Booster: Kooperative Softwareentwicklung im World-Wide Web

Michael Baentsch, Georg Molter, Peter Sturm
Fachbereich Informatik, Universität Kaiserslautern
Postfach 3049, 67653 Kaiserslautern
E-mail: {baentsch, molter, sturm}@informatik.uni-kl.de
URLs: http://www.uni-kl.de/AG-Nehmer/{baentsch, molter, sturm}

Kurzfassung:

Die Realisierung zunehmend komplexer Softwareprojekte erfordert das direkte und indirekte Zusammenwirken einer immer größer werdenden Zahl von Personen. Die dafür benötigte Infrastruktur ist mit der zunehmenden globalen Rechner-Vernetzung bereits vorhanden, doch wird ihr Potential von herkömmlichen Werkzeugen in der Regel bei weitem nicht ausgeschöpft. Das in diesem Artikel vorgestellte Rahmenmodell für Softwareentwicklung wurde explizit im Hinblick auf die globale Kooperation von Entwicklern entworfen. WebMake, eine auf diesem Modell basierende Softwareentwicklungsumgebung, adressiert das Ziel seiner Einsetzbarkeit im globalen Maßstab durch die Verwendung des World-Wide Web als Datenspeicherungs- und Kommunikationsinfrastruktur.

1. Einleitung

Softwareentwicklung ist ein verteilter Prozeß, an dem mehrere Personen direkt oder indirekt beteiligt sind. Er umfaßt die unmittelbare und direkte Kooperation zwischen Entwicklern, die gemeinsam ein Softwareprojekt durchführen. Darüberhinaus beinhaltet er auch indirekte Kooperationsformen durch das vorgegebene Umfeld (z.B. Datei- und Betriebssysteme), durch den Einsatz von Werkzeugen (wie z.B. Bindern, Compilern, aber auch durch die Verwendung der Entwicklungsumgebung selbst) und durch die Verwendung von Bibliotheken (z.B. Graphikbibliotheken, mathematischen Bibliotheken und effizienten Realisierungen häufig verwendeter Datenstrukturen), die jeweils von anderen Gruppen zu einem anderen Zeitpunkt entwickelt wurden. Die meisten CSCW-fähigen Entwicklungsumgebungen unterstützen nur die direkte Kooperation und Kommunikation zwischen mehreren Personen innerhalb eines Projektes. Indirekte Kooperation findet dagegen außerhalb der Entwicklungsumgebung statt; sie beschränkt sich meist auf die Beschaffung der benötigten Werkzeuge und Bibliotheken mit Hilfe von File-Transfer-Mechanismen, der langwierigen und fehleranfälligen Installation dieser Pakete vor Ort und im Idealfall der anschließenden Verwendung.

In den nachfolgenden Abschnitten wird das Booster-System [3] vorgestellt, das als Prototyp eines sogenannten „*Global Software Highway*" (GSH) aufgefaßt werden kann, einer global verteilten offenen Infrastruktur, in der alle in Softwareentwicklungsprozessen benötigten Dokumente gespeichert und alle involvierten Werkzeuge und beteiligten Personen koordiniert werden. Die damit verbundene Gleichstellung von direkten und indirekten Kooperationsformen und die Globalisierung der Softwareentwicklung verspricht einen Synergieeffekt, der Einfluß auf wichtige, aber bisher weniger beachtete Phasen des Entwicklungsprozesses hat, wie z.B. Auffinden und Verwenden bereits realisierter Teilfunktionalitäten, Verteilung, Aktualisierung und Wartung von Softwareprodukten, sowie die Abrechnung in Anspruch genommener Dienste.

Erschienen in: Konferenzband Entwicklung und Management verteilter Anwendungssysteme EMVA'95, GI-Fachgruppe Betriebssysteme, Dortmund, Oktober 1995

2. Der „Global Software Highway”

Der GSH soll in Anlehnung an den von Bill Clinton und Al Gore geprägten „*Information Superhighway*” Softwareprodukte zugänglich machen und das weltweit verfügbare Entwicklungspotential bündeln. Er bildet eine global verteilte Infrastruktur für Softwareentwicklungsumgebungen sowie für Softwareprojekte. Die Aufgabenbereiche des GSH umfassen:

- *Entwickeln* von Software einschließlich dem verteilten Übersetzen und Binden von Bibliotheken und ausführbaren Programmen. Der GSH muß zu diesem Zweck ein globales Repository für den geschützten Zugriff auf alle Dokumente zur Verfügung stellen, auf das Softwareentwicklungsumgebungen zurückgreifen können.
- *Anbieten* von Softwareprodukten und vorhandenen Entwicklungskapazitäten. Im GSH können verfügbare Softwarelösungen oder freie Kapazitäten einzelner Entwicklergruppen weltweit angeboten werden. Es sind sehr unterschiedliche Formen des Anbietens vorstellbar einschließlich professioneller Produktwerbung und virtueller Märkte, auf denen unabhängige Softwarehändler Produkte anderer Hersteller vertreiben.
- *Auffinden* von Softwareprodukten und gewünschten Entwicklungskapazitäten. Durch geeignete Werkzeuge soll die Suche nach gewünschten Produkten und Dienstleistungen unterstützt werden. Auf diesem Gebiet agieren neben den „Endverbrauchern” auch unabhängige Berater, die das Angebot der entsprechenden Märkte sichten und konkurrierende Angebote vergleichen.
- *Verwenden*. Dieser Aufgabenbereich umfaßt Bezug, Installation, Integration, Nutzung und Wartung von Software. Der GSH muß zu diesem Zweck Mechanismen enthalten, mit deren Hilfe Produkte für eine bestimmte Hard- und Softwareumgebung zur Verfügung gestellt werden. Dienstleistungen wie zum Beispiel Online-Diagnose und Fernwartung können ebenfalls in den GSH integriert werden.

Die Vision eines solchen „Global Software Highway” kann nur dann Realität werden, wenn die Rechte jedes einzelnen Entwicklers gewahrt bleiben und wenn in Anspruch genommene Dienstleistungen korrekt abgerechnet werden können.

3. Ein Rahmenmodell für globale Softwareentwicklung

Einen ersten Schritt auf dem Weg zum Global Software Highway stellt das Booster-Modell für verteilte Softwareentwicklung dar, das eine Reihe von Abstraktionen und Mechanismen anbietet, auf deren Basis Komponenten einer verteilten Entwicklungsumgebung realisiert werden können. Gleichzeitig stellt es hinreichende Offenheit und Erweiterbarkeit sicher, um eine Integration unterschiedlichster Unterstützungswerkzeuge und sogar Entwurfsmethoden zu ermöglichen.

Eine Softwareentwicklungsumgebung im Booster-Modell enthält Mechanismen zur verteilten Datenhaltung sowie eine Menge von Entwicklungswerkzeugen, die maßgeschneiderte Lösungen für abgegrenzte Teilaufgaben realisieren. Zur Beschreibung aller Subjekte, Objekte und Aktivitäten, die im Software-Lebenszyklus auftreten, stellt Booster ein spezielles Strukturmodell zur Verfügung. Die darin enthaltenen, nach dem objektorientierten Paradigma in einer Klassenhierarchie angeordneten Abstraktionen beschreiben die Daten, auf denen die einzelnen Unterstützungswerkzeuge arbeiten und die durch die Datenhaltungs-Komponente dauerhaft gespeichert werden können. Die wesentlichen Elemente des Strukturmodells werden im folgenden vorgestellt (vgl. Bild 1).

Die Komponenten eines Softwareprojektes werden durch Graphen mit typisierten Knoten und Kanten beschrieben. Sogenannte *Description-Knoten* repräsentieren atomare Bestandteile

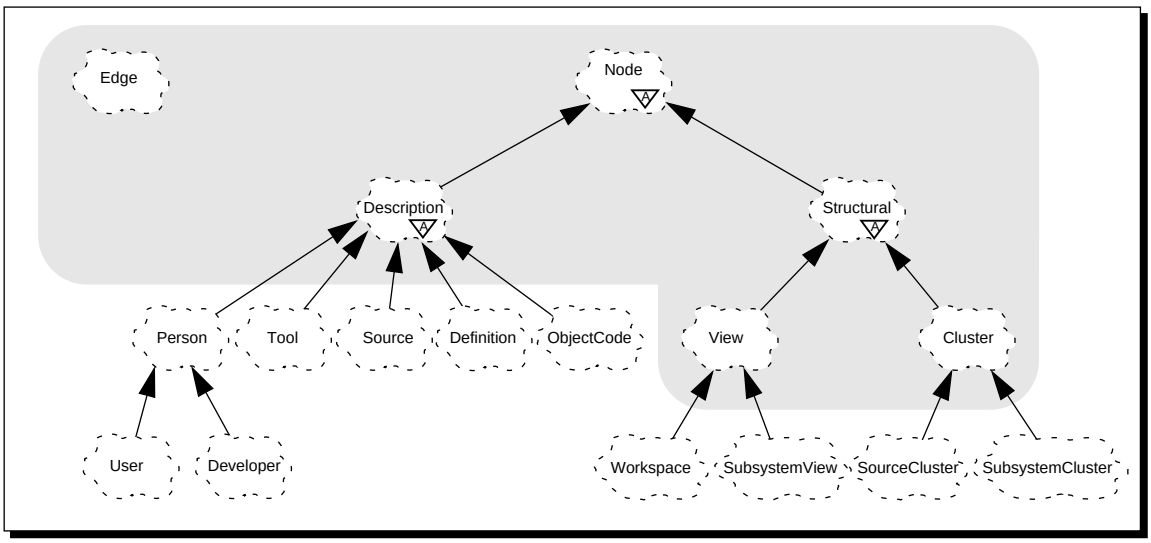

Bild 1: Übersicht über die zentralen Klassen des Booster-Strukturmodells

des modellierten Systems, beispielsweise Design-Dokumente wie Deklarations- oder Quelldateien, aber auch in den Software-Lebenszyklus involvierte Personen (z.B. Entwickler und Anwender). Um die anwendungsspezifische Semantik und Funktionalität zu fassen, sieht das Strukturmodell spezielle Arten von Description-Knoten vor, die beliebig verfeinert werden können. Die zweite Gruppe von Knoten bilden die *Strukturknoten*, die es erlauben, eine Menge von Knoten zu einer neuen Abstraktion zusammenzufassen. Da Strukturknoten selbst Knoten sind, können sie wiederum von Strukturknoten gruppiert werden und so fort. Dies erlaubt den rekursiven Aufbau beliebiger Hypergraphen, die im Strukturmodell eingesetzt werden, um komplexe Systeme zu repräsentieren.

Das Booster-Modell definiert zwei Arten von Strukturknoten: *Cluster-Knoten* und *Views*. Clusterknoten *enthalten* andere Knoten; sie bilden damit eine Abstraktion für das in der nächstniedrigeren Ebene im Detail modellierte System, dessen Struktur zunächst verdeckt ist. Durch explizites Expandieren eines Clusterknotens kann jedoch die von den enthaltenen Knoten aufgespannte Struktur sicht- und manipulierbar gemacht werden. Die konkrete Bedeutung eines Clusters im Kontext einer bestimmten Anwendung wird durch die jeweilige Clusterknoten-Subklasse beschrieben, beispielsweise durch die auch in Bild 2 dargestellte Klasse *Subsystem-Cluster*, die dazu dient, eigenständige Softwarekomponenten zu kapseln. Einerseits wird dadurch von den Implementierungsdetails abstrahiert, andererseits transformiert der Strukturknoten das gekapselte Subsystem (hier in eine Bibliothek) und stellt es nach außen zur Verfügung.

Views sind die zweite Art von Strukturknoten. Sie beschreiben Projektionen auf der Menge der Knoten und Kanten. Anders als Clusterknoten kapseln Views die von ihnen referenzierten Knoten nicht ein; während ein Knoten nur zu maximal einem Cluster gehören kann, kann er in beliebig vielen Views liegen. Der mit Hilfe von Clusterknoten hierarchisch strukturierte Graph beschreibt den Aufbau und die Eigenschaften der Komponenten eines Softwareprojektes. Auf diesem Gesamtgraphen erlauben Views die maßgeschneiderte Projektion von in einem bestimmten Zusammenhang relevanten Details; so kann beispielsweise eine Sicht auf ein bestimmtes Subsystem herausprojiziert werden. Views ermöglichen es damit den Entwicklern, den Überblick über das erstellte komplexe Softwaresystem mit seiner großen Menge von Designdokumenten zu behalten. So enthält das Strukturmodell beispielsweise eine Subsystem-View-Klasse, die eine Gruppe von Entwurfsartefakten beschreibt, aber anders als Subsystem-Cluster-Instanzen nicht einkapselt; dies wird ebenfalls durch Bild 2 illustriert. Andere Knoten

können weiterhin direkte Beziehungen mit den Knoten des durch die View beschriebenen Subsystems eingehen, während in einem SubsystemCluster enthaltene Knoten nach außen nicht sichtbar sind.

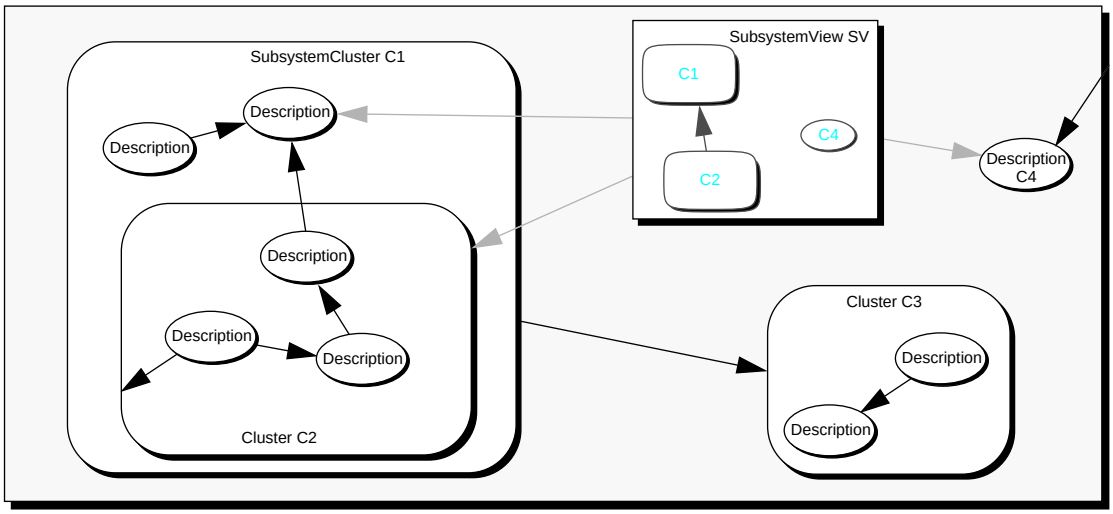

Bild 2: Beispielszenario mit einigen der im Text beschriebenen Strukturkomponenten von Booster

Knoten im Strukturmodell sind über gerichtete und typisierte Kanten verbunden, Instanzen der Klasse *Edge*. Sie beschreiben zum einen den statischen Aufbau des modellierten Systems, d.h. die (möglicherweise anwendungsspezifischen) Beziehungen zwischen seinen Elementen wie beispielsweise funktionale Abhängigkeiten zwischen Systemkomponenten, und zum anderen dienen sie als Kommunikationskanäle für die verbundenen Objekte. Da nur der Ursprungsknoten einer Kante das Senden einer Nachricht über diese Kante initiieren kann, resultiert Auftragskommunikation mit einer Request- und einer beliebigen Zahl von Reply-Nachrichten als Interaktionsschema der im Strukturmodell repräsentierten Objekte.

4. Booster und das World-Wide Web

Ein erster Prototyp einer Softwareentwicklungsumgebung im Booster-Modell ist das WebMake-System, das das World-Wide Web [4] als Datenspeicherungs- und Kommunikationsinfrastruktur nutzt.

Prinzipiell ist das nach dem Client/Server-Prinzip aufgebaute World-Wide Web als eine unstrukturierte Sammlung von Hypertext-Dokumenten zu verstehen, die Querverweise zu anderen Objekten enthalten und so den Aufbau eines Geflechts von Dokumenten erlauben. Die einzelnen Dokumente sind in der sogenannten *Hypertext Markup Language* (HTML) abgefaßt. Der Datentransfer von den Servern zu den Clients erfolgt unter Verwendung von Protokollen aus der TCP-Protokollfamilie, wie z.B. dem *File-Transfer-Protokoll* (FTP) oder dem *Hypertext-Transfer-Protokoll* (HTTP).

Der Verbreitungsgrad, den das WWW gefunden hat und der es als eine geeignete Infrastruktur für global verteilte Anwendungen erscheinen läßt, gründet sich im wesentlichen auf die freie Verfügbarkeit aller Komponenten. Es ist inzwischen genauso einfach, einen WWW-Server zu installieren, um selbst Daten in das World-Wide Web einzuspeisen, wie es die Benutzung der graphischen Benutzeroberflächen Mosaic oder Netscape auf der Client-Seite bereits seit einigen Jahren ist.

Das im vorigen Kapitel eingeführte Strukturmodell zur Repräsentation und Strukturierung der Dokumente eines Softwareprojekts wurden folgendermaßen durch Elemente des WWW realisiert: Allgemeine Knoten wurden auf beliebige, im World-Wide Web adressierbare Dokumente abgebildet; Kanten sind Hyperlinks in der HTML-Beschreibungssprache; typisierte Kanten wurden ebenso wie Strukturknoten durch zusätzliche, automatisch verarbeitbare textuelle Annotationen in HTML-Dokumenten implementiert. Interaktion zwischen Clients (und damit verschiedenen Benutzern) sowie zwischen Servern wurde realisiert durch die Verwendung der - noch nicht endgültig standardisierten - WWW-Protokolle *Client Communications Interface* (CCI) und *Common Gateway Interface* (CGI). Für eine eingehende Beschreibung der Techniken, die die Anbindung von Booster an die Mechanismen des World-Wide Web ermöglichten, sei an dieser Stelle auf [2][2] verwiesen.

Auf Basis dieser Strukturmodell-Implementierung im World-Wide Web wurde eine Reihe von Werkzeugen entwickelt, die zusammen WebMake bilden, eine erste prototypische Softwareentwicklungsumgebung nach dem Booster-Rahmenmodell. Sie unterstützt die verteilte Arbeit mehrerer Entwickler an einem Softwareprojekt ebenso wie verteiltes Übersetzen und Binden der einzelnen Softwarekomponenten, die im WWW abgelegt sind. Als Zugangsapplikation dient ein spezieller WWW-Client mit direktmanipulativer graphischer Oberfläche (vgl. Bild 3). Anwendungsentwickler werden außerdem von WebMake bei der Portierung von mit Unterstützung des Booster-Systems entwickelter Software auf verschiedene Hardwareplattformen unterstützt: Software-Module können automatisch auf vertrauenswürdige Rechnerknoten versendet werden, auf denen das Übersetzen und Binden für die jeweils gewünschte Zielarchitektur möglich ist. Als Rückgabedaten werden von einem solchen Server entweder die entsprechenden Objektdateien oder aber Fehlermeldungen geliefert.

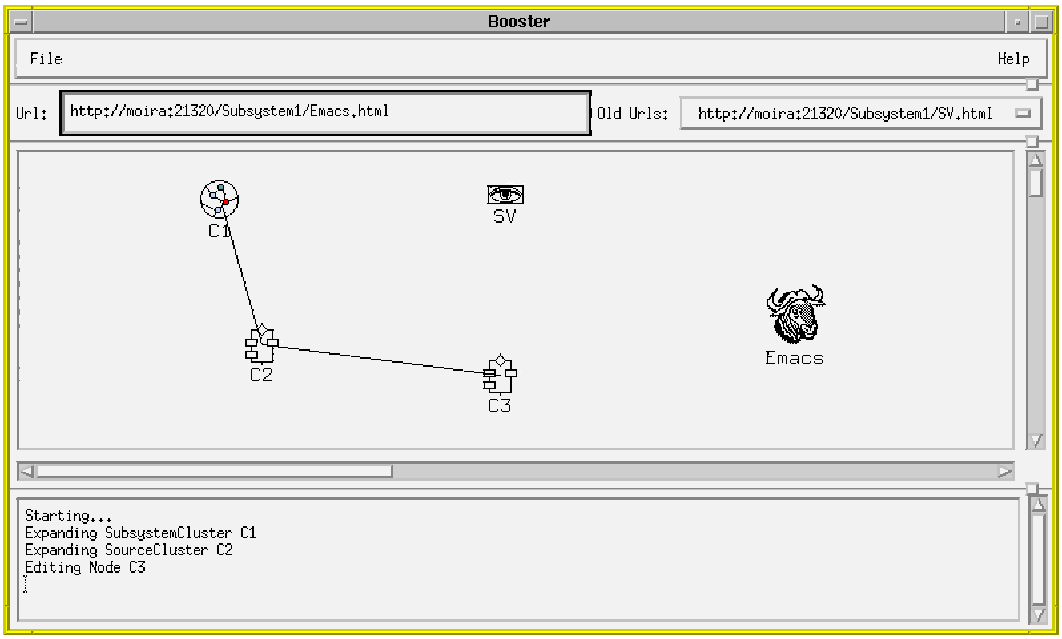

Bild 3: Graphische Benutzeroberfläche von WebMake, die direktmanipulativen Zugang zu den unterliegenden Mechanismen bietet

5. Erfahrungen und Ausblick

Versuche mit der Prototyp-Implementierung haben ergeben, daß der durch die Parallelverarbeitung von Booster erreichte Leistungsvorteil gegenüber üblichen Ein-Rechner-Entwicklungsumgebungen ab mehr als 12 eingesetzten Rechnern nicht mehr existiert. Eine Analyse der Probleme bestätigte die Vermutung, daß die verwendete WWW-Technologie (HTTP-Protokoll Version 1.0, CERN- und NCSA-httpd) nicht dazu in der Lage ist, die erforderliche hohe Auftragsrate zu verarbeiten, die etwa dann auftritt, wenn viele Knoten von anderen Knoten quasi gleichzeitig Updates anfordern. Bei sehr stark frequentierten Diensten im World-Wide Web tritt dieser Effekt ebenfalls zutage. Die in der Entwicklung befindliche nächste Generation von WWW-Servern und -Protokollen berücksichtigt diese Problematik unter anderem durch Serverrealisierungen auf der Basis leichtgewichtiger Prozesse und durch die Integration eines Session-Protokolls in die nächste Version von HTTP.

Die Einführung bidirektionaler Kanten für die effiziente Weitergabe bzw. Konsistenzerhaltung von Daten innerhalb des Booster-Systems als erforderlich. Dies geschah bisher nur an einigen ausgewählten Stellen, um die vorher aufgetretenen lokalen Leistungsengpässe zu vermeiden. Diese entstanden beispielsweise deshalb, weil durch separate Komponenten des WebMake-Systems zur Ermittlung von Abhängigkeiten dieselben Dokumente wiederholt traversiert wurden. Durch ein einfaches Rückverfolgen der bidirektionalen Hyperlink-Kette kann dies nun vermieden werden. Die fehlende Verfügbarkeit bidirektionaler Links stellt ein prinzipielles Problem im World-Wide Web dar. Im an der Universität Graz entwickelten Hypermediasystem Hyper-G [1] wurde diese Problematik ebenfalls erkannt und behoben.

Ein weiteres Problem, das ebenso wie die im weiteren noch angerissenen Themen im wesentlichen direkt durch die Verwendung des World-Wide Web als Infrastruktur begründet ist, läßt sich auf das Schlagwort ‘Skalierbarkeit’ verkürzen: Da immer derselbe Rechner für die Bereitstellung derselben Dokumente verantwortlich ist, kann es passieren, daß er durch gleichzeitige parallele Zugriffe von verschiedenen Clients bis zum Quasi-Zusammenbruch der Server-Maschine belastet werden kann. Ansätze zur Lösung dieses Problems, die in Booster - weil hier durch das hohe Maß an Parallelverarbeitung von besonderer Bedeutung - angedacht sind, basieren auf Replikations- und Caching-Techniken.

Die Mechanismen, die in unserem System für mehr Interaktion zwischen den verschiedenen Komponenten des WWW sorgen, sollten noch wesentlich stärker in die Basiskomponenten des Web Eingang finden, um zum Vorteil sowohl der Nutzer als auch der Dienstanbieter neue, interessante Anwendungen zu ermöglichen. Ein Beispiel für derartige Mechanismen ist die Fähigkeit von Servern, Daten unaufgefordert an (zuvor registrierte) Clients zu schicken, um so beispielsweise Benutzer kontinuierlich auf dem laufenden zu halten.

Ferner müßte auch noch eine Lösung für die fehlende Adressierungs- und Serverausfall-Transparenz für das Web entwickelt werden; dies ist für Booster umso dringlicher, als ein Ausfall eines Rechners die Weiterarbeit an einem System unmöglich machen kann, sofern dieses eine dadurch unzugänglich gewordene Komponente enthält.

Wie bereits kurz angedeutet, verfügt das WWW über keinerlei Strukturierungsmechanismen; es besteht aus chaotisch miteinander vernetzten Dokumenten. Dies hat unter anderem zur Folge, daß Web-Benutzer sich des öfteren „lost in Hyperspace” wiederfinden. Mit unserem hierarchischen Strukturierungsansatz könnte hier zumindest teilweise Abhilfe geschaffen werden. Die fehlende Sicherheit im Web ist ebenfalls ein großes Manko: Wenn es nicht möglich ist, sicher festzustellen, ob sowohl der Dienstnehmer als auch der Server jeweils die sind, für die sie sich ausgeben, bzw. solange unautorisierte Zuhörer jede Kommunikation auf dem Web mitprotokollieren können, kann das WWW nicht auf vertrauenswürdige Weise genutzt werden. Da Accounting und Sicherheit in Booster ebenfalls eine herausragende Problemgruppe darstel-

len, haben wir uns unabhängig von allgemeinen Bestrebungen auf diesem Gebiet (vgl. z.B. [6], [7], [8]) eigene Lösungsansätze erarbeitet, die im wesentlichen auf Public-Key-Verschlüsselungsverfahren basieren.

Der nächste Schritt auf dem Weg zu einer für jeden möglichst einfach zugänglichen Softwareentwicklungsumgebung besteht darin, das graphische Benutzerinterface aufbauend auf den Erfahrungen, die wir mit der gegenwärtigen Version gemacht haben, mit Hilfe der HotJava-Technologie[5] [5] zu realisieren. Sobald dieser Schritt vollzogen ist, wird die spezielle Booster-Funktionalität für jeden Anwender des HotJava- und Netscape-WWW-Browsers (und damit für effektiv ca. 90% aller WWW-Benutzer) zugänglich sein, da Java-Programme auf jedem Rechner ausgeführt werden können, auf dem ein entsprechender, interpretierender WWW-Browser vorhanden ist. Wir erwägen außerdem, ein einfaches, HTML-basiertes Interface zum Booster-System zu realisieren, um somit jedem Web-Teilnehmer die Gelegenheit zu geben, die Vorteile des verteilten Softwareentwicklungssystems Booster zu nutzen (wenn auch nur in eingeschränkter Form, da in diesem Fall keine speziellen Client-Fähigkeiten wie z.B. kontextsensitive Popup-Menüs zur Verfügung stehen).

6. Zusammenfassung

Das in diesem Artikel vorgestellte Booster-System erlaubt die global verteilte Entwicklung, Verwaltung und Distribution von Software auf der gegenwärtig am weitesten verbreiteten Speicherungs- und Kommunikationsinfrastruktur, dem World-Wide Web. Es wurde in diesem Rahmen der verteilte Prototyp WebMake entwickelt, der bereits WWW-Server als Datenspeicher und Compileserver einsetzt. Beim Design und der Realisierung der verschiedenen Aufgaben, die Booster erfüllen soll, sowie der eingehenderen Beschäftigung mit den aktuell eingesetzten Mechanismen zur Datenspeicherung und Kommunikation wurden viele interessante Problemgebiete sichtbar, die für die weitere Entwicklung großer Informationssysteme von Bedeutung sind und die gelöst werden müssen, bevor die Vision eines Global Software Highway Realität werden kann.

7. Literaturverweise

[1] K. Andrews, F. Kappe: Soaring through Hyperspace, *A Snapshot of Hyper-G and its Harmony Client*; Proc. of Eurographics Symposium on Multimedia/Hypermedia in Open Distributed Environments, 1995; pp. 181-191.

[2] M. Baentsch, G. Molter, P. Sturm: *WebMake: Integrating distributed software development in a structure-enhanced Web*; Computer Networks and ISDN Systems, April 1995, Vol. 27, No. 6, pp. 789-800.

[3] M. Baentsch, G. Molter, P. Sturm: *Booster: A WWW-based Prototype of the Global Software Highway*; Proc. of the 2nd International Workshop on Services in Distributed and Networked Environments (SDNE 1995); IEEE Computer Society Press.

[4] T. Berners-Lee, et.al: *The World-Wide Web*; Communications of the ACM, August 1994, Vol. 37, No. 8, pp. 76-82.

[5] J. Gosling, H. McGilton: *The Java Language Environment: A White Paper*; URL=http://java.sun.com/whitePaper/javawhitepaper_1.html, 1995.

[6] P.M. Hallam-Baker: *Shen: A Security Scheme for the World Wide Web;* URL = http://info.cern.ch/hypertext/WWW/Shen/ref/shen.html.

[7] K.E.B. Hickman: *The SSL Protocol;* URL=http://home.mcom.com/info/SSL.html.

[8] E. Rescorla, A. Schiffman: *The Secure HyperText Transfer Protocol;* Internet Draft; URL = http://www.commerce.net/information/standards/drafts/shttp.txt.