CARL VON OSSIETZKY universität OLDENBURG

Entwicklung eines Frameworks zur Simulation von Komponentennetzwerken

Proposal eines Individuellen Projektes

Marko Hoyer
Hans-Fleischer Straße 29-31
26133 Oldenburg
Marko.Hoyer@informatik.uni-oldenburg.de

12. Mai 2004

Betreuer: Jun.-Prof. Dr. Ralf Reussner
Dipl.-Wirtsch.-Inform. Steffen Becker

Inhaltsverzeichnis

1 Einleitung

Das hier vorgestellte Individuelle Projekt befasst sich mit der Simulation des Laufzeitverhaltens von Komponentennetzwerken. Hierbei steht die Analyse eines Netzwerkes vor dessen Implementierung im Vordergrund. Das jetzt folgende praktisches Beispiel soll im Vergleich zur mathematisch exakten Berechnung den Sinn der Simulation verdeutlichen.

1.1 Simulationsumgebung vs. mathematische Berechnung

Zur Entwicklung eines Systems sollen eine Reihe von Komponenten eingesetzt werden, welche in Kombination die Dienste des Systems implementieren. Die Komponenten sind hierbei durch ihre Konnektoren, ihre Dienste, einen inneren Kontrollfluss und ihre Laufzeiteigenschaften gekennzeichnet. Weiterhin ist die Verknüpfung der Komponenten untereinander durch den Entwickler des Systems vorgegeben.

Ziel der Simulation soll die Auswertung der Dienste des Gesamtsystems sein. Hierbei spielt neben der Antwortzeit der einzelnen Dienste des Systems die Identifizierung von 'Flaschenhälsen' innerhalb des Komponentennetzwerks eine Rolle.
Besteht das System ausschließlich aus linear zusammenhängenden Komponenten, deren Dienste der Reihe nach von einer ankommenden Anfrage durchlaufen werden, so gestaltet sich die Analyse des Systems recht einfach. Problemfälle lassen sich anhand der Einzelzeiten identifizieren und die gesamte Antwortzeit kann durch Addition der Einzelzeiten relativ leicht ermittelt werden.
Beinhalteten die Komponenten innerhalb des Systems jedoch Verzweigungen, so müssen alle sich ergebenen Pfade einzeln berechnet und mit einer bestimmten Gewichtung gewertet werden.
Weiterhin ergibt es sich in Systemen häufig, das bestimmte Dienste einer Komponente von mehren Diensten des Gesamtsystems benötigt werden. Ein Beispiel hierfür sind Dienste, die Daten aus einer Datenbank auslesen. Hierbei geht die Analyse des Systems über die Pfade hinaus. Es müssen nun die Antwortzeiten der Dienste der Komponenten dynamisch auf die Anzahl zu einem Zeitpunkt ankommender Anfragen angepasst werden.
Übersteigt die mathematisch exakte Analyse des Systems bereits hier die Grenzen des sinnvoll machbaren, so erscheint die exakte Berechnung bei der Verteilung der einzelnen Komponenten auf verschiedene Prozessoren als unmöglich.

An dieser Stelle kann die Simulation ansetzen. Es werden nun nicht mehr

die mathematisch exakten Gegebenheiten berechnet sondern anhand der Simulation eines Modells mit einer bestimmten Ungenauigkeit ermittelt. Weiterhin lassen sich bei der Simulation 'Flaschenhälse' identifizieren, die bei der mathematischen Berechnung nur schwer zu ermitteln sind. Hierzu kann beispielsweise einfach das Zeitverhalten einer Anfrage Dienst für Dienst aufgezeichnet und hinterher ausgewertet werden. Bild 1 zeigt schematisch eine solche Simulation.

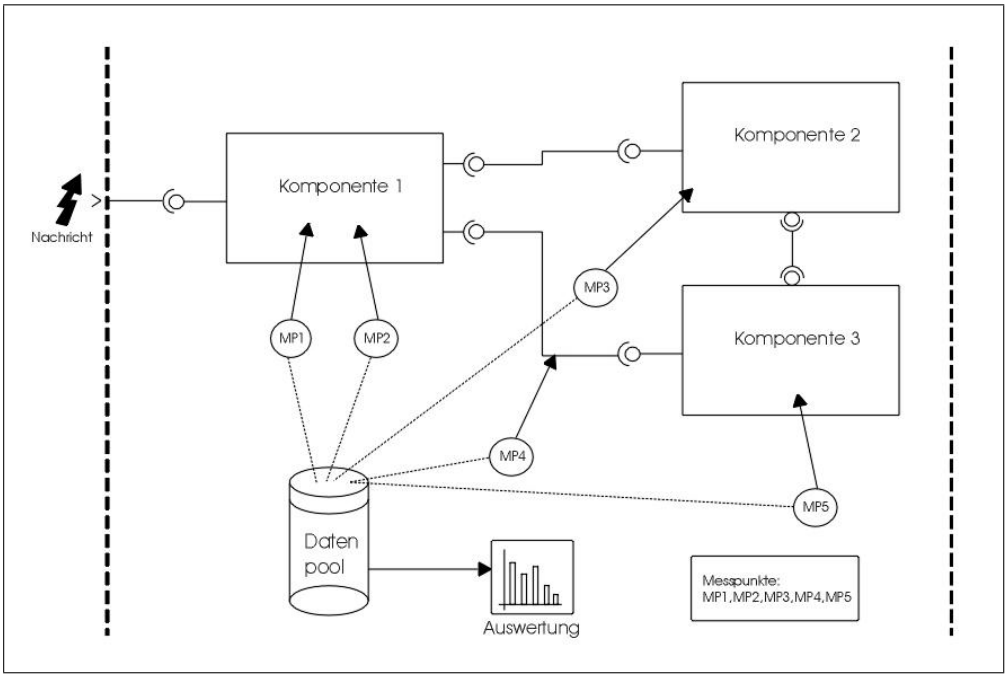

Abbildung 1: Schematische Darstellung der Simulation

2 Ziel des Individuellen Projekts

Da, wie in der Einleitung (siehe Kapitel 1.1) verdeutlicht, die exakte Berechnung eines Komponentennetzwerks zu komplex oder gar unmöglich ist, soll im Rahmen dieses Projektes eine Simulationsumgebung zur Analyse und Auswertung einer bestimmten Konfiguration von Komponenten entstehen. Hierzu setzt sich das Projekt die Implementierung der Simulationsumgebung und die Bereitstellung der Infrastruktur zur Nutzung dieser Umgebung in Form eines Frameworks zum Ziel.

Die Erstellung des Frameworks teilt sich in mehrere Inkremente, die sequentiell entworfen, entwickelt und getestet werden sollen. Der Inhalt dieser

Inkremente ist im Folgenden beschrieben.

2.1 Entwicklung der Basisfunktionalität

Das erste Inkrement befasst sich mit der Erstellung der Basisfunktionalität der Simulationsumgebung. Hierzu gehört die Entwicklung von Strukturen zur Repräsentation des Komponentennetzwerkes unter Integration der Klassenbibliothek zur statischen Modulierung des Kontrollflusses einer Komponente (siehe Kapitel 3.2). In diese Strukturen werden zunächst statisch Verzögerungszeiten der Dienste der Komponenten und der verwendeten Konnektoren eingearbeitet. Diese werden dann im zweiten Inkrement um gewisse dynamische Aspekte erweitert (siehe Kapitel 2.2).
Um die Strukturen zu einem Netzwerk zusammenzusetzen, benötigt das Framework einige Komponenten, deren Entwicklung ebenfalls in dieses Inkrement fällt.

Weiterhin werden in diesem Abschnitt des Projektes die Datenstrukturen, welche als Nachrichten die Komponenten durchlaufen und Informationen (z.B. das Eintreffen oder Verlassen eines Dienstes) sammeln, entworfen. Diese werden in einem Pool gesammelt und können nach der Simulation ausgewertet werden. Methoden zur Auswertung der Daten werden zum Teil im dritten Inkrement entwickelt.

Zur Synchronisation mehrerer Nachrichten wird eine Uhr benötigt, die somit erforderlich für die Basisfunktionalität ist und daher ebenfalls in das erste Inkrement fällt.

2.2 Erweiterung um dynamische Aspekte

Nachdem das erste Inkrement abgeschlossen und das Ergebnis ausreichend getestet wurde, befasst sich das zweite Inkrement mit der Erweiterung um dynamische Aspekte.

Hierzu gehört hauptsächlich die dynamische Anpassung der Verzögerungszeiten der Dienste der Komponenten. Diese kann in Abhängigkeit von der Anzahl gleichzeitig eintreffender Nachrichten (bei Single-Threaded-Komponenten) oder der Aufteilung der Komponenten auf verschiedene Prozessoren geschehen. Weiterhin wäre die Anpassung der Zeiten der Konnektoren in Abhängigkeit der zu verarbeitenden Nachrichten denkbar.

Verzweigungen im Kontrollfluss einer Komponente werden im ersten Inkrement mit einer statischen Wahrscheinlichkeit aufgelöst. Dies kann zu Problemen führen, wenn der Kontrollfluss durch die Verzweigung zurückgeführt wird und somit Rekursion entstehen kann. Hier gilt es nun eine dynamische Anpassungsfähigkeit in Abhängigkeit der Rekursionstiefe einer Nachricht zu

ermöglichen. So ließe sich beispielsweise eine Schleife modulieren, welche bei den ersten 9 Durchläufen mit einer Wahrscheinlichkeit von 1.0 zu dem Dienst vor der Verzweigung zurückspringt. Erst beim 10 Durchlauf der Verzweigung wird diese Verlassen. Zur Umsetzung wird ein Stack benötigt, welcher den Weg der Nachricht protokolliert.

Ein weiterer entscheidender Entwicklungspunkt dieses Inkrements besteht in der Möglichkeit, von Diensten einer Komponente neue Nachrichten ins Netzwerk zu senden. Durch diese Modulierung lassen sich Threads simulieren, die von Diensten zur Abarbeitung einer bestimmten Aufgabe erzeugt werden.

2.3 Auswertung des Datenpools

Nach Vollendung des zweiten Inkrements steht bereits eine funktionsfähige Simulationsumgebung mit ausreichender Dynamik zur Verfügung. Die Komponenten, Konnektoren und das Netzwerk können durch den Nutzer spezifiziert werden. Weiterhin besteht die Möglichkeit, Nachrichten an einen Systemdienst zu senden und deren Weg zu protokollieren.

Die gesammelten Daten liegen nach der Simulation in Form einer Datei vor, deren Auswertung jedoch noch unflexibel und unübersichtlich ist. Hier kommt das dritte Inkrement ins Spiel. Es soll nun eine GUI entworfen werden, die die Auswertung der Daten erleichtert. Hierfür ist sowohl eine angemessene Präsentation der Daten als auch eine Möglichkeit der Anfrage des Benutzers vorgesehen.

An dieser Stelle endet die Erstellung des Frameworks im Rahmen des Individuellen Projektes. Der folgende Abschnitt enthält einige Vorschläge über Erweiterungsmöglichkeiten der Simulationsumgebungen.

2.4 Erweiterungsmöglichkeiten ausserhalb des Projektes

Bis auf die Auswertung der gesammelten Daten bietet das Framework an dieser Stelle kaum Bedienkomfort. Eine sinnvolle Erweiterung wäre hier die Entwicklung einer GUI. Diese kann beispielsweise die Verwaltung der Komponenten und Konnektoren (Erstellen und Speichern) übernehmen. Weiterhin wäre eine graphische Anzeige und Erstellung des Komponentennetzwerkes denkbar.

3 Organisatorisches

3.1 Beteiligte Personen

Das Projekt wird von Juniorprofessor Dr. Ralf Reussner und Dipl.-Wirtsch.-Inform. Steffen Becker betreut. Es ist ein regelmäßiges Treffen mit mindestens einem der Betreuer alle ein bis zwei Wochen geplant.

3.2 Benötigte Resourcen

Für die Bearbeitung des Projektes werden folgende Resourcen benötigt, die zum Teil von der Abteilung, die das Individuelle Projekt betreut, gestellt werden.

- Zur Datensicherung und Versionsverwaltung steht ein Server mit einem CVS Zugang zur Verfügung. Auf diesem Server können beliebige Dokumente und der Quellcode des Projektes gespeichert werden.
- Es wird eine Klassenbibliothek zur Verfügung gestellt, welche zur Modellierung der statische Kontrollflußstrukturen von Komponenten dient.
- Der BSCW Server (Basic Support for Cooperative Work), eine Art Groupware Server zur Kommunikation und abteilungsinternen Veröffentlichung von Dokumenten, existiert. Ein bereits eingerichteter Account steht zur Verfügung.
- Zur Entwicklung des Frameworks wird die Programmiersprache C# verwendet. Die Entwicklung erfolgt unter Verwendung der Entwicklungsumgebung *Mircosoft Visual Studio .NET 7.1* mit dem *Microsoft .NET Framework 1.1*.
- Zur Verifikation des Quellcodes und als Debug-Hilfe werden die Tools .NUNIT und .log4Net verwendet.
- Die Dokumentation des Quellcodes wird unter Verwendung von .NDOC aus den Kommentaren generiert.
- Alle Entwicklungen und Tests werden auf einem privaten PC mit dem Betriebssystem *Microsoft XP Professional* ausgeführt.

3.3 Produkte

Im Laufe des Projektes werden die hier aufgelisteten und kurz erläuterten Produkte entstehen.

- *Proposal*, die einleitende Dokumentation des Projektes
- *Ausarbeitung*, die komplette Ausarbeitung des Projektes
- *Framework*, die Implementierung des entwickelten Frameworks
- *Vortragsfolien*, es wird einen kurzen einleitenden Vortrag vor dem Projekt und einen zusammenfassenden nach dem Projekt geben.

3.4 Zeitliche Planung

Das Individuelle Projekt ist für einen Zeitraum von vier Monaten (17 Wochen) angesetzt. Hierbei steht vor Antritt des Projektes das Proposal bereits zur Verfügung und das Projekt wurde im Rahmen eines Seminartreffens kurz vorgestellt. Es folgt nun der grobe zeitliche Plan des Projektes.

Wochen	**Produkt**	**Beschreibung**
1		genauere Erfassung der Problematik und Sichtung der Klassenbibliothek
4	erstes Inkrement	Entwicklung der Basisfunktionalität
1	erstes Inkrement	Entwicklung eines ersten Prototyps
5	zweites Inkrement	Erweiterung um dynamische Aspekte
3	drittes Inkrement	Entwurf einer GUI zum Auswerten der gesammelten Daten
3	Dokumentation	Zusammenstellung der Dokumentationsfragmente, die während der drei Inkremente entstanden sind