GRAPHIC STANDARDS: IGES AND PDES IN AN AEC ENVIRONMENT

James A. Turner
Architecture and Planning Research Laboratory
The University of Michigan
2000 Bonisteel Blvd.
Ann Arbor, MI 48109

ABSTRACT

The idea made a lot of sense: many diverse CAD systems communicating a common project data-base through a neutral format translator. The "Initial Graphics Exchange Specification", kindly known as IGES (pronounced "I guess" by its proponents, and "I guess not" by its opponents) was the the initial effort, and is either loved or hated; there is no "neutral" ground. Has it succeeded? Has it failed? Is there a future in this neutral format business? Was CAD meant to be "design" or "drafting"? Does industry support it? What does it mean for architecture? Is a "one-to-many" translator a wonderful idea, but impossible to implement? Is a complete set of "one-to-one" translators a better idea?

This paper will give a short history of IGES, discuss its reason for being, list its strengths and weaknesses, examine its inner workings, and introduce the current effort of the IGES committee: a total "Product Design Exchange Specification", PDES (and internationally as STEP). It will also discuss the techniques used by the PDES application committees to model their various products, and give a case study of the effort of the AEC committee in modeling an architectural "product".

The paper will conclude with the opinions on the future of IGES by the author (a four year member of the IGES/PDES organization).

THE NEED FOR STANDARDS

The scenario was familiar for most early developers of architectural computer programs: After a course or two in Fortran or Basic (often audited), or a few evenings perusing one of the few available texts, the more analytical faculty member or student began to see the "big picture" - computers would change the practice of architecture. As a demonstration of that fact we choose an area to attack, such as structures, energy, acoustics, or graphics, and began writing code.

Along the way we stumbled through the (then undefined) stages of program development such as algorithm, data-structure, data-base, and user-interaction design, implementation, and testing. We had few models to work from and, by default, introduced ourselves to the concepts of iteration, subprograms, stacks, linked-lists, networks, tables, computer graphics, and non-graphics interaction. We became "serial" thinkers and experts in the nuances of the resident compilers and interpreters (and the local card reader).

Most of the programs took on a similar flow: The program user had to define the application specific characteristics of a building (location and dimensions) in a specified format, usually by preparing a deck of cards or by typing lines into a file. This was an error-prone process of, for the geometric description, labeling and computing the coordinates of each point, and ordering sets of points for each room, beam, wall, etc. This set of data that we now know as the
"project-dependent" data-base had to be entered for each "run" of the program. The same was true for the "project-independent data-bases" such as steel beam data, luminaire characteristics, system costs, U values, average rainfall tables, etc. Luckily cards could be duplicated, and files could be copied, so it wasn't much of a problem. Besides, most users had done the same analysis by hand (and still do), and appreciated the speed and accuracy they were now able to achieve. This was the architecture of the future: How could we complain?

The complaining began soon after a collection of analysis programs became available, perhaps written by the same local expert, or programming "guru". The discerning student might have at his command programs to analyze both the heat loss/gain of a building and the luminaire demands of each of its rooms. If graphic hardware was available, perspectives and parallel projections might also be drawn. It began to become tedious: Define a building for an energy analysis; define it again for a lighting analysis; and, once more for pictures. The iteration involved in "debugging" the project dependent data-base was usually long, and only the most die-hard user would repeat the process, for the same building, in a different format for each analysis. Many users even complained about having to use output from one program (size of steel members) as input to another (cost estimating). There were many long nights hunched over a drawing (instructors still made us draw) labeling points, sweating over a teletype punching keys, and standing in line waiting for output.

Because of these problems the concepts of "shared data-bases" and "integrated data-bases" evolved, and became an important and necessary consideration in any architectural CAD endeavor. It is impossible these days for a commercial product to be an independent entity, ignorant of its competitors. Users are educated, well versed in CAD jargon and the current state of its development. They demand that all design software share project dependent data between similar systems, and between dissimilar systems. Even smaller AEC offices own two or more CAD systems which are often employed on the same projects, and they will soon demand that project independent data be available on existing and future systems.

![Diagram](image)

**Figure 1:** The relationship between similar and dissimilar systems and project dependent and independent data-bases.

**DATA EXCHANGE**

The problem of data-base sharing was a popular topic after the initial enthusiasm for computer graphics and architectural CAD wore off. But, like documentation, it is a very practical necessity, not as exciting as colored lines on a screen or instantaneous results of a tedious computation. As a result, the amount of early data exchange work done, especially in the AEC community, was small. Not until, (1) commercial vendors developed and began pushing their own standards, and (2) the National Bureau of Standards organized and directed the development of a national CAD data exchange standard, was anything put "on paper".

196
The problem of complete AEC data exchange is a difficult one. The life of an AEC project takes on many phases: Building program, Pre-design, Concept design, Detail design, Construction documentation, Construction, Facility maintenance, Redesign, Demolition.

Each of these phases has its own special data-base needs, along with sharing common data-bases with other phases. For instance, a complete geometric description of space is known from concept design to demolition. Fairly exact schedules of equipment, doors, windows, and wall finishes are known from detail design to construction. And, use, ownership, furniture, and maintenance schedules are necessary for facility maintenance. To complicate matters, the early phases, building program and pre-design, have more incomplete and difficult to categorize and exchange "design intent" data. It is no wonder that the early efforts were in the exchange of a small portion of this data: the exchange of graphics, specifically construction drawings. The format which reached national standard status was IGES.

INITIAL GRAPHICS EXCHANGE SPECIFICATION: IGES
The IGES organization began as an offshoot of a data exchange project initiated by the U.S. Air Force, the Integrated Computer-Aided Manufacturing Project (ICAM) [2], which aimed at the exchange of data between aircraft manufacturers. Other pioneering contributors were Boeing Corp., the General Electric Corp., and the NASA-sponsored Integrated Program for Aerospace Vehicle Design (IPAD) [5]. The initial goal of IGES was to define a neutral file format for the exchange of data between state-of-the-art, commercially available CAD/CAM systems. It was a hefty task and the committee soon settled into the problems of exchanging drafting data, but in a framework they hoped would eventually support the exchange of product definition and design data, as well (and the other phases of an AEC project).

The hope of the committee was to develop a national consensus standard for an "indirect translator" (figure 2) that would eventually replace the forecasted exponential growth of "direct translators" (figure 3). During the initial effort the number of CAD/CAM companies were growing rapidly, and the problem of pairing systems with one-to-one, direct translators was staggering.

![Diagram](image.png)

Figure 2: The indirect translator approach would only take 2 * N translators: One translator from system A's data-base to the neutral file format, and one translator from the neutral file format into system A's data-base.

197
A BRIEF HISTORY OF IGES

The idea to create such a standard surfaced at the Department of Defense Manufacturing Technology Advisory Group in Detroit in 1979. The IGES concept was discussed at a meeting attended by representatives of the Air Force, Army, Navy, NASA, (all four provided initial funding) and the National Bureau of Standards; and various presentations were made by suppliers, system designers, and standards groups. It was decided at that meeting to name the effort "IGES", and to use the Boeing CAD/CAM Integrated Information Network, and General Electric's Neutral Data Base projects as prototypes. A Technical Committee was formed, and the project was under way [18,24,28].

After two public meetings in November and December, 1979, the first IGES specification was published in January, 1980, and, in May, recommended by ANSI Subcommittee Y14.26 to be part of a proposed national standard.

In September, 1981, IGES Version 1.0 was approved as ANSI standard Y14.26M, and in February, 1983, IGES Version 2.0 was published by NBS. These two initial documents were widely distributed, and represent the state-of-the art of most current implementations. There was much testing, both commercially, and by IGES authors, after version 1.0 was published. IGES put itself up for public criticism by having Booz-Allen Hamilton, Inc. analyze and formally report on the testing of ten vendor's implementations. This period marked what I call phase one of IGES: A standard for drawing data exchange was "on the street", being implemented, tested, and criticized. It would live or die on its own merits.

In April, 1986, IGES Version 3.0 was published by NBS, and included significant editing changes, but minor content additions. The next significant milestone in the IGES history was May, 1984, when an ad hoc (special project) committee was formed to begin the definition of a full product data exchange specification.

THE IGES ORGANIZATION

IGES is composed of a chairman, a secretary, two types of committees, and various special project groups [7]. Technical committees deal with the technical development and modification of the standard. These are, for the most part, groups who meet because of a common interest in an application area. Their goal is to lobby, through a formal RFC (Request For Change), CO (Change Order), and RP (Recommended Practice) procedure, for additions and modifications to the standard. Technical committees in the past have been in the areas of finite element, electrical, drafting, technical publication, mechanical, manufacturing, and AEC. The chairman is usually the spokesman for the committee.
The administration committees consist of the Steering committee, the Technical Planning committee, and the Edit committee. The Steering committee sets goals and strategies and provides for policy and procedure for IGES activities. It consists mainly of members of supporting corporations and government organizations. The Technical Planning committee advises the chairman on technical and planning issues related to technical activities. Members of the committee include the Edit committee chairman and current project managers.

The Edit committee’s function is to coordinate the development and integration of IGES products, mainly the published standard and other technical publications. The committee is composed of a chairman, the chairman of the Technical committees, and current project managers.

Occasionally a task will be assigned as a project to a non-partisan group. The group will be organized by a project manager to attack a non-application specific project. Current projects are the Experimental Solids Proposal, which looked at supporting CSG and B-Rep representations of solid models, and PDES, an investigation into complete product data exchange.

A TYPICAL IGES MEETING
Regular IGES meetings are held quarterly, usually in January, April, July, and October, while the various administration committees and special project groups meet more often. We try to follow the sun and move around the country accordingly. We never meet in Chicago in the winter or in Tempe in the summer. We try to schedule the meetings in a hotel large enough to board the participants, and to provide us with a large meeting room and several smaller “break-out” rooms. IGES has over 600 members, but a typical meeting will be attended by only 150 - 180.

The meeting lasts for a whole week, starting early on Monday and ending early on Friday. Monday is usually scheduled for the Edit committee, with general registration and an orientation for new members in the evening. There has been a large turnover of new members in the last few years, and an orientation is necessary (28).

On Tuesday through Thursday mornings the General Assembly, which consists of all members, meets. This is a time for general gab and presentations by various vendors, committees, lobbyists (although they are not introduced as such), and various “experts” who attempt to teach us how to do our job better. There is also a good amount of “pep talk” to energize us for our afternoon committee work. The afternoons, or late mornings, are spent in technical committee meetings. We divide up into the smaller rooms, renew old acquaintances, and get down to business.

THE AEC TECHNICAL COMMITTEE IN AN IGES CONTEXT
The AEC committee has been in existence since February, 1984, and is one of the most active and populated of all committees (25). It averages about 15 - 20 members at the quarterly meetings, with most of them being long-term, dedicated supporters. Their charge, like most technical committees, is to influence the standard: to make additions which will support the exchange of AEC specific data; and to ensure that future versions of the standard support the data exchange problems of future AEC systems.

The makeup of the committee is interesting. Unlike other committees who have only electrical or only finite element members, the AEC committee has become a melting pot for a variety of quasi-related disciplines. They have adopted architects, geographers, process plant designers, building system engineers (mechanical, structural, etc.), and ship builders. (Are ships floating buildings? Or are buildings static ships?)

The committee’s main impact on the standard has been in the areas of non-drafting data. They feel
that IGES, as a drafting, or graphic, standard is quite good, and will support most AEC-type drawings. AEC drawings tend to have more lines, notes, and dimensions than other disciplines, but there is nothing unique in the elements of the drawings: AEC drawings can be made from the set of IGES entities. The committee has, instead, attempted to add new entities which would support the transfer of tabular data (relations, rectangular data-sets), network data (links, nodes, and associated attributes), and new geometric entities such as enclosed areas (IGES is currently ill-defined in its specification of polygon data, or groups of polygon data), and sets of tangent enclosed areas (such as found in DIME files) [29,30].

The proposed tabular data entity has been accepted into the standard and will appear in version 4.0. (This was after three years of discussion, invention, review, and politicking.) It was only after joining forces with the (powerful, respected) Electrical committee that it was accepted. The collection of proposals, letters, reports, and ballot comments would make a nice primer for future proposal writers. AEC committee members sometimes view themselves as a "bastard child" of the IGES organization: It's all right to come to the meetings, but don't expect to be acknowledged.

PRODUCT DATA EXCHANGE SPECIFICATION: PDES

It was always the goal of the IGES organization to provide a standard which would allow the exchange of drafting, design, and definition data. Drafting support was attacked first, partially because it is the easiest to conceptualize, and partly because one of the goals of the committee was to support data exchange capabilities as soon as possible to current CAD/CAM systems, which were mostly drafting-based.

As more systems became design oriented, their internal data-bases began to contain more non-graphic data. Products were modeled; not just as 2-dimensional drafted projections of real objects, but as the real objects, with well-defined 3-dimensional geometries and, in most cases, associated non-geometric data. This was the evolution of CAD/CAM systems from drafting to modeling to design, and IGES moved to respond to it.

PDES is a research project of IGES [19]. With this move IGES has graduated from a fast-paced, "get it down on paper so we can use it" standards organization, to a slower paced research group. Rather quickly the technical committees shifted their emphasis from the production of the next IGES document to the problem of carefully analyzing and formally describing their products. In a drafting exchange standard disciplines could share entities, but would that be in a product data exchange standard? Before that question could be answered the disciplines had to determine which entities supported their products.

The goal of the PDES effort is to produce a standard. Unlike the initial IGES effort, the PDES group has mandated a very organized, three layer approach, consistent with the ANSI/SPARC 3-Schema architecture [9]. Each product of each discipline must be described, or modeled, using a formal graphical notation, with supporting text describing a "universe of discourse" (the definitions and relationships) of the products. These "application layer" models will be submitted to a "logical layer" committee who will extract entities and determine which are unique to the discipline and which are general, or can be applied to two or more disciplines (to hopefully cut down on the number of unique entities). When this task is completed the logical layer committee will submit their collection of conceptual entities to the "physical layer" committee, who will determine appropriate file formats. The three layers are known as: application, logical, and physical; or external, conceptual, and internal. It is hoped that the passing of models from the top level to the bottom level will constitute a reduction in entities to a manageable (and implementable) size.
THE AEC TECHNICAL COMMITTEE IN A PDES CONTEXT
During the PDES initiation effort the AEC committee spent a great deal of time discussing AEC projects as "products". It's one thing to think of a printed circuit board or a mechanical part as a product, and quite another to think of a building, or a ship as a single product that one could model and present to the logical layer committee.

The first task was to learn a notational language for describing a product. There were two suggested choices: IDEFIX and NIAM [3,4,15,17,22,23,29,33]. Both are popular graphic-based, conceptual modeling (sometimes called reference modeling, enterprise modeling, or information modeling) techniques used in a data-base design environment, a very organized and consistent method for describing a complex enterprise for the purpose of extracting the fields, records, and tables of a data-base. The goal was not to design a data-base, but to understand an AEC product, and to communicate that understanding to a committee unfamiliar with the product.

Both methods begin with a "universe of discourse" - the text describing the objects and the relationships and constraints between the objects. For an enterprise such as a dental clinic, this might involve interviewing each employee, and observing the daily operation. An example of a portion of a universe of discourse for buildings might be [34]:

To provide a flat, horizontal surface on which desired human activities can take place, all buildings contain at least one floor. In primitive buildings, the ground may be used as the floor. In better buildings, the floor may be a deck laid on the ground or supported above ground on structural members, such as the joist indicated in Fig. ...

The next step is to determine the objects. For the example they are: surface, activities, buildings, floor, ground, deck, members, and joist. Once the objects are known the relationships become obvious: to provide, on which, take place, contain, used as, may be, laid on, supported above, and on. It is also necessary to determine "subtypes", such as structural and human, and "cardinalities", such as all, at least, and may be.

Once a universe of discourse is approved its objects and relationships can be interpreted and presented graphically:

![Diagram of NIAM modeling notation](image)

Figure 4: Example of the NIAM modeling notation

The process of agreeing on a universe of discourse for an AEC product (which the committee is just beginning to understand) was an impossible task. Instead, they began by graphically modeling a building, and a general HVAC system [27,32]. This is still one of the main tasks and, for a variety of reasons, will never be completed. The first reason is that an AEC product is too complex, too varied, and has too many phases to be completed in the allotted time. The second reason is that the AEC committee, unlike other committees, does not want application-specific
entities. For instance, the committee does not want the logical layer committee to study their models and determine that a "brick entity" and "door-swing entity" is necessary to support the transfer of AEC product data. They feel that if enough generic entities for transferring objects, attributes, values, relationships, and constraints, perhaps in the form of single values, tuples, tables, networks, and hierarchies (along with the IGES-like drafting and geometric data), were provided, a large percent of AEC product data could be transferred. The AEC PDES effort has produced a large collection of models which highlight that need.

THE IGES FILE STRUCTURE
Like most popular CAD/CAM systems the IGES file format is based on entities: geometric, annotation, and structure [6,21]. Geometric entities consist of objects like points, lines, arcs, splines, surfaces, solid primitives, CSG trees, and B-rep solids. Annotation entities consist of textual information like dimensions and notes. Structure entities allow one to combine other entities into an aggregate entity such as groups and subfigure definition and instances.

An IGES file is organized into five distinct sections: start (S), global (G), directory entry, (D) parameter data (P), and terminate (T). The start, global, and terminate sections contain mostly administration data (name of file, author, units of measurement, count of each type of record, etc.). The directory entry and parameter data sections contain the entity data. The directory entry fields contain general information about an instance of an entity, its line color, line weight, level, etc., and the parameter data records contain the physical data associated with the instance, x,y,z coordinates, radius, start point, end point, etc. A sample IGES file is shown in figure 5.

CONCLUSION: WILL IGES SURVIVE?
I conclude with a list of conditions necessary for IGES to survive and become a viable national (and possibly international) consensus standard. PDES is a much longer term venture, and has already been successful in understanding a problem which may never have a (complete) solution. After each condition I have included a personal observation and conclusion.

1) There will have to be a need for a CAD data-base exchange standard.

There will always be a need for data exchange and, therefore, always a need for a standard, commercial or consensus. IGES can fulfill this requirement in the short-term.

2) The number of CAD/CAM systems must remain high.

The number of vendors claiming to support AEC activities remains high, yet most offices using CAD choose from only a few of the top products. If this continues over the next few years (of vendor "shake-out"), there may evolve only three or four choices (similar to the evolution of American automobile manufacturers) - direct translators would be more advantageous, and the need for IGES would disappear.

3) The IGES file format will have to be compact and efficient, generating files similar in size to those of the native system's internal data-base files.

The fact that all IGES files must be formatted as 80 column ASCII records means that IGES files are typically very large; usually much larger than the resident system's internal data-base file. A proposal for a more compact format was accepted into the standard a few years ago, but few, if any, vendors support it. This is a serious problem.

The fact that each entity instance must be preceded by a two-record directory entry means that most IGES translators must make two passes - similar to most C compilers -
Figure 5: Sample IGES File.
once for definitions, and once for instances. Often they must be written with complicated linked list structures because of the liberal nesting and referencing allowed.

4) The IGES organization must grow to include a good portion of the commercial vendors, and major users.

Currently, there is only minimum attendance by vendors at the quarterly meetings. One reason for this is that most of the large vendors have developed their own standards for communicating with their systems. For instance, Intergraph Corporation has developed its Intergraph Standard Interchange Format (ISIF), and AutoCAD has developed its DXP format. These standards tend to be much simpler to understand and implement, and more compact, than IGES.

5) The IGES organization must allow for the democratic and swift evolution and modification of the standard.

The turn around time for a moderate modification to the standard is long - at least 9 months (and possibly much longer in the case of a major entity). This time must be reduced. There is also a movement to eliminate further IGES work so that PDES can push forward. The committee must realize that they are two different standards, and must continue to evolve simultaneously.

6) The standard must evolve within the context of developing international standards.

There is quite a bit of similar international work being done. IGES and PDES must develop in accord with STEP (European), SET (French), and DIN (German). STEP and PDES have, at least in the minds of some, formed a merger. Some of the meetings are held together, and many of the long-term participants have been members of both groups.

7) The standard must go through the normal ANSI acceptance procedures.

Vendors need a user-approved standard.

8) The standard must lend itself to conventional implementation.

One of the largest groups at a typical IGES quarterly meeting is the implementors: those responsible for making additions and modifications to their product’s translators. They give valuable comments and guidance to RFC developers.

9) The standard must not be dictated by a single concern: commercial, military, or industrial.

The IGES organization is very political, similar to the US congress: changes to the standard are proposed, discussions are held, and a vote is taken; and lobbyists are everywhere. IGES began as a small project sponsored in part by the military, and its influence is still felt: We are often entertained by high-ranking computer-aided-soldiers who claim that the next world war will be fought via neural files.

10) IGES must not become too large: It must know what its charge is and stay within those limits.

The IGES standard is currently a good neutral format for storing drafting data, but is muddied with extremely application-specific entities. For instance, entity 406, the Property entity, has different forms for storing “drilled hole” information, “pin
numbers", and the results of finite element calculations. The latter form takes no less than 32 pages and is mis-named "Tabular Data". These entities belong in PDES.

Application-specific uses of IGES entities belong in "Implementor's Guides", or "Recommended Practice Guides", not in a drafting standard.

REFERENCES
2. ICAM Program Prospectus, Update B, September, 1979.
29. Thompson, Paul, "Natural Language Analysis, Information Modeling and Database Engineering", Control Data Corporation, Minneapolis, MN.