# Proposal

With the continuing advancement of network technologies, our information society has entered such an era that "network is computer". More and more programs run over a collection of autonomous computers linked by a network and are designed to produce an integrated computing facility. Concerned on supporting the remoting and communication needs between client and server programs or between distributed peers. This project is focused on implementing an infrastructure that allows the distributed program components to communicate over a network in a reliable, efficient and generic way. The goal of the mechanism is to hide the distributed nature of remote objects, i.e. the distributing and remoting part shall be transparent to clients. This infrastructure utilizes proxies to represent remote services, and can serve multiple objects in a parallel way. In particular, this project will port the above infrastructure to Java and shall generate insights into implementing remoting on the Java platform.

Nowadays network programming plays such an important role in application development that it is even hard to find an integrated program all running in a single process on a single machine. The type of program, consisting of complex components running in processes across a network, is very common today. Responsibility distribution has become the trend. In the so-called client/server model, client and server programs run on either end of a networked system. A client program sends requests and receives answers from the server; the server program which receives the requests sends out replies after performing associated computing tasks. Since client and server programs typically run on different computers (this kind of client/server application is defined as distributed application), objects need to be able to talk with one another over the network.They all aim to extend an object-oriented programming system by distributing objects to different processes and hosts. Note that neither of those technologies is transparent to users; clients more or less know about the remote nature of the services. User transparency is an important parameter in software design, it could simplify the end user's programming work, make user applications more organized and easier to maintain. Furthermore, in a traditional Client/Server program, only client sends requests to server, normally it's not possible for a server to question a client. Though in many situations, a server may wish to request further information or help from the client. In the Distributed Objects system, clients and servers are enabled to talk back, i.e. the server and client can exchange roles freely. We don't expect to replace the existing strong remoting technologies by such a small-scale project, but we would like to discover aspects and options of Distributed Objects system design and communication programming on Java.