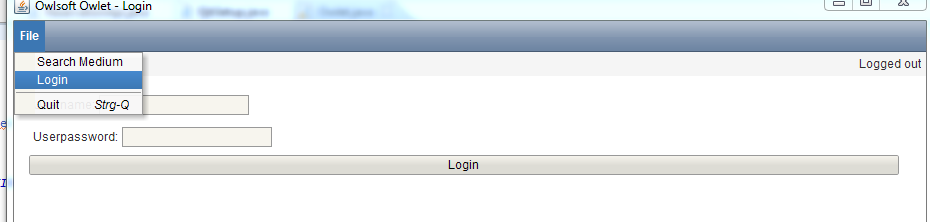
# Timebox 3 LDAP

Nach dem Einloggen in der Owlet Maske überprüft der Server ob der User in der Datenbank existiert. Falls dies nicht der Fall ist wird auf die LDAP Authentifizierung zurückgegriffen.



Da keine Timebox eine Userverwaltung vorgesehen hat, bekommen alle User alle Rollen zugewiesen. Um die JMS Timebox zu testen bekommen User mit einem d oder t im Namen die Operator Rolle zugewiesen.

Bei dieser Timebox traten speziell beim Testen ein Probleme auf. Bei mehrfachen Anfragen an das LDAP wurde die Verbindung abgelehnt und so konnte sich der User nicht anmelden.

# Timebox 4 Corba

Diese Timebox war sehr Aufwendig, da wir eine zu Komplexe Klassenstruktur für Corba haben. Dadurch mussten wir sehr viele IDL Files erstellt werden. Die automatisch generierten Files konnten nicht verwendet werden aufgrund unserer Klassenstruktur.

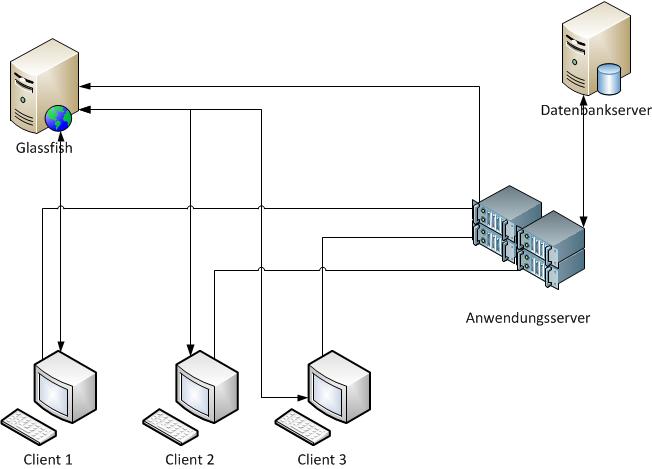
Der Client wurde während der Timebox als Unit Test entwickelt. Für die Abgabe steht aber eine Konsolenapplikation zur Verfügung.

Während der Entwicklung traten viele Fehler auf, weil unsere Interfaces nicht übermittelt werden konnten, um dieses Problem zu umgehen wurden eigene Corba Wrapper entwickelt.

# Timebox 5 Messaging

Die Abbildung zeigt die Abhängigkeiten von der Messaging API. Wenn eine Message auftritt wird diese vom Anwendungsserver an das Topic vom Webcontainer in diesem Fall der Glassfish geschickt. Der Anwendungsserver speichert die detaillierten Informationen.

Die Clients erhalten von dem Webservice eine einfache Textnachricht und wissen so, dass auf dem Server etwas passiert ist. Nach Erhalt dieser Nachricht laden die Clients die in der Datenbank gespeicherten Objekte über den Anwendungsserver



# Timebox 6 EJB

EJB wurde auf den vorhandenen RMI Client aufgebaut. Es konnten die EJB spezifischen Tags gesetzt werden und die Grundlegende Kommunikation funktionierte. Serverseitig hatten wir das Problem, dass intern RMI verwendet wurde, dadurch verlief die Handhabung der Klassen falsch. Dies wurde durch die Local Annotation gelöst, nach setzen dieser Annotation verwendete der Server die richtigen Objekte und nicht die RMI Kommunikation.

Ein ungelöstes Problem ist immer noch. Wenn der Server eine Exception an den Client übermittelt beendet der Server den Socket und somit kann der Client nicht mehr mit dem Server kommunizieren.