TDTS04 Computer Networks and Distributed Systems

Real-time Chat and Othello (using Corba)

Authors
Maximilian Bragazzi Ihrén, maxbr431@student.liu.se
Markus Lewin, marle943@student.liu.se

VT1
Version 1.0

1. Audit Descriptions

Ver.	Audit Description	Date
1.0	Written for report.	150306

2. Question 1

Explain what is the purpose of the name server that is being used by the chat system. How would a call by the client have looked like if the name server would not have existed?

The purpose of the name server is to handle on which connected application the requested function is defined.

In the lab, this can be (for example) seen in the client, where the variable “chatImpl” is set to the resolve of the string “Chat”, which is an interface in the .idl file provided. The name server then knows that that particular interface is defined in the server, and sets the “chatImpl”-variable to an object reference to the server.

Therefore, if the name server had not existed, the client would have had to establish a direct connection to the server. Or, if still being a distributed system, the client would have to ask all the connected devices/applications which of them had the requested function.

3. Question 2

The server in the lab is using the callback function. This is one way of implementing asynchronous method calls in CORBA and does not affect how objects are implemented in the server. Please explain why this is the case. Are there other alternative options for how asynchronous calls can be implemented in CORBA?

The callback method does not affect the implementation of objects in the server because the server is not affected by the way the client or anything else is implemented, since the server gets the ChatCallback variable into it's functions, which is all it needs to call the client back.

The current method used in the lab is referred to as “one way”. Another method for asynchrony is referred to as “notification”. This operates on an event basis, meaning that a user can “subscribe” to a certain kind of event, and when the notification service receives an event of the kind you are subscribed to, it tries to send the event your way.

In our implementation it is possible to use a “notification”-based service, as a client might subscribe to the “Chat”-messages when joining the server, and later the “Game”-messages when joining the game.

One would assume that the notification-service would be more efficient in this particular project, but the lab designers might have opted against it due to it being more complex.