Integration of history-based parametric translators using the automation APIs

Byungchul Kim* and Soonhung Han

iCAD Laboratory,
Department of Mechanical Engineering,
Korea Advanced Institute of Science and Technology,
ME 3080, 373-1, Guseong-dong,
Yuseong-gu, Daejeon, Korea
Fax: (82)42-861-6080
E-mail: mir7942@icad.kaist.ac.kr
E-mail: shhan@kaist.ac.kr
*Corresponding author.

Abstract: In this paper, a history-based parametric method was proposed and implemented. However, the size of each translator that exchanges the feature and parametric information tends to be large, due to the implementation of duplicated functions. Furthermore, because a history-based parametric translator uses a procedural model as a neutral format, it requires a geometric modelling kernel to generate an internal explicit geometric model. To solve this problem, we implemented a shared integration platform, TransCAD system, which separates translators from a neutral file. The translators for commercial CAD systems can communicate with only TransCAD. To support communication with TransCAD, we exposed the functions of TransCAD by using Automation APIs developed by Microsoft. Each translator uses Automation APIs of TransCAD either to translate a parametric CAD model from the sending CAD system into XML format or to translate from the XML format into the model of the receiving CAD system. This paper introduces the TransCAD system and its function for the exchange of feature and parametric models.

Keywords: Automation API; CAD model exchange; macro-parametric; translator integration.

Reference to this paper should be made as follows: Kim, B. and Han, S. (2007) 'Integration of history-based parametric translators using the automation APIs', *Int. J. Product Lifecycle Management*, Vol. 2, No. 1, pp.18–29.

Biographical notes: Byungchul Kim received his master's degree from the Department of Mechanical Engineering at the Korea Advanced Institute of Science and Technology (KAIST). He is now a PhD student at KAIST. His research interests include issues related to CAD model exchange, interoperability among CAD applications and geometric modelling.

Soonhung Han received his PhD at the University of Michigan, Ann Arbor. He is a professor at KAIST. His research interests include issues related to STEP, intelligent CAD, geometric modelling and virtual reality in design.

1 Introduction

1.1 Motivation

The parametric technology of commercial CAD systems can generate product models quickly by changing the given design parameters. As design information from parts suppliers is needed to design a given product, it is important to exchange CAD models among various CAD systems. At present, the representative standards for the exchange of CAD models are IGES and STEP AP203 (Kemmerer, 1999). However, because these standards use the B-rep model to exchange the geometric information of products, the parametric information cannot be exchanged. Because the CAD models that are exchanged by using IGES or STEP AP203 do not contain parametric information, it is difficult to modify these models. To resolve this problem, studies have been carried out on the exchange of parametric information.

The pioneering research in this field was the Erep project by Hoffmann and Juan (1993). This research defined the Erep file format as a neutral format that specified modelling features, constraints, and assembly models. The contribution of the Erep project was that it laid the groundwork for the exchange of parametric information.

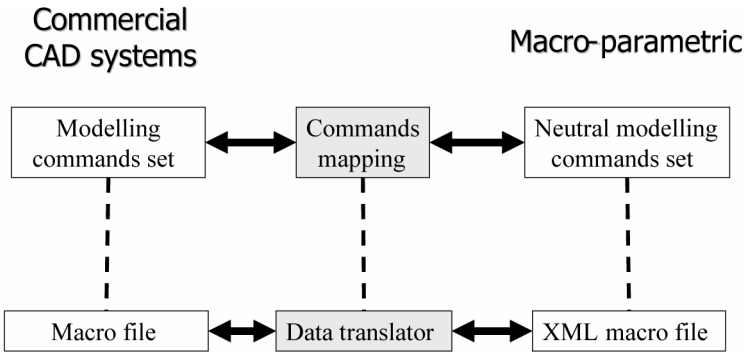
Rappoport (2003) developed Universal Product Representation (UPR) to exchange parametric information. UPR is a union of the data types supported by CAD systems. Rappoport presented the architecture that would utilise UPR as a neutral format for the exchange of parametric information. The striking result of this research was the use of a geometric model when the data types cannot be exchanged by using the parametric information alone. UPR is now being used at Proficiency (Proficiency homepage), a company that provides interoperability software for CAD/CAM/CAE systems.

At present, the standardisation for the exchange of parametric information is being performed in the ISO community. The Parametrics group of ISO TC184/SC4 STEP Working Group 12 is working to archive this standardisation. Five standards for exchanging parametric information are presently under development: STEP Part 55, STEP Part 108, STEP Part 109, STEP Part 111, and STEP Part 112 (Pratt and Anderson; 2001; Pratt, 2004; Pratt et al., 2005).

STEP Part 55 specifies a data model for specifying a procedural and hybrid representation of models using STEP. STEP Part 108 specifies the parameterisation and constraints for explicit geometric product models. STEP Part 109 specifies the kinematic and geometric constraints for assembly models. STEP Part 111 specifies modelling features. STEP Part 112 specifies the modelling commands for the exchange of procedurally represented 2D CAD models. The exchange of parametric models using STEP was tested in the CHAPS project (Stiteler, 2004).

Researchers at the Korea Advanced Institute of Science and Technology (KAIST) have proposed a macro-parametric approach (Mun and Han, 2001; Choi et al., 2002; Mun et al., 2002, 2003; Yang et al., 2003; Yang et al., 2004). The macro-parametric approach is a history-based parametric approach. In this approach, macro information, which is the recording of the modelling commands sequence or the modelling history is exchanged. Figure 1 shows the data exchange model that is used in the macro-parametric approach.

Figure 1 Data exchange model of the macro-parametric approach (Mun et al., 2003; Mun and Han, 2001)



Mapping in the macro-parametric approach comprises two levels. The first is a schema mapping level between the modelling commands set of a CAD system and the neutral modelling commands set. The second is a data translation level between the macro file of the commercial CAD system and the XML macro file. To translate models between CAD systems, the macro information that is generated by a commercial CAD system is translated into an XML macro file, which is again translated into the command sequence of the receiving CAD system. A set of neutral modelling commands has been defined by analysing several commercial CAD systems and this set of commands is used as the schema of a given XML macro file.

To verify the macro-parametric approach, macro-parametric translators for CATIA V5, Pro/Engineer, Unigraphics, SolidWorks and Autodesk Inventor are being implemented. However, problems have been found during the implementation of such macro-parametric translators. This paper introduces the problems that were found during implementation of macro-parametric translators and how these problems could be resolved with our approach. To resolve these problems, in this work, we implemented an integration platform, TransCAD. In addition, to support communication with TransCAD, we exposed the functions of TransCAD by using Automation APIs developed by Microsoft.

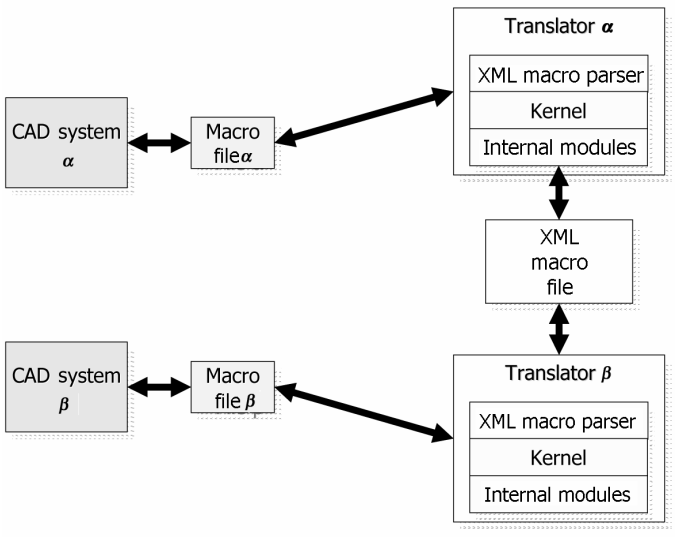
The reminder of this paper is organised as follows. The reminder of this section specifies the problems that were found while implementing macro-parametric translators. Section 2 introduces the proposed solution, and specifies the structure of TransCAD and the macro-parametric translators. Section 3 presents the implementation of the procedure and experiments. Section 4 presents our conclusions.

1.2 Problems with macro-parametric translators

In the previous researches, macro-parametric translators have had architectures similar to that shown in Figure 2. In the architecture shown in Figure 2, when the model is translated from CAD system α to CAD system β, the translator obtains the model data from the macro file of CAD system α. Then, the translator generates an internal explicit model inside the translator using the data from CAD system α to obtain the necessary data. Next, the translator translates the modelling commands of CAD system α to neutral modelling commands using the internal model of the translator. These commands are

saved as an XML macro file by the XML macro parser. The translator of CAD system β reads the XML macro file by using the XML macro parser. After that, the translator regenerates the internal explicit model. From this model and the XML macro file, the translator creates a macro file of CAD system β. Finally, CAD system β reads the macro file, and then generates a parametric model.

Figure 2 Architecture of macro-parametric translators of previous studies



During the sequence of these processes, all of the translators require a geometric modelling kernel to generate an internal explicit model. Furthermore, an XML macro parser is required to read and write XML macro files. Translators also need additional modules to implement their own functions. As a result, the size of the translators gets big from implementing the many required modules.

A translator needs an explicit internal model for geometric calculations. Because the XML macro file or the native macro file of a given CAD system does not have an explicit model, having only a procedural model, the translator should internally create an explicit geometric model from the procedural model. It needs a geometric modelling kernel to create an explicit geometric model from the procedural model.

Furthermore, as commercial CAD systems have their own selection mechanisms, translators should translate the selected entity of the sending CAD system to those of the receiving CAD system (Mun and Han, 2005). For instance, SolidWorks uses 3-D coordinates as its selection mechanism. SolidWorks selects the nearest geometric entity from the coordinates value. On the other hand, CATIA V5 names every entity of the model using the feature data and the sketch data of the model. Such different selection mechanisms should be translated. However, the development of necessary translators is difficult, because the translation mechanisms of given entities are complex.

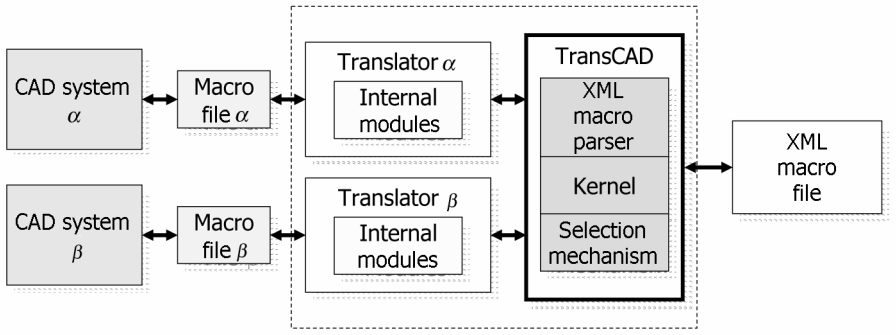
A translator uses neutral modelling commands set as its neutral format. If some of the commands of neutral modelling commands are changed, then all translator source codes should change. This change management of the neutral modelling commands occurs quite often. This increases the development time of each translator.

2 Separation of common functions as a module

2.1 Proposed architecture of macro-parametric translators

To resolve the aforementioned problems, we implemented an integration platform called TransCAD. TransCAD is a layer between the translators and the XML macro file. Figure 3 shows the relation between the translator, the XML macro file, and TransCAD.

Figure 3 Proposed architecture of macro-parametric translators



In the proposed architecture, TransCAD provides what each translator should commonly provide. The translators only need to map the information from TransCAD to the macro information of the receiving CAD system. Most operations can be performed by TransCAD. Such a simple structural change can resolve the aforementioned problems.

Common modules, such as the geometric modelling kernel and the XML macro parser, are transferred from each translator to TransCAD. As a result, the size of the translators reduced. As geometric modelling functions are added to TransCAD, TransCAD can generate an explicit model from a procedural model. Therefore, each translator does not need to create an explicit model from a procedural model. In addition, visual verification and model modification can be performed by TransCAD. Furthermore, in this work, various selection mechanisms were implemented in the TransCAD. Finally, as the system structure prevents translators from directly accessing a given XML macro file, the modification according to the change of the neutral modelling commands set does not propagate to the translators.

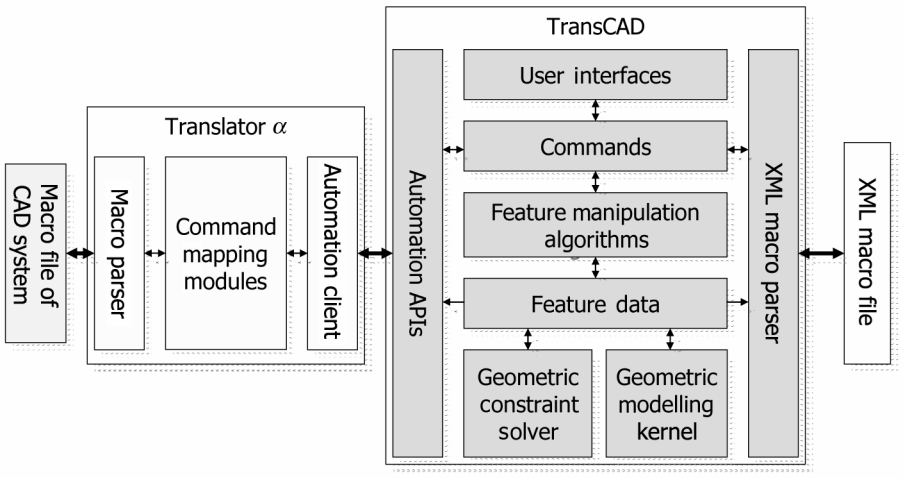
In the new system architecture, an interaction mechanism between the translators and TransCAD should be provided by TransCAD. Accordingly, TransCAD Automation APIs that expose the TransCAD functions were implemented. This Automation, developed by Microsoft, is a COM-based technology that allows developers to create custom applications to automate tasks. In addition, Automation APIs allow applications to expose their functionality to scripting languages and other interpreting languages (Microsoft Corporation, 1996). Automation APIs are used on many commercial applications to expose application functions. CAD systems, such as CATIA V5 and SolidWorks, provide such Automation APIs. However, the functions provided by these systems are not appropriate to our purpose. The commercial CAD systems provide Automation APIs that implement their data structure. However, the neutral modelling commands set have different form from the data structure of the commercial CAD

systems. To overcome this difference, Automation APIs that implement the neutral modelling commands are implemented in TransCAD.

2.2 Architecture of TransCAD and the translators

The system architecture of TransCAD with the macro-parametric translators is shown in Figure 4.

Figure 4 TransCAD architecture and the macro-parametric translators



TransCAD comprises eight modules. The geometric modelling kernel is a module for Boolean operations, filleting, chamfering and shape creation. This module manages geometric and topological data. The geometric constraint solver implements a parametric function. Because commercial CAD systems utilise parametric data, a geometric constraint solver is essential. Without a constraint solver, TransCAD may generate an inaccurate model because of the geometric tolerance. The feature data module is the data structure, which is used in TransCAD. The feature data is that data used by the geometric modelling kernel and by the geometric constraint solver, and it is abstracted as a unit of a feature. Creation and modification of the CAD model in TransCAD are conducted through the feature data. Various feature manipulation algorithms utilise the feature data. To facilitate implementations, the feature manipulation algorithms are wrapped as commands such as the undo/redo, the Automation APIs interface, and the user interface. Modification and creation of the feature data are conducted using the commands. There are also modules for Automation APIs, the XML macro parser, and the user interfaces, which are used for interfacing TransCAD with outside applications.

For interaction between TransCAD and the translators, the TransCAD APIs are implemented by using the Automation. The Automation allows translators to easily access TransCAD APIs, even if the applied translators are implemented using different programming languages. The Automation can also be used to interface with other programs. For instance, PDM system can access model information in the TransCAD using the Automation APIs.

The XML macro parser is a module that translates an internal model of TransCAD to an XML macro file, or vice versa. Additionally, the user interface module enables users to verify or modify TransCAD models.

2.3 Procedures for CAD model exchange

Figure 5 shows the procedure of translating CAD models of CAD system α to an XML macro file. First, the translator acquires information from the native macro file of CAD system α by using the macro parser of CAD system α (process 1). Then, the information is mapped to the TransCAD APIs (process 2), and the mapped APIs are interfaced (process 3). In process 3, the TransCAD APIs are called through the Automation client module. For this process, the macro information is translated to Automation API calls. The calls of the Automation APIs reactivate the TransCAD commands. Then, the commands manipulate the feature data and create an internal model. Finally, when the translators request TransCAD to export the internal model as a XML macro file (process 4), TransCAD generates the corresponding XML macro file (process 5).

Figure 5 Translation process: pre-processing

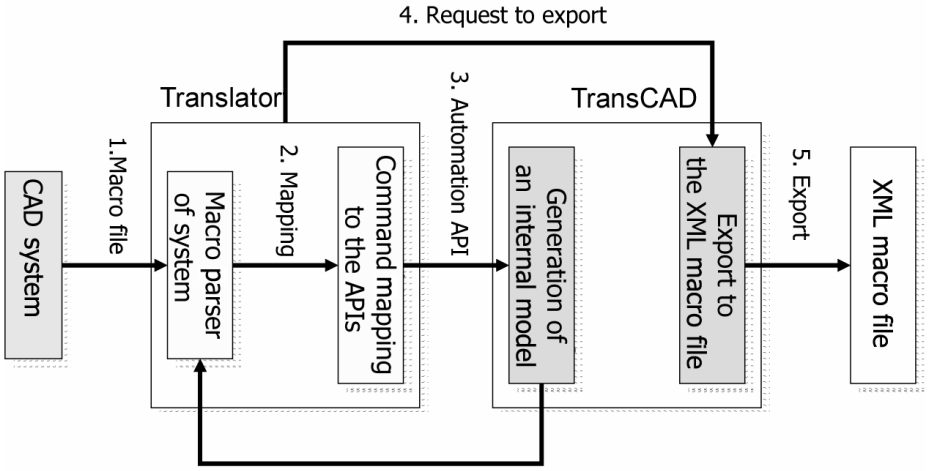
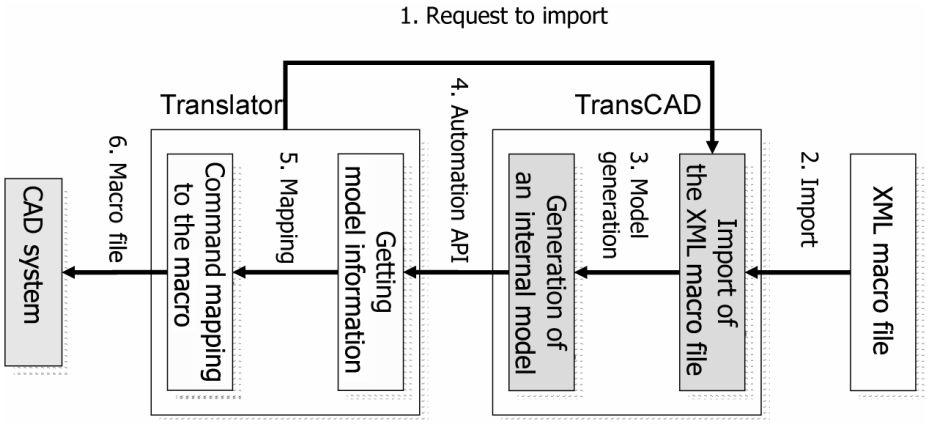


Figure 6 shows the procedure for translating an XML macro file to a CAD model of CAD system β. First, the translator requests TransCAD to import the XML macro file (process 1). Then, TransCAD reads the XML macro file (process 2) and generates an internal model (process 3). Next, the translator calls the TransCAD APIs and obtains the information to be translated (process 4). This information is retranslated to the native macro file of CAD system β by mapping the commands (process 5). Finally, when CAD system β reads the macro file, a parametric model is regenerated inside CAD system β (process 6).

Figure 6 Translation process: post-processing



3 Implementation and experiments

Table 1 shows the implementation environment of TransCAD and macro-parametric translators.

Table 1 The implementation environment

	Software name
Target CAD systems	Autodesk Inventor and Dassult CATIA V5
Operating system	Microsoft Windows XP SP2
Programming language	C++
Geometric modelling kernel	OpenCASCADE 5.2
Geometric constraint solver	2D DCM 46.0
XML parser	MSXML 4.0
GUI implementation	Microsoft Foundation Class (MFC)
Automation API implementation	Active Template Library (ATL)

An XML parser was used to read and write the XML macro file. The Microsoft Foundation Class (MFC) library was used to implement the TransCAD user interfaces. The Active Template Library (ATL) was used to implement the TransCAD Automation APIs. Figure 7 shows key classes and their relations as used in TransCAD in the simplified form of a UML diagram.

A model exchange test was carried out using Autodesk Inventor and Dassult CATIA V5. The L-block shape, which was used by the ISO Parametrics group, is modelled using Inventor. The L-block shape is composed of three features and two sketches (the first column of Table 2). Then, the Inventor translator retrieves the information of the L-block. Meanwhile, the same model is created in TransCAD by using the TransCAD APIs. The created model is saved to an XML macro file, which is composed of neutral modelling commands. For the translation to the CATIA model, the saved XML macro file is imported to TransCAD and the model information in TransCAD is retrieved by using the

TransCAD APIs. Next, the CATIA translator translates the model information into CATIA commands, and generates a CATIA macro file. Finally, the CATIA macro file generates the L-block model in CATIA. Table 2 shows the feature mapping among Inventor features, neutral modelling commands, and CATIA features. Figure 8 shows the implementation result.

Figure 7 Simplified UML diagram of the TransCAD classes

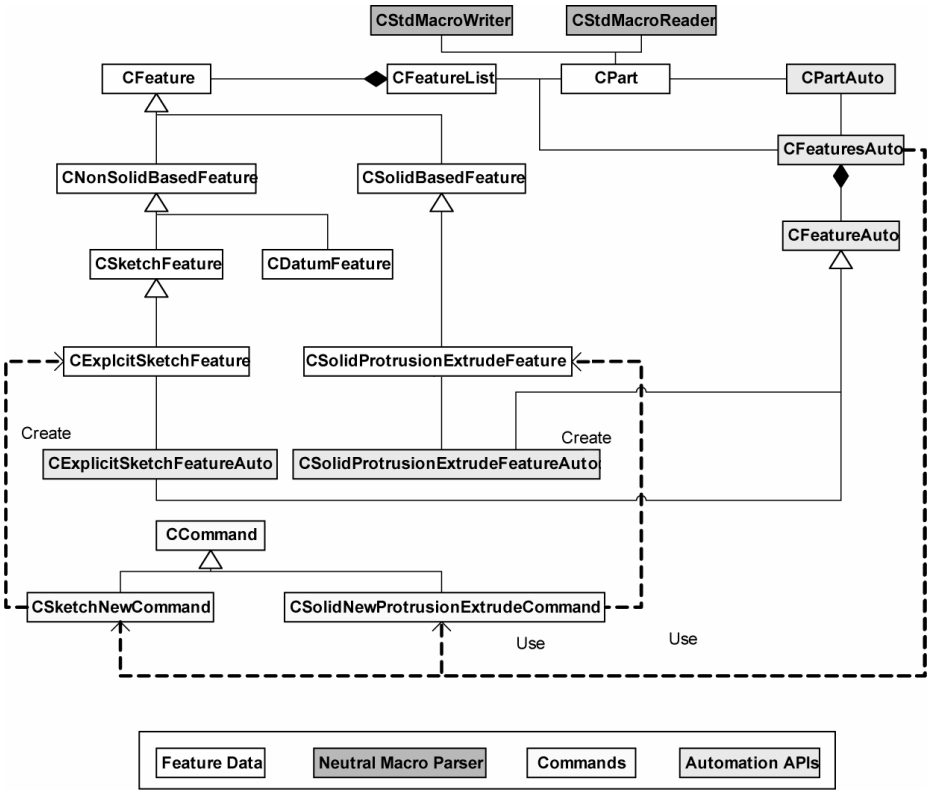
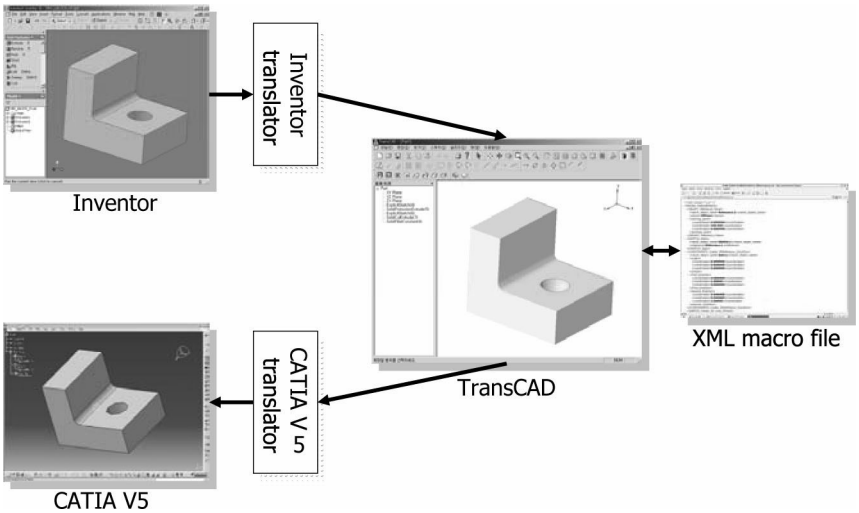


Table 2 The feature mapping among Inventor feature, neutral modelling commands and CATIA features

Inventor	*Neutral modelling commands*	*CATIA V5*
Extrude (with join option)	SOLID_Create_Protrusion_Extrude	Pad
Extrude (with cut option)	SOLID_Create_Cut_Extrude	Pocket
Fillet	SOLID_Operate_Filleting_Fillet_Constant	Fillet

Figure 8 illustrates the considered parametric model being exchanged between Inventor and CATIA. It can be verified that the same model is generated in TransCAD and that the model is saved as an XML macro file. Furthermore, the feature trees in Inventor and CATIA are preserved. This indicates that the parametric information was exchanged.

Figure 8 Implementation result



4 Conclusion

Our study considered how to enhance the implementation of macro-parametric translators. The size of each translator increases because of the many duplicate modules performing various functions, such as generating explicit models, translating selection mechanisms and managing neutral modelling commands changes. As a solution to this problem, TransCAD was inserted between the translators and the XML macro file. Automation APIs were used for communication between the translators and TransCAD.

Because a geometric modeller was implemented and shared by the translators, the effort required to implement each translator was reduced. Furthermore, the translators do not access the XML macro file directly. Therefore, the translator integration could be centred around TransCAD.

A problem remains involving placement of the translators and TransCAD on the same computer. In addition, other applications should also be installed on the same computer to utilise the functions of TransCAD. To achieve this, our future work will involve replacing the Automation APIs with remote procedure calls, such as XML Web Service. Another remaining problem is that OpenCASCADE used in this work does not provide sufficient functions, and it is known to be unstable as compared with ACIS or Parasolid. For this reason, we are considering replacing OpenCASCADE with ACIS.

References

Choi, G., Mun, D. and Han, S. (2002) 'Exchange of CAD part models based on the macro-parametric approach', *Int. J. CAD/CAM*, Vol. 2, pp.13–21. Available at: www.ijcc.org

Hoffmann, C.M. and Juan, R. (1993) 'Erep, an editable, high-level representation for geometric design and analysis', *Geometric and Product Modeling*, North Holland, pp.129–164.

ISO 10303-203 (1994) 'Industrial automation systems and integration – Product data representation and exchange – Part 203: Application protocol: Configuration controlled 3D designs of mechanical parts and assemblies', Geneva, Switzerland: International Organization for Standardization (ISO).

ISO CD 10303-111 (2003) 'Industrial automation systems and integration – Product data representation and exchange – Part 111: Integrated application resource: Construction history features', Geneva, Switzerland: International Organization for Standardization (ISO).

ISO CD 10303-112 (2005) 'Industrial automation systems and integration – Product data representation and exchange – Part 112: Integrated application resource: Standard modelling commands for the procedural exchange of 2D CAD models', Geneva, Switzerland: International Organization for Standardization (ISO).

ISO DIS 10303-108 (2003) 'Industrial automation systems and integration – Product data representation and exchange – Part 108: Integrated application resource: Parameterization and constraints for explicit geometric product models', Geneva, Switzerland: International Organization for Standardization (ISO).

ISO DIS 10303-109 (2003) 'Industrial automation systems and integration – Product data representation and exchange – Part 109: Integrated application resource: Kinematic and geometric constraints for assembly models', Geneva, Switzerland: International Organization for Standardization (ISO).

ISO DIS 10303-55 (2003) 'Industrial automation systems and integration – Product data representation and exchange – Part 55: Integrated generic resource: Procedural and hybrid representation', Geneva, Switzerland: International Organization for Standardization (ISO).

Kemmerer, S.J. (1999) STEP: The Grand Experience, NIST Special Publication SP 939, US Government Printing Office, Washington, DC 20402, USA, July.

Microsoft Corporation (1996) Ole Automation Programmer's Reference: Creating Programmable 32-Bit Applications (Microsoft Technical Reference), Microsoft Press, Book.

Mun, D. and Han, S. (2001) 'Exchange of CAD models using macro parametric approach (in Korean)', *Transactions of the Society of CAD/CAM Engineers*, Vol. 6, pp.254–262.

Mun, D. and Han, S. (2005), 'An approach to persistent naming and naming mapping based on OSI and IGM for parametric CAD model exchanges', Paper presented in the Proceedings of the *5th Japan-Korea CAD/CAM Workshop, Digital Engineering Workshop* (DEWS), Tokyo, Japan, February 24–25.

Mun, D., Han, S., Kim, J. and Oh, Y. (2003) 'A set of standard modeling commands for the history-based parametric approach', *Computer-Aided Design*, Vol. 35, pp.1171–1179.

Mun, D., Kim, B. and Han, S. (2002) 'A hybrid parametric translator using the feature tree and the macro file (in Korean)', *Transactions of the Society of CAD/CAM Engineers*, Vol. 7, pp.240–247.

Pratt, M.J. and Anderson, B. D. (2001) 'A shape modelling applications programming interface for the STEP standard', *Computer-Aided Design*, Vol. 33, pp.489–559.

Pratt, M.J. (2004) 'Extension of ISO 10303, the STEP standard, for the exchange of procedural shape models', Paper presented in the Proceedings of *International Conference on Shape Modeling and Applications 2004*, Genoa, Italy, June 7–9, pp.317-326.

Pratt, M.J., Anderson, B.D. and Ranger, T. (2005) 'Towards the standardized exchange of parameterized feature-based CAD models', *Computer-Aided Design*, Vol. 37, pp.1251–1265.

Proficiency homepage, Available at: http://www.proficiency.com/

Rappoport, A. (2003) 'An architecture for universal CAD data exchange', ACM Symposium on Solid Modeling and Applications 2003, ACM Press, Proceeding, pp.266–269.

Stiteler, M. (2004) 'Construction History and ParametricS: Improving affordability through intelligent CAD data exchange', *CHAPS Program Final Report*, Advanced Technology Institute.

Yang, J., Han, S., Kim, B. and Part, C. (2003) 'A macro parametric data representation for CAD model exchange using XML (in Korean)', *Transactions of the Korean Society of Mechanical Engineers A*, Vol. 27, pp.2061–2071.

Yang, J., Han, S., Kim, B., Cho, J. and Lee, H. (2004) 'An XML-based macro data representation for a parametric CAD model exchange', Paper presented in the Proceedings of the *Int. CAD Conference and Exhibition*, May 24–28, 2004, Pattaya Beach, Thailand, also in *Computer-Aided Design and Applications*, Vol. 1, pp.153–162.