# Übersicht

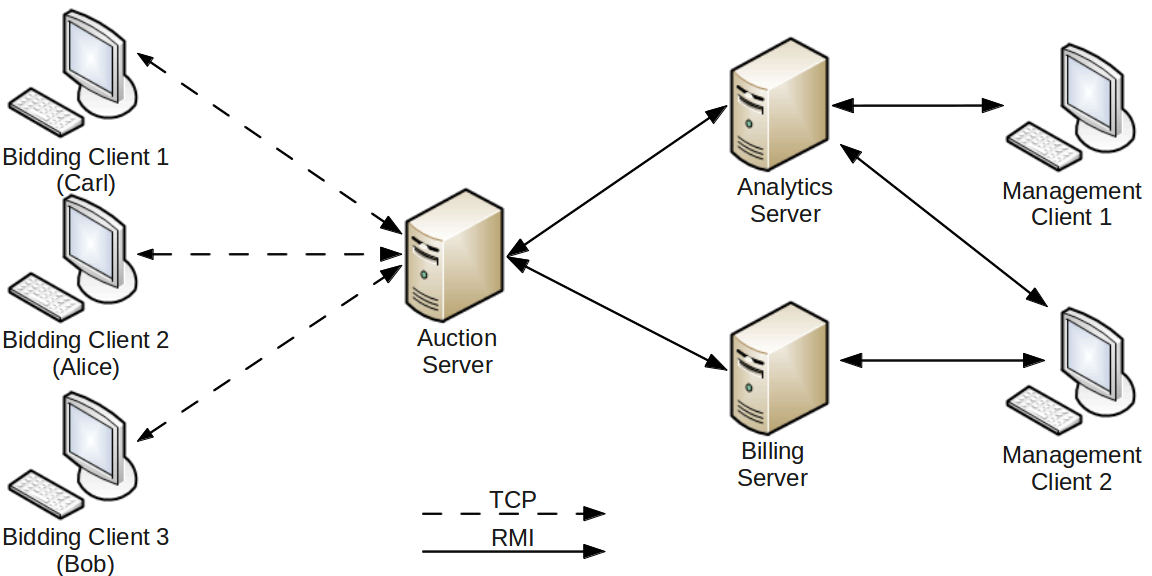


Abb1.: Managment Client + Gesammtstruktur

# UML

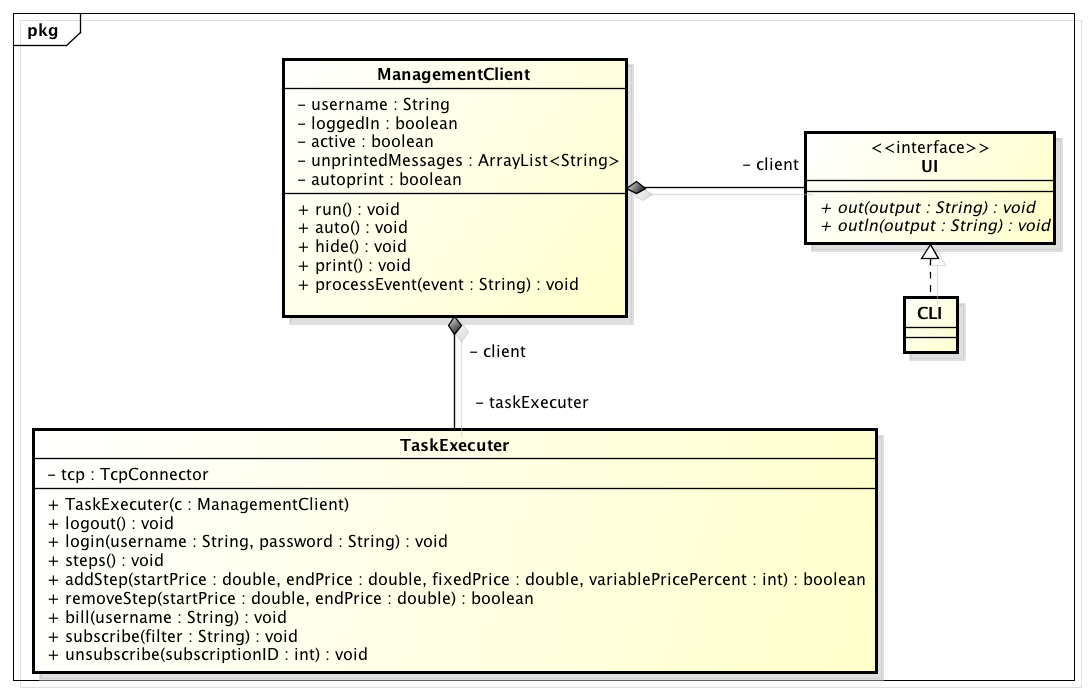


Abb2.: Entwickeltes Klassenkonzept

# Implementierung

Der Management Client hat die Aufgabe, mit Billing undAnalytics Server zu kommunizieren und alle nötigen Befehle zu übermitteln.

Befehle um mit den Billing Server zu aggieren:

* !login <username> <password>: Anmelden mit spezifischen Daten
* !steps: Auflistung aller Preisstufen
* ! <startPrice> <endPrice> <fixedPrice> <variablePricePercent: hinzufügen einer neuer Preisstufe
* !removeStep <startPrice> <endPrice>: Entfernen einer Preisstufe
* !bill <userName>: Zeigt die Gebote des gewählten User. Die Anzeige begrenzt sich auf Auktionen die bereits abgeschlossen worden sind. Zusätzlich werden die Auktions Abgaben angegeben, welche mittels der Abgabeliste ermittelt wird.
* !logout: Setzt den Verbindungsstatus zurück. Um auf Befehle, wie ‚bill‘ zugreifen zu können, muss sich der Managment Client User wieder über das BillingServiceSecure remote Object mit den Befehl ‚login‘ Verbinden.

Befehle um mit den Analytics Server zu aggieren:

* !subscribe <filterRegex>:.Den Management ein bestimmtes Event abonnieren lassen. Mehrfache Abonnierungen möglich.  
    
  Beispiel:

!subscribe '(USER\_.\*)|(BID\_.\*)'  
Created subscription with ID 17 for events using filter '(USER\_.\*)|(BID\_.\*)'

* !unsubscribe <subscriptionID>: Eine Abonnierung eines Events mittls ID wieder abmelden.

Befehlseingaben werden geprüft. Bei unbekannten Befehlen oder falschen Syntax wird der Client über das UI mit vordefinierten Errormessages verständigt.

## Remote Message Management

Der Management Client implemetiert die RMI Methode ‚processEvent‘. Die Methode wird von Analytic Serveraufgerufen, wenn eine bestehente Event-Abonnierung (subscription) mit einen von Server generierten Event übereintrifft.

Es gibt zwei Arten die eingehenden Nachrichten anzuzeigen:

* *Automatic*
* *on-demand*

Dadurch ergeben sich folgende Befehle, um die Moduse zu wählen bzw. zu aggiren:

* !auto: Events werden automatisch, sofort ausgegeben. (none default)
* !hide: Events werden nicht ausgegeben und im Hintergrund gespeichert.
* !print: Gespeicherte Nachrichten (durch Befehl !hide), werden ausgegeben (Reihenfolge)

## Offene Fragen

* Verbindungsaufbau + Eventmanagement
* Management von gleichen Ausgaben, bei mehreren Abonneten (subscriptions). Server oder Clientseitig.
* Implementierung Command Pattern
* Zweck des TcpConnectors

# Ergänzung 10.02.2014

-BillingServer fehlt  eine logout Methode. Ist nötig, da der ManagmentCLient die Option hat ein logout aufzurufen.  
-Die ceatePriceStep Methode hat bei den BillingServerSecure Parameter: double,double,double, double. Die Methode   
addStep im Managment Client hat aber Parameter: double,double,double, int.   
-Die login methode vom BillingServer gibt das Object BillingServerSecure zurück, Ich weis leider nicht was ich  
damit im Managment Client tun soll. Einfach nur CLient Aufrufen und Rückgabe von BillingServerSecure ignorieren?  
-Ich musste leider mit mehr Attributen als im UML aufgezeichnet arbeiten, da es anders nicht möglich war(zb StringRemote-Adresse). Klassen und Methoden-Strukturen wurdn eingehlaten.

-Managment Client muss die RMI Interfaces von BillingServer und Analytics Server implementiern. Fehlt im UML  
-Managment Client benötigt ebenfalls ein RMI Interface (für die processEvent Methode). Fehlt im UML.

## Ressourcen

Abb1: Aufgabenstellung, elearning.tgm.ac.at/mod/assign/view.php?id=20323, zuletzt Aufgerufen am 26.01.13  
Abb2: UMLManagementClients2601.png