Technical Report
CMU/SEI-93-TR-1
ESC-TR-93-175

An Overview of PCTE: A Basis for a Portable Common Tool Environment

Fred Long
Ed Morris

March 1993

Technical Report
CMU/SEI-93-TR-1
ESC-TR-93-175
March 1993

An Overview of PCTE: A Basis for a Portable Common Tool Environment

Fred Long
Ed Morris

CASE Technology

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

This report was prepared for the

SEI Joint Program Office
HQ ESC/AXS
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER

(signature on file)

Thomas R. Miller, Lt Col, USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number F19628-95-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 52.227-7013.

This document is available through Research Access, Inc., 800 Vinial Street, Suite C201, Pittsburgh, PA 15212. Phone: 1-800-685-6510. FAX: (412) 321-2994. RAI also maintains a World Wide Web home page. The URL is http://www.rai.com

Copies of this document are available through the National Technical Information Service (NTIS). For information on ordering, please contact NTIS directly: National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161. Phone: (703) 487-4600.

This document is also available through the Defense Technical Information Center (DTIC). DTIC provides access

Table of Contents

List of Figures

An Overview of PCTE: A Basis for a Portable Common Tool Environment

Abstract: Environment framework technologies are becoming increasingly popular as aids to the construction of software engineering environments populated with integrated CASE tools. PCTE, a framework technology, is generating significant interest among environment users and builders as a mechanism for data management. This report details the history and current status of PCTE and PCTE-based environments. The major capabilities of PCTE are identified, and the techniques for integration of tools into PCTE are discussed.

1 Introduction

There is a long history in the U.S. of academic, commercial and governmental efforts to develop environment frameworks and Software Engineering Environments (SEEs). Many of these efforts, including frameworks like the Common APSE Interface Set (CAIS-A) and environments like the Ada Language System (ALS) and Ada Integrated Environment (AIE), have contributed to our understanding of the challenges involved in building SEEs. Ongoing government efforts such as Software Technology for Adaptable Reliable Systems (STARS), Process Software Life Cycle Support Environment (Pro/SLCSE) and the Integrated Computer Aided Software Engineering (I-CASE) procurement, as well as commercial efforts such as AD/Cycle (IBM) and Cohesion (Digital Equipment), are intended to construct viable SEEs for the government sector and commercial sectors, respectively.

Another set of efforts in Europe have also been addressing the problems of constructing SEEs. While domestic and European efforts have been primarily independent, there has been extensive cross-fertilization of ideas. Recently, there has been increasing interest in the U.S. concerning the progress of the Portable Common Tool Environment (PCTE) effort initiated in Europe. The PCTE effort has strong support from both European government and industry, as well as from the European representatives of leading U.S. computer manufacturers.

A growing number of major U.S. computer manufacturers are committing to providing PCTE support on their platforms. Digital Equipment and ISSI have announced that they are jointly developing a PCTE implementation. Emeraude (France) and Heuristix (India) have made similar announcements. IBM has announced that it will expand the IBM AIX Software Development Environment WorkBench/6000 integration services to provide data integration supporting the PCTE standard. In addition, a large numbers of tool vendors have discussed porting their software to a PCTE platform. Many vendors and individuals expressed interest and support for PCTE at the 5th International Workshop on Computer Aided Software Engineering (CASE 92).

The growing interest in PCTE was also highlighted at a recent forum discussing plans for the development of a North American PCTE implementation. This forum (NAPI 1992) attracted

approximately 100 participants, including both proponents and opponents of PCTE standardization.

In light of the growing interest, the Software Engineering Institute (SEI) has identified a need to educate the SEE community concerning the current state of the various PCTE implementations and versions. This report attempts to assist the education process by dispelling the myths and countering the misinformation (both positive and negative) that is appearing concerning PCTE.

The report does not provide an in-depth technical description of any PCTE version or implementation, and therefore is not intended for organizations which are intending to implement PCTE based tools or environments. This report specifically does not endorse PCTE or any other technology. Rather, the report is intended to inform managers and engineers who are currently planning and making decisions concerning CASE technology and SEEs.

Some confusion has developed over the varying characteristics and capabilities of systems claiming to be PCTE. Actually, as evident from the discussion of the history of PCTE in Section 2, the acronym PCTE refers to a number of divergent systems. To avoid confusion, this report will adhere to the convention that a PCTE version is explicitly named when reference is made to an individual system. When reference is made to the core capabilities available in all of the systems, the general term “PCTE” will be used

In keeping with these conventions, PCTE 1.5 refers to the set of PCTE specifications published in 1988. PCTE+ refers to an upward compatible set of specifications published by the Independent European Programme Group (IEPG) of NATO. ECMA PCTE refers to a set of specifications published by the European Computer Manufacturers Association. This last set of specifications is being promoted as a potential ISO standard.

Section 2 of this report provides background information on PCTE, including its history, a comparison with other framework technologies, and a summary of recent developments. Section 3 provides a primer on PCTE technology. Section 4 discusses the integration support provided by PCTE. Section 5 identifies ways in which tools can be introduced into the PCTE environment. Section 6 details the current and future implementations of PCTE and SEEs built on top of it. Section 7 discusses major issues concerning PCTE. Finally, Section 8 concludes with a recommended approach for organizations wishing to incorporate PCTE or other environment frameworks. This section is based on earlier SEI work in identifying lessons learned from previous environment efforts. An appendix listing sources of additional information about PCTE is provided at the end of the text.

2 Background

The need for a SEE has long been recognized [STONEMAN 80]. Subsequent experience in attempting to build SEEs has led to the recognition that they are difficult to specify and expensive to build. This has led an increasing number of organizations to investigate ways in which the burden of building a SEE could be shared across multiple organizations. A particularly attractive way of sharing the burden is to allow different organizations to develop parts of the total environment solution. To allow this to occur, however, a set of common agreements must be made defining the manner in which tools will operate, interact and share data. The mechanisms that support tool operation, interaction and data sharing are called framework and platform services. Brown, Earl, and McDermid also identify a set of target services supporting situations where the target platform differs from the host [Brown 92]. Target services are not discussed further in this report.

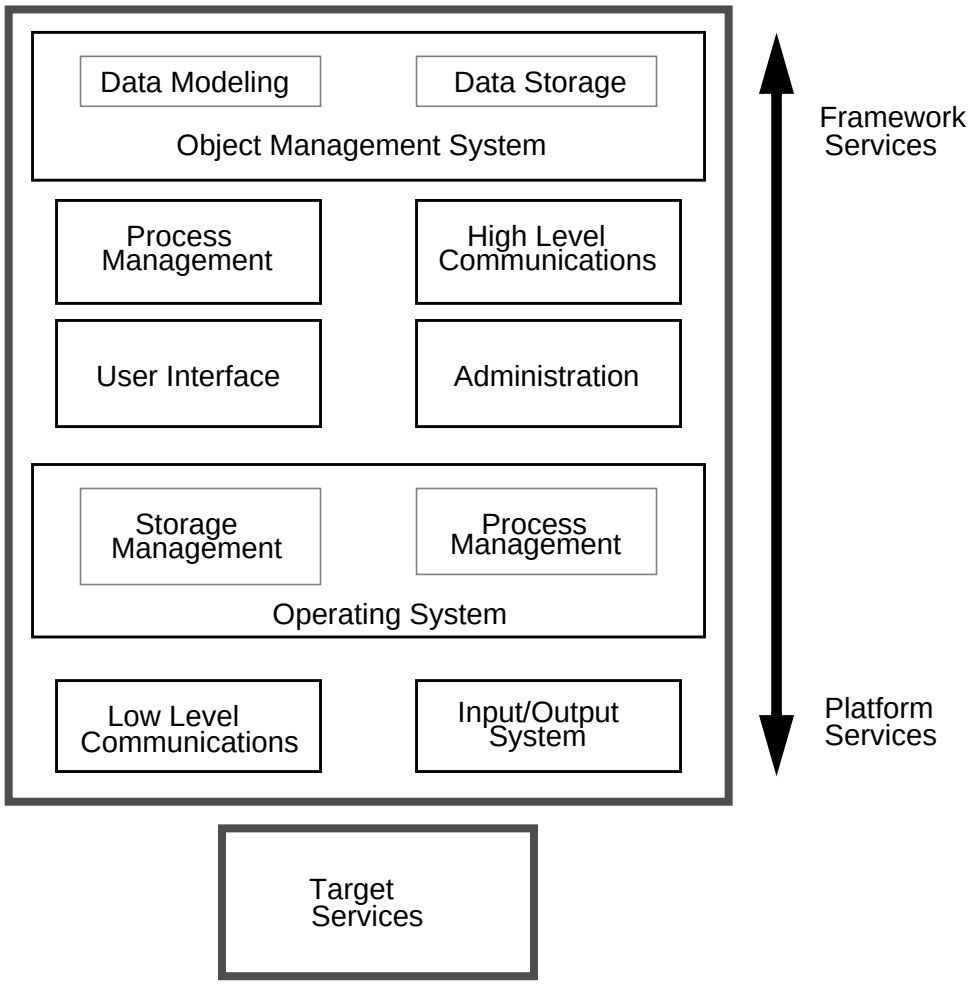

Figure 2-1 SEE Infrastructure Services[1]

1. Adapted from Brown, Earl, and McDermid [Brown 92].

Platform services provide a virtual machine layer and are considered to include basic operation system functions. Framework services are specifically targeted toward the development of SEEs. However, the distinction between platform and framework services is blurred. As a specific framework service becomes universally accepted, it may become included within platform services.

The total set of framework, platform, and target services provided to a tool represents a public tool interface and can be called the SEE Infrastructure Services. PCTE is vying to provide a number of these infrastructure services, particularly in the framework areas of data modeling and data storage (Figure 2-1), but also in the areas of process management, communications, and administration services.

2.1 The PCTE Standards

In October 1983, the European Strategic Programme for Research and Development in Information Technology (ESPRIT) started a project to produce “A Basis for a Portable Common Tool Environment”. This project and its result became known by the acronym PCTE. The original partners were Cie. Bull SA (France), GEC Research Laboratories (UK), International Computers Limited (UK), Nixdorf Computer AG (W. Germany), Olivetti SpA (Italy), and Siemens AG (W. Germany).

The aim of the PCTE project was to produce a public tool interface specification, known officially as the European Standard Tool Interface, which would enable the results of other ESPRIT projects to be shared more easily. An early decision was made to make the PCTE interface specification upward compatible with Unix System V to ease tool migration. The project was completed successfully, and the C language PCTE interface specifications were placed in the public domain in September 1986. *Note that what is commonly called “PCTE” is an interface specification for a set of program callable interfaces, not a particular implementation of the specification, nor a compete environment populated with tools*. The specification is language and operating system independent, although Ada and C bindings have been developed, and C++ bindings are in process.

The two-volume interface specifications published in 1986 were known as PCTE 1.4. Volume 1 contained the basic mechanisms: execution, communication (file input/output facilities), inter-process communication, object management system, activities (concurrency control), distribution, and emulation of Unix system calls. Volume 2 contained the user interface specification. The user interface specified was unique and not in any way related to the X Window System. Ada language specifications corresponding to Volume 1 were published in May 1987 (with the C language specifications taking priority in case of conflict).

As work continued within the PCTE project to improve the specifications, the PCTE Interface Management Board (PIMB) was set up to control further evolution of the PCTE interfaces. PIMB formed the PCTE Interface Control Group (PICG) to manage change requests. In 1988, the next revision of the PCTE interface specification, known as PCTE 1.5, was published. The

changes in Volume 1 were minor and were mainly to correct errors. The changes in Volume 2 were much more substantial and took into account the growing importance of the X Window System.

PICG later handed over the task of making PCTE into an international standard to the European Computer Manufacturers Association (ECMA). Under ECMA Technical Committee 33 (TC33), two task groups were established: the Task Group on ECMA PCTE (TGEP) to define ECMA PCTE, and the Task Group on the Reference Model (TGRM) to develop a reference model to assist the standardization process.

In the meanwhile, it was recognized that there were deficiencies in PCTE if it were to be used in the development of defense systems. The two main problem areas identified were security and Unix operating system dependence (because of the “Unix” flavor of the interfaces). A project was set up by Technical Area 13 (TA13) of the Independent European Programme Group (IEPG) of NATO to develop a PCTE-based public tool interface that would be suitable for both civil and military use. This project was known as PCTE+.

The PCTE+ project started by producing a requirements document, called the EURAC [EURAC 88], which was based on the Requirements and Design Criteria for the Common APSE Interface Set (RAC) [RAC 86]. It then went on to develop the PCTE+ interface specifications themselves. Both C and Ada language specifications were developed simultaneously and the final version of the PCTE+ interface specifications, PCTE+ Issue 3 [PCTE+ 88], was published in October 1988. The PCTE+ project is providing two implementations (France and the UK) and evaluation of the PCTE+ specification. The PCTE+ specification is closely related to its PCTE 1.5 counterpart. Discussion of the differences between PCTE 1.5 and PCTE+ is available elsewhere ([Boudieretal 88], [Tedd 89]).

The developments of PCTE by ECMA and IEPG gave rise to the possibility of two different PCTE-based standards: PCTE+ and ECMA PCTE. However, TGEP decided to base their work on PCTE+ Issue 3, with minimal divergence [Davis 90]. See Dawes and Davis for a description of the ECMA PCTE standards and their relation to PCTE+ [Dawes 91].

The ECMA PCTE specifications were produced in the form of an abstract specification, ECMA-149 [ECMA149 90], which was approved by the ECMA General Assembly in December 1990; and language bindings for the C language, ECMA-158, and Ada, ECMA-162, ([ECMA158 91] and [ECMA162 91], respectively). A C++ language binding is in progress.

The TGRM produced the ECMA reference model for describing and comparing the capabilities of frameworks. This reference model is sometimes known as the “Toaster” model due to a toaster-like graphic that appears in the original reference model report. The model identifies a set of framework services. These services can potentially be provided via a variety of framework architectures. In a completed SEE framework, not all services must be provided by a single framework product. The ECMA reference model has been further developed in conjunction with NIST and is now known as the NIST/ECMA Reference Model [NIST_ECMA 91].

2.2 CAIS and PCTE

While the PCTE standards were being developed in Europe, work was under way in the U.S. to develop the Common APSE Interface Set (CAIS). It had become evident that the tool interfaces provided by the Ada Language System (ALS) and Ada Integrated Environment (AIE), developed under U.S. Army and U.S. Air Force contract, respectively, had diverged. Tools written for one environment could not be used in the other. The motivation behind the development of the CAIS was to provide a standard interface for tools across the ALS and AIE environments. This effort was initially led by the Kernel Ada Program Support Environment (KAPSE) Interface Team (KIT), and was soon augmented by the KAPSE Interface Team from Industry and Academia to become KIT/KITIA.

The initial objective of KIT/KITIA was to define standard APSE interfaces for “transportability” and “interoperability” between tools built for the ALS and AIE. This resulted in an interim standard, DOD-STD-1838, which was published on October 1986. At the same time, a set of requirements and criteria for the CAIS were drawn up, known as the RAC [RAC 86]. The CAIS work continued, and SofTech was contracted to define the revised CAIS-A (MIL-STD-1838A). This was published in April 1989, and two implementations of CAIS-A have since been built.

The initial motivations behind the PCTE and CAIS projects were similar, differing primarily in the focus on Ada and defense needs by CAIS. Not surprisingly, they produced very similar interfaces. Indeed, both projects benefited from cross-fertilization from the other, resulting from the American awareness of the original PCTE project and the European involvement in KIT/KITIA. In a report published in January 1986, the Ada-Europe Environment Working Group found that PCTE 1.4 specifications conformed very closely to the RAC [Conform 86]. Only two main areas were identified where PCTE 1.4 did not meet the requirements of the RAC, namely security and the need for Ada specifications. (This latter need was subsequently fulfilled.) As the PCTE specifications have evolved they have grown even closer to fully meeting all the requirements of the RAC. ECMA PCTE and CAIS-A provide very similar functionality in broadly similar ways. However, CAIS-A does not identify a C binding, and does not meet the strong upward migration (from Unix) constraint for PCTE.

2.3 Other Framework Technologies

There are a number of other software engineering environment framework technologies that can be compared and contrasted with ECMA PCTE. CAIS-A, with many similarities to ECMA PCTE has already been mentioned. Others are discussed below. However, as indicated earlier, we take no position on the relative value and ultimate acceptance of any of these standards (or the many others that have been proposed).

2.3.1 OMG

It is felt by some that object-oriented technology represents a superior mechanism (when compared to ER technology) for integrating tools. A number of organizations are working to

develop object-oriented databases, file systems, and operating systems which presumably could form the framework for an object-oriented software development environment.

The Object Management Group (OMG) is an international consortium of over 100 tool vendors and users dedicated to endorsing specific industry technologies for object-oriented software development in a distributed computing environment. Major participants in OMG include Sun, Hewlett Packard (HP), Digital Equipment (DEC), and NCR. Some members of this group are also strongly committed to ECMA PCTE.

OMG has endorsed the Common Object Request Broker Architecture (CORBA), a mechanism by which an object can make requests of other objects in the environment. CORBA defines a central object manager that handles communication and allows objects to interact transparently over a network using a remote procedure call (RPC) mechanism.

2.3.2 ATIS

A Tool Integration Standard (ATIS) is the result of work originally carried out by Atherton Technology and DEC. ATIS represents a public tool interface specification and it provides data integration and facilities for encoding process. The data integration facilities are based on an object-oriented data model and allow methods to be associated with objects.

ATIS was further developed by the CASE Integration Services (CIS) group. This group was composed of end users, hardware vendors, CASE tool vendors, and framework and environment vendors. CIS has been transformed into the ANSI technical committee X3H6 which, among many other issues, is discussing extending ECMA PCTE with object-oriented capabilities. ATIS is also being discussed by the ANSI technical committee X3H4 as a basis for the second generation of IRDS.

ATIS is considered by some to be the major competitor to ECMA PCTE as a framework service. Some observers believe that the object-oriented approach represented by ATIS is a superior technology, making ECMA PCTE technology obsolete even before it is adopted. Among the advantages claimed by advocates of ATIS are better support for small objects (objects just a few bytes in size) and more powerful process integration facilities. For a more detailed comparison of ATIS and PCTE see Black's article "ATIS, CIS, PCTE, and Software Back Plane" [Black 91].

Other observers suggest that ATIS and PCTE provide complementary facilities which may be advantageous to combine. DEC has distributed information concerning the coupling of ECMA PCTE and CORBA technologies to produce an ATIS implementation [Argento 92]. DEC believes this approach can provide a robust and flexible framework for CASE tool integration.

2.3.3 AD/Cycle

AD/Cycle is a set of data integration services and support tools developed and supplied by IBM [Mercurio 90]. AD/Cycle support services provide a framework for the construction of

tools. User interface, workstation, work management, tool, repository, and library services are supported.

The AD/Cycle software architecture is designed primarily to operate on, and support development for, large computer systems. PCTE is designed to operate on a distributed network of workstations. However, the AD/Cycle repository services provide facilities very similar to those of the PCTE object management services. In any case, the initial reception to the portions of the AD/Cycle architecture that have been released has been mixed. Criticisms include extreme complexity, incomplete support, and high expense. However, the AD/Cycle architecture has the potential advantage (and associated disadvantages) of strong, centralized data resource management.

2.3.4 OOTIS

IBM's Object-Oriented Tool Integration Services (OOTIS) is an extension to ECMA PCTE with the goal of providing object-oriented operation dispatch, database access, and tool composition on top of basic ECMA PCTE capabilities. The OOTIS Data Definition Language and metaschema extend the ECMA PCTE Data Definition Language and metaschema. OOTIS provides a uniform method of access to all objects, whether stored as an ECMA PCTE object, or in an object-oriented database.

IBM has submitted OOTIS to OMG and X3H6 for consideration as a mechanism for the integration of data and control capabilities.

2.3.5 SoftBench Broadcast Message Server

A rather different kind of framework service is offered by HP's SoftBench Broadcast Message Server (BMS). This provides message-passing facilities as a tool integration mechanism (and is rather different from the message queue facilities provided by PCTE). In BMS, tools broadcast, and receive, messages. A tool can specify which messages or classes of message it is interested in receiving, but a tool which broadcasts a message need not be aware of which tools will receive the message. There are facilities to "encapsulate" existing tools which were not written to participate in message traffic in a message handling wrapper. The BMS facilities are complementary to the facilities of PCTE, and it has been proposed that a merger of PCTE and BMS provides a very good tool integration framework [Oliver 91].

BMS is based on ideas originally developed in the FIELD system developed at Brown University [Reiss 90]. Other vendors have also incorporated message-passing into their environments, including DEC (FUSE), IBM (WorkBench 6000, based on BMS) and Sun (Tooltalk).

CASE Communiqué, an industry forum for control integration, set up by HP, IBM, Informix Software, and CDC, is working on standards for message formats applicable to systems like BMS. They are focusing on the semantics of the messages such that they will be mappable to any message syntax. ANSI X3H6 is also working in this area.

2.3.6 PCIS

Because of the similarities between PCTE and CAIS, consideration was given to producing a merger of the two standards. This effort became known as the PCIS Programme with PCIS (Portable Common Interface Set) being an amalgamation of the acronyms PCTE and CAIS. Two meetings were held, sponsored by the Ada Joint Program Office and IEPG TA13 to discuss convergence of the two standards. The first was held in Waltham, USA on January 25 through February 5, 1988, and the second in Winnersh, UK on April 14-28, 1989. Both meetings produced reports which discussed the differences between PCTE and CAIS and how they could be resolved ([Waltham 88] , [Winnersh 89]). Subsequently, it was decided that PCIS should be more than a simple convergence of PCTE and CAIS, and that it should be developed as the next generation of public tool interface. Two workshops were held in 1991, the first at Heathrow, UK, from April 29 to May 3, and the second in Redondo Beach, USA, on June 3-7. An International Requirements and Design Criteria (IRAC) for the Portable Common Interface Set (PCIS) was produced in November 1991 [IRAC 91]. Recently, it was announced that the PCIS Programme will define and assess an environment framework. This environment framework is intended to provide a basic set of capabilities to environment builders. The work is to be completed by the end of 1993. The definition will be based on the IRAC and on the NIST/ECMA Reference Model, and ECMA PCTE will be used as the underlying tool interface. The French environment Enterprise II will be used as a baseline. A prototype of the framework will be built and an environment developed on top of this which can be demonstrated as an APSE.

2.4 Ongoing Developments

The current PCTE standards are those produced by ECMA. The technical group that produced them, TGEP, continues to work on and maintain these standards. Technical comments (only) may be sent to TGEP at tgep@win.icl.co.uk; to date, over 2,000 have been received. They are also considering known or suspected problems, strategic issues that remain unresolved in the present standard, and requested enhancements. The intention is to promote the ECMA standard to an ISO standard via a “fast-track” procedure. In addition to this formal standardization activity, implementations of ECMA PCTE are being developed (see Section 6). Implementations of PCTE 1.5 and PCTE+ continue to be enhanced and evaluated but the specifications themselves are now frozen. In light of the development of ECMA PCTE, IEPG TA13 has declared its intention of letting the PCTE+ specifications die. Similarly, PCTE 1.5 implementations are being maintained but represent an evolutionary dead end in the PCTE family.

If ECMA PCTE is to be widely used, not only must the interfaces be established as standards but the data schemas used by ECMA PCTE tools must also be standardized (see Section 5.3). ECMA TC33 has set up an ad hoc committee to consider the problem of common data schemas. At present, this committee is collecting known PCTE schemas. More generally, the ISO/IEC JTC1/SC7 WG11 is considering the Description of Data for Software Engineering standard (DDSE) as a basis for a higher level description of ECMA PCTE common schemas.

The CASE Data Interchange Format (CDIF) standard is being considered by ECMA TC33 for data interchange between different ECMA PCTE-based environments and between ECMA PCTE-based and environments utilizing other framework capabilities.

The PIMB continues with various activities to promote greater use of PCTE and to encourage development of tools and software engineering environments using PCTE. It formed the PCTE Promotions Group (PPG) to help in this work and publishes the PCTE Newsletter. In May 1992, the PIMB reformed itself into a new non-profit making international company called the PCTE Interface Management Board (PIMB) Association.

2.5 Summary

The term “PCTE” covers a number of different tool interface specifications. Their development started in 1983 and continues today. The PCTE interface specifications have not been developed in isolation, but have benefited from ideas introduced from other work. In turn, ideas from PCTE have been used elsewhere. Also, PCTE is related to other major developments in the area of tool support interfaces. The chronology of the major standards discussed in this section, and the interrelations between them, are shown in Figure 2-2.

.

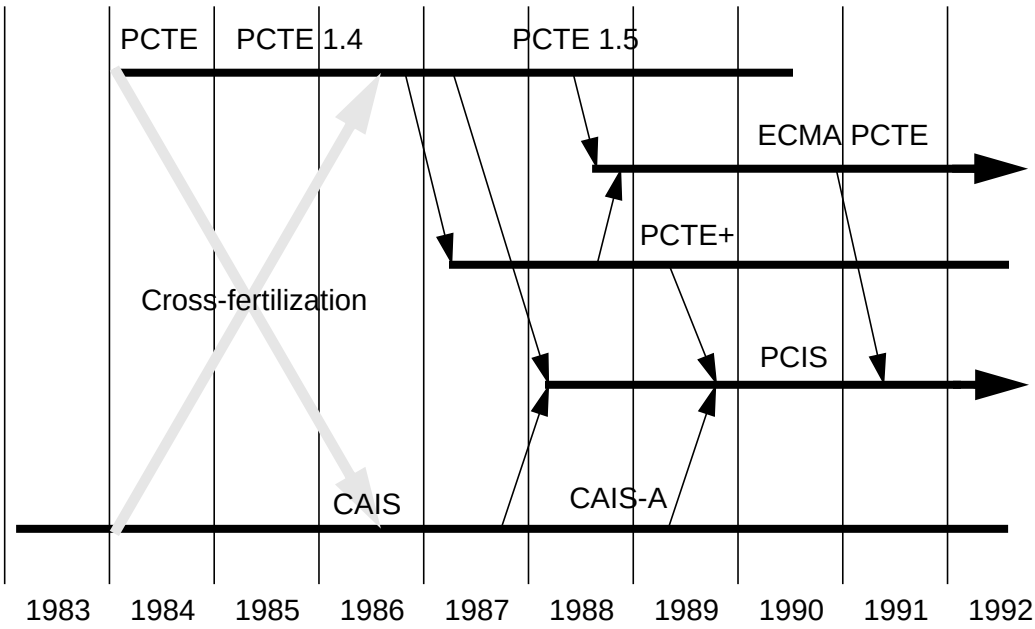

Figure 2-2 Interrelations Between the Major Standards

Of the family of PCTE specifications that have been developed, only the ECMA PCTE standard has long term viability. Both PCTE 1.5 and PCTE+, while contributing to the understanding of framework and environment issues, are evolutionary dead ends. The currently marketed PCTE 1.5 implementations present a somewhat different interface than ECMA PCTE, but are based on similar technology.

3 What is ECMA PCTE?

This section provides a high-level overview of ECMA PCTE. It concentrates on describing the object management system, processes and activities in some detail. Other facilities in ECMA PCTE are summarized. In the final section, the differences between ECMA PCTE and PCTE 1.5 are outlined to allow the reader to contrast the capabilities offered in currently available systems with those that will be offered in ECMA PCTE implementations.

ECMA PCTE is a tool interface specification (i.e., a set of specifications of facilities that can be used by software tools). These facilities go beyond those necessary for data management alone, and include capabilities for process control, security, network management, auditing, and accounting. ECMA PCTE attempts to provide an integrated (though not necessarily complete) set of environment services in these areas. It also provides an abstract specification of the facilities, using formalized English and some formal notation, and language bindings (to C and Ada, at present, with a C++ binding being developed).

3.1 The Object Management System

ECMA PCTE provides facilities for tools to store data, and to retrieve and manipulate that data. The ECMA PCTE data management system is called the *object management system* (OMS) and is of the entity-relationship-attribute type. The entities are called *objects*.

The basic data item in ECMA PCTE is the *object*. An object may, but need not, have *contents*. An object's contents can be thought of as corresponding to a file in a traditional operating system. ECMA PCTE provides facilities to read and write object contents. An object may have *attributes* attached to it. These are, typically, small pieces of information associated with the object, such as owner, creation date, etc. There are facilities to set, read and modify attributes on objects.

ECMA PCTE's OMS also allows *links* from one object to another. Thus, an arbitrary network of linked objects may be built up. Links are typed, and may also carry attributes. Facilities are provided to create and delete links and to traverse links. This latter facility allows users and tools to navigate within the network of linked objects and, hence, to access objects in the OMS.

ECMA PCTE distinguishes between *primitive objects* and *composite objects*. A primitive object has no components whereas a composite object is a collection of primitive objects and other composite objects. There are operations that can be applied to a composite object as a whole. Also, *version control* operations, for both primitive and composite objects, are built into the system.

The ECMA PCTE OMS is typed; that is, all the objects, links and attributes have a type associated with them. The type model can be used to insure the integrity of certain data in the database. For example, the typing structure can prohibit a document object from being written to by a object representing test data.

Types are defined and managed in *schema definition sets* (SDSs) (which are themselves stored in the OMS) and the type model is represented by objects and links in the OMS. An object type may inherit properties from several other object types. Inheritance is not a characteristic of link and attribute types. An SDS will usually contain a collection of related type definitions associated with some tool or use of the OMS. ECMA PCTE provides facilities for adding, removing and modifying SDSs.

An running program in ECMA PCTE always has a *working schema* associated with it. A working schema is a composition of SDSs. A process can access only those objects, links and attributes whose types are defined in SDSs included in its working schema. This working schema mechanism controls the process' visibility into the database at a given time, thereby aiding security and access control. However, an executing tool may reconfigure the working schema by adding or removing SDSs.

ECMA PCTE was designed to be implemented on a heterogeneous network of workstations. The placement of objects across the network is usually transparent to the user. Facilities are designed to allow an implementation to continue to run, even if part of the network becomes disconnected.

3.2 Processes and Concurrency Control

ECMA PCTE has incorporated process management along with object management capabilities (as opposed to adopting the POSIX process management approach) due to the operating characteristics of tools. Ideally, one would like to enclose a tool's operations on the object base such that changes to the object base are made in an "all or none" (atomic) fashion. This would facilitate the aborting of the tool's operations by leaving the database in a consistent state. However, many CASE tools are implemented as several operating system processes. In order to accommodate such tools, support for atomic changes to the object base should encompass several operating system processes. Unix and POSIX do not provide this capability. Therefore, in order to constrain the effect of CASE tools composed of several operating system processes and to the implement desired atomic data management operations, unique process management capabilities were required.

In ECMA PCTE terminology, a running program is called a *process*. There are facilities to start (synchronously or asynchronously), stop, suspend and resume processes and to query the status of a process. These facilities are similar to those commonly found in operating systems. Each process is associated with a static context, which corresponds to the executable code of a program in a traditional operating system, and is stored in the OMS. The static context also has associated information which is used to control where in a distributed network a process may be invoked. In addition to the object representing the executable code, the execution of a process is also represented by an object in the OMS, called a *process object*. Hence, the basic OMS navigation and query facilities can be used to find information about ECMA PCTE processes by applying them to the process objects. For example, the status of a process (whether it is running, suspended, etc.) is represented by an attribute on the process object.

Hence, this information can be found by using the OMS facility to query an object's attributes, rather than requiring extra process querying operations. Also, other objects can be linked to the process objects, thereby making the connections between a process and the data and devices it is accessing explicit in the OMS. For communication between processes there are facilities to manage *pipes* and *message queues* (similar to the facilities found in Unix).

Since an ECMA PCTE implementation will likely be running on a network of workstations, there could be several processes active simultaneously, all accessing the OMS. ECMA PCTE provides what it calls *activities* to control concurrent OMS access. An activity may involve a number of cooperating processes and all processes run within some activity. There are three categories of activity: *unprotected*, *protected*, and *transaction*. Processes that run in unprotected activities are not protected against simultaneous access (by other tools) to the object base data they are using. However, processes that run in protected or transaction activities may place *locks* on the objects and links that they are manipulating so as to prevent concurrent access to the same object or link by other (protected or transaction) activities. In a transaction activity the effect of operations performed by processes can be made atomic, that is, either all the OMS changes made by the sequence of processes happen or else none happen. Each activity is represented by an object in the OMS. Locks are represented by links between the object representing the activity and the locked object.

3.3 Other Facilities in ECMA PCTE

ECMA PCTE incorporates a sophisticated security model. *Discretionary* and *mandatory* access controls are provided. Discretionary access control facilities allow the owners of objects to grant or deny other users access rights to objects. This corresponds to the access permissions that are commonly found in operating systems. Mandatory access control facilities allow users and objects to be classified at security levels (e.g., confidential, secret, top secret), and access to be granted or denied according to the relative classifications. Mandatory confidentiality controls prevent the disclosure of information to unauthorized users, and mandatory integrity controls prevent unauthorized users from writing to an object. Accounting and auditing facilities are also provided.

ECMA PCTE supports a *notification mechanism* which can be used to inform a process when some change occurs to an object in the OMS. The process indicates that it wishes to receive a message when modifications or deletions are made.

Foreign tools that run on a system other than ECMA PCTE are not excluded from interacting with ECMA PCTE tools. ECMA PCTE is designed to run on a network alongside or on top of another operating system, or with a bare machine connected. ECMA PCTE tools may transfer data to or from tools running in such foreign environments.

As has already been stated, ECMA PCTE was designed to be implemented on a network of workstations. To allow ECMA PCTE to continue running when part of the network becomes disconnected, crucial data, including some SDSs, are replicated on all workstations. ECMA

PCTE provides facilities to manage this replication and to control distribution (connection and disconnection of workstations, placing of an object or process on a specific network node, etc.).

3.4 Differences Between ECMA PCTE and PCTE 1.5

Although ECMA PCTE is the accepted standard, the only PCTE implementation that has received widespread use (primarily in Europe) is Emeraude V12 (see Section 6). Emeraude V12 PCTE is now being marketed in the U.S., and an important question for potential users is whether PCTE 1.5 forms a logical stepping stone to ECMA PCTE.

While sharing a common PCTE ancestry and ER model, the PCTE 1.5 and ECMA PCTE specifications define very different environment frameworks. PCTE 1.5 is not directly upwardly compatible with ECMA PCTE. On the other hand, it is likely that many of the lessons learned from use of PCTE 1.5 also apply to ECMA PCTE. In order to provide preliminary data for comparison between ECMA PCTE and PCTE 1.5, this section describes a few of the differences.

As a result of the original requirements placed on the design, PCTE 1.5 interfaces are similar to Unix. In many cases, PCTE 1.5 services basically provide a pass-through to the Unix services on which are based. Therefore, migrating a Unix tool to PCTE 1.5 is not a difficult task. ECMA PCTE interfaces, however, are intended to be independent of the underlying operating system. Support for foreign tools (such as Unix tools) is provided, however. While groups of Unix tools can benefit without change from shallow integration with ECMA PCTE, the tools will need extensive reworking to fully use ECMA PCTE facilities.

In addition to different interfaces, ECMA PCTE provides enhanced functionality (vs. PCTE 1.5) in many areas, and decreased functionality in a few, notably the user interface. Some of the major differences between the services provided by PCTE 1.5 and ECMA PCTE include:

- ECMA PCTE provides an expanded range of link and attribute types over those offered by PCTE 1.5. This expanded range of link and attribute types allows for a more detailed set of characteristics and relationships between objects.
- In ECMA PCTE, the notion of a composite object is further refined, allowing for grouping of primitive and composite objects. PCTE 1.5 offers a limited set of operations that may be applied to a collection of objects.
- ECMA PCTE identifies a set of version control operations. The PCTE 1.5 specification does not identify version control operations. Emeraude's PCTE 1.5 implementation does provide such operations, however.
- In PCTE 1.5, links between objects are always paired such that the source object of one member of a link pair is the target object of the other member of the link pair. In ECMA PCTE, links may be paired or may stand alone.
- In ECMA PCTE, running processes, activities, and locks are represented as objects in the object base. This is not the case in PCTE 1.5.

- PCTE 1.5 provides signals as an inter-process communication mechanism, as well as pipes and message queues. ECMA PCTE provides only pipes and message queues.
- PCTE 1.5 access control mechanisms are essentially those offered in Unix and are similar to the discretionary access controls provided by ECMA PCTE. ECMA PCTE also provides mandatory access controls which allow users and objects to be classified at security levels.
- PCTE 1.5 contains a user interface based on the X Window System. This includes the usual facilities for manipulating windows, menus, icons, etc. ECMA PCTE does not include any definition of user interface facilities but ECMA TC33 has accepted the recommendation that the X Window System should be the portability platform for PCTE-based tools with respect to the user interface

One very significant difference between PCTE 1.5 and ECMA PCTE is the amount of interest that has been generated by the two specifications. While only a few implementations (all non-domestic) of PCTE 1.5 exist, a growing number of hardware vendors within the U.S. have committed to providing ECMA PCTE implementations for their platforms. Tool vendors are making similar commitments to port their tools to ECMA PCTE when it becomes available. A more complete discussion of the plans of various vendors can be found in Section 6.

4 What Can ECMA PCTE Provide?

The ECMA PCTE abstract specification defines a language independent interface providing support mechanisms for software engineering environments populated with CASE tools [ECMA149 90]. Neither the available PCTE 1.5 implementations nor the ECMA specification defines a set of tools or an environment populated by tools. As such, available PCTE 1.5 implementations and the ECMA PCTE specification should not be judged as environments, but instead on the ease with which they allow environment builders to write, assemble, and customize integrated software development environments.

One approach to evaluating the support provided by an environment framework is to map it against a reference model detailing core services provided by such frameworks. A reference model for environment frameworks has been developed and is supported by the National Institute of Standards and Technology along with ECMA [NIST_ECMA 91]. A mapping of ECMA PCTE against this reference model is provided by Brown, Earl, and McDermid [Brown 92].

A second approach to evaluating the capabilities of an environment framework is to compare the services offered by the framework to those required in order to integrate tools. Wasserman has identified five types of integration that are necessary in an environment: platform, which represents the set of services that provide network and operating system transparency; presentation, which involves the consistency of the user interface; data, which refers to the sharing and interoperability of data across tools; control, which refers to the communication, sequencing, and interoperation of tools; and process, which refers to the organization of tools and frameworks into a mechanism for supporting the software process [Wasserman 90].

Thomas and Nejmeh have further refined the properties that characterize the relationships between tools for four of Wasserman's five types of integration (Figure 4-1). Because Thomas and Nejmeh focus on the relationship between tools, they do not refine the properties of platform integration. However, they do suggest that a successful integration framework will likely contain mechanisms to enhance the degree of integration along the dimensions of Wasserman's model [Thomas 92]. It is unlikely that any single framework product will provide consistently strong support for all forms of integration. Thus, weak support along a particular dimension does not imply that a framework product is inappropriate, but rather that additional support (from enhancement or another framework product) is necessary. Sections 4.1 through 4.5 discuss the support for the types of integration provided by ECMA PCTE.

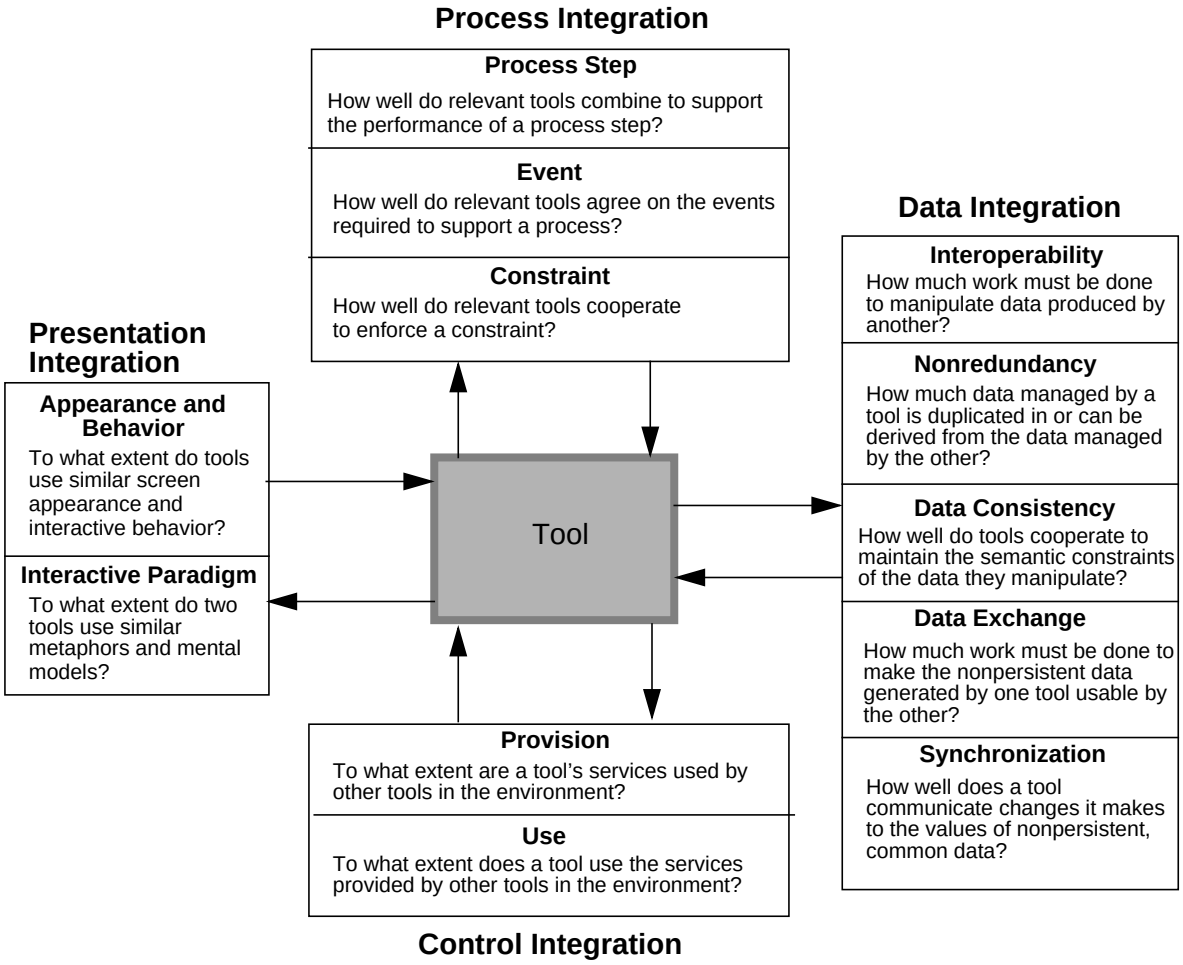

Figure 4-1 Properties of Integration[1]

4.1 Platform Integration

The intent of platform integration is to provide network and operating system transparency to software tools. A framework with a high degree of support for platform integration would allow tools to operate in a distributed environment without knowing the details of the environment. In addition, the porting of tools from platform to platform would be simplified. This framework could also support SEEs residing on heterogeneous networks, with platform differences hidden from the SEE and the tool by the framework. Such support was an original design goal of PCTE, and distribution is part of the both the PCTE 1.5 and ECMA PCTE specifications.

Limited experience suggests that PCTE 1.5 implementations can serve as effective substrates for tools, offering a significant degree of support for platform integration. Hewlett Packard has constructed a prototype implementation of BMS using PCTE 1.5 services [Oliver 91]. Oliver

1. From Thomas and Nejmeh's "Definitions of Tool Integration for Environments" [Thomas 92].

found the PCTE 1.5 interface to be useful and complete for the needs of the BMS and for a small number of simple tools. In addition, the designers of the BMS/PCTE 1.5 prototype found that they could ignore distribution and instead rely on the PCTE 1.5 transparent distribution facilities. PCTE 1.5 also proved to be a useful portability platform, as Hewlett-Packard engineers were able to quickly port several thousand lines of tool source code between heterogenous workstations running PCTE 1.5 implementations.

It should be remembered that ECMA PCTE is only one candidate for inclusion as a SEE framework component; many other components will be necessary. To achieve a higher level of platform integration, each of these other components must be standardized across platforms.

One obvious candidate for inclusion in any eventual environment infrastructure is the X-Window System. Other candidates are represented in the work of various groups attempting to define infrastructure standards (e.g., ECMA, X3H4, X3H6, OMG, etc.).

Additional infrastructure needs will likely be identified as experience is gained with the initially limited set of infrastructure services. Such experience is likely to come from U.S. and European initiatives to build environments on top of framework services, such as STARS, EAST, and Enterprise II. Framework specifications (including ECMA PCTE) will likely change as experience is gained in environment construction and use.

4.2 Presentation Integration

In an effort to provide the consistent “look and feel” characteristic of environments with strong presentation integration (such as the Apple Macintosh computer), PCTE 1.4 provided a unique user interface system. However, in 1989 ECMA abandoned work on a user interface specification for ECMA PCTE and adopted the X Window System as the user interface portability platform [Dawes 91]. Therefore, ECMA PCTE only indirectly supports presentation integration.

Use of the X Window System alone guarantees only weak presentation integration. Additional framework support in the form of look and feel formats such as Motif can potentially lead to stronger presentation integration within a SEE. Clearly, tools which conform to the X Window System and a “look and feel” standard such as Motif will display increasingly similar appearance and behavior.

However, the X Window System and “look and feel” standards taken together do not address such important factors for a consistent user interface as the content of screens and menus and the vocabulary used for various operations. Many of today’s “integrated” tools (sometimes even those of the same toolset) have poorly conceived and inconsistent interactive paradigms. Lack of consistency in these paradigms has led to the diverse interfaces that are apparent when comparing the various X-Windows and Motif (or Open Look) based tools currently available. There is little effort directed toward standardizing interactive paradigms.

Therefore, the use of ECMA PCTE (or any other framework) will not guarantee strong presentation integration. However, it can facilitate such integration by encouraging vendors to recognize that their long term interests are best served by cooperation rather than competition. Ultimately, however, strong presentation integration between tools will require additional cooperation on the part of tool vendors, SEE vendors, and framework suppliers.

4.3 Control Integration

Strong control integration requires that tools be able to notify one another of events and potentially control the execution of other processes. Available PCTE 1.5 implementations and the ECMA PCTE abstract specification provide a number of low level services that can be used in the implementation of more sophisticated control integration mechanisms. These low level services are primarily similar to basic UNIX mechanisms. In addition, ECMA PCTE provides a notification mechanism to alert entities to access to objects.

A level of communication and control interface higher than that provided by PCTE 1.5 and ECMA PCTE would be useful to tool builders, since several low level ECMA PCTE service calls could be replaced by fewer “high level” calls. Also, ECMA PCTE, while providing communication and control primitives, does not define a concept of operation identifying how messaging will work between multiple tools. It also does not specify the format and contents of the messages.

Framework products that provide messaging capabilities are available for many computer manufacturers, including Sun, Hewlett Packard, IBM, and Digital. The potential for integration of these messaging capabilities is represented by the prototype re-implementation of Hewlett Packard’s BMS product on top of Emeraude’s implementation of PCTE 1.5 [Oliver 91]. In this prototype, BMS provides additional messaging capabilities for tool registration and message broadcast, implemented using the PCTE 1.5 message queue mechanism [Oliver 91]. A similar integrated messaging capability is being developed by Vista Technologies, Inc. for their Hyperweb system.

An alternative approach to adding control integration capabilities to ECMA PCTE is being investigated by Digital [Argento 92]. This approach combines a CORBA interface to ECMA PCTE data model, effectively combining control and data capabilities. As indicated previously, Digital believes it can implement ATIS in this manner.

Even with the addition of CORBA or BMS-like capabilities to a framework incorporating ECMA PCTE mechanisms, the degree of control integration possible remains governed by the extent to which a tool makes it possible for other tools to invoke its functionality, and the extent to which it notifies other tools of operations. In order to address the issues of when a tool notifies and responds to others, the CASE Communiqué group has been formed. This group, sponsored by Hewlett Packard, IBM, and Informix, consists of over fifty members which are trying to define common communication interfaces for classes of services. Initially, this group has focused on communication interfaces in areas such as version and concurrency control,

document generation, reverse engineering, analysis and design and debugging. The model developed is intended to support multiple implementations of control integration services.

4.4 Data Integration

As indicated in previous sections, ECMA PCTE has its roots in the early 1980's concept of an SEE. In keeping with this heritage, ECMA PCTE primarily provides services for data integration via shared object management [Wallnau 91]. The relationships between objects, attributes and links, version history, and activities or transactions can be defined in a schema. The objects in the schema are commonly used to represent the data used by tools, while the links are used to define the relationships between tool data.

Among the many services provided by ECMA PCTE in support of data integration are:

- creation, manipulation, traversal and deletion of objects and links
- creation and "snapshotting" of object versions
- creation and manipulation of schema
- management of distribution
- process creation, control, and termination
- security mechanisms
- management of message queues
- notification of access to an object
- concurrency control

The services provided by ECMA PCTE address many of the problems of data integration. Experience with database systems and earlier software environments suggests that data in a shared repository is often less redundant and more consistent than data in multiple repositories. The consistency of data is further supported by the distribution and security mechanisms of ECMA PCTE. In addition, the common schema can help improve the interoperability of tools, since tools sharing a schema can (hopefully) comprehend and access the data of other tools sharing the schema.

As with other forms of integration, there is little experience enabling us to evaluate whether ECMA PCTE-based environments will provide the desired level of data integration. It is likely that users currently do not fully understand the type and degree of data integration desired. It is also difficult to predict how and when tool vendors and environment builders will choose to use ECMA PCTE services to provide data integration.

This stated, we believe (based on our own experience with PCTE 1.5) that ECMA PCTE can offer a useful degree of data integration, even when tool data are only weakly integrated into ECMA PCTE. Particularly useful are the capabilities to establish links between divergent data (e.g., that data produced by testing tools, document tools, compilers, and analysis and design tools), and create configurations based on those links. Also helpful is the potential to navigate

the object base due to the links. While ECMA PCTE does not directly define a navigation capability, our experience show that tools can be easily built to provide database navigation for end users (browsing tools) and for other tools (such as documentation tools).

While the potential benefits of ECMA PCTE are significant, two major issues must be resolved prior to adoption of the technology on a wide scale. These issues include limited support (in the ECMA PCTE specification) for frequently accessed (often called fine-grained) objects, and the need for the development of common schemas. These issues and others are discussed in Section 7.

4.5 Process Integration

A desired goal of many integrated environments is the automation of a software process. A high degree of such automation (process integration) insures that tools interact effectively in support of a defined software process.

One criterion for an environment framework such as ECMA PCTE should be the level of support provided which enables environment builders to perform successful process integrations. Among necessary capabilities for process integration are the management of user and group workspaces, control and monitoring of a defined software process, the specification of user roles and security, process guidance and enforcement, and the management of tools and objects.

ECMA PCTE provides a degree of support for a number of the needs of process integration, particularly in the areas of specifying user roles and security and the management of tools and objects. A commonly recognized limitation of ECMA PCTE is the lack of support for the sort of high level control integration capabilities on which much of process integration support is built.

It should be recognized that process integration is more than the sum of the other forms of integration. Process integration requires the use of the other forms of integration in pursuit of an organization's unique approach to building software. Unfortunately, work at the SEI suggests that few organizations have carefully defined the manner in which they build software (their *software process*). There is even less experience with the development of environments encoding software process (*process-centered environments*), in large part due to a nascent understanding of the software process and tool integration technology. Experience developing and using environments will be essential in order to fully understand the types of framework support necessary.

In recognition of the limited experience with automated process support, the Software Technology for Adaptable Reliable Systems (STARS) effort is attempting to provide empirical evidence regarding appropriate process concepts and SEE process support. Two STARS prime contractors have committed to using ECMA PCTE as an environment framework on which process centered environments will be built. Paramax (a STARS prime contractor) has implemented an ECMA PCTE subset on top of Emeraude PCTE 1.5 V12. It is hoped that

STARS and other similar efforts will identify the requirements and provide evidence for the value of ECMA PCTE (and other framework technology) in support of process integration.

5 How is ECMA PCTE used?

Many organizations have expended significant resources in adopting tools which have been carefully chosen to support the organization's software process. These tools are hosted on the organization's current computing platform and few (if any) are designed to take advantage of the services provided by ECMA PCTE. Important considerations regarding the potential adoption of ECMA PCTE by these organizations are the difficulty of migrating existing Unix tools ("foreign" tools) to make use of ECMA PCTE capabilities, and the expected benefits of the resulting integration. Equally significant over the long term (as increasing numbers of tools are designed to make use of ECMA PCTE services) are the costs of integrating ECMA PCTE-based (or "native") tools into viable SEEs.

An understanding of the techniques that will be used to integrate foreign and native tools into an ECMA PCTE environment can also be beneficial in evaluating the inevitable integration claims from tool vendors. In reality, tools which claim integration with ECMA PCTE can accomplish that integration in different ways with different costs and benefits. The various techniques used to integrate tools into a ECMA PCTE environment, alone with the associated advantages and disadvantages, are discussed in the following sections.

5.1 Foreign Tool Encapsulation

ECMA PCTE provides facilities to allow data exchange or communication with tools running on a foreign system. No change is required to the foreign tool. The integration of the foreign tool is limited since foreign tools cannot make use of the many services provided by PCTE. For some applications, this level of integration may be adequate, and it provides an easy way to allow non-ECMA PCTE tools to cooperate with tools in the ECMA PCTE environment. With most pre-existing tools provided by a third party supplier, there will often be no way for tool users to modify the tool to run on ECMA PCTE. Hence, foreign tool encapsulation may be the only option for some tool users.

Foreign tool encapsulation will often involve transferring data from a non-ECMA PCTE tool into the ECMA PCTE OMS and back again (Figure 5.1). Using this approach, tool data is commonly represented as the contents of a ECMA PCTE file object. The ECMA PCTE object may be versioned and linked to other objects in the object base in the standard manner. To use the tool, the contents of the ECMA PCTE object are copied to the file system. At the end of tool processing, the tool data is replaced in the repository, potentially as a new version.

In experiments carried out at the SEI, this level of integration has been found to be useful in providing limited data integration between tools which were developed independently of PCTE 1.5 and each other.

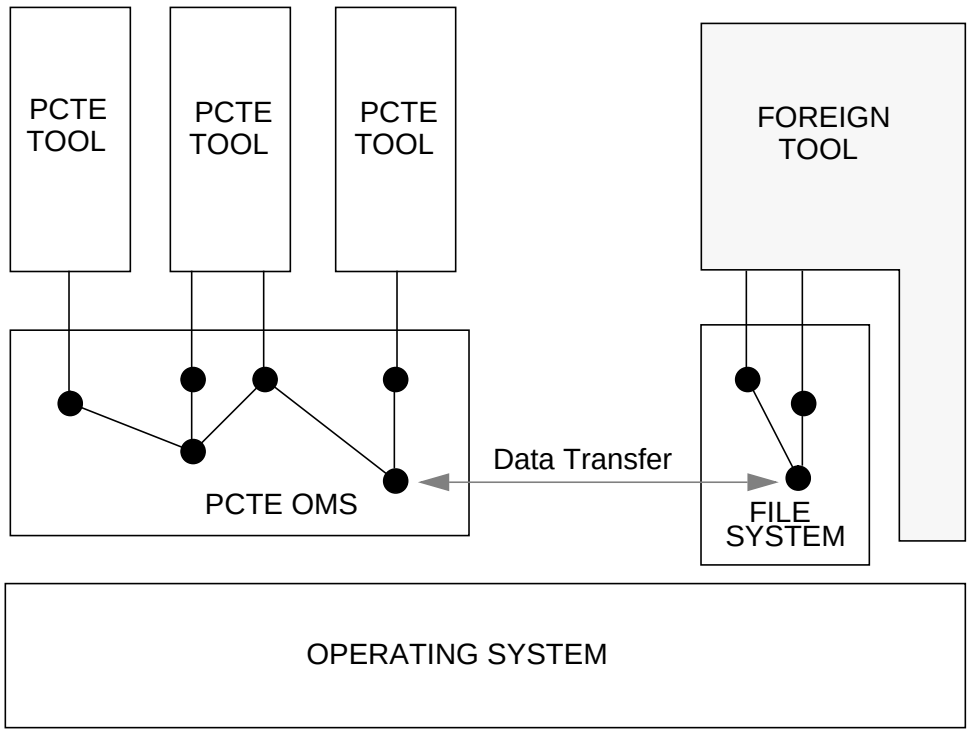

Figure 5-1 Foreign Tool Encapsulation

5.2 File Level Integration

The next level of integration of a tool into ECMA PCTE is at the coarse-grained, or file, level. Tools integrated into ECMA PCTE at the file level use the contents of ECMA PCTE objects exactly as they would use files in a conventional operating system. The functionality of the tool is not decomposed, and the process structure of the tool is not altered (Figure 5-2).

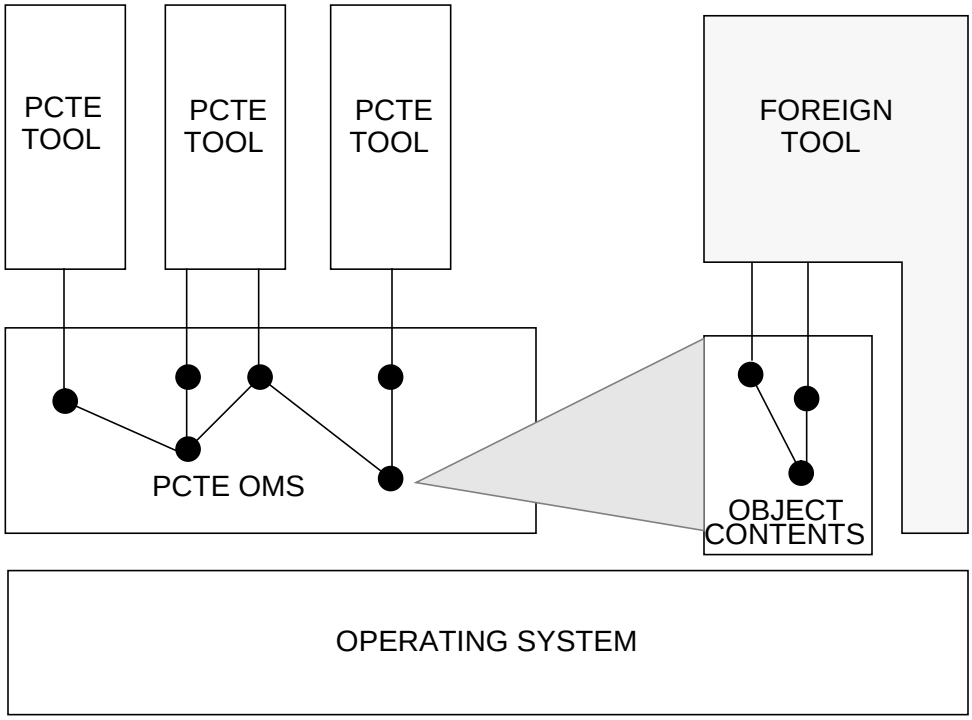

Figure 5-2 File Level Integration

This level of integration allows the tool's data to be stored in the OMS where it can be explicitly linked to other, related data. For example, a documentation tool could be integrated in this way, and documentation produced by it could be explicitly linked to the source code to which it relates. Tools such as browsers and document generators can be developed to follow links to related objects. This in itself is a considerable advantage over what is available with conventional operating systems. Another advantage from file level integration with ECMA PCTE comes from the potential for centralized configuration management and version control over related data.

It is comparatively easy to implement or port a tool to ECMA PCTE in this way. Most of the concepts of ECMA PCTE do not need to be understood by the person doing the implementation or porting and, essentially, all that is needed is to replace traditional file input/output by object contents input/output. However, in general the source code of the tool will have to be modified to achieve this.

File level integration, while providing support for data integration, does not take full advantage of the data structuring facilities of ECMA PCTE. Other tools must access the data at the file level and the environment knows nothing about the internal structure of the data (stored as an object's contents), nor of the internal structure of the tool's process.

5.3 Structured Integration

More extensive use of ECMA PCTE is possible when the data structure and the process structure of a tool is represented in the OMS (Figure 5-3). For example, the internal structure of a document or piece of code could be represented in the OMS by having separate paragraphs or procedures stored as the contents of separate objects, and relationships between components explicitly represented through the object typing and inter-object links. This means that other tools (for example, browsers) will have direct access to this structure without having to parse an entire document or program.

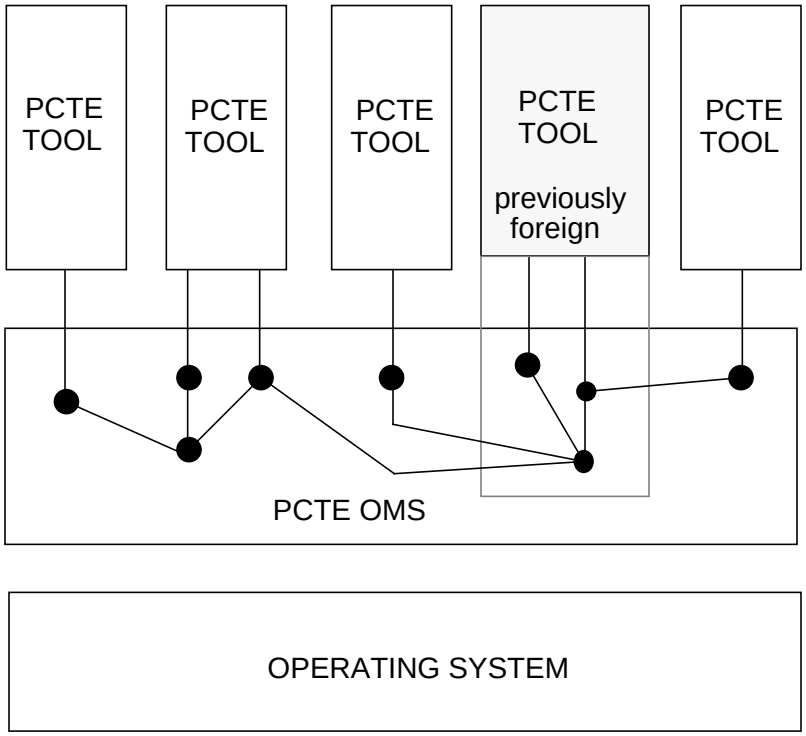

Figure 5-3 Structured Integration

The ECMA PCTE schema for the data structure must be defined before such a tool can be implemented. Either an existing schema is reused or extended, or a new schema must be designed. Obviously, the former alternative is preferable, but if an existing schema cannot be reused, then a new one must be designed. This requires considerable effort and experience (comparable to that required for schema design in any database application) and is a far from trivial task. In fact, it requires a detailed understanding of the database mechanisms and of the application domain, and political understanding coupled with the willingness to compromise.

One must also consider the question of the granularity of the data to be represented in the contents of ECMA PCTE objects. In some applications, one may want to explicitly represent data structures down to a very fine level where each object is only a few bytes in size.

However, ECMA PCTE was designed with the assumption that the contents of objects would be “file-sized” pieces of data (i.e., in the hundreds to thousands of bytes range).

The question also arises of the structure of the tool itself. ECMA PCTE provides facilities to enable a tool to be decomposed into a number of processes. If this is done then other tools within the environment can be aware of the internal state of the tool while it is running. Furthermore, ECMA PCTE locking can be used to obtain optimal concurrency control between tools, and the transaction facilities of ECMA PCTE can be used to ensure that the tools always leave the OMS in a consistent state. From a process point of view, the advantages are that the tool’s execution is more modular. Sub-processes can be reused, monitored more easily, changed more easily, and also replaced with others from different tools (i.e., delegate services to other tools).

To take advantage of all the facilities of ECMA PCTE, a tool will have to be written from scratch, or will have to be extensively revised from a traditional operating system-based tool. The transition from an operating system-based tool to a ECMA PCTE-based tool is in some ways analogous to the transitioned from character-oriented (i.e., VT 100) to bit-mapped terminals. While it is possible to provide a window interface to a command-line based tool, the full benefit of a bit-mapped display is not realized until the tool is extensively revised to make use of the higher level of user interaction common to bit-mapped displays (i.e., multiple menus, windows, point and click interfaces, etc.).

The transition to ECMA PCTE will require a major effort. The advantages are that data can be better understood, used, and analyzed, and processes are more controlled, explicit, and delegable.

6 What Implementations are Available?

As indicated in previous sections, three unique environment framework interfaces (PCTE 1.5, PCTE+, and ECMA PCTE) fall under the PCTE rubric. The current and potential availability of implementations of the divergent PCTE interfaces differs, and therefore each PCTE variant is addressed separately. In addition, a number of currently available or planned environments built on top of PCTE are discussed.

While a summary of the current availability of the various PCTE versions may be useful to determine the state of the technology and identify sources of information, in the long run tool vendors and users need be concerned only with ECMA PCTE. No new implementations of PCTE 1.5 are in progress, and the builder of the most widely distributed PCTE 1.5 implementation, GIE Emeraude, has stated that its product will be replaced by an ECMA PCTE implementation. In addition, there has been a formal commitment to abandon work on PCTE+ interfaces at the end of an assessment phase for current implementation activity.

While an attempt was made to insure the accuracy of this information, more comprehensive and up to date information regarding PCTE activity is available from the North American PCTE User Group (NAPUG). Addresses and phone numbers for NAPUG as well as for providers of PCTE and related environments and services are available in Appendix A. Readers interested only in activity related to ECMA PCTE might consider skipping sections 6.1 and 6.2 (PCTE 1.5 and PCTE+).

6.1 PCTE 1.5

The most readily available version of PCTE 1.5 is the GIE Emeraude V12 product. This product is available on a number of Unix platforms, and implements the PCTE 1.5 specification plus additional services. Emeraude V12 is distributed in the United States by Software Design and Analysis (SDA), Mark V Systems, and IPSYS USA. These companies also offer PCTE training and consulting expertise. Emeraude V12 serves as the framework basis for two commercially available software engineering environments (EAST and Enterprise II) and a growing number of existing or soon-to-be-released commercial toolsets, including Vista Technology's Hyperweb and the DMR Group's Cobol Environment.

The Société Française de Génie Logiciel (SFGL) EAST software environment is now being marketed to organizations in the aerospace industry. Currently, SFGL does not have a US distribution network in place. However, information about EAST and SFGL is available through Mark V Systems. U.S. operation and distribution are scheduled to be introduced prior to the end of 1992. Organizations interested in additional information about the EAST environment should request a copy of the SFGL document titled "EAST Introduction".

Syseca's Enterprise II environment was originally built for the French Ministry of Defense, but is now offered as a commercial product. It provides a user interface, tools and schemas on top

of Emeraude V12. Syseca is currently negotiating with potential distributors of the Enterprise II environment in the United States.

Vista Technologies Hyperweb system is just now entering the PCTE market. Hyperweb utilizes the Emeraude V12 object base to provide support for a set of tools to make it easier to create, access, and manipulate data using the PCTE OMS. Hyperweb provides a message passing capability on top or the basic Emeraude V12 services in order to implement hypertext-like links between objects in the Emeraude database

The DMR Group of Montreal is currently constructing a toolset on top of Emeraude V12 and the EAST environment to support their own Management Information System process and methodology. The toolset, designed to compliment DMR's consulting business, contains a third party Cobol toolset (compiler, linker, debugger, etc.) along with DMR tools to support process modeling, estimation, testing, and configuration management.

In addition to the larger scale toolset and environment efforts built on Emeraude V12, CASE tool vendors have voiced support for the product, and rumors persist of in-house integrations undertaken by the vendors. At least one paper is available detailing the results of such integration efforts [Oliver 91]. None of these products has actually reached the marketplace, however.

A number of research and development projects have also experimented with the Emeraude V12 product. These projects include the Atmosphere project in Europe and the STARS project in the U.S. The STARS effort is also investigating the capabilities of the EAST and Enterprise II environments.

Only one other publicly available implementation of PCTE 1.5 has been reported. In September of 1991, Verilog of France announced a version of PCTE 1.5 for the VAX/VMS architecture. Verilog's PCTE 1.5 is built on ORACLE relational database technology. SQL queries can be made of a relational view of the PCTE object base.

6.2 PCTE+

No complete implementation of the PCTE+ specification is readily available, although a number of partial implementations and prototypes have been developed.

VULCAN, developed by Heuristix and marketed in the U.S. by S2 Systems, is an integrated CASE toolset built on top of a framework derived from PCTE+ specifications. During the development of the VULCAN product, Heuristix engineers adopted the PCTE+ specification to fill a need for common data management services. Only those portions of the PCTE+ specification necessary to support the needs of the VULCAN toolset were implemented. It is unclear whether the PCTE+ component of the VULCAN toolset will be marketed independently. S2 and Heuristix have recently announced the intention to build an implementation of ECMA PCTE.

A few organizations (GIE Emeraude and EDS-Scicon) have built PCTE+ prototypes. GIE Emeraude has fielded a PCTE+ prototype to be assessed by various European organizations. Emeraude feels that this prototype provides a solid basis for a full ECMA PCTE implementation.

EDS-Scicon[1] is currently developing an implementation of PCTE+ for the U.K. Ministry of Defence. This implementation is hosted on VAX/VMS and will include a number of integrated commercial tools. EDS-Scicon PCTE+ is implemented in Ada on top of an ORACLE database, but is constructed to operate on a number of relational databases via an SQL interface. A prototype of the EDS-Scicon implementation is now available. The production version of EDS-Scicon's PCTE+ will implement the ECMA PCTE security model rather than the PCTE+ equivalent.

6.3 ECMA PCTE

As mentioned previously, there is no commercially available implementation of ECMA PCTE. However, a number of organizations are developing such implementations, with a few pointing toward a 1993 release date. Recently, the list of organizations committed to implementing ECMA PCTE products has grown to include IBM and Digital Equipment. In Europe, the Eureka project is involved in three ECMA PCTE efforts, including work by GIE Emeraude/SFGL, EDS-Scicon/ICL, and Softlab.

GIE Emeraude is basing a partial implementation of the ECMA PCTE standard on their existing PCTE 1.5 implementation. When available, this product will implement a subset of ECMA PCTE, but will provide a level of compatibility with the current Emeraude V12 product and thereby offer a logical upgrade path for current users. GIE Emeraude's complete ECMA PCTE variant is being implemented based on their PCTE+ prototype. It is scheduled for field testing around mid-1993, with public release around the end of 1993.

EDS-Scicon/ICL plan to use their current VAX/VMS PCTE+ prototype as the basis for a full implementation of ECMA PCTE. We are not aware of a planned release date for the production version of this implementation.

The Softlab effort is planning to build an ECMA PCTE implementation on top of the current Maestro II Project Support Environment. The Maestro II product is centered around an object management system which is intended to provide the basis of the ECMA PCTE implementation.

Digital Equipment, in concert with International Software Systems Incorporated (ISSI) has announced the development of an ECMA PCTE implementation built on a commercial database product and incorporating capabilities from ATIS and the Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA). Work underway at

---

[1] EDS-Scicon is the former SD-Scicon. SD-Scicon was recently purchased by the U.S. firm Electronic Data Systems (EDS), and the name was changed to incorporate that of the new parent company.

Digital Equipment is attempting to define a common ground between ATIS and PCTE. In addition, it was announced that the COHESION environment will support ECMA PCTE as implementations become available. Digital also indicated an aggressive schedule for fielding an ECMA PCTE product in the 1993 time frame.

At the CASE'92 conference in Montreal IBM detailed an effort to develop an ECMA PCTE implementation. In addition to the announcement, a tool was demonstrated which browsed a prototype version of IBM's ECMA PCTE object base.

S2 and Heuristix have recently announced an ECMA PCTE strategy. They are committed to producing a full ECMA PCTE implementation along with supporting tools and services for tool builders. The release plan indicates an initial release of an ECMA PCTE subset based on Heuristix existing PCTE+ capabilities. This release will be followed by full ECMA support. Heuristix is also exploring extensions to the ECMA PCTE OMS in order to provide better fine-grained support. In addition, an integration of ECMA PCTE with Hewlett Packard's BMS is planned.

For those interested in experimenting with the ECMA PCTE, Paramax has implemented a partial Ada binding supporting the ECMA interface on top of Emeraude V12. They have also built a browser for the Emeraude OMS. The Paramax work is available via anonymous ftp (to the public file server stars.ballston.paramax.com).

Finally, at the recent NAPUG meeting (September, 1992), a North American ECMA PCTE initiative sponsored by several government agencies was briefed. This effort is anticipated to provide a forum and focus of North American involvement in ECMA PCTE by developing a publicly available reference implementation of PCTE. Details of this initiative should be available in the near future.

As there are no ECMA PCTE products, no current environments have been built on the framework. However, a number of environment efforts are committed to building on the ECMA PCTE framework, including STARS, which is planning to build two environment prototypes based on ECMA PCTE, and the Eureka project, which will likely provide some environment support built on ECMA PCTE. In addition, it is possible that some of the organizations committed to implementing ECMA PCTE will produce environments built on their framework.

It also appears that a growing number of CASE tool vendors are expressing interest in PCTE. However, many appear to be waiting for a viable ECMA PCTE implementation prior to committing to serious reworking of their tools. For example, Cadre Technologies has recently announced the expected fielding of an ECMA PCTE compliant version of Teamwork in the 1994/1995 time frame. For the initial ECMA PCTE compliant version of Teamwork, Cadre will perform a file level integration with the PCTE OMS. Later, Teamwork will be redesigned to take full advantage of the services provided by PCTE.

Finally, the PCIS effort is attempting to define a common environment framework for NATO. The PCIS environment framework is to consist of a generic set of facilities provided to

environment developers. Syseca's Enterprise II is being considered as a baseline. The environment framework will provide common user interface, data, and control integration facilities. The PCIS effort is to completed by the end of 1993.

7 Issues Concerning ECMA PCTE

The release of the Emeraude ECMA PCTE implementation is scheduled for 1993. It is likely that additional implementations may come from the various PCTE+ baselines in Europe and India, and from domestic organizations including IBM and Digital.

In part, the NAPI initiative is due to fear that the United States will fall far behind Europe in ECMA PCTE technology. The fear that is in part driving the growing interest in ECMA PCTE in the U.S. is threefold: domestic efforts to produce an equivalent framework for software environments (such as CAIS-A) have foundered; domestic organizations have a profound fear of becoming non-competitive with international competition using better technologies (presumably ECMA PCTE); and domestic organizations are unsettled by embracing technology over which their is little indigenous expertise or control (as ECMA PCTE pushes toward ISO standardization, there is no domestic equivalent of ECMA).

While the technological push for ECMA PCTE was initiated from Europe, a major pull toward adopting the technology appears to be originating in the U.S. This may be due in part to the large size of a number of domestic, primarily DoD, software initiatives (such as the CIM initiative). Domestic vendors which have promised ECMA PCTE support are certainly responding in part to the demands of such software initiatives.

In spite of this growing support, ECMA PCTE is best considered to be a hypothesis in need of testing. Long term acceptance of ECMA PCTE will depend on the manner in which a number of remaining challenges are resolved. In order to identify important challenges, comments were collected by electronic mail, at forums such as NAPUG, and based on our own experience using PCTE 1.5. Major issues include:

- Reliance on unproven technology
- Competition from object-oriented technologies
- Support for “fine-grained” objects
- The challenges of migrating tools to ECMA PCTE
- Incorporation of other technologies into a SEE framework
- Development of environments based on ECMA PCTE

7.1 Reliance on Unproven Technology

Perhaps the most significant argument against the standardization of ECMA PCTE is the lack of validating experience with the technology. The growing momentum toward ECMA PCTE is counterbalanced by the glaring fact that there are no commercially available ECMA PCTE implementations. While it is true that the ECMA PCTE specification has undergone international scrutiny and assessment, this scrutiny is no match for experience using ECMA PCTE to produce large, production-quality environments.

Sources within the framework community suggest that the initial implementations of ECMA PCTE will include incompatible subsets and super-sets, due to the immaturity of technology and the rush to field products. This will pose a problem for tool vendors, as any individual tool implementation using an ECMA PCTE subset or super-set will not automatically port to other implementations.

When the ECMA PCTE implementations are released, potential users can also expect a period characterized by “teething” problems. These teething problems will likely be due in part to the typical performance and usage problems associated with new software. However, the more significant problems will involve the migration of tools to the framework and the specification and development of useful environments. These other problems will be address in subsequent sections.

Experience gained from using the early ECMA PCTE implementations will be critical for defining the requirements and specifications of future environments. We believe that such experimentation can also help an organization gain an understanding of unique process and integration needs, and potentially provide a competitive advantage regardless of which framework products eventually gain favor.

7.2 Competition from Object-Oriented Technologies

ECMA PCTE may prove to be a good framework technology and repository for tool data. However, ECMA PCTE will likely compete for pre-eminence with Object Oriented Database Management Systems (OODBMS) as they mature. It is also sometimes claimed that OODBMS can address a larger market encompassing both application development environments (addressed by ECMA PCTE) and application systems needs (not addressed by ECMA PCTE).

OODBMS technologies combine data and control integration technologies, and offer capabilities not provided by ECMA PCTE. OODBMS advantages include inheritance from parent to child types, the association of data with the actions performed on the data, flexibility in type definitions, and a close correspondence between the domain and the database model [Ochuodho 92]. Disadvantages include lack of powerful concepts for modelling relationships and inflexibility due to the pre-specifying of all operations on data.

ATIS, an object-oriented system originally developed by Atherton Technology, is a competing technology with ECMA PCTE. At least two companies (Atherton and Digital) have developed divergent versions of ATIS. Between them, there is a moderately-sized installed base. A few environments have been built on top of ATIS, but these environments tend to be small subsets of a SEE (such as the Verdix VADS APSE) or have not been extensively used in production settings.

While ATIS cannot be legitimately called a standard since no formal body has endorsed the specification, the existence of an installed base does present a powerful argument in its favor. In addition, ATIS is being evaluated by the ANSI committee X3H4 as a potential basis for the

second version of the IRDS data repository standard (IRDS2), and by the ANSI committee X3H6 to determine how well ATIS will satisfy control and process integration requirements for tool integration.

The current NAPI initiative has been opposed by some advocates of the ATIS approach, who argue that ATIS is technologically superior to ECMA PCTE, primarily because of its objected oriented approach which associates actions with data, thus combining control and data integration capabilities. In addition, ATIS advocates point out that at least two (divergent) implementations of ATIS are available today, while no ECMA PCTE implementation has been fielded.

In spite of the concerns of ATIS advocates, ECMA PCTE is being promoted for ISO standardization via the “fast track”. Some observers believe that ECMA PCTE is likely to be standardized regardless of the direction taken by domestic organizations.

Domestic organizations supporting the NAPI initiative suggest that since ISO standardization is likely to happen, a North American role in influencing the current and future direction of the standard is essential. This group argues that active involvement in the investigation and implementation of the technology is essential in order to gain influence.

7.3 Support for “Fine-Grained” Objects

It is claimed by both opponents and proponents of ECMA PCTE (and PCTE 1.5) that the technology is best used to model coarse relationships between data, rather than the fine relationships within data often needed by a single tool. This distinction is commonly phrased in terms of better support for “coarse-grained” objects than for “fine-grained” objects.

Unfortunately, the phrase “support for fine-grained objects” is somewhat misleading. ECMA PCTE limitations do not appear to be related to the size of the object accessed, but rather to the frequency of access to the object. This characteristic of ECMA PCTE is the result of a policy that requires that permissions be checked for access to all objects.

The ECMA PCTE access policy appears appropriate for the large data artifacts commonly left by tools. The use of an ECMA PCTE object to represent a “small” object entails overhead that may be inappropriate for the object we are trying to represent ([Kelter 91], [Kelter 92]).

Within a single tool (or within a very tightly integrated set of tools), it is likely that some subset of the data should not be subject to such rigorous security checking. For example, each entry in the symbol table of a compiler is considered a unique object by the compiler, but for performance reasons few compiler designers would choose to check permissions each time the compiler searches the symbol table. To help alleviate the problems of fine-grained data access, the ECLIPSE environment used a two-tier database [Bott 89], with PCTE used for coarse-grained storage of tool artifacts, and other technologies for fine-grained access within artifacts.

While in general situations requiring high frequency retrieval of information from the object base are not optimal for implementation as ECMA PCTE objects, ECMA PCTE does provide some capability to model fine-grained data as attributes of an object. In some applications, there may be classes of fine-grained objects that can and should be represented as attributes on ECMA PCTE objects. There are other fine-grained objects, however, for which storage as an attribute may be inappropriate.

One result of the lack of support for fine-grained data is that the competition between ECMA PCTE (ER) and competing OODBMS technology may not result in one solution or the other predominating. Approaches coupling the advantages of ECMA PCTE with object orientation are already appearing in proposals such as IBM's OOTIS (Figure 7-1), and Digital's commitment to combine CORBA, ECMA PCTE and ATIS technology.

When the integration situation demands a flexible solution that can model the relationships between data from various tools, an ER (ECMA PCTE) approach may be appropriate. When the situation demands closer inter-operation of a tightly related set of tools, perhaps with more frequent access of related data, an OO approach may be appropriate. As of this date, however, ECMA TC33 has no plans to extend ECMA PCTE to provide better support for frequently accessed objects.

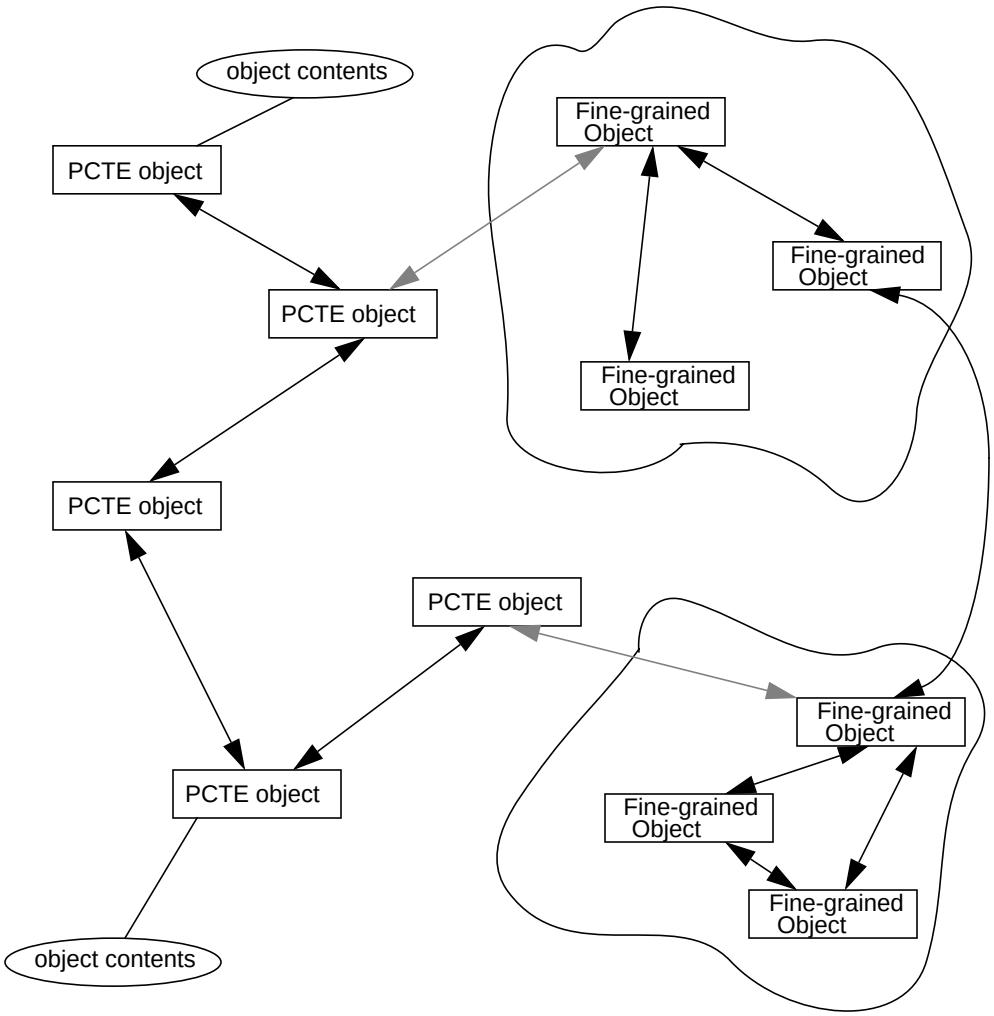

Figure 7-1 Coupling of ECMA PCTE and Object Oriented Capabilities[1]

7.4 Migrating Tools to ECMA PCTE

Many vendors have said they are “considering”, “discussing”, “investigating”, or “committed to” ECMA PCTE but few appear to have concrete plans and project staff in place. However, tool vendors can be expected to follow the integration strategy of first delivering only coarse, file-level integrations requiring few source code changes, followed later by structure-level integrations as the market develops. True structure-level integrations may only slowly become available on ECMA PCTE implementations.

This strategy of slow migration to ECMA PCTE is dictated by the uncertain availability of the ECMA PCTE framework as well as the small size of most CASE tool firms. Few tool vendors

1. Adapted from Harrison, Ossher, and Kavianpur [Harrison 92].

can afford to maintain separate and perhaps vastly different versions of a tool for the ECMA PCTE and traditional operating systems markets without significant user demand.

The transition to deep, structure-level integrations with ECMA PCTE will not be a simple task for tool vendors. Most tools will require significant redesign to make good use of ECMA PCTE services. Unfortunately, there is little publicly available information detailing how tools should be designed to make best use of ECMA PCTE. One potential source of such information is NAPUG, where tool vendors frequently discuss their plans and implementations.

In principle, the availability of the higher level database services of ECMA PCTE will make writing new tools easier, because tool vendors will no longer be required to build database capabilities into tools. In addition, it is hoped that as ECMA PCTE frameworks become available on more computing platforms, and are more widely accepted, vendors will be able to reduce the number of versions of software they are forced to maintain.

A number of CASE tool vendors have already chosen to make use of commercial database technology in order to simplify implementation and maintenance of their software. Migration to ECMA PCTE may be simpler for these vendors and for other organizations already using commercial database technology as part of the framework on which tools are constructed. However, it is unlikely that ECMA PCTE will meet the complete database needs of all tools. Tools that require high performance or frequent access to data will likely continue to provide a custom database capability, although hopefully with enhanced data access and messaging capabilities.

7.5 Incorporating Other Technologies into a SEE Framework

ECMA PCTE and related work (including the other PCTE variants and CAIS-A) derive from the early attempts [STONEMAN 80] to provide tool integration primarily based on a centralized data repository and shared process support [Wallnau 91]. The most important and well-developed class of services offered by ECMA PCTE includes those related to object management, although other services for process management, network management, security, accounting, and auditing are provided.

As already identified, the data integration services offered by ECMA PCTE must be (minimally) supplemented in a SEE framework by services to support presentation integration (the X Window System and Motif/Open Look) and control integration (messaging technology such as the HP BMS). In addition, it is expected that improved support for fine-grained objects will be offered by the completed framework.

However, the services provided by ECMA PCTE (and all other potential framework standards) in many cases overlap with other potential framework standards, such as POSIX and IRDS. In order to identify deficiencies and overlaps in services, the Program Support Environment Working Group (PSEWG) of the Navy Next Generation Computer Resources (NGCR) program is mapping framework services (including those of ECMA PCTE) against the

NIST/ECMA Reference Model for Computer Assisted Software Engineering Frameworks [NIST_ECMA 91].

It is intended that this mapping will help define a minimal set of standards that provide the greatest (or optimal) degree of framework capability. It is also hoped that the PSEWG work will identify overlaps and gaps in framework capabilities. Problem areas may become the focus of further standardization activities.

Unfortunately, the divergent work of various groups involved in standardization (ECMA, X3H4, X3H6, ISO, POSIX, etc.) almost guarantees that instead of a single set of standards to provide framework support, multiple standards will continue to compete for preeminence. For the service area of data repository support alone, X3H4 and X3H6 both have investigated technology (IRDS and ATIS) that provides service overlaps with ECMA PCTE. The role of ECMA PCTE as a standardized provider of repository support in domestic environment frameworks is likely, based on the strong probability that it will be approved as an ISO standard, and the expressed interest of vendors such as IBM and Digital. However, the role of ECMA PCTE is not assured, since the existence of other competing technology will concern tool vendors and users.

7.6 Developing Environments Based on ECMA PCTE

The integration of individual tools into process-centered environments incorporating ECMA PCTE as a framework capability will require a number of activities for which there is little or no precedent. These activities include:

- tight (structural) integration of individual and groups of tools into ECMA PCTE
- the identification and implementation of appropriate process supports and constraints.

7.6.1 Developing Tight Integrations of Tools

The development of tight structure-level integrations of groups of tools into the ECMA PCTE framework requires the agreement on common schemas. The existence of common schemas will allow tools to rely on the availability and characteristics of data from other tools, potentially reducing data redundancy and increasing data interoperability, consistency, exchange and synchronization. This will reduce the need to specify unique database and file formats, and should simplify the writing of new tools.

First, however, common schemas must be developed, tried, tested, and standardized. It is likely that collections of organizations (including tool vendors and users) will meet to agree on a common set of schemas. Unfortunately, agreement on all but the simplest schema may be difficult, since it is likely that each tool vendor and user will push for their unique interpretation of an appropriate schema (i.e., the schema that most closely matches the current implementation of their particular tool).

Therefore, the initial common schemas that are developed are likely to be general, and represent a sort of least common denominator of competing tools. Only a subset of data of many categories of tools will be represented in common schemas, due to both the unique performance needs of tools and the desire of vendors to maintain some competitive advantage. Among questions to be answered about the resulting common schemas will be whether they provide adequate integration to support highly specialized processes, and how such common schemas will be controlled and enhanced.

It is also likely that some organizations will chose to develop proprietary, perhaps application-dependent schemas and environments for market or internal use. The resulting environments might offer tighter integration than that available using common schemas. Unfortunately, developers of unique environments will find it difficult to use existing tools or port new tools to the environment. They must bear the cost of administering, maintaining and evolving the complete environment themselves. Nevertheless, some organizations will likely choose this path.

It should be noted that the difficulty in developing a suitable set of shared conventions (in this case represented by schemas) is not unique to ECMA PCTE, but is common to other integration technologies. Whether the technology requires a common schema, or set of standards defining when, why, and how access to external tools takes place, users and tool providers will struggle to reach consensus.

A secondary problem of constructing tight tool integrations is the development of a suitable basis toolset to assist tool developers. This toolset will likely include toolsets for browsing the object base, associative retrieval (query) mechanisms and schema generation tools. This basic toolset is not a component of the ECMA PCTE standard. However, work at the SEI has identified the critical nature of such a toolset in developing and manipulating even the simplest object bases.

A final problem is the identification of conventions concerning the responsibilities of environment components. Currently, Unix tools rely on conventions to define tool characteristics such as the storage location for temporary files (/tmp), managing devices like printers, passing parameters between tools, and error checking. Conceptually similar conventions are necessary in order to develop “well behaved tools” using the ECMA PCTE framework. NAPUG is taking a lead in developing such conventions.

7.6.2 Identifying and Implementing Process Supports and Constraints

A SEE should not be considered to be a set of random integrations of tools. Rather, it should be implemented to fit within an organization’s software process. In existing environments, the supported process is buried within the tools and services provided through the ordering of tool invocations, sharing of information between tools, and the usage patterns of the tools [Brown 92].

As our understanding of process support increases, it is likely that a trend toward explicit automated process support and enactment will accelerate. This trend is evident in recent tool announcements such as Hewlett Packard's Synervision and Cap Gemini's Process Weaver. Future integrated SEEs will probably incorporate explicit process support and enactment.

7.6.3 A SEE Conceptual Model

Figure 7-2 presents a conceptual model of a future integrated SEE incorporating ECMA PCTE. The SEE framework will consist of a set of capabilities, provided by an incompletely defined set of framework components. ECMA PCTE is likely to be included in this framework, but other components such as messaging systems (such as BMS), user interface systems (the X Windows System), object oriented database systems (such as ATIS), object request brokers (such as CORBA), and others, are also potential candidates for inclusion. The distinction between the operating system and other framework services will become less clear over time.

In this model, tools will become providers of one of more focused services, rather than provide services (such as object management) which are not central to the core methodology or capability of the tool. The replacing of proprietary user interface technologies in tools with use of the non-proprietary X Window System is an early indicator of this trend.

Some tools will be tightly integrated with ECMA PCTE. Other tools will be foreign, and perhaps be built to utilize other framework products. Mechanisms to related tools built on foreign services will provide useful, but perhaps "loose" integrations between foreign tools and ECMA PCTE tools.

Process control will be provided under the explicit direction of automated process definition and enactment tools. This will offer an enhanced degree of flexibility in terms of the processes that can be supported by a specific environment.

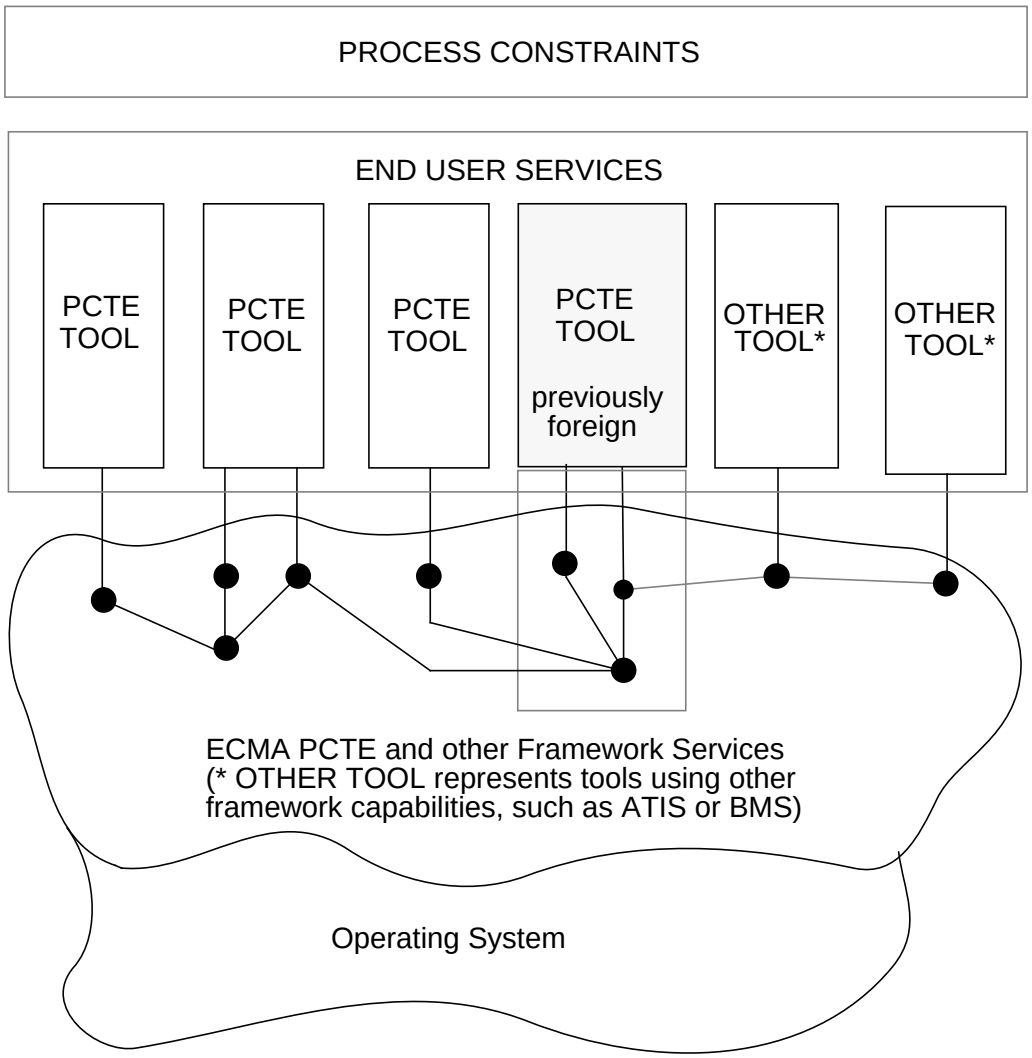

Figure 7-2 An Integrated SEE

8 Conclusion

The experiences of past and current environment efforts suggest that the development of a full life-cycle SEE is a difficult and expensive task. While the ongoing work with ECMA PCTE is promising, it is unclear at this point whether the technology will support a very significant increase in the level of integration possible. SEE frameworks and environments currently have little market penetration (Hewlett Packard's BMS appears to have the widest acceptance). Few organizations have experience with framework technology. Very few reports documenting the results from tool and environment integrations have been published. While major domestic and foreign firms are now promising ECMA PCTE support in 1993, it remains to be seen whether a production quality framework or associated environment can be fielded by that date.

Even if ECMA PCTE technology eventually proves to provide exceptional support for software engineering environments, the transition of the technology into general use will be difficult. Not only will organizations be required to change the process by which they build software (a characteristic of the adoption of any environment technology), but tool vendors must restructure their toolsets to take advantage of the services provided. It is unlikely that a wide variety of tools built to take advantage of the full range of ECMA PCTE services will be available prior to the 1995-1996 time frame. In the interim, we will likely see sets of tools that support parts of the software engineering process. These sets of tools will likely rely on both ECMA PCTE and standard operating system services.

ECMA PCTE may offer a significant improvement in the degree of tool integration within a SEE. There is sufficient likelihood of ECMA PCTE succeeding that organizations should start planning for its success. However, the possibility for failure is such that organizations need to acknowledge the risk, manage it, and continuously reassess the technological and political momentum behind ECMA PCTE.

A careful strategy would entail that organizations gain experience with tool and environment integration using newly available framework technologies (including ECMA PCTE) while avoiding wholesale commitments until mature framework and environment products appear. Suggestions for maintaining this position include:

- Work toward a good definition of your organization's tool integration needs. This will allow you to more easily identify frameworks and environments that provide good value.
- Gain experience with current CASE tools, environments, and integration frameworks on non-critical projects. When mature products become available, you will already possess skills to build, customize, or maintain your environment.
- Maintain a flexible posture toward the various technologies available until technical and market positions are clarified.

- Identify procedures for eventual transition of tool and environment technologies. Without good technology transition strategies, it is less likely that you will reap the benefits that you desire.
- Join and attend meetings of industry groups (such as NAPUG) in order to stay abreast of developments in the field.

Framework and environment technologies, including ECMA PCTE and related environments, but also BMS, ATIS, and other products offered by various vendors, provide a valuable mechanism to investigate the requirements and capabilities of SEEs. In addition to increasing understanding of tool and environment practices and benefits, experience with current technology can help us to gather a baseline of productivity and quality data from which to judge future efforts.

Acknowledgements

We would like to thank the many people who provided excellent comments on previous drafts of this paper, including Ian Thomas, Christian Bremeau, Udo Kelter, John Akerley, Alan Brown, Dave Carney, and Jock Rader. We are also grateful to the many others who were very helpful, but who we may have neglected to include on this long list.

Appendix A Sources of Information about PCTE

General Information

The ECMA PCTE specifications can be obtained from:

European Computer Manufacturers Association
114, rue du Rhône
CH-1204 Geneva
Switzerland
fax. +41 22 786 52 31

PCTE+ documents can be obtained from:

Gérard Boudier
GIE Emeraude
c/o Bull
68, route de Versailles
78430 Louveciennes
France
fax. +33 1 39 02 42 06

The PCTE 1.5 specifications can be obtained from:

PCTE Interface Management Board
Commission of the European Communities
31, avenue de Beaulieu
B-1160 Brussels
Belgium
fax. +32 2 236 83 64

A free subscription to the PCTE Newsletter can be obtained by sending a request to:

EC2
269 - 287, rue de la Garenne
92024 Nanterre Cedex
France
fax. +33 1 47 80 66 29

The North American PCTE User Group (NAPUG), organized by Gérard Memmi and Ian Thomas, meets periodicly. To join the electronic mailing list for the group, send email to:

L.Grundy@bull.com

The group maintains an electronic bulletin board. Technical information and comments may be sent to this bulletin board by addressing email to:

pcte@bull.com

An ftp server containing information about PCTE has been set up by Christian Brémeau, of Software Design and Analysis Inc. Its address is:

ftp.sda.com

and the PCTE information is in the directory:

SDA/PCTE

PCTE and Related Environments

For information about the Emeraude implementation of PCTE, contact:

GIE Emeraude
Sales Department, PC 6A1
68, route de Versailles, BP 3
78430 Louveciennes
France
fax. +33 1 39 02 47 38

Software Design and Analysis, Inc.
1113 Spruce Street, Suite 500
Boulder, CO 80302
USA
fax. +1 303 938 5005

Mark V Systems Limited
16400 Ventura Blvd Suite 303
Encino, CA. 91436
tel. +1 818 995 7671

IPSYS Software
28 Green Street
Newbury, Ma. 01951
USA
fax. +1 508 462 9198

For information about the EAST environment, contact:

Société Française de Génie Logiciel (SFGL)
14, rue de la Ferme
92100 Boulogne
France

For information about the Enterprise II environment, contact:

Syseca
315, Bureaux de la Colline
92213 Saint-Cloud Cedex
France
fax. +33 1 49 11 7645

For information about the Heuristix work with PCTE+, contact:

S2 Systems, Inc.
445 Main Street
Woburn, MA 01801
USA
fax. +1 617 932 8564

Training Courses

PCTE training courses are offered in the US by SDA and Mark V Systems:

Software Design and Analysis, Inc.
1113 Spruce Street, Suite 500
Boulder, CO 80302
USA
fax. +1 303 938 5005

Mark V Systems Limited
16400 Ventura Blvd Suite 303
Encino, CA. 91436
USA
tel. 818 995 7671

References

[Argento 92} Argento, Agusto; Bonferini, Chaira; Dematte, Fabrizio; and Manca, Serena. “ECMA PCTE, CORBA, and ATIS: The Easy Way to O-O ECMA PCTE Services”. *PCTE Newsletter,* 10 (June 1992): 20-27.

[Black 91] Black, Eric. “ATIS, CIS, PCTE and the Software BackPlane,” *Proceedings of the Fourth International Conference on Software Engineering and its Applications*. Toulouse, France 1991. 601-615.

[Bott 89] Bott, Frank, ed. *ECLIPSE — An Integrated Project Support Environment.* London, U.K.: Peter Peregrinus Ltd., 1989.

[Boudieretal 88] Boudier, Gerald; Gallo, Ferdinando; Minot, Regis; and Thomas, Ian. “An Overview of PCTE and PCTE+,” *Proceedings of the ACM SIGSOFT/ SIGPLAN Software Engineering Symposium on Practical Software Development Environments*. Boston, MA: November, 1988. 248-257.

[Brown 91] Brown, Alan W. *A Critical Review of the Current State of IPSE Technology* (CMU/SEI-91-TR-29, ADA244294). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, October 1991.

[Brown 92] Brown, Alan W.; Earl, Anthony N.; and McDermid, John A. *Software Engineering Environments-Automated Support for Software Engineering*. London, UK: McGraw-Hill 1992.

[Conform 86] Ada-Europe Environment Working Group. *PCTE Conformance to the RAC (Ada/WG/6/51)* Issue 2: January 1986.

[Davis 90] Davis, Hugh. “Relationship between ECMA PCTE and PCTE+”. *PCTE Newsletter,* 4 (June 1990): 12-14.

[Dawes 91] Dawes, John and Davis, Hugh. “ECMA PCTE,” *Software Engineering Environments*, Volume 3. Chichester, UK: Ellis Horwood, 1991. 381-400.

[ECMA149 90] European Computer Manufacturers Association. *Portable Common Tool Environment (PCTE): Abstract Specification, ECMA-149:* December 1990.

[ECMA158 91] European Computer Manufacturers Association. *Portable Common Tool Environment (PCTE): C Programming Language Binding, ECMA-158.*: June 1991.

[ECMA162 91] European Computer Manufacturers Association. *Portable Common Tool Environment (PCTE): Ada Programming Language Binding, ECMA-162.*: December 1991.

[EURAC 87] NATO IEPG TA13. *Requirements and Design Criteria for Tool Support Interfaces (EURAC):* July 1987.

[Grangé 91] Grangé, Jean-Loise and Obbink, Henk. *An Overview of the ATMOSPHERE Project.* Cap Gemini Innovation/Phillips: March, 1991.

[IRAC 91] Tri-Service Group on Communications and Electronics (TSGCE) Special Working Group (SWG) on Ada Programming Support Environments. *International Requirements and Design Criteria (IRAC) for the Portable Common Interface Set (PCIS), Final Draft Version 0.2:* November 1991.

[Harrison 92] Harrison, William; Ossher, Harold; and Kavianpour, Mansour. "Integrating Coarse-Grained and Fine-Grained Tool Integration," *Proceedings of the Fifth International Workshop on Computer-Aided Software Engineering, Montreal, Canada, July 1992*. Los Alamitos, CA: IEEE Computer Society Press, 1992. 23-35.

[Kelter 91] Kelter, Udo. *Supporting Fine-Grained Data in PCTE.* Hagen, Denmark: University of Hagen, Department of Computer Science, April 1991.

[Kelter 92] Kelter, Udo*. H-PCTE — A High-Performance Object Management System for System Development Environments.* Hagen, Denmark: University of Hagen, Department of Computer Science, February 1992.

[Mercurio 90] Mercurio,V.J; Meyers,B.F; Nisbet,A.M.; and Radin,G. “AD/Cycle Strategy and Architecture,” *IBM Systems Journal* 29, 2*:* 170-188.

[NIST_ECMA 91] National Institute of Standards and Technology and the European Computer Manufacturers Association. *A Reference Model for Frameworks of Software Engineering Environments. NIST Report SP 500-201/ECMA Report TR/55 (Version 2)*. Gaithersburg, MD: December 1991.

[Ochuodho 92] Ochuodho, S.J. “Object-Oriented Database Support for Software Project Management Environments: Data Modeling Issues,” *Information and Software Technology*, 34, 5 (May 1992): 283-307.

[Oliver 91] Oliver, H. “Adding Control Integration to PCTE,” *Software Development Environments and CASE Technology, European Symposium, Königswinter, Germany, June 1991*. Berlin, Germany: Springer-Verlag, 1991. 69-80.

[PCTE1.5 88] Commission of the European Communities*. PCTE, A Basis for a Portable Common Tool Environment, Functional Specification Version 1.5.* Brussels, Belgium: 1988.

[PCTE+ 88] NATO IEPG TA13. *PCTE+ Ada and C Functional Specifications, Issue 3.*: October 1988.

[RAC 86] Ada Joint Program Office, United States Department of Defense. *Requirements and Design Criteria for the Common APSE Interface Set (CAIS):* October 1986.

[Reiss 90] Reiss, Steven P. “Interacting With the FIELD Environment,” *Software Practice and Experience* 20, S1 (June 1990): 89-115.

[STONEMAN] Buxton, John N. *STONEMAN: Requirements for Ada Programming Support Environments*. Washington, DC: U.S. Department of Defense, 1980.

[Tedd 89] Tedd, Mike. “PCTE+ — The Evolution of PCTE,” *Software Engineering Environments: Research and Practice*. Chichester, UK.: Ellis Horwood, 1989. 143-150.

[Thomas 89] Thomas, Ian. “PCTE Interfaces: Supporting Tools in Software Engineering Environments.” *IEEE Software*, 6, 6 (Nov 1989): 15-23.

[Thomas 92] Thomas, Ian and Nejmeh, Brian A. “Definitions of Tool Integration for Environments.” *IEEE Software*, 9, 2 (Mar 1992): 29-35.

[Wasserman 90] Wasserman, Anthony. “Tool Integration in Software Engineering Environments,” *Software Engineering Environments: Proceedings of the International Workshop on Environments, September 18-20, 1989,Chinon, France*. Berlin, Germany: Springer-Verlag, 1990. 137-149.

[Wallnau 91] Wallnau, Kurt and Feiler, Peter. *Tool Integration and Environment Architectures* (CMU/SEI-91-TR-11, ADA237810). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. May 1991.

[Waltham 88] Ada Joint Program Office, United States Department of Defense. *Comparison of CAIS-A and PCTE+, The Waltham Report:* June 1988.

[Winnersh 89] Ada Joint Program Office, United States Department of Defense. *PCTE+ and CAIS-A Convergence, The Winnersh Report*: November 1989.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
Unclassified

1b. RESTRICTIVE MARKINGS
None

2a. SECURITY CLASSIFICATION AUTHORITY
N/A

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for Public Release
Distribution Unlimited

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE
N/A

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
CMU/SEI-93-TR-1

5. MONITORING ORGANIZATION REPORT NUMBER(S)
ESC-TR-93-175

6a. NAME OF PERFORMING ORGANIZATION
Software Engineering Institute

6b. OFFICE SYMBOL (if applicable)
SEI

7a. NAME OF MONITORING ORGANIZATION
SEI Joint Program Office

6c. ADDRESS (city, state, and zip code)
Carnegie Mellon University
Pittsburgh PA 15213

7b. ADDRESS (city, state, and zip code)
ESC/AVS
Hanscom Air Force Base, MA 01731

8a. NAME OFFUNDING/SPONSORING ORGANIZATION
SEI Joint Program Office

8b. OFFICE SYMBOL (if applicable)
ESC/AVS

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
F1962890C0003

8c. ADDRESS (city, state, and zip code))
Carnegie Mellon University
Pittsburgh PA 15213

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO	PROJECT NO.	TASK NO	WORK UNIT NO.
63756E	N/A	N/A	N/A

11. TITLE (Include Security Classification)
An Overview of PCTE: A Basis for a Portable Common Tool Environment

12. PERSONAL AUTHOR(S)
Fred Long Ed Morris

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (year, month, day)	15. PAGE COUNT
Final	FROM TO	March 1993	57

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB. GR.

18. SUBJECT TERMS (continue on reverse of necessary and identify by block number)
PCTE, framework technologies, data management, portable common tool environment

19. ABSTRACT (continue on reverse if necessary and identify by block number)

Environment framework technologies are generating enormous interest as mechanisms to aid in the construction of software engineering environments populated with integrated CASE tools. PCTE, a framework technology, is generating significant interest among environment users and builders as a mechanism for data management. This report details the history and current status of PCTE and PCTE-based environments. The major capabilities of PCTE are identified, and the techniques for integration of tools into PCTE are discussed.

(please turn over)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED ■ SAME AS RPT □ DTIC USERS ■

21. ABSTRACT SECURITY CLASSIFICATION
Unclassified, Unlimited Distribution

22a. NAME OF RESPONSIBLE INDIVIDUAL
Thomas R. Miller, Lt Col, USAF

22b. TELEPHONE NUMBER (include area code)
(412) 268-7631

22c. OFFICE SYMBOL
ESC/AVS (SEI)

ABSTRACT — continued from page one, block 19