MZ-Platform: a component switching and executing environment

N. Matsuki
Digital Manufacturing Research Center,
National Institute of Advanced Industrial Science and Technology (AIST), Japan

Abstract

We have started a national research and development project of a manufacturing technology information infrastructure called MZ-Platform. MZ-Platform is a component-based software development framework, designed to support manufacturing enterprises to enhance digitalized technologies in a self-innovative way. The major characteristic is that it can switch the running program by changing their components dynamically. It can load JavaBeans components searched by a component mining method, make connections with other active components and execute them. We show some application examples related to design and a manufacturing, such as a CAD (Computer Aided Design) data check tool and manufacturing process-scheduling system.
Keywords: component-based software development, component mining, CAx.

1 Introduction

We have developed a new component-based software development framework called MZ-Platform. This research is financed by METI (Ministry of Economy Trade and Industry) started in 2001 planned for five years research project [1]. The aim of this project is to develop a new tool for manufacturing industry, particularly for small and medium enterprises, that enables engineers themselves to develop wide variety of applications, such as manufacturing process-scheduling system, CAx (represents CAD, CAM, CAE, CAT as a whole) tools, POP (point of production) programs to strengthen their competitiveness.

Information technology (IT) tools become inevitable for manufacturing enterprises for the purpose to enhance their productivity, however, the cost to



Data Mining V, A. Zanasi, N. F. F. Ebecken & C. A. Brebbia (Editors)
© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9

purchase IT tools, to maintain the systems and to train engineers and operators for tools, becomes heavy burdens for small and medium enterprises. Furthermore, the burden becomes much heavier if they try to customize tools to get the most of its ability.

According to our survey, current manufacturing support tools have abundant general functions but are too general for small and medium enterprises because those tools have designed for the purpose to satisfy the diversified request from aerospace, shipbuilding and automobile industries. Particularly die and metal mold design for mechanical parts, of that design small and medium enterprises play a key role in Japan, requires specialized techniques and skills that are difficult to be replaced by current tools whose functions are mainly focused on supporting tasks in the early stage of product design.

We conclude that easy-to-use software development tools, generous supplies of software parts (components) will encourage small and medium enterprises. MZ-Platform is designed for this purpose and we developed basic software modules of MZ-Platform: Component-Bus, Application-Builder, and Remote-Component-Collaboration. Furthermore, we developed several applications. One is called MZ-Checker, which can verify the quality of 3-dimensional geometry data, such as the distance or the break angle of two adjacent free-from surfaces. We regard that MZ-Checker development attested the capability of MZ-Platform as a development tool. In the paper we show the algorithm, system architecture of MZ-Platform, and component mining and building methods implemented in Application-Builder.

2 Background and related work

The idea of component-based development (CBD) is well known in the software engineering [2]. In CBD, software systems are built by assembling components already developed and prepared for integration. Component-based software engineering (CBSE) has become a subdiscipline of software engineering, and lots of researches has been done, however, its concern is mainly on the software architecture and software architecture description languages [3]. On the other hand, commercial CBD tools, such as Visual Basic, .NET framework of Microsoft Corporation, are quite powerful but they do not have special feature directly related to manufacturing application development.

In the development of CAD system, Dassault Systems, the major CAD vendor, adopts component-based development concept called OM component model [2] for their product CATIA. OM component model will be utilized when we use enhance the CATIA functions. In spite that they seem to have ample functions, we do not use them because our component-framework does not based on CATIA.

In order to design a tool to make manufacturing support systems, particularly CAx systems, flexibility of component connection is a key concept because the CAx data represented in object-oriented language, such as Java, become a component.

Data Mining V, A. Zanasi, N. F. F. Ebecken & C. A. Brebbia (Editors)
© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9

3 MZ-Platform: a component-based software development framework

MZ-Platform is fully event-driven component development and execution framework, based on Java and JavaBeans component model technology. The architecture is shown in Figure 1. When users have demands to make a CAx software tool through the use of MZ-Platform, the first thing they have to do is to consult the Component Library (Figure 1). We are preparing many components, such as GUI components, graphic library components, database access components and components for remote access in Component Library. As a result, users may find components that are commonly used in CAx tools in the Component Library.

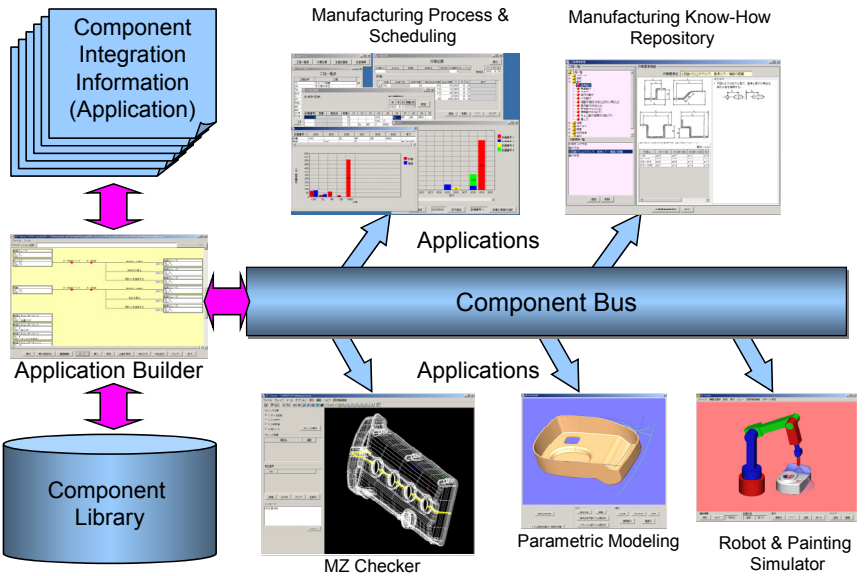


Figure 1: MZ-Platform architecture.

The some functions specific to user-demands have to be designed and developed in the conventional programming environment and these functions are to be made in the form of components. After storing these components to Component Library, users can develop a software tool using Application Builder in the way that they load components from the Library, and "wire" those components. Throughout the process, users can check the actions of components and assembled components (an application) anytime they want through the functions of Application Builder. In this way, we suppose that nonspecialists of software can develop a manufacturing tool in a short time less than half the time it used to be.

Data Mining V, A. Zanasi, N. F. F. Ebecken & C. A. Brebbia (Editors)
© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9

3.1 Component Bus

The core of MZ-Platform is the event-handling module called "Component Bus". Component Bus controls all events from components using the connection objects. When an event from component A to component B occurred, Component Bus receives this event, and if component B is ready to activate, invokes component B and sends the event to component B. To establish a connection between components, Component Bus uses the reflection of Java language, which enables all the methods that component have can be seen through a reflection interface.

Connection objects are designed to propagate events. The concept of connector object plays an important role in the study of Architectural Definition Language (ADL), but few component frameworks currently available have such kind of intermediate object. Almost all of component frameworks connect components directly because of simplicity. We employ connector objects because they allow us abundant flexibility to change the component assembly in dynamic way, namely, to change the connection while the program is running.

It is frequently seen in CAx applications that, according to the change of real number of geometry dimensions, the algorithm (and program) is to be switched. And the problem is that there is no absolute dimension value (the threshold value) when it should switch, but it depends on the environment like the accuracy of computer hardware. For example, at the corner of break lines, we usually define a semicircle to smooth them. But if the break angle becomes near 180 degrees, we can no longer define a semicircle at the corner because the radius goes infinity. Usually, the switch of the algorithm is coded within one component, however, it is difficult to switch correctly as the threshold value depends on the environment. Component Bus can handle this switching much easier because the switch of program can be coded as the assembly of components, and the threshold value can easily be accommodated.

3.2 Component Library

We developed GUI (Graphical User Interface) components, 3-dimensional geometry visualization components based on Java3D, and other component helpful to build manufacturing support applications. These components are serializable JavaBeans and are stored in JAR format files. Component Bus loads components from Component Library using component mining mechanism. If a new component is developed, that has to be stored in Component Library to be utilized as a part of MZ-Platform.

The abundance of Component Library is one of the key issues for MZ-Platform to be an easy-to-use tool and, at the same time, the assistance how to use the components is also important. As this project is planned to continue until 2005, we try to enrich Component Library continuously and create new component mining method.

3.3 Application Builder

Application Builder is the user interface module to define the component assembly. The example of user defined component assembly screen is



Data Mining V, A. Zanasi, N. F. F. Ebecken & C. A. Brebbia (Editors)
© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9

shown Figure 2. Each rectangle of the screen represents a component, and lines between rectangles represent the method invocations, whose name (in Japanese) is shown above the line.

Once the connection between components is defined, Component Bus invokes the components and creates a connector object simultaneously.

Users can select several different modes of Application Builder using buttons at the bottom of the screen, which are application execution mode, screen layout mode and load/save components. During the execution of Application Builder, users can alter the properties of components like colors of the screen background, message string of dialog and similar attributes of components. Also, the usability of the screen layout of an application can be verified while the software is under construction, because the components on Application Builder are all activated.

Application Builder supports a compound component. Several components are assembled into one component; namely, a hierarchical component can also be defined in the same screen.

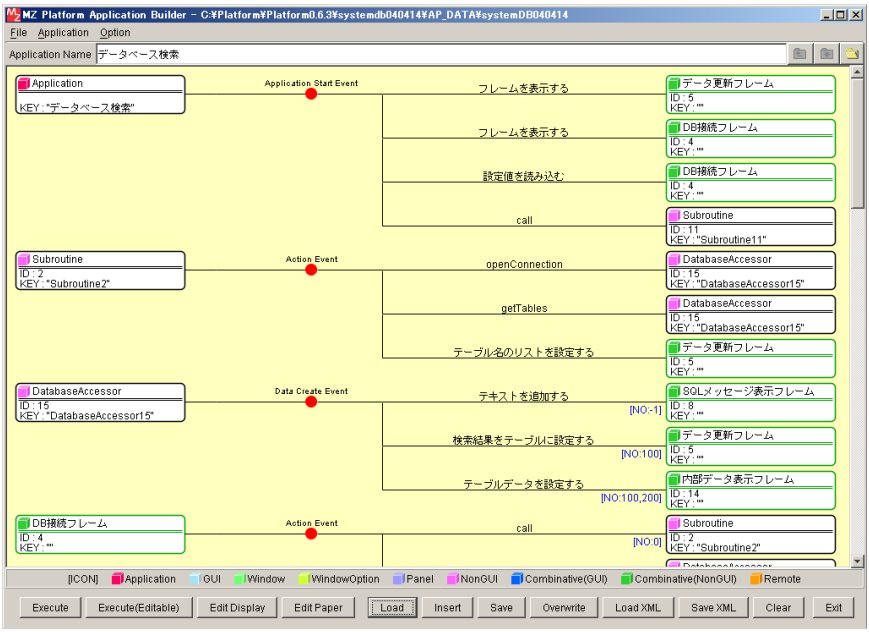


Figure 2: Application Builder GUI example.

Component integration information defined by Application Builder can be stored in the XML (eXtensible Markup Language) format file. In XML file, comments for component and argument explanation (written in Japanese) are also added.

© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9

3.4 Remote component collaboration

Recently, it becomes common that several enterprises are involved to design and manufacture a new product. To support these processes, collaboration of distributed software is one of key features of manufacturing tools. MZ-Platform has the collaboration capabilities to invoke remote component and send a component from one MZ-Platform to the other. Remote component methods (or services) can be found using UDDI (Universal Description, Discovery and Integration) protocols [4]. Messages are sent by SOAP (Simple Object Access Protocol) format, and they can reach the other MZ-Platform site if it is inside the firewall. There are several major protocols: CORBA (Common Object Request Broker Architecture) [5] defines RPC (Remote Procedure Call)-like remote object invocation protocols; Java has RMI (Remote Method Invocation) to use the service of remote Java object. The feature of Remote Component Collaboration is that users can assemble remote components as the same way as local components in their own Component Library. Component Bus creates a XML message if an event for a remote component occurs. A process for remote communication called Broker creates a SOAP message and sends it to the destination Broker. During this procedure, UDDI server provides destination information to Broker, as shown Figure 3.

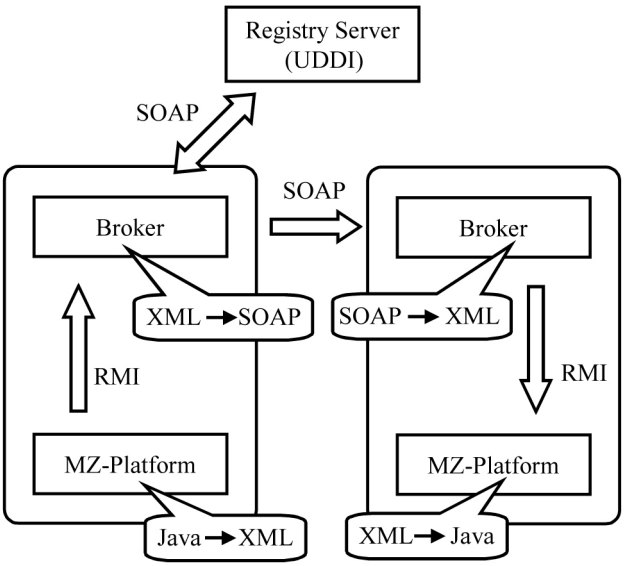


Figure 3: Remote component collaboration.

4 Application

To prove effectiveness and usability of MZ-Platform, we have developed an application called “MZ-Checker” (Figure 4). The main function of MZ-Checker

© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9

is to verify the quality of 3-dimensional CAD data. And we proved that MZ-Platform is capable to develop commercial level programs. We believe that the development period of time, 6 months for whole development in this case, is considerably short compared to the same level of application development. Moreover the easiness to add new function is shown by an example, which is developed and tested on MZ-Checker only two days, whose function is to calculate the angle between a selected axis-vector and surface normal vectors that are finely sampled and to display them on the screen as color-coded surfaces for the purpose of metal-mold model checking.

As a CAx tool, MZ-Checker has lots of features. SASIG (Strategic Automotive product data Standards Industry Group) announces that CAD data that have inferior quality are recognized as a major cause of rework and cost by many automobile industry organizations [7]. JAMA (Japan Automobile Manufacturing Association) [8] and SASIG have published a guideline, for the purpose to prevent inferior quality CAD data to be created, called the PDQ (Product Data Quality) guideline. At the time we developed, MZ-Checker is the only tool that has the full conformity with PDQ guideline Furthermore MZ-Checker can load various types CAD data, such as STEP (AP203, AP214), IGES; it is also used as a viewer of CAD data. As a part of this research project, we started to distribute MZ-Checker for small and medium enterprises.

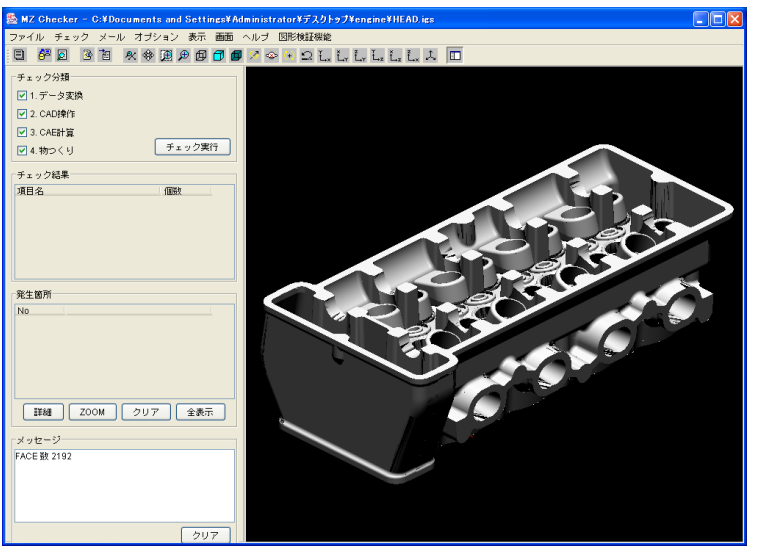


Figure 4: MZ-Checker.

5 Component mining

In general, component mining is a reuse activity that relies on the selection and integration of pre-existing component-rich software base. And from the aspect of

© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9

software reuse, the process of component mining can be divided into three phases as selection, specialization and integration [8, 9, 10]. Application Builder of MZ-Platform has its own standardized interface regulations based on Java language specification, and Application Builder provides integration functions of components with its GUI as shown above. As a result, we introduce our characteristic feature of the selection phase of component mining.

Many researches have been done in selection phase of component mining, or the component retrieval problems; however, they focus on the formalization of selection structure or the process of component selection or query. We, on the contrary, focus on the query support mechanism to suggest an appropriate component candidate, which satisfies the software developer's needs.

In detail design of a software development, the developer is usually given a set of requirements, which are represented as a set of free-form natural language sentences. If any requirement sentence can be transferable to some special format sentences, which corresponds to a component-event-component action, which is as understandable as natural language sentence, then it will be great help for the developer to select components. Moreover, this means that the developer can, not only select components but also specialize and integrate components at one time. Figure 5 shows a simple example. One-digit electronic desktop calculator is developed on MZ Platform. The calculator application is represented by 19 special format sentences that correspond to component-event-component instructions in Application Builder, and also that are easily transformed to natural Japanese language representation.

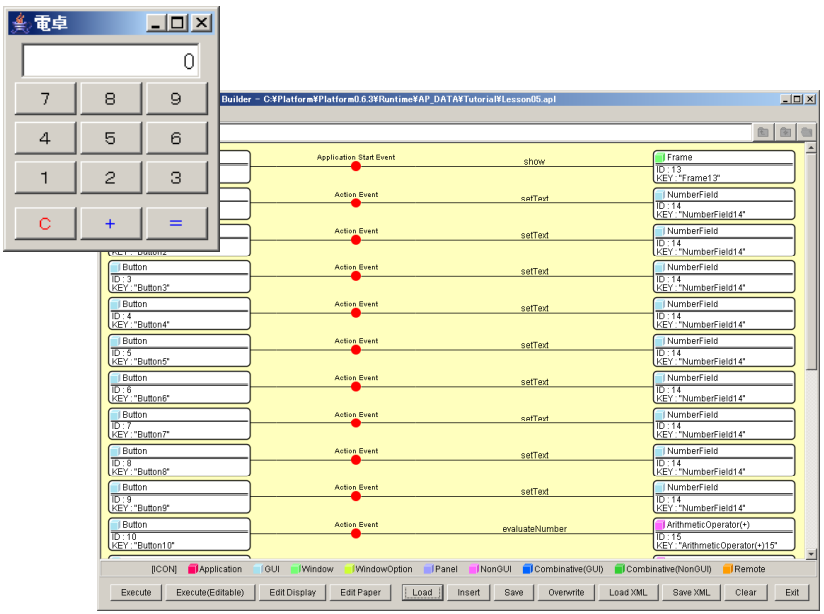


Figure 5: Electronic desktop calculator example.

© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9

6 Conclusions and future work

We have developed a component-based software development framework MZ-Platform. It enables engineers in small and medium enterprises to develop programs specific to their needs. Basic modules of MZ-Platform are designed and developed. There is plenty of scope for improvement. For example, the user-interface of Application Builder can be more simple and friendly to the nonspecialists of programming, and easy-to-use component mining support. As mentioned above, we are going to improve and enrich components from now on.

References

[1] N. Matsuki: A National R&D Project Plan to Establish Manufacturing Information Infrastructure in Japan, Proc. of ITIT Symposium on Development Manufacturing Technology Infrastructure, AIST, (2001), p 6-8.
[2] I. Crnkovic, M. Larsson: Building Reliable Component-Based Software Systems, Attech House, 2002, ISBN 1-58053-327-2, MA, USA.
[3] N. Medvidovic, R. N. Taylor: A Classification and Comparison Framework for Software Architecture Description Languages, IEEE Transactions on Software Engineering, Vol 26, No 1, Jan. 2000.
[4] http://www.uddi.org.
[5] http://www.omg.org.
[6] http://www.sasig.org.
[7] http://www.jama.or.jp.
[8] C. W. Krueger: Software Reuse. ACM Computer Survey, 24(2):131-183(June 1992).
[9] D. Spinellis: Explore, Excogitate, Exploit: Component Mining. IEEE Computer, 32(9):114-116(Sept.1999).
[10] H. Hili, F. Mili, A. Mili: Reusing Software: Issues and Research Directions. IEEE Transactions on Software Engineering, 21(6): 528-562(June 1995).
[11] R. Anderl and R. Mendgen: Parametric design and its impact on solid modeling applications, ACM Proc. 3rd Symposium of Solid Modeling, (1995) 1.



Data Mining V, A. Zanasi, N. F. F. Ebecken & C. A. Brebbia (Editors)
© 2004 WIT Press, www.witpress.com, ISBN 1-85312-729-9