IDA DOCUMENT D-1416

(2)

A SURVEY OF BALLISTIC MISSILE DEFENSE (BMD) CONTRACTOR AND COMMERCIAL OFF-THE-SHELF (COTS) SOFTWARE ENGINEERING ENVIRONMENTS (SEEs)

David A. Wheeler
Michael Frame
Judy Popelas

Dennis W. Fife, *Task Leader*

August 1993

Prepared for
Ballistic Missile Defense Organization

INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311-1772

DEFINITIONS

IDA publishes the following documents to report the results of its work.

Reports

Reports are the most authoritative and most carefully considered products IDA publishes. They normally embody results of major projects which (a) have a direct bearing on decisions affecting major programs, (b) address issues of significant concern to the Executive Branch, the Congress and/or the public, or (c) address issues that have significant economic implications. IDA Reports are reviewed by outside panels of experts to ensure their high quality and relevance to the problems studied, and they are released by the President of IDA.

Group Reports

Group Reports record the findings and results of IDA established working groups and panels composed of senior individuals addressing major issues which otherwise would be the subject of an IDA Report. IDA Group Reports are reviewed by the senior individuals responsible for the project and others as selected by IDA to ensure their high quality and relevance to the problems studied, and are released by the President of IDA.

Papers

Papers, also authoritative and carefully considered products of IDA, address studies that are narrower in scope than those covered in Reports. IDA Papers are reviewed to ensure that they meet the high standards expected of refereed papers in professional journals or formal Agency reports.

Documents

IDA Documents are used for the convenience of the sponsors or the analysts (a) to record substantive work done in quick reaction studies, (b) to record the proceedings of conferences and meetings, (c) to make available preliminary and tentative results of analyses, (d) to record data developed in the course of an investigation, or (e) to forward information that is essentially unanalyzed and unevaluated. The review of IDA Documents is suited to their content and intended use.

The work reported in this document was conducted under contract MDA 903 89 C 0003 for the Department of Defense. The publication of this IDA document does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that Agenc

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1993	Final

4. TITLE AND SUBTITLE
A Survey of Ballistic Missile Defense (BMD) Contractor and Commercial Off-the-Shelf (COTS) Software Engineering Environments (SEEs)

5. FUNDING NUMBERS
MDA 903 89 C 0003

Task T-R2-597.2

6. AUTHOR(S)
Dennis W. Fife, David A. Wheeler, Michael Frame, Judy Popelas

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Institute for Defense Analyses (IDA)
1801 N. Beauregard St.
Alexandria, VA 22311-1772

8. PERFORMING ORGANIZATION REPORT NUMBER
IDA Document D-1416

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
BMDO/GSI
The Pentagon, Room 1E149
Washington, DC 20301-7100

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; unlimited distribution: 21 January 1994.

12b. DISTRIBUTION CODE
2A

13. ABSTRACT (Maximum 200 words)

This annotated briefing describes the results of a survey of Ballistic Missile Defense (BMD) contractor software engineering environments (SEEs). It also reports the current status of selected commercial off-the-shelf (COTS) SEEs that employ a general interoperability component such as a data repository or an intercommunication tool. The purpose of this briefing is to aid the BMD Organization (BMDO) in evaluating SEE-related risk mitigation. Contractor information was derived from one-on-one interviews with program offices or contractors for BMD system elements. COTS SEE information was derived from hands-on demonstrations of the products. The conclusions of this briefing are that more then 230 different products are now being used for BMD software development. This large number of different products means that a huge investment will be necessary for the BMD SEE program to come close to matching BMD element development environments. Also, COTS SEEs that depend heavily on interoperability or intercommunication tools are not sufficiently mature to be recommended without significant reservation. In particular, adequate performance and user acceptance have not been demonstrated for large-scale projects comparable with BMD system development.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Software Engineering Environment (SEE); Software Development Environment (SDE); Integrated Program Support Environment (IPSE); Software Tools	106
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

IDA DOCUMENT D-1416

A SURVEY OF BALLISTIC MISSILE DEFENSE (BMD) CONTRACTOR AND COMMERCIAL OFF-THE-SHELF (COTS) SOFTWARE ENGINEERING ENVIRONMENTS (SEEs)

David A. Wheeler
Michael Frame
Judy Popelas

Dennis W. Fife, *Task Leader*

August 1993

Approved for public release; distribution unlimited.

INSTITUTE FOR DEFENSE ANALYSES
Contract MDA 903 89 C 0003
Task T-R2-597.2

Preface

This document was prepared by the Institute for Defense Analyses (IDA) under the task order, SDIO Software Technology Plan, and contributes to an objective of the task, to provide "an annotated briefing providing risk mitigation alternatives pertinent to GPALS software engineering environments (SEEs)." The work was sponsored by the Ballistic Missile Defense Organization (BMDO).

The following IDA research staff members were reviewers of this document: Dr. Brian Cohen, Ms. Audrey Hook, Dr. Reginald Meeson, Mr. Clyde Roby, and Ms. Beth Springsteen.

A Survey of Ballistic Missile Defense (BMD) Contractor and Commercial Off-the-Shelf (COTS) Software Engineering Environments (SEEs)

David A. Wheeler
Michael Frame
Judy Popelas
Dennis W. Fife, Task Leader

Briefing Outline

This is an annotated briefing describing the results of a survey of Ballistic Missile Defense (BMD) contractor software engineering environments (SEEs) and commercial off-the-shelf (COTS) SEEs. This work was performed by the Institute for Defense Analyses (IDA) for the Ballistic Missile Defense Organization (BMDO). The intended audience is BMDO personnel responsible for software development or for monitoring development of the BMD SEE. The annotations copy the slides' main bullets so readers can quickly locate specific details.

This briefing has four major sections:

- **Introduction**

The introduction gives background information and a description of the purpose and scope of this briefing.

- **Survey of BMDO Element SEEs**

This section presents a summary of results from IDA's survey of the BMDO elements' current SEEs. IDA has acquired much more detailed information which is available separately, but some of those details include competition-sensitive or proprietary information, and therefore are not included in this briefing.

- **Advanced COTS SEEs**

This section presents information about advanced COTS SEEs. These are categorized by IDA into three basic types: Portable Common Tool Environment (PCTE)-based SEEs, A Tool Interface Standard (ATIS)-based SEEs, and messaging-based SEEs. Examples of PCTE-based COTS SEEs are EAST and Enterprise II. Examples of ATIS-based SEEs are Atherton's Software BackPlane, Verdix' VADS APSE, and CRI's Life*Cycle. Examples of messaging-based SEEs are Hewlett-Packard's SoftBench and Sun's ToolTalk.

- **Conclusions**

This briefing closes with conclusions. An appendix is included that provides more information on specific COTS SEEs. This appendix is not considered part of the main briefing but is instead considered background material. References are also included.

Briefing Outline

- Introduction
  - Background
  - Briefing Purpose and Scope
- Survey of BMDO Element SEEs
- Advanced COTS SEEs
  - PCTE
  - ATIS
  - Messaging-based
- Conclusions

Background: What is a SEE?

- **A Software Engineering Environment (SEE) is the "set of automated tools, firmware devices, and hardware necessary to perform the software engineering effort" [DoD-STD-2167A]**

The set of automated tools in a SEE typically includes items such as compilers, text editors, requirements and design tools, and configuration management (CM) tools. Tools that support software engineering are called CASE (computer-aided software engineering) tools. Ideally, a SEE contains tools to support all software products under development or maintenance.

- **A SEE must be complete and flexible to develop and maintain a large software system.**

As systems get large, more tools are needed to efficiently coordinate people and control the product under development or maintenance.

Background: What is a SEE?

- A Software Engineering Environment (SEE) is the "set of automated tools, firmware devices, and hardware necessary to perform the software engineering effort" (DoD-STD-2167A)
- A SEE must be complete and flexible to efficiently develop and maintain a large software system

Background: BMDO History Relating to SEEs

The following is a brief outline of the history of BMDO's relation to SEEs:

- **BMDO is developing a large amount of software.**

Current estimates are at millions of lines of code.

- **BMDO views advanced SEEs as a way to improve productivity and supportability.**

- **The Army's Space and Strategic Defense Command (SSDC) has contracted with International Software Systems, Inc (ISSI) of Austin, Texas, to develop the BMD SEE.**
  - The requirements for the BMD SEE are advanced. They require a process-driven SEE, permitting users to define a software development and maintenance process and then automating (termed *enacting*) that process. The requirements also demand a repository-based SEE, integrating tool data through a central repository.
  - Potential users of this BMD SEE include the National Test Facility (NTF) integration and Post-Deployment Support System (PDSS) organizations.

- **SDIO (BMDO) policy 3405 requires delivery of software into the BMD SEE.**

- **BMDO views the SEE as high risk.**

In a number of meetings BMDO personnel have stated that the BMD SEE program includes significant risk. The next slide describes this risk.

Background: BMDO History Relating to SEEs

- BMDO is developing a large amount of software
- BMDO views advanced SEEs as a way to improve productivity and supportability
- Army SSDC contracted with ISSI to develop BMD SEE
  - Requirements are advanced: process driven & repository based
  - Potential users include NTF integration & PDSS
- SDIO (BMDO) policy 3405 requires delivery of software into the BMD SEE
- BMDO views the SEE as high risk

Background: BMD SEE Risks

The primary risk is that the BMD SEE may not be beneficial to BMDO because of one or more of a number of potential problems:

- The BMD SEE **may solve the wrong problem.**

If there is insufficient communication between the BMD SEE developer and the eventual users, the system will not meet the real needs of the users.

- The **costs associated with the SEE may exceed practical budgets (including adapters, training, and translation).**

In addition to the basic BMD SEE development costs, which are likely to be large, these costs include the costs of developing tool adapters to include existing tools in the BMD SEE, training for users, and probably the translation of data from existing tools into a form that BMD SEE tools can use.

- **Users** (i.e., software maintainers and possibly developers) **may not accept the required changes to their work.**

These changes include the additional (one-time) cost of **data translation** to a tool supported by the BMD SEE and **re-training** if the BMD SEE supports different tools than the ones the user already knows. More pervasive changes include the introduction of process enactment and the use of a data repository, which will likely require the user to adopt a **different work process.**

- The BMD SEE **may have inadequate performance or capabilities.**

Different projects are investigating advanced SEEs, such as STARS, I-CASE, and ProSLCSE, but currently most projects do not use advanced SEEs. Early work with repository-centered SEEs indicated that, at least for slower hardware, inadequate database performance can be a significant problem. The tools supported by the BMD SEE may be inadequate for the software development and maintenance tasks.

- The BMD SEE **may not be deployable early enough**.

Additional analysis of BMD SEE risks has been performed by IDA [Fife 1992].

Background: BMD SEE Risks

Primary Risk: BMD SEE may not be beneficial to BMDO.

- May solve wrong problem
- Costs associated with the SEE may exceed practical budgets (including adapters, training, and translation)
- Users may not accept the required changes to their work: data translation, re-training, different work process
- May have inadequate performance or capabilities
- May not be deployable early enough

Briefing Purpose and Scope

No explicit backup text for this slide

Briefing
Purpose and Scope

- **Briefing purpose: to provide sponsor with information on current SEEs to help manage risk**
- **Briefing scope:**
  - **Survey of existing (BMD contractor) SEEs**
  - **Evaluation of advanced COTS SEEs**
  - **Comparison of these to the planned BMD SEE**

Briefing Outline

This portion of the briefing presents a summary of IDA's survey of BMDO element SEEs. IDA has acquired much more detailed information, which is available separately, but some of those details include information considered competition-sensitive or proprietary. Therefore information will be presented without attributing specific SEE aspects with specific elements.

Briefing Outline

- Overview
  - Background
  - Briefing Purpose and Scope
- Survey of BMDO Element SEEs
- Advanced COTS SEEs
  - PCTE
  - ATIS
  - Messaging-based
- Conclusions

Outline for Survey of BMDO Element SEEs

No explicit backup text for this slide

Outline for Survey of BMDO Element SEEs

- Approach
- Software Development Product Classes and Categories
- Survey Results by Category
- Comparison of BMD SEE with BMDO Element SEEs
- BMD SEE Issues Raised by this Survey

Approach for Survey of BMDO Element SEEs

- **From interviews, questionnaires, and SDPs, gathered data on 6 projects**

Data for this survey was gathered primarily through personal interviews conducted by two IDA staff members. Interviews were structured around a product inventory questionnaire that dealt with 39 different types of tools, grouped into 8 categories. Additional questions, dealing with tool acquisition, in-house tool development and customization, tool use by software developers, and the software development process were also addressed. In one case, an interview could not be set up (ERINT) and only the written responses to the questionnaire were used. Data was also gathered from software development plans where available.

Data was collected from four program offices and two Brilliant Eyes (BE) contractors. These six entities are the six projects referred to throughout this briefing. The four program offices contacted were Ground-Based Radar (GBR), Patriot, Theater High-Altitude Defense (THAAD), and Extended Range Interceptor (ERINT). Both Corps Surface to Air Missile (Corps SAM) and Ground Based Interceptor (GBI) were not yet far enough along in their acquisition process to participate. The two BE contractors were TRW and Rockwell. TRW had one subcontractor, Hughes, while the Rockwell effort involved two divisions of Rockwell (SSD and ASSD) and three subcontractors: IBM/FSC, Harris, and Paramax.

- **Developed a set of tables summarizing data**

After collection, the data regarding tool use was organized into a set of tables. The tables identify, for each of the 39 types of tools, the particular tools in use at each of the 6 projects. Because this data is competition sensitive, it is not included as part of the briefing. However, non-attributive summaries of the tool usage data are included. Two things to note when looking at the summary data are first, some types of tools are not used by all six projects (in fact, a few types of tools are not in use by any of the six projects), and second, several tools of a given type are often in use within one project.

Approach for Survey of BMDO Element SEEs

- From interviews, questionnaires, and SDPs, gathered data on 6 projects
  - 4 program offices
    - GBR, Patriot, THAAD, ERINT
    - Corps SAM & GBI not yet far enough along to participate
  - 2 BE contractors
    - TRW (with subcontractor Hughes)
    - Rockwell (with subcontractors IBM/FSC, Harris, and Paramax)
- Developed a set of tables summarizing data
  - Some types of tools not in use at all participating projects
  - For many types of tools, more than one tool of the same type will be in use at a project

Software Development Product Classes and Categories

The software development tools discussed in this survey have been organized into four classes. Each class has been divided into two categories, for a total of eight categories. Within each category, several distinct product types have been identified.

- The first class, platforms, contains the two categories of hardware development platforms and software development platforms. An example of a hardware development platform product type is "machine type". An example of a software development platform product type is "operating system".

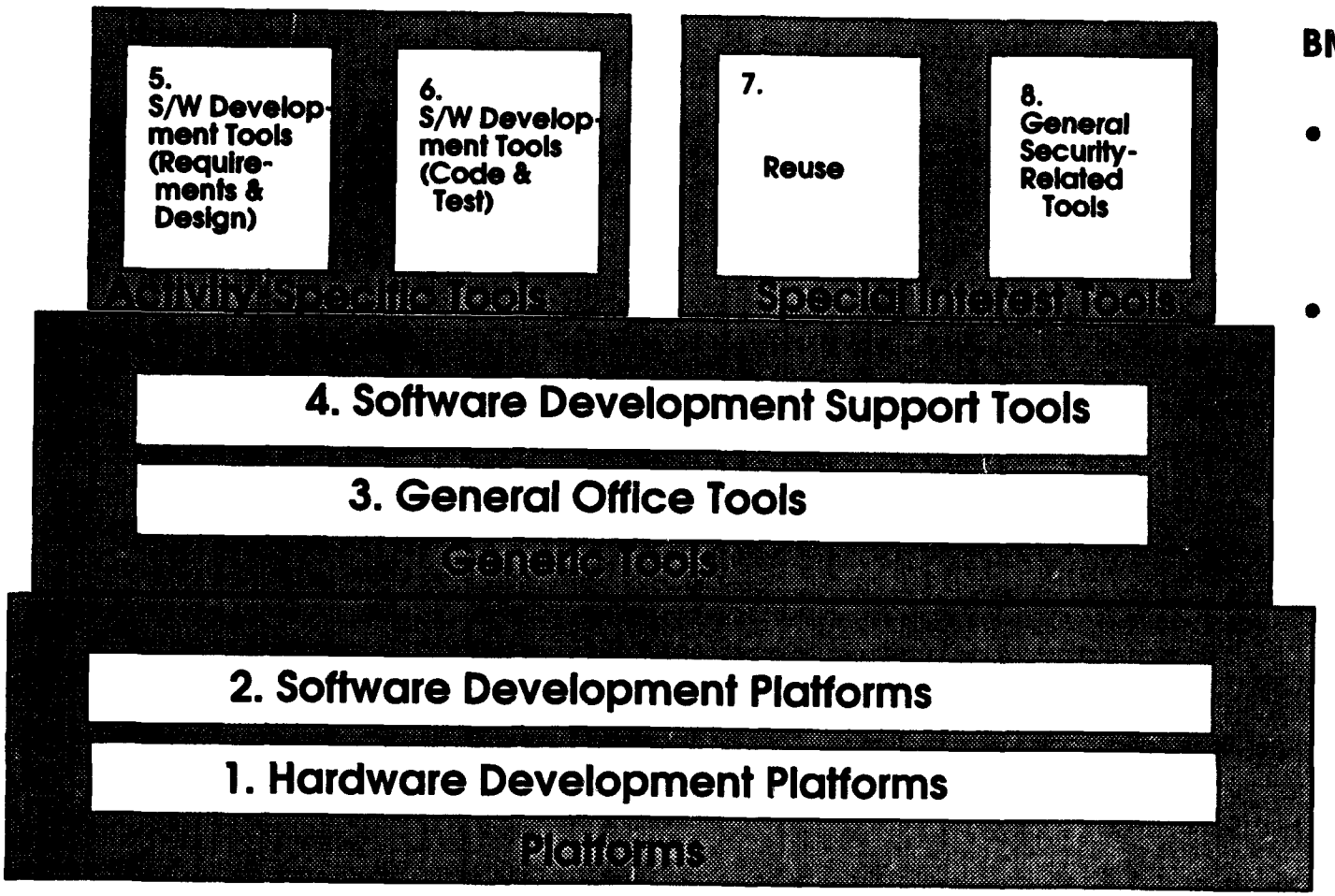
- The second class, generic tools, contains the two categories of general office tools and software development support tools. An example of a general office tool product type is "DBMS". An example of a software development support tool product type is "version and configuration management".

- The third class, activity-specific tools, contains the two categories of software development tools (requirements & design) and software development tools (code & test). An example of a software development tools (requirements & design) product type is "cost & schedule estimation". An example of a software development tools (code & test) product type is "compiler".

- The fourth class, special interest tools, contains the two categories of reuse and general security related tools. This class contains categories of tools which are of special interest to BMDO. An example of a reuse product type is "reuse library locator/browser". An example of a general security-related tool product type is "virus detector."

The data collected about the BMDO SEEs shows that over 230 different products in these eight categories are currently in use. Ninety percent of these tools are commercially available; only 10% are proprietary or developed in-house. Some of these products are hardware platforms, operating systems, and user interfaces (X-windows). Excluding these three types of products still yields over 200 different tools currently in use.

Software Development Product Classes & Categories



BMDO SEEs

- **Over 230 different products**
- **90% COTS**

Hardware & Software Development Platforms (Categories 1 & 2)

- **37 different products in use**

The product types for hardware development platforms are machine type and network type. The product types for software development platforms are operating system, network software, user interface (for example, X-windows), repository, intertool communication, and tape management.

- **Machines and Operating Systems**

Unix workstations are used by all six projects. Apple Macintoshes or IBM-compatible PCs are used by five out of six projects, typically for project administration, management, and estimation activities. Software to support these activities is more plentiful on the personal-size machines and often cheaper than it would be for a Unix workstation. DEC VAX computers running VMS are in use by four of the six projects, although usually in conjunction with several other types of machines.

- **User Interface**

An X-windows based product is in use in all six projects. Note that this does not mean that every person in the contracting organization uses X-windows (for example, personal computer users do not necessarily use X-windows). Still, the use of X-windows is quite prevalent. Three projects specified that Motif is used, while one project specified that Open Look is used.

- **Intertool Communication**

Intertool communication facilities are not yet common in BMDO projects. Two of the six projects use a product based on HP's Broadcast Message Server (BMS).

Hardware & Software Development Platforms (Categories 1 & 2)

- 37 different products in use
- Machines and Operating Systems
  - Unix workstations used by all 6 projects surveyed
  - Apple Macintoshes or IBM-compatible PCs used by 5 of 6 projects, typically for project administration, management, & estimation
  - Among others, DEC VAX computers running VMS are used in some capacity in 4 out of 6 projects
- User Interface
  - X-Windows used by all
- Intertool Communication
  - Only 2 projects using a COTS product, both based on HP BMS

General Office Tools (Category 3)

- **20 different products in use**

The product types for general office tools are document preparation and word processing, graphics drawing, data base management system (DBMS), spreadsheet, electronic mail, document commenting groupware, and calendar/roster management.

- **Document Processing**

Of the document processing tools available for Unix platforms, Interleaf is the most commonly used, being found at four of the six projects. FrameMaker is used in only one of the six projects. On the PC-compatible platforms, Microsoft Word is common, being used by five of the six projects.

- **Graphics Drawing**

Four of the six projects use graphics drawing tools beyond what is available in the document processing tools. The most commonly used tools are Canvas (used in three projects) and MacDraw (used in two projects). In general, they are used in conjunction with a word processor which does not have a built-in drawing capability, and these generally used on a personal computers.

- **DBMS**

A wide variety of DBMSs are in use, with no predominant product across the projects. All of the ones noted were relational databases. Oracle, Sybase, Ingres, and DEC RDB are among those used. The DBMS in use is important because in-house tools are often built on it.

- **Spreadsheet**

The most common spreadsheet product is Microsoft's Excel, used by four of the six projects.

General Office Tools (Category 3)

- 20 different products in use
- Document Processing
  - 4 out of 6 use Interleaf, 5 out of 6 use Microsoft Word; only 1 uses Framemaker
- Graphics Drawing
  - 4 projects use these tools (3 use Canvas & 2 use MacDraw)
- DBMS
  - No obvious winner—Oracle, Sybase, Ingres, DEC RDB all are in use
- Spreadsheet
  - Microsoft Excel used by 4 out of 6

Software Development Support Tools (Category 4)

- **35 different products in use**

The product types for software development support tools are version and configuration management, problem tracking/reporting, project management, inspection results database or database analysis, and process enactment.

- **Configuration Management**

Fifteen different configuration management tools are in use. Twelve are COTS, while three are proprietary or are customizations of COTS tools.

- **Problem Reporting/Tracking**

Five of the six projects had determined which tools they would use for this function at the time of the interviews. Three of the projects are using proprietary tools, one is using a commercial tool in conjunction with proprietary forms, and two are using COTS tools.

- **Project Management**

Microsoft Project is used by four of the six projects.

- **Inspection Results**

Only two of the six projects are maintaining an inspections results database. The Lotus 1-2-3 and Microsoft Excel spreadsheet products, augmented by propriety forms, are being used are being used to capture the inspection results data.

- **Process Enactment**

No project is using a process enactment tool at this time. This is important because the BMD SEE is process driven. This survey shows that all projects will require a learning curve to use a process enactment tool.

Software Development Support Tools (Category 4)

- 35 different products in use
- Configuration Management
  - 15 different tools used, both COTS and proprietary
- Problem Reporting/Tracking
  - Most organizations' tools are in-house and proprietary
- Project Management
  - 4 out of 6 use Microsoft Project
- Inspection Results
  - Rarely done (2 out of 6); spreadsheets with proprietary forms used
- Process Enactment: No current use found

Software Development Tools (Requirements & Design) (Category 5)

- **36 different products in use**

The product types for software development tools specific to requirements and design are cost and schedule estimation tools, requirements and design tools (for example, Cadre Teamwork), requirements tracing and change impact analysis tools, simulators or animators, document generator, reverse engineering tools, and requirements capture tools.

- **Cost and Schedule Estimation**

All six projects use tools based on the COCOMO model; four projects use REVIC and two use WICOMO.

- **Requirements and Design Tools**

DCDS, currently planned for integration into the BMD SEE, is not being used by any project as a design tool. Four projects are using Cadre's Teamwork and two projects are using i-Logix's Statemate. RDD-100 and IDE's Software through Pictures (StP) are among the several other tools being used by one project.

- **Requirements Tracing Tools**

Two projects expressed dissatisfaction with the tools available for doing requirements tracing. Of the tools currently in use, Cadre's Teamwork RqT is the most common, being used by three of the six projects. Two of the projects are using the Microsoft Excel spreadsheet to trace requirements. One project is using RDD and another is using HyperCard, which is a hypertext system.

Software Development Tools (Requirements & Design) (Category 5)

- 36 different products in use
- Cost & Schedule Estimation
  - All use COCOMO-based models; 4 use REVIC, 2 use WICOMO
- Requirements and Design tools
  - DCDS not used for design
  - 4 use Cadre Teamwork, 2 use i-Logix Statemate
  - RDD-100 and several other tools mentioned once
- Requirements Tracing tools
  - Mentioned as problem area
  - Cadre Teamwork RqT (3 of 6) most common
  - Two using Microsoft Excel (spreadsheet); others trying RDD and Hyper-Card (hypertext system)

Software Development Tools (Code & Test) (Category 6)

- **88 different products in use**

The product types for software development tools specific to code and test are text editor, language-sensitive editor, compiler, assembler, builder (for example, *make*), static analyzer (including metrics), pretty printer, symbolic debugger, test coverage analyzer, profiler and timing analyzer, other testing tools, compiler library query tool, and a remote compilation system.

- **Editors**

Many different text editors are in use. Vi is used on all six projects. Both DEC's LSE (language sensitive editor) and Cadre's LSE are used on two projects. Emacs, which will be integrated into the BMD SEE, is currently being used on one project.

- **Compilers**

Twenty-seven different compilers are being used. This large number is due to the large number of development platforms and target platforms in use, as well as differing needs, e.g. fast compilation of source, fast or small executable. Of the 27 compilers in use, 12 are Ada compilers. Each project uses an average of four different Ada compilers. The Ada compilers in use are VAX Ada, DEC Ada, Intermetrics Ada, Tartan Ada, Rational Ada, Alsys Ada, Sun Ada, Verdix Ada Development System (VADS), DDCI Ada, TLD Ada, RS/6000 Ada, and GEM Tech Desktop Ada.

- **Static Analysis**

DRC's Adamat is used on three projects and DEC's DECset SCA is used on two projects. Verilog's Logiscope is used on one project, for both static analysis and test coverage analysis.

Software Development Tools (Code & Test) (Category 6)

- 88 different products in use
- Editors
  - Many in use; 6 out of 6 respondents still use vi
  - DEC's LSE & Cadre's LSE each used by two projects
- Compilers
  - 27 different products in use; all projects use more than one
  - 12 different Ada compilers in use
  - Average of 4 different Ada compilers per project
- Static Analysis
  - — DEC's DECset SCA (2) and DRC's AdaMAT (3) most common. Verilog's Logiscope also mentioned.

Reuse & Security (Categories 7 & 8)

- **25 different products in use**

The types of products in the reuse category are reuse library locator/browser, reuse library contents, GUI prototyping (including GUI builders and components), domain-specific program generators, and embedded real-time operating systems or kernels. The types of products in the general security-related tools are virus detection, environment integrity checks, system administrative aids, and password generators or checkers.

- **Only one product used by multiple projects**

Of the 25 different tools in use in these two categories, UNAS/SALE is the only product used by more than one project. It is being used by two projects. UNAS stands for Universal Network Architecture Services, and was originally developed by TRW. UNAS is a suite of portable and reusable building blocks, services, and instrumentation for architecting a distributed software system. SALE stands for the Software Architect's Lifecycle Environment, and is a "UNAS compiler." SALE includes a graphical user interface, structural semantics checking, and a verification system which can detect certain dangerous distributed system conditions (such as deadlock, livelock, and race conditions). SALE generates code and some architectural reports.

- **Reuse Libraries**

Two projects are using reuse libraries; both libraries are proprietary.

- **Security**

Security is generally based on operating system capabilities (passwords, file protection, etc.) and on configuration management tools. Virus detection on Macintoshes and IBM-compatible PCs is also common.

Reuse & Security (Categories 7 & 8)

- 25 different products in use
- Only one product is used by multiple projects
  - (UNAS/SALE used by two organizations)
- Reuse Libraries
  - Two projects use proprietary reuse libraries
- Security
  - Generally based on operating system and CM tools, with virus checking for Macintoshes & IBM-compatible PCs.

Comparison of the BMD SEE with BMDO Element SEEs

This table shows how the products planned for integration into the BMD SEE compare to the products actually in use by the BMDO projects.

- All six BMDO projects use Unix workstations. The Sun SPARC is used by four of the projects and is a reasonable platform choice for the initial release of the BMD SEE. However, many other Unix workstations are in use (five projects use DEC stations, two use HPs, and 2 use IBMs), and it would be wise to support these Unix workstations as well.

- A user interface based on X-Windows is a good choice for the BMD SEE; all six projects are using it.

- Most projects are not currently using an intertool communications ability or a repository.

- Framemaker is not a common choice among the BMDO projects; Interleaf is the preferred product for Unix platforms.

- Cadre's Teamwork is the most commonly used requirements and design tool, so the BMD SEE project made a reasonable selection in choosing to integrate it. On the other hand, the integration of DCDS may prove fruitless, since it is not currently being used during development. IDE's Software through Pictures and i-Logix's StateMate are used more often than DCDS during development.

- Emacs is used on two of the six projects, so it is not a bad choice for integration into the BMD SEE. However, vi is in use on all six projects.

- Verdix VADSpro is as good a choice as any for an Ada compiler integration, but many other compilers will require integration.

- There are many other tools in use on the six BMDO projects that are not scheduled to be integrated into the BMD SEE.

Comparison of the BMD SEE with BMDO Element SEEs

Product Category	BMD SEE	BMDO Elements
1. Hardware Development Platforms — Machine type	Sun SPARC	Many different; esp. Unix, IBM-compatible PCs, & Apple Macintoshes; some DEC VAX (VMS)
2. Software Development Platforms — User Interface	X-windows	X-windows used extensively
2. Software Development Platforms — Intertool Communication, Repository	Emeraude: PCTE	Some HP BMS; most use none
3. General Office Tools — Document Preparation	Frame: Framemaker	Most Interleaf, some also use Microsoft: Word
5. Software Development Tools (Requirements & Design) — Requirements & Design	DCDS, Ascent Logic: RDD-100, Cadre: Teamwork	Cadre: Teamwork; some Ascent Logic: RDD-100, IDE: StP, i-Logix: Statemate, others
6. Software Development Tools (Code & Test) — Text editors	Emacs	Mostly vi, many others including emacs
6. Software Development Tools (Code & Test) -— Compilers	Verdix: VADSpro	Many different ones, including Verdix
Others	None	Many other tools

BMD SEE Issues Raised by this Survey

Several issues have been raised, based on the results of this survey.

- **Address how roughly 200 tool adapters or translators can be provided to match actual development environments**

Some of the most commonly used of them are Interleaf, vi, Microsoft Project, DRC AdaMAT, Verilog Logiscope, IDE Software through Pictures (StP), and i-Logix Statemate. It will be difficult to provide tool adapters or data translators to accommodate all 200 of these tools. Adding adapters for the most commonly used tools would make no significant difference in the basic problem.

- **Address how contractor data can be delivered into the BMD SEE if the SEE has no similar tool**

If the BMD SEE does not provide tools similar to all of those currently in use among the six BMD projects, then there will be a problem trying to deliver the data produced by these tools into the BMD SEE.

- **Address how to provide contractors experience in new areas: process enactment, tool integration, PCTE, transitioning data to the BMD SEE**

The BMD SEE will introduce capabilities or requirements in several areas where BMD projects have little or no experience. BMDO needs to address how to provide contractors experience in these new areas.

BMD SEE Issues Raised by this Survey

- Address how roughly 200 tool adapters or translators can be provided to match actual development environments
  - Commonly used tools include Interleaf, vi, Microsoft Project, and to a lessor extent, DRC AdaMAT, Verilog Logiscope, and i-Logix Statemate
  - Addition of adapters for these commonly used tools would make no significant difference in the basic issue
- Address how contractor data can be delivered into the BMD SEE if the SEE has no similar tool
- Address how to provide contractors experience in new areas: process enactment, tool integration, PCTE, transitioning data to the BMD SEE

Briefing Outline

This portion of the briefing presents information on the advanced COTS SEEs. These are categorized into three basic types: those based on the Portable Common Tool Environment (PCTE), those based on A Tool Interface Specification (ATIS), and those based on messaging. Examples of PCTE-based COTS SEEs are EAST and Enterprise II. Examples of ATIS-based SEEs are Atherton's Software Back-Plane, Verdix' VADS APSE, and CRI's Life*Cycle. Examples of messaging-based SEEs are Hewlett-Packard's SoftBench and Sun's ToolTalk.

A "SEE framework" is considered to be a relatively fixed set of core facilities [NIST 1991] supporting a SEE. The concept of advanced SEEs has been in existence since the mid-1970s, and there are many SEE-related efforts, approaches, and experiences [Penedo 1988] [Brown 1992a] [Brown 1992b, 243-271].

Since this briefing is intended for an audience which is already familiar with advanced COTS SEEs, only two slides for each type will be presented. The first slide presents recent activities, where "recent" is defined as within the past year. The second slide for each type presents the strengths and weaknesses of that type of SEE. The appendix includes additional information about different COTS SEEs for those who are not already familiar with them.

Briefing Outline

- Overview
  - Background
  - Briefing Purpose and Scope
- Survey of BMDO Element SEEs
- → Advanced COTS SEEs
  - PCTE
  - ATIS
  - Messaging-based
- Conclusions

Recent (ECMA) PCTE Activity

ECMA PCTE stands for European Computer Manufacturers Association Portable Common Tool Environment.

- **NIST**

  NIST (National Institute of Standards and Technology) is sponsoring North American PCTE Initiative (NAPI) and Integrated Software Engineering Environments (ISEE) - a testing facility within NIST's Computer Science Laboratory. It participates in North American PCTE User's Group (NAPUG) and PCTE Information Management Board (PIMB). NIST will have a liaison role in ECMA/TC33.

- **SEI**

  Carnegie-Mellon University's Software Engineering Institute (SEI) has conducted experiments related to tool integration as well as the integration of PCTE with components of other frameworks message based frameworks, HP's Broadcast Message Server (BMS).

- **Paramax**

  The fact that Paramax has encountered semantic problems in the standard is important. Until the standard has been "debugged" by trying to implement it, it is difficult to judge its value or to recommend expending effort to "improve" it. For this reason, IBM's attempt at an early "full" implementation of ECMA PCTE is important as a basis for evaluating the standard itself.

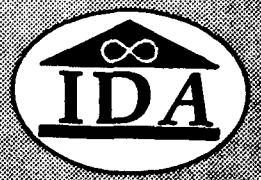
- **IBM**

  IBM's implementation will be partially ECMA PCTE Level 1 (Core Module) conformant. Specifically, IBM has omitted Notification, Replication, and Archiving. In place of archiving, considered too primitive, IBM plans to provide a new backup/restore facility. In addition, IBM has not implemented process execution services that overlap with operating system services.

  IBM has added extensions which include administration facilities, authentication, environment and workstation tools (such as a Shell and Graphical User Interface (GUI)). IBM is working with third party tool vendors to integrate at least one tool for each phase of the Software Development Life Cycle.

  IBM states they are committed to full ECMA PCTE (level 4) compliance. In addition, IBM is actively working on understanding the problems of handling fine-grain objects and overcoming those problems.

Recent (ECMA) PCTE Activity



- **NIST**
  - — Establishing validation test suite - promoting PCTE as a public domain tool integration standard
- **SEI**
  - — Experimenting with tool integration using PCTE (Emeraude) and BMS (ToolTalk)
- **Paramax**
  - — Creating first implementation of Ada binding - 20% complete - found semantic problems in standard
- **IBM**
  - — ECMA PCTE implementation in Beta as of 3/93; has announced plans to release in 9/93 for RISC/6000, HP, possibly other platforms

PCTE Strengths and Weaknesses

Strengths

While ECMA PCTE addresses a number of aspects of frameworks, its major strength is generally viewed to be object management and support for security, accounting and auditing. The security features are of special interest since they seem to be so often ignored by other framework technologies. ECMA PCTE specifies security capabilities to implement both mandatory access control (MAC) and discretionary access control (DAC).

There is strong support for ECMA PCTE from some organizations. The US federal government (NIST and the DoD) and some large commercial vendors (IBM and Bull) are promoting ECMA PCTE either through participation in standardization efforts, or by developing environments based upon ECMA PCTE. In addition, International Standards Organization (ISO) is considering ECMA PCTE for standardization.

Weaknesses

There is not much activity by tool vendors to integrate their products into ECMA PCTE-based environments. ECMA PCTE is not object-oriented and does not support fine-grain objects. Many believe object-oriented frameworks will be better able to handle fine-grained objects, and that they will be more accepted by users. In addition, although tool integration can be achieved, really useful levels of integration are still complex and require source code changes by tool vendors (and, therefore, a commitment by tool vendors to support ECMA PCTE).

In the United States, ATIS- and BMS-based frameworks seem to be more widely accepted, however, the total number of users of any type of framework is still small.

The lack of any commercial quality implementation of ECMA PCTE must be viewed as a weakness. Success of implementations of earlier generations of PCTE frameworks and recent announcements by IBM and Bull should help to reduce concern in this regard.

A major weakness of ECMA PCTE is shared by the BMS- and ATIS-based frameworks: there is still not enough experience in the use of these frameworks, and the SEEs based upon them, to determine when and how they are beneficial. This experience cannot be gained unless developers of many large projects are willing to perform disciplined software development using an environment, keep careful records of their experience, and then report that experience. However, many SEE users view positive results as a competitive advantage they are unwilling to share with competitors.

PCTE Strengths and Weaknesses

Strengths	Weaknesses
Tool integration support	Not as much tool vendor activity as desired - full integration is difficult
Object management (actually based on the ERA Model)	Competition from OO technologies No support for fine-grain objects
Security features (accounting, auditing)	
Strong support from some large government and commercial organizations (e.g., NIST, DoD, & IBM) An ECMA standard; appears closest to becoming an ISO standard	ATIS and BMS oriented products are relatively more successful commercially
	As yet no complete ECMA PCTE production implementation

ATIS Recent Activities

- **Atherton Technology**

In May of 1993, the President and the Vice President of Engineering of Atherton Technology bought all Atherton shares from the French company, Thomson CSS, making Atherton a totally American company. Although Atherton has experienced a reduction in size, it currently employs R&D staff members with a cumulative total of 28 years of experience with the Software BackPlane. Atherton has consultancy arrangements with many original Software BackPlane developers, including the chief architect who now works for them half-time. Much of the Sales and Marketing staff are new, recruited from IBM, HP, and Sun. Two upcoming additions to the Software BackPlane are the Reuse Library Softboard and the CASE BackPlane, which brings HP's Broadcast Message Server into the Software BackPlane.

- **Verdix**

Verdix is placing more development and marketing effort on its VADSpro product, at the expense of its VADS APSE product. VADSpro is less expensive than VADS APSE, and much of the market finds it to be a more suitable product for their software development needs. A small part of the market is willing to pay for the increased cost of the tool integration capabilities that come with VADS APSE.

- **CRI**

CRI has obtained a non-exclusive right to the Atherton Software BackPlane source code, and is now able to make modifications and additions to it to suit its product line. In 1992, CRI established an office in Seattle, Washington, to support its business in the American aerospace industry. They are projecting growth for this office from its initial 20 staff members to 200 staff members by the year 1995.

- **Movement toward off-the-shelf, already-populated SEEs**

Both CRI and Verdix have indicated that their market does not want to integrate tools into an SEE, but rather prefers to buy a ready-to-go, already-populated SEE. Verdix has responded by developing and marketing its VADSpro product. CRI has responded by integrating more COTS tools into Life*CYCLE and by developing and adding more functionality to the base Life*CYCLE product.

ATIS
Recent Activities

- **Atherton Technology**
  - Bought themselves out from Thomson
  - Reduced development staff, but key technical people still consulting
  - Integrated HP BMS into Software BackPlane
- **Verdix**
  - Reduced development and marketing of VADS APSE
- **CRI**
  - Obtained non-exclusive rights to Atherton Software BackPlane source code
  - Established office in US (Seattle, WA) in 1992
- **Movement toward off-the-shelf, already populated SEEs**
  - VADSpro has stronger market presence than VADS APSE
  - CRI continually adds functionality to base Life*CYCLE product

ATIS Strengths and Weaknesses

Strengths

One major strength of the ATIS-based SEEs is their object-oriented repository. An object orientation allows the repository to handle data from many different tools without causing an explosion in the number of procedural interfaces required to process the data. The PCTE community has recognized the advantages of object orientation; they have become member of the OMG (Object Management Group) and are trying to make PCTE more object oriented.

A second strength is in CRI's signed commitment to the European Space Agency to enhance and maintain its Life*CYCLE product beyond the year 2000. This commitment is evident in the tools CRI has developed for its Life*CYCLE SEE, such as a traceability tool and a process enactment tool, as well as in the large number of COTS tools it has integrated into Life*CYCLE.

Weaknesses

ATIS-based SEEs, like PCTE-based SEEs, are capable of a strong integration of tools into their environments. Tool data can be "understood" and acted upon by other tools. However, this tight integration comes at an increased cost in the integration effort. If only a loose integration is sought, such as that provided by a messaging-based SEE, then a much smaller integration effort is needed.

The Verdix VADS APSE appears to provide a user-friendly environment for the specification, design, coding, and testing of Ada software. Unfortunately its configuration management capabilities, which have been crafted to support Ada units and libraries, are inefficient for large-scale development in other languages.

CRI is a foreign-owned company (Danish). IBM-Denmark owns 50% of it, and four Danes own the other 50%.

Although there are on-going efforts to standardize ATIS, supported by several companies including Atherton, DEC, Boeing, Loral, and Verdix, there is no international-level standard available today.

ATIS
Strengths and Weaknesses

- **Strengths**
  - Object-oriented repository
  - CRI committed to enhancements and maintenance of Life*CYCLE product beyond year 2000
  - CRI Life*CYCLE integrated with relatively large number of COTS tools
- **Weaknesses**
  - As with PCTE, significant effort required to integrate tools fully
  - VADS APSE CM adapted to Ada, not as efficient for other languages
  - Foreign ownership of CRI
  - No standard in place

Messaging-based SEE Recent Activities

- **CASE Communiqué (HP, IBM, et al.) and CASE Interoperability Alliance (Sun et al.)**

CASE Communiqué is an industrial consortium which was formed in October 1991 by Hewlett-Packard (HP), IBM, Informix, and Control Data. As of February 2, 1993, it had 149 member organizations and 317 participants, including Cadre, IDE, Interleaf, Frame, and Alsys. CASE Communiqué is developing a standard message set which can be mapped to a BMS-based framework environment. Its formal mission statement (from May 1992) is "to provide an open forum dedicated to the cooperative development of industry acceptable standard specifications for control integration in CASE framework environments."

On October 2, 1992, DEC, Silicon Graphics, Inc., and SunSoft proposed at CASE World "The CASE Interoperability Message Sets." This document appeared to be a competitor to CASE Communiqué's work, since defined at a high level (semantics, no syntax) the abstract messaging service environment and some user scenarios, most of them at the edit-compile-debug stage of development but also including analysis and design [Digital 1992].

In November 1992 the ANSI standards committee X3H6 developed a project proposal to attempt to get these various groups to work together. In January 1993 a joint proposal was presented by CASE Communiqué and the CASE Interoperability Alliance (CIA, which includes SunSoft and Silicon Graphics, Inc.) to ANSI X3H6 [CC 1993]. The document is very high level since in particular it does not define the message sets, but it can be viewed as a signal that the various groups are attempting to work together. On October 21-22, 1993, CASE Communiqué and CIA are to present their jointly defined standard message specifications (called "servicegrams").

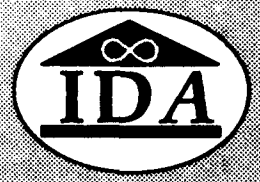
- **Common Open Systems Environment (COSE) agreement**

IBM, HP, SunSoft, and Unix Systems Laboratory (USL) have since announced a joint effort called the Common Open Systems Environment (COSE). COSE is to provide users with a common cross-platform Unix environment. Univel and the Santa Cruz Operation (SCO) also announced support for the effort. At least one of the drivers of COSE is the impending release of Microsoft's Windows NT, which all see as dangerous to their market share [Azzara 1993, Horwitt 1993, Wagner 1993]. However, caution seems warranted since some previous unification efforts between different Unix organizations (particularly ACE) have failed [Johnson 1993, Krill 1993].

A COSE developers' conference is planned for October 1993 and products are to be available in the first half of 1994. Among the products included in this agreement are the Open Software Foundation's (OSF) Motif, Sun's Tooltalk and DeskSet tools, and HP's Visual User Environment (VUE). Also to be included in this agreement are systems management and products for distributed systems (somehow fusing Sun's ONC+ and OSF's DCE).

HP stated to IDA that they viewed ToolTalk as similar to the underlying BMS in HP SoftBench, and thus ToolTalk would simply replace BMS with relatively fewer changes at higher levels. However, HP claimed that its product, HP Encapsulator, would be the main interface and that no one would use the ToolTalk interface. Sun claimed to IDA that ToolTalk would be the main interface used by tools, and not the HP encapsulator. There is still some uncertainty as to what interface an application developer should use for integrating tools into a messaging-based SEE.

Messaging-based SEE Recent Activities



- CASE Communiqué (HP, IBM, et al.) and CASE Interoperability Alliance (Sun et al.)
  - CIA proposed its own message set (Oct 1992)
  - Jointly presented a high-level agreement to ANSI X3H6 (Jan 1993)
- Common Open Systems Environment (COSE) agreement
  - IBM, HP, SunSoft, USL to provide common user environment (reacting to Microsoft NT)
  - Includes OSF's Motif, HP VUE, SunSoft ToolTalk and HP Encapsulator; roles of the latter two currently unclear
    - — HP claims that ToolTalk simply replaces BMS, and HP Encapsulator is to be the main interface
    - — Sun claims that ToolTalk is to be the main interface and not HP encapsulator

Messaging-Based SEE Strengths and Weaknesses

Strengths

One of the key strengths of messaging-based environments is that a number of tools have been integrated into them in some way.They are available now, and can work in conjunction with data integration approaches. For HP SoftBench, these tools include all the tools in HP SoftBench plus its optional add-on products, Alsys' Ada compilation system, IPL's SofTest (a software testing tool), IDE's Software through Pictures, and Cadre Teamwork (which is to be improved in the future). For Sun ToolTalk, these include the DeskSet tools and Sun Ada, with more promised.

Weaknesses

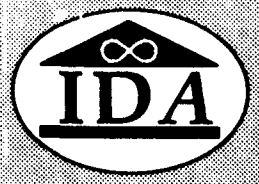
A weakness is that messaging-based SEEs do not, by themselves, integrate data. They usually assume that the communicating tools share some sort of repository; one acronym common in the literature is a WUSR—a 'well-understood shared repository'. In all of today's environments the WUSR is a Unix-based file system using NFS, though some experiments have been done using PCTE as the WUSR.

Currently standardization is in flux. ANSI X3H6 is actively working on specifying a messaging standard for such systems, but it has not released a specification and probably will not for some time.

Security is currently not well addressed in these systems. Currently there is no security in implementations to ensure that only certain tools register to handle certain requests, and this provides opportunities for spoofing. Most messages are intent on supporting a single user, with the tools handling most inter-person control. Tools that are especially considered for handling multi-person control are the process management system(s), the configuration management system (which is responsible for getting the correct product versions), mail (for notifying humans), and, to a limited extent, the operating system (which is to provide simple read and write permissions for each user). The current implementation of HP SoftBench requires that all users belong to the same (Unix) user group, which unfortunately makes it even more difficult to restrict data access to specific groups of people.

As mentioned earlier, the key to this approach is standardizing the messages. If two tools agree on a message set, then at least those tools can communicate using that message protocol. However, for there to be a great deal of integration, tool messages must be defined for a number of different types of tools. Vendors have recognized this and have formed consortia to deal with this issue.

Messaging-based SEE Strengths and Weaknesses



Strengths

- Many tools already integrated (at some level)
- COTS products, available now
- Can work *with* data integration

Weaknesses

- Does not, by itself, integrate data
- Standardization ongoing, in flux
- Security not strongly addressed: most messages support single user. Depends on CM, process management, and OS for multi-person control
- If tool type's messages are not already standardized, other tools won't know how to access it (without extra effort)

Briefing Outline

The followin_ re conclusions based on the survey of BMDO element SEEs and advanced COTS SEEs.

Briefing Outline

- Overview
  - Background
  - Briefing Purpose and Scope
- Survey of BMDO Element SEEs
- Advanced COTS SEEs
  - PCTE
  - ATIS
  - Messaging-based
- Conclusions

Conclusions

- **More than 230 different products are used for BMD software development.**

Even if the different hardware platform and operating system products are ignored, there are still over 200 different products used for software development.

  - This large number of different products means that a huge investment will be necessary for the BMD SEE program to come close to matching BMD element development environments.
  - The use of COTS SEEs would eliminate the risk associated with the BMDO SEE development, and they are available immediately. However, they still do not provide significant coverage of the products used in development.
  - Requiring a common toolset would reduce the toolset gap somewhat, but in at least some cases there are valid reasons for such a diverse set of products. For example, different Ada compilers are used because each has a niche where it is best suited—one may be especially good at generating code for Motorola 68K machines, while another is very good at generating code for 1750A processors.

- **COTS SEEs not very mature.**
  - Currently there is no complete ECMA PCTE implementation. Emeraude v12 is an implementation of PCTE 1.5, with extensions to make it closer to ECMA PCTE, but it is not complete. IBM's announced PCTE will not completely implement the ECMA specification either.
  - ATIS-based SEEs have a number of advantages, for example, they are object-oriented. However, CRI's demonstration of its Life*Cycle product showed many rough edges. It was not possible for IDA to determine if these problems were very temporary and would quickly disappear or if they were signs of deeper problems.
  - Messaging-based SEEs are gaining some acceptance commercially and in the BMD elements. However, the details of interface standardization are still uncertain. Currently there is some debate on what application programming interface (API) should be used to interface to a messaging-based system.

Conclusions

- More than 230 different products used for BMD software development
  - Huge investment will be necessary for SEE program to come close to matching development environments
  - COTS SEEs can help, but they still do not provide significant coverage of the products used in development
  - Requiring a common toolset would reduce this somewhat, but there are valid reasons for such a diverse set of products
- COTS SEEs not very mature
  - No complete ECMA PCTE production implementation
  - CRI's demonstration (ATIS based) showed many rough edges
  - Messaging-based SEEs gaining acceptance commercially and in elements; there are interface standardization questions

Appendix: Advanced COTS SEEs

- This appendix provides background material on advanced COTS SEEs
- For each of the three types of COTS SEEs (PCTE, ATIS, messaging-based), appendix presents
  - Introduction
  - Current Products

PCTE-Based SEEs

This set of slides presents information related to PCTE, the Portable Common Tool Environment standard from ECMA (European Computer Manufacturers Association). The general outline of the presentation is:

- **Introduction**

  Introduction to PCTE, including an overview of the evolution of the PCTE standard and the goals and intentional omissions of the standard

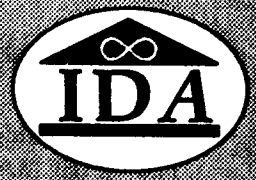
- **Current Implementations**

  Implementations of the three generations of PCTE standards

- **Relationship to other Frameworks**

- **Product Integration Summary**

PCTE (Portable Common Tool Environment) Based SEEs



- Introduction
- Current Implementations
- Relationship to other Frameworks
- Product Integration Summary

Introduction to PCTE

- **A Public Tool Interface (PTI) - not an implementation**

  PCTE (Portable Common Tool Environment) is an *interface standard* established by the European Computer Manufacturers Association (ECMA). That is, it defines program-callable interfaces, but it is not an implementation. It is not a complete framework; there are desirable components that are not addressed

- **Four levels of conformance**

  (1) Core Module Conformance (includes object management; schema management; file, pipes, and devices; volumes and archives; process execution; messaging; notification; concurrency and integrity control; replication; networking and distribution; discretional (DAC) security)

  (2) Core Module + Mandatory (MAC) Security

  (3) Level 2 + Auditing Services

  (4) Level 3 + Accounting Services

- **Application Program Interface (API)**

- **Abstract Language Definition, C and Ada Language Bindings**

  There is an abstract language specification (ECMA 149) as well as bindings to the languages C (ECMA 158) and Ada (ECMA 162). An effort was underway to create C++ bindings; however, that effort is now on hold.

- **ECMA standards**

- **Being considered for an ISO standard**

Introduction to PCTE

- A Public Tool Interface (PTI) - not an implementation
- Four levels of conformance
- Application Program Interface (API)
- Abstract Language Definition, C and Ada Language Bindings
- ECMA standards
  - ECMA 149 - PCTE Abstract Specification
  - ECMA 158 - PCTE C Language Binding
  - ECMA 162 - Ada Language Binding
- Being considered for an ISO standard

PCTE Evolution

We are considering the three major generations of the PCTE standard.

- **PCTE 1.5 - 1988 - culmination of early efforts - basis for first implementations**

  PCTE 1.5 has been implemented by Emeraude (in particular the Object Management System). East and Entreprise II are environments built upon Emeraude.

PCTE+ - 1988 - closely related to PCTE 1.5, but more thought given to security and independence from UNIX

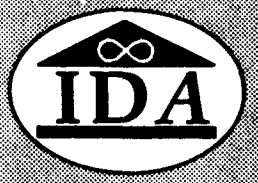
  PCTE+ has no complete implementations.

- **ECMA PCTE - 1990 - based on PCTE+**

  ECMA PCTE has been partially implemented by IBM and is supposed to be released in September, 1993.

Only the ECMA PCTE standard is under active development. PCTE 1.5 and PCTE+, as standards, are a "dead end". Vendors are continuing to enhance and add tools to their PCTE 1.5 implementations. They claim that upgrading to ECMA PCTE will not be too difficult because ECMA PCTE has as many commonalities with PCTE 1.5. Unless they change their current implementation strategy, East and Entreprise II must wait for an ECMA PCTE version from Emeraude before they can be conformant.

PCTE Evolution



- PCTE 1.5 - 1988 - culmination of early efforts - basis for first implementations
- PCTE+ - 1988 - closely related to PCTE 1.5, but more thought given to security and independence from UNIX
- ECMA PCTE - 1990 - based on PCTE+
  - — This is the current standard

PCTE Goals and Non-Goals

No explicit backup text for this slide

PCTE Goals and Non-Goals

GOALS	NON-GOALS
Complete interface	Not an implementation
Support for tools written in a wide variety of languages	No built-in tools
Components • Object Management System (OMS) - actually an ERA Model • Process Control • Access Control (MAC and DAC)	Not complete - omitted are • Tool Integration Services • Query Services • Backup Services • User Interface Services • Task Management Services

Implementations of PCTE

Implementations of frameworks, SEEs, and CASE Tools in each of the three versions of PCTE are listed in the preceding table. All the *existing* implementations are PCTE 1.5. There are *no complete implementations* of either ECMA PCTE, or its immediate evolutionary predecessor PCTE+.

Implementations of PCTE

Category	PCTE 1.5	PCTE+	ECMA PCTE
Frameworks	Emeraude V12 - has additional services	Emeraude prototype EDS-Scicon prototype (VAX/VMS - ORACLE)	Partial implementation by IBM - avail. 9/93 "Commitments" from Emeraude/East (1993-partial) EDS-Scicon Softlab Heuristix DEC/ISSI (1993(?))
Environments	EAST - built on "V12" Entreprise II - U.I., tools and schemas - built on "V12" Bull ISD - announced Summer 1993		"Commitments" from STARS and Eureka projects
CASE Tools	Hyperweb - message passing capability - built on V12 object base Cobol Env from DMR Group	VULCAN (Heuristix & S2) - integrated toolset	Cadre planning to fully integrate Teamwork (1994/1995)

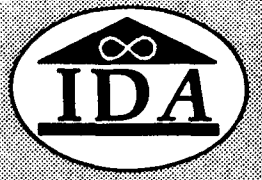
PCTE and Other SEE Frameworks

- **Vendors of other frameworks claim PCTE has limited support and capability and may not survive, but**

- **They already support PCTE, or are in the process of supporting it, or will support it if necessary**

Vendors of ATIS- and BMS-based frameworks are hedging their bets on ECMA PCTE. They either have determined how to integrate their framework with ECMA PCTE, or say their framework is orthogonal to ECMA PCTE and that integration will be possible if it is ever necessary. The fact that the development of an ECMA standard of C++ bindings has been put on hold would seem to give some support to these vendors' pessimism.On the other hand, the attention ECMA PCTE is getting from NIST, SEI, IBM, DEC and others indicates that some others are not so pessimistic.

PCTE and Other SEE Frameworks

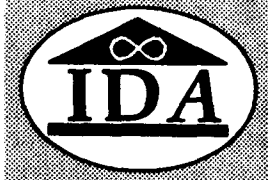


- **Vendors of other frameworks claim PCTE has limited support and capability and may not survive, but**
- **They already support PCTE, or are in the process of supporting it, or will support it if necessary**
  - **DEC will integrate ECMA PCTE with capabilities from ATIS and CORBA**
  - **CMU/SEI experimented successfully with integration of "V12" and BMS**

PCTE SEEs
Product Integration Summary

No explicit backup text for this slide

PCTE SEEs
Product Integration Summary



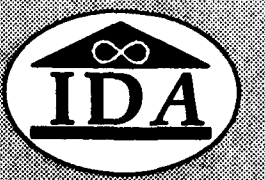
Product Integration Summary

Product Categories	East	Enterprise II	Emeraude v12
1 Hardware Development Platforms	• Sun, IBM Workstations	• Sun, HP, DEC, IBM Workstations • TCP/IP	• Sun, HP, IBM, DEC Workstations • TCP/IP
2 Software Development Platforms	• Unix • OSF/MOTIF + built in user interface • Emeraude v12	• Unix • NFS, X25, Ethernet • OPENLOOK or MOTIF • Emeraude v12	• Unix • NFS • OPENLOOK or MOTIF
3 General Office Tools	• Built in document editor • 2167A document models • Conversion to/from Framemaker, Word, Latex, ASCII • Conversion to Lotus 1-2-3	• E-Mail (office system environment) • Documentation Manager (uses TPS and/or Framemaker)	• User Interface • Text Editor (mouse driven) • OMS
4 Software Development Support tools	• Built in OMS, Configuration and Version Management, OMS Browsing Services • Process Management Services (2167A process model) • Project Management Services	• Project Manager Tool (interfaced with ARTEMIS) • Configuration Manager • Traceability Tool	• Emeraude_VCM (configuration/version manager)

PCTE SEEs
Product Integration Summary (Cont.)

No explicit backup text for this slide

PCTE SEEs
Product Integration Summary (cont.)



Product Integration Summary (cont.)

Product Categories	East	Enterprise II	Emeraude v12
5 Software Development Tools (Requirements and Design)	• Cadre Teamwork SA/RT • Cadre Teamwork SD • AdaNice • C-Nice • ObjectMaker	• IDE Software through Pictures (StP)	• Data modeling editor
6 Software Development Tools (Code and Test)	• Built in BUILD (make) • KEY ONE (Ada and C), vi • FORTRAN • C • Alsys Ada • CPX • ACT	• Support for Ada (Alsys and Telesoft), C, C++, LTR 3 • *Make* facility	• Makefile generator
7 Reuse	• OMS Browser and query facility	• Reusable Object Dictionary	• Data query and manipulation
8 General Security-Related Tools	• Built-in to OMS	• Built-in to OMS	• Built-in to OMS

Introduction: ATIS-based SEEs

- **A Tool Integration Standard (ATIS)**

The Atherton Software BackPlane served as the starting point for the ATIS (A Tool Integration Standard) tool interface specification, developed jointly by DEC and Atherton Technology. ATIS specifies the semantics and interfaces of an SEE that deal primarily with data integration; it also addresses some aspects of control integration. Like Software BackPlane, ATIS is object-oriented. ATIS has been endorsed by the Case Integration Committee.

- **Atherton Software BackPlane provides framework for creating SEEs**

The Atherton Software BackPlane (SBP), introduced in 1987 by Atherton Technology, provides a framework for creating SEEs. In particular, it provides an object-oriented data repository and repository services, as well as the capability to control access to data objects, to audit access of the data objects, to version objects, to create configurations of objects, to automate and control the software development process, and to integrate external tools into the environment.

Introduction: ATIS-based SEEs

- A Tool Integration Standard (ATIS)
  - Initially developed by DEC and Atherton
  - Based on the Atherton Software BackPlane
- Atherton Software BackPlane provides framework for creating SEEs
  - Object-oriented data repository
  - Access control on objects
  - Versioning of objects
  - Configuration Management
  - Process Enactment
  - COTS tool integration

Current Products: ATIS-based SEEs

- **ATIS**

Atherton Technology's Software BackPlane (SBP) served as the starting point for the ATIS (A Tool Integration Standard) tool interface specification, developed jointly by DEC and Atherton Technology. ATIS specifies the semantics and interfaces dealing primarily with data integration; it also addresses aspects of control integration. Like SBP, ATIS is object-oriented.

- **Atherton Software BackPlane**

The Atherton Software BackPlane (SBP), introduced in 1987 by Atherton Technology, provides a framework for creating SEEs. It was the first commercial product of its kind in the United States.

- **Verdix VADS APSE and VADSpro**

Developed by Verdix Corporation of Herndon, Virginia, the VADS APSE is an integration of the Verdix Ada Development System (VADS) and the Atherton Software BackPlane. The VADS APSE product was introduced in October of 1991. Verdix VADSpro, a subset of VADS APSE that eliminated the user capability to integrate tools, was introduced in 1992.

- **CRI Life*Cycle**

Computer Resources International's (CRI) Life*Cycle product has added traceability, process enactment, and change report tracking functionality on top of Atherton's Software BackPlane. Many third party tools have been integrated into the environment.

- **Loral CORCASE**

Loral's Software Productivity Laboratory has developed CORCASE, an SEE that integrates more than 10 CASE tools into the Atherton Software BackPlane. CORCASE also incorporates the DoD's 2167A software engineering process.

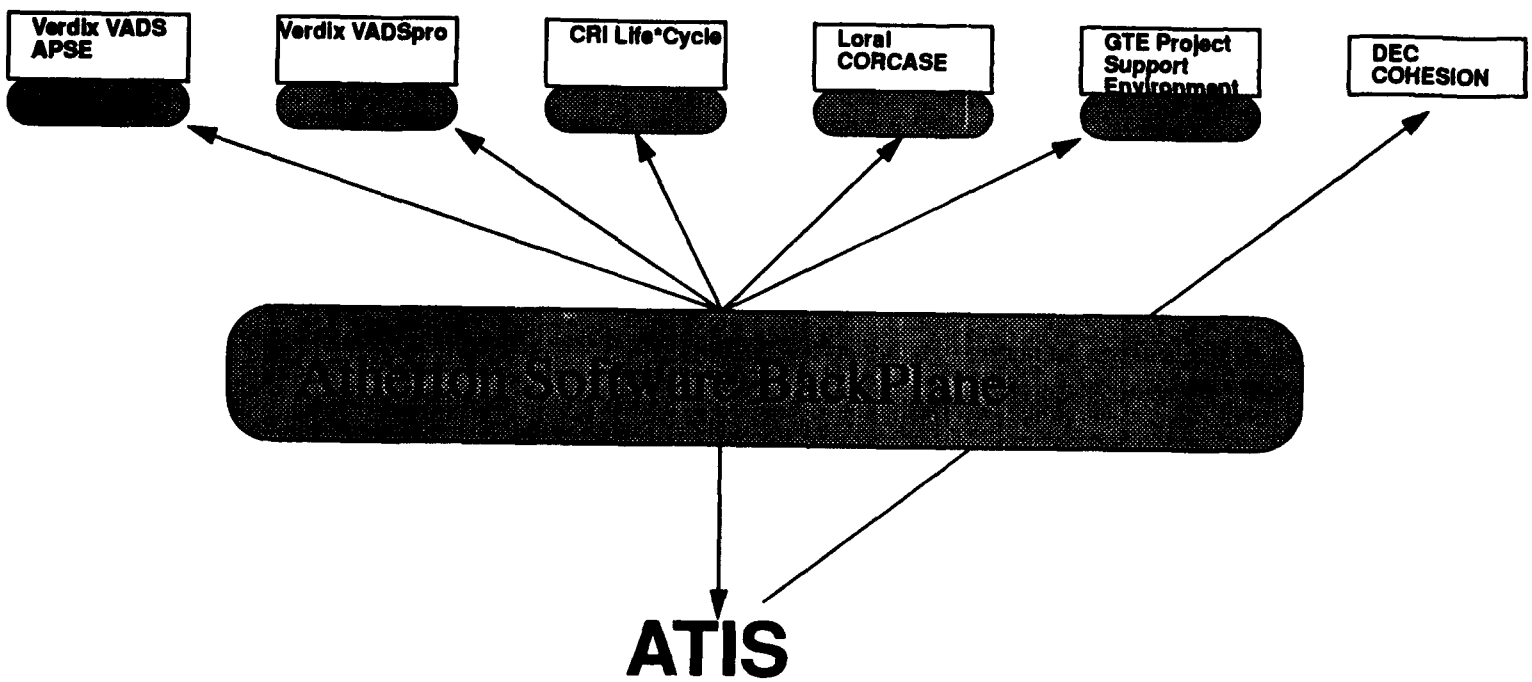
- **GTE Project Support Environment**

GTE Government Systems customized SBP to incorporate and automate their software development methodologies and procedures. They created a requirements analysis and change impact analysis tool through extensions to the SBP, and integrated third-party tools (VADS APSE, Software through Pictures, Interleaf) to supplement their environment.

- **DEC COHESION**

COHESION, based on ATIS, supports development of easily ported applications in an environment of heterogeneous platforms.

Current Products: ATIS-based SEEs



Atherton Software BackPlane

There are several characteristic features of the Atherton Software BackPlane, which will be present in all SEEs that are built on top of it.

- **Object - Oriented Data Base**

The Atherton Software BackPlane is built on an object-oriented data repository and data base management services. It comes with a pre-defined type hierarchy, which can be expanded to support new kinds of objects. Both attributes and methods are inherited, with single inheritance currently supported and multiple inheritance soon to be supported. The object orientation allows for easier expandability. Originally, Software BackPlane was based on an entity-relationship model. As more types of tool data were added to the type hierarchy, the number of new procedural interfaces grew tremendously and became unwieldy. At that point, Atherton added an object-oriented type manager, which permitted inheritance of methods along the type hierarchy. They report that the number of procedural interfaces was reduced from 450 to 17, demonstrating the advantages of object orientation.

- **Versioning and Configuration Management**

The Software BackPlane supports both successor versions of objects and branch versions of objects. Parallel development without conflict is supported through checkout, checkin, and merge functions. The checkout function produces a private writable copy of an object. A checkin function takes the private writable copy of the object and makes it a public read-only copy. *Multiple people can checkout an* object in parallel. The first person to complete their changes can do a checkin function. The other people must use the merge function to add their changes to the public, read-only version.

The Software BackPlane's predefined type hierarchy contains a 'collection' type, which allows the definition and management of collections of objects produced by multiple tools. Collection definitions can be versioned, just like other objects. This mechanism supports the tracking of the source versions, tool versions, and the exact tool options used to create each release of a product.

- **Access Control and Auditing**

The project manager is able to control the read and write access privileges of users, according to the project needs. Access control can be set on individual objects in the data repository, or on collections of objects. Access to tools can also be controlled.

History information for each object is stored in the data repository and can be queried at any time. Information about an object includes who manipulated the object, when, and for what reason.

Atherton Software BackPlane

- Object - Oriented Data Base
  - Contains pre-defined, expandable type hierarchy
  - Supports inheritance of both attributes and methods along the type hierarchy
- Versioning and Configuration Management
  - Supports checkout, checkin, and merge functions on versioned objects, including collections
  - Tracks the version of all sources and tools used to create a build
- Access Control and Auditing
  - Controls access to objects in the repository
  - Controls access to tools
  - Keeps a history of all accesses of objects in the repository

Atherton Software BackPlane

- **Tool Integration**

Tools are encapsulated into the Software BackPlane by adding subtypes to the predefined type inheritance hierarchy. Subtypes can be added for all of the objects that the tool manipulates. New methods can be added for the subtypes, if needed. In addition, existing, inherited methods for the new subtypes can be used without modification, used with modification, completely redefined, or ignored.

There are three increasing levels of tool integration. At the first level, the tool is registered with the environment. This allows the tool to be invoked by the environment. At the second level, the tool's arguments are registered with the environment. This allows the environment to version the tool's data. At the third level, methods for the tool's arguments are registered with the environment. As an example, registering methods that read and write a tool's data file interactively would support the cut and paste operations across tools. By knowing the internal types to which the cut and paste refer, the environment is able to perform the necessary type conversions.

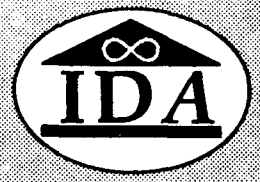
- **Generic Operating System Interface**

Software BackPlane provides all integrated tools with a Generic Operating System, which acts as a translator to native operating system functions. Once a tool is ported to Software BackPlane using the Generic Operating System services, no further changes are required to run the tool on any supported platform.

- **Process Enactment**

Process enactment allows the Software BackPlane to be customized to support and automate the software development processes of the organization. Process enactment is accomplished through the use of triggers. The type manager for the data repository allows triggers (code) to be added that will be executed either before or after a method for an object is executed. A trigger receives the same parameters as the method it is associated with. Set-up and take-down triggers can be stacked. Triggers can be used to implement notifications.

Atherton Software BackPlane



- **Tool Integration**
  - Encapsulates tools by adding subtypes to the predefined type hierarchy
  - Provides three levels of tool integration
- **Generic Operating System Interface**
  - Provides tool portability
- **Process Enactment**
  - Allows the environment to be customized to the organization's software development process

Verdix VADS APSE and VADSpro

- **Verdix**

Verdix Corporation, headquartered in Herndon, VA, was founded in 1982. It specializes in providing Ada-based software development tools for self-hosted and embedded applications, on a wide variety of hardware platforms. For the past six years it has been profitable in every quarter. In a survey done in 1991 by IBM, Verdix VADS products were found to have captured 70% of the Unix Ada market and 85% of the real-time Ada market.

- **VADS APSE**

Verdix's VADS APSE product was introduced in October of 1991. It is an integration of the Verdix Ada Development System (VADS) product and the Atherton Software BackPlane. As such, it possesses the characteristics of both.

The influence of the VADS product on the VADS APSE integration is apparent in the programming support given to the user. The VADS APSE environment supports an Ada language-sensitive editor (VADSedit) that provides syntax completion, syntax checking during edit, and source code navigation. Any of the Verdix Ada self-hosted, target-hosted, or cross compilers can be integrated into the environment. Similarly, source level debuggers to run either on a host or on the target are available. VADS APSE displays objects in terms familiar to Ada programmers, that is, as Ada units and Ada libraries rather than as files and directories. Version information and compilation status information is displayed graphically to the user. Dependencies among units and libraries are displayed to the programmer when compilations and builds are requested.

The influence of the Atherton Software BackPlane is equally apparent, in the versioning and CM support, in access control and process tailoring, and in its object orientation. VADS APSE provides a version management capability that supports parallel development. Versioning is done at the Ada unit and Ada library levels. Configuration management tracks all objects and information associated with a build, including design, source code, documentation, test cases, and tools used, including the version of the tools and the options supplied to them. Versioning and CM extend to all data produced by all tools that have been integrated into the environment. Access to project data can be can be controlled based on role's assigned to users. Process tailoring can be done by controlling the execution of a specific operation based on a user's role.

Existing tool integrations include Cadre Team*work*, IDE Software through Pictures, Frame Framemaker, and Interleaf 5. Users who wish to further customize the environment do so through the Integration SoftBoard product. Customization is done with C code, and training is strongly encouraged.

- **VADSpro**

The VADSpro product was created in response to market pressures for a less expensive, off-the-shelf environment. VADSpro contains all features of VADS APSE except the ability to integrate new tools into the environment.

Verdix VADS APSE and VADSpro

- Verdix
  - Leading supplier of Ada Development Systems
  - Profitable every quarter for the past six years
- VADS APSE
  - Provides strong support for implementation phase of Ada development through integrated, user-friendly VADS products
  - Provides support that spans the lifecycle through integrations of COTS tools, including Cadre Teamwork, IDE STP, Frame, and Interleaf
  - Provides support for user to integrate any desired tool into environment, but requires significant effort
- VADSpro
  - VADS APSE, minus support for new tool integration

CRI Life*CYCLE

- **CRI**

CRI is a Danish company, founded in 1979. It is owned 50% by IBM Denmark and 50% by four founding members. CRI has always been profitable, experiencing a doubling of revenue and profits each year since its inception. The Life*CYCLE and other CASE products are their second largest revenue producers, with secure military networks being their first. Most of their customers come from the aerospace industry, and include Boeing and the European Space Agency. In 1992, they opened an office in Seattle, Washington.

- **Life*CYCLE**

CRI's goal for its Life*CYCLE product is to provide an integrated environment with tools to support activities in all phases of the software development lifecycle. The strategy for attaining this goal is to integrate leading edge, third party tools where possible, and to develop tools for the software development activities that aren't covered. Just as for VADS APSE, the versioning and CM functions, repository services, access control and process tailoring support, etc. are provided by the Software BackPlane.

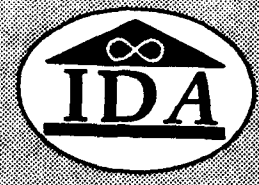
Third party tools that have been integrated to-date include: RDD-100, HOOD-SF, ASA (Automated Structured Analysis), Interleaf, FrameMaker, VADS APSE, Rational Ada, IBM Ada & C, Sun C, Logiscope, and several relational database management systems. They expect to have an integration of Cadre Teamwork soon.

The tools and capabilities that CRI has developed include Life*SUPPORT, for document and interface changes and problem report handling; Life*LINK, which includes its own RDBMS, for traceability handling; and Life*FLOW, a graphical process enactment tool.

CRI has also developed a generic 2167A process, that can be easily customized. It contains support for all DIDs.

Unfortunately, CRI personnel were not able to give a good demonstration of the Life*CYCLE product at IDA. Whether that was because of simple unpreparedness or because of "rough edges" in the product itself is not possible to tell.

CRI Life*CYCLE



- CRI
  - Aerospace industry customer base, includes Boeing, European Space Agency
  - Revenue and profits have doubled every year since formation in 1979
- Life*CYCLE
  - Provides CRI tools that "fill the gap" in supporting activities that span the software development lifecycle
  - Provides integrations of large number of existing COTS tools (several of European origin) including RDD-100, HOOD-SF, ASA, Interleaf, Frame-Maker, VADS APSE, Logiscope, and several compilers and DBMS
  - Provides support for user to integrate any desired tool into environment, but requires significant effort
  - Poor demonstration given at IDA

Introduction: Messaging-Based SEEs

The following slides discuss messaging-based SEE frameworks, as exemplified by HP SoftBench and SunSoft ToolTalk. Much of this work is based on the Field environment, a research SEE at Brown University [Reiss 1990].

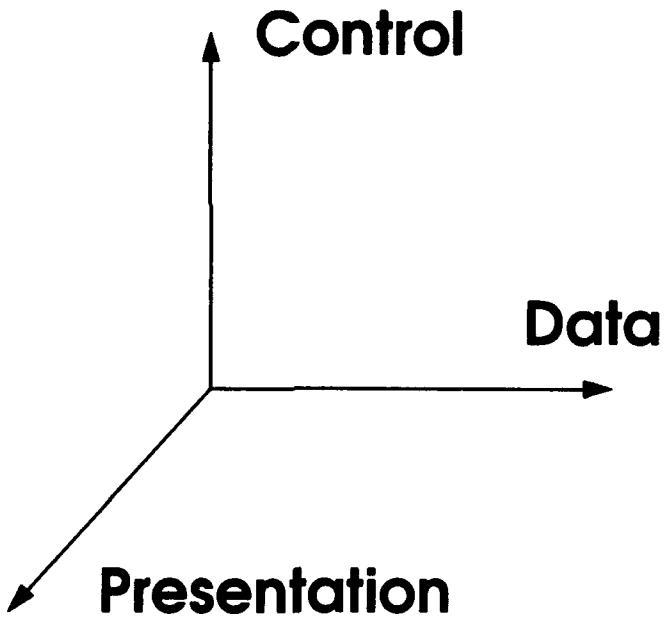
- **An approach to control integration**

Integration between tools has often been viewed in multiple dimensions. These dimensions usually include data integration (in which tools share data), presentation integration (in which tools share a similar user interface), and control integration (in which tools are able to communicate and/or control each other).

Messaging-based SEEs are a form of *control* integration. As such they stress communication between tools. This is not in conflict with presentation or data level integration, so a messaging-based system could be combined with a PCTE or ATIS based SEE.

Introduction: Messaging-Based SEEs

- An approach to control integration



Dimensions of Integration

Introduction: Messaging-Based SEEs

- **Tools send messages to other tools through an intermediary.**

The intermediary determines which tools receive the messages by doing pattern-matching. The intermediary may start up a tool that is not currently running to handle a message, as well as sending messages between actively running tools.

- **A tool can:**
  - Register for the kinds of messages it wishes to receive. Usually a tool will register for a number of different messages.
  - Send a request for an operation to be performed. If there is no tool registered to handle that request, an error is returned. Note that the tool to handle the message might not be currently running, in which case the messaging system may cause the tool to begin running first.
  - Send a notification that an event has occurred. Notifications are of events which other tools may or may not be interested in, and no error is returned if there is no tool registered to receive (handle) the notification.
  - Receive a request or notification. Note that there may be many tools registered to receive a notification, but no more than one tool can register to receive a specific request (this is to keep tools from interfering with each other). In the case where there are multiple different tools of the same type (for example, multiple text editors), the system may give the user the choice of which tool to register to receive a request.

- **Key: standardize messages.**

This approach requires that there be a standard for the messages. This standard is essentially a communication standard, and is analogous to a data schema in data integration. Without such a standard, other tools will not know what messages to send or receive.

Introduction: Messaging-Based SEEs

- Tools send messages to other tools through an intermediary.
- A tool can:
  - *Register* for the kinds of messages its wants to receive.
  - Send a *request* message to invoke an operation.
  - Send a *notification* message that an event has occurred.
  - *Receive* a message.
- Key: standardize messages.

Messaging-based COTS Products

There are two key products for messaging-based COTS SEEs:

- **HP SoftBench**

HP SoftBench was the first commercial messaging-based SEE. HP SoftBench consists of a set of tools (sufficient for editing C programs) and a framework. The key part of the current framework is HP's Broadcast Message Server (BMS), which is the message intermediary.

- **SunSoft ToolTalk**

SunSoft's ToolTalk is a product very similar to HP's BMS. It provides the intermediary through which various tools send and receive messages.

Messaging-Based COTS Products

- HP SoftBench
- SunSoft ToolTalk

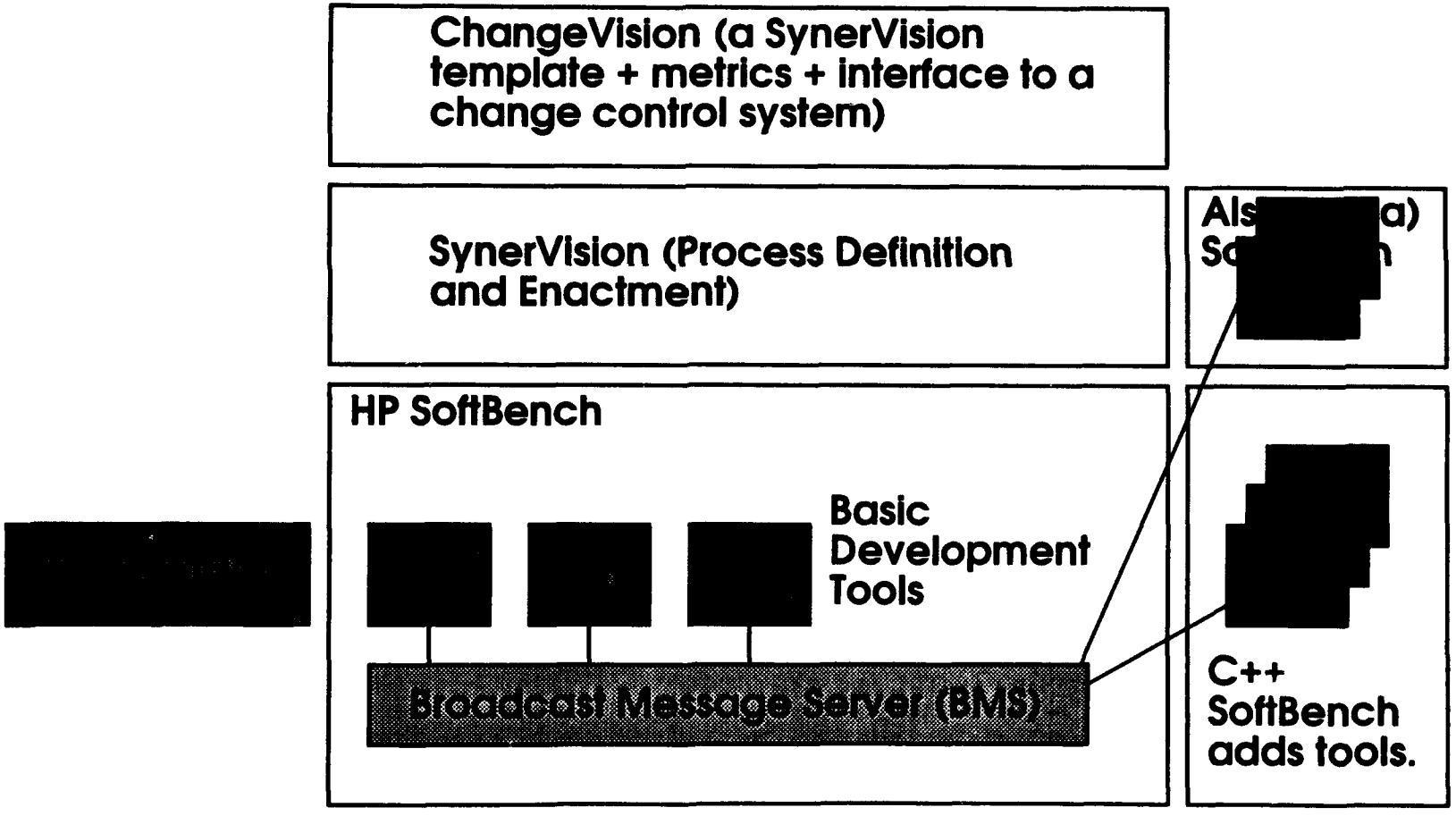
HP SoftBench

HP SoftBench can be divided into two parts: a toolset, and a framework. The toolset that comes with HP SoftBench is intended to be a basic set for developing C, Fortran, and Pascal programs, and includes:

- Tool Manager: invokes, terminates, and lists running tools.
- Development Manager: lists files in current context and invokes tools for highlighted files. This is a textual interface; HP has also integrated its VUE package on the HP platform which on those platforms allows visual selection.
- Program Editor: a mouse-oriented, language-sensitive editor. Also provided are interfaces to allow use of emacs or vi.
- Program Builder: Builds make files, runs make, and has a window to help in fixing errors. This product can display a dependency graph, and a double-click on the graphical object moves the user to that location in the makefile.
- Version Management Interface: An interface for SCCS and RCS commands from a menu.
- *Static Analyzer:* Performs cross-reference queries.
- Monitor: Displays BMS Messages (for debugging environment problems).
- Program Debugger Interface: An interface to a debugger. This is dbx on Sun workstations and xdb (an X windows debugger) for others.
- File Comparison Tool: Shows the changes to a file.
- Mail Encapsulation: an encapsulation of the X-windows mailer mailx.
- Incremental Linker (on HP machines only).

The C++ SoftBench product contains all of the above and adds some C++ specific tools, in particular C++ support to the editor, debugger, and static analyzer and a graphical inheritance tree editor. Alsys makes a product called Ada SoftBench, which is Alsys' Ada compilation system integrated with SoftBench. The Encapsulator (which allows users to integrate their own tools) is an optional tool. SynerVision is essentially a process definition and enactment tool, while ChangeVision is a pre-developed template for SynerVision that provides an interface to a commercial change request system (QualTrak DDT), as well as some metric & reporting tools.

HP SoftBench



Sun ToolTalk

- **Newer; similar capabilities to HP BMS**

Sun has developed a product called ToolTalk, which is also an interapplication communication product similar to HP's Broadcast Message Server (BMS), ToolTalk provides the capability to register for messages, send a request, send a notification, and receive a request or notification.

- **Included with Sun's Solaris Operating System**

ToolTalk is bundled with Sun's Solaris 2.x operating system for SPARC machines. This is significant, because this makes the ToolTalk program widely available at no additional charge to many users, providing vendors an incentive to take advantage of it. ToolTalk is also available on other Unix workstations besides Sun Microsystems Computer Corp., including Cray Research, Inc., Digital Equipment Corp., Hewlett-Packard, IBM Corp., Intergraph, Silicon Graphics, Inc., and SPARC-compatible vendors.

- **Some tools integrated with ToolTalk**

Sun's DeskSet applications use ToolTalk. These include the Text Editor and Mail Tool, as well various utility programs (Audio Tool, *Binder, Color Changer, Icon Editor, PageView).* Sun stated that the newest version of Sun Ada uses ToolTalk as well.

A number of activities have increased the visibility and importance of ToolTalk, as discussed in the slide "Messaging-based SEEs: Recent Activities."

Sun ToolTalk

- Newer; similar capabilities to HP BMS
- Included with Sun's Solaris Operating System
- Some tools integrated with ToolTalk

References

[Azzara 1993] Azzara, Mike. March 29, 1993. *Cheers for Admitting This Mistake.* Open Systems Today. Manhasset, NY: CMP Publications, Inc.

[Brown 1992a] Brown, Alan W., and Maria H. Penedo. July 1992. *An Annotated Bibliography on Integration in Software Engineering Environments.* ACM Software Engineering Notes. ACM SIGSOFT.

[Brown 1992b] Brown, Alan W., Anthony N. Earl, and John A. McDermid. 1992. *SEEs: Automated Support for Software Engineering.* McGraw-Hill Interantional Series in Software Engineering. McGraw-Hill.

[CC 1993] CASE Communique and CASE Interoperability Alliance (CIA). *Proposed Messaging Architecture: A Joint Proposal.*

[Chen 1992] Chen, Minder, and Ronald J. Norman. March 1992. *A Framework for Integrated CASE.* NY, NY: IEEE Software.

[Digital 1992] Digital Equipment Corporation (DEC), Silicon Graphics Inc. (SGI), and SunSoft. October 1, 1992. *The CASE Interoperability Message Sets: Release 1.0.* Requests to be sent to casemsg.ext@sun.com.

[Fife 1992] Fife, Dennis and David A. Wheeler. January 1992. *Technical Risk Analysis of the GPALS Software Engineering Support Environment.* IDA document D-1118. Alexandria, VA: Institute for Defense Analyses.

[Horwitt 1993] Horwitt, Elisabeth. March 29, 1993. *COSE aims at net systems management.* ComputerWorld. Framingham, MA: CW Publishing Inc.

[Johnson 1993] Johnson, Maryfran. March 29, 1993. *Past failures temper hopes for Unix unity: Vendors urged to abandon politics, differing agendas.* ComputerWorld. Framingham, MA: CW Publishing Inc.

[Krill 1993] Krill, Paul. March 29, 1993. *ISV Reaction: 'Once Bitten, Twice Shy': High hopes for Coalition are Tempered by Track Record of Past Unix Alliances.* Open Systems Today. Manhasset, NY: CMP Publications, Inc.

[NIST 1991] National Institute for Science and Technology (NIST). December 1991. *Reference Model for Frameworks of Software Engineering Environments*. (Technical Report ECMA TR/55, 2nd edition). NIST Special Publication 500-201. Gaithersburg, MD: NIST.

[Penedo 1988] Penedo, Maria Heloisa (Lolo) and William E. Riddle. June 1988. *Guest Editor's Introduction: Software Engineering Environment Architectures*. IEEE Transactions on Software Engineering, Vol. 14, No. 6. NY, NY: IEEE. pp. 689-695.

[Reiss 1990] Reiss, Steven P. July 1990. *Connecting Tools Using Message Passing in the Field Environment*. NY, NY: IEEE Software. pp 57-66.

[Wagner 1993] Wagner, Mitch. March 29, 1993. *A Move Toward Unity: IBM, HP, Sun and others form alliance against Windows NT*. Open Systems Today. Manhasset, NY: CMP Publications, Inc.