# Developing Enterprise Systems with CORBA and Java Integrated Technologies

**Safwan Al Salaimeh, Amer Abu Zaher**
**Irbid National University, Irbid, Jordan**

**ABSTRACT:** The questions of corporate systems development based on integration of CORBA and Java technologies and tools are rated. Here is brought the comparative analysis of using the program models and integration of distributed corporate systems development technologies.

**KEYWORDS:** data processing, corporate systems, distributed process data, distributed process data, distributed process data, client application, information environment.

## 1. The concept of distributed corporate systems

If the data processing makes by a number of geographically-distributed computer powers this corporate system is distributed. As applied to computer network the distributed data processing means technology when the application programs running on one network node provide substantial process and representation of data, and another one network node with database server on it realizes data access and select operations. Besides the distributed process data, a geographically-distributed data warehouses exist, for example when a different databases, tables or their different parts are stored on a different network nodes. At the same time it is necessary during any method of data fragmentation to make the fragmentation clear for data processing. In other words user does not need to know the exact data allocation [DJP99].

The state of design tools distribution determines so that the workstation usually connected to a network and a modern application program represents then control script but not the monolithic piece of code. And this script puts in action a number of geographically-distributed objects, which exist like applets, servlets, scriptlets and other design structures. Using such kind of design tools in banking information systems and e-commerce systems will greatly grow up in the near future because their divisions may function on different hardware platforms, heterogeneous computer networks, under various operation systems, using various programming languages and remote access technologies [SGR99, Mor99, SS00, Tha00].

## 2. The client-server computing models

The most of working corporate systems at present time use the "client-server" technology. There are three basic parts in *client-server models*: *user interface*, which displays the information, realizes graphical user interface functions and forms queries to server in compatible with it format; *functional logic*, which realizes necessary computing, logical comparisons, additional data retrievals and business rules, typical for concrete application program; *database*, which executes data retrievals, modifies data and process them in compliance with received commands. Depending on these component location methods on client workstation and server are possible 2-tier, 3-tier and n-tier client-server corporate systems architecture models.

*2-tier models can be:* with *intellectual client*, when the client workstation contains user interface and functional logic, and the server contains – database; with *intellectual server*, when the client workstation contains user interface, but the server contains functional logic and database. The last model is better protected, has lower communication channel traffic, the transaction realization is good enough and provides user rights restriction properly.

*3-tier models with distributed services* provide independent functional relationship between user interface, functional logic and database. User interfaces and smaller part of functional logic are situated on the client workstations. The greater part of functional logic is situated on the application (middleware) server. The database is situated on the database server. The applications in the model with distributed services are independent; they interact through the computer network with the application server, which interacts with the database server as needed.

*Object-oriented network models of distributed services* join the distributed databases models and the distributed process data. The software of such kind models consist the aggregate of object units, interactive between each other through the computer network using the standard interfaces. This approach lets use of units over and over again and more economically use computer resources. In this model each object depending to situation can be server or client. The object computing architecture based on the distributed network services represents a new highly developing division of computer technologies which is widely uses for distributed corporate systems development.

The distributed process data is extremely perspective for dynamic Web-oriented applications development. The theme of "super thin" clients realization is closely concerned with them, when is using only the browser medium for execution of client application programs. The effectiveness of such kind of a technology for corporate system development is explained by easiness of HTML-browser realization for any operating system. If we take into account that the cost of servicing for one server and cost of servicing for a thousands connected "thick" clients to it are not comparable then we can make a conclusion – successfully realized corporate system is sharply bringing down the overhead charges to service and maintenance.

## 3. The integration technologies for corporate systems' development

Today, the most perspective integration technologies for corporate systems' development with distributed database branches and remote services (functional logic) are ActiveX/DCOM, RMI/CORBA and Enterprise Java Beans/CORBA. The distributed component model of the ActiveX/DCOM [OS98, Pri00] is intended for the application servers' development, registration and distributed application objects management. It's main demerits are, firstly, server works under the MS Windows NT/2000 Server only, and secondly, there is no multi-user access environment for the several application servers co-ordination which are controlling transactions.

The remote access technology RMI [Mor99] is considered that is the best decision for work with remote objects, which lets to develop qualitative Internet-applications. It has all merits of the Java language environment, provides object-oriented programming, ensures high level of security, reliability, free threading, multiplatform and operation system independence support. The RMI is built on the concept in which invocation with its parameters translates in thread byte code for network transmission. The server side implements reverse operation, method invocation and result transmission to the client application.

The common object request broker architecture CORBA [OS98, Pri00] is based on the idea of uniting multivendor applications in the common instrumental environment that lets this technology to work in heterogeneous (hybrid, compound) systems, i.e. there are operational workstations of a different types in a network at the same time, which operate under different OS and serve applications, written on the different programming languages. The CORBA technology with maximum of convenience provides development and maintenance of software environment for distributed corporate systems, which is called the CORBA-application.

Each object request broker ORB producer, which practically is the CORBA, theoretically can propose its own protocol of transport service for data transmission. In this case here is the expanded name of technology, for example CORBA/IIOP, which reflects the name of the IIOP network protocol. For the ORB interoperableness support from different producers the Object Management Group OMG has been developed the unified protocol named GIOP. Its reflection in TCP/IP has got the name of IIOP (Internet Inter ORB protocol) protocol. Using the IIOP protocol any CORBA/IIOP application can interact with other CORBA/IIOP-applications regardless of hardware platform, software environment and operation systems producers.

There are next integrating problems, which were set to the CORBA technology designers: to seize by the common standard all industrial application programs; to describe all interfaces by the unified language tool called IDL; to unite multivendor applications in the common information environment, regardless of a programming languages used for their development, operation systems used by them and a place of their execution. The component approach, named the CORBA technology, which has been designed by the Object Management Group consortium, gives a possibility to use the prepared components in new developing systems. There is no need to develop new systems from the beginning at that, we can use the experience of previous developers, stored in the already developed aggregate of object components.

The interaction of a client application with CORBA-object realizes by the Object Request Broker (ORB). ORB is charged with search of needed object as well as its preparation for the request. To establish a connection with a remote object the client application contains layer called “Bibb” (Stub). The Stub appears as plenipotentiary of the object: using the object interface client applies to Stub as if it was the object itself.

When the Stub receives method invocation, it interacts with “broker” ORB, which is located on the client workstation. The client ORB sends to the network a broadcasting message. One of Smart Agents (intellectual agents) responds to this message. It is installed in the client’s network neighborhood (both the local area network and the Internet). The Smart Agent models network registry which contains registration records of all known to itself object servers. It traces necessary for itself network address of the server and sends a request to ORB object on server station. Thus the Smart Agent finds one of the servers, which can grant access to the requested by the client application object. If were found several of such servers, then the Smart Agent will select the least charged of them.

The "request broker" ORB is installed on the server too. This "broker" addresses to the one more component of the present architecture – Basic Object Adaptor (BOA). With the aid of it the received invocations from client redirect to the object, named Skeleton. The Skeleton makes direct request to the necessary object, creates an instance of the requested object. Whereupon the server ORB "reports" to intellectual agent who has been initiated the connection that the object is ready to execute client's requests.

A broadcasting message is used for it which is addressed to all the "subnet", from which the invocation has been received. If there are many intellectual agents then selects the first responded to request. When the Smart Agent, who made the request is found, the connection between it and ORB object, installs on the client workstation, realizes using the UDP protocol, which uses network resources more solicitously than TCP-connection. The request of a broker, installed on server, through the basic adaptor BOA comes to Skeleton, which puts an invocation parameters into object address space stack whereupon as a matter of fact the invocation (request) is realizing.

The BOA object (adaptor) implements the role of a call filter: with the help of BOA methods, server through the Skeleton can declare some its fields and properties available read only or hide them at all from this object. Inasmuch as data, in CORBA technology, is chains of bytes, the client have to place into invocation buffer its own authorized key in systems, which are protected from "outdoor" penetration of clients.

The "salt" of the CORBA technology is object interface definition using the facilities of IDL language. Whereupon the IDL compiler automatically creates the Stub and Skeleton objects. The data exchange about the object interface between developers implements in the terms of high-level language. At the same time the interface definition IDL compiler translates its text to machine instructions of a concrete computer (client or server). As the result, the high degree of independence of data exchange from hardware facilities of server and client is reaching.

There must be at least one Smart Agent in the client network neighborhood to realize the technology. If data exchange implements in the office local area network, *The Smart Agent* installs on the main computer (fileserver or SQL-server). When data exchange implements in the Internet – *The Smart Agent* installs on the one of the host computers. When the server is installing, the automatic object registration implements in one or more Smart Agents. Thus, the Smart Agent "knows" addresses of all its servers. It lets to raise the systems' reliability. If one of the servers is failed, the Smart Agent will repeat an invocation and if the failure appears again then will switch to another server.

## Conclusion

The realized practical scientific analysis of possibilities and quality of modern integration technologies ActiveX/DCOM, RMI/CORBA and Enterprise Java Beans/CORBA lets make conclusion that these technologies bring together the most progressive methods of object-oriented programming, visual component design and network computing decision making.

## References:

[SGR99] **Dirk Slama, Jason Garbis, Perry Russel** – *Enterprise CORBA*. Prentice Hall PTR, 1999. 368 p.

[RD98] **Robert Orfali, Dan Slama -** *Client/Server Programming with Java and CORBA*. Second Edition. John Wiley & Sons, Inc. 1998. 712 p.

[RD98] **Jason Pritchard -** *COM and CORBA side by side: Architectures, Strategies and Implementations*. Addison Wesley. 2000. 372 p.

[Mor99] **Mike Morgan -** *Java 2 for Professional Developers*. Sams Publishing. 1999. 720 p.

[SS00] **Vivek Sharma, Rajiv Sharma -** *Developing e-Commerce Sites: An Integrated Approach*. Addison Wesley. 2000. 400 p.

[Tha00] **Maghrai Thakkar -** *e-Commerce Applications Using Oracle 8i and Java from scratch*. Que Corporation USA. 2000. 336 p.