

J.1 Überblick

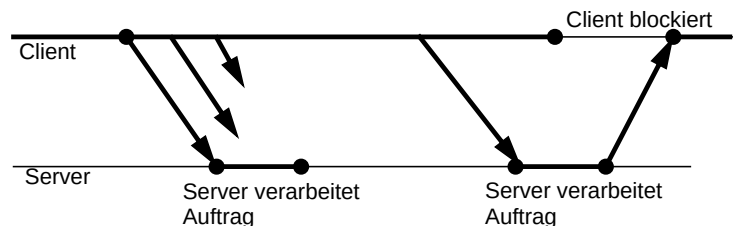
- Asynchroner RPC
 - ◆ Ausprägung ohne/mit Rückgabewert
- Asynchroner RPC am Beispiel von RMI und Java-1.5
- Aufgabe 7
 - ◆ Hinweise zur Aufgabe
- Literatur:
 - ◆ B. Liskov: "Promises: Linguistic Support for Efficient Asynchronous Procedure Calls in Distributed Systems", SIGPLAN, 1988
 - ◆ A.L. Ananda, B.H. Tay, E.K. Koh: "A Survey of Asynchronous Remote Procedure Calls", SIGOPS, 1992

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

J.2 Asynchroner RPC

J.2 Asynchroner RPC

- Asynchroner RPC
 - ◆ Fernaufruf kehrt sofort nach Übergabe der Parameter zurück
- Es gibt zwei Ausprägungen
 - ◆ ohne Rückgabewert / mit Rückgabewert
- Vorteile
 - ◆ Parallelverarbeitung von Client und Server (effizientere Implementierung)
 - ◆ Geeignet für hohen Durchsatz wenn kein Rückgabewert gefordert

Übungen zu "Verteilte Systeme"
© Universität Erlangen-Nürnberg • Informatik 4, 2006

AsynRPC.fm 2006-05-30 12.43

J.2

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

J.3 Asynchrone PRCs mit Rückgabewert

J.3 Asynchrone PRCs mit Rückgabewert

- Problem: Zuordnung des Rückgabewert
- Verschiedene Implementierungskonzepte
 - ◆ Es wird ein spezieller Stellvertreter (handle) bereit gestellt

```
Int_Handle handle = multiply(4,6);
if(handle.poll()){
    y = handle.get();
}
y = handle.get();
```

- ◆ Annahme des Ergebnis über einen speziellen Anweisungsblock (vgl. Interrupt-Routine)
- ◆ Spracheinbettung durch "Future-Variablen"

```
FUTURE int future;
int num = 4711;
future = multiply(4,6);
print(num+future);
```

Übungen zu "Verteilte Systeme"
© Universität Erlangen-Nürnberg • Informatik 4, 2006

AsynRPC.fm 2006-05-30 12.43

J.3

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

J.4 Asynchrone Methodenaufrufe in Java 1.5

J.4 Asynchrone Methodenaufrufe in Java 1.5

```
import java.rmi.*;
import java.util.concurrent.*;

class Client implements Callable {
    Service s;

    public static void main (String[] p) throws Exception {
        new Client();
    }

    public Client() throws Exception {
        s = (Service)Naming.lookup("something");
        ExecutorService es =
            Executors.newSingleThreadExecutor();
        Future f = es.submit(this);
        System.out.println("result:"+f.get());
    }

    public Integer call() throws Exception{
        return s.multiply(5,5);
    } }
}
```

Übungen zu "Verteilte Systeme"
© Universität Erlangen-Nürnberg • Informatik 4, 2006

AsynRPC.fm 2006-05-30 12.43

J.4

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

J.5 Aufgabe 7

J.5 Aufgabe 7

- Erhalt der Schnittstellendefinition

```
interface MultiplyServer {
    int multiply(in int val1, in int val2);
};
```

- Erweiterung der Stub-Schnittstelle durch Methoden für asynchrone Aufrufe

```
struct MultiplyServerStub {
    int16_t multiply(int16_t val1, int16_t val2);
    Future<int16_t> async_multiply(int16_t val1, int16_t val2);
};
```

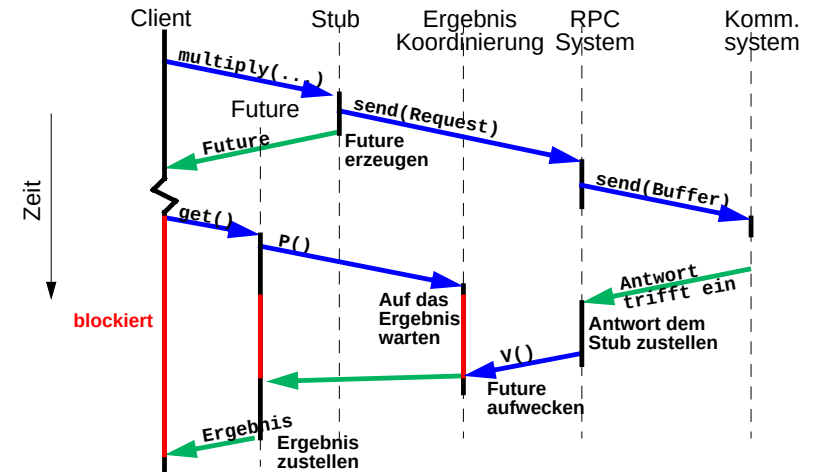
- Abfrage des Ergebnis mit Hilfe eines Platzhalterobjektes

```
template <class t>
struct Future {
    t get(); /* wait for result (blocking!) */
    bool poll(); /* test whether the result is available */
};
```

2 Asynchroner Aufruf

J.5 Aufgabe 7

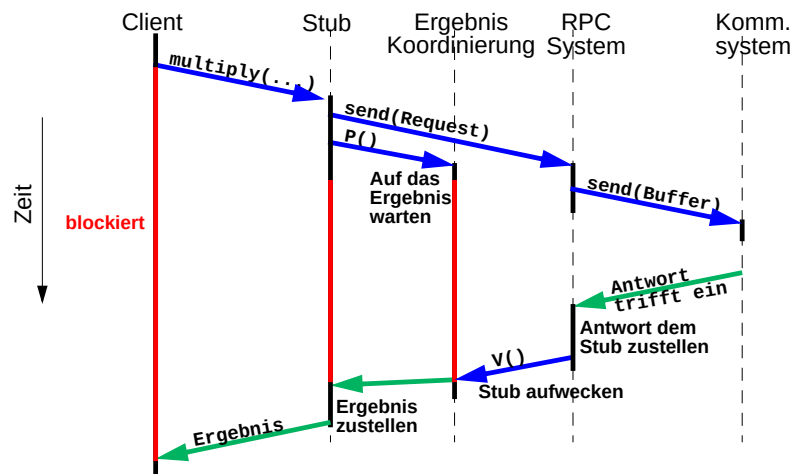
- Entkopplung durch Ergebnisplatzhalter (Future-Objekt)



1 Synchroner Aufruf

J.5 Aufgabe 7

- Ausgangssituation (Aufgabe 6)



2 Asynchroner Aufruf

J.5 Aufgabe 7

- Alternativer Ablauf

