Lastenheft

MMT Komplexpraktikum II, SS 2009

Allgemeines

Im AMACONT-Projekt wurde ein auf XML basierendes Dokumentenformat entwickelt, welches es erlaubt, genau solche adaptiven Webanwendungen zu beschreiben. Darauf aufbauend wird im DFG-Forschungsprojekt HyperAdapt der Ansatz verfolgt, Adaption mittels Aspektorientierung zu modellieren, um so eine klare Trennung zwischen der eigentlichen Kernanwendung und der zusätzlichen Facette der Adaption zu erreichen. Für die Darstellung in einem herkömmlichen Browser gibt es jedoch Standardformate wie XHTML, welche stärker auf Präsentationsaspekte fokussiert sind. Daher ist es zum Zeitpunkt der Auslieferung einer Website erforderlich, die Webanwendung einerseits schrittweise an den aktuellen Kontext des Nutzers (sein Endgerät, seinen Ort, seine Präferenzen, etc.) und andererseits in ein präsentationsorientiertes Format wie XHTML umzuwandeln.

Ziel des Komplexpraktikums ist es daher, eine Serveranwendung zu entwickeln, welche die schrittweise Überführung des AMACONT-Formates in XHTML übernimmt.

Stufenweise Architektur

Die Umwandlung des AMACONT-Formates in XHTML erfolgt nicht in einem Schritt, sondern stufenweise, unter Hinzunahme zusätzlicher Informationen auf jeder Stufe. Beispielsweise soll die Adaption in einem separaten Schritt geschehen, unter Zuhilfenahme des aktuellen Benutzerkontexts.

Für die Realisierung einer solchen Prozessabfolge existieren Pipelinearchitekturen wie Apache Cocoon. Diese ermöglichen es, verschiedenste Transformationskomponenten über eine Konfigurationsdatei in eine beliebige Reihenfolge zu bringen und deren Ein- und Ausgabekanäle miteinander zu verknüpfen.

Zunächst ist es also erforderlich, sich einen Überblick über die einzelnen Fragmente einer AMACONT-Anwendung zu verschaffen und deren Einwirkung an verschiedenen Stellen in der Pipeline zu klären. Anschließend kann ein erster Prototyp der Gesamtpipeline mit Stubs der einzelnen Komponenten aufgestellt werden, die später dann mit den konkreten Teilergebnissen verfeinert werden können. Zur Verbesserung der Performance sind Caching-Strategien zwischen den einzelnen Stufen zu implementieren.

Medienobjekte wie Bilder, CSS oder JavaScript-Dateien müssen einerseits ohne das Durchlaufen dieser Pipeline ausgeliefert werden können, andererseits muss aber auch die Möglichkeit bestehen, diese zu adaptieren (z.B. Bilder verkleinern oder in Graustufen umwandeln oder browserspezifisches CSS ausliefern). Hierfür sind spezielle, parallel angelegte Pipelines zu entwerfen.

Ein weiteres zentrales Element von AMACONT-Anwendungen ist der Anwendungskontext. Dieser steht parallel zu allen Stufen zur Verfügung, um beispielsweise Entscheidungen bezüglich der Adaption zu treffen. Bislang umfasste dieser Kontext nur nutzerspezifische Eigenschaften wie Präferenzen, Geräteeigenschaften oder Aufenthaltsort des Nutzers. Daneben gab es jedoch noch

andere Informationsinseln, die zwar nicht dem Kontext zugeordnet waren, aber dennoch eine analoge Rolle spielten. Dazu gehören beispielsweise Datenbankanbindungen, verfügbare Webservices oder Informationen über den Systemzustand. Diese sollen nun in ein (partitioniertes) Gesamtkontextmodell integriert werden, von dem Teile nutzerspezifisch, anwendungsspezifisch oder requestspezifisch sein können.

Zu realisierende Stufen

In AMACONT gibt es eine Reihe fester Stufen, die in der Pipeline abgearbeitet werden müssen, um die umfangreiche Funktionalität bereitzustellen. Die Reihenfolge der einzelnen Stufen ergibt sich beinahe zwingend:

1. Aktualisierung des Kontextes
2. Einlesen des angeforderten XML-Dokumentes
3. Anreichern des Dokuments mit Kontextinformationen (z.B. Datenbankinhalte oder Daten von Webservices)
4. Stufenweise Anpassung/Spezialisierung des Dokuments unter Berücksichtigung des aktuellen Kontexts
5. Transformation des Dokuments in das XHTML-Zielformat
6. Übermittlung an den Client

Aktualisierung des Kontextes

Ausgehend von den vom Client übermittelten Informationen muss der Anwendungskontext aktualisiert werden. Da diese Informationen häufig eine geringe Qualität und dafür höhere Quantität aufweisen (z.B. Mausposition, Fenstergröße) und so für Anwendungsentwickler weniger interessant sind, ist es erforderlich, eine Kontextmodellierungskomponente zu entwickeln, welche in der Lage ist, Informationen zu verdichten und dann in das standardisierte Kontextmodell zu schreiben. Im Falle des Komplexpraktikums ist zunächst nur eine einfache Lösung vorzusehen, die sich aber einfach erweitern lässt.

Einlesen des angeforderten XML-Dokumentes

Das Amacont-Dokument muss zunächst eingelesen werden. Hierfür kommen grundsätzlich die beiden Technologien DOM und SAX in Frage. Cocoon bietet jedoch einfache Möglichkeiten der Deserialisierung, so dass dies nicht von Hand programmiert werden muss.

Anreichern des Dokuments mit Kontextinformationen

Da die Dokumente zur Laufzeit mit Inhalten, beispielsweise aus Datenbanken oder von Webservices, angereichert werden können, muss eine Komponente implementiert werden, welche diese Informationen aus dem Kontext lädt und in das AMACONT-Dokument an der entsprechenden Stelle einfügt. Insbesondere ist hier die Herausforderung des iterierten Einfügens, beispielsweise zum Erstellen einer Linkliste, zu lösen.

Stufenweise Anpassung des Dokuments

Zur Anpassung an den aktuellen Kontext kann das Dokument in mehreren Schritten spezialisiert werden. Hier genügt zunächst die Transformation in zwei aufeinanderfolgenden Stufen, der aspektorientierten und der variantenorientierten Adaption, welche unabhängig voneinander arbeiten. Bei der aspektorientierten Adaption müssen Kontextinformationen ausgelesen werden und anschließend entschieden werden, welche der anwendungsspezifischen Adaptionsaspekte unter

diesem Kontext zum Einsatz kommen. Anschließend wird das Dokument durchlaufen und an den entsprechenden Stellen der jeweilige Aspekt eingefügt. Sind mehrere Aspekte auf eine Stelle gerichtet, ist ein (erweiterbarer) Konfliktbehebungsmechanismus vorzusehen.

Die variantenorientierte Adaption hingegen durchsucht das Dokument nach variablen Konstrukten und wählt innerhalb von diesen genau eine Variante, je nach aktuellem Kontext aus. Die anderen Varianten entfallen einfach.

Transformation in XHTML

Zuletzt muss das nun an den Nutzer angepasste Dokument in XHTML umgewandelt werden, damit der Browser es verarbeiten kann. Dazu muss für alle AMACONT-Komponenten eine spezifische Abbildung erfolgen. Es ist auch möglich, diese Abbildung in ausführbaren JavaScript-Code durchzuführen, der anschließend ein Dokument auf dem Client generiert.

Wichtig an dieser Stelle ist noch das Einfügen der speziellen Kontextsensoren, damit der Server von Kontextupdates des Clients benachrichtigt wird.

Weitere umzusetzende Anforderungen

Testumgebung

Das Testen von Cocoon-Komponenten ist bisher sehr umständlich und nicht innerhalb von Eclipse möglich. Deshalb soll eine einfache Testumgebung geschaffen werden, die es ermöglicht, eine Komponente isoliert oder innerhalb der Pipeline auf Fehler zu testen und zu debuggen. Es muss hier möglich sein, ein Eingabedokument vorzugeben, welches dann an die Komponente „gefüttert“ wird.

Basistechnologien

Cocoon

Cocoon ermöglicht es, verschiedenste Transformationskomponenten über eine Konfigurationsdatei in eine beliebige Reihenfolge zu bringen und deren Ein- und Ausgabekanäle miteinander zu verknüpfen.
http://cocoon.apache.org/2.2/
http://blog.machinimatrix.org/2008/08/17/cocoon-22-installation-tutorial/
http://cocoontutorial.logabit.com/#Sect_Was_ist_Cocoon (Cocoon 2.1, dafür Deutsch)