Solid Model Numerical Representation: An Emerging Skill for Engineering Graphics Students

Cameron W. Coates[1], Kam Fui Lau[2], Michael Brown[3]

Abstract – In this paper, an IGES Parser Utility program has been developed that enhances the ability of an engineering student to interpret solid model data and use this data to understand how engineering graphic components are represented by numerical data files and their transferability among different CAD programs. The program is designed such that it can be used consistently throughout the Engineering Graphics course with several topics within the course. It is also designed to integrate easily with web based and other digital media applications. The background, objectives, architecture and justification for the application are described and implementation plans are proposed.

Keywords: Engineering Graphics, IGES file, Solid Model, CAD

INTRODUCTION

The majority of engineering students who take Engineering Graphics are provided limited exposure to the various ways 3D information can be numerically captured and transferred. Boundary Representation and Constructive Solid Geometry are typically discussed in a few classes however this type of information is often quickly forgotten as the students progress throughout their Computer Aided Design (CAD) or Computer Aided Manufacturing (CAM) course. As the breadth and depth of modeling packages rapidly increase, understanding numerical representation of 3D data and their applicability to exporting or importing data will become more important. Engineering students should not only know the various file formats available for file transfer, but also be able to understand and manipulate these formats in order to combine capabilities of the various modeling packages.

Since the explosive development of software packages tailored towards CAD/CAM, various neutral data formats have also been designed in order to allow for the digital representation and exchange of product definition data. Two of the more prominent systems are the Initial Graphics Exchange Specification (IGES) format and the Standard for the Exchange of Product model data (STEP) format. While the STEP file format is more recent and may allow for more information to be captured and transferred [Lockhart et al, 1], both file types have similar data exchange architectures and are capable of exporting wireframe, surface or solid model information. The IGES file is an ASCII file that uses numerical representation for Engineering Graphics components. In this paper, we will focus on classroom instruction for IGES interpretation, however the program developed can be modified with little effort in order to accommodate the interpretation of STEP files as well. When CAD/CAM files are converted from one program to an IGES format and then interpreted by another program, data is often lost or misinterpreted. Additionally, the designer's intent is often lost, as the exchanged data do not incorporate details such as sketches, constraints and features. Researchers have addressed this problem in several ways. Dori [2] introduced a scheme that employed elements of machine vision and geometric analysis that served to augment the domain of the "IGES function". This scheme provided the originating system's human readable output as an additional possible source to

[1] Associate Professor, Engineering Studies Program, Armstrong Atlantic State University, Savannah GA.

[2] Associate Professor, Information Technology Dept. , Armstrong Atlantic State University, Savannah GA.

[3] Chief Technology Officer, OpenVision Inc., Hilton Head, SC.

be viewed. Grabowski and Glatz [3] designed an IGES Model Comparison System (IMCOS) format that was able to detect loss of information and functionality resulting from model exchange via IGES processors. More recently, Bionconi [4] provided a five-year review (2006) of various other solutions that have been proposed. While many of these solutions are highly effective, there is not a general consensus on a standard solution or set of solutions that should be taught within the Accreditation Board of Engineering and Technology (ABET) approved Engineering Graphics curriculum. As the complexity and number of CAD packages increase, companies want engineering graduates who are able to understand, manage and troubleshoot communication between these packages. Branoff [5] surveyed several engineering design companies in 2001 to determine the types of skills that applicants would need to secure a position doing constraint-based modeling. This author concluded that companies want graduates who can identify and fix problems in 3-D geometry, use powerful knowledge-based systems to design complex assemblies, and be flexible enough to do design and development work. Cumberland [6] also performed a similar survey of twenty eight companies and concluded that data translation, file and data management, constraint-based solid modeling, web technologies, simulation, animation and a study of current trends and issues were among their essential requirements for engineering graduates. Consequently, it is imperative that Engineering Graphics students, at a minimum, understand how the more popular neutral file formats translate CAD files and what the current issues are regarding static data exchange, one-way data exchange, redundancy and information loss. Students should therefore learn the fundamentals of the interpretation process so that the transferable mechanism is not treated as a "black box", and they will be able to effectively manage and troubleshoot these systems.

The Engineering Graphics curriculum is already quite full and there is not much time available within a one-semester course for exploration of file transferable mechanisms. The efficiency with which IGES file essentials are taught can be increased with the use of interpretive software. For example, if an IGES parser utility can be developed that is able to interpret the various numerical elements and provide the geometric equivalencies; this can be added to a multimedia presentation, a web-based presentation or a digital tutorial system. Additionally, the ability to explain what information gets lost in the file transfer process and why, would be greatly enhanced. In this paper, the authors describe a novel software application that provides geometric interpretation of the IGES file; this IGES parser utility allows for immediate feedback to the Engineering Graphics students and should serve as an efficient instructional tool in understanding the fundamentals of neutral file formats.

OVERVIEW OF THE IGES PARSER APPLICATION

The National Bureau of Standards published its *Digital Representation for Communication of Product Definition Data* (The Initial Graphics Exchange Specification (IGES)) in January 1980 [USPRO, 7]. The goal of IGES is to facilitate the exchange of engineering drawing among different engineering graphics programs.

In the Armstrong Atlantic State University Engineering Studies program, we use SolidWorks CAD software in our Engineering Graphics course. SolidWorks files are stored as binary files for two conditions; one file is for the parts document (.SLDPRT) and the other for assembly information (.SLDASM) for a multi-part drawing. However, the SolidWorks preprocessor is also able to convert its files to many formats, including the IGES file format, which is the industry standard ASCII file for engineering drawing.

The IGES Parser Utility is a 32-bit Microsoft Windows® based software application designed to help engineering students understand and appreciate the numerical representation of engineering graphics elements in the IGES file. The design of the software can also serve to support an introductory module on CAD/CAM systems for Information Technology and Computer Science students. For these students, such a module may strengthen their awareness and appreciation for other fields where Information Technology applications are prominent (such as Engineering). Similarly, the software can be used to increase the awareness of the engineering student of the importance of information technology functions such as file reading and parsing, that they might normally take for granted. Special focus is given to designing the program so that it is user friendly without sacrificing efficiency and functionality. The IGES Parser Utiility window is shown in Figure 1. The data is provided in tabular form such that the entity designation, value and description are distributed among three columns.

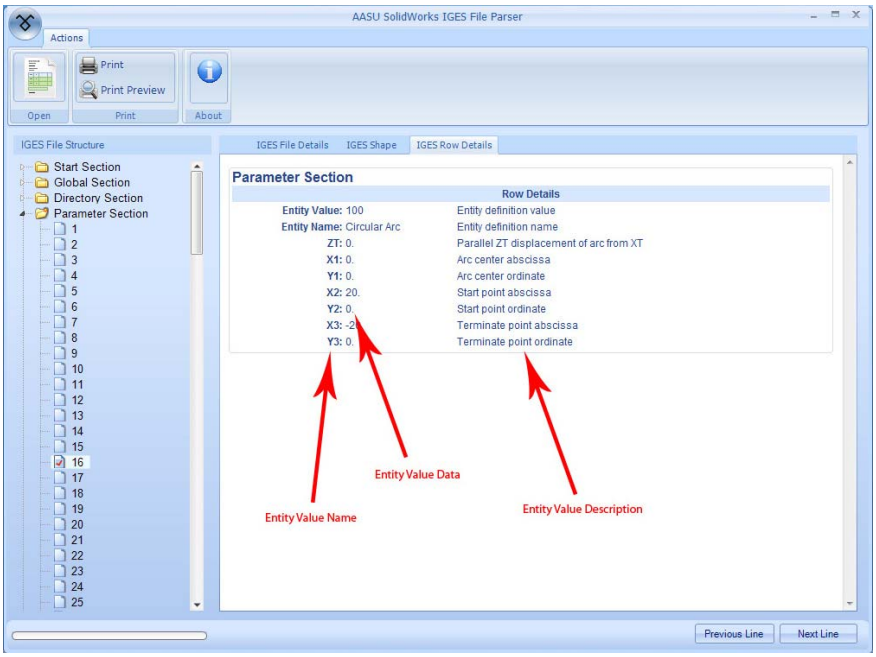

Figure 1 *IGES Parser Utility*

The functional purpose of the software application is to parse and display in a readable format various sections and data contained in an IGES (Initial Graphics Exchange Specification) file. Primary functionality of the software application includes but is not limited to:

- File open dialogs
- Tree view structure of IGES file sections and data
- Web-page style summary of IGES file sections
- Web-page style detail of IGES file section data
- 3D CAD viewer for representation of the object
- Print and print preview dialogs

IGES File Structure

The fundamental unit of data in the file is the entity. Entities are categorized as geometrical and non-geometrical. Geometrical entities represent the definition of the physical shape and include points, curves, surfaces, solids, and relations that are collections of similarly structured entities. Non-geometrical entities typically serve to enrich the model by providing a viewing perspective in which a planar drawing may be composed and by providing annotation and dimensioning appropriate to the drawing. Non-geometrical entities further serve to provide specific attributes or characteristics for individual or groups of entities and to provide definitions and instances for groupings of entities. The definitions of these groupings may reside in another file. Typical non-geometrical entities for drawing definition, annotation, and dimensioning are the view, drawing, general note, witness line, and leader. Typical non-geometrical entities for attributes and groupings are the property and associativity entities.

An IGES file consists of 5 sections: Start, Global, Directory Entry, Parameter Data, and Terminate. The file may include any number of entities of any type as required to represent the product definition. Each entity occurrence

consists of a directory entry and a parameter data entry. The directory entry provides an index and includes descriptive attributes about the data. The parameter data provides the specific entity definition. The directory data are organized in fixed fields and are consistent for all entities to provide simple access to frequently used descriptive data. The parameter data are entity-specific and are variable in length and format. The directory data and parameter data for all entities in the file are organized into separate sections, with pointers providing bi-directional links between the directory entry and parameter data for each entity. The specification provides for groupings whose definitions will be found in a file other than the one in which they are used.

Software Platform and Development Tools

Microsoft Windows® is the selected platform for application development and deployment as it is widely used across most university computing laboratories. The application is designed to run on Microsoft Windows® XP, Vista, Server 2000, Server 2003, and Server 2008.

Minimum Requirements:

- Microsoft Windows® XP, Vista, Server 2000, Server 2003, and Server 2008
- Microsoft .NET framework 3.5
- 30MB hard drive space
- 1Ghz or faster processor
- 512MB ram
- Compatible 3-D graphics card

Development tools used to create the IGES Parser Utility software application are a combination of programming and graphic tools. The programming language used to develop the application is C# (pronounced “c-sharp”).

Development Software:

- Microsoft Visual Studio 2008
- Adobe Photoshop CS3

Microsoft Visual Studio is an Integrated Development Environment (IDE) from Microsoft. It can be used to develop Console and Graphical User Interface (GUI) applications along with Windows Forms applications, web sites, web applications, and web services in both native code together with managed code for all platforms supported by Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework and Microsoft Silverlight. In our case we used Microsoft Visual Studio to aid in the rapid development of a Windows 32-bit Forms Application using managed C#

Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web designer, class designer, and database schema designer. It allows plug-ins to be added that enhance the functionality at almost every level - including adding support for source control systems (like Subversion and Visual SourceSafe) to adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer). The Microsoft Visual Studio IDE is shown in Figure 2.

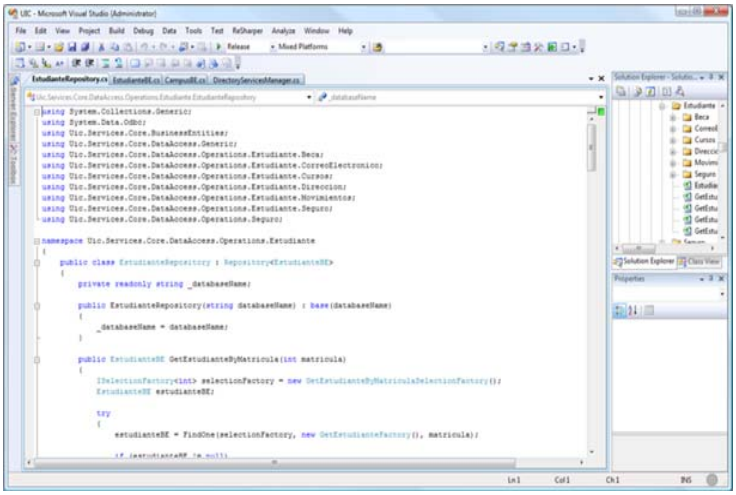

Figure 2 *Microsoft Visual Studio IDE*

Code Architecture

Careful consideration was given to the scalability of the programming code. XML was the data format chosen to encapsulate the parsed file data due to its portability. For example, given the data is packaged in XML format it can be easily distributed or ported to a web application.

Open Cascade is an open-source library used to visualize the shape in the 3D control environment represented in the application interface. Basic functionality is used currently and consists of simple "view-only" visualization within the application interface. This gives the end user the ability to "see" what type of object the software application is representing.

The Directory section organizes and gives structure to the information in the Parameter Data section. There can be only one directory entry for each Parameter Data section entity. Directory section entries may reference other Directory section entries. This would occur when a transformation matrix is specified in order to represent structures. The supported parameter data is the data being communicated. The Table 1 lists some directory entries be supported by the IGES Utility Parser.

The Parameter Data section contains the data that defines the entity. For example, if the entity is a circular arc, you'll find the center, start, and end points, unit normal, and whatever else is required to define it. Entity parameter details are parsed and represented in a line break format. An example of a circular arc data is provided in Table 2. As the program is used for institutional purposes only a few common entity types are fully described to the end user. More definitions may be added in the future as needed.

Figure 3 shows the process flow of the IGES File Parser application. The flow is described from the standpoint of an end user running the application and selecting a valid IGES file from their local computer.

Table 1 Directory Entries that can be supported

Entity Name	**Input/Output**	**Entity Number**
Circular Arc	I/O	100
Composite Curve	I/O	102
Copious Data (Forms 1,2,11, and 12)	I/O	106
Plane (Forms 0 and 1)	I/O	108
Line	I/O	110
Parametric Spline Curve	I/O	112
Parametric Spline Surface	I/O	114
Point	I/O	116
Ruled Surface (Form 1)	I/O	118
Tabulated Cylinder	I/O	122
Transformation Matrix	I/O	124
Rational Bspline Curve	I/O	126
Rational Bspline Surface	I/O	128
Offset Surface	I	140
Boundary Entity	I/O	141
Curve on Parametric Surface Entity	I/O	142
Bounded Surface	I/O	143
Trimmed Surface Entity	I/O	144
Manifold Solid B-Rep Object	I/O	186
Color Definition	I/O	314

Table 2 *Circular Arc Data Example*

Index	Name	Type	Description
1	ZT	Real	Parallel ZT displacement of arc from XT, YT plane
2	X1	Real	Arc center abscissa
3	Y1	Real	Arc center ordinate
4	X2	Real	Start point abscissa
5	Y2	Real	Start point ordinate
6	X3	Real	Terminate point abscissa
7	Y3	Real	Terminate point ordinate

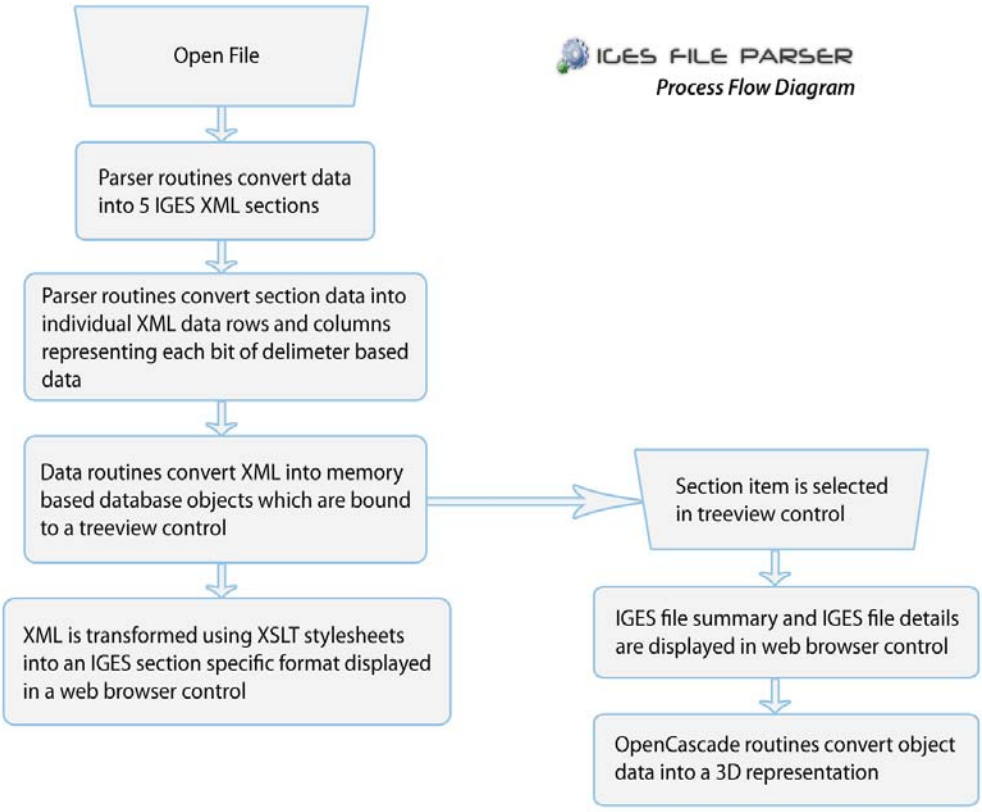

Figure 3 *Software Architecture of IGES Parser Utility*

CONCLUSION

We have successfully created a prototype: The IGES Parser Utility for reading an IGES file created from SolidWorks, and interpreting the numerical representation of each engineering graphics element line by line. The software is low cost and utilizes minimal processing power and memory. It conveniently utilizes open-source software for 3-D visualization and is therefore quite feasible for academic applications. The IGES Parser Utility can serve as an efficient instructional supplement for Engineering Graphics students who are learning the fundamentals of neutral file formats. Additionally, the data packaging is such that it can be easily distributed or ported to a web application. Therefore it should be beneficial for distance based Engineering Graphics courses that rely heavily on the Internet and for non-traditional students who may rely mostly on digital media to pursue their coursework.

Future Work

Future enhancements include the expanding of the dictionary for numerical representations of graphical entries such as Conic Arc, Surface of Revolution, and others. We are currently working on incorporating the graphics Application Programming Interfaces (API's) for displaying the graphics within the application. We are also developing an application to display the corresponding element in the IGES Parser Utility when the user steps through the program codes in the IGES file. Once complete, we intend to perform a controlled study of the educational effectiveness of the IGES Parser Utility within the Engineering Graphics curriculum. Additionally, the question of what data gets lost and why, the discontinuity of design intent when a file is converted to IGES or to similar formats such as STEP and miscommunication issues between CAD packages will be explored through the development of an advanced parser utility program whose basic architecture is derived from the IGES Parser Utility program described in this work.

REFERENCES

[1] Lockhart, S.D., Johnson, C.M. Engineering Design Communication, Prentice Hall New Jersey 2000 pp. 386.

[2] Dori, D., "Enhancing CAD/CAM systems communication by understanding engineering machine drawings," Proceedings of the 17th conference on ACM Annual Computer Science Conference, Louisville, KY 1989 pp. 437.

[3] Grabowski, H., and Glatz, R., "IGES Model Comparison System: A Tool for Testing and Validating IGES Processors," IEEE Computer Graphics and Applications, 7(11), Nov 1987 pp. 47-57.

[4] Bianconi, F, Conti, P., and Di Angelo, L, "Interoperability among CAD/CAM/CAE Systems: A Review of Current Research Trends," Proceedings of the conference on Geometric Modeling and Imaging: New Trends, London, UK, July 2006 pp. 82-89.

[5] Branoff, T. J., Hartman, N.W., and Wiebe, E.N., "Constraint-Based, Solid Modeling: What do Employers Want Our Students to Know?," Engineering Design Graphics Journal, 67(1), 2003 pp. 6-11.

[6] Cumberland, R. R. " The foundation of a progressive engineering graphics curriculum: A directed project report," Unpublished masters thesis, Purdue University, West Lafayette, 2001.

[7] The U.S. Product Data Association (US PRO): Initial Graphics Exchange Specification 5.3. 1997.

Cameron W. Coates

Cameron W. Coates is an Associate Professor of Engineering Studies at Armstrong Atlantic State University in Savannah, GA. Dr. Coates received his PhD in Aerospace Engineering from the Georgia Institute of Technology, Atlanta, GA in 2001. His research interests are Structural Mechanics, Structural Health Monitoring, Finite Element Modeling, and Engineering Education.

Kam Fui Lau

Kam Fui Lau is an Associate Professor and Interim Department Head of Information Technology at Armstrong Atlantic State University, Savannah GA. Dr. Lau received his PhD in Computer Science from the University of Rhode Island, Kingston, RI in 1999. Dr. Lau's research interests are Real Time Systems and Science, Technology, Engineering and Mathematics (STEM) Education.

Michael Brown

Michael Brown is the Chief Technology Oficer for OpenVision, Inc. Mr. Brown is also an undergraduate student in Information Technology at Armstrong Atlantic State University, Savannah GA. Mr. Brown has been developing high-end web and windows applications for the last 10 years with tremendous success. Mr. Brown is a recipient of the Commander's Award for Public Service and the 2006 CBETA Outstanding Achievement in Government Award.