Experience in Development of Translators for AP203 Edition 2 Construction History

Sharon L. Barber[1], Amar Junankar[2], Saurabh Maitra[3], Ganeshram Iyer[4] and Venkat Devarajan[5]

[1]Imagecom, Inc., sharon.barber@aspire3d.com
[2] Imagecom, Inc., amar.junankar@aspire3d.com
[3] Imagecom, Inc., saurabh.maitra@aspire3d.com
[4] Imagecom, Inc., ganeshram.iyer@aspire3d.com
[5]Imagecom, Inc., venkat@aspire3d.com

ABSTRACT

Over the last two decades, the mechanical CAD industry has witnessed the development of many powerful 3D CAD software products built upon proprietary geometric kernals. The ISO 10303 STEP standard was developed, with the Application Protocol (AP 203) in particular, to provide a medium for product data exchange of lower-level geometry of mechanical parts and assemblies. A recent version, AP203 Edition 2, introduced the exchange of product data using a hybrid model containing construction history, GD&T, parameters and other high-level content. This paper presents an interpretation of the schema definitions provided under the current standard, presents a generic approach to map feature history data between CAD systems, discusses an implementation of this approach for data translation, and discusses the issues in the standard realized during this effort.

Keywords: STEP, AP203, construction history, feature tree.
DOI: 10.3722/cadaps.2010.565-578

1 INTRODUCTION

The ISO 10303-203 (Configuration controlled design of mechanical parts and assemblies) is probably the most widely used part of the standard which provides for the exchange of wireframe, surface and boundary representation solid models together with associated administrative data [18]. With the expansion of ISO 10303 standard, (specifically Procedural and Hybrid Representation [10] and Elements for the procedural modeling of solid shapes [12]), more commonly known as STEP AP203 Edition 2 standard, CAD systems can now exchange parametric construction history models. A construction history is primarily a sequence of operations that create shape features [18]. In keeping with the scope of our project and paper, we too provide a definition for construction history: all shape features in the order in which they appear in the feature tree of a CAD system is the construction history.

It should be noted however that no commercial CAD system, fully supports the Edition 2 standard [2], and none export / import the construction history model as prescribed by the STEP AP203 Edition 2 standard.

This paper presents the experiences encountered during the development of translators to export / import construction history models between dissimilar CAD systems. To implement the technology, two recognized 3D feature-based, parametric CAD systems were chosen viz. Pro/Engineer (Wildfire 3.0 M90) by PTC and SolidWorks (2007 SP2.0 and 2009 SP 3.0) by Dassault Systemes. Custom EXPRESS grammar parsers were not developed, as that was not the focus of the project, but commercial off the shelf software was used (STEPTools ST-Developer 12 [23]) to generate the STEP files. The AP203 schema version referred to is dated September 2007 [1] although a newer version is currently available. Imagecom has been closely following the updates to the schema, but, at the time of writing this paper, the issues described in this paper have not been addressed.

The intended audience for this paper is software groups who specialize in development of translators for exchange of product model data between dissimilar CAD systems, and specifically groups who are looking to the AP203 Edition 2 standard as a possible solution to the interoperability problem. We hope the paper is of interest to the broader CAD audience should they want more details on limitations and workarounds associated with translating product model data to/from STEP AP203 Edition 2.

2 LITERATURE REVIEW

The lack of product representation interoperability among dissimilar CAD systems has been attributed to the loss of billions of dollars by the US industry alone [25]. Multiple commercial as well as government sponsored efforts have been launched over the decades to solve this problem. This section reviews some of these efforts.

2.1 State of Research and Standardization

Over the years various standards have been proposed and implemented successfully that exchange various aspects of product model data among heterogeneous CAD systems. The earliest of these was the Initial Graphics Exchange Specification (or IGES) which describes the product in terms of geometric and non-geometric information, with non-geometric information being divided into annotation, definition, and organization. The geometry category consists of elements such as points, curves, surfaces, and solids that model the product [16].

The IGES has been largely superseded, by the ISO 10303 standard, more commonly known as the STEP standard. The STEP Application Protocol (AP) 203 – Configuration Controlled Design – the file format most commonly implemented by commercial CAD companies enables the exchange of product model data amongst heterogeneous systems. The schema version popularly implemented has been Edition 1 of the STEP AP 203 standard. Edition 1 enabled CAD systems to exchange data using BREP (Boundary REPresentation). A newer version of the STEP AP 203 standard, viz. Edition 2 has been proposed and several iterations of this upgrade have been made. This new version enables exchange of product data using procedural and hybrid models [18].

In [18] and [19] the authors take the view that with STEP AP 203 Edition 2, standardized exchange of CAD models containing design intent has been successfully demonstrated. They do acknowledge on the other hand that translators developed for this standard have to be much smarter than simply reading and writing data that can be mapped one-to-one between CAD systems, as was the case with Edition 1. The authors credit Hoffmann [6] with suggesting the idea implemented by Edition 2 i.e. exchanging the construction history between CAD systems. They make an uncommon distinction between 'design intent' and 'design rationale'. By defining 'design intent' as corresponding to the way the facilities provided by the CAD system are utilized, they assert that the STEP file aims to exchange ‘design intent’ between CAD systems and not 'design rationale' which is the designer's motivation in choosing a particular methodology. To differentiate between features and operations that are exchanged with construction history, the authors quote Shah and Mantyla [24], that a feature has some associated application semantics, which CAD systems do not yet capture, while construction history is primarily a sequence of operations that create shape features. The authors also talk about operational granularity i.e. about how different CAD systems allow modeling of the same feature (or operations per the STEP standards) in slightly different ways. A one-to-one mapping for each feature in every CAD system, while ideal, is rarely possible, claim the authors.

Other authors [14] [28] [29] provide interpretations and details of the STEP format for design intent exchange and in varied domains. Researchers, though, have not stopped with the STEP standard as the solution to the interoperability problem. Varied solutions to exchange more than just shape data have been proposed. In [25] the authors suggest incorporating and representing knowledge in the next-generation of CAD systems. By providing a foundation for interoperability of knowledge the authors state that the next-generation of CAD systems could possibly solve this problem from the ground up. In [5] the authors address the problem of exchanging semantic information along with other product information such as shape by presenting a one-to-many framework built using a Domain Independent Form Feature model. They define a feature as a set of faces with adjacency relationship which enables the association of knowledge. The authors provide a unified ontology that eliminates the duplication in terminology used in multiple domains to describe the same feature or shape.

2.2 State of the Art

This section summarizes some commercial attempts to solve the interoperability problem.

Proficiency Inc. [20], recently acquired by ITI Transcendata [8], is probably the most well known company to provide data translation products and services. Inventor Ari Rappoport, affiliated with Proficiency, has a patent [22] filed for a method and apparatus for mechanical data exchange between parametric computer aided design systems. Proficiency's solution is built around a technology called Universal Product Representation (UPR), which can be thought of as a superset of mechanical CAD features, history, constraints, dimensions and sketches found in major CAD systems today. What this superset allows is the mapping of a feature from MCAD system A to the same feature in MCAD system B. In cases where there isn't a one-to-one correspondence between systems, the Proficiency product, Collaboration Gateway, will try to create the parametric equivalent of the original features, usually a combination of features. If this parametric equivalence is not possible, the Collaboration Gateway replaces the feature with a BREP equivalent with additional information in the target system [21]. .

Elysium Inc. is another data translation company that provides multi-CAD data exchange. CADFeature, their feature-based digital design exchange product [3] extracts 3D design intent via a plug-in that connects to a CAD system, records the building steps during a model replay operation and then replicates the building steps in the target CAD system to recreate the model [4].

Theorem Solutions also provides a broad range of translators. They have developed a set of Generic Collaboration Objects (G.C.O.), a set of objects into which all forms of data can be represented and held. Existing only in memory form, G.C.O. allows the process of developing new products to be sped up significantly. The G.C.O. can be described as a hub between all conversion processes and as such eliminates the need for multiple read/write routines, the method normally associated with direct database conversion [26].

Imagecom Inc, with whom the authors are employed, owns patented software [27] that performs 2D to 3D conversion. Other solutions developed by Imagecom include 3D to 3D translation using an in-house neutral file format named Universal Features Object (UFO) and API-based implementations to exchange product model data among numerous CAD systems. The UFO stores features with a definition that is common among most major CAD systems. Differences in feature definition in individual CAD systems from this common definition in the UFO, is handled intelligently by the API-based implementations that perform mapping between dissimilar features. The unique aspect of the UFO is that it is equally capable of storing data for 2D to 3D conversions as well as 3D feature-based translations [7].

3 ISSUES IN STEP CONSTRUCTION HISTORY TRANSLATION

This section outlines some of the issues encountered and the workarounds used during the development of the STEP AP203 Edition 2 translators, after a brief overview of the development effort.

3.1 Overview

The development effort targets commonly supported features that generally map between CAD systems on a one-to-one basis. Features supported are extrusions, revolves, holes (including pattern holes), fillets and chamfers. Within this set of features, many of the variations of creating each type are included in this effort. For example, an extrusion feature might be generated as a blind extrusion, through all extrusion, up to surface extrusion, up to vertex extrusion, or as a two-directional extrusion with any combination of the above termination types. Tab. 1 summarizes the supported features with a summary description of said features.

Feature Type	*Feature variations*
Extrusion	Blind, up-to-next, up-to-vertex, up-to-face, through all, mid-plane, two-direction
Revolve	Blind, bi-directional, up-to-vertex, up-to-face
Hole	Simple, Counterbore and countersink holes, each with various termination types including blind (with and without drill tip angle), through all, up to vertex, up to face
Fillet	Single or multi-edge fillet with constant radius
Chamfer	Single or multi-edge chamfers of type equal-distance, d1 X d2 and dist-angle

Tab. 1: Supported features with variations.

In addition to the five commonly supported CAD features, an effort was made to export and import datum features defined in the feature tree. These datum features include datum points, datum planes, datum axes and datum coordinate systems. In the case of datum features, a one-to-one mapping is difficult to achieve, thus mapping algorithms have been developed. A more detailed analysis of these mappings is discussed in section 3.6.

3.2 Issues Encountered

This subsection describes the issues encountered, interpretations made and mappings created during the implementation.

- Feature tree creation: Although a single feature in each CAD system creates a single item in the feature tree, this is not always the case with a STEP feature which resulted in the modification of the design intent. A workaround is implemented and discussed in this section.
- Extrusion and revolve feature schema: Decisions had to be made to select the appropriate STEP entities from several similarly defined entities to retain full design intent.
- Methods for defining end conditions: Strategies to map end conditions from CAD system to currently defined STEP entities were developed in order to distinguish between several similar yet unique end conditions.
- Datum feature definitions and intelligent translation: Mapping of CAD datum features to STEP entities results in a complete loss of design intent for each datum. The implemented workaround is discussed in more detail.
- 2D profile mapping and intelligent translation: Incomplete standards required the mapping of 2D profiles to 3D STEP entities. ISO10303-112 is currently under periodical review as of October, 2009.

3.3 Feature Tree Creation

ISO 10303-55 defines the procedural model as a "list of construction operations" in which the list represents the order in which the operations must occur. This entity, the procedural_shape_representation_sequence, has three subtypes – one each for solid model, surface model and wire frame model. The solid model sequence, procedural_solid_representation_sequence, was the desired class for this application. The description for this entity indicates that not all items in the list need be of type solid, however, the resulting model must be a solid model [9]. In the schema used for this project (September 2007), the three subtypes for the procedural_shape_representation_sequence (solid, surface and wire frame) are not defined. Since the completion of this prototype system, a new schema has been posted which does not contain the subtypes [1]. Therefore, the feature tree shown in each sample code segment utilizes the top-level class, procedural_shape_representation_sequence, while following the constraints associated with the procedural_solid_representation_sequece, as defined above. The procedural_representation_sequence, the supertype to the entity used above, contains two lists of items – one for features in the model and a second for suppressed features. For the purpose of this project, suppressed features are not processed.

As procedural_solid_representation_sequence is chosen to represent the feature tree, an understanding of solid constructional operations as they are defined in the step standards is necessary. A solid model, as mentioned earlier, must be the final result of the constructional operation sequence. Within STEP, a modified solid, which inherits from the solid_model, is created when an operation is performed on a solid. This is consistent with our earlier definition of a CAD system feature tree process, in which each operation modifies the current part, creating a new or updated part. Within the subset of features handled in this system, several of the step classes map directly to classes that inherit from the modified_solid. This construct contains a data element called the base_solid. This base solid references the original solid upon which the operation is to be performed [12]. The features handled in this application that inherit from this modified solid are the holes, fillets and chamfers. In particular, fillet and chamfer features both map to the edge_blended_solid and its various subtypes and the hole feature maps to the solid_with_hole and its various subtypes. Tab. 2 shows the mapping of supported CAD features to the respective AP203 edition 2 entities.

As an example of the operation sequence in STEP, if a solid_with_hole is created on an extrusion feature, the base solid will be the extrusion. If a subsequent feature is added, for example an edge chamfer, the edge_blended_solid will again contain a base solid, but this time, the base solid will be the result of the solid_with_hole operation. Each intermediate result is added to the item list in the procedural representation sequence. Therefore, in this example, the sequence would contain three

Feature	*Feature subtype*	*Part 111 mapping*
Fillet	Constant Radius	solid_with_constant_radius_edge_blend
Chamfer	equal distance	solid_with_single_offset_chamfer
	D1 x D2	solid_with_double_offset_chamfer
	D X Angle	solid_with_angle_based_chamfer
Hole	simple blind	solid_with_stepped_round_hole
	blind with drill tip	solid_with_conical_bottom_round_hole
	simple through simple through with exit csink simple up to surface simple up to vertex	solid_with_stepped_round_hole_and_solid_with_through_depression
	cbore blind	solid_with_stepped_round_hole
	cbore blind w drill tip	solid_with_conical_bottom_round_hole
	cbore through cbore through with exit csink cbore up to surface cbore up to vertex	solid_with_stepped_round_hole_and_solid_with_through_depression
	csink blind	solid_with_stepped_round_hole
	csink blind w drill tip	solid_with_conical_bottom_round_hole
	csink through csink through with exit csink csink up to surface csink up to vertex	solid_with_stepped_round_hole_and_solid_with_through_depression

Tab. 2: Supported features that map to modified solid subtypes.

items in the list – the initial extrusion, the solid with hole, and the edge blended solid. In Fig. 1, a sample part as that described has been created in SolidWorks 2007. This part was then exported to a STEP file. As evidenced in the STEP file segments shown in Fig. 2, line 122 represents the STEP feature tree whose item list contains three elements representing the extrusion, hole and chamfer features respectively.

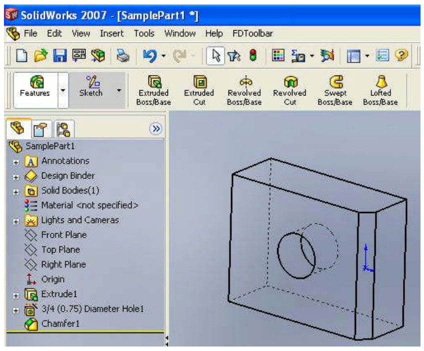
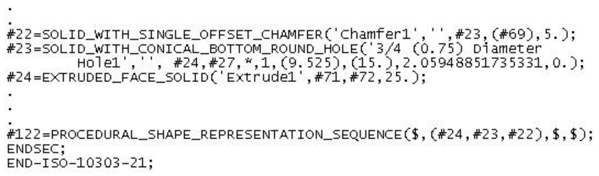


Fig. 1: Simple part with three features - extrusion, hole and chamfer.

Note, the feature tree construct is a procedural_shape_representation_sequence as discussed earlier. Once available, this element will be replaced with the procedural_solid_representation_sequence.



```
.
.
#22=SOLID_WITH_SINGLE_OFFSET_CHAMFER('Chamfer1','',#23,(#69),5.);
#23=SOLID_WITH_CONICAL_BOTTOM_ROUND_HOLE('3/4 (0.75) Diameter
        Hole1','', #24,#27,*,1,(9.525),(15.),2.05948851735331,0.);
#24=EXTRUDED_FACE_SOLID('Extrude1',#71,#72,25.);
.
.
.
#122=PROCEDURAL_SHAPE_REPRESENTATION_SEQUENCE($,(#24,#23,#22),$,$);
ENDSEC;
END-ISO-10303-21;
```

Fig. 2: Feature tree portion of step file.

3.4 Extrusion and Revolve Feature Schema

In the case of extrusion features, AP203 edition 2 schema appeared to offer two possibilities for creating this data. The first option was via subtypes of the modified_solid called the solid_with_depression or solid_with_protrusion for extrusion cut and extrusion join features respectively. The second option was via subtypes of the swept_face_solid. The solid_with_depression is a compound class that includes a solid_with_through_depression that can be used for identifying an extrusion termination type. However, using the modified_solid subtypes was deemed insufficient for several reasons. First, there is no equivalent class that can be used for a revolve feature and second, these classes can only be used to represent an extrusion in a single direction as opposed to bi-directional extrusions. Additionally, the solid_with_protrusion entity does not provide capabilities for any end condition other than a blind extrusion. The decision was thus made to instead use the swept_face_solid subtypes, extruded_face_solid and revolve_face_solid and their various subtypes to store extrusion and revolve features respectively.

The extruded_face_solid and revolve_face_solid both contain subtypes that allow for more complete mapping of the features. These classes follow the same pattern so we will discuss the extrusion feature only with the understanding that revolve features work similarly. With the subtypes associated with the extruded_face_solid, the prototype software could export and import a wide range of extrusion features. Tab. 3 shows the various extrusion features supported in the effort. With the exception of the drafted extrusion, all of these features could be transferred from either of the CAD systems to the second CAD system. In the case of the draft extrusion, Pro/Engineer and Solid Works have unique methods of creating a draft in the extrusion that would need mapping software to process the step data.

Feature	*Feature Subtype*
Extrusion	Blind extrusion
	Through all extrusion
	Up to Vertex extrusion
	Up to Next extrusion
	Up to Surface extrusion
	Midplane extrusion
	Drafted extrusion
	Bidirectional extrusion with any combination of termination type - blind, up to vertex, up to surface, up to next, through all - with or without drafts

Tab. 3: Supported extrusion feature variations.

The entity extruded_face_solid or revolve_face_solid creates the desired volumetric shape. For the first feature, the shape created is added directly to the feature tree sequence defined in section 3.2. Subsequent revolve or extrusion features require a second step to volumetrically add or subtract that shape from the base solid. The step object, Boolean_result, specifically handles that operation. The input to the boolean_result is the model created prior to this new feature and the extrusion or revolve feature being added to the part as well as the actual operation to be performed. The operations supported are union, intersection, and difference, where union and difference correspond to a join and cut operations respectively. Fig. 3 shows a modification to the first sample part in which a second extrusion (boss) has been added. The second extrusion, “Extrude 2”, is volumetrically joined to the base part which, at this time, only consists of a single extrude, “Extrude 1”, as seen in the feature tree in the figure. The two additional features, hole and chamfer, appear after the two extrusion features. In generating the STEP data for this second extrusion, an extruded_face_solid is created with all necessary data to fully define the extrusion and then a boolean_result object is created which will join the two extrusion features. Fig. 4 contains the STEP data associated with these operations. In the STEP file, #27 defines “Extrude 1”, #28 defined “Extrude 2”, and the volumetric joining of these two is shown in #24.

In the example part and STEP file shown in Fig. 3 and Fig. 4, four features were created and four items were added to the procedural sequence. In other words, there was a direct correlation between the number of features and the number of items in the sequence list. Similarly adding an extrusion or revolve feature to a part will add a single item to the list.

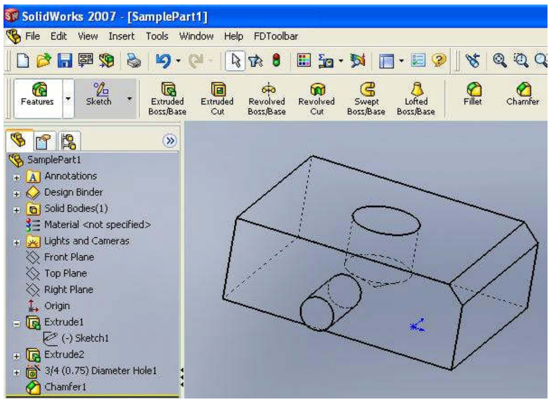
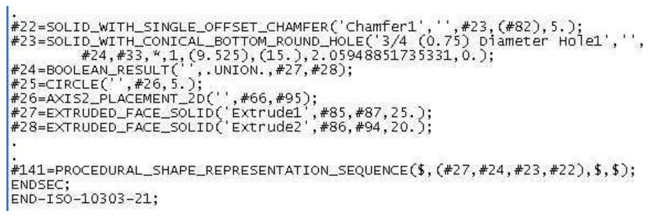


Fig. 3: Sample part with two extrusion features.



```
.
#22=SOLID_WITH_SINGLE_OFFSET_CHAMFER('Chamfer1','',#23,(#82),5.);
#23=SOLID_WITH_CONICAL_BOTTOM_ROUND_HOLE('3/4 (0.75) Diameter Hole1','',
        #24,#33,*,1,(9.525),(15.),2.05948851735331,0.);
#24=BOOLEAN_RESULT('',.UNION.,#27,#28);
#25=CIRCLE('',#26,5.);
#26=AXIS2_PLACEMENT_2D('',#66,#95);
#27=EXTRUDED_FACE_SOLID('Extrude1',#85,#87,25.);
#28=EXTRUDED_FACE_SOLID('Extrude2',#86,#94,20.);
.
.
#141=PROCEDURAL_SHAPE_REPRESENTATION_SEQUENCE($,(#27,#24,#23,#22),$,$);
ENDSEC;
END-ISO-10303-21;
```

Fig. 4: Sample step code for two extrusion features.

Note in the figure above that the sequence list, #141, contains four items and the feature tree contains four features. However, it requires two STEP objects to create the second extrusion – an extruded_face_solid and the boolean_result to volumetrically add the second extrusion feature to the first. Only the boolean_result object (#24) is added to the sequence list. The corresponding import software is responsible for analyzing the boolean_result obtained from the item list to create the second extrusion feature in the target CAD system.

3.5 Methods for Defining End Conditions

Exporting and importing extrusion/revolve and hole features with end conditions defined in terms of a reference was a major consideration in the development process. Although the methods used for each of these two features appear to be quite similar and both, ultimately inherit from a solid_model, there are slight variations in creating the STEP file as the extrusion/revolve features versus the hole features, which leads to similar variations in defining the end conditions of each feature. End conditions for these features are a key issue in propagating design intent.

3.5.1 Extrusion and Revolve Feature end Condition

As discussed earlier, the extrusion and revolve features were defined using the extruded_face_solid and revolve_face_solid respectively. Both of these inherit from the swept_face_solid, but more importantly, both of them have subtypes for defining numerous variations of end conditions, called trim conditions in STEP. These two classes are extruded_face_solid_with_trim_conditions and revolved_face_solid_with_trim_conditions. These add a trim condition and trim intent in two different directions, allowing for comprehensive feature definitions in one or two directions. In the case of a single direction extrusion, the second set of trim data was left null. The trim conditions supported in STEP are blind, offset, through all, unspecified and uptonext whereas the trim conditions supported by the development effort are blind, through all, uptonext, uptovertex and uptosurface. Blind and through extrusion types mapped between CAD systems and step files in a one-to-one manner, however it was clearly necessary to develop mapping algorithms to distinguish between the uptonext, uptovertex and uptosurface termination types. Tab. 4 shows the mapping of CAD extrusion types to the appropriate STEP data for both the trim condition and the trim intent. As shown in this table, up to vertex and up to surface appear identical at this point, but it is the data stored in the generalized surface select that was used to distinguish between these extrusions. In each case, the generalized surface used was an

Extrusion feature trim data		
CAD system end condition	*Trim condition step data*	*Trim intent*
blind	length measure	blind
through all	Null	through_all
up to next	Null	up_to_next
up to vertex	generalized_surface_select	up_to_next
up to surface	generalized_surface_select	up_to_next

Tab. 4: Extrusion feature end condition to CAD mapping.

elementary surface, which contains an axis2_placement_3d data object. This object contains an origin, normal vector and direction vector where both vectors are defined as optional data. It is this optional data that was used to distinguish between the vertex and the face. In the case of the up to vertex, the elementary surface data contained the origin point only whereas in the case of the up to surface, both the origin and the normal vector were defined.

A simple part was generated which included a base extrusion with an extrusion cut. Multiple STEP files were created for this sample part by varying the end condition of the second extrusion. Fig. 5 shows the second extrusion being modified for three different end conditions. STEP file data for the various end conditions for this are shown in Fig. 6. In the extruded_face_solid_with_trim_conditions, the last six data items represent the trim condition selections in both direction (data for revolve angle or extrusion depth or references such as surface), trim intent in both directions (see Tab 6.) and non-negative offset values defined in both direction. The offset values are set to 0 in all cases for this effort. The sample part extruded in a single direction, so the second trim select and trim intent are null ($). Note in Fig. 7, that all three of the extrusions, #23 in (a) and #24 in (b) and (c), have used the .Up_To_Next. trim intent. The distinction between up to next, up to vertex and up to surface occurs in the trim select data. In (a), the up to next extrusion contains null trim select data, whereas both (b) and (c) trim condition is set to an elementary surface (#23). In both cases, the elementary surface references an axis2_placement_3d (#30), but in (b), the axis2_placement_3d data for origin is set whereas in (c), the axis2_placement_3d data for origin and axis are both set, distinguishing between up to vertex (point only) vs. up to surface (point and normal).

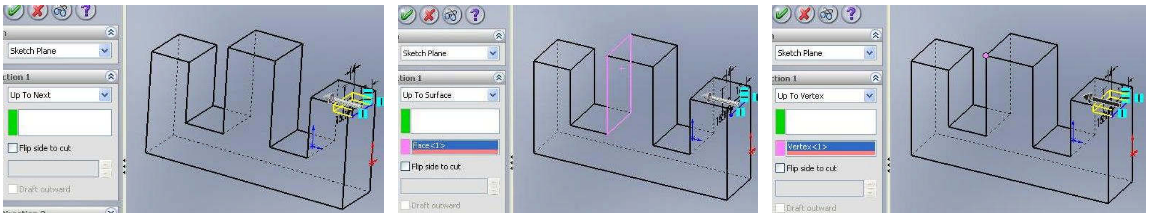
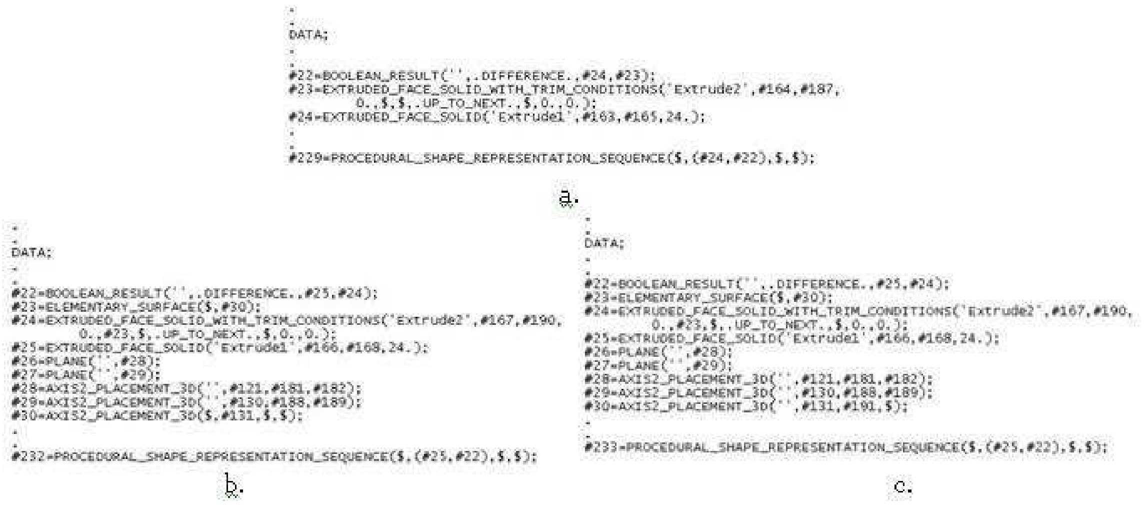


Fig. 5: Simple part created with base extrusion followed by an extrusion. End conditions: (a) extrusion up to next, (b) extrusion up to surface where surface is shown in pink, and (c) extrusion up to vertex in which the vertex point is highlighted in pink.

3.5.2 Hole Feature Termination Types

Hole feature, like extrusion features, contain end conditions for blind, through all, up to surface and up to vertex. The STEP class, solid_with_hole, does not contain data for trim conditions. In STEP, the solid_with_hole class inherits from the class solid_with_depression. This supertype may be either blind or through. In the case of a through depression, the object is a complex instance of type solid_with_hole and contains the additional subtype solid_with_through_depression. The solid_with_through_depression contains data to identify a set of exit faces. In order to successfully transfer design intent, the software was modified to utilize the exit faces data to distinguish between several different end conditions.

The following mappings were used for the hole end conditions in the solid_with_through_depression. If the hole end condition was of type through all, the exit faces data was left empty, or null. If the hole end condition was of type "up to surface", an exit face was created using an elementary surface with location and normal data to define the surface.



```
.
.
DATA;
.
.
#22=BOOLEAN_RESULT('',.DIFFERENCE.,#24,#23);
#23=EXTRUDED_FACE_SOLID_WITH_TRIM_CONDITIONS('Extrude2',#164,#187,
        0.,$,$,.UP_TO_NEXT.,$,0.,0.);
#24=EXTRUDED_FACE_SOLID('Extrude1',#163,#165,24.);
.
.
#229=PROCEDURAL_SHAPE_REPRESENTATION_SEQUENCE($,(#24,#22),$,$);
```

a.

```
.
.
DATA;
.
.
#22=BOOLEAN_RESULT('',.DIFFERENCE.,#25,#24);
#23=ELEMENTARY_SURFACE($,#30);
#24=EXTRUDED_FACE_SOLID_WITH_TRIM_CONDITIONS('Extrude2',#167,#190,
        0.,#23,$,.UP_TO_NEXT.,$,0.,0.);
#25=EXTRUDED_FACE_SOLID('Extrude1',#166,#168,24.);
#26=PLANE('',#28);
#27=PLANE('',#29);
#28=AXIS2_PLACEMENT_3D('',#121,#181,#182);
#29=AXIS2_PLACEMENT_3D('',#130,#188,#189);
#30=AXIS2_PLACEMENT_3D($,#131,$,$);
.
.
#232=PROCEDURAL_SHAPE_REPRESENTATION_SEQUENCE($,(#25,#22),$,$);
```

b.

```
.
.
DATA;
.
.
#22=BOOLEAN_RESULT('',.DIFFERENCE.,#25,#24);
#23=ELEMENTARY_SURFACE($,#30);
#24=EXTRUDED_FACE_SOLID_WITH_TRIM_CONDITIONS('Extrude2',#167,#190,
        0.,#23,$,.UP_TO_NEXT.,$,0.,0.);
#25=EXTRUDED_FACE_SOLID('Extrude1',#166,#168,24.);
#26=PLANE('',#28);
#27=PLANE('',#29);
#28=AXIS2_PLACEMENT_3D('',#121,#181,#182);
#29=AXIS2_PLACEMENT_3D('',#130,#188,#189);
#30=AXIS2_PLACEMENT_3D('',#131,#191,$);
.
.
#233=PROCEDURAL_SHAPE_REPRESENTATION_SEQUENCE($,(#25,#22),$,$);
```

c.

Fig. 6: STEP file segments for three extrusion cut end conditions. (a) extrusion up to next, (b) extrusion up to vertex, and (c) extrusion up to surface.

If the hole end condition was of type “up to vertex”, the exit face data was again set to an elementary surface, but only the location data was set. The end conditions, “up to vertex” and “up to surface” followed the same pattern used in the extrusion feature as was shown in Fig. 6. Since the solid with hole is a complex object, a method for distinguishing the “up to next” termination type was not mapped. A suggestion might be to create an empty exit face in which the origin and normal are both set to null. This method was not chosen since the class definition for axis2_placement_3d indicates that, although the normal and direction vectors are both optional data, the location is required data.

3.6 Datum Feature Definitions and Intelligent Translation

Datum features create a unique set of issues when exporting and importing between the CAD systems and the AP203 CH. The initial analysis of the task highlighted two areas of concern. The first issue is the varied methods that are possible for creating datum features within a CAD system. The methods for generating datum features, while similar in many aspects, were vastly different in some instances between the two CAD systems. Added to this is the limitation of obtaining datum information via the API’s. As an example, Pro/Engineer WF 3 supports general and sketched datum points. In the case of sketched datum points, the modeler may place the datum points anywhere on the referenced surface. However, in SolidWorks 2007, the only sketched datum point occurs when the modeler selects the option to add a datum point in the center of a face. The second issue encountered was the lack of specific class definitions within AP203 edition 2 associated with datums. Due to the various limitations, the prototype development focused on the creation of ‘intelligent’ algorithms for defining and importing the various datum elements.

3.6.1 Datum Feature to Step Correlation

Four datum elements were analyzed prior to the implementation. These four are datum planes, datum points, datum axis, and datum coordinate systems. Due to time constraints, the development effort focused on datum planes and datum axes, although recommended mappings were generated for each. Tab. 5 shows the recommended mapping that was generated for this prototype system.

In Wildfire 3, a datum point is defined as either a sketch point or a general point. In both cases, the STEP entity selected was a compound_representation_item. This construct contains data called item element which is either a list or a set of representation items. In the case of a sketched datum point, the decision was made to generate a list of items in which the first item in the list is an axis2_placement_3d to define the planar surface, followed by a list of Cartesian points. In the case of a general datum, the list would only contain Cartesian points.

Datum Feature	*Part 111 mapping*
Plane	plane or offset_surface
Axis	axis1_placement
Point	compound_representation_item
Coordinate System	axis2_placement_3d

Tab. 5: Recommended AP203 mapping for datum features.

Similarly, a datum plane can be defined in a number of methods. As AP203 classes do not contain methods to create a plane using references, with the exception of an offset plane, the step class construct, plane, was used to represent all planes. It then becomes the responsibility of the interface software to generate a plane in the target CAD system utilizing that system's referencing capability.

3.6.2 Algorithm Sequence for Datum Interface Software

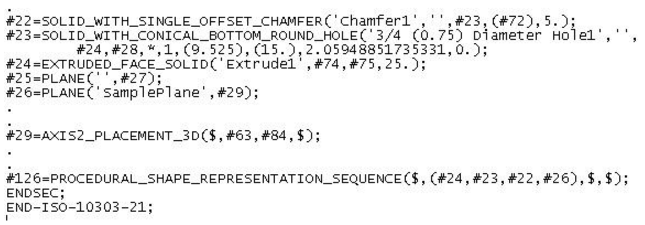
Import and export functions were implemented in the each of the CAD systems for datum planes and datum axes. In the case of export function, simple algorithms were generated for each CAD system interface to extract the data using the API's and create either a simple plane, containing a point and a normal, or an offset plane, with point, normal, and distance. Although not comprehensive in either CAD system, a significant subset of the datum features was supported.

To import the datum features from the step file, heuristic approaches was developed and implemented for both SolidWorks and Pro/engineer. No formal study has been performed to analyze statistical data on the methods that a modeler might use when creating datum features. The algorithms used in the prototype software are, by no means, listed as recommended practices, but are intended to highlight the necessity of creating 'smart' import software that take into account the target CAD system's strengths and weaknesses. These algorithms need to be evolved as per recommended modeling practices. It is of note, however, that any design intent by the modeler expressed in datum features may be lost in the translation from one CAD system to another.

In the case of datum axis feature, algorithms were inconsistent in that a datum axis feature in Pro/Engineer first searched for the step axis being congruent with an edge, then with the axis of a cylindrical surface. If neither of those conditions is met, a third check is made for a reference point such as an edge endpoint and a surface normal. In SolidWorks, the algorithm checks for cone or cylindrical surface prior to an edge.

The algorithms for datum plane features followed a more consistent pattern first developed by the company in the patented 2D to 3D conversion software effort. To create a datum plane, the algorithm will first look for a bounded surface, such as a planar face or an arc edge. If no planar surface is found, a check is made for unbound surface (a plane). If there is still no match, the algorithm will search to see if the point falls on a cylindrical surface, and if one if found, a second check for a parallel or perpendicular plane is performed.

The part created in Fig. 1 was modified to include a plane created parallel to another plane through a point labeled 'sample plane'. This plane feature has been exported to the STEP file as a 'plane' element. The plane was included in the model as a datum feature and appears in the feature tree, hence it is added to the STEP construction history item list. Note that in Fig. 7, #126 now contains four items in the feature list as opposed to three items in the sample step code shown in Fig. 2. The fourth item in the list, #26, references the plane, which, in turns, references item #29, axis2_placement_3d, which part 42 indicated "can be used to … define a placement coordinate systems" [9]. The data associated with the axis2_placement_3d includes a point (location) and a normal vector (axis) which is set in the creation of the plane. The third data element, ref_direction, is an optional vector and is left null in this case and the plane is not an oriented plane.



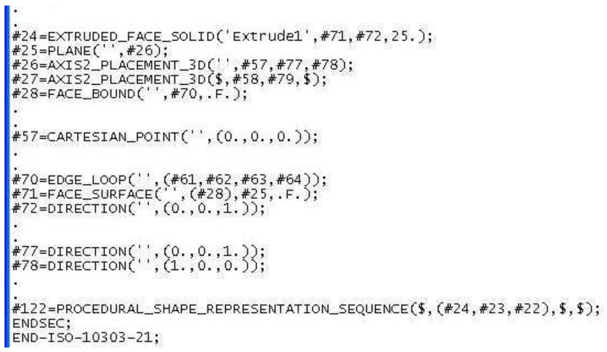
```
.
#22=SOLID_WITH_SINGLE_OFFSET_CHAMFER('Chamfer1','',#23,(#72),5.);
#23=SOLID_WITH_CONICAL_BOTTOM_ROUND_HOLE('3/4 (0.75) Diameter Hole1','',
        #24,#28,*,1,(9.525),(15.),2.05948851735331,0.);
#24=EXTRUDED_FACE_SOLID('Extrude1',#74,#75,25.);
#25=PLANE('',#27);
#26=PLANE('SamplePlane',#29);
.
.
#29=AXIS2_PLACEMENT_3D($,#63,#84,$);
.
.
#126=PROCEDURAL_SHAPE_REPRESENTATION_SEQUENCE($,(#24,#23,#22,#26),$,$);
ENDSEC;
END-ISO-10303-21;
```

Fig. 7: Step file showing datum plane include in feature tree.

3.7 2D Profile Mapping and Intelligent Translation

When developing the extrusion and revolve feature export and import functions, the 2D sketch for each feature is processed. At the time of the development, AP203 edition 2 did not define any constructs for defining these 2d sketches in a STEP file. This problem was circumvented by creating a 3D bound surface identified by the collection of 3D entities necessary to define that surface. In the case of a circular sketch, a single 3D circle identifies the sketch.

By using a 3D face to represent the sketch, it was necessary to development algorithms for converting between the 2D sketch and the 3D face. In order to achieve the transformations between the 2D sketch and 3D face, it was necessary to export and import the transformation matrix in some form. The 2D sketch was defined using a face_surface, a subtype of face containing a set of face_bound elements. The face_surface also contains a data element, face_geometry, which can be of type elementary surface, in this case an oriented plane, that is used to calculate the transformation between the 2D and 3D sketches. Fig. 1 contains an extrusion feature with a single closed loop profile. In the STEP code segment shown in Fig. 8, the extrusion feature is represented by #24 with #71 referencing the face surface, #70 referencing the extrusion direction. The extrusion depth is given as 25 units. The face surface, #71, is the representation for the 2D sketch. The first data in the face surface, #28, references a face bound which, in turn, references one of more edge loops consisting of a collection of oriented edges. Part 42 [9] contains detailed information on low level geometry. The second data in the face surface, #25, is used to store data necessary for a 2D to 3D transformation and vice versa. The data used for this transformation is a plane object referencing an axis2_placement_3d object in which both the required and optional data are set, thus creating the equivalent of a 3D coordinate system. In this case, the coordinate system origin is given by #57 which is the world origin (0, 0, 0), the normal vector is given by #77 which is the positive z axis, and the orientation of the plane is given by #78 which is the positive x direction. One final data element in the face surface is the same_sense flag which was used to store the flag indicating extrude in the same direction or reverse direction from the face surface. In this instance, the flag is set to false (.F.) and the extrusion is in the same direction as the reference plane.



```
.
.
#24=EXTRUDED_FACE_SOLID('Extrude1',#71,#72,25.);
#25=PLANE('',#26);
#26=AXIS2_PLACEMENT_3D('',#57,#77,#78);
#27=AXIS2_PLACEMENT_3D($,#58,#79,$);
#28=FACE_BOUND('',#70,.F.);
.
.
#57=CARTESIAN_POINT('',(0.,0.,0.));
.
.
#70=EDGE_LOOP('',(#61,#62,#63,#64));
#71=FACE_SURFACE('',(#28),#25,.F.);
#72=DIRECTION('',(0.,0.,1.));
.
.
#77=DIRECTION('',(0.,0.,1.));
#78=DIRECTION('',(1.,0.,0.));
.
.
#122=PROCEDURAL_SHAPE_REPRESENTATION_SEQUENCE($,(#24,#23,#22),$,$);
ENDSEC;
END-ISO-10303-21;
```

Fig. 8: Sample step file for profile sketch.

4 SUMMARY AND CONCLUSIONS

STEP AP 203 Edition 2 provides a way to store design intent using constructional operations and facilitates exchange of feature based CAD models. Schema procedural_representation_sequence (ISO 10303-55) [10] defines sequence of constructional operations in the solid model. Schema solid_shape_element_schema (ISO 10303-111) [12] defines representation of protrusion, hole, chamfer and fillet features. Use of positioned_sketch to define planar protrusion profile is also equivalent of 2D sketch profile in CAD systems. Entities solid_with_hole and edge_blended_solid from solid_shape_element_schema were used to represent hole and chamfer/ fillet features respectively. However, solid_shape_element_schema does not define revolve solid and trim conditions for protrusion features. Thus, entities extruded_face_solid and revolve_face_solid defined in geometric_model_schema (ISO 10303-42) [9] were used to represent extrusion and revolve features with trim conditions. One disadvantage in using extruded_face_solid and revolve_face_solid is that entity face_surface is used to define a 3D face of solid created by feature. This needs conversion between 2D sketch profile in CAD systems and 3D face_surface. Trim conditions blind, through all, and up to next can be defined using available trim_intent and trim_condition_select entities. However, for up to surface and up to vertex trim conditions, generalized_surface_select data needs to be mapped.

STEP AP 203 Edition 2 also provides a way to capture design intent behind using a particular procedural_shape_representation_sequence. Entities defined in solid_shape_element_schema also provide way to define design rationale using the rationale text attribute. This can be further mapped to CAD feature data (e.g. notes associated with features in Pro/Engineer).

Since the completion of this development, new AP203 constructs have been defined by the International Standards Organization and are currently in the review process. Part 112 defines 2D sketch objects [13].

Imagecom, Inc. has developed commercial translators (currently in closed beta), to exchange design intent information between two commercially available CAD systems, SolidWorks 2009 and Pro/Engineer Wildfire 4.0, through the STEP AP 203 Edition 2 using the mappings detailed in this paper. Sample AP203 files created by our translators are available by request for Modelers and Developers for evaluation or other purposes.

ACKNOWLEDGEMENTS

Imagecom would like to extend their gratitude to the 3D Team of the Engineering Business Group at Tank-Automotive Research, Development, and Engineering Center (TARDEC). Special thanks go to Ms. Suhair Ghannam for her continued support during the course of the research and development.

REFERENCES

[1] CAx-Implementor Forum: http://cax-if.org/documents/AP203E2_Sep2007/ap203_configuration_controlled_3d_design_of_mechanical_parts_and_assemblies_mim.lfepm_50001009.html
[2] CAx-Implementor Test Rounds.: http://cax-if.org/joint_testing_info.html#testrounds
[3] Elysium Inc.: http://www.elysiuminc.com/
[4] Elysium Inc.: http://www.elysiuminc.com/PDF/ElysiumCADfeatureProfile080901tm.pdf
[5] Gupta: R.; Gurumoorthy, B.: A Feature-based Framework for Semantic Interoperability of Product Models, Journal of Mechanical Engineering, 54 (2008) 6, pp 446-457.
[6] Hoffmann: C. M.: Geometric and Solid Modeling. Morgan Kaufmann, SanMateo, CA, 1989.
[7] Imagecom Inc, http://www.aspire3d.com
[8] International TechneGroup Inc.: http://www.transcendata.com/news/iti/articles/ITI%20Proficiency%20Press%20Release.pdf.

[9] ISO 10303-42: Industrial automation systems and integration – Product data representation and exchange: Integrated Generic Resource: Geometric and Topological Representation. Geneva, Switzerland. International Organization for Standardization. 2003.

[10] ISO 10303-55: Industrial automation systems and integration – Product data representation and exchange: Integrated Generic Resource: Procedural and Hybrid Representation. Geneva, Switzerland. International Organization for Standardization. 2005.

[11] ISO 10303-108: Industrial automation systems and integration – Product data representation and exchange: Integrated Application Resource: Parameterization and constraints for explicit geometric product models. Geneva, Switzerland. International Organization for Standardization. 2005.

[12] ISO 10303-111: Industrial automation systems and integration – Product data representation and exchange: Integrated application resource: Elements for the procedural modeling of solid shapes. Geneva, Switzerland. International Organization for Standardization. 2007.

[13] ISO 10303-112: Industrial automation systems and integration – Product data representation and exchange: Integrated application resource: Modelling commands for the exchange of procedurally represented 2D CAD models. International Organization for Standardization. 2006.

[14] Kim, E.; Park: G; Choi: I; Jeong, Y.: Information management for building structural design on the commercial CAD system, IEEE, 2004, 24 – 27.

[15] Kim: J.; Pratt: M.; Iyer: R.; Sriram, R.: Standardized data exchange of CAD models with design intent, Computer Aided Design, 40(7), 2007, 760-777.

[16] Leyton, M.: A Generative Theory of Shape. Springer-Verlag, Berlin, 2001, (Lecture Notes in Computer Science, Volume LNCS 2145)

[17] NIST: http://ts.nist.gov/standards/iges/

[18] Pratt: M.; Kim: J.: Experience in the Exchange of Procedural Shape Models using ISO 10303 (STEP), ACM Proceedings of the 2006 ACM symposium on Solid and physical modeling, 2006, 229-238.

[19] Pratt: M.: Extension of ISO 10303, the STEP Standard, for the Exchange of Procedural Shape Models, Proceedings of the Shape Modeling International 2004 (SMI'04), IEEE.

[20] Proficiency Inc.: http://www.proficiency.com/

[21] Proficiency Inc.: http://www.proficiency.com/downloads/mediaDesktopEngeering11-04.pdf

[22] Rappoport: A.: United States Patent: 6,614,430, System and method for the exchange of CAD data.

[23] STEPTools Inc, http://www.steptools.com/products/stdev/

[24] Shah: J. J.; Mantyla: M.: Parametric and Feature-based CAD/CAM. Wiley, 1995.

[25] Szykman, et. al.: The Role of Knowledge in Next-generation Product Development Systems, ASME Journal of Computing and Information Science in Engineering, 2001.

[26] Theorem Solutions, http://www.theoremsolutions.com/pages/tecov.htm

[27] United States Patent: 7,492,364, System and method for creating and updating a three-dimensional model and creating a related neutral file format

[28] Zhang: Y.; Luo: X.: Design Intent Information Exchange of Feature-based CAD Models, 2009 World Congress on Computer Science and Information Engineering, 2008, IEEE.

[29] Zhou: X; Jia: H.; Lu: Y; Ding: W.: Product Model Data Exchange Technology of Heterogeneous Systems in Collaborative Design Environment, Proceedings of the International Conference on Artificial Reality and Telexistence 2006, IEEE.