Übungsaufgabe #7: Asynchroner RPC

21.06.2005

In dieser Aufgabe soll das RPC-System um eine Unterstützung von asynchronen Fernaufrufen mit Rückgabewert erweitert werden.

Um die Möglichkeit zu haben asynchronen Fernaufrufe abzusetzen sollen die erzeugten Stubs erweitert werden. Für jede, in der Schnittstellenbeschreibung geforderte, Methode soll nun im Stub zwei Methoden erzeugt werden. Neben der bisher erzeugten Methode mit der geforderten Signatur soll eine weitere mit dem Namenszusatz `async_` generiert werden. Diese Methode erhälten die selben Parameter, liefern jedoch ein Objekt vom Typ `Future` zurück.
Beispielsweise soll aus folgender Schnittstellenbeschreibung:

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

folgende zwei Methoden erzeugt werden:

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

Im Gegensatz zur ursprünglichen Methode, welche blockierend ist und erst zurückkehrt, sobald ein Ergebnis vor liegt, soll die `async_`-Methode sofort zurückkehren. Das Warten auf ein Ergebnis soll in eine Methode des `Future`-Objekts verschoben werden. `Future` stellt dabei ein Platzhalter für das Ergebnis dar. An ihm kann man prüfen, ob schon ein Ergebnis vorliegt und man kann daran blockieren um auf das Ergebnis zu warten. Das `Future`-Objekt soll folgende Funktionalität bereitstellen:

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

Da das Ergebnis, welches das `Future`-Objekt repräsentiert, von der Signatur der ursprünglichen Methode abhängt, kann man das `Future`-Objekt, wie oben, als Template implementieren. Alternativ kann man auch spezielle `Future`-Objekttypen (z.B.: `int16Future`) durch den Stubgenerator erzeugen.

Abgabe: bis 05.07.2005 16:00 Uhr