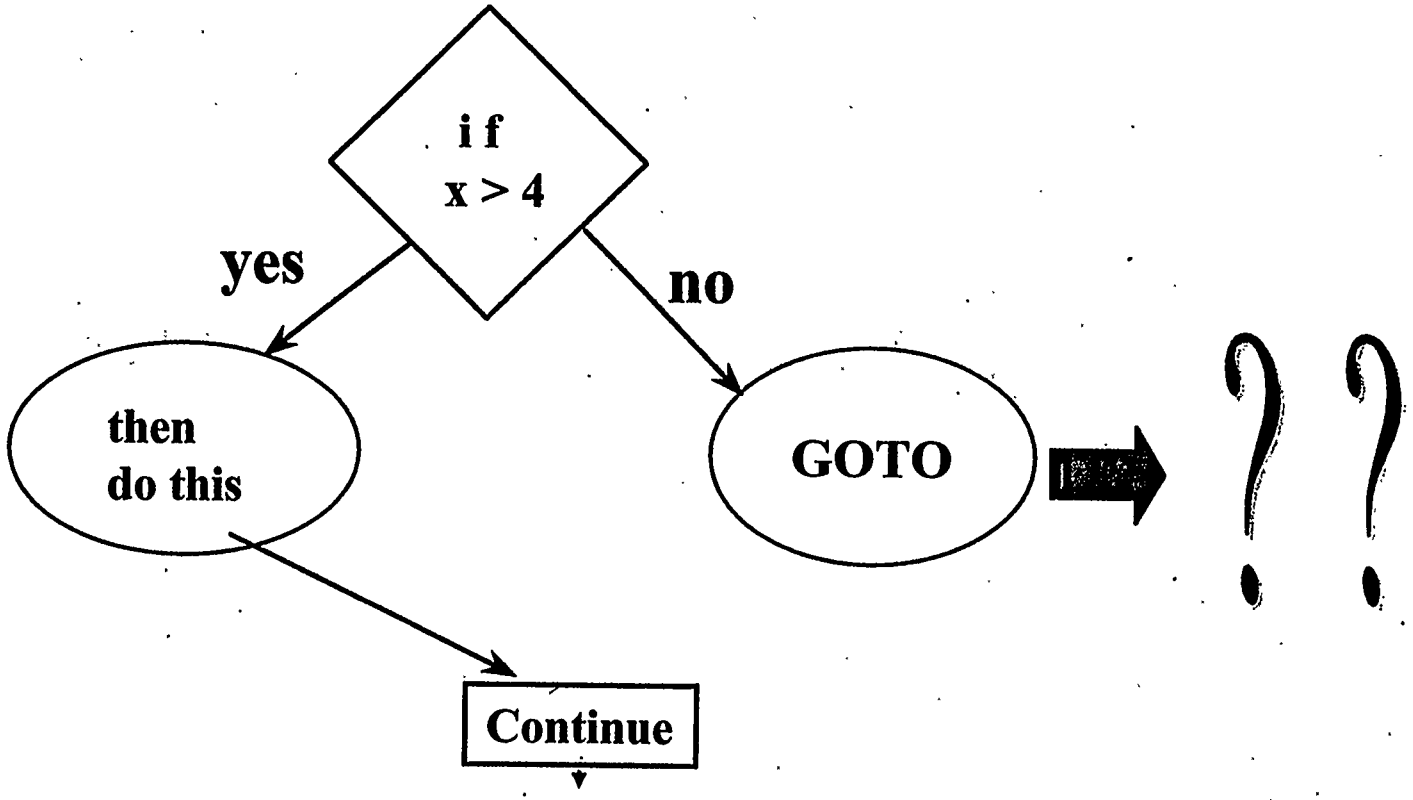
NASA SATC
Software Assurance Technology Center

Risks to Product Quality

- Large, complex modules with poor branching
- Inadequate internal documentation decreases usability and increases maintainability costs
- Incomplete or disjointed calling pattern
- Modules are too specialized - unnecessary hard coding
- System is too generic - contains unused modules

NASA SATC
Software Assurance Technology Center

Branching & Complexity

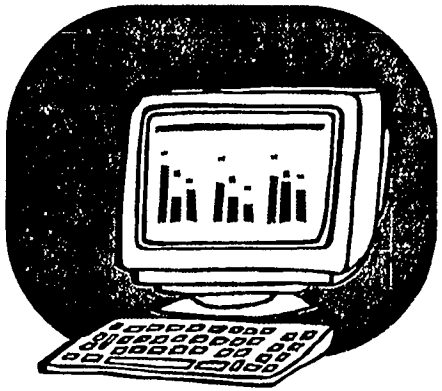


"Goto, n.: A programming tool that exists to allow structured programmers to complain about unstructured programmers." . . . Ray Simard

NASA SATC
Software Assurance Technology Center

Object Oriented Metrics

- Introduction
  - Metrics Basics
- Product Metrics
  - Product Quality Attributes
  - Risks to Product Quality
- *Object Oriented Metrics*
  - OO Quality Attributes
  - SATC's OO Metrics
  - Establishing Baselines
- Metrics & Reliability
  - Developing Standards
- Conclusion





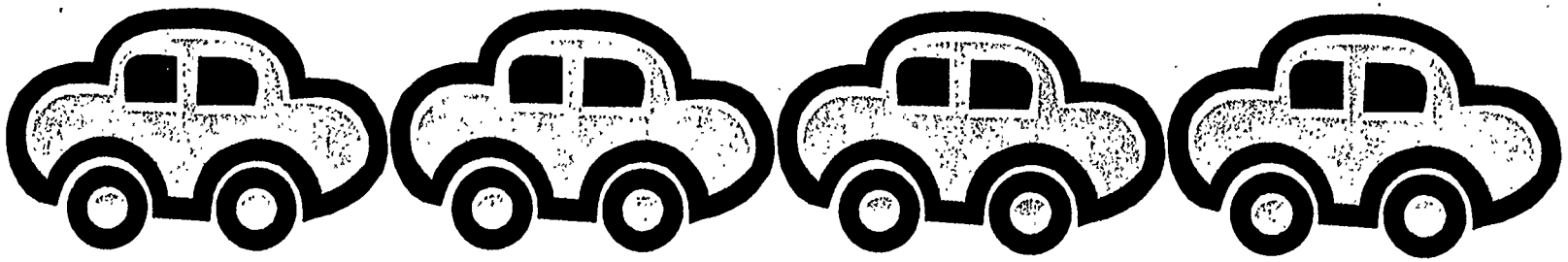
The 3 Most Important Goals of Object Oriented Development

1. Reuse! 2. Reuse! 3. Reuse!

OO classes : Software development

::

Interchangeable parts : Manufacturing



NASA SATC
Software Assurance Technology Center

Choosing Metrics

Cyclomatic Complexity
Lines of Code — *Traditional*
Comment Percentage

Weighted Methods per Class
Response for a Class — *Architecture*
~~**Lack of Cohesion of Methods**~~
Coupling Between Objects

Depth of Inheritance Tree — *Tree Structure*
Number of Children

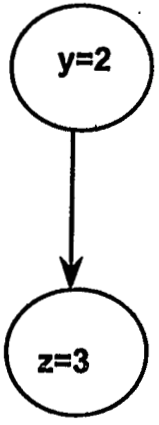
Traditional Metrics Work for Methods

- The Cyclomatic Complexity (McCabe) is used to evaluate the implementation of a method.
- Various Line Counts can be applied to methods. These include counting all physical lines of code, the number of statements and the number comment lines. Thresholds vary greatly depending on the language.
- Comment Percentage is calculated by the number of on-line and stand alone comments divided by the total lines of code less the number of blank lines.

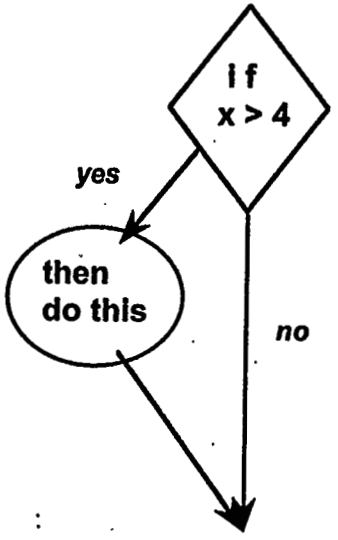
NASA SATC
Software Assurance Technology Center

Cyclomatic Complexity

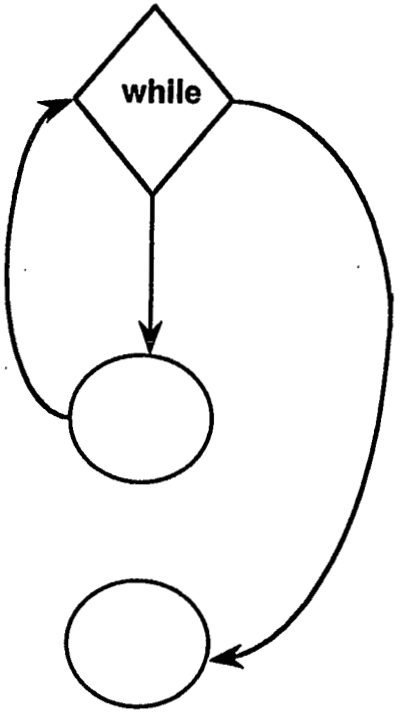
Number of Independent Test Paths = e - n + 2



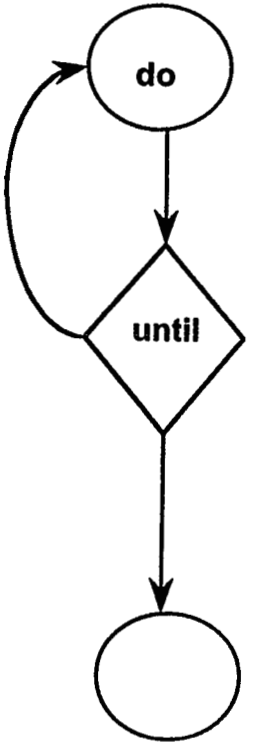
sequence:
1-2+2=1



if / then:
3-3+2=2



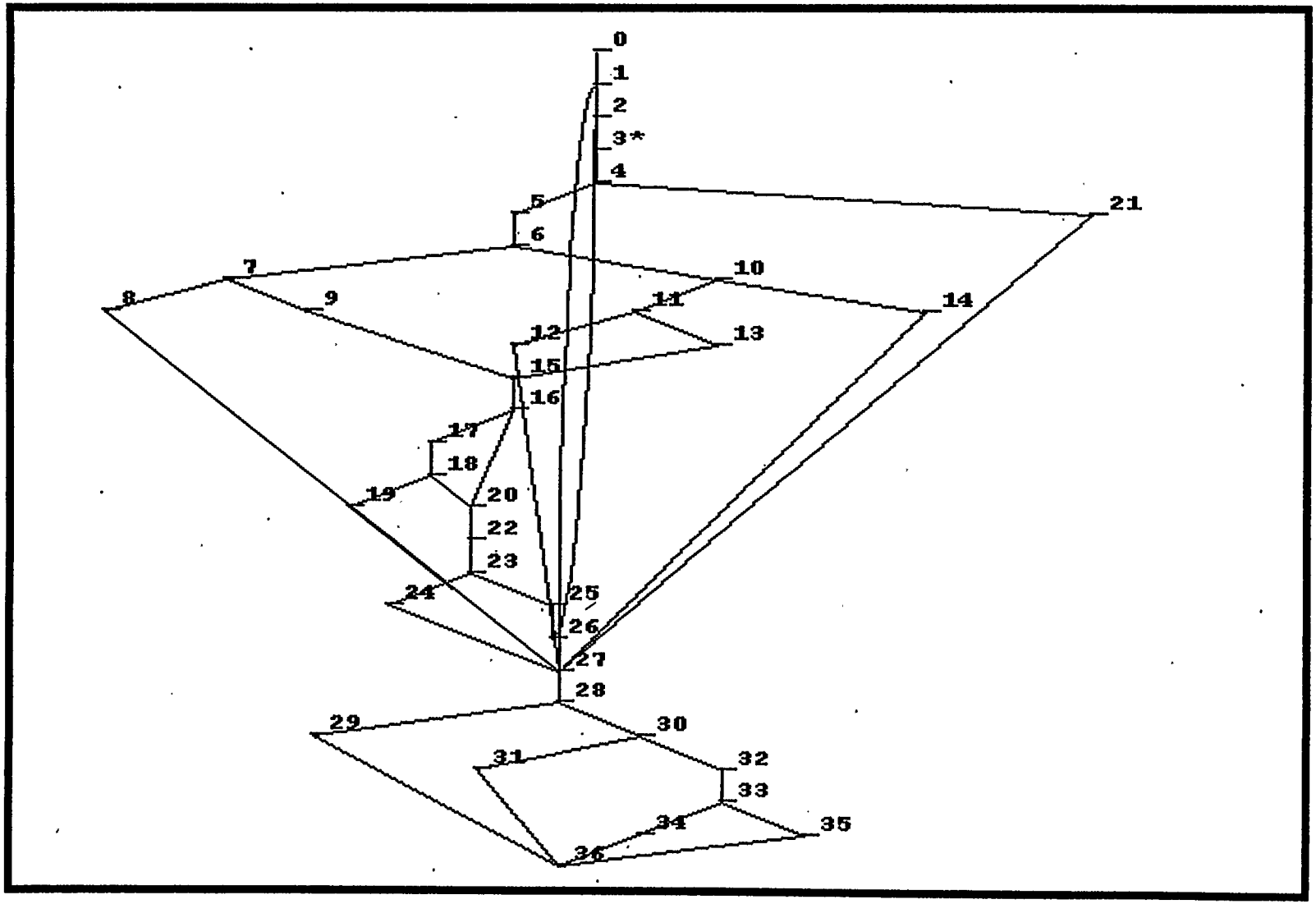
while loop:
3-3+2=2



until loop:
3-3+2=2

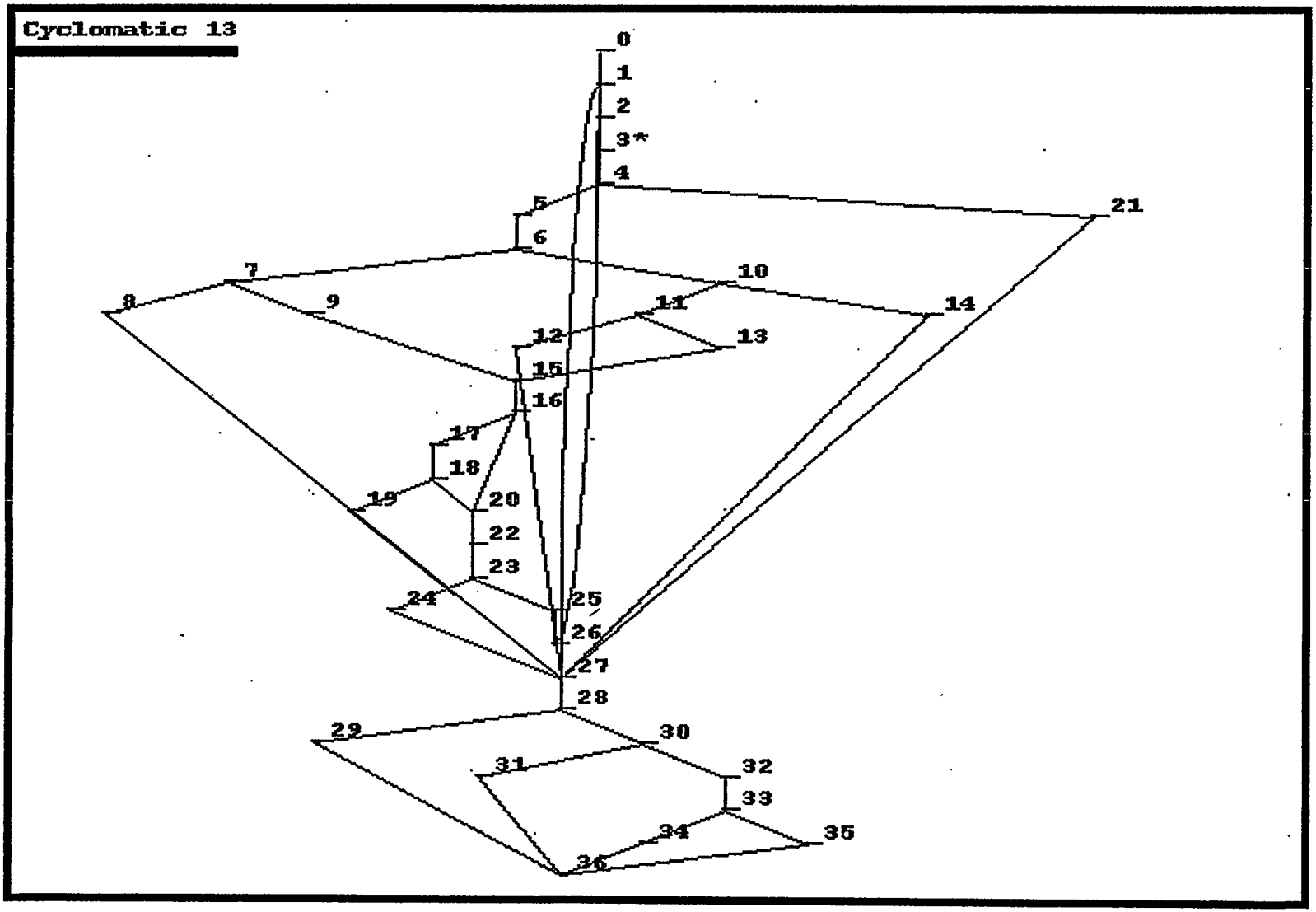
NASA SATC
Software Assurance
Technology Center

Cyclomatic Complexity



NASA SATC
Software Assurance Technology Center

Cyclomatic Complexity



NASA SATC
Software Assurance Technology Center

Size Metrics

Make sure you know what your tool is counting!

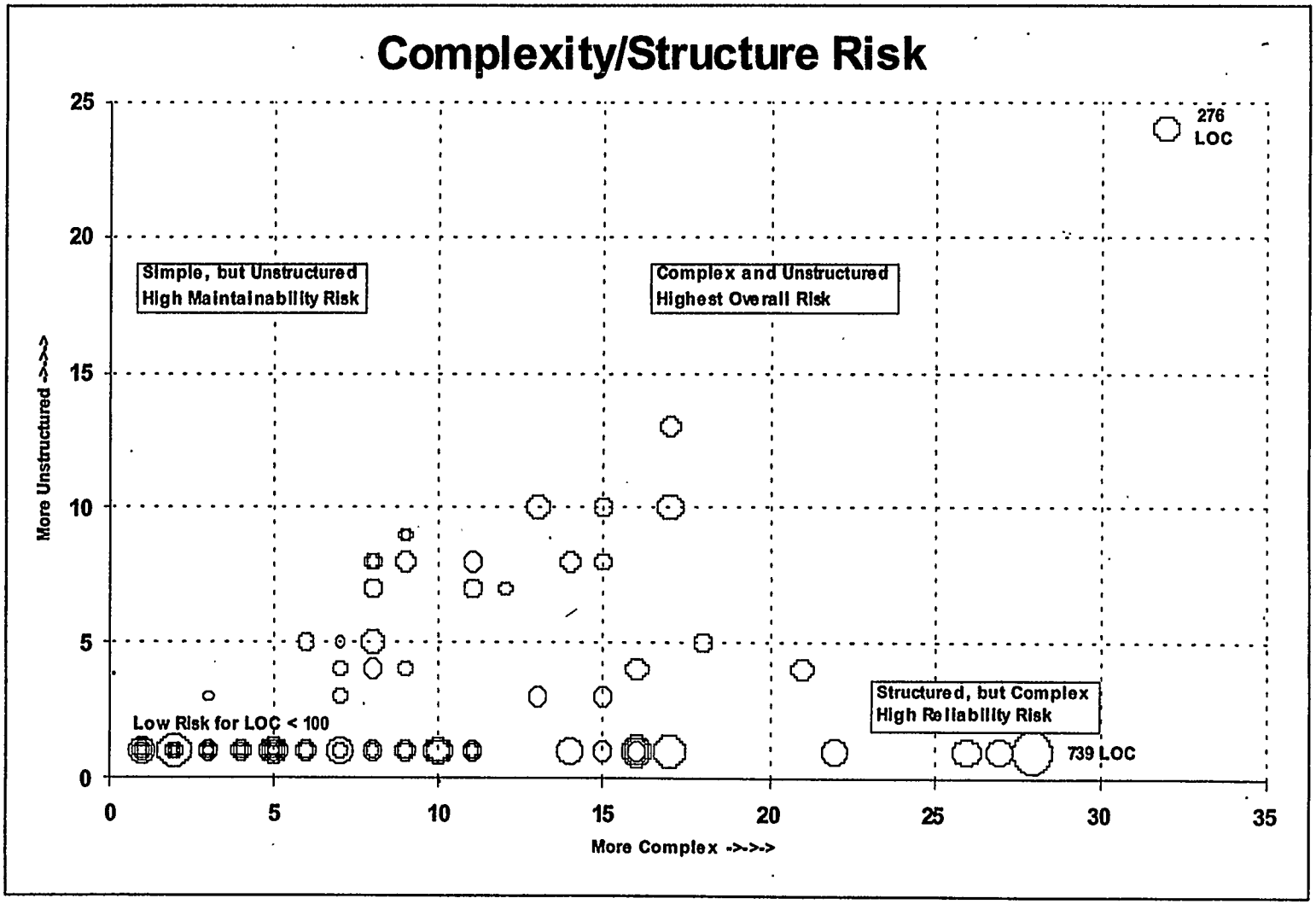
How are multiline statements counted?

How are multiline comments counted?

Are preprocessor directives (#include, With, Import) counted?

NASA SATC
Software Assurance Technology Center

Size Metrics - The Bubble Plot



Comment Quality

```
/*
Name:
Source file:
Author:
  JAS
Description:
   This program ...
 Modified:
   29 April 1999 : file created JAS
Assumptions, limitations & notes:
*/
```

NASA SATC
Software Assurance Technology Center

Comment Quality

Wanted:

A tool that can tell if a comment…

- **is all special characters (e.g. not really a comment)**
- **is part of standard file header information**
- **contains a date**
- **contains key words like change, fix, correct, add, revise**

Comment Quality

- Comment Percentage =

$$\frac{\text{Number of comments} * 100}{\text{LOC - blank lines}}$$

- READ the comments in a sample set of files.
- Base the minimum acceptable comment percentage on the quality of the comments and the complexity of the code.
- 20-30% comments is usually considered adequate.

NASA SATC
Software Assurance Technology Center

Architecture Metrics

The overall complexity of an OO system depends on

- **Within-class attributes**
  - **Number of methods**
  - **Method complexities**
- **Between-class attributes**
  - **Interclass dependencies**
  - **Message passing potential**

Within Class Metrics

- **Number of Methods (NOM) is the simple count of *methods* in a class.**

- **Weighted Methods per Class (WMC) is the sum of the method complexities in a class.**

NASA SATC
Software Assurance Technology Center

Between Class Metrics

- **Response for Class (RFC) is the number of unique methods that can be directly invoked in response to a message to an object of a class. This provides a "worst case" measure of the possible impact of a fault in a class.**

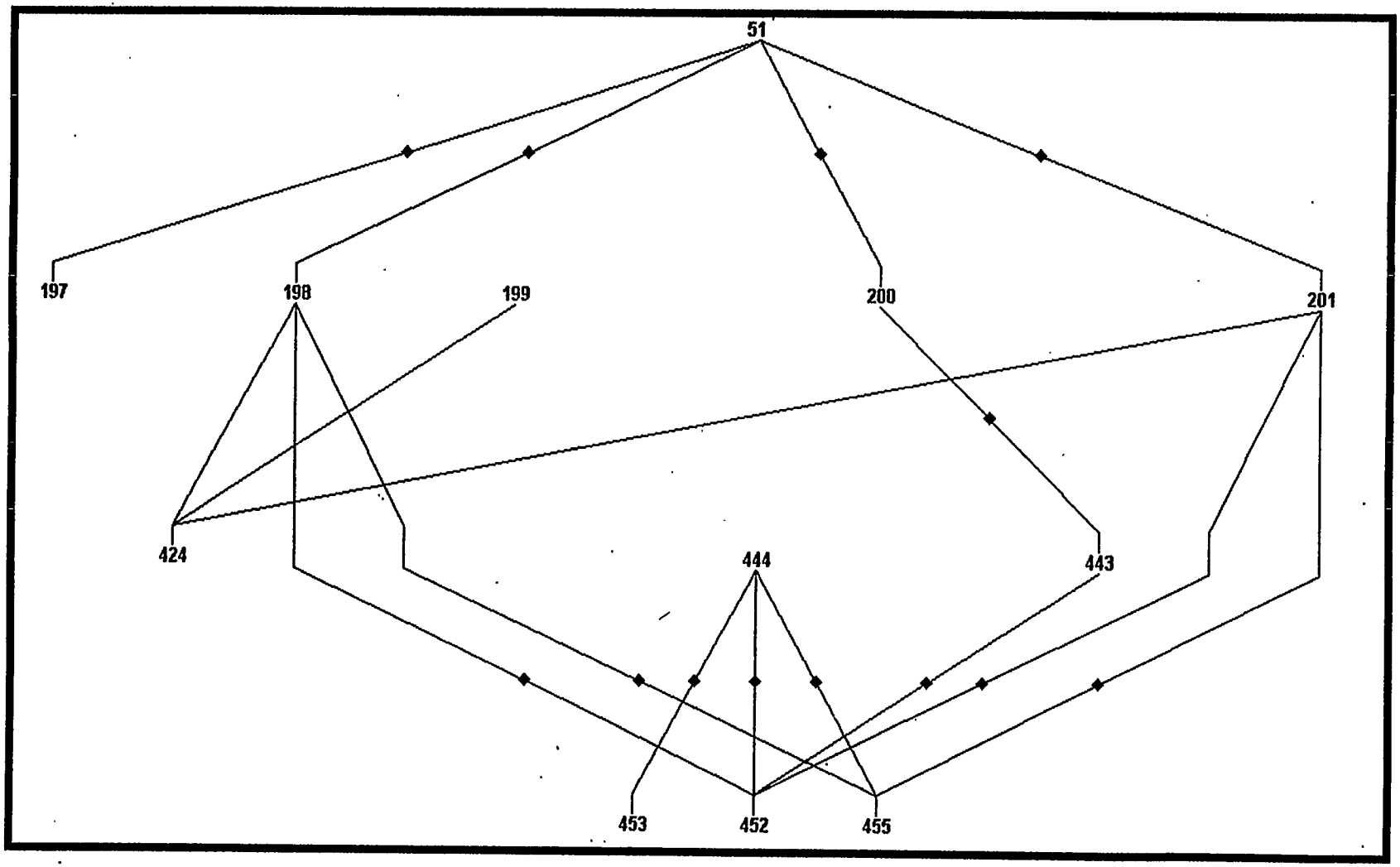
- **Weighted Methods per Class (WMC) is the sum of the method complexities in a class. The average method complexity for a class is WMC/NOM.**

Tree Structure Metrics

- **Depth of Inheritance is the number of edges between a class and the root class of the inheritance tree.**

- **Number of Children is the number of edges starting from a class in the inheritance tree.**

A Class Inheritance Tree



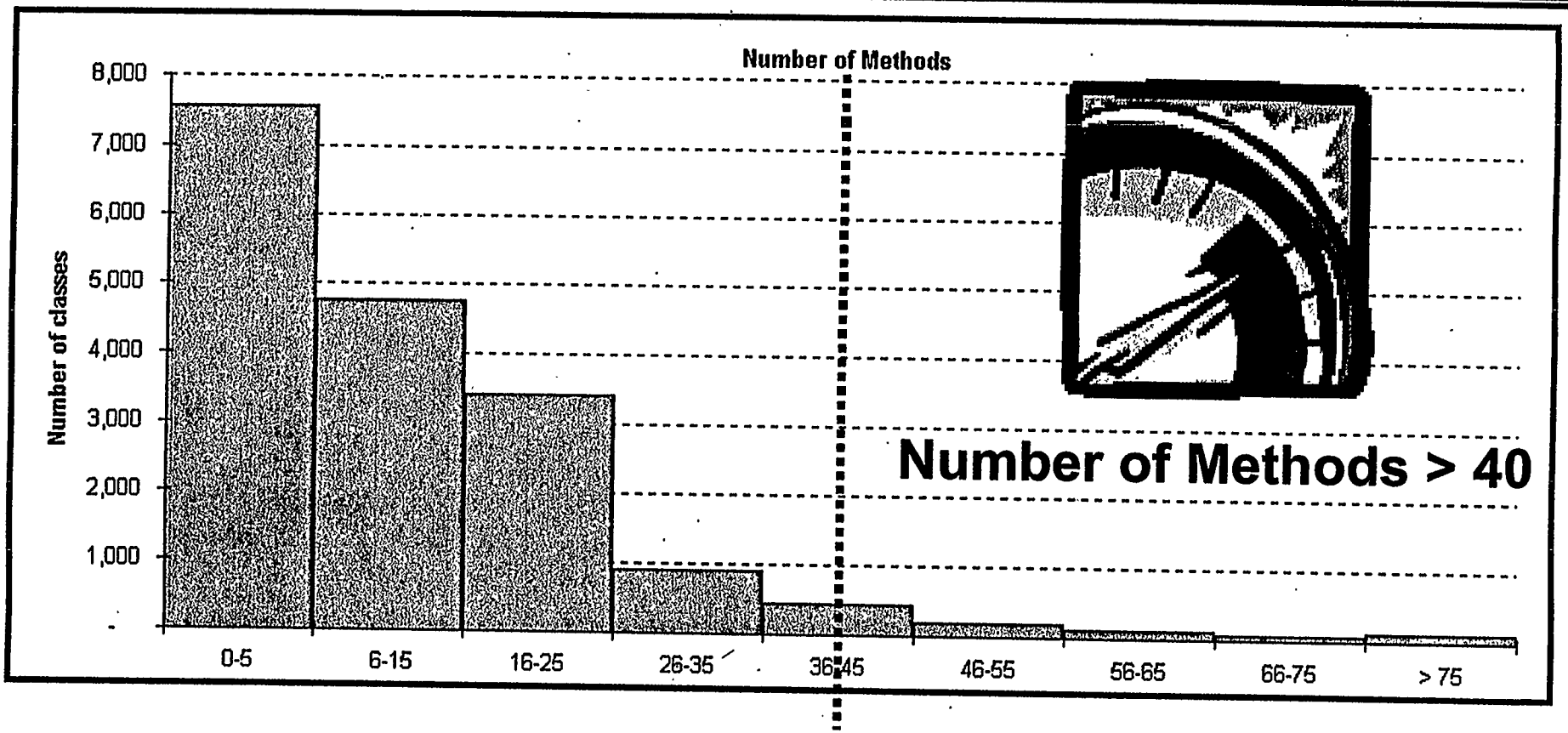
NASA SATC
Software Assurance Technology Center

Establishing Baselines



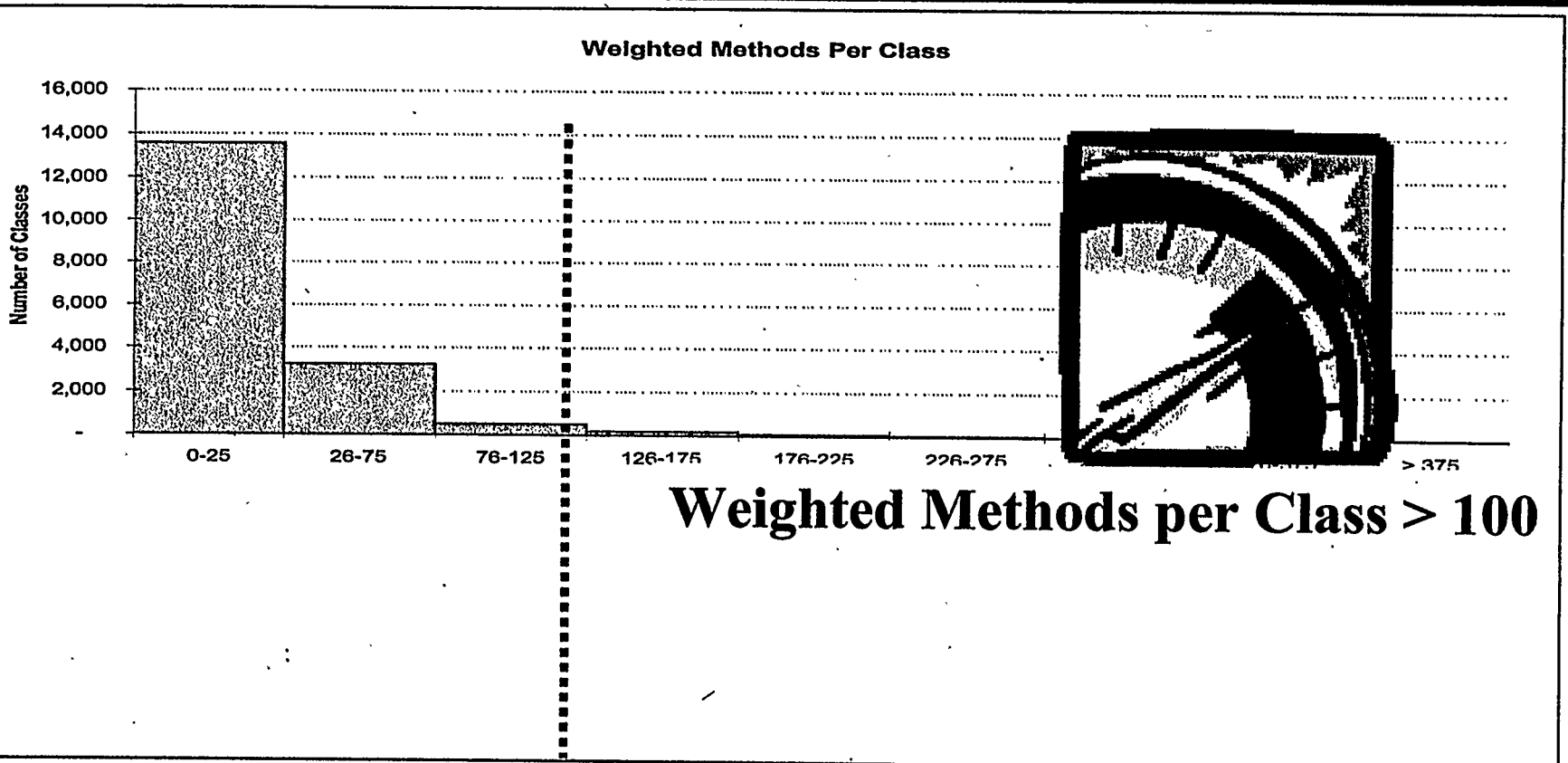
NASA SATC
Software Assurance Technology Center

Number of Methods



High number of methods may have greater impact on children through inheritance. May also indicate application specific, decreasing reusability.

Weighted Methods per Class



Weighted Methods per Class > 100

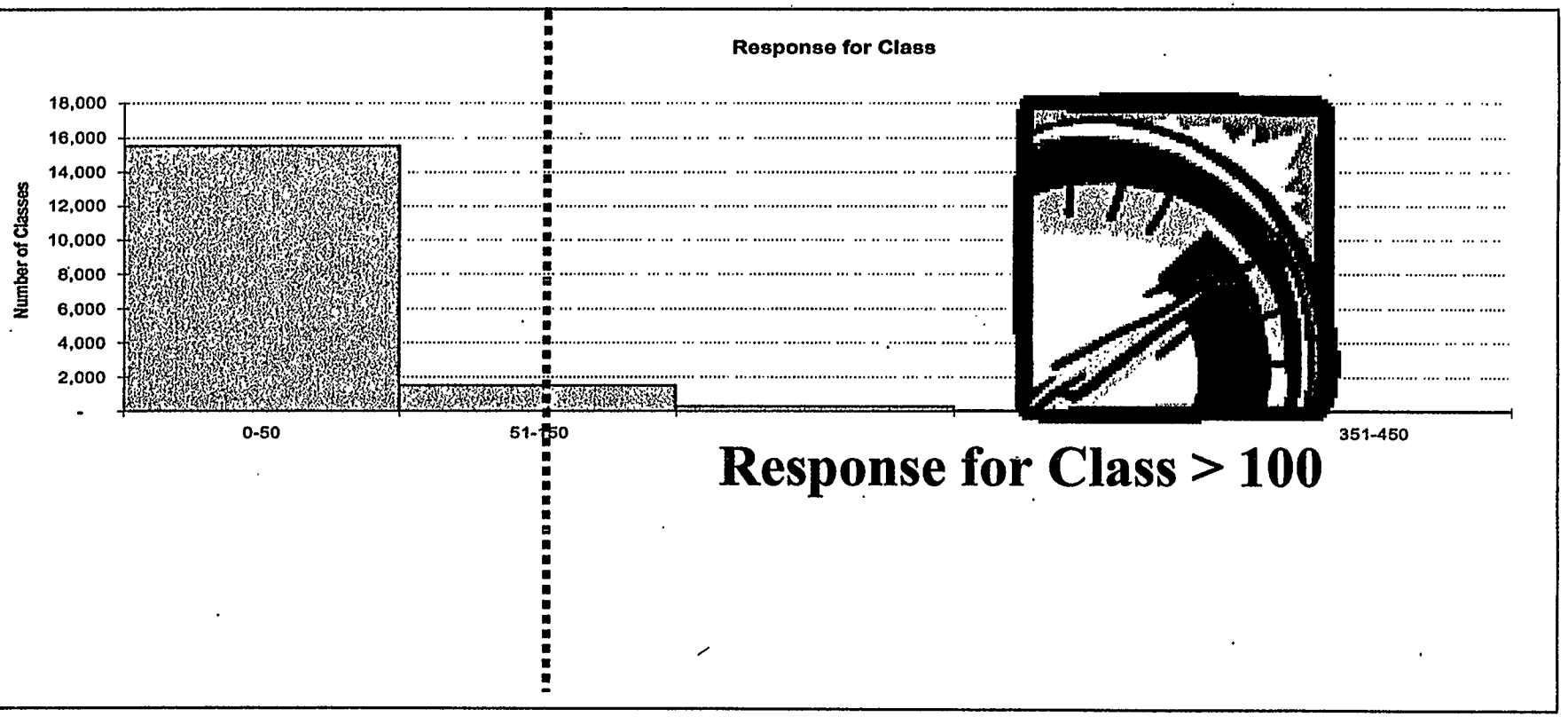
Complexity of a method ==> 1-5 best, up to 10 acceptable

Number of methods in a class ==> 1- 20

WMC ==> < 100 (5 complexity * 20 methods) ; should not exceed 200 (10 * 20)

Software Assurance Technology Center

Response for Class I

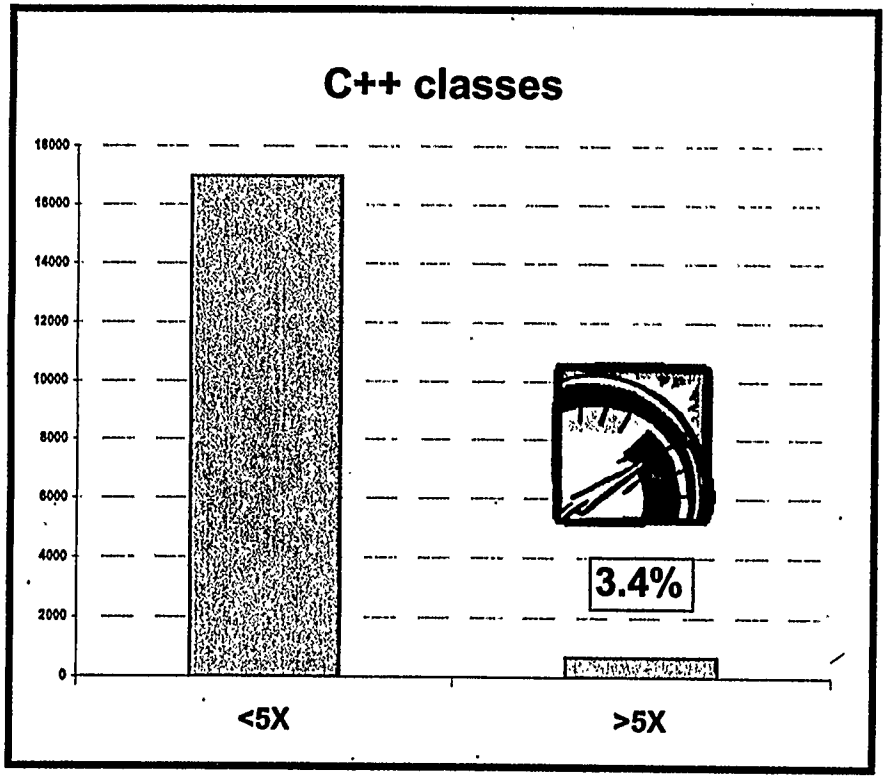


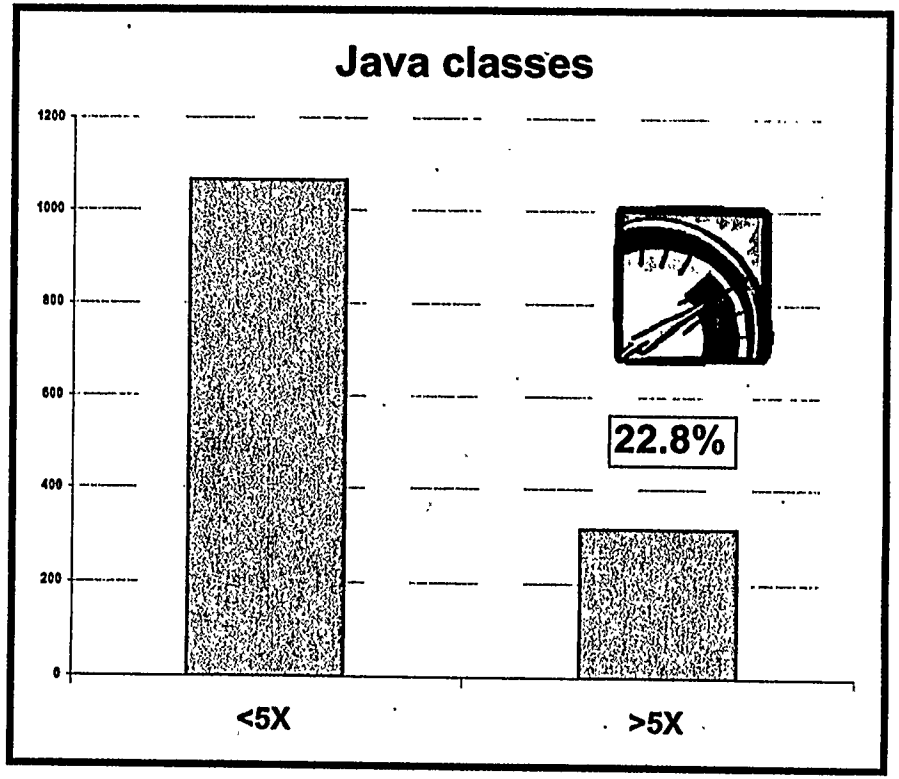
Response for Class > 100

Response for a Class < 50 , acceptable up to 100.
> 100 ==> greater complexity and decreased understandability changes very difficult due to the potential for a ripple effect.

NASA SATC
Software Assurance Technology Center

Response for Class II

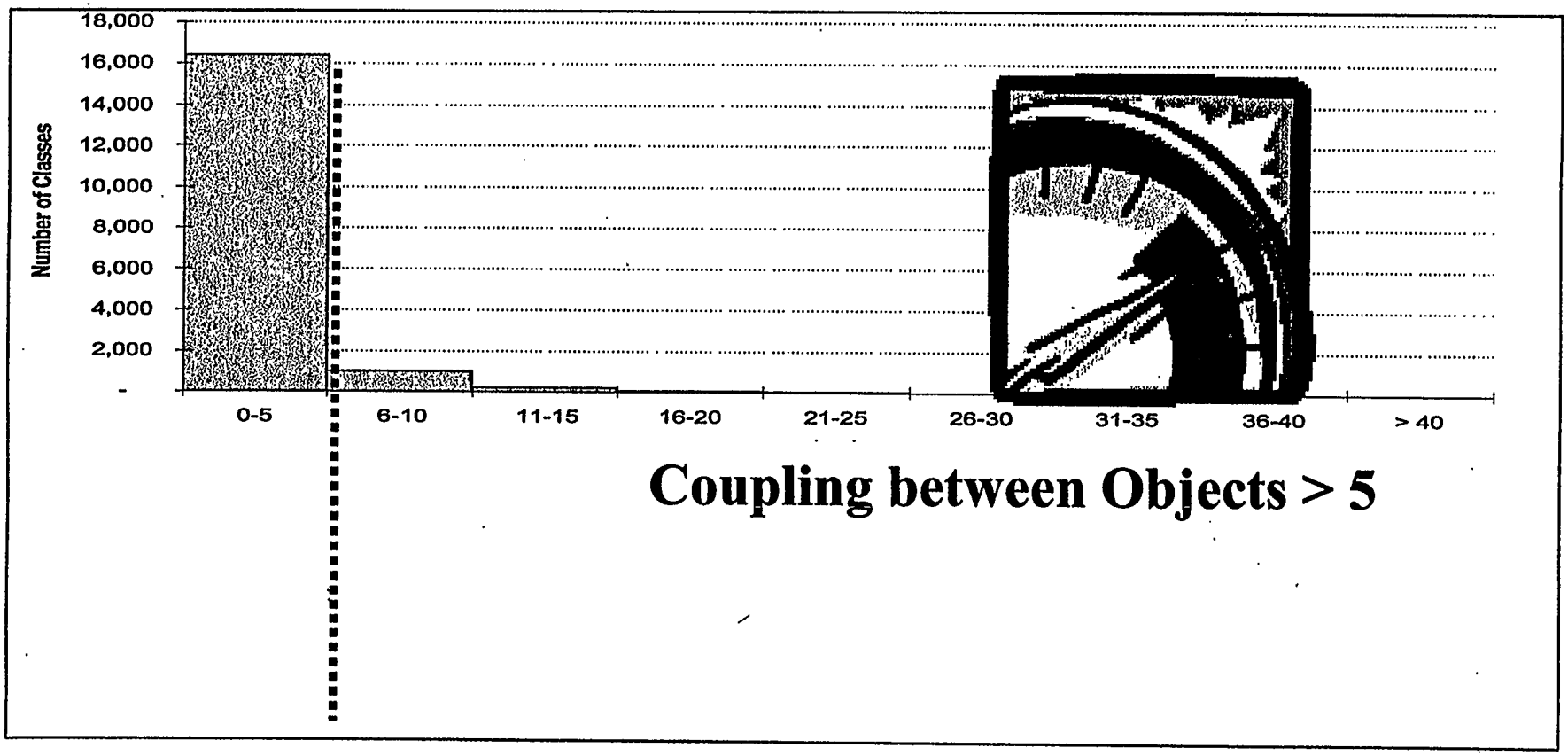




Classes with RFC Greater than 5 times #Methods

NASA SATC
Software Assurance Technology Center

Coupling Between Objects



Higher CBO indicates classes that my be difficult to understand
Decreased reuse and increased maintenance.

NASA SATC
Software Assurance Technology Center

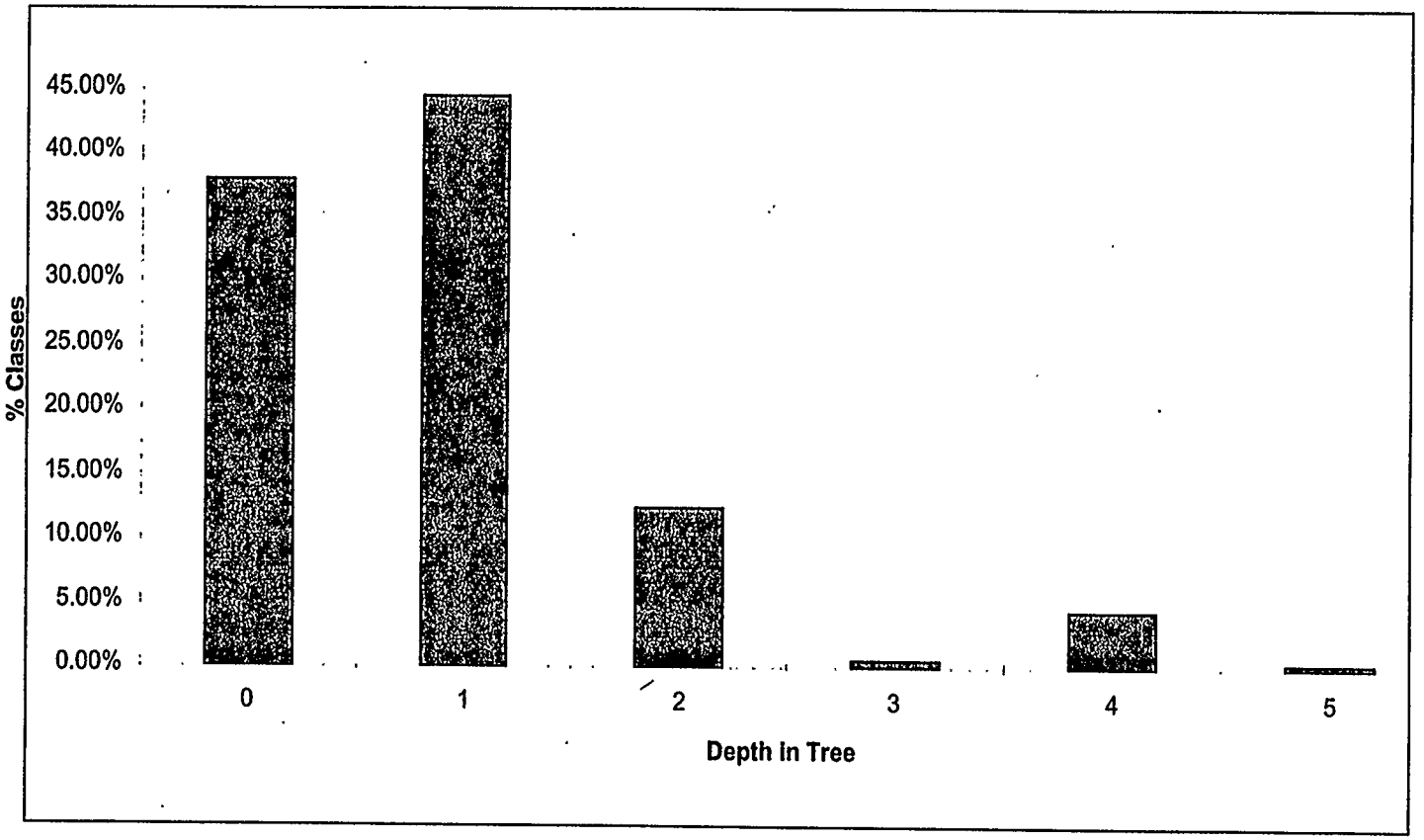
Class Hierarchy Metrics

- **The Depth of a Class within the inheritance hierarchy is the maximum number of steps from the class node to the root of the hierarchy and is measured by the number of ancestor classes.**

- **The Number of Children is the number of immediate subclasses subordinate to a class in the hierarchy.**

NASA SATC
Software Assurance Technology Center

Depth in Tree

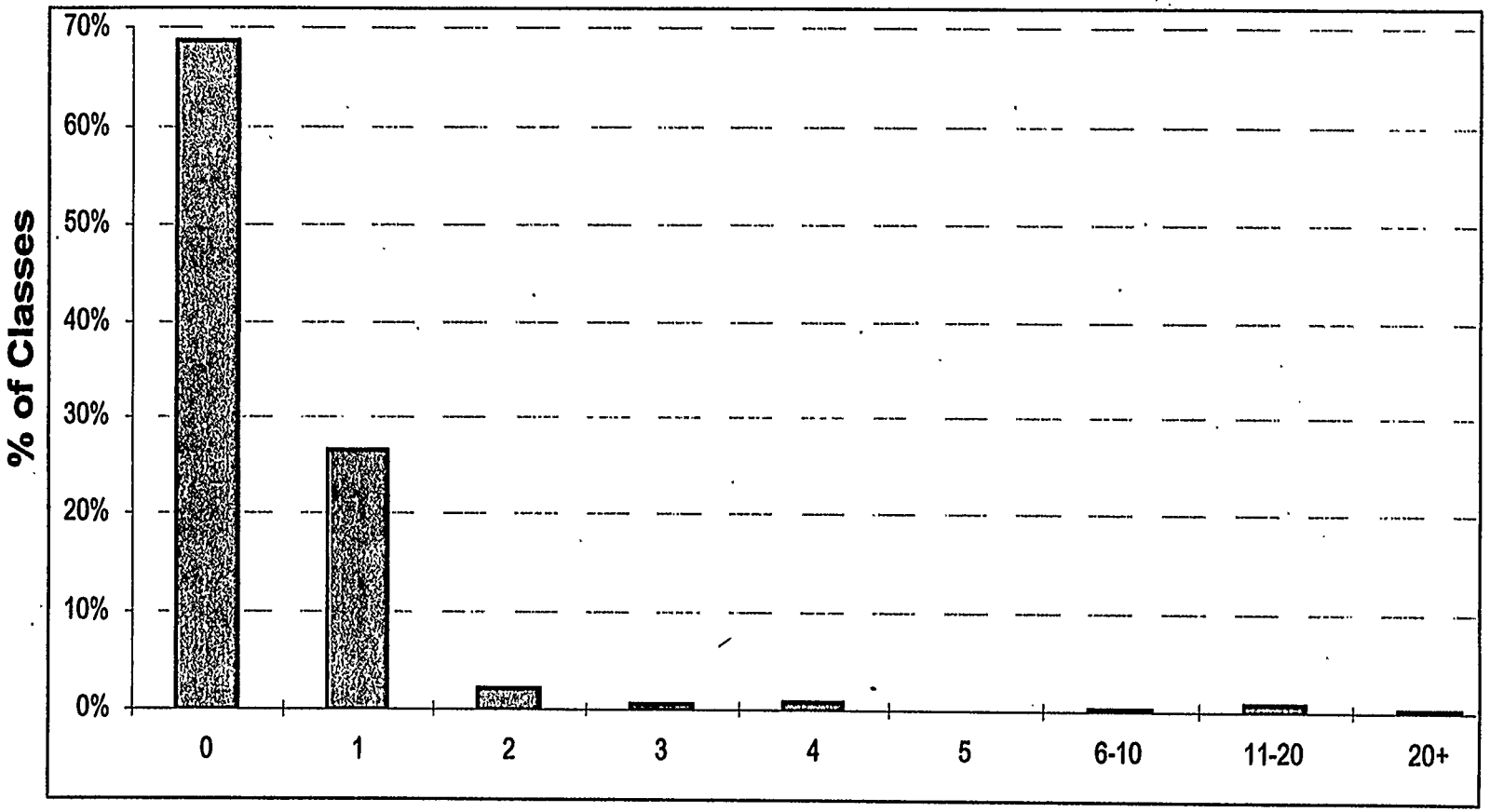


DIT = 0 ==> "root"

Higher percentages for DIT's of 2 and 3 show a higher degree of reuse.

NASA SATC
Software Assurance Technology Center

Number of Children



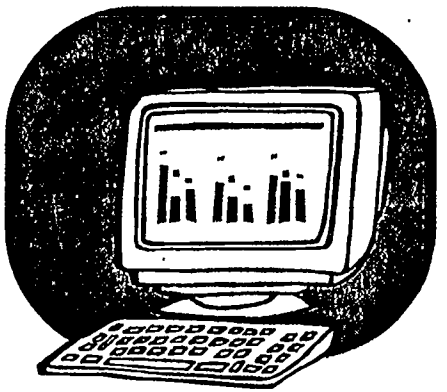
NOC = 0 ==> leaf nodes

Higher the number of child classes, the more benefit there may be due to inheritance

NASA SATC
Software Assurance Technology Center

Metrics & Reliability

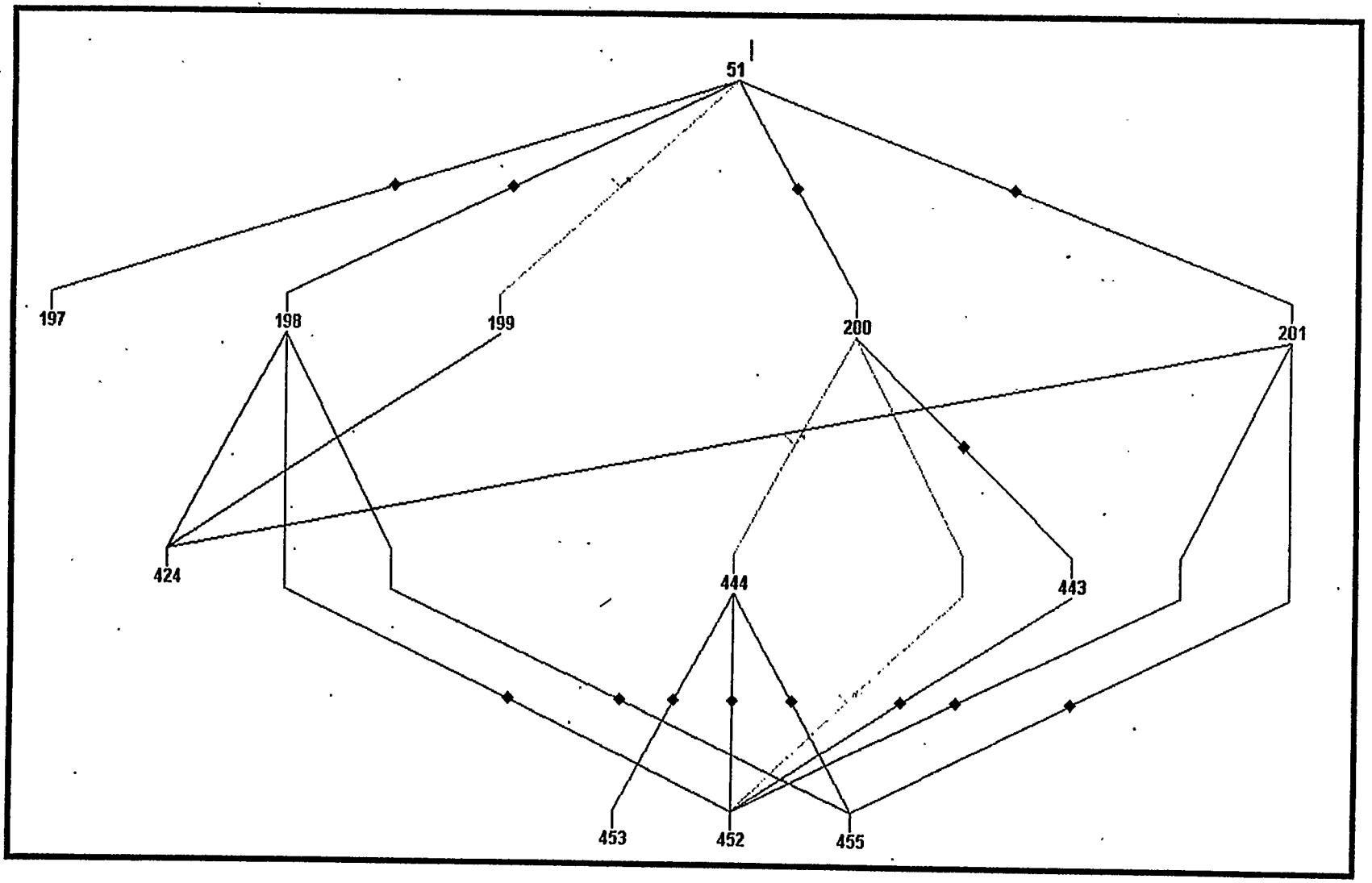
- Introduction
  - Metrics Basics
- Product Metrics
  - Product Quality Attributes
  - Risks to Product Quality
- Object Oriented Metrics
  - OO Quality Attributes
  - SATC's OO Metrics
  - Establishing Baselines
- *Metrics & Reliability*
  - Developing Standards
- Conclusion





NASA SATC
Software Assurance
Technology Center

System Reliability

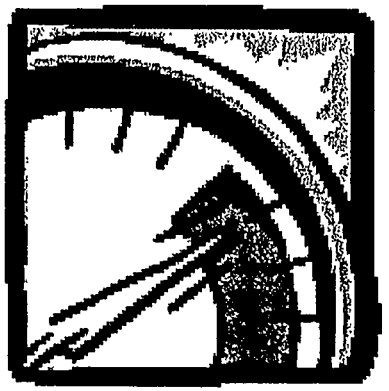


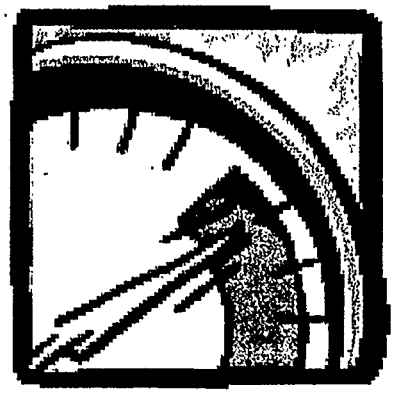
NASA SATC
Software Assurance Technology Center

Identifying Reliability Risks

Any class that meets *at least two* of the following criteria is flagged:

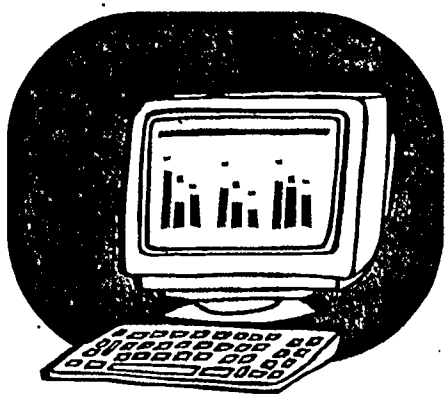
- **Response for Class > 100**
- **Coupling between Objects > 5**
- **Response for Class > 5 time the number of methods in the class**
- **Weighted Methods per Class > 100**
- **Number of Methods > 40**

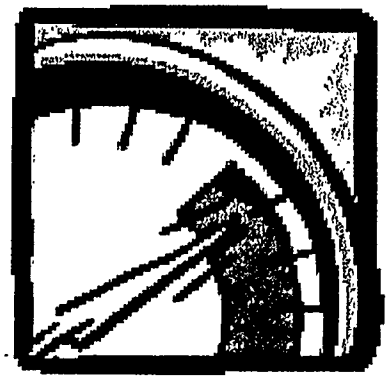




Conclusion

- Introduction
  - Metrics Basics
- Product Metrics
  - Product Quality Attributes
  - Risks to Product Quality
- Object Oriented Metrics
  - OO Quality Attributes
  - SATC's OO Metrics
  - Establishing Baselines
- Metrics & Reliability
  - Developing Standards
- *Conclusion*



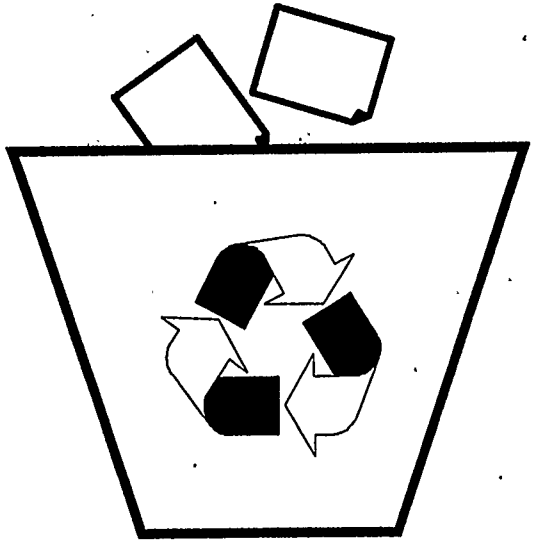


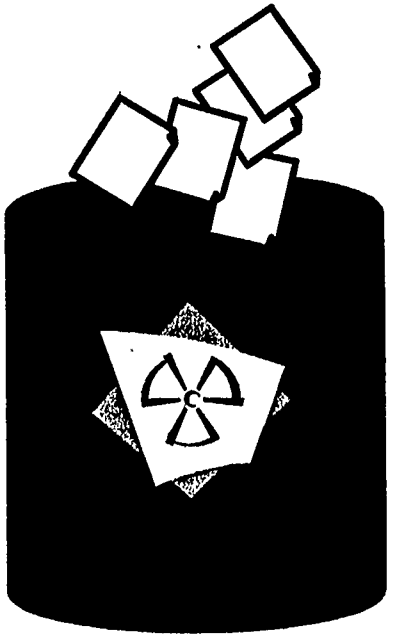
- **Object Oriented Design is no longer new.**
- **Investment in OO should pay off in reusable classes and interchangeable software parts.**
- **OO Metrics are well known, tools to collect them are available.**
- ***Using* OO metrics is currently more art than science.**
- **OO metrics should be part of a continuous improvement effort.**

NASA SATC
Software Assurance Technology Center

Object Oriented Metrics Support Reliability

Collecting metrics now will allow us to identify better OO metrics and reliable, reusable OO classes in the future.





References

Basili V.R., "Software Modeling & Measurement: The Goal Question Metric Paradigm," *Technical Report CS-TR-2956*, U. MD, 1992.

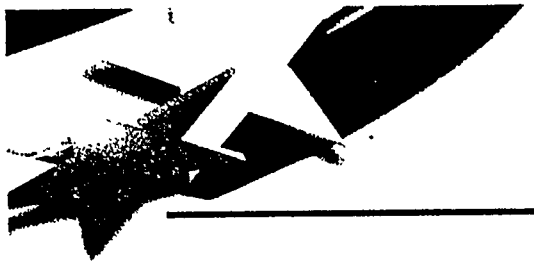
Chidamber S.R. & Kemerer C.F., "A Metrics Suite for Object Oriented Design," *IEEE Trans. Software Eng.*, V20, #6, June, pp. 476-493, 1994

Deitel & Deitel, *C++: How to Program*, Prentice Hall, 1998

Grady, *Practical Software Metrics for Project Management and Process Improvement*, Prentice Hall, 1992

Lorenz & Kidd, *Object-Oriented Software Metrics*, Prentice-Hall, 1994

McCabe, T J, L A, Dreyer et al. , "Testing an object-oriented application", *Journal of the Quality Assurance Institute*, (October), pp21-27, 1994



IDART
Information Design
Assurance Red Team

The Adversaries' View

Software Quality Forum 2000

Julie Bouchard

jfbouch@sandia.gov

(505) 845-9724

Information Design Assurance Red Team

http://www.sandia.gov/idart

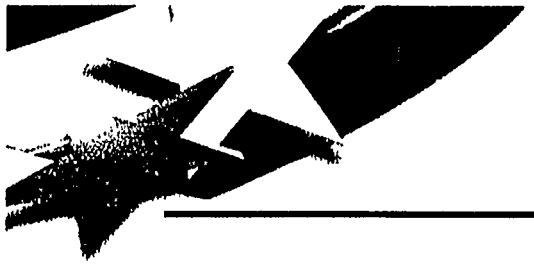
Sandia National Laboratories

Albuquerque, NM 87185-0449

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

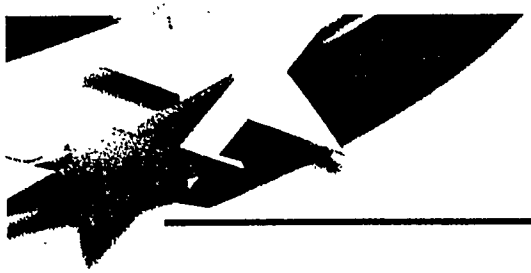


Sandia
National
Laboratories



Outline

- What is a Red Team?
- Threats and Adversaries
- Assessment and Reporting
- Experiences



What is a Red Team?

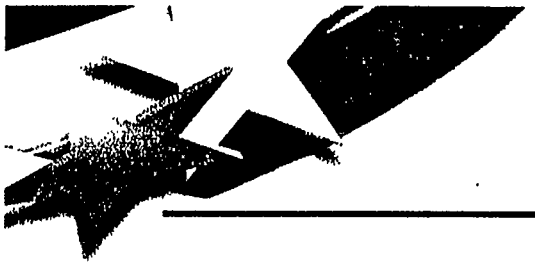
- The Red Team is a model adversary

- Differs from a real adversary:

  Cooperation:

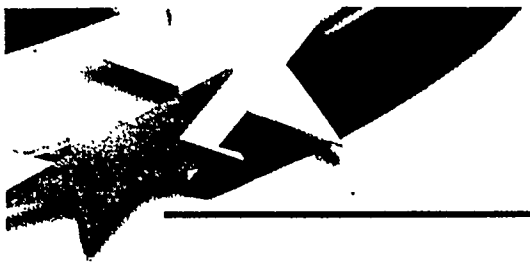
  – Disclose all tools, techniques, and methods
  – Don't broadcast findings
  – Don't want to damage your system
  – Team member with team goals for the assessment or experiment

Sandia National Laboratories



Why Red Team?

- You don't want to be the next VICTIM of a large or damaging or highly publicized attack
  - mission
  - publicity
- Red Team shows you the opposition without exposing you to the dangers
- Red Team plays the opposition but is on your team
- If you know about a vulnerability, you can do something to address it



Threats

- **Hackers- any target is equally good**
  - **US Embassy in China- also Atl. Braves, Sheraton (CNN)**
- **Win over the government- any government system is vulnerable**
  - **Library of Congress "US Congress Web Site- defeated!" (AP)**
- **Effect mission- after you to deny or impact your mission capability**
  - **Belgrade flood of NATO site- during bombing (IDG, PC World)**
- **Economic gain- after you because you have information that can be sold or because you handle payroll, credit cards, etc.**
  - **CD Universe hacker/blackmailer- credit card #s (AP, CNN)**



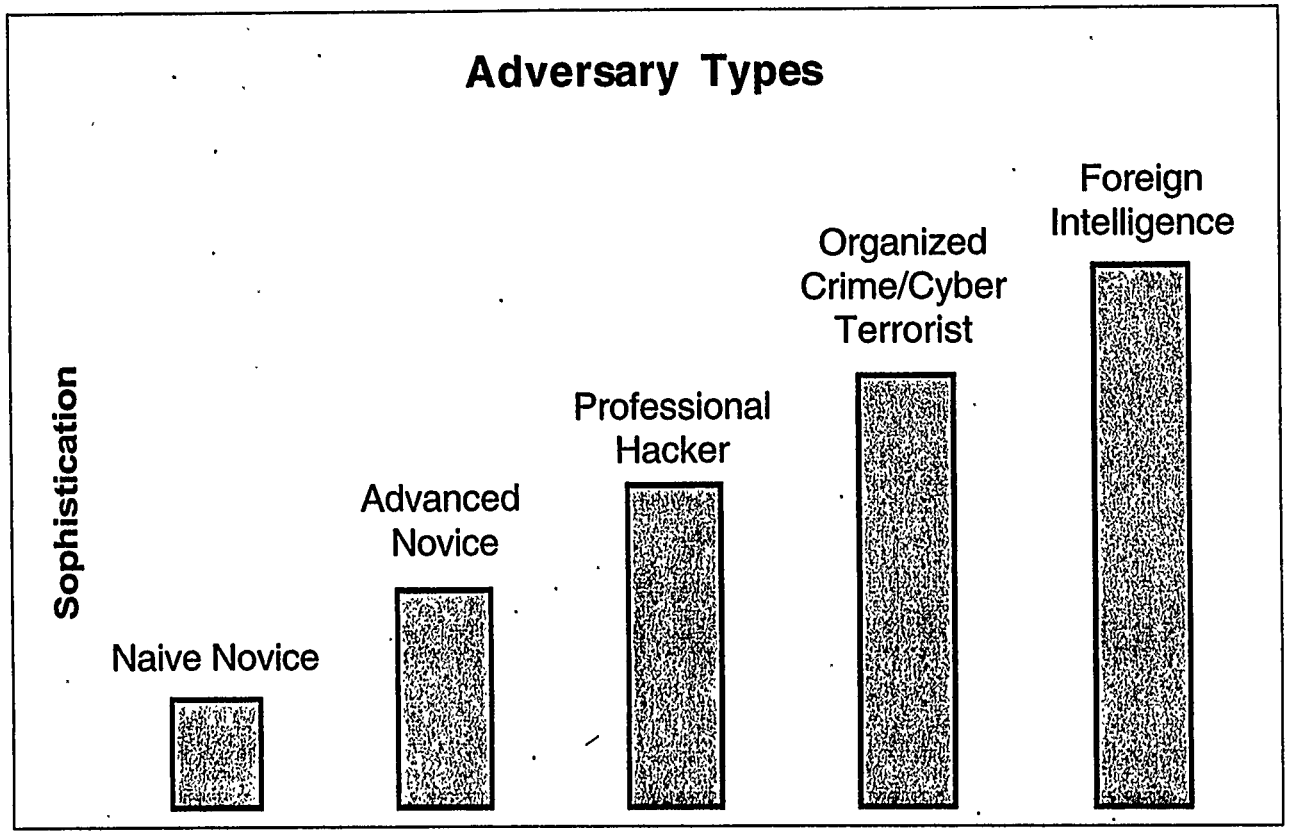
How We Model Adversaries?

Variables in our models include:

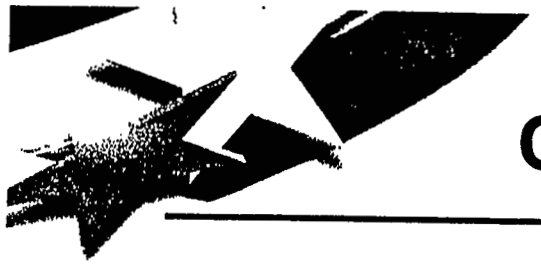
- **Level of sophistication**
  - *Hacker, Terrorist, Nation State, Foreign Intelligence… ?*
- **Goals and Objectives**
  - *What is the adversary's overall goal?*
- **Funding**
  - *Money and "magical powers" that can be purchased- including technology and inside information*
- **Risk Aversion**
  - *How hard does the adversary avoid detection?*
- **Other Behavioral Expectations**
  - *Creativity, determination, ...*

Sandia National Laboratories

Adversary Model

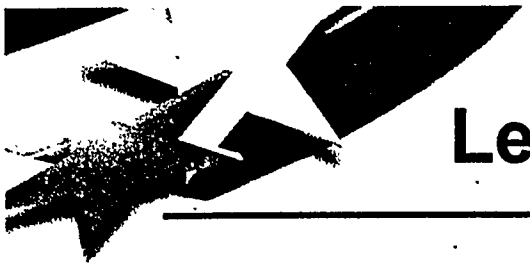


Sandia National Laboratories



Our Favorite Adversary

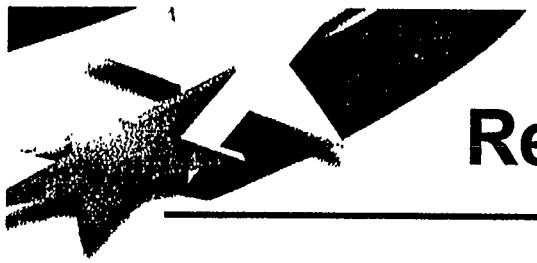
- **Cyber-terrorist**
  - *A terrorist organization or small nation state*
  - *Based on theoretical models & some new data*
- **Has specific goals & objectives**
  - *Usually to limit effectiveness of a critical info system.*
- **Well funded**
  - ***Can acquire anything money can buy:*** *consultants, recent university graduates, commercial technology, open data, inside information, inside assistance & some custom attacks*
  - *Can learn all design information*
- **Risk averse- getting caught prevents achieving goal**
- **Creative & very clever…**



Less Interesting Adversary

- **Average Hacker**
  - *The motivated by thrills or challenge hacker*
  - *Based on reports of actual attacks*
- **Goals & Objectives**
  - *Thrills or challenge more than target or objective*
  - *Visibility- often claim credit for hack under pseudoname*
- **Very Limited Funding- seemingly endless time**
  - *Has all resources of internet available*
  - *Has funding and time for some research/development of new/modified attacks*
- **Risk averse- don't want to do jail time**
- **Creative & very clever…**

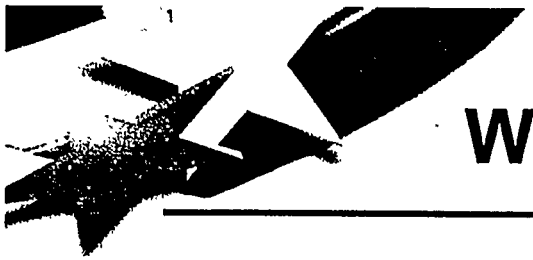
Sandia National Laboratories



Red Team Generalizations

- **Goal-oriented**
  - *Specific mission that may not be known to the defenders.*
  - *Focuses on capturing flags (as defined in the exercise).*
- **Well informed**
  - *They've seen all the docs.*
- **Lazy**
  - *Uses no more resources than necessary to capture the flag*
- **Shy**
  - *Seeks to avoid detection*
- **Creative & sneaky**
  - *Seeks unconventional attacks where you least expect it.*
  - *They are often accused of cheating.*
  - *Endowed with "magical mystery powers."*

Sandia National Laboratories

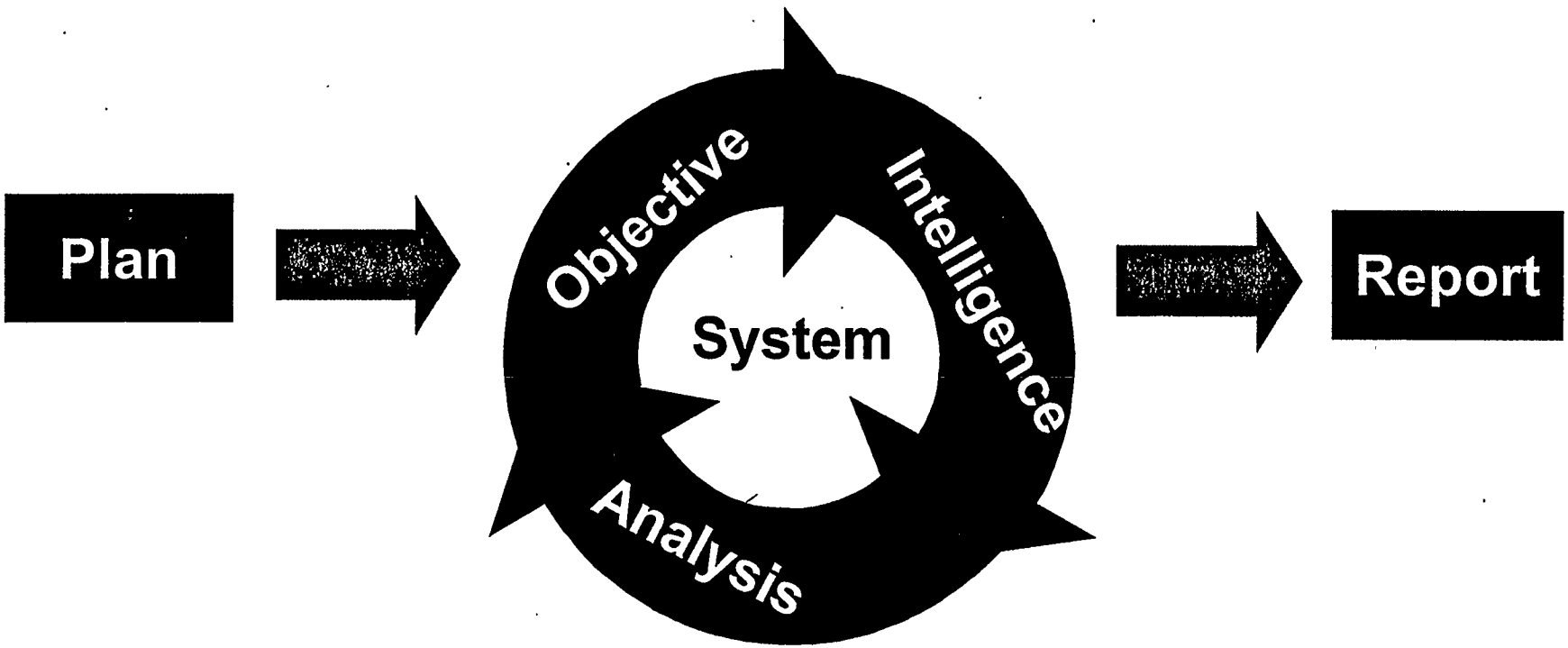


When to use a Red Team?

- **Evaluating developmental systems**
- **Studying a particular adversary**
  - *How often do you get to pick a bad guy's brain?*
- **Studying adaptations & interactions**
  - *both human or cyber*
- **Planning experiments**
- **Scripting or planning attacks**
  - *even when you are NOT interested in adaptation.*
- **Whiteboarding**
  - *War gaming on a white board*

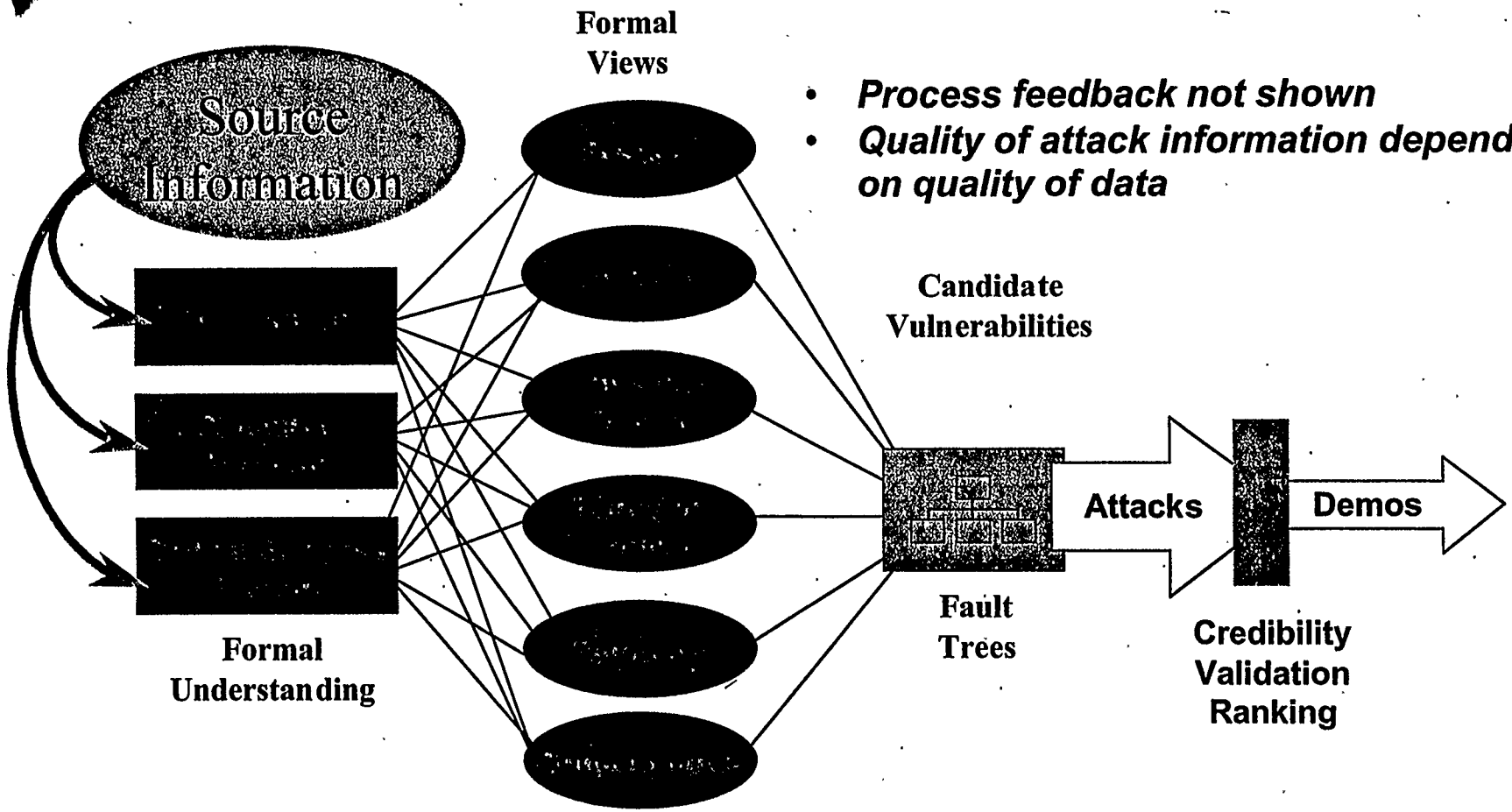
Sandia National Laboratories

Red Team Process Model



Sandia National Laboratories

Assessment Views



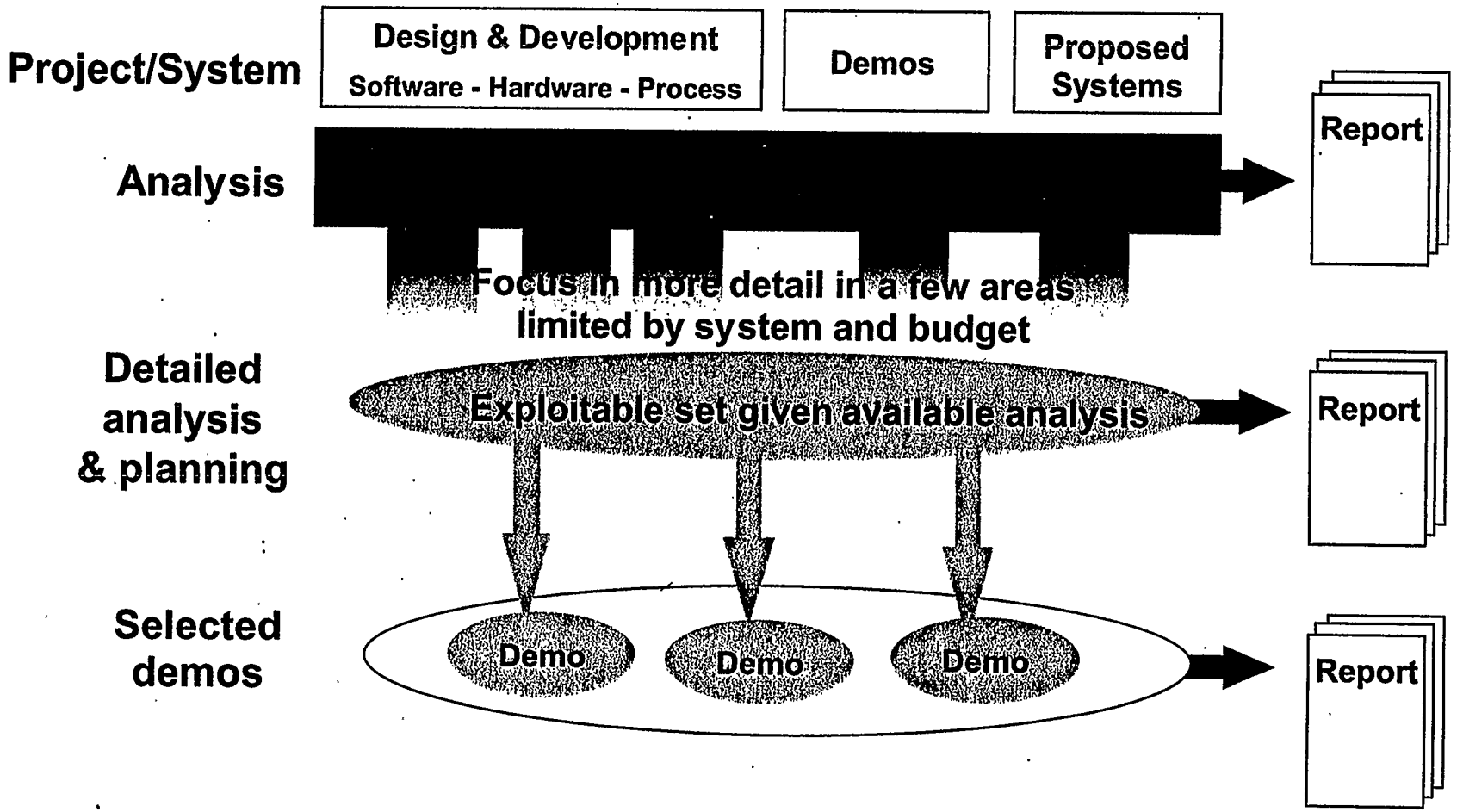
- *Process feedback not shown*
- *Quality of attack information depends on quality of data*

Domain Expertise

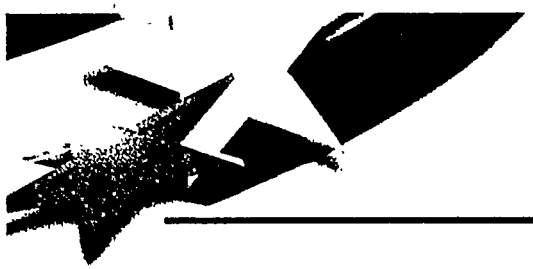
Sandia National Laboratories

Reporting

Scope of analysis depends on system and budget.



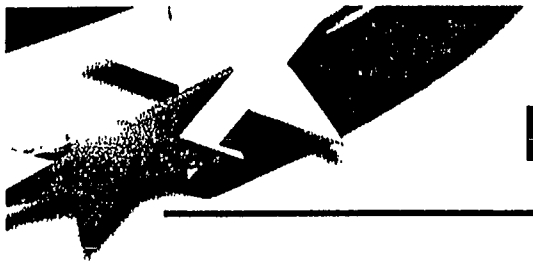
Sandia National Laboratories



Vs. Penetration Test

Attribute	Penetration Test	Red Team
Configuration Assessment	Snapshot	Processes
Disclosure	Limited	Full
Reporting	Test Results	Analysis
Interaction Analysis	Interfaces	System
Adversary	Hacker / Outsider / Door Rattling	Sophisticated / Insider / Targets
Ideal Targets	Production	Developmental
Overall System View	Certification	Understanding

Sandia National Laboratories



Red Team Experiences

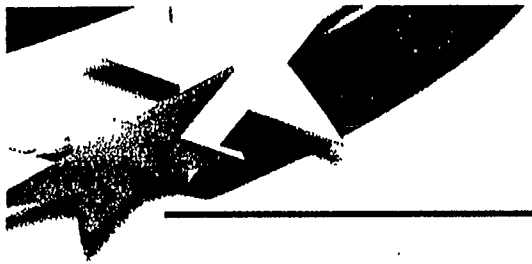
- *Dept of Defense*
  - Joint Command & Control Warfare Center
  - Joint Program Office for Biological Defense
  - DARPA
- *Other US Government*
  - US Dept. of Treasury
  - US Dept of Energy
- *Private Sector*
  - Gas & oil
  - Banking & finance
  - Electric power generation & distribution

Sandia National Laboratories



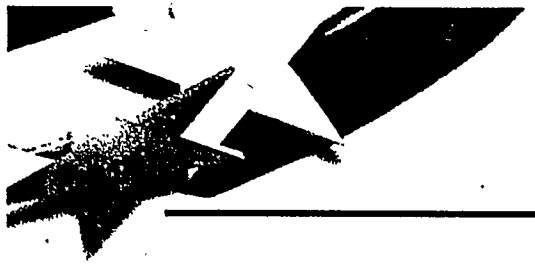
Experiences

- **Identifying vulnerabilities during design**
  - *real quick turn-around (time-to-market issue)*
  - *they propose design, we theoretically attack*
  - *they propose revised design, we theoretically attack*
  - *they build a system with less vulnerabilites*
- **Identifying vulnerabilities in existing systems**
  - *critical infrastructure*
  - *know they are being hacked*
  - *want to know vulnerabilites so they can effectively use their limited budget to address most critical problems*



Experiences

- **Research into better security systems**
  - *Experiments with Intrusion Detection systems, layered defense, dynamic defense, protected applications*
    - **hypothesis: dynamic defense is more effective at preventing adversary attack than is static defense**
    - **red team: attack several configurations (some configuations are static and others are dynamic)**
    - **metric: adversary effectiveness (achieve flag) and adversary work factor (time, cost, ...)**
  - *Whiteboards on dynamic defense, denial of service, ...*
  - *Attempting to build better adversary understanding among those who design, implement, and maintain security systems*



In Summary…

- You don't want to be the next VICTIM of a large or damaging or highly publicized attack
- Multiple threats exist
- There is more than one adversary to consider
- The Red Team is a MODEL adversary
- Red Teams can be used during development, with existing systems, or as part of a research team.

IDART
Information Design Assurance Red Team

Sandia National Laboratories



Useable Metrics for Software Process Improvement within the CMM

"There are thousands of ways to fail when building software applications, and only a very few ways to succeed." Capers Jones

Joe Schofield, Sandia National Labs

Sandia is a multi-program laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000



Major Topics

Mapping Metrics to CMM KPAs - as promised!

Why this work is/was needed

OO Measurements - adding value for the client?

Lines of code and Functions Points - what's the dif?

Measuring your metrics - what good are they anyhow?

A recommended framework for metrics usage within the CMM

Getting value from your metrics efforts - a few illustrations

How to kill a defect!

Next steps - yours! *What might you do differently as you leave?*



Metrics Map - Level 2

	Product (for SPE)			Process (for SQA)		
KPA	***Size**	**Value**	**Quality**	**Productivity**	**Value**	**Quality**
RM	Use Cases	Use Cases/$	Defect/Use Case	Use Cases/PM		% changed or added
SPP	FPs			Expected FP/PPM, costs, defects, cycle (time)		
SPC				Actual FP/PPM, costs, defects, cycle (time)		
SM	FPs	FP/$	Defect/FP	FP/PPM	FP/$ Defect/FP	FP churn & creep % of FPs SMed
CM	# of artifacts		artifact churn	When CMed # of artifacts CMed / artifacts	L/C$/FP CSRs/FP	defects/L/C defects/FP
SQA	FP	$/FP	Defects/FP	# of projects satisfy SQA / # of projects		

*As with Function Points, consider Size a function of complexity

Joe Schofield:2/9/00

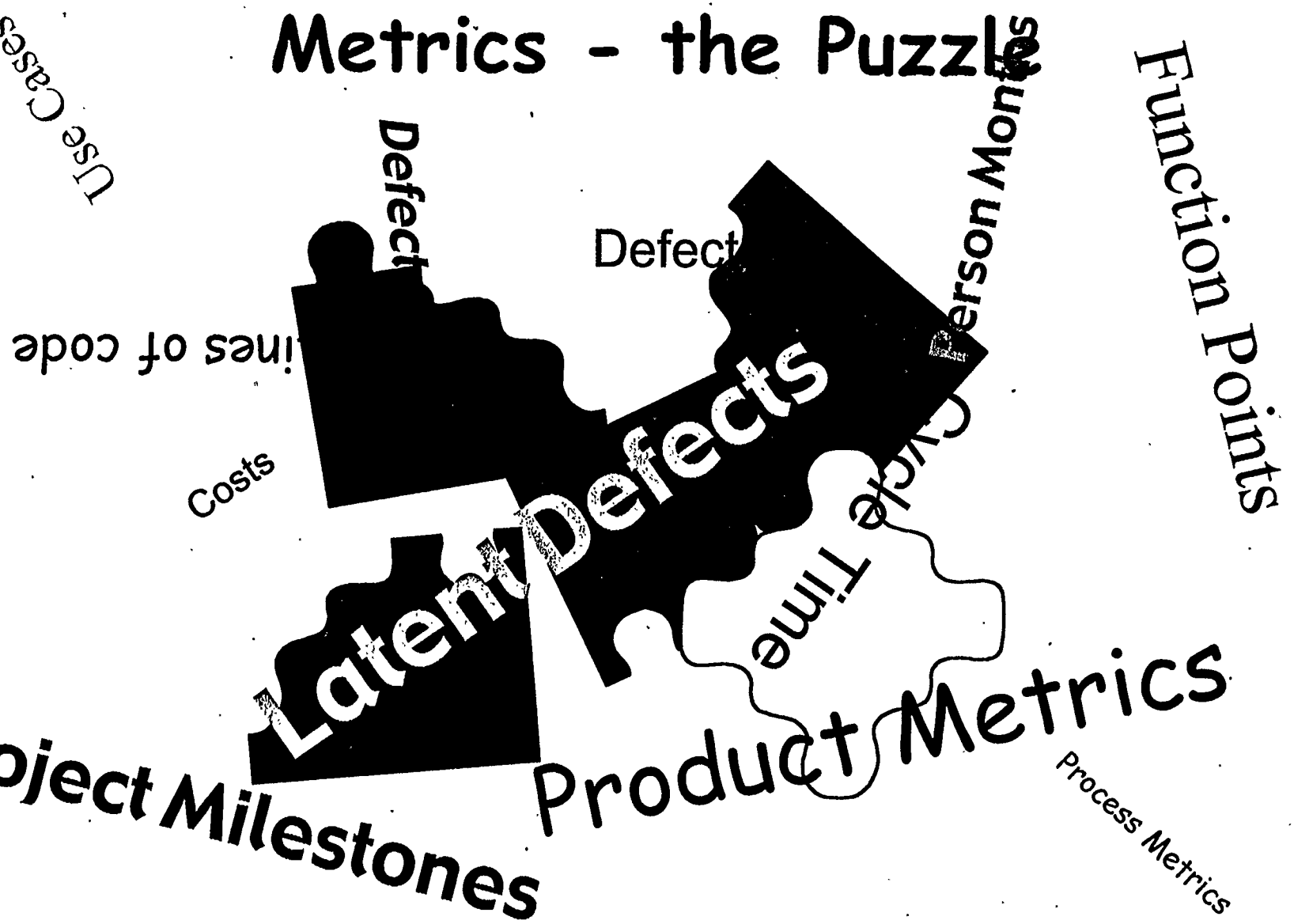
Sandia National Laboratories

Metrics Map - Level 3 Sampling

	Product (for SPE)			Process (for SQA)		
KPA	**Size**	**Value**	**Quality**	**Productivity**	**Value**	**Quality**
OPF	max FPs	max $	min defect	Org. FPPPM	Org. $/FP	Org. defect rates
OPD				MTTF	$/Defect	
ISM				tailoring results, $, and defect rates		
SPE	FPs, Use Cases	FP/$	defect/FP	Test coverage efficiency cost/test case Regression testing efficiency, defect by origin		
IC						
TR	tailored hours	time $avings	defect reduction	hours PP/ FPPPM	$/FPPPM	$/defect reduction
PR		$avings from uncovered defects	# of defects	hours of rework not performed		defect origin



Metrics - the Puzzle





Joe Schofield:2/9/00

Sandia National Laboratories

The Motive

Observation: A limited number of process metrics are available by KPA for CMM users. Most deal with project management aspects of software engineering. For instance, the SPC does not identify *metrics* on its home page, and the SEI includes neither *metrics* nor *quantitative management* on its short cut topics list.

Many skeptics of CMM have no quantified measures on which to demonstrate their own software engineering capability.

The emergence of OO techniques is not adding clarity to metrics and their value.

Intervention: Propose a set of metrics to support an organization that aspires towards CMM level 2 (and eventually level 4 - Quantitative Management). Metrics would demonstrate that activities being performed satisfy the intent of KPA goals and further, quantify improvement in the KPAs.

Audience: Those interested in software process improvement, CMM, metrics, and software value.



A Review of Recent OO Metric Presentations

- "Far too much emphasis on defining new metrics . . . Needed but research is unbalanced." "Early size information (attributes, methods) explains most of the effort variance!!" "Even imperfect measurement provides you with insight into your system quality . . ." - Lionel Briand, *Measurement and Quality Modeling of OO Systems.*

- Number of statement per hour (a.k.a. - lines of code) for C++ efforts range from 3 to 92, and hours per class development range by a factor of 3 from 16 to 48. SPC, *Measurement for Object-Oriented Software Projects*

- "Complexity not simple to identify or measure. Established rules of thumb no longer valid." Suggested architectural metrics include:
  - *weighted methods per class* - a count of methods implemented in a class
  - *response for a class* - cardinality of the set of all methods that can be invoked in response to an object . . .
  - *Coupling between object classes* - the number of other classes to which a class is coupled

Applying Object Oriented Metrics, Software Assurance Technology Center, NASA, Dr. Linda Rosenberg



Measuring Objects - Help Needed

OO metrics

- are psychologically attractive within the OO community
- do not deal with full life-cycle issues
- have not been applied to testing
- have on conversion rules to LOCs
- have no conversion rules to Function Points
- lack automation
- are difficult to enumerate
- are not supported by software-estimating tool
- indicate about 2.5 times as many defects per Function Point with OO techniques
- notable gaps in reuse data
- indicate a 50% abandonment while applying OOA and OOD approaches

Interpreted from Jones, Software Quality - Analysis and Guidelines for Success



Do you really want to count KLOCs?

"I regard the usage of LOC for cross-language studies to be an example of professional malpractice . . . Using physical LOC for measurement purposes is fundamentally harmful."

"The widespread military usage of the obsolete LOC metric is probably why SEI made such a serious mistake."

1/3 of LOC studies evaluated physical LOCs, 1/3 logical LOCs, and 1/3 didn't define which types of LOCs they used!

KLOC is a very unreliable using either physical or logical lines or logical statement forms

In one organization that was counting LOCs, four different counting methods with a 500% variation in counts were being used within 100 feet of each other

Function points are superior to LOCs

Interpreted from Capers Jones, Software Quality.- Analysis and Guidelines for Success



Arguments favoring Function Points

Hundreds of industry and government "subscribers" (now includes SEI)

Most widely used software measurement value

Certification for counters, benchmarking across industry

Sample Function Point uses:

- 4500 FPs per manager domain
- 20 FPs developed per staff month with .75 bugs per FP
- 10 work hours per Function Point
- 386 FPs supported per staff member
- $407 to build a new FP; $58 per year to maintain



Measuring your metrics

Rubin's had three:

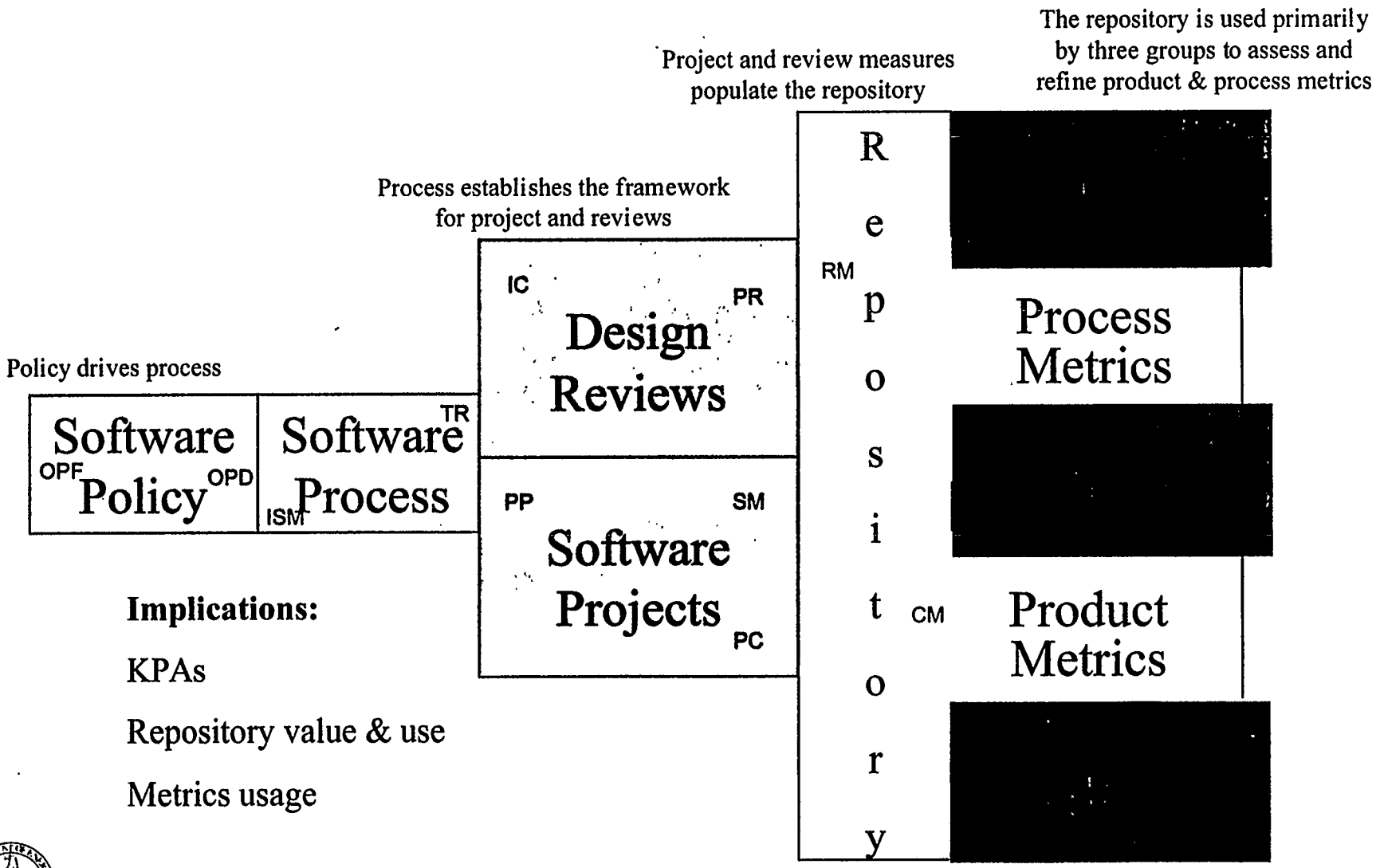
- meaningful outside of IS
- in place for two years or more
- used for management decision-making (SEPG or MSC)

You may want to ask yourself these questions:

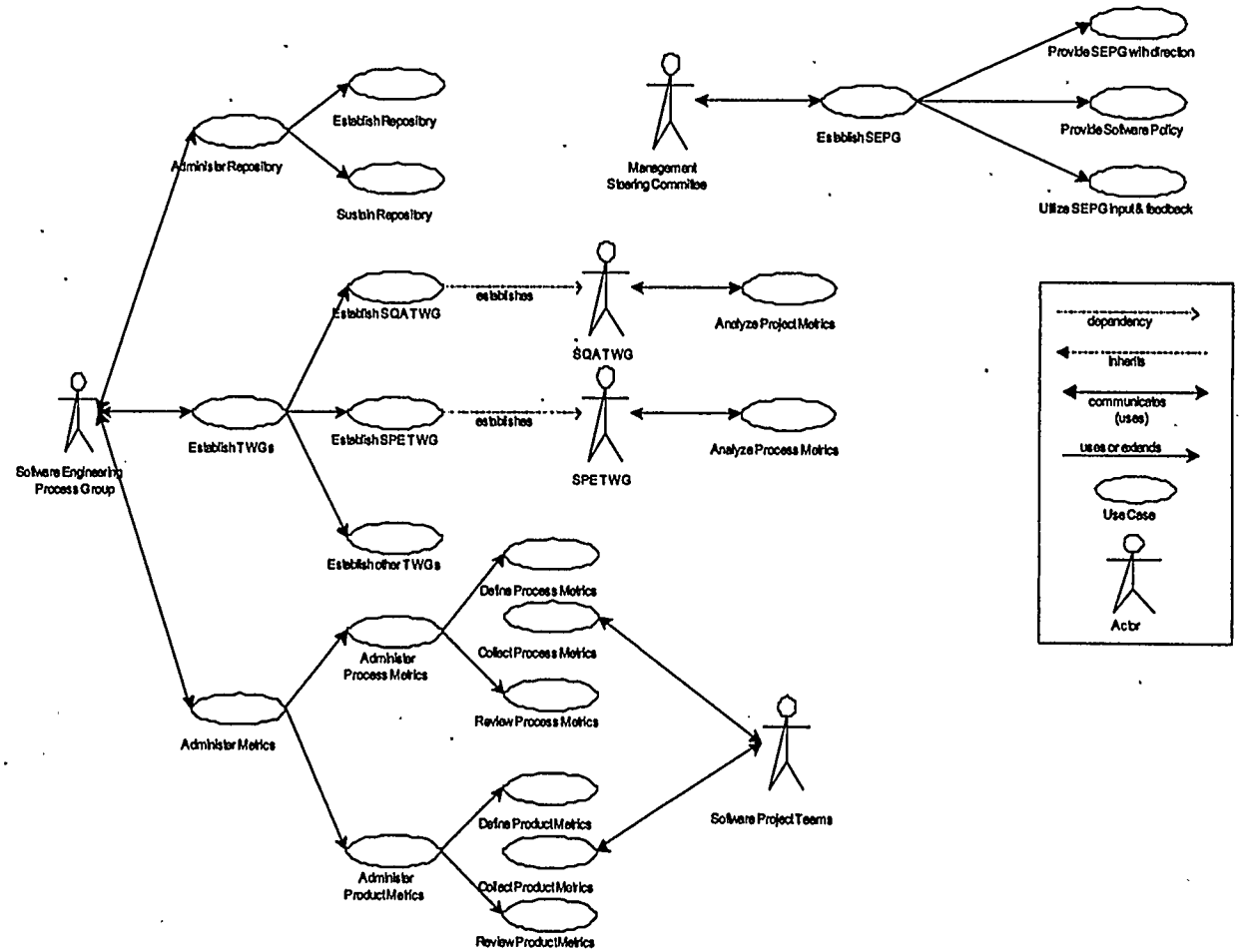
- What was the cost of the defects we delivered last year, the year before?
- Where in the development of our software did these defects originate?
- What was the difference in size and complexity of projects that were delivered early, on time, and late?
- Which tools and methods have been the most effective in increasing productivity and reducing costs over the past three years?
- What was our return on investment after introducing OO techniques two years ago?



A Proposed Metric-Flow Mapping



CMM-based Metrics Usage High-level Use Case Diagram



Implications for:

MSC

SEPG

SQA

SPE

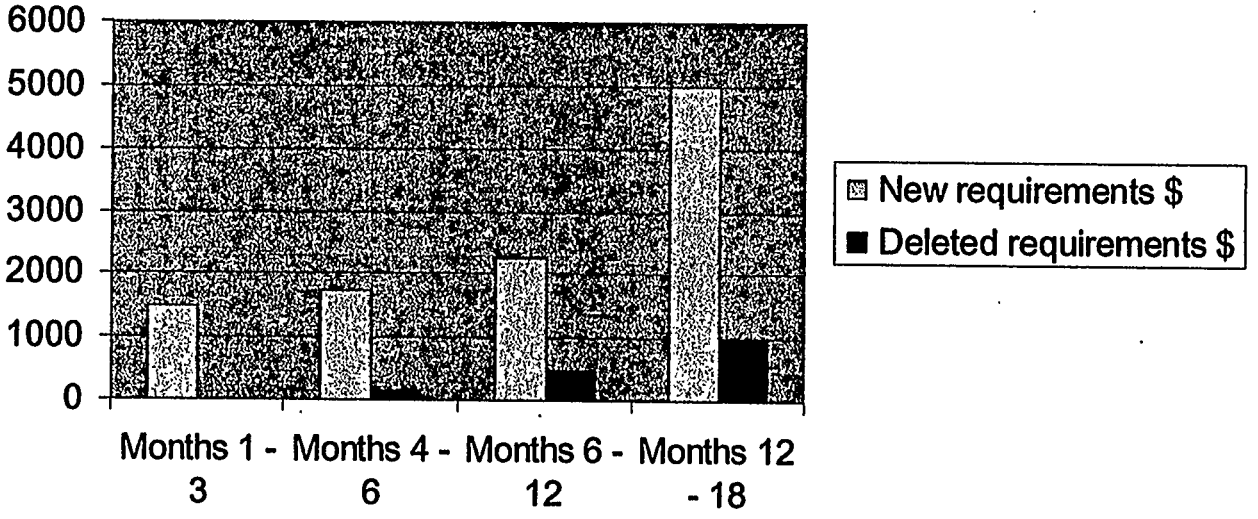


Joe Schofield:2/9/00

Sandia National Laboratories

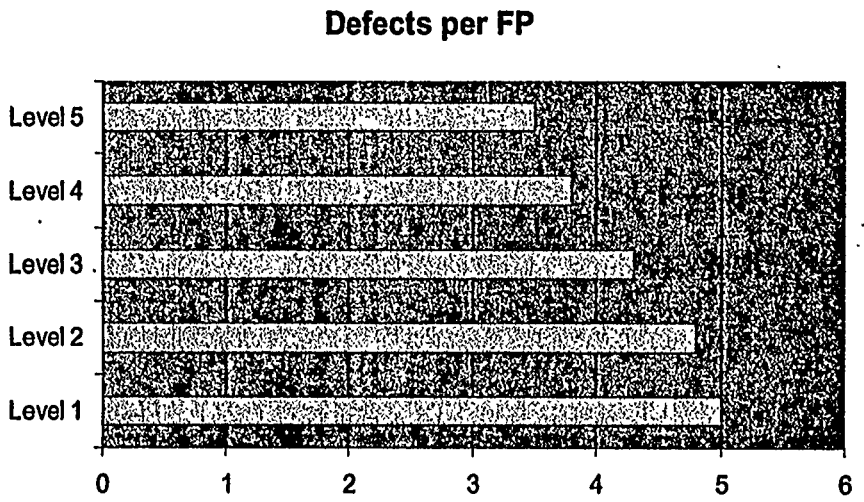
The Cost of Requirements Creep & Churn

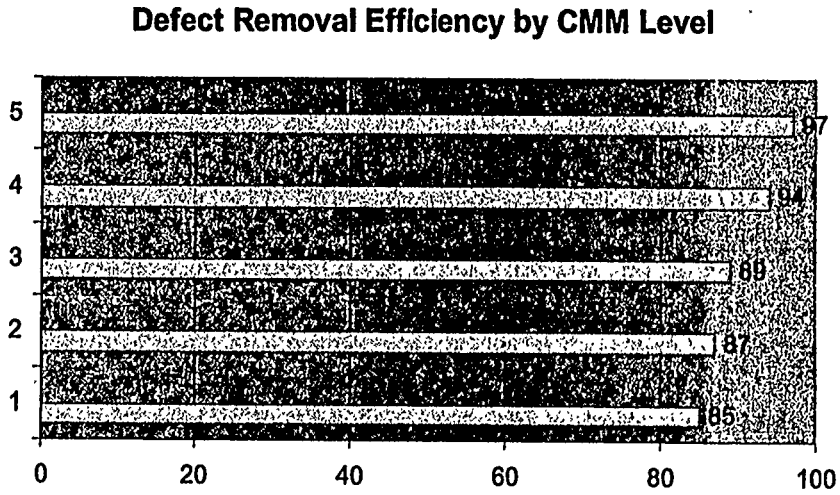
Development Costs for New Requirements by Month Per Function Point



Interpreted from Jones, Estimating Software Costs

Defect Per FP and Defect Removal Efficiencies for CMM Levels



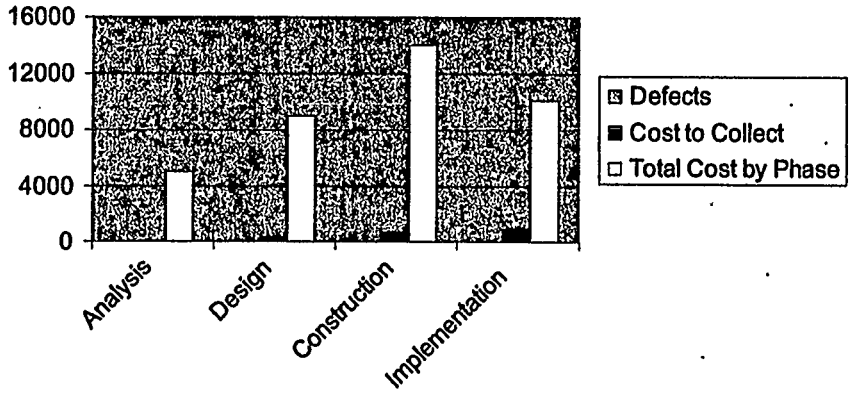


Interpreted from Jones, Estimating Software Costs

Defect Origin and Removal Cost-based Analysis

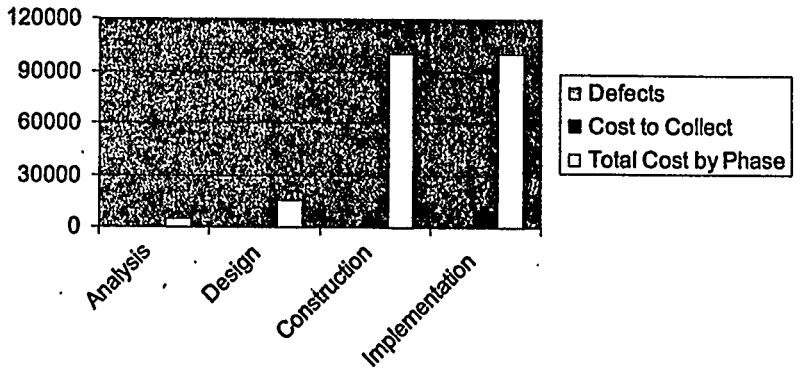
Phased Introduced	Defects	Cost to Correct	Total Cost by Phase
Analysis	50	100	5000
Design	30	300	9000
Construction	20	700	14000
Implementation	10	1000	10000

Defect Cost Yields a Different Priority



Phased Introduced	Defects	Cost to Correct	Total Cost by Phase
Analysis	50	100	5000
Design	30	500	15000
Construction	20	5000	100000
Implementation	10	10000	100000

Another Possible Cost Comparison



"Defecticide"

Count 'em by:

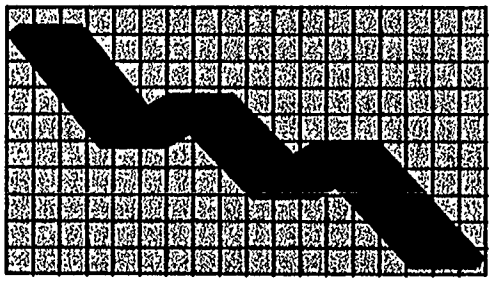
- number in an application
- number per Function Point for a given process / technology
- defect removal technique
- defect introduction
- latent defects, percent user discovered
- severity of defects
- cost per defect



In closing, a story

Opportunity or challenge?





Or do you know?

This report has been reproduced directly from the best available copy. It is available electronically on the Web (http://www.doe.gov/bridge).

Copies are available for sale to U.S. Department of Energy employees and contractors from—

Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831
(423) 576-8401

Copies are available for sale to the public from—

National Technical Information Service
US Department of Commerce
5285 Port Royal Road
Springfield, VA 22616
(800) 553-6847