

# Building CORBA Objects With DOS Software Applications\*

Jim-Min Lin<sup>1</sup>, William Chu<sup>1</sup>, Winston Lo<sup>1</sup>, Hongji Yang<sup>2</sup>, Chih-Wei Lu<sup>3</sup>

<sup>1</sup>*Department of Information Engineering and Computer Science  
Feng Chia University, Taiwan, R.O.C.*

<sup>2</sup>*Department of Computer Science, De Montford University, England*

<sup>3</sup>*Department of Information Management, Shu-The Junior College  
of Technology and Commerce, Taiwan, R.O.C.*

*e-mail: jimmy@fcu.edu.tw*

## **Abstract**

CORBA is becoming the most important middle-ware that supports object-oriented and client/server paradigm in distributed computing systems. However the application systems based on CORBA are still scarce up to date. One main reason is that only few CORBA object services have been developed. To have a new CORBA application, a programmer should make efforts to design a program with CORBA interface from scratch. In our previous work [1], a re-engineering approach was proposed to convert RPC-based programs to CORBA objects, which has successfully speed up the development of CORBA applications. However the source code is required in this approach. In many cases, software designers may not get hold of the source code, it will be significant to adapt existing PC application software in binary code mode to the object services under CORBA. Our study is addressing this problem. A graphical parking lot management system, which integrates dBase III and AutoCAD V2.6i under DOS has been implemented to demonstrate the feasibility of our approach.

## **1. Introduction**

Object-oriented and client/server programming paradigm have been important trends for the distributed computing software development. CORBA (Common Object Request Broker Architecture)[2-3], which is proposed by OMG(Object Management Group), is one of the most important middle-ware that supports such a software development paradigm.

However, CORBA applications are still not widely used up to date. One main reason is that only few CORBA object services have been developed and provided for CORBA clients. A user who wants to have a new CORBA application needs design a program with CORBA interface from scratch. The speed of software development is obviously limited. Therefore, it will be significant to have a method to fast building a CORBA object service.

Software reuse is a significant way to improve the speed of software development. There are thousands of existing Application software(AP) for Personal Computer

(PC) users. CORBA supports object services in heterogeneous environments, including MS-Windows and UNIX. Therefore, it will be an efficient and economic way to construct CORBA object services by reusing existing PC APs.

Microsoft COM/DCOM [4] and Active X [5] are proposed for integrating binary-code APs under an MS-Windows environment. However it does not support the integration of DOS APs.

There are several advantages in reusing existing PC APs as CORBA objects:

1. Diverse tools/functions support: Since there has been already plenty of existing PC APs, CORBA users can also have the benefits of these harness APs.
2. Time and money saving: By reusing an existing software as a module in a newly developed software system, the coding of this module is unnecessary. Additionally, many PC APs can be purchased in a low price, therefore, both time and money can be saved.
3. Ensuring software reliability: Most of the software have been well designed by the venders and well tested by millions of customers. Reusing such software may have higher software reliability.
4. Evolving software easily: Although PC APs may evolve with different released versions, they usually have compatible user interface. An old version of a PC AP which had been re-engineered into CORBA can be easily upgraded by a newly released one in our approach.
5. Decentralizing: Most of traditional PC APs are designed and used for single user. Through porting a PC AP on CORBA, the tool can provide services for multiple clients and become a decentralized software.
6. Revaluing existing software: Many PC APs may be retired due to its limited functionality. Reusing these PC APs can revalue them.

In this paper, a CORBA Functional Integration Model (CFIM) is proposed for integrating APs in binary code mode into CORBA. No common data format or source codes of the integrated software is required in our approach. This paper focuses on how to wrap a DOS software as a CORBA object. A graphical parking lot management system, which integrates

---

\* *This research was supported by National Science Council under Grant NSC87-2213-E035-005.*

dBase III and AutoCAD V2.6i under DOS has been implemented to demonstrate the feasibility of our approach.

This paper is organized as following: related works in software integration is introduced in Section 2; The CFIM for binary-code level software integration and the design of a wrapper program for DOS tools are addressed in Section 3; We also give an example – a wrapper for dBase III Plus, to illustrate the design of a DOS wrapper in Section 4. Finally, a brief conclusion is made in Section 5.

## 2. Related Works

Most of the software integration systems use a common data (format) as the foundation of tool integration. We call this type of software integration as a *data-integration* approach. Earlier data-integration systems consist of several software units connected through the data stream flow, e.g., the edit-compile-link-debug loop in Borland C Integrated Development Environment (IDE) [6]. The other type of the traditional data-integration systems emphasizes its diverse information-processing capability, for example, the spreadsheet, table structuring, publisher, graphic-drawing packages in Lotus-1-2-3. Traditional data-integration systems are so tightly coupled that the systems lack the flexibility of adding new components and enhancing the system.

Microsoft uses DDE and OLE mechanisms to support the data integration of software under MS-Window [7]. To support software integration through DDE and OLE, software should be designed with DDE or OLE protocol interface. The software integration using data-integration scheme requires that the integrated software components support common data (format). Unfortunately, if a tool, e.g., AutoCAD or dBase III, does not support common data (format), the integration may be impossible.

The UNIX pipe [8] is an inter-process communication (IPC) mechanism that connects the standard output channel of writer process to the standard input channel of reader process in a UNIX system. Virtual Machine Pipe (VMP)[9,10] has a similar idea with UNIX pipe in performing software integration. Our previous work proposed a VMP based functional integration model, FIM[11,12], for non-CORBA software integration. This paper extends FIM to integrate PC APs into CORBA distributed system.

## 3. CORBA Function Integration Model

The architecture for CORBA Functional Integration Model (CFIM) is shown in Figure 1. There exists a coordinator, several servers running DOS, Windows, CORBA based applications and some clients. Each DOS and Windows AP is encapsulated with a wrapper program. A wrapper program serves as an interface between a non-CORBA based application and other CORBA services. A wrapper is an I/O interceptor with CORBA interface protocol in CFIM. Based on VMP technology, the CFIM provides the solution for data and platform independent software integration. It only requires the understanding of external operations, the format of input commands and output results

of integrated software, but does not require knowing the internal details of program source codes. CFIM provides a framework that supports the following mechanisms: (1) multitasking, (2) tools I/O interception/redirection and IPC, (3) tools coordination, and (4) a functional integration interface specification mechanism. VMP technology provides the primitive operations to support multitasking, tools IPC, and tools coordination. CFIM enhances the VMP technology by offering a high level integration interface specification language, that is used to specify the integration protocol, and a generator, that generates the low level interface to each integrated PC AP according to the specified protocol.

### 3.1 Functional Integration Interface Specification Mechanism

Since a functional integration approach may involve labor intensive low level integration, such as I/O interception and redirection, a high level CORBA Functional Integration Interface Specification Language (CFIISL) is developed to assist the integration process in a cost effective way. An integration specification specified by CFIISL defines how software systems will be integrated. A CORBA Functional Integration Interface Generator (CFIIG) is used to generate the corresponding low level VMPs for integrated tools. CFIISL includes the specification of the definitions of the software-IC and software-circuit.

A software-IC definition describes the integration-related features of an integrated function unit (tool). It contains the following information: (1) Software-IC name, e.g., AutoCAD, (2) Software-IC type, or name of operating system upon which the tool is running, e.g., "MS-DOS", (3) Software-IC filename, or full path name of the executable file of the tool, and (4) Software-IC pins. A software-IC may have one or more software-pins. Each software-pin is an input (or output) operation interface of the tool with certain characteristics. The characteristics of a software-pin is specified by: (a) Pin ID, i.e., name of a pin; it is usually the name of an I/O device name, e.g. "KEYBOARD". (b) Pin type, i.e., the direction of the command/data stream that goes through the pin, either "INPUT" or "OUTPUT". (c) Pin data type, i.e., the data type of I/O stream, e.g. "ASCII\_STRING", "INTEGER16", "INTEGER32", "COORDINATE", ...etc. and (d) Pin entry, i.e., I/O redirector vector or signal type. The pin entry represents the location of an I/O redirector (that is the vector number) if the software-IC is an MS-DOS application, whereas a pin entry is the signal type if the software-IC is a Windows application. For example, the keyboard input-redirector vector for a MS-DOS application is INT 09H; whereas the keyboard message interception is achieved by sub-classing the function *GetMessage*.

A software circuit consists of several channels that connect software-pins among software-ICs. Each channel is basically an one-way communication and defined by the

