Summary and Suggestions
CHAPTER 7

SUMMARY AND SUGGESTIONS

7.1 Summary of the Work

CORBA has a set of protocols that helps to solve the problems associated with heterogeneity and interoperability in distributed systems. It uses IIOP as its communication protocol and Common Data Representation (CDR) as its transfer syntax. Its Interface Definition Language (IDL) exposes the syntax of the contract a server object offers to the client. The stubs generated from the IDL hide the details of communication and facilitate the client and server to interact by packing data, sending and receiving messages and unpacking data. This process for presentation conversion is an expensive part of network communication. The following methods were followed to improve the performance of presentation layer conversion:

- A componentized IDL compiler is designed and implemented to bring about extensibility, flexibility and reuse. The compilation process occurs in three phases. The intermediate phase facilitates the mapping between multiple IDLs, target languages and communication protocols. The design facilitates optimizations at the presentation generation stage and back end of the compiler. It also facilitates CORBA-RMI interoperability and CORBA-DCOM interoperability.

- Novel IPC mechanisms like shared memory, multiple segments of shared memory, parallelizing the operations of shared memory segments using threads and establishment of multiple sockets for communication were implemented. The performance of the system using these IPC mechanisms was evaluated based on RTT, marshal/unmarshal times and networking time. It is observed that using shared memory, networking time and the number of copy operations is reduced. Around 45% of the total time is spent in context switching of the threads. While sending arrays using shared memory, around 50% of the RTT is spent on marshalling. While using sockets, less time is spent on marshalling when compared to networking since all the parameters are transmitted.
Changes were implemented in the data representation formats of CORBA. It includes the following:

♦ Correct allocation of the send and receive buffer space depending upon the size of the data being transmitted. The back end of the compiler analyses the type of parameters being transmitted by an operation and allocates the correct size of buffer for it.

♦ Representation of the Boolean arrays in bit format. On using this bit format for the boolean array, it is found that the throughput increases exponentially.

♦ Removal of alignment at word boundaries. This method eliminates the extra padding bytes. Using this representation, the RTT for the operations using primitive and composite types is much less than RTT in the case of CDR format. It depends on the size of the parameters being transmitted.

♦ The parameters were reordered in the descending order of their size to minimize the number of padding bytes. When the parameters are reordered in the descending order of size, the number of padding bits in CDR is minimized. This is done by regenerating the interface file by the presentation generator to minimize the wastage of space and reassembling the bits in correct order at the server side. It facilitates interoperability as well as reduces RTT. Depending on the data types transmitted as parameters and their ordering, this method achieves an improvement of around 2% to 20%.

♦ Implementation of secure communication between CORBA objects at the link level using a novel symmetric key encryption method. This method encrypts the CDR parameters as a polynomial function. It makes use of algebraic polynomial functional mappings for the plaintext to cipher transformation, which provides for poly-alphabetic substitution. The proposed method transforms the plaintext blocks into 64-bit ciphers achieving an input-output bit transformation ratio of 3:4.

♦ Novel hybrid stub code was generated by optimizing the encoder/decoder software using MOGA to achieve a balance between size and speed of stub code. Here the static and dynamic frequency of the data types in the
interface has been considered. The model has been checked over a marshalling engine which simulates the marshalling functionality of CORBA and a Real-Time CORBA ORB called ZEN ORB which is used for Real-Time Distributed Embedded Systems. In the simulated ORB, the size of the hybrid stub code was almost same as the size of dynamic stub code and time savings in the order of 45% was achieved. A maximum of 54% savings in heap space, 56% in time savings and 11% savings in hard disk space has been achieved in Zen ORB. Using persistent ORB connections, the speed of the hybrid stub code generated was further increased upto 36%.

- Deactivation of CORBA objects is done by the ORB, when the server terminates. This leads to under-utilisation of the resources. Hence a static stub-based framework for creation, destruction and garbage collection of CORBA objects has been designed and implemented. Its interface manage the activation, deactivation and destruction of the CORBA objects when not used by the client.

- CORBA does not support selective inheritance. Hence a static method for selective inheritance has been implemented and tested.

### 7.2 Suggestions for Further Work

The proposed enhancements in the transmission medium and encoding formats can be extended to bring about interoperability between CORBA and other distributed object models dynamically. A dynamic bridge has to be designed for this purpose. This bridge uses DII and DSI instead of SIl andSSI stubs generated by the IDL compiler. The proposed method of achieving static selective inheritance in CORBA can be extended to achieve dynamic selective inheritance in CORBA using DII and DSI.

The proposed method of encryption/decryption can be used to successfully implement a CORBA-based security framework with authentication, authorization, non-repudiation, auditing and key management services. The algebraic polynomial functional mappings can be used in the authentication service with key distribution center which distributes the keys to
client and server securely. In non-repudiation service, the message to be sent is first encrypted using the algebraic symmetric cryptographic algorithm and the session key of the authentication phase. The entire encrypted message is hashed using a one-way hash function to yield the message digest. The first 32 bytes of the encrypted message is encrypted using sender's private key in an asymmetric cryptographic algorithm to yield the digital signature.

Scheduling services do not consider the security levels of the machines on which the tasks are executed. The machines have to be authenticated before the tasks are scheduled for these machines. The proposed security method can be extended to be used in scheduling services.

Virtual Private Networks (VPNs) allow users to communicate and access information over the public Internet and other IP-based networks. VPNs suffer from remote accesses to nodes over which there is no adequate administrative control. Secure virtual connections or 'encryption tunnels' using the proposed APT mappings can bring about the plaintext to cipher transformation.

The input-output transformation ration in the encryption/decryption algorithm that uses algebraic polynomial functional mappings is 3:4. This is because a polynomial function is obtained from the input text. So space has to be allocated for each coefficient of the function in the cipher text. On the other hand if a status vector is obtained from the observed and input vectors, and this is XORed with BaseX, then the size of the cipher text and the input text become equal. Hence an input to output transformation ratio of 1:1 can be obtained.