A Neutral XML Design Framework For Generating Parametric Parts In Multiple CAD Systems

Jonathan E. Sadler[1], Ryan D. Day[1], Parker G. Bronson[1] Jacob L. Tovar[1] John L. Salmon[1]

[1]Brigham Young University,

Corresponding author: John Salmon, johnsalmon@byu.edu

Abstract. Engineering companies often require designers to work with and operate between different CAD systems. This motivates the necessity of having a neutral design standard for CAD models to facilitate generation, customization, and parameterization across multiple systems. Current standards often fail to incorporate file history, design intent, and are designed to be an intermediary format for translation. This research proposes a simplified, neutral design format based on XML that can be used to generate models in different CAD systems and versions. The format utilizes advanced features such as finding tools to address issues that arise due to directionality and selection without a user interface. To test the feasibility of the neutral format, scripts were made that generated a quadcopter and an airplane wing rib. Comparisons between the generated models in both NX and CATIA are performed to evaluate the accuracy, flexibility, and similarity of the results of the generated neutral XML design framework.

Keywords: XML, Parametric Design, Design Intent, Neutral Format
DOI: https://doi.org/10.14733/cadaps.2019.923-935

1 INTRODUCTION

In today's global environment, a company may require the use of many different CAD systems in order to satisfy the needs of their customers. Many customers need parts that must be integrated into assemblies which may or may not have been created in the same system as the supplier. The suppliers must then use neutral formats to transfer the model or else have to reconstruct the part entirely in the correct system. Many times design intent is lost in this process. Through the history of CAD there have been many attempts to create a neutral standard that transfers and preserves design intent across systems in order to solve these problems. Current standards have limitations – they mainly preserve geometric data and not design intent. Errors in transferring these types of neutral CAD formats across systems and interpreting their design intent cost the automotive industry an estimated one billion dollars each year [5].

Finding an improved neutral format or a system to replace the need for an intermediary neutral format is therefore imperative to reduce unnecessary labor. Most solutions in literature focus on neutral formats treated as files that are difficult to edit and understand [12, 11, 17]. Our solution focuses on utilizing an XML script to represent design intent. The features of this script are defined in our neutral XML design framework and include *if* statements and *for* loops. The XML script is designed to be created independently of a CAD system and then translated by a program to generate a model in a CAD system. Moreno and Baz discussed a similar concept of automation using scripting, but their solution focuses on automating the creation of technical drawings with parametric extensions for non parametric programs. [13]. We focus on displaying design intent from parametric CAD systems in a human readable format.

2 LITERATURE REVIEW

The basis for ideal modern neutral formats was presented in the Editable Representation (EREP) developed by Hoffman and Juan [8]. This paper developed the basic architecture for modern feature design history used universally in CAD systems today [4]. The ideas associated with EREP have been implemented in many CAD systems with differences in naming and types of specialized features. Although there have been attempts at standardizing these names and feature types [16, 18], no standards have been widely implemented. Since the time Hoffman and Juan published their research, there have been many attempts to capture parametric CAD data in a neutral format based on the EREP version of modeling summarized and discussed below.

The current most commonly used methods for neutral data exchange involve the Initial Graphics Exchange Specification (or IGES) file and STEP file formats. These formats only support geometry transfer without parameters and constraints. IGES, while still a widely-supported type today, is considered inferior to the STEP format because it imports models primarily as surfaces and has many problems with file transfers that require human intervention to fix [3]. The STEP file is an implementation of the ISO 10303 STEP standard that transfers solid geometry and is commonly used in industry [15]. Although it is good at representing the solid geometry of a model through the STEP AP203 Edition 1 standard, this protocol is missing critical design features and intent. Parameters and constraints cannot be represented in this format, and so it cannot faithfully represent design history to show design intent [14].

There have been many attempts to extend the STEP file in order to include a parametric design history with robust transfer ability. The Enabling Next Generation Mechanical Design (ENGEN) project attempted to extend the STEP format based on Part 42 of the STEP standard to transfer design intent [1]. They proposed including parameters, constraints, and design history with the STEP file. The problem with the ENGEN project is that it focused on 2D profile data, and therefore does not have enough information to accurately represent a history-based parametric model. There are other neutral formats that have used modeling representations identified in different parts of the STEP standards [11]. At the present time there has not been a widely adopted extension of the STEP standard that includes all design intent information associated with a model [10]. ISO released STEP AP203 version 2 with the ability to represent parametric construction history models, but this also has not been adopted as a method of transfer among CAD systems [2]. No commercial CAD software has implemented AP203 version 2. Even if these systems adopted a AP203 version 2, the EXPRESS schema of the STEP standard associated with this AP is not easy to read and write and is mainly meant to be viewed as a neutral format rather than as an editable schema [20]. ‘

In addition to an extended STEP standard, the macro-parametric approach has been another area explored for neutral transferability. When a CAD model is constructed, most software packages generate an accessible macro file. In the macro-parametric approach, the macro file is converted into a neutral format, which can then be used with another translator to reconstruct the model in a different CAD system. In this way, all important design considerations can be captured, including parameters.

An example of this was done by Choi, Mun, and Han [6]. In their experiment, they designed a one way translator from SolidWorks to CATIA. They took the macro file generated from a user-created part in SolidWorks, converted it into a neutral format, and then converted the neutral format to CATIA. (See reproductions of their framework in Fig. 1 and 2).

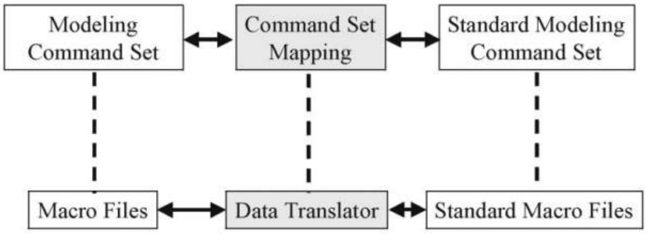

Figure 1: Architecture for macro-parametric approach. Reproduced from [6].

Kim and Han further expanded on the former idea by introducing a TransCAD module that does most of the boilerplate translation generally performed by individual CAD system translators, reducing the work needed to implement translators for additional CAD systems. [9]. More recently, Manhua, Yanjun, and Guojiang implemented a similar idea called MidCAD [12].

These two studies use XML as the intermediary neutral format for translation. This is advantageous because it makes the neutral format easily supported over the internet [19]. Yang et al. also produced a method that attempted to encapsulate CAD data in XML format [19]. Their XML format is based on the EXPRESS schema of a STEP file, and acts similar to a STEP file. A macro file is translated into a STEP compliant XML format, which can then be reconstructed in another CAD system.

The limitations of the macro-parametric approach is that the approach require two interpreters for each CAD system, one to translate the macro file into the neutral format, and another to translate a neutral format file to a CAD model. It is non-trivial to implement each interpreter, even with tricks such as using the TransCAD or MidCAD modules to reduce complexity. There is also no easy way to modify the neutral file that will be ported to another system: it merely acts as a proxy for the part to be transferred, including, at times, complex, hard-to-understand content in the XML. Another relevant issue that Kim and Han discuss is the problem of transferring selected features. CAD systems implement different methods for determining selection. For example, Siemens NX uses 3D coordinates to determine selection and CATIA V5 references sketch and feature data. This can cause problems when translating neutral formats from one CAD system to another [9]. A universal solution to this problem is not addressed in any of the macro-parametric implementations mentioned. Finally, not all CAD systems produce an easily deciphered macro file, and so the macro-parametric approach is not guaranteed to work on every system.

A third method for neutral transferability is one presented by Staves, Salmon, and Red [17]. Their research discusses a system that is used in order to construct a part in real-time in different CAD systems. The neutral format this research uses for transferring data across these systems is based on an SQL database. The system stores features according to the feature's parameters in the SQL database, which can then be queried by other users for translation into other CAD systems. This system is very adaptable because if more information is required to accurately neutralize a feature, it is easily added. There are still drawbacks to this system. One is the problem of having to construct a two-way translator for any CAD system one desires to use with the program. The neutral format, though it is editable, is only accessible by one who is familiar with SQL syntax. It is also necessary to have the required SQL software installed on every computer that will use this neutral format.

In addition to the neutral format transferability problem, our research solves issues faced by companies who use a CAD specific API to generate customizable parts. In using the API of a specific CAD system to automate part creation, a company has to tightly couple itself to the current version of the API of the CAD system. When the CAD system updates their API, changes have to be made to every applicable program

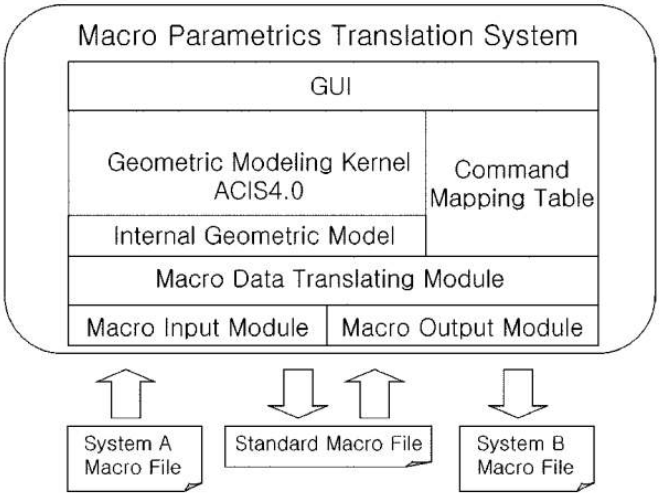

Figure 2: Implementation of macro-parametric approach. Reproduced from [6].

in the company. In addition, the programs only work with one CAD system. Having a neutral format with interpreters allows for a part generation program to work on as many systems as there are interpreters for the neutral format. When an update is pushed out, changes only need to be made to the interpreters of that specific CAD system instead of to every applicable program.

3 METHODOLOGY

The neutral framework seeks to accomplish several tasks. Primarily the system must be able to recreate CAD models that reflect the design intent defined in the neutral framework. A second task that needs to be accomplished is that the generated models in different CAD systems must be accurate to each other. The overall process for how this was implemented is presented in Fig. 3.

The design framework begins with a neutral script that expresses the operations to be performed. XML was utilized for this task as it can be modified quickly by multiple users and is human readable. Previous studies done by Yang [19] have shown it to be effective and representing CAD geometry while remaining human and machine readable. Like Yang, various CAD systems were used to determine appropriate terminology to describe the features. Current industry standard schemas used for design intent representation such as the

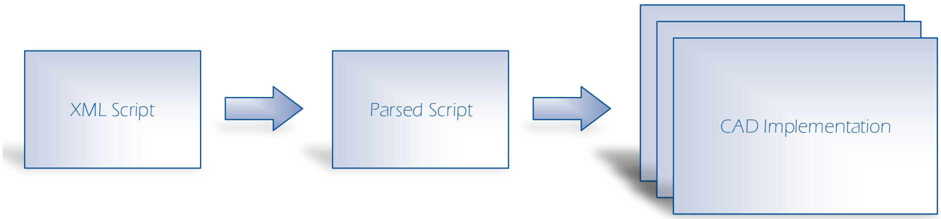

Figure 3: Pipeline for script processing

EXPRESS schema of a STEP file can be difficult to understand and produce [7]. More research will be required to find a balance between ease of use and language robustness.

Some functionality was changed to reduce implementation complexity. First, the use of complex features associated with XML such as XML namespaces were avoided to help reduce the learning curve of working with the format. An example XML script can be seen in the Listing of Fig. 4.

```
<InstructionSet>
    <Instructions>
        <Revolve>
            <Sketch>
                <Plane FromDefault="XY"/>
                <Line>
                    <Name>Axis</Name>
                    <StartPoint>
                        <X>0</X>
                        <Y>0</Y>
                        <Z>0</Z>
                    </StartPoint>
                    <EndPoint>
                        <X>0</X>
                        <Y>10</Y>
                        <Z>0</Z>
                    </EndPoint>
                </Line>
                ...
            </Sketch>
            <Axis FromName="Axis" />
            <StartAngle>0</StartAngle>
            <EndAngle>360</EndAngle>
        </Revolve>
    </Instructions>
</InstructionSet>
```

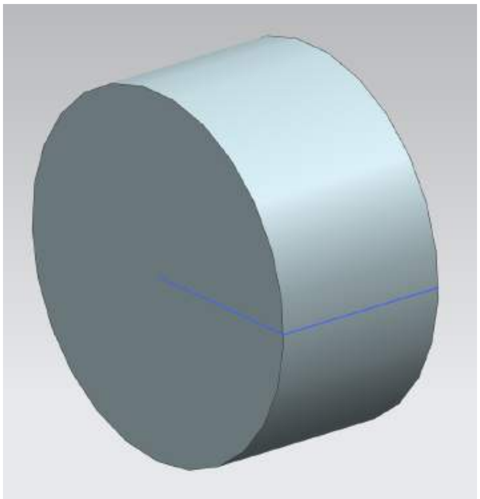

Figure 4: Basic XML example script (left) with associated CAD output (right)

The primary commands found in the script are used to make CAD features (i.e., extrudes, revolves, etc.). Included in the commands are the data required to make the features. For example, extrudes generally require a sketch and a value defining how far the sketch will be extruded. These features can either be referenced directly or by referencing previously created objects by their provided name.

One difficulty that arises with scripting is the ability to reference existing geometry. This was solved by use of "Finders" (See code Listing of Fig. 5). The finder code allows geometry to be searched for based on different criteria. The current method for searching existing features is based on near/far geometry. Likewise, input parameters allow finding different objects based on existing geometry (e.g., retrieving the end points from a line). This functionality is not generally found in neutral exchange formats.

Additional functionality can be added by means of adding additional scripts. By referencing other scripts, more complex features can be created by reducing the need to have repeated code. Scripts also allow feature references and values to be passed into the script. An example can be seen in the Listing of Fig. 6.

The scripts also allow for conditional statements and loops. This gives it the flexibility to make parts more parametrically. This is similar to research done by Moreno [13]; however, it adds human readability and the flexibility to work between different CAD systems. Currently supported is the ability to have *if* statements as well as *for* and *while* loops.

Finally, input commands allow the program to prompt the user for information. This information may be to identify or select already existing features or defined values or quantities. This functionality allows the

```xml
<Finder>
    <Name>Point</Name>
    <References>
        <Reference Type="Near">
            <Object>
                <Plane FromDefault="XY"/>
            </Object>
        </Reference>
        <Reference Type="Near">
            <Object>
                <Plane FromDefault="YZ"/>
            </Object>
        </Reference>
        <Reference Type="Near">
            <Object>
                <Plane FromDefault="XZ"/>
            </Object>
        </Reference>
    </References>
    <Output>Point</Output>
</Finder>
```

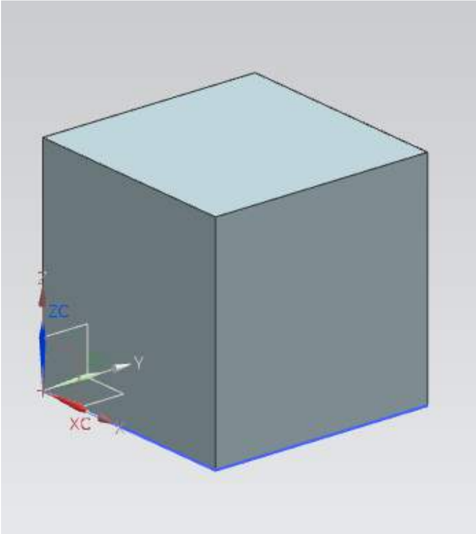

Figure 5: Finder example script (left). If the script is used with the cube (right), the origin would be the point that is found.

format to work with and modify already existing models.

When the program is executed, users are presented with the scripts that are available. The XML scripts are currently stored locally; however, with few additional steps they can be stored on a central server. Upon selection, the XML is loaded into memory, interpreted, and checked for accuracy. Following this step, the data is translated by the respective CAD systems and versions for model generation. This generally involves finding referenced features within the part as well as making necessary API calls.

```xml
<Script>
    <ScriptName>Revolve</ScriptName>
    <Input>
        <Expression>
            <Name>Angle</Name>
            <Value>125</Value>
        </Expression>
        <Instruction>
            <Name>BaseSketch</Name>
            <Value FromName="Sketch" />
        </Instruction>
    </Input>
</Script>
```

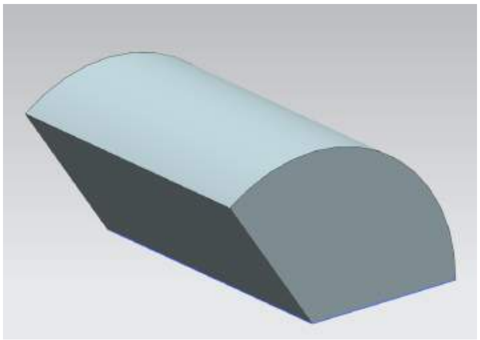

Figure 6: XML script that specifies the script to call and the parameters to pass in (left) with associated CAD example output (right). The angle parameter and sketch are used withing the called script.

4 IMPLEMENTATION AND CASE STUDIES

The above described framework was implemented and tested on a variety of simple shapes and features to evaluate the consistency, accuracy, and flexibility of the framework. The attributes of accuracy and flexibility are discussed in the following sections. Subsequent sections present two case studies that implement, demonstrate, and evaluate the neutral framework on more complex CAD models: a quad-copter and an airplane rib.

4.1 Accuracy

When transferring any CAD object from one CAD system to another, the most important metric is accuracy. If a neutral design framework cannot generate identical models in different systems, it is less useful than simply creating a model manually and transferring it. Most research today on neutral CAD formats is focused on simultaneously transferring an accurate model, feature history and other design intent parameters. For this research, accuracy was determined by selecting different points and measurements on each model and comparing them between systems. Matching values indicate an accurate representation. In addition, since XML scripts use the same steps to create a part, design history can be compared for matching features.

4.1.1 Flexibility

A major benefit of focusing on scripts as opposed to neutral file formats is the ability to easily modify new models. Features can easily be adapted to generate different types of models. The ability to generate multiple parts from a single script, by simply changing a few parameters, is an advantage in this framework over other neutral languages regardless of CAD system and version.

This level of flexibility allows changes in parameters such such as feature size and quantity of features. The flexibility of models is demonstrated in Figures 7 and 8. Three different cubes were created of varying parameters in the XML generation file. The scripts were then run and compared to ensure the files were generated in the same manner. In all cases the cubes were consistent from one CAD program to the next.

Using this inherent flexibility, models can be modified by changing parameters or scripts. Any features or scripts following the changes will reference updated features and accommodate the change.

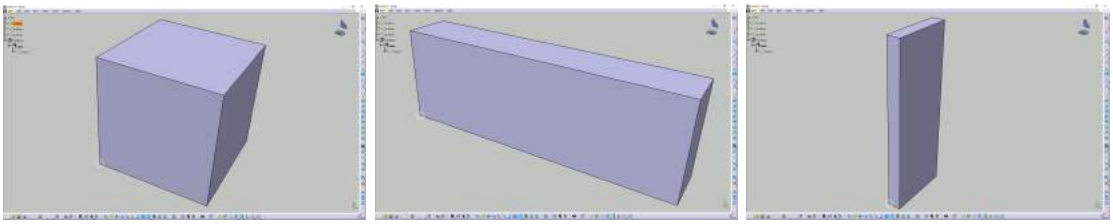

Figure 7: CATIA Cube Script Variations

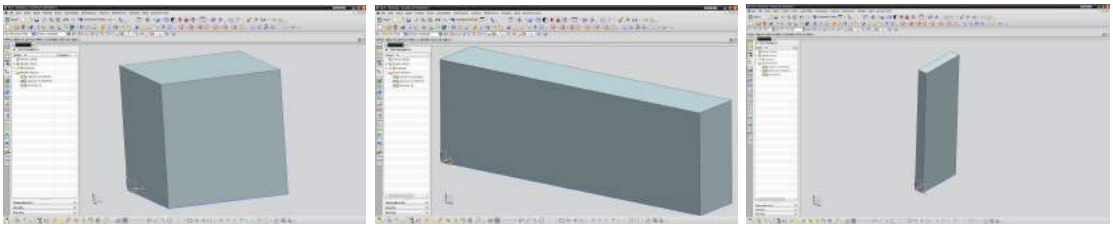

Figure 8: NX Cube Script Variations

4.2 Case Study - Quad-copter

While creating features such as boxes and cylinders can test basic functionality, they do not push the limits of the system. To assess the capabilities of the system and test the functionality of the framework using multiple features in sequence, a basic quad-copter design was created, which is rendered in Fig.9. This design consists of 60 features with additional dependencies on one another. The design was chosen to demonstrate the ability to create features relative to other features while maintaining accuracy of the design. It should be noted that a product of this nature would typically be created using assemblies. The purpose of creating the quadcopter in a single part was simply to test the frameworks ability to handle nested and interdependent features. Research is currently being done to allow the framework to handle assemblies.

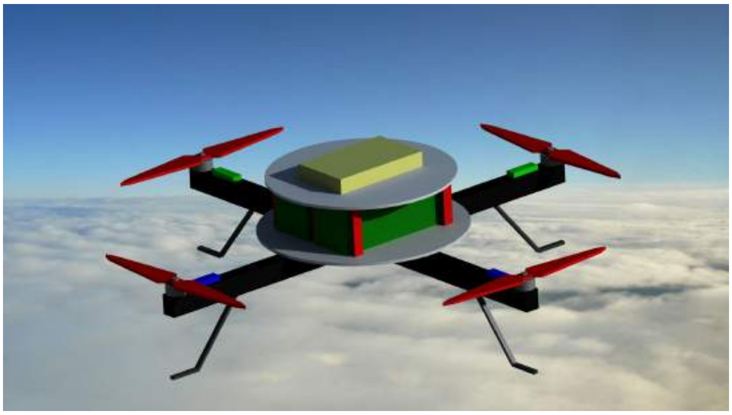

Figure 9: Rendering of the quad-copter model used to compare the neutral XML design framework in NX and CATIA

In other implementations, model notes and governing parameters can be transferred by the same XML scripts. For this case study, we used NX 8 and CATIA V5 as the two CAD systems in which to test the framework. In each system, the model of the quad-copter was generated from the XML format and then measurements were taken and compared in a number of ways to test accuracy. Fig. 10 presents the isometric views of the quad-copter in NX and CATIA after the execution of the neutral XML script.

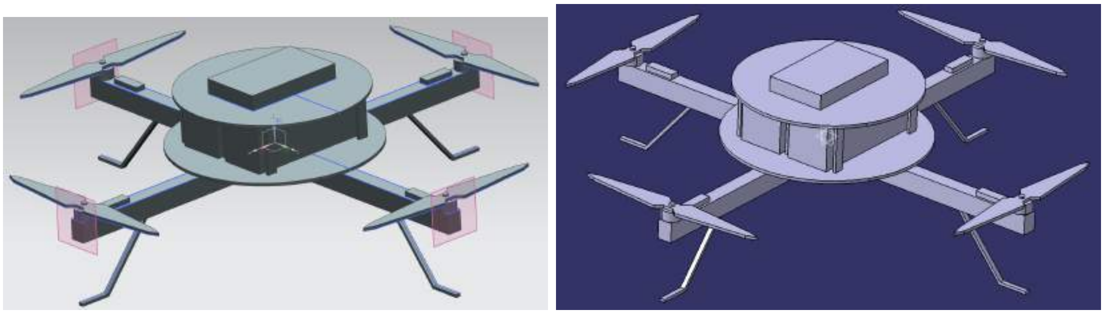

Figure 10: Isometric Views of the Quad-copter in NX (top) and CATIA (bottom)

Although the models look identical across the systems at first inspection, measurements were taken to confirm this initial assessment. Fig.11 and Fig.12 show example comparisons between the two CAD systems for distance and volume measurements respectively. Through these comparisons, it was discovered that the volume measurement and propeller surface area were slightly different, with a 1.09% and 0.96% difference, respectively. However, this could be due to differences in the calculation operations, as well as how the values are computed, since some area's are "overlapping" in one CAD system and may be subtracted, while in the another it is not. Although these can be considered minor differences, they would still need to be addressed or checked adequately and, of course, the total volume should be identical indicating that the overall space of both systems are equivalent.

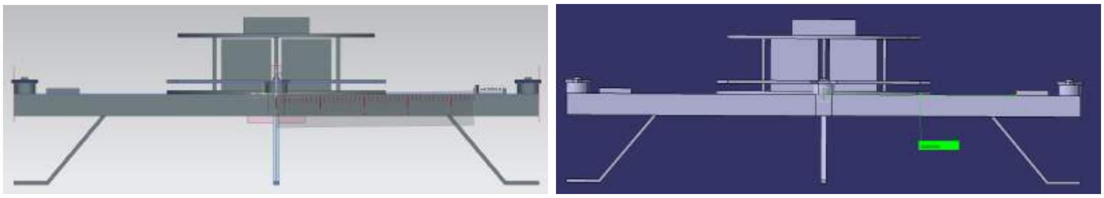

Figure 11: Example processes for comparing distances in both CAD systems

A summary of the comparisons of the various metrics are included and presented in Tab. 1. Although most comparison metrics are relatively close, the overall volume was just over 1% larger in NX as mentioned above. In some models this may be significant and would require revisiting. All other metrics are zero or less than 1%.

Additional inspection into the calculation of volume does in fact suggest that the different CAD systems compute volume differently. In Fig. 13, parts of the legs of the quad-copter are potentially evaluated differently with the overlap considered "extra" volume in the NX CAD system.

4.3 Case Study - Airplane Rib

The second case study explored the generation of a complex airplane rib model. This example shows the ability of the XML framework to execute more complicated operations on a surface. Each airplane rib was

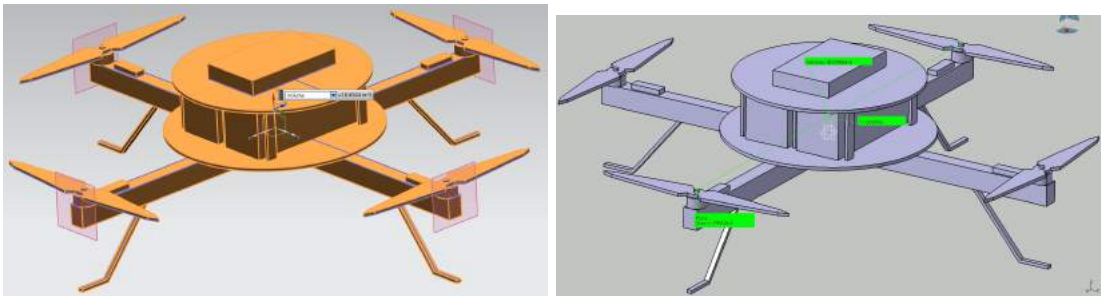

Figure 12: Example processes for comparing volume in both CAD systems

Test Metric (in,in^2, or in^3)	NX	CATIA	Absolute Difference	Percent Difference
Volume	18.6004	18.39972	0.20068	1.091%
Propeller Surface Area	1.8129	1.79553	0.01737	0.9674%
Arm Surface Area	8.8603	8.8113	0.049	0.556%
Leg Angle	49.9967	50.364	0.36724	0.7292%
Motor to Motor Distance	11.5	11.5	0	0%
Center to LED box Distance	4.5	4.5	0	0%

Table 1: Evaluation of Quad-copter Similarity/Accuracy between NX and CATIA models

generated using the same script within the XML framework. Fig.14 shows a comparison between the models generated in both CAD systems. Based on the generated visuals and the measured values, the system remains accurate for complex geometry. For example, the surface area for the 4th pocket of the model was calculated to be 19734.4707 mm^2 in NX while that in the CATIA system was 19734.992 mm^2, for a percent difference of less than 0.003%. For the 7th pocket, comparing the surface areas were 18892.6888 mm^2 and 18893.463 mm^2 for NX and CATIA respectively, for a 0.004% difference.

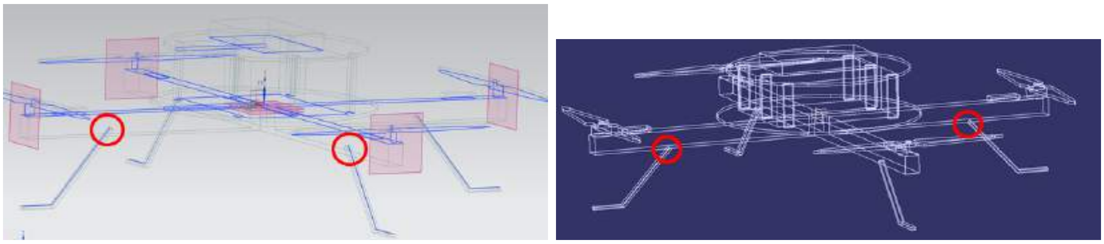

Figure 13: Identified volume differences

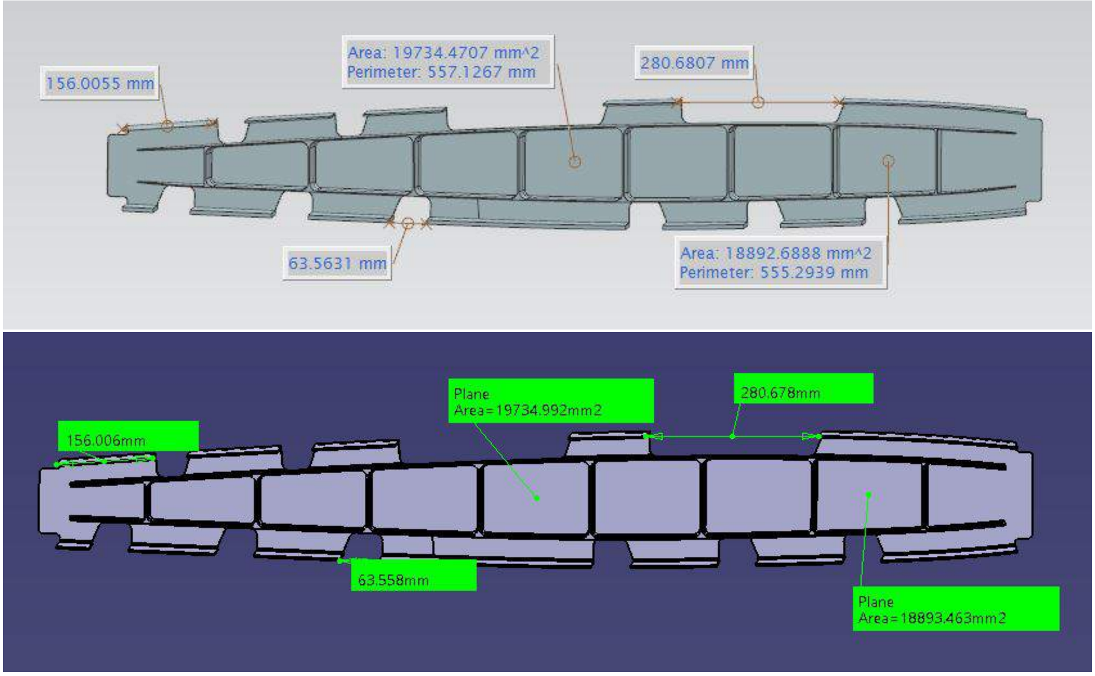

Figure 14: NX (top) vs. CATIA (bottom) Comparison for the Airplane Rib Case Study

5 CONCLUSION

The use of an XML neutral script allows design intent to be carried not only across CAD systems, but versions as well. By incorporating features such as conditionals and the ability to call other scripts, additional flexibility and adaptability can be added. Care must be taken to ensure files can be generated consistently across CAD systems and versions.

In the future when CAD systems adopt AP203 version 2, an alternative implementation is to make a translator that changes our simple neutral format to a AP203 version 2 implemented STEP file. Through this method one could maintain the human readability of our neutral language with the robustness and already implemented translators for AP203.

5.1 Future Research

More research needs to be done to compare accuracy, particularly with splines because of their diverse implementations in CAD systems. More work also needs to be performed to analyze how error values change when parts are scaled. This would be a lot more effective with the inclusion of additional CAD systems to help mitigate differences between CAD internal implementations.

Another area of future research is justifying the ontological choices of our neutral XML design framework. Right now the feature names are based on commonly used names in common CAD programs such as Siemens NX, Dassault CATIA and SolidWorks, PTC Creo, and Autodesk Inventor and Fusion 360. However, a more thorough study must be performed in order to justify and optimize the feature ontology we have selected.

The implementation of our program also requires more research for industrial applications. Some of these include the ability to handle assemblies and translating an existing CAD model into an XML script.

ACKNOWLEDGEMENTS

This research was funded in part by Spirit AeroSystems, Inc.

ORCID

Jonathan E. Sadler, http://orcid.org/0000-0001-8122-7319
John L. Salmon, http://orcid.org/0000-0002-8073-3655

References

[1] Anderson, B.; Ansaldi, S.: Engen data model: a neutral model to capture design intent. Proceedings of the Tenth International IFIP WG 5.2/5.3 Conference PROLAMAT 98, 1–12, 1998.

[2] Barber, S.L.; Junankar, A.; Maitra, S.; Iyer, G.; Devarajan, V.: Experience in development of translators for ap203 edition 2 construction history. Computer-Aided Design and Applications, 7(4), 565–578, 2010. https://doi.org/10.3722/cadaps.2010.565-578.

[3] Bhandarkar, M.P.; Downie, B.; Hardwick, M.; Nagi, R.: Migrating from IGES to STEP: one to one translation of IGES drawing to STEP drafting data. Computers in Industry, 41(3), 261–277, 2000. https://doi.org/10.1016/S0166-3615(99)00052-4.

[4] Bianconi, F.: Towards a procedural cad model for data exchange: problems and perspectives. In Congreso Internacional Conjunto XVII Ingegraf–XV ADM: De la Tradición al Futuro, 1–10, 2005.

[5] Brunnermeier, S.B.; Martin, S.A.: Interoperability costs in the us automotive supply chain. Supply Chain Management: An International Journal, 7(2), 71–82, 2002. https://doi.org/10.1108/13598540210425821.

[6] Choi, G.H.; Mun, D.; Han, S.: Exchange of cad part models based on the macro-parametric approach. International Journal of CAD/CAM, 2(1), 13–21, 2002.

[7] Goh, A.; Hui, S.; Song, B.: An integrated environment for product development using step/express. Computers in Industry, 31(3), 305–313, 1996.

[8] Hoffmann, C.M.; Juan, R.: EREP: An Editable, High-level Representation for Geometric Design and Analysis. In Geometric Modeling for Product Realization, 129–164. North-Holland Publishing Co., Amsterdam, The Netherlands, 1992.

[9] Kim, B.; Han, S.: Integration of history-based parametric translators using the automation apis. International Journal of Product Lifecycle Management, 2(1), 18–29, 2007. https://doi.org/10.1504/IJPLM.2007.012872.

[10] Kim, J.; Pratt, M.J.; Iyer, R.G.; Sriram, R.D.: Standardized data exchange of cad models with design intent. Computer-Aided Design, 40(7), 760–777, 2008. https://doi.org/10.1016/j.cad.2007.06.014.

[11] Lee, H.C.: Evaluation of shape representation methods in step for korean industry. Evaluation, 4(02), 2015.

[12] Manhua, C.; Yanjun, S.; Guojiang, S.: Building a midcad platform for cad model integration and data exchange. In Electrical and Control Engineering (ICECE), 2011 International Conference on, 2046–2049. IEEE, 2011. https://doi.org/10.1109/ICECENG.2011.6057999.

[13] Moreno, R.; Bazán, A.: Design automation using script languages. high-level cad templates in non-parametric programs. In IOP Conference Series: Materials Science and Engineering, vol. 245, 1–9. IOP Publishing, 2017. https://doi.org/10.1088/1757-899X/245/6/062039.

[14] Pratt, M.: Extension of the standard iso10303 (step) for the exchange of parametric and variational cad models. Proceedings of the Tenth International IFIP WG 5.2/5.3 Conference PROLAMAT 98, 5(5.3), 1–12, 1998.

[15] Pratt, M.J.: Introduction to iso 10303-the step standard for product data exchange. Journal of Computing and Information Science in Engineering, 1(1), 102–103, 2001. https://doi.org/10.1115/1.1354995.

[16] Rappoport, A.: An architecture for universal cad data exchange. In Proceedings of the eighth ACM symposium on Solid modeling and applications, 266–269. ACM, 2003. https://doi.org/10.1145/781606.781648.

[17] Staves, D.R.; Salmon, J.L.; Red, W.E.: Associative cad references in the neutral parametric canonical form. Computer-Aided Design and Applications, 14(4), 408–421, 2017. https://doi.org/10.1080/16864360.2016.1257184.

[18] Tessier, S.; Wang, Y.: Ontology-based representation and verification to enable feature interoperability between cad systems. In ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 1163–1174. American Society of Mechanical Engineers, 2011.

[19] Yang, J.; Han, S.; Cho, J.; Kim, B.; Lee, H.Y.: An xml-based macro data representation for a parametric cad model exchange. Computer-Aided Design and Applications, 1(1-4), 153–162, 2004. https://doi.org/10.1080/16864360.2004.10738254.

[20] Zhang, Y.P.; Zhang, C.C.; Wang, H.B.: An internet based step data exchange framework for virtual enterprises. Computers in Industry, 41(1), 51–63, 2000.