DezSys07 – Distributed Pi-Calculator

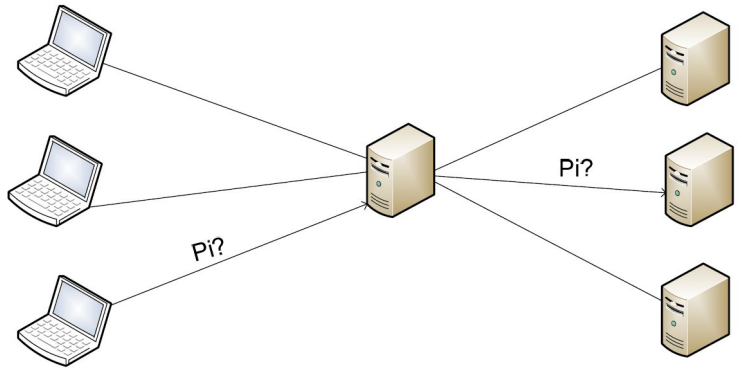
Geyer, Ritter 4AHIT

08.01.2015

Inhaltsverzeichnis

Aufgabenstellung

Distributed PI Calculator



Als Dienst soll hier die beliebig genaue Bestimmung von pi betrachtet werden. Der Dienst stellt folgendes Interface bereit:

```
1 // Calculator.java
2 public interface Calculator {
3         public BigDecimal pi (int anzahl_nachkommastellen);
4 }
```

Ihre Aufgabe ist es nun, zunächst mittels Java-RMI die direkte Kommunikation zwischen Klient und Dienst zu ermöglichen und in einem zweiten Schritt den Balancierer zu implementieren und zwischen Klient(en) und Dienst(e) zu schalten. Gehen Sie dazu folgendermassen vor:

1 Entwicklen Sie ein Serverprogramm, das eine CalculatorImpl-Instanz erzeugt und beim RMI-Namensdienst registriert. Entwicklen Sie ein Klientenprogramm, das eine Referenz auf das Calculator-Objekt beim Namensdienst erfragt und damit pi bestimmt. Testen Sie die neu entwickelten Komponenten.

2 Implementieren Sie nun den Balancierer, indem Sie eine Klasse CalculatorBalancer von Calculator ableiten und die Methode pi() entsprechend implementieren. Dadurch verhält sich der Balancierer aus Sicht der Klienten genauso wie der Server, d.h. das Klientenprogramm muss nicht verändert werden. Entwickeln Sie ein Balanciererprogramm, das eine CalculatorBalancer-Instanz erzeugt und unter dem vom Klienten erwarteten Namen beim Namensdienst registriert. Hier ein paar Details und Hinweise:

- Da mehrere Serverprogramme gleichzeitig gestartet werden, sollten Sie das Serverprogramm so erweitern, dass man beim Start auf der Kommandozeile den Namen angeben kann, unter dem das CalculatorImpl-Objekt beim Namensdienst registriert wird. dieses nun seine exportierte Instanz an den Balancierer übergibt, ohne es in die Registry zu schreiben. Verwenden Sie dabei ein eigenes Interface des Balancers, welches in die Registry gebinded wird, um den Servern das Anmelden zu ermöglichen.

- Das Balancierer-Programm sollte nun den Namensdienst in festgelegten Abständen abfragen um herauszufinden, ob neue Server Implementierungen zur Verfügung stehen.

- Java-RMI verwendet intern mehrere Threads, um gleichzeitig eintreffende Methodenaufrufe parallel abarbeiten zu können. Das ist einerseits von Vorteil, da der Balancierer dadurch mehrere eintreffende Aufrufe parallel bearbeiten kann, andererseits müssen dadurch im Balancierer änderbare Objekte durch Verwendung von synchronized vor dem gleichzeitigen Zugriff in mehreren Threads geschützt werden.

- Beachten Sie, dass nach dem Starten eines Servers eine gewisse Zeit vergeht, bis der Server das CalculatorImpl-Objekt erzeugt und beim Namensdienst registriert hat sich beim Balancer meldet. D.h. Sie müssen im Balancierer zwischen Start eines Servers und Abfragen des Namensdienstes einige Sekunden warten.

Testen Sie das entwickelte System, indem Sie den Balancierer mit verschiedenen Serverpoolgrössen starten und mehrere Klienten gleichzeitig Anfragen stellen lassen. Wählen Sie die Anzahl der Iterationen bei der Berechung von pi entsprechend gross, sodass eine Anfrage lang genug dauert um feststellen zu können, dass der Balancierer tatsächlich mehrere Anfragen parallel bearbeitet.

Gruppenarbeit
Die Arbeit ist als 2er-Gruppe zu lösen und über das Netzwerk zu testen! Nur localhost bzw. lokale Testzyklen sind unzulässig und werden mit 6 Minuspunkten benotet!

Benotungskriterien
o 12 Punkte: Java RMI Implementierung (siehe Punkt 1)
o 12 Punkte: Implementierung des Balancers (siehe Punkt 2)
  o davon 6 Punkte: Balancer
  o davon 2 Punkte: Parameter - Name des Objekts
  o davon 2 Punkte: Listing der Server (dyn. Hinzufügen und Entfernen)
  o davon 2 Punkte: Testprotokoll mit sinnvollen Werten für Serverpoolgröße und Iterationen

Quellen
An Overview of RMI Applications, Oracle Online Resource, http://docs.oracle.com/javase/tutorial/rmi/overview.html (last viewed 28.11.2014)

Zeitaufzeichnung

Schätzung

Beschreibung	Zeitaufwand (Min.)
Projekterstellung + Design	60
Implementierung	420
Testen	240
Dokumentation	120
Gesamt	840 Minuten

Tatsächlich benötigte Zeit

Stefan Geyer

Datum	Beschreibung	Zeitaufwand (Min.)
12.12.2014	Projekterstellung + Design	30
12.12.2014	Implementierung Server	85
14.12.2014	Implementierung Balancer	70
27.12.2014	Implementierung Balancer	90
08.01.2015	Durchführen von Unittests	70
	Gesamt	345 Minuten

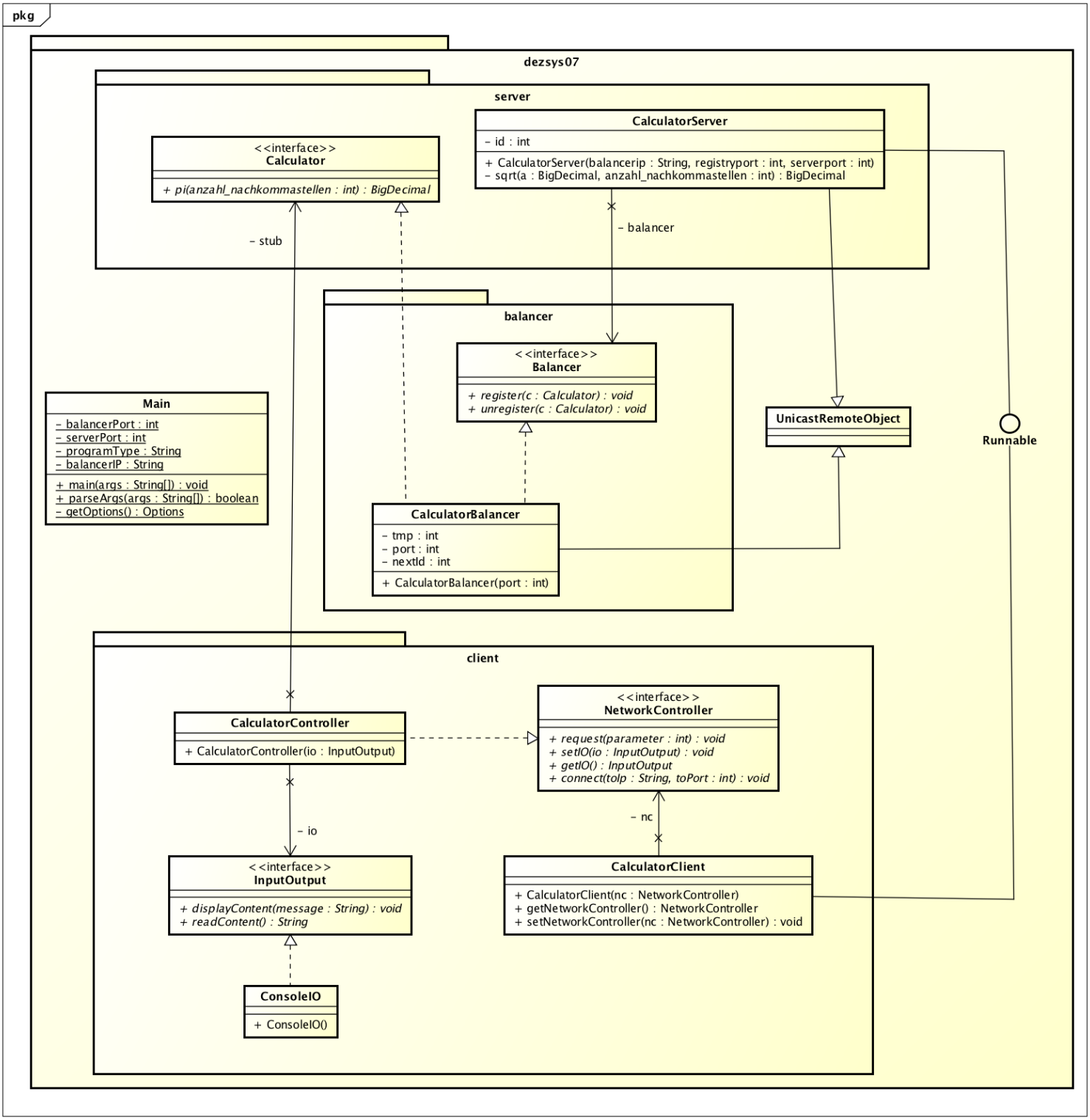
Mathias Ritter

Datum	Beschreibung	Zeitaufwand (Min.)
12.12.2014	Projekterstellung + Design	30
12.12.2014	Implementierung Client	115
14.12.2014	Implementierung Balancer	70
27.12.2014	Implementierung Balancer	90
27.12.2014	Überarbeiten des UML-Klassendiagramms	30
27.12.2014	Durchführen von Systemtests	80
27.12.2014	Bearbeiten des Protokolls	50
	Gesamt	465 Minuten

Gesamt

345 + 465 = 810 Minuten

Designüberlegungen



Implementierung

Client

Es wurde ein Client implementiert, welcher eine Zahl von der Konsole einliest (Anzahl der Nachkommastellen von Pi) und danach eine Anfrage an den Balancer sendet. Die Antwort wird ebenfalls auf der Konsole ausgegeben.

Zuerst wird, falls notwendig, ein neuer SecurityManager initialisiert. Dabei wird ein Policy-File eingelesen, damit der Client über alle Berechtigungen verfügt. [1]

```
if ( System.getSecurityManager() == null ) {
        System.setProperty("java.security.policy",
                System.class.getResource("/java.policy").toString());
   System.setSecurityManager( new SecurityManager() );
}
```

Das Policy-File hat folgenden Inhalt:

```
grant{
   permission java.security.AllPermission;
};
```

Danach führt der Client ein Lookup in der Registry des Balancers aus, um den Stub für den Calculator zu erhalten. Der Stub muss auf das Calculator-Interface gecastet werden. [1]

```
this.stub = (Calculator) Naming.lookup(url);
```

Der Client erhält nun pi, indem er die Methode des Stubs ganz normal aufruft, als würde er ein Objekt lokal besitzen.

```
BigDecimal pi = stub.pi(parameter);
```

Balancer

Der Balancer hat die Aufgabe, alle Anfragen des Clients auf die verfügbaren Server zu verteilen. Ein Client kommuniziert somit nie direkt mit einem Server, sondern immer über den Balancer. Der Balancer implementiert das Interface Calculator, genau so wie alle anderen Server auch.

Zuerst wird, falls notwendig, ein neuer SecurityManager initialisiert. Details dazu siehe oben unter "Client" (es handelt sich um den exakt selben Code).

Der Balancer initialisiert nun eine neue Registry und meldet sich selber bei dieser an. [1]

```
Registry registry = LocateRegistry.createRegistry(port);
registry.rebind("Balancer", this);
```

Nun kann ein Client unter "Balancer" einen Stub auf diesen Balancer erhalten. Dadurch, dass er Calculator implementiert, kann ein Client nun den Stub auf Calculator casten und die Methode pi aufrufen. Wichtig ist, dass die Remote Interfaces (Calculator) auch Remote extenden. [1]

```
public interface Calculator extends Remote { ... }
```

Der Balancer implementiert in der Methode pi() nun den Algorithmus zur Verteilung der Anfragen an alle verfügbaren Server. Dabei werden die Server nach der Reihe durchgegangen. Es kann sich jederzeit ein Server an- oder abmelden.

Server

Der Server bekommt die Anfragen vom Balancer weitergeleitet und beantwortet sie letztendlich. Der Balancer übermittelt das Ergebnis der Anfrage zurück an den Client.

Zuerst wird, falls notwendig, ein neuer SecurityManager initialisiert. Details dazu siehe oben unter "Client" (es handelt sich um den exakt selben Code).

Der Server führt ein Lookup auf den Balancer aus, um sich anschließend an- und wieder abmelden zu können. [1]

```
this.balancer = (Balancer) Naming.lookup("rmi://" + balancerip + ":" +
        registryport + "/Balancer");
```

Danach meldet sich der Server beim Balancer an.

```
this.balancer.register(this);
```

Der Server implementiert Calculator, somit auch die Methode Pi. Nun kann der Server alle Anfragen des Balancers verarbeiten. Wird der Server wieder beendet, so meldet er sich beim Balancer wieder ab.

```
this.balancer.unregister(this);
```

Testbericht

Neben den Unittests wurden zusätzlich folgende Tests jeweils auf 2 verschiedenen Computern durchgeführt: Auf einem Computer wurde der Client/die Clients gestartet, auf dem anderen Computer der/die Server und der Balancer.

Test 1: 1 Client, Balancer, 2 Server

Zuerst wurden der Balancer, die beiden Server und der Client gestartet. Danach hat der Client 3 Anfragen an den Balancer geschickt, welche abwechselnd an die beiden Server übermittelt wurden. Somit ist der Test erfolgreich verlaufen.

Balancer

```
2014-12-27 19:25:53 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
2014-12-27 19:25:55 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator2
2014-12-27 19:26:02 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
```

Server 1

```
2014-12-27 19:15:17 INFO  CalculatorServer:119 - Server with the ID 2 registered at the registry
```

Server 2

```
2014-12-27 19:15:14 INFO  CalculatorServer:119 - Server with the ID 1 registered at the registry
```

Client

```
2014-12-27 19:25:42 INFO  CalculatorController:81 - Connection to Balancer established.
10
2014-12-27 19:25:53 INFO  ConsoleIO:33 - 3.1415926543
30
2014-12-27 19:25:55 INFO  ConsoleIO:33 - 3.14159265358979323846264338329
20
2014-12-27 19:26:02 INFO  ConsoleIO:33 - 3.14159265358979323856
```

Test 2: 2 Clients, Balancer, 2 Server

Zuerst wurden der Balancer, die beiden Server und die Clients gestartet. Danach haben beide Clients abwechselnd Anfragen geschickt, welche vom Balancer korrekt weitergegeben wurden. Die Server haben alle Anfragen korrekt beantwortet, somit ist dieser Test korrekt verlaufen.

Balancer

```
2014-12-27 19:38:47 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
2014-12-27 19:38:51 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator2
2014-12-27 19:39:13 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
2014-12-27 19:39:20 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator2
```

Server 1

```
2014-12-27 19:38:14 INFO  CalculatorServer:119 - Server with the ID 1 registered at the registry
```

Server 2

```
2014-12-27 19:38:17 INFO  CalculatorServer:119 - Server with the ID 2 registered at the registry
```

Client 1

```
2014-12-27 19:38:25 INFO  CalculatorController:81 - Connection to Balancer established.
260
2014-12-27 19:38:47 INFO  ConsoleIO:33 - 3.14159265358979323846264338327950288419716939937510582097
02701938521105559644622948954930381964428810975665933446128475648233786783165271201909145648566923460348610454326650
2014-12-27 19:39:13 INFO  ConsoleIO:33 - 3.14159265358979323846264338327950288419716939937515
```

Client 2

```
2014-12-27 19:38:36 INFO  CalculatorController:81 - Connection to Balancer established.
360
2014-12-27 19:38:52 INFO  ConsoleIO:33 - 3.14159265358979323846264338327950288419716939937510582097494459230781640
02701938521105559644622948954930381964428810975665933446128475648233786783165271201909145648566923460348610454326648
2014-12-27 19:39:20 INFO  ConsoleIO:33 - 3.14159265358979323846264338327950288419716939937510582097494459230781790
```

Test 3: 2 Clients, Balancer, 2-3 Server

1. Starten des Balancers, 2 Server, 2 Clients
2. Client 1 schickt eine Anfrage
3. Client 2 scickt eine Anfrage
4. 1 Server kommt neu hinzu
5. Client 2 schickt eine Anfrage
6. 1 Server wird beendet
7. Client 1 schickt eine Anfrage

Dieser Test ist erfolgreich verlaufen, da die Anfragen korrekt weitergeleitet und verarbeitet wurden.

Balancer

```
2014-12-27 19:59:16 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
2014-12-27 19:59:47 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator2
2014-12-27 20:00:41 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator3
2014-12-27 20:00:58 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
```

Server 1

```
2014-12-27 19:58:51 INFO  CalculatorServer:119 - Server with the ID 1 registered at the registry
```

Server 2

```
2014-12-27 19:58:54 INFO  CalculatorServer:119 - Server with the ID 2 registered at the registry
^C2014-12-27 20:00:50 INFO  CalculatorServer:109 - Server with the ID 2 unregistered at the registry
```

Server 3

```
2014-12-27 20:00:32 INFO  CalculatorServer:119 - Server with the ID 3 registered at the registry
```

Client 1

```
2014-12-27 19:58:59 INFO  CalculatorController:81 - Connection to Balancer established.
25
2014-12-27 19:59:16 INFO  ConsoleIO:33 - 3.14159265358979323846264339
20
2014-12-27 20:00:58 INFO  ConsoleIO:33 - 3.14159265358979323856
```

Client 2

```
2014-12-27 19:59:45 INFO  CalculatorController:81 - Connection to Balancer established.
30
2014-12-27 19:59:47 INFO  ConsoleIO:33 - 3.14159265358979323846264338327
50
2014-12-27 20:00:41 INFO  ConsoleIO:33 - 3.14159265358979323846264338327950288419716939937515
```

Test 4: 1 Client, Balancer, 0-1 Server

1. Starten: Balancer, Server, Client (je einer)
2. Client schickt Anfrage
3. Server wird beendet
4. Client schickt Anfrage

Dieser Test ist erfolgreich verlaufen, da die Anfragen korrekt weitergeleitet und verarbeitet wurden. Sobald kein Server mehr verfügbar war, hat der Client eine entsprechende Meldung auf seine Anfrage erhalten.

Balancer

```
2014-12-27 20:07:32 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
2014-12-27 20:07:43 ERROR CalculatorBalancer:102 - Keine Server zur Beantwortung der Anfrage verfuegbar
```

Server

```
2014-12-27 20:07:10 INFO  CalculatorServer:119 - Server with the ID 1 registered at the registry
^C2014-12-27 20:07:35 INFO  CalculatorServer:109 - Server with the ID 1 unregistered at the registry
```

Client

```
2014-12-27 20:07:14 INFO  CalculatorController:81 - Connection to Balancer established.
50
2014-12-27 20:07:32 INFO  ConsoleIO:33 - 3.14159265358979323846264338327950288419716939937515
150
2014-12-27 20:07:43 ERROR CalculatorController:46 - Anfrage konnte nicht beantwortet werden, da derzeit kein Server verfuegbar ist.
```

Test 5: 1-2 Clients, Balancer, 2-3 Server

1. Starten: Balancer, Server, Client (je einen)
2. Client sendet Anfrage
3. Zweiter Server & Zweiter Client wird gestartet
4. Client 2 sendet Anfrage
5. Client 1 sendet Anfrage
6. Server 1 wird beendet
7. Client 2 sendet Anfrage
8. Client 1 sendet Anfrage

Dieser Test ist erfolgreich verlaufen, da die Anfragen korrekt weitergeleitet und verarbeitet wurden.

Balancer

```
2014-12-27 20:46:01 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
2014-12-27 20:46:16 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator2
2014-12-27 20:46:20 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator1
2014-12-27 20:46:34 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator2
2014-12-27 20:46:42 INFO  CalculatorBalancer:97 - Anfrage eines Clients weitergeleitet an Server //127.0.0.1:1099/Calculator2
```

Server 1

```
2014-12-27 20:45:57 INFO  CalculatorServer:119 - Server with the ID 1 registered at the registry
^C2014-12-27 20:46:30 INFO  CalculatorServer:109 - Server with the ID 1 unregistered at the registry
```

Server 2

```
2014-12-27 20:46:07 INFO  CalculatorServer:119 - Server with the ID 2 registered at the registry
```

Client 1

```
2014-12-27 20:46:00 INFO  CalculatorController:81 - Connection to Balancer established.
25
2014-12-27 20:46:01 INFO  ConsoleIO:33 - 3.14159265358979323846264439
10
2014-12-27 20:46:20 INFO  ConsoleIO:33 - 3.1415926543
28
2014-12-27 20:46:42 INFO  ConsoleIO:33 - 3.14159265358979323846264338400
```

Client 2

```
2014-12-27 20:46:09 INFO  CalculatorController:81 - Connection to Balancer established.
30
2014-12-27 20:46:16 INFO  ConsoleIO:33 - 3.14159265358979323846264338329
40
2014-12-27 20:46:34 INFO  ConsoleIO:33 - 3.14159265358979323846264338327950288419760
```

Quellen

[1] An Overview of RMI Applications, Oracle Online Resource, http://docs.oracle.com/javase/tutorial/rmi/overview.html (last viewed 08.01.2015)