"A GENERIC ARCHITECTURE FOR COMPUTER INTEGRATED MANUFACTURING SOFTWARE BASED ON THE PRODUCT DATA EXCHANGE SPECIFICATION"

James E. Fowler

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards
and Technology
National Engineering Laboratory
Center for Manufacturing Engineering
Factory Automation Systems Division
Machine Intelligence Group
Gaithersburg, MD 20899

September 1989

Table of Contents

List of Figures

NATIONAL

PDES

TESTBED

September 20, 1989

A Generic Architecture for Computer Integrated Manufacturing Software Based on the Product Data Exchange Specification

1.0 Introduction

The Product Data Exchange Specification (PDES) is an emerging standard that is intended to address the problems of data exchange and representation for a variety of manufacturing enterprises. The National Institute of Standards and Technology (NIST) has a long-standing research program that addresses the problems of integration and development of automated manufacturing systems. This document presents a software architecture that incorporates PDES into the software applications that are part of NIST's work in Computer Integrated Manufacturing (CIM).

1.1 Background

The Automated Manufacturing Research Facility (AMRF) at NIST is an environment for the development, implementation, and testing of CIM concepts [Simpson82]. One aspect of the work that has been performed as part of the AMRF project is the sharing of data between the individual systems that contribute to the production of mechanical parts. The AMRF Part Model [Hopp87] was conceived

as a data representation that specified an unambiguous format for description of individual mechanical parts.

While the AMRF Part Model provided for the exchange of part data between systems within the AMRF, it was never intended to be a universal solution to the problems of product data exchange. It was developed as a short term solution to a long term problem that was not being satisfactorily addressed by the existing data exchange standards of that time: notably the Initial Graphics Exchange Specification (IGES) [Smith88a]. As the shortcomings of IGES for the purpose of data exchange in an automated environment became more apparent to industry, an international standards effort was initiated to address the problems of product data representation and exchange. In the United States, the initial results of this effort have recently been published as a working draft: PDES Version 1.0 [Smith88b]. The PDES document has reached draft proposal status in the International Organization for Standardization (ISO), where it is unofficially known as the Standard for the Exchange of Product Model Data (STEP).

NIST is playing an integral role in both the coordination of the various international voluntary committees that develop PDES and in the actual testing and validation of the specification. Simultaneously, CIM research continues at the AMRF and has surpassed the capabilities that the AMRF Part Model was designed to fulfill. As a result, there is an immediate need to incorporate the capabilities of PDES into the various systems of the AMRF. Doing so not only advances the possibilities for research in CIM in the AMRF, but also serves to assist in the test and validation of PDES.

1.2 Motivation

The motivation for the design of this architecture is the need to meet the multiple challenges of incorporating PDES into the AMRF. One of the most formidable challenges results from the instability of the developing standard. PDES is still very much in its formative stages, and therefore will change in the coming years. The challenge is to create an architecture that is easily adaptable to changes in PDES, in particular, one that does not require a major software recoding effort for every change in PDES.

Another challenge results from the size of the draft: currently it is over 2000 pages long. No single person can possibly be an expert in every aspect of the standard. An application developer may have expert knowledge in a few of the subject areas contained in PDES,

but is not usually an expert in the techniques and terminology used to represent that subject area in the standard. The goal is to insulate application developers from the details of PDES and its implementation, yet provide the application with the capabilities it requires.

Finally, inasmuch as PDES will be a single standard that encompasses a myriad of application areas, we desire a single architecture based on the standard that satisfies the needs of related but different software applications used in conjunction at the AMRF. Thus the architecture has to be generic to the environment of the AMRF (i.e., automated manufacturing of mechanical parts), and not specific to say, design or inspection applications. The architecture presented here is intended to meet each of these challenges.
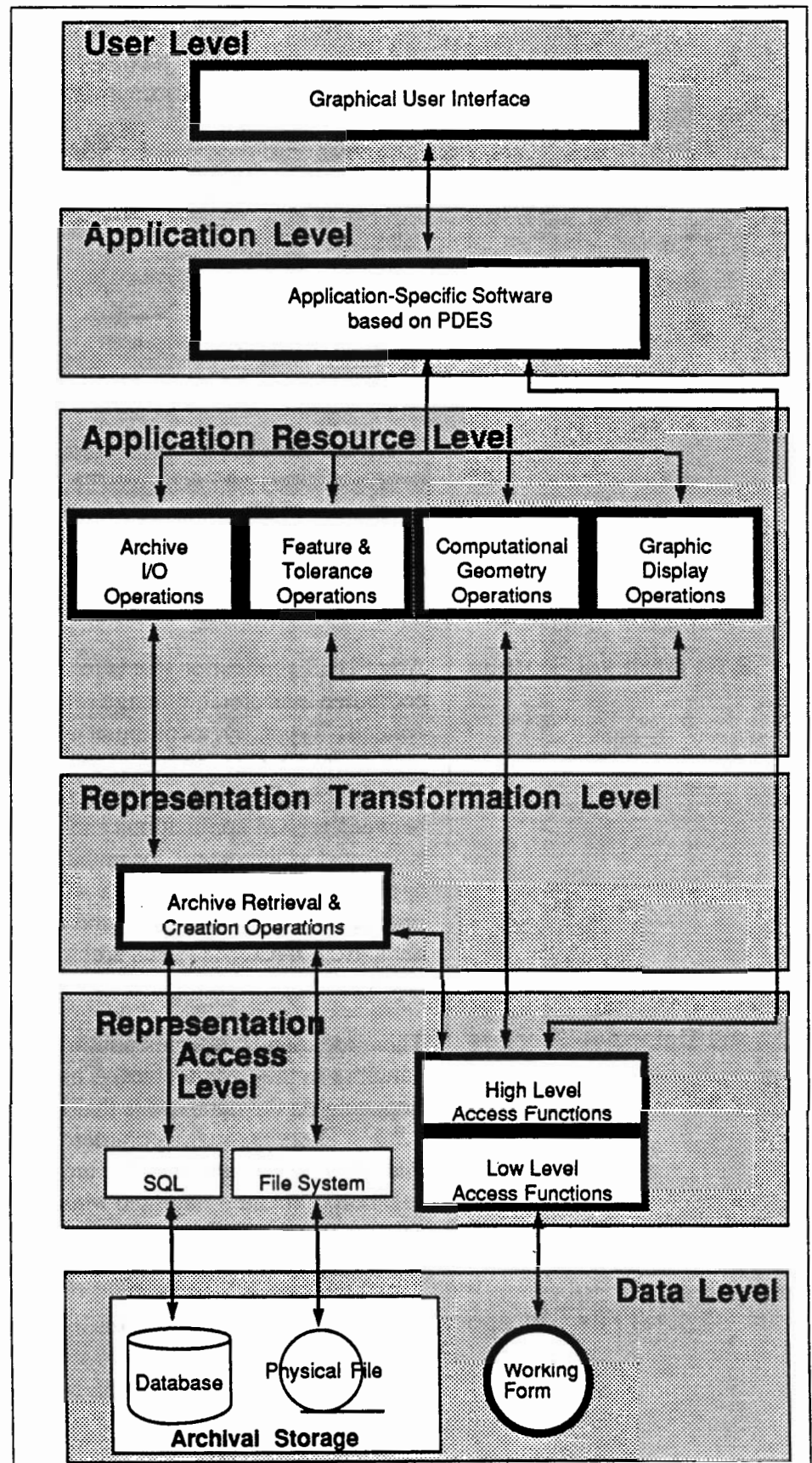
2.0 Description of Architecture

The architecture shown in Figure 1 is structured into a multilayer design. The design provides increasing functionality from the bottom layer, which consists of PDES data representations, to the top layer, where the user interacts with an application. The layers are logically divided by functionality, with higher layers building upon the functions of lower layers. The layered structure serves to insulate higher level functions from the details of low level representations and operations, while at the same time increasing maintainability and providing for expandability.

An extremely important detail of the software that deals with PDES is not shown in the architecture. This is because it is essentially invisible to applications and users, thus it is not of direct concern to high level developers. The function of this software is to read the actual PDES information model[1] specification which is written in the Express[2] modeling language [Schenck89]. From the information model specification, the software automatically creates the definitions for PDES data structures; thus allowing these data structures to be automatically updated whenever the PDES standard changes. Additionally, other highly useful software is automatically created from the PDES information model: this includes the Low Level Access Functions and parsers that are implicit in the Archive Creation and Retrieval Operations (see Figure 1). The software that performs these functions is known as FedEx; see [Clark89] for further information.

The following sections describe each component of the architecture and how these components relate. Details of the actual implementations are not the focus here. Instead the aim is to show what functionality is provided and how this functionality meets the challenge of incorporating PDES into the AMRF.

---

1. An *information model* is a formal description of a collection of abstract concepts (universe of discourse) which composes all of the ideas to be used by systems.

2. The *Express modeling language* is a computer-sensible language that was conceived as a mechanism for defining data objects, the behavior of data objects, and the relationships between data objects in PDES.

Figure 1: Application Architecture

User Level

Graphical User Interface

Application Level

Application-Specific Software
based on PDES

Application Resource Level

| Archive I/O Operations | Feature & Tolerance Operations | Computational Geometry Operations | Graphic Display Operations |

Representation Transformation Level

Archive Retrieval &
Creation Operations

Representation Access Level

High Level
Access Functions

Low Level
Access Functions

SQL

File System

Data Level

Database

Physical File

Working Form

Archival Storage
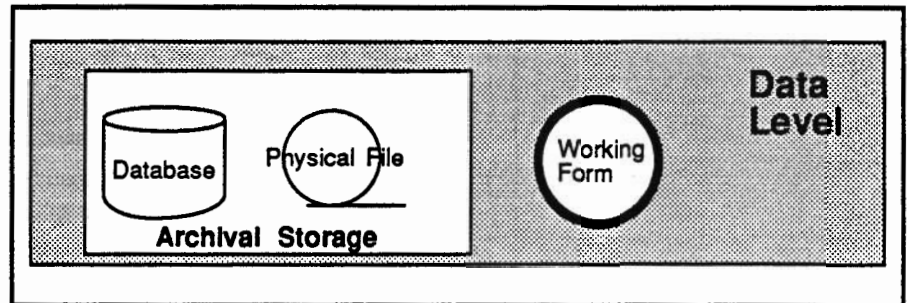
2.1 Data Level

The lowest level of the architecture is the data level. The data level constitutes the actual computer representations of PDES models.

Figure 2: Representations in the Data Level

There are three possible manifestations of PDES models: a database, a neutral exchange file, and memory resident data structures. Each of these data representations are equivalent in their PDES representational capabilities, however their use in PDES-based application software differs considerably. The three representations are described in the following sections.

2.1.1 Archival Storage

Any PDES application needs to be able to create a permanent, computer-interpretable image of the PDES information it has created. Typically, a computer-readable data file residing on disk or tape storage media is used as the means for saving data. In environments where large amounts of data will be shared frequently between related applications and systems, a database can be employed as the means for maintaining permanent records of application data. Since PDES will be used both as a mechanism for product exchange (via files) and as a structure for a shared database [DACOM87] this architecture is intended to support both types of archival storage.

2.1.1.1 Database Storage

There are many strategies possible for the implementation of a database in this architecture. The basic requirement we are imposing on the database is that it be SQL-compliant (see section 2.2.1). There are many commercially available database products that could be employed in the architecture. It will also be possible to incorporate the Integrated Manufacturing Data Administration System (IMDAS) [Libes88] distributed database system developed at NIST.

2.1.1.2 File Storage

There are several mechanisms by which PDES model data could be represented in a neutral file format. The file could be an ASCII text file, a binary file, an encoded/compressed file, and so on. Currently,

the PDES working draft calls for an ASCII format file. The mapping between a PDES information model and a PDES neutral exchange file (shown as a physical file in Figure 2) is specified in [Altemueller88].

2.1.2 Working Form

While the database and physical file representations are used as archival storage mechanisms, a data representation that resides in computer memory is needed to support efficient access to PDES data by an application[1]. When an application is invoked using PDES data stored in one of the archives, the working form data structures will contain an equivalent representation of the data that was stored in the archive. As the application creates new data or modifies the original data, the modifications will exist only in the working form: the contents of the working form progressively diverge from the existing contents of the archive. The contents of the working form are archived on demand from the application or user. Thus, consistency between archival storage and the working form is not dynamically maintained; they are consistent initially when the working form is loaded and thereafter as required by the application and/or user.

Consistency between working forms and archival storage mediums is by itself a non-trivial issue. If we insist that an application has exclusive use of archival storage (a "single-user" environment), then consistency can be maintained by locking the archives until the application terminates. While simplifying the consistency issue, this restriction is not particularly desirable in a manufacturing environment; i.e. more than one application may require access to a given PDES model at any given time (a "multi-user" environment). Providing the mechanisms to support a multi-use environment is a function that will be addressed in the implementation of the architecture.

---

1. In future architectures, it may be necessary to use the database as the working form due to the potentially large size of PDES models. This will require highly efficient database implementations and interfaces so as not to degrade application performance.

2.2 Representation Access Level

The Representation Access Level comprises the functions that serve as the interfaces to the PDES data representations.

Figure 3: Interfaces to PDES Data Representations
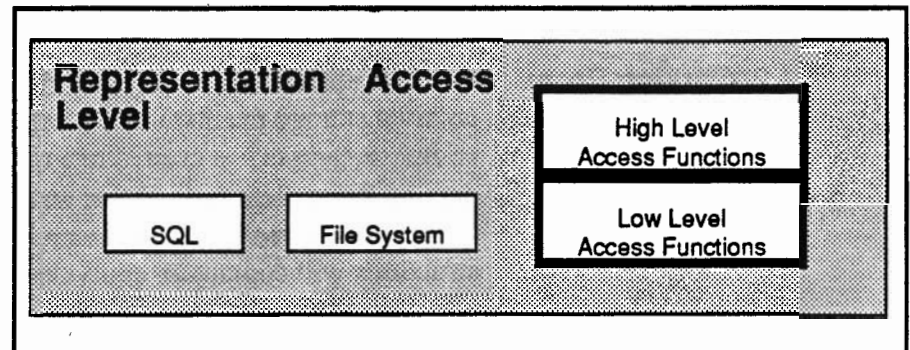
Functions at higher levels in the architecture retrieve PDES data through these interfaces. Each data representation requires its own particular interface mechanism. The interfaces are described in the following sections.

2.2.1 SQL Interface

The Structured Query Language (SQL) defines a set of facilities for defining, manipulating, and controlling data in a relational database [Date87]. SQL has become an internationally adopted standard as a procedural interface to relational databases [ANSI86]. Adopting the SQL interface allows any SQL-compliant database product to be incorporated into the architecture. The IMDAS database currently used in the AMRF provides an SQL interface.

2.2.2 File System

The file system is the interface to a physical file. For a given computer system, the file system is operating system dependent. Nonetheless, we can rely on the file system to allow for the opening, closing, creation and deletion of files as well as providing structured reading/writing capabilities of the contents of those files. Additionally, high level programming languages (e.g. C, Lisp, etc.) provide system-independent access to the file system, yielding a ready-made interface to physical files.

2.2.3 Working Form Interface

The interface to the working form data structures is actually comprised of two levels of interface functions: the high and low level access functions. The motivation for separating the working form access functions in this way is two-fold. First, the low level functions are arcane and thus require that the application developer have intimate knowledge of the structure of the PDES information model. Insulating the application developer from the details of PDES requires a second layer of functions that group low level

function calls into data abstractions that the user is more accustomed to. The second motivating factor for the two sets of functions is that the low level functions are independent of the exact contents of the PDES information model. Therefore, as the PDES information model evolves over time, the low level functions need not be updated; however, the high level functions do. Details of the implementation of both levels of functions will be described in a forthcoming document.
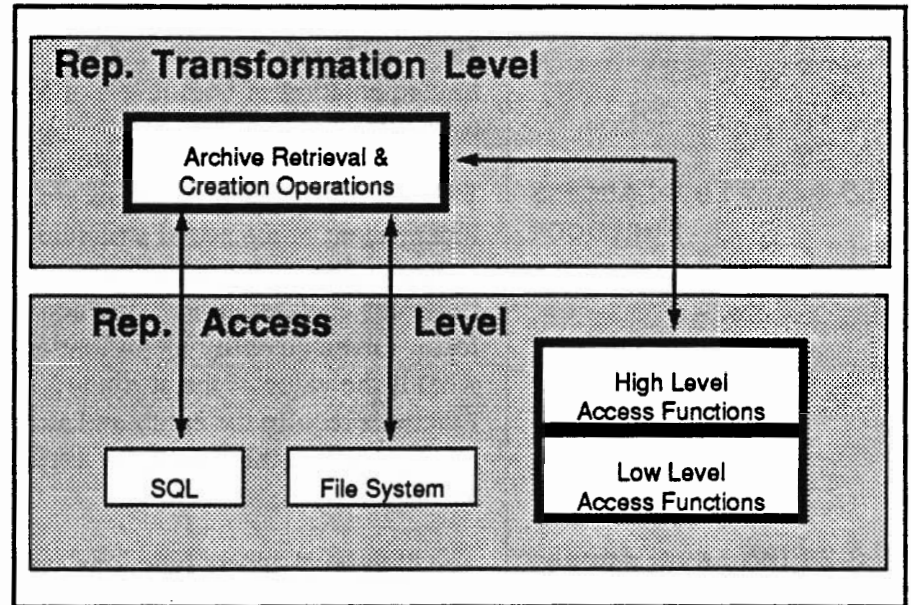
2.2.3.1 Low Level Access Functions

The low level access functions are designed to be very generic and independent of the actual structure of the PDES information model. They are few in number and principally come in two flavors: query functions and assignment functions. The query functions are of the form "Given an entity identification and an attribute identification, what is the value of the attribute?" Conversely, the assignment functions require the entity and attribute identifications and an attribute value, then cause the attribute to be assigned the given value.

2.2.3.2 High Level Access Functions

The high level access functions are designed to provide an abstract interface to the detailed information found in the PDES information model. As in the lower level functions, the high level functions are grouped into dual categories: query and assignment functions. The high level query functions answer questions of the form "What are the center vector, normal vector, and radius of the circle with this identifier?" or "What are the identifiers of the edges bounding the face with this identifier?" The assignment functions, of course, provide the converse operation. The nature of these high level functions makes their implementation dependent upon the structure of the PDES information model; this is a drawback. Considering that the number of functions useful at this level is unbounded, the maintenance of these functions will be a burden as PDES evolves. We have not yet begun to look at the possibility of automatically updating these functions as PDES evolves; it is a subject for future research.

2.3 Representation Transformation Level

Figure 4: Interface between Working Form and Archives

The Representation Transformation Level encompasses the functions that exchange data between the archival storage representations and the working form data structures. These


functions are known collectively as the Archive Retrieval & Creation Operations (see Figure 4). Since there are two archival storage formats, there are two principal pieces in the Archive Operations. One piece deals with the functions which exchange data between the working form and the database through the SQL interface. The other piece deals with the exchange between the working form and the physical file through the File System interface. This latter piece makes use of a parser and an associated report generator. Both of these pieces provide the same facilities for higher level functions in the architecture: that is, *save working form to destination* (where destination is either the database or a physical file), and *load working form from source* (again either the database or a physical file). A by-product of the architecture is that it is straightforward to create a physical file from the contents of the database and to load the database from a physical file.

2.4 Application Resource Level

The Application Resource Level implements the functions that are considered useful across a broad spectrum of PDES-based applications. Functions at this level are able to make calls to functions at lower levels of the architecture as well as to functions at the same level. For example, the Feature & Tolerance

Figure 5: Function Modules within the Application Resource Level
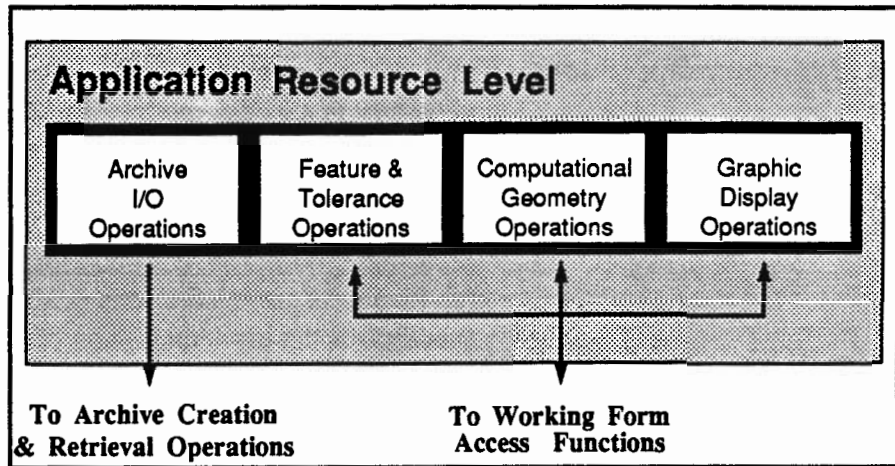
Operations, the Computational Geometry Operations, and the Graphic Display Operations all rely on the working form access functions to supply an interface to the PDES data. The application resource level is not limited to the modules shown in Figure 5; development of greater numbers of applications based on PDES will necessitate expanded functionality in this level of the architecture. This level of the architecture should be viewed as a software library of functions that are to be used as resources for developing specific applications.

2.4.1 Archive I/O Operations

The Archive I/O Operations module implements a high level resource for the purpose of loading/saving PDES models from/to the different archival formats. This module relies on the lower level Creation and Retrieval Operations to perform the detailed transformation between archival storage and the memory resident working form data structures. It would be possible for an application program to call the lower level archive routines directly; however doing so makes the application program vulnerable to future changes that will undoubtedly take place in the mapping between PDES, the working form, and the archival storage formats. Using the archive routines at the resource level isolates an application from these considerations.

2.4.2 Feature & Tolerance Operations

The Feature and Tolerance Operations module implements a high level resource for the purpose of manipulating mechanical form features and tolerance data in a PDES model. The module provides an application with facilities for editing the relationships, groupings, and data necessary to define PDES features and tolerances without having to be aware of the details of the entity structures.

2.4.3 Computational Geometry Operations

The Computational Geometry Operations module implements a high level resource for the purpose of creating and manipulating geometry in a PDES model. The module provides an application with facilities for establishing geometry entities, for creating geometry entities based on computations on existing geometric entities, for computing the results of geometric questions, and for performing other related functions. Applications which make use of the functions provided in this module do not need to know the specifics of the implementation of PDES entities.

2.4.4 Graphic Display Operations

The Graphic Display Operations module implements a high level resource for the purpose of producing and manipulating graphical visualizations of PDES models as well as the tools for building graphical user interfaces. Applications can invoke the utilities provided in this module to produce a user interface of their own design and graphic displays such as are commonly found on modern bit-mapped computer workstations. Depending upon the complexity of the display desired, applications may need to employ functions from the computational geometry operations module for display generation. Applications may also need to devise mappings from abstract PDES entity types to the graphics data required for displays.

2.5 Application Level

The Application Level is the level at which application-specific software resides. By application-specific software, we mean software that is intended to address a specific task, such as design or process planning, and that intends to use PDES as the fundamental data representation. With regard to the work that is

Figure 6: Functions available to software at the Application Level
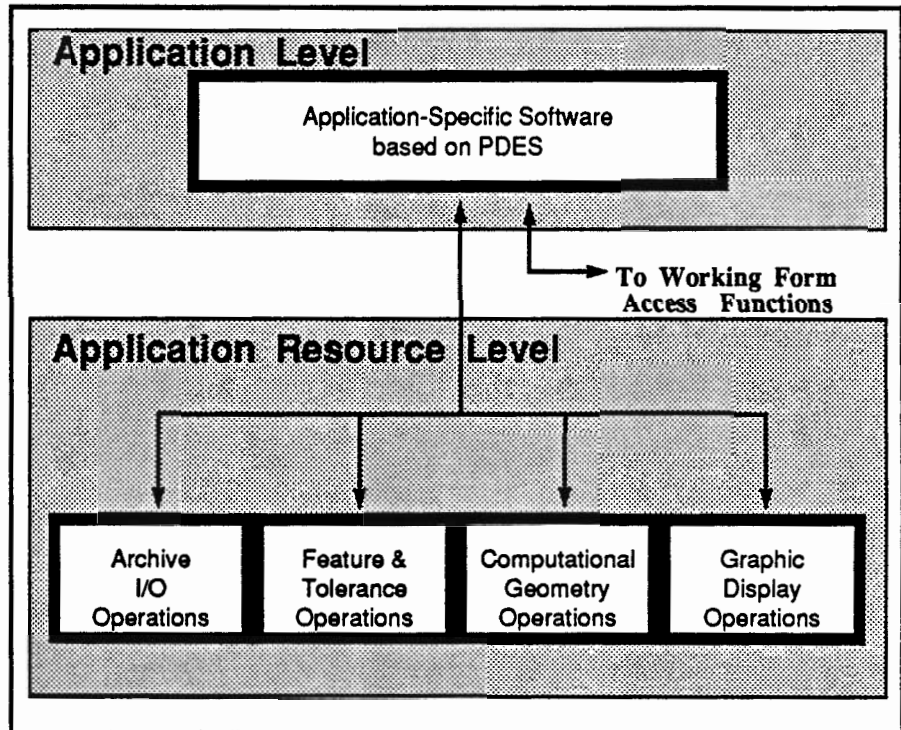
currently underway in the AMRF, the applications that are migrating toward the use of PDES include mechanical part design, process planning, and inspection planning. These three applications in particular will be implemented at the Application level of this architecture. As our PDES-based implementations evolve, other AMRF applications, such as Equipment Programming, may incorporate the architecture as well.

Software written at the application level has a rich collection of functions available to it for the creation, manipulation, interrogation, and display of PDES model data as well as a number of other highly useful utilities that greatly simplify application development. The architecture thus follows the software engineering paradigm of allowing application developers to concentrate on the task at hand rather than on a variety of background issues. Note that Application-specific software has direct access to the working form

access functions in addition to the functions provided in the
Application Resource Level (see Figure 6). While we would prefer
that the architecture follow the conventions of a layered protocol,
we cannot presume to foresee the resources required by every
application for its particular use of PDES data. Therefore an
application can bypass the Application Resource Level to reach the
data interface directly. Naturally, bypassing the Application
Resource Level when the resources of interest are available there
is discouraged.

2.6 User Level

The User Level is the level at which a human user interacts with the application software residing at the Application Level. This level is included in the architecture only for completeness; that is, we are not specifying how the user interface should work[1] or even in fact whether applications are required to provide a user interface. The aspects of a user interface are left entirely up to the application-specific software. Regardless, provision is made in the Application

Figure 7: Sample User Interface
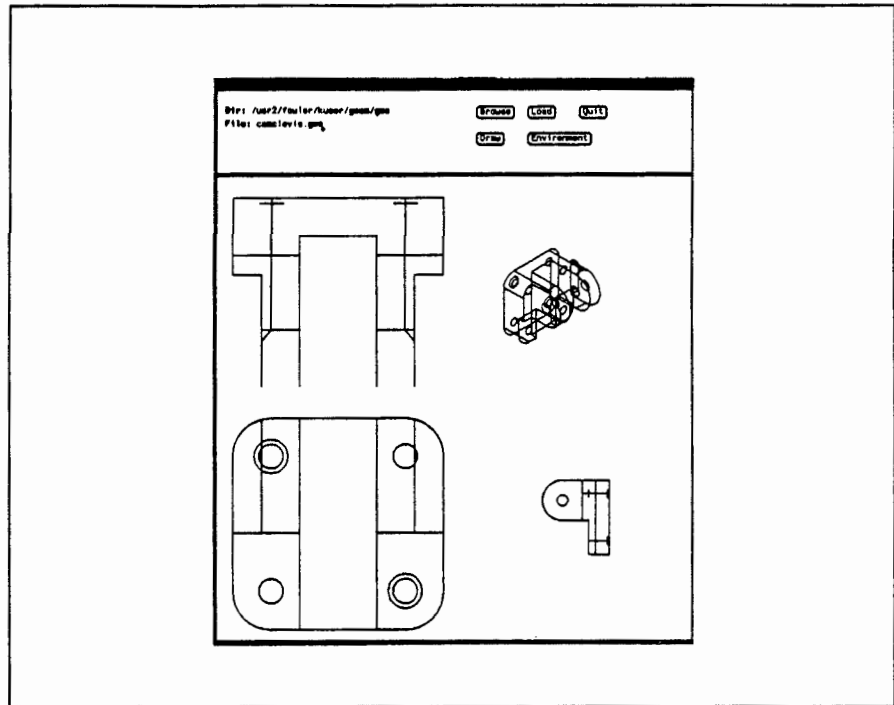
Resource Level to facilitate the creation and maintenance of a contemporary, graphical, window-oriented user interface.

---

1. The subject of user interface guidelines has been discussed in AMRF-related projects. Work in this area was presented at the Manufacturing Data Preparation Project (MDP) Workshop at NIST in June 1988.

3.0 Role of Off-The-Shelf Software

There are three major areas in this architecture where off-the-shelf software can be immediately employed in its implementation. These three areas are the database, the Computational Geometry Operations, and Graphic Display Operations modules in the Application Resource Level. There are no other opportunities to employ off-the-shelf software in the architecture as it has been described simply because no commercially available software exists that is compatible with PDES. Using commercially available software is desirable not only because it speeds implementation of the architecture but also because it ameliorates the transfer of this technology to industry.

There is not much more to say about the aspects of incorporating commercially available database software than has already been covered in previous sections. Using commercially available software in the other two modules does, however, warrant futher discussion.

3.1 Graphics Software

The Graphic Display Operations module is a logical place to incorporate off-the-shelf software. Functions such as creation, manipulation, and maintenance of graphic data and displays are provided in commercially available graphics software libraries. Some of these graphic libraries conform to the Programmer's Hierarchical Interactive Graphics System (PHIGS) [ISO87]. Other graphics libraries provide functionality in the spirit of PHIGS but may not conform precisely to the standard.

Aside from the manipulation of graphics data, the Graphic Display Operations module is intended to provide functions for creating and maintaining a user interface. Software tools useful for user interfaces include mechanisms such as windows, buttons, and menus. PHIGS does not define explicit functions to provide these types of mechanisms, although it is conceivable that these mechanisms could be coded using PHIGS functions, albeit with a significant investment in labor. Fortunately, there are user interface toolkits available, notably X-Windows [Scheifler86].

In the best of all worlds, it would be possible to use a graphics library in conjunction with a user interface toolkit to provide most of the functions desired for the Graphic Display Operations module. Unfortunately, as of this writing, not all graphics libraries cooperate with X-Windows. In the future, we presume that the level of integration between these two highly useful and related software tools will increase (see, for example, PHIGS Extensions to X-

Windows [Rost89]). In the meantime, the module will be implemented with available graphics libraries that do cooperate with X-Windows, but with sufficient flexibility to adapt to future developments in the graphics/interface world.

3.2 Computational Geometry Software

The Computational Geometry Operations module is another candidate for the incorporation of off-the-shelf software. Clearly, all three-dimensional Computer Aided Design (CAD) systems sold commercially incorporate software functions that perform operations such as vector and matrix algebra, compute intersections between curves and surfaces, etc. Additionally, CAD systems that provide solid modeling operations must have embedded software that performs solids calculations such as boolean operations. Many of these geometric modeling functions are specified in the Application Interface Specification (AIS) [CAMI86], which defines a method by which application software can interface to a geometric modeler. Geometric modeling systems that conform to AIS or a similar interface could be efficiently integrated into the Computational Geometry Operations module. The specific requirements for the integration of an off-the-shelf modeling system with this architecture are found in [Fowler89].

4.0 Existing Architectures

At this point, we would like to consider PDES-based architectures that are currently in existence and compare them to the design proposed in this paper. To the author's knowledge, there are few PDES-based systems in existence [IGES/PDES89], and of the known architectures, only one is well documented. This system, GMAP, developed by Pratt and Whitney for the U.S. Air Force, is currently installed and available for use at NIST [Perlotto89].

4.1 GMAP

The Geometric Applications Interface program (GMAP) allows for the creation of PDES product model instances, and for the development and evaluation of application programs based on PDES. A simplified outline of the GMAP architecture is shown in Figure 8.

Figure 8: GMAP System Architecture
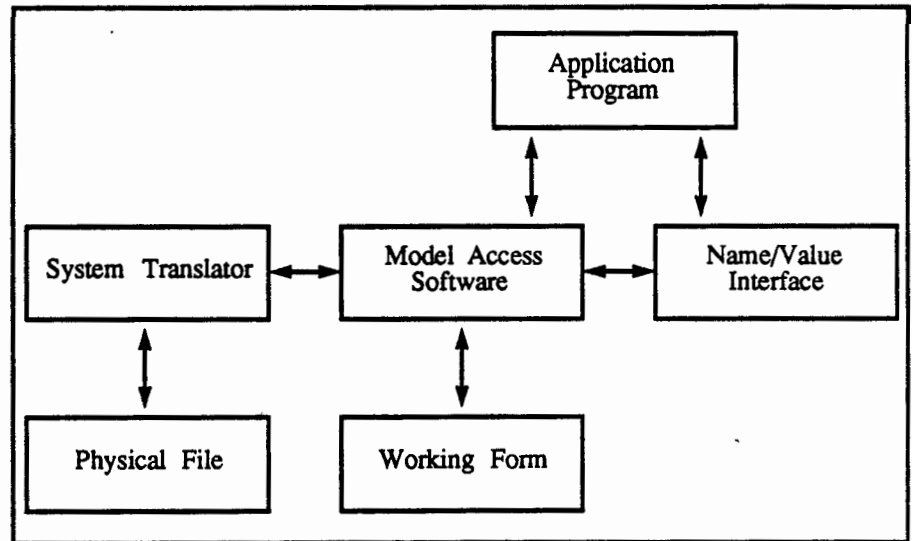
As with the architecture that is the topic of this paper, a significant portion of the GMAP system is not shown above: the Express parser and its associated software, known as the Schema Manager. The Schema Manager has the role of defining the structure, type, and access of data in the GMAP system based on information parsed from the Express representation of PDES.

The functionality of the GMAP system compares to that of the NIST architecture up to the Representation Transformation Level, with one exception: the GMAP system lacks an interface to a database. Thus, the NIST architecture and the GMAP system share similar functionality for translating a physical file into a working form and back again. Both architectures also allow for creating, interrogating, and navigating through the model represented in the

working form. However, the combination of Model Access Software and Name/Value Interface is contained within the Low Level Access Functions of the NIST system. The High Level Functions have no counterpart in the GMAP system.

From an implementation standpoint, the GMAP system is implemented in the Pascal programming language. The NIST system will be realized through the use of SQL, the C programming language, and will be accessible to LISP-based applications. Additionally, we do not wish to preclude the migration of the architecture to an object-oriented implementation; moving from a C implementation to a C++ implementation is a natural transition.

5.0 Conclusion

This document has presented the design of a software architecture that provides for the efficient integration of PDES in the AMRF. A variety of automated mechanical part manufacturing related software applications can be created at the highest level of the architecture, making use of the software tools at lower levels in the architecture to expedite the development of task solutions. High level application developers are largely insulated from the details of handling a complicated and continuously evolving data standard. Finally, by allowing for the expeditious development of applications based on PDES, we are in turn promoting the development of the standard by providing a realistic environment for the test and validation of the standard.

Appendix A: References

[Altemueller88]   Altemueller, J., "Mapping From Express to Physical File Structure", ISO TC184/SC4/WG1 Document 4.2.2, March 1988.

[ANSI86]   American National Standards Institute, "Database Language SQL", Document ANSI X3.135-1986.

[CAMI86]   Computer Aided Manufacturing International Inc., "Application Interface Specification", Vol's. I & II, Arlington, Texas, 1986.

[Clark89]   Clark, S., "FedEx: The NIST Express Parser", National Institute of Standards and Technology, Gaithersburg, MD, May 1989 Draft.

[Date87]   Date, C., "A Guide to the SQL Standard", Addison-Wesley, November 1987.

[DACOM87]   D. Appleton Company, "Development Plan for a Product Data Exchange Specification - PDES Version 1.0", Report No. D-669-87-03, Submitted to South Carolina Research Authority July 8, 1987.

[Fowler89]   Fowler, J., "Requirements for a Solid Modeling Software System in a PDES-based Architecture", National Institute of Standards and Technology, Gaithersburg, MD, August 1989 Draft.

[Hopp87]   Hopp, T.H., and Tu, J.S., "Part Geometry Data in the AMRF", NBSIR 87-3551, National Institute of Standards and Technology, Gaithersburg, MD, April 1987.

[Libes88]   Libes, D., and Barkmeyer, E., "The Integrated Manufacturing Data Administration System (IMDAS) - An Overview", Computer Integrated Manufacturing, Vol. 1., No. 1, pp. 44-49, 1988.

[IGES/PDES89]   Oral Presentations by John Zimmerman (Allied-Signal Aerospace Co.) and by Chia Hui Shih (SDRC) on PDES Implementations, General Session, Joint IGES/PDES/ISO Meeting, San Antonio, Texas, April 12, 1989.

[ISO87]   International Standards Organization, "Programmer's Hierarchical Interactive Graphics System", Draft Standard ISO dp9592-1:1987(E), Geneva, Oct. 1987.

[Perlotto89]   Perlotto, K.L., "The Use of GMAP Software as a PDES Environment in the National PDES Testbed Project", NISTIR 89-4117, National Institute of Standards and Technology, Gaithersburg, MD, June 1989.

[Rost89]   Rost, R.J., Friedberg, J.D., and Nishimoto, P.L., "PEX: A Network-Transparent 3D Graphics System", IEEE Computer Graphics & Applications, Vol. 9, No. 4, July 1989.

[Scheifler86]   Scheifler, R.W., and Gettys, J., "The X Window System", ACM Trans. on Graphics, Vol. 5, No. 2, April 1986.

[Schenck89]   Schenck, D., ed., "Information Modeling Language Express: Language Reference Manual", ISO TC184/SC4/WG1 Document N362, May 1989.

[Simpson82]   Simpson, J., Hocken, R., and Albus, J., "The Automated Manufacturing Research Facility of the National Bureau of Standards", Journal of Manufacturing Systems, Vol. 1, April 1982.

[Smith88a]   Smith, B., Rinaudot, G.R., Reed, K.A., and Wright, T., "Initial Graphics Exchange (IGES) Version 4.0", NBSIR 88-3813, National Institute of Standards and Technology, Gaithersburg, MD, June 1988.

[Smith88b]   Smith, B., and Rinaudot, G., eds., "Product Data
              Exchange Specification", NISTIR 88-4004,
              National Institute of Standards and Technology,
              Gaithersburg, MD, December 1988.

4. TITLE AND SUBTITLE

A Generic Architecture for Computer Integrated Manufacturing Software Based on the Product Data Exchange Specification

5. AUTHOR(S)

James E. Fowler

6. PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
GAITHERSBURG, MD 20899

7. CONTRACT/GRANT NUMBER

8. TYPE OF REPORT AND PERIOD COVERED

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)

10. SUPPLEMENTARY NOTES

☐ DOCUMENT DESCRIBES A COMPUTER PROGRAM; SF-185, FIPS SOFTWARE SUMMARY, IS ATTACHED.

11. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

The Product Data Exchange Specification (PDES) is an emerging standard that is intended to address the problems of data exchange and representation for a variety of manufacturing enterprises. The National Institute of Standards and Technology (NIST) has a long-standing research program that addresses the problems of integration and development of automated manufacturing systems. This document presents a software architecture that forms the basis for the incorporation of PDES into the software applications that are part of NIST's work in Computer Integrated Manufacturing (CIM).

12. KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)

AMRF, CAD, Computer Aided Inspection, Computer Integrated Manufacturing (CIM), Database, Factory Automation, IMDAS, PDES, Product Data Exchange, Process Planning.

13. AVAILABILITY

☒ UNLIMITED

☐ FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).

☐ ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.

☒ ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES

28

15. PRICE

A03

ELECTRONIC FORM