Werkzeuge zur kollaborativen Softwareerstellung – Stand der Technik

Michael Geisser (Universität Mannheim)

Tobias Hildenbrand (Universität Mannheim)

Lars Klimpke (Universität Mannheim)

Asarnusch Rashid (Forschungszentrum Informatik, Karlsruhe)

Zusammenfassung

Dieses Arbeitspapier untersucht, wie existierende Werkzeuge die kollaborative Erstellung von Software unterstützen. Zu diesem Zweck wurde eine Marktanalyse durchgeführt, deren Ergebnisse hier präsentiert und diskutiert werden. Hierbei werden zunächst Werkzeuge für die verschiedenen Phasen des Softwareentwicklungsprozesses betrachtet und analysiert, wobei der Fokus auf Werkzeugen für die Anforderungsanalyse liegt. Dies liegt darin begründet, dass sich gerade hier eine intensive Kollaboration aller Beteiligten kritisch für den Erfolg des Projekts zeigt. Anschließend werden kollaborative Anforderungen an Plattformen aufgezeigt und schließlich existierende Kollaborationsplattformen vorgestellt und analysiert.

Als Fazit lässt sich konstatieren, dass es eine Vielzahl an spezialisierten Werkzeugen gibt, die teilweise auch eine kollaborative Erstellung von Software unterstützen. Zudem decken große Firmen (IBM, Microsoft, Borland) den kompletten Softwareentwicklungsprozess mit ihren Paketen ab und binden verstärkt kollaborative Funktionen in ihre Komplettpakete mit ein. Dadurch decken sie einen Großteil des Funktionsumfangs der Kollaborationsplattformen ab, verfügen im Gegensatz zu letzteren aber nicht über einige spezielle Funktionen zur Unterstützung einer erfolgreichen Kollaboration in der Softwareentwicklung.

Inhaltsverzeichnis

1 Einleitung

Durch die fortschreitende Globalisierung der Wirtschaft und eine damit verbundene Zunahme an globalen, strategischen Partnerschaften, internationalen Firmenzusammenschlüssen in Joint Ventures und global aufgestellten Weltkonzernen wird auch ein verteiltes Entwickeln von Software unumgänglich (vgl. bspw. Karolak (1998)). So sprechen laut French und Layzel (1998) vor allem folgende organisatorische Faktoren für eine verteilte Softwareentwicklung:

- Für Reiseverpflichtungen unmotivierte Teammitglieder,
- Mangelndes Expertenwissen in einigen geografischen Regionen,
- Hohe Reise- und Umzugskosten,
- Physischer Standort von eingesetzter Hardware.

Möglich wurde ein verteiltes Entwickeln in den letzten Jahren durch technologischen Fortschritt, der eine ausreichende Kommunikationsinfrastruktur, Bandbreite sowie Leistung mit sich brachte (vgl. Lloyd u.a. 2002).

Ein verteiltes Arbeiten führt in der Regel zu Arbeitsteilung und damit zu kollaborativen[1] Ansätzen. Zudem ist die Softwareentwicklung schon an sich ein Prozess, der ohne Zusammenarbeit nur schwer oder gar nicht durchlaufen werden kann (vgl. auch Geisser und Hildenbrand (2005)). So stellen Cook und Churcher fest:

> *„Software Engineering is inherently a team-based activity"*
> (Cook und Churcher, 2003, S.290).

Dieses Arbeitspapier soll daher untersuchen, wie existierende Werkzeuge die kollaborative Erstellung von Software unterstützen. Hierzu werden in Kapitel 2 zunächst Werkzeuge für die verschiedenen Phasen des Softwareentwicklungsprozesses betrachtet. Der Fokus liegt hierbei auf Werkzeugen für die Anforderungsanalyse, da sich gerade hier eine intensive Kollaboration kritisch für den Erfolg des Projekts zeigt (vgl. Beyer und Holtzblatt (1995)). Anschließend werden in Kapitel 3 existierende Kollaborationsplattformen für den gesamten Softwareerstellungsprozess vorgestellt und kollaborative Anforderungen an ebendiese aufgezeigt. Kapitel 4 fasst schließlich die Ergebnisse dieser Arbeit zusammen.

[1] Auf eine Definition des Begriffs Kollaboration im Rahmen der Softwareentwicklung sei an dieser Stelle verzichtet und auf Rashid u.a. (2006) verwiesen.

2 CASE-Tools

Werkzeuge zur Unterstützung bestimmter Prozesse des Software-Engineerings (SE) werden allgemein unter dem Namen CASE-Tools subsumiert. CASE steht hierbei für *Computer-Aided Software Engineering*. Mit diesen Werkzeugen können zum einen Aktivitäten des SE automatisiert werden, zum anderen bieten CASE-Tools auch methodische Informationen zur Unterstützung der einzelnen SE-Phasen. Da mittlerweile zahlreiche Werkzeuge existieren, die vor allem Routine-Aktivitäten des SE unterstützen, allerdings nur wenige explizit einer kollaborativen Softwareerstellung dienen (vgl. Sommerville (2004)), wurde im Forschungsprojekt CollaBaWü[2] eine ausführliche Marktstudie zu CASE-Tools sowie speziell auch zu Kollaborationsplattformen durchgeführt, deren Ergebnisse in diesem und im nächsten Kapitel präsentiert werden.

Ausgehend von einer 105 Werkzeugen umfassenden Liste wurden für diese Marktstudie 47 Werkzeuge aufgrund ihrer Beschreibung und Verbreitung ausgewählt und genauer analysiert (eine Liste dieser Werkzeuge findet sich in

Tabelle 7 im Anhang A).

Auf den folgenden Seiten werden die Ergebnisse der Analyse, eingeteilt in die Kategorien

- Anforderungsanalyse,
- Architekturentwurf,
- Implementierung und
- Testen vorgestellt.

Da sich gerade die Anforderungsanalyse durch eine hohe Kollaboration aller beteiligten Akteure auszeichnet, soll dieser ersten Phase des SE-Prozesses besondere Aufmerksamkeit geschenkt werden. Allgemeine Kollaborationsplattformen sowie kollaborative Anforderungen an dergleichen werden ausführlich in Kapitel 3 vorgestellt.

[2] CollaBaWü beschäftigt sich mit der kollaborativen, komponentenbasierten Entwicklung von Unternehmenssoftware im Finanzdienstleistungsbereich von Baden-Württemberg, siehe http://www.collabawue.de/ (02.01.2006).

2.1 Werkzeuge für die Anforderungsanalyse

Zunächst soll die werkzeugseitige Unterstützung für die Anforderungsanalyse, im Folgenden Requirements Engineering (RE) genannt, analysiert werden. Für das RE wird hierbei unterschieden nach den Phasen

- Anforderungserhebung und -analyse,
- Anforderungsspezifikation,
- Anforderungsvalidierung und
- Anforderungsmanagement (vgl. Sommerville (2004)).

Zunächst wird die Marktverbreitung kommerzieller RE-Werkzeuge untersucht, anschließend werden die vier führenden Werkzeuge vorgestellt und miteinander verglichen werden:

Nach einer Marktstudie der Standish Group sind die führenden Werkzeuge im RE-Bereich DOORS der Firma Telelogic sowie RequisitePro aus der Rational Suite von IBM (die Ergebnisse der Marktstudie sind in Abbildung 1 dargestellt, vgl. StandishGroup (2002)). Dass sich daran auch bis heute nichts geändert hat, bestätigen in einer aktuellen Studie die beiden Herausforderer RTM von Serena und das Borland-Produkt CaliberRM: gefragt nach ihren größten Wettbewerbern geben beide die zwei führenden Hersteller im Markt an (vgl. Hood u.a. (2005)). Der eigentliche Konkurrent aller Anbieter ist aber Microsoft mit seiner Office Suite, da ein Einsatz von professionellen RE-Werkzeugen in der Praxis noch nicht stark verbreitet ist (vgl. StandishGroup (2002)).

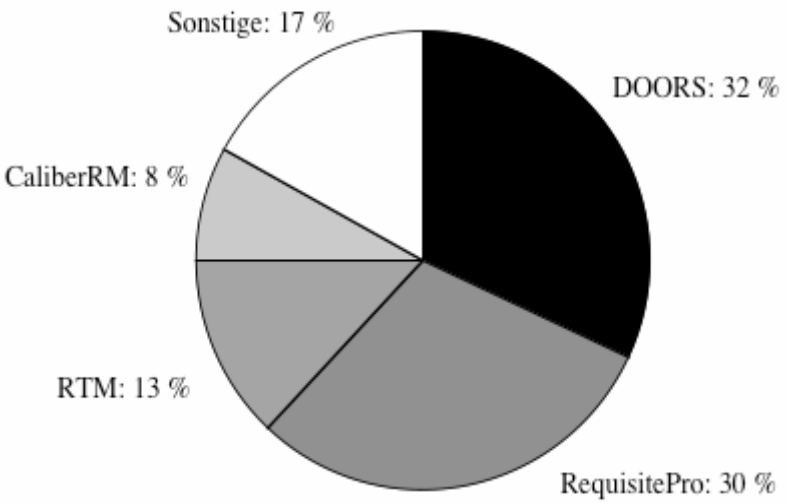

Abbildung 1: Marktanteile der RE-Werkzeuge

Bei einer näheren Betrachtung der vier Werkzeuge mit der größten Marktverbreitung fällt auf, dass alle fast ausschließlich das Anforderungsmanagement fokussieren. Während die Anforderungsspezifikation nur von DOORS (wenn auch sehr rudimentär) und die Validierung von Anforderungen von keinem Werkzeug unterstützt wird, bieten zumindest DOORS, CaliberRM und RequisitePro einige Funktionalitäten für die Anforde-

rungserhebung und -analyse. Der klare Schwerpunkt aller vier Werkzeuge liegt aber, wie schon erwähnt, im Anforderungsmanagement: sowohl das Änderungsmanagement als auch das Management der Nachvollziehbarkeit (Traceability) wird von allen vier Werkzeugen sehr gut unterstützt. Zudem verfügen alle betrachteten Werkzeuge über ein Rollenkonzept, um die Anwendung mehreren Benutzern mit verschiedenen Rechten zu erlauben. Auch Schnittstellen zu anderen, im Prozess nach gelagerten Werkzeugen, bieten alle Werkzeuge (vgl. Hood u.a. (2005)). Im Folgenden wird auf die Besonderheiten der einzelnen Werkzeuge eingegangen, dabei fließen sowohl Herstellerangaben der jeweiligen Produktseiten im Internet, als auch Ergebnisse der Werkzeugevaluationen von Hood u.a. (2005), Schienmann (2002) sowie Versteegen (2004) mit ein. Die Angaben beziehen sich auf die Versionen DOORS 7.1[3], RequisitePro 2003.06.14[4], RTM 5.3.6[5] und CaliberRM 2005[6].

DOORS steht für „Dynamic Object-Oriented Requirements System“ und ging aus einem Projekt für die European Space Agency (ESA) hervor (vgl. Schienmann (2002)). Neben einer Unterstützung des Managements für Änderungen und der Nachvollziehbarkeit bietet es die Möglichkeit, Anforderungen visuell (mit Hilfe der UML) zu beschreiben. Zudem enthält DOORS eine eigene Skriptsprache und lässt sich so flexibel anpassen und erweitern. Im Unterschied zu den anderen RE-Werkzeugen verfügt DOORS über ein eigenes Repository und nutzt keine kommerziell erhältliche Datenbank anderer Hersteller. Schwachstellen können beim Import und Export erkannt werden, da hier nur Word und Excel als Formate unterstützt werden. Die anderen Werkzeuge unterstützen zumindest XML oder CSV als Austauschformate (vgl. Hood u.a. (2005) und Versteegen (2004)).

RequisitePro gehört seit 2003 zu IBM und ist fester Bestandteil der Rational Software Suite, die den RUP (Rational Unified Process) werkzeugseitig unterstützt. Dieses Werkzeug ist sehr eng mit Microsoft Word integriert und verfügt über eine umfangreiche Webschnittstelle (vgl. Hood u.a. (2005) und Schienmann (2002)). Die Anforderungsspezifikation wird, wie auch von RTM und CaliberRM, nicht unterstützt – hierfür wird auf spezialisierte Modellierungswerkzeuge verwiesen.

RTM (Requirements and Traceability Management) ist erst seit der Übernahme der Firma Integrated Chipware im Jahr 2004 im Produktportfolio von Serena und ist ebenfalls eng mit Microsoft Word integriert. RTM erlaubt die Statusverfolgung von Anforderungen und zeichnet sich durch sehr gute Möglichkeiten im Management der Nachvollziehbarkeit und beim Anlegen von Anforderungen aus, in beiden Fällen ist es sogar den anderen Werkzeugen überlegen (vgl. Hood u.a. (2005)). Die Erhebung von Anforderungen wird allerdings überhaupt nicht unterstützt.

CaliberRM gehört seit 2002 zu Borland (ursprünglich wurde es von „Technology Builders, Inc.“ entwickelt, dann aber 2001 zunächst von „Starbase Corporation übernommen, siehe StandishGroup (2002) und konnte in den letzten zwei Jahren seinen Umsatz um 80% steigern (siehe Hood u.a. (2005)). Es erlaubt eine sehr feine Granularität bei der Abbildung von Anforderungen sowie deren Beziehungen untereinander und zu externen Informationen. Auch die Anforderungserhebung wird besser als von den anderen Werkzeugen unterstützt, indem eine sehr umfangreiche Analyse der Anforderungen durch eine Verbindung von Projektumfang, Zeitplan und Kosten mit der Ressourcenzuordnung ermöglicht wird. So können die Wirkungen von Änderungen auf diese Variablen leicht identifiziert werden. Als einziges Produkt ermöglicht es, einen Web-Client zur Bedienung auch im Offline-Betrieb zu nutzen (vgl. Hood u.a. (2005)).

---

[3] http://www.telelogic.com/products/doorsers/doors/index.cfm (28.04.2005)
[4] http://www.ibm.com/software/awdtools/reqpro/ (28.04.2005)
[5] http://www.serena.com/Products/rtm/home.asp (28.04.2005)
[6] http://www.borland.com/caliber/ (28.04.2005)

Eine Übersicht zum Funktionsumfang der besprochenen RE-Werkzeuge mit einer Zuordnung zu den einzelnen Phasen des RE-Prozesses bietet Tabelle 1[7].

Werkzeug	Erhebung und Analyse	Spezifikation	Validierung	Management
DOORS (Telelogic)	+	+	-	+++
RequisitePro (IBM)	+	-	-	+++
RTM (Serena)	-	-	-	+++
CaliberRM (Borland)	++	-	-	+++

Tabelle 1: Übersicht über die vier führenden RE-Werkzeuge

Eine Studie der Firma Yphise evaluiert die vier besprochenen Werkzeuge[8] danach, ob und wie stark Projektkosten und -dauer, Wartungskosten sowie das Projektrisiko verringert werden können und inwiefern der Kundennutzen der resultierenden Software durch Einsatz der RE-Werkzeuge maximiert werden kann. Dabei kommt Yphise zu folgendem Ergebnis:

> *„DOORS stands out in all the assessment dimensions of our list of requirements. It provides the most complete and homogeneous coverage of these dimensions. This outlines its maturity and reinforces its leadership. This makes it a perfect candidate for investment in requirements management"* (Yphise, 2005, S.14).

Die Auswahl eines geeigneten Werkzeugs für das RE hängt auch von dem durchzuführenden Projekt sowie anderer im Softwareentwicklungsprozess verwendeter Werkzeuge ab und sollte daher systematisch erfolgen. Hier sei auf Schienmann (2002) verwiesen, der eine umfassende Liste von Kriterien für die Bewertung von Werkzeugen herausarbeitet und darauf aufbauend ein systematisches Vorgehen bei der Auswahl eines Werkzeugs vorschlägt. Komplementär dazu können auch die Hinweise zu diesem Thema von Hood u.a. (2005) sowie Versteegen (2004) betrachtet werden.

[7] Legende zur Tabelle: +++ = wird sehr gut unterstützt; ++ = wird gut unterstützt; + = wird rudimentär unterstützt; - = wird nicht unterstützt.

[8] Zusätzlich wurde noch das hier nicht besprochene Werkzeug Reconcile der Firma Compuware in die Yphise-Studie aufgenommen. Für Informationen zu diesem Werkzeug sei auf Yphise (2005) sowie die Produktseite im Internet verwiesen: http://www.compuware.com/products/reconcile.htm (28.04.2005).

2.2 Werkzeuge für den Architekturentwurf

Eine Analyse der Werkzeuge für den Architekturentwurf bez. deren Eignung für die kollaborative Softwareerstellung ergab folgendes Ergebnis:

Hersteller	Tool Name	UML Unterstützung	Modellgetriebene Entwicklung	Import/Export von Modellen	Reverse Code Engineering	Codeerzeugung aus Diagrammen	Eigene Vorlagen	Querverweise zwischen Modellen	Design Traceability	Veröffentlichung und Berichterstattung im Internet	Generierung von Dokumenten	Automatisches Layout der Diagramme	Testunterstützung	Modellierung von Geschäftsprozessen	Modellierung von Datenbanken	Versionskontrolle	Echtzeitkommunikation
Microsoft	Visual Studio .Net	x				x	x				x			x	x		
Open Source (IBM)	Eclipse	x	x	x		x	x	x	x	x						x	x
IBM	Rational Rose XDE Developer	x	x			x	x	x						x	x		
Visual Paradigm	Visual Paradigm	x				x	x		x		x	x					
IBM	Rational Software Architect[9]/Modeler[10]	x	x			x	x		x	x	x	x	x		x	x	
Borland	Borland Together	x		x		x	x	x			x	x				x	
Sparx Systems	Enterprise Architect	x		x	x	x					x		x			x	
Open Source	Argo UML	x		x	x	x					x						
Select	Select solution for MDA	x	x	x	x	x									x		
Gentle-ware	Poseidon	x		x	x	x				x	x						x

Tabelle 2: Werkzeuge für den Architekturentwurf

Jedes der getesteten Werkzeuge bietet eine UML-Unterstützung, die beim Architekturentwurf wesentlich ist. Aus den erzeugten UML-Diagrammen können von jedem Programm automatisch Quelltexte oder Quelltextfragmente erzeugt werden.

[9] Vgl. http://www3.software.ibm.com/ibmdl/pub/software/rational/web/datasheets/rsa.pdf (1.12.2005)
[10] Vgl. http://www3.software.ibm.com/ibmdl/pub/software/rational/web/datasheets/rsm.pdf (1.12.2005)

Die Stärke von **Visual Studio .Net**[11] liegt in der Modellierung und Generierung von Code und Dokumenten. Nur Visual Studio .Net und **Rational Rose XDE Developer**[12] unterstützen die Modellierung von Datenbanken. Der **Rational Software Architect/Modeler** von IBM und das quelloffene **Eclipse**[13] (zusammen mit diversen quelloffenen und kommerziellen Plugins[14]) liefern für den Architekturentwurf den größten Funktionalitätsumfang. Außer Eclipse liefert beispielsweise nur noch **Poseidon**[15] die Möglichkeit, einen Instant Messenger zur Kommunikation im Team einzubinden. **Visual Paradigm**[16] unterstützt im Gegensatz zu den meisten anderen Programmen die Möglichkeit, die Entwicklung des Codes von der Analyse- über die Design- bis hin zur Implementierungsphase zurückzuverfolgen (*Design Traceability*). Diese Funktion liefert ansonsten nur noch Eclipse[17]. Außerdem liefert Visual Paradigm genauso wie **Borland Together**[18] die Möglichkeit, das Layout der Diagramme automatisch zu gestalten. Der **Enterprise Achitect**[19] von Sparx Systems ist das einzige Werkzeug, das zusätzlich das Testen (Unittests, Integrationstests, Systemtests, Akzeptanztests) unterstützt. **Argo UML**[20] und **Select Solution for MDA**[21] liefern im Wesentlichen die gleichen Funktionen und unterscheiden sich lediglich in der Dokumenterzeugung bzw. Datenbankmodellierung. Zu erwähnen ist noch, dass die modellgetriebene Entwicklung von den Werkzeugen Eclipse[22], den verschiedenen Werkzeugen von IBM sowie Select unterstützt wird.

[11] http://www.microsoft.com/germany/msdn/visualstudio/default.mspx (1.12.2005)
[12] http://www-306.ibm.com/software/rational/offerings/design.html (1.12.2005)
[13] http://www.eclipse.org (1.12.2005)
[14] http://www.eclipseplugincentral.com (1.12.2005)
[15] http://www.gentleware.com (1.12.2005)
[16] http://www.visual-paradigm.com/ (1.12.2005)
[17] Z.B. durch das xProcess-Plugin: http://www.ivis.com/public/products/xprocess/index.cfm (1.12.2005)
[18] http://www.borland.com/together/index.html (1.12.2005)
[19] http://www.sparxsystems.com.au/ea.htm (1.12.2005)
[20] http://argouml.tigris.org/ (1.12.2005)
[21] http://www.selectbs.com/ (1.12.2005)
[22] Z.B. durch das Plugin von openarchitectureware http://architecturware.sourceforge.net/ (1.12.2005)

2.3 Werkzeuge für die Implementierung

Das Ergebnis der Analyse für Werkzeuge der Implementierungsphase hinsichtlich ihrer Eignung zur kollaborativen Softwareerstellung ist nachfolgend dargestellt:

Hersteller	Tool Name	Unterstützung mehrerer Plattformen	Unterstützung mehrerer Programmiersprachen	Device Application	Entwicklung von Servern und Datenschichten	Entwicklung und Wartung von Applikationen	Unterstützung bereits existierender Projekte	Mobile Internetapplikationen	Integriertes Testwerkzeug	Integrierter Debugger	Unterstützung von Refactoring	Übertragbarkeit des Projekts	Unterstützung mehrerer Datenbanken	Entwicxklung von Plugins	Dokumentations- und Hilfefunktion	Projektmanagement	Versionskontrolle
Microsoft	Visual Studio .Net		x	x	x	x	x	x	x	x		x	x		x	x	x
Open Source (IBM)	Eclipse	x	x	x		x	x		x	x	x	x	x	x	x	x	x
JetBrains	IntelliJ	x								x	x				x		x
IBM	Rational Software Architect		x		x	x	x	x	x		x			x	x		x
Borland	Jbuilder	x		x		x		x		x	x			x	x	x	x
Open Source (Sun)	NetBeans	x	x	x				x	x	x	x	x	x	x	x	x	x

Tabelle 3: Werkzeuge für die Implementierung

Die getesteten Werkzeuge liefern im Großen und Ganzen die gleiche umfassende Funktionalität und unterscheiden sich nur in sehr wenigen Punkten. Auszunehmen ist hiervon **IntelliJ**[23], welches nur relativ wenige kollaborative Funktionen liefert.

Visual Studio .Net bringt alle wesentlichen Funktionen mit, die für die Implementierung wichtig sind. Hierbei fehlen nur die Unterstützung für verschiedene Plattformen, die Möglichkeit, fehlende Funktionalitäten durch Plugins nachzurüsten sowie Unterstützung für *Refactoring*.

Das quelloffen erhältliche Werkzeug **Eclipse** bietet ebenfalls fast alle genannten Funktionen, wobei eine große Stärke in der Erweiterbarkeit durch Plugins und der durchdachten API hierfür liegen. Lediglich im Bereich Mobile Web und Serverentwicklung fehlen einige Funktionen.

[23] http://www.jetbrains.com/idea/features (1.12.2005)

Auch **Netbeans**[24] bringt von Haus aus viele wichtige Funktionen mit. Die Schwächen liegen hier ebenfalls im Bereich der Serverentwicklung sowie in der Unterstützung der Wartung bereits existierender Projekte.

Borlands **Jbuilder**[25] liefert ebenfalls viele für die Implementierung wichtige Funktionen, allerdings fehlen hier so grundlegende Funktionen wie z.B. ein integriertes Testtool oder die Unterstützung verschiedener Datenbanken.

Der **Rational Software Architect**[26] kann genauso wie Visual Studio .Net nicht mehrere Plattformen unterstützen, ebenso fehlen Projektmanagementfunktionen. Ansonsten unterstützt er aber auch alle wichtigen Funktionen.

[24] http://www.netbeans.org/products/ide/features.html (1.12.2005)
[25] http://www.borland.com/jbuilder (1.12.2005)
[26] http://www-306.ibm.com/software/awdtools/architect/swarchitect/index.html (1.12.2005)

2.4 Werkzeuge für das Testen

Bei den Werkzeugen für das Testen wird zwischen Debugging- und den eigentlichen Test-Werkzeugen unterschieden. Während mit den Debugging-Werkzeugen während dem eigentlichen Testen aufgetretene Fehler beseitigt werden sollen, können mit den Test-Werkzeugen die restlichen Aufgaben des Testens durchgeführt werden. Nachfolgend zunächst die Ergebnisse der Analyse für die Debugging-Werkzeuge:

Hersteller	**Tool Name**	Generierung von Berichten	Rückverfolgung von Änderungen	Zeitliche Rückverfolgung	Patch Viewer	Plausibilitätsprüfung	Anfragesystem	Kommentierung und Verlinkung mit dem Fehler	Ablauforganisation	Import/Export von Daten	Diskussionsforum	Wissensdatenbank	Internetbasierte Rückverfolgung	Unterstützung mehrerer Projekte	Rollenbasierte Sicherheit	Unterstützung von Gruppen	E-mail Benachrichtigung	Internetbasierte Administration
IBM	Rational Clear Quest	x	x						x		x		x	x	x	x	x	
Open Source	Bugzilla	x		x	x	x	x	x								x	x	x
TrackStudio	TrackStudio	x	x						x				x	x	x	x	x	
NetResults	Problem-Tracker	x	x	x					x	x	x	x		x	x	x	x	x
Compuware	TrackRecord	x							x								x	x

Tabelle 4: Debugging-Werkzeuge

Bei diesen Werkzeugen sind mitunter große Unterschiede in den Funktionen feststellbar: **Trackrecord**[27] liefert beispielsweise nur wenige Funktionen wie die Generierung von Berichten oder die Möglichkeit von E-Mail-Benachrichtigungen. Allerdings bietet es genauso wie **ProblemTracker**[28] und **Bugzilla**[29] die Möglichkeit einer webbasierten Verwaltung. ProblemTracker liefert zusätzlich als einziges Programm die Möglichkeit, verschiedene Inhalte zu importieren und zu exportieren. Außerdem gibt es hier die Möglichkeit, eine Wissensdatenbank aufzubauen, in die man dann etwa einen Bereich für bekannte Probleme oder häufig gestellte Fragen (FAQ) integrieren kann. Zudem

[27] http://www.compuware.com/products/qacenter/405_ENG_HTML.htm (1.12.2005)
[28] http://www.problemtracker.com/pt_home.html (1.12.2005)
[29] http://www.bugzilla.org (1.12.2005)

liefert ProblemTracker ebenso wie **Rational Clear Quest**[30] die Möglichkeit, ein Diskussionsforum zu integrieren. Weiterhin unterstützen sowohl diese beiden Produkte als auch **Trackstudio**[31] die Möglichkeit, in einem Projekt unterschiedliche Rollen zu vergeben. Bugzilla unterscheidet sich von den anderen Produkten deutlich. So gibt es hier beispielsweise nicht die Möglichkeit, Rollen zu vergeben, allerdings liefert es als einziges Programm eine Plausibilitätsprüfung sowie die Möglichkeit, den Fehler mit einem Kommentar mit dem Fehlerbericht zu verlinken.

Die Analyse einer übersichtlichen Auswahl der eigentlichen Test-Werkzeuge ergab folgendes Ergebnis:

Hersteller	**Tool Name**	Unterstützung mehrerer Programmiersprachen	Unterstützung mehrerer Plattformen	Analyse der Testabdeckung	Multihasen-Applikationen	Generierung von Berichten	Integration bereits existierender Prozesse	Integration von Capture/Relay-Werkzeugen	Übersicht über die Performance der Komponenten	Automatische Analyse der Programmqualität	Erkennung von Speicherlecks	CPU profiling	Debuggen von Threads	Unterstützung von Anwendungen mit verschiedenen Protokollen	Regressionstests	Intelligente Aufzeichnung der Testscripte	Unterstützung mehrerer Compiler	Verteiltes Testen	Objektgetriebenes Testen	Agent für Debuginformationen	Komprimieren von Testergebnissen
Borland	Optimizeit		x	x	x	x	x		x	x	x	x	x	x	x						
Software Research	TestWorks		x	x										x	x						
QCS	Cantata		x	x	x	x	x	x													
PushToTest	TestMaker		x	x		x								x							
AutomatedQA	TestComplete	x			x											x	x	x	x	x	x
Mercury	Quality Center		x			x									x	x					

Tabelle 5: Testing-Werkzeuge

Die größte Funktionalität liefert bei dieser Werkzeugkategorie **Optimizeit**[32]. Es liefert als einziges Werkzeug Funktionen zum Hardwaretest, wie z.B. *CPU Profiling* oder Erkennung von Speicherlecks. Außerdem analysiert es automatisch die Qualität des zu testenden Programms und kann Threads debuggen. Deutlich weniger Funktionen besitzt **TestWorks**[33]. Es unterstützt aber mehrere Plattformen und unterschiedliche Protokolle. Außerdem bietet TestWorks eine Funktion zur Analyse der Testabdeckung

[30] http://www-306.ibm.com/software/awdtools/clearquest (1.12.2005)
[31] http://www.trackstudio.com/ (1.12.2005)
[32] http://www.borland.com/optimizeit (1.12.2005)
[33] http://www.soft.com (1.12.2005)

sowie Regressionstests. **Cantata**[34] bietet als einziges Tool die Integration von Capture/Relay-Werkzeugen und kann ebenso wie Optimizeit bereits existierende Prozesse integrieren. **TestMaker**[35] liefert ebenso wie TestWorks im Wesentlichen nur Basisfunktionen wie die Unterstützung verschiedener Plattformen und verschiedener Protokolle sowie die oben genannte Testabdeckungsanalyse. Außerdem kann es ebenso wie Cantata und Optimizeit automatisch Berichte generieren. **Testcomplete**[36] unterscheidet sich schließlich völlig von den anderen Testwerkzeugen. Es unterstützt zwar genauso wie Optimizeit und Cantata Multiphasen-Applikationen, bietet ansonsten jedoch keine der Funktionen der anderen Werkzeuge. Allerdings liefert es als einziges Programm u.a. eine Unterstützung für unterschiedliche Programmiersprachen und Compiler sowie verteiltes und objektorientiertes Testen. Außerdem ist es als einziges Programm in der Lage, die Testergebnisse zu komprimieren. Neben TestComplete kann nur noch das **Quality Center**[37] von Mercury Testscripte intelligent aufzeichnen.

[34] http://www.qcsltd.com/cantata/cantata.htm (1.12.2005)
[35] http://www.pushtotest.com/ (12.12.2005)
[36] http://www.automatedqa.com/products/testcomplete/tc_features.asp (12.12.2005)
[37] http://www.mercury.com (12.12.2005)

3 Anforderungen an Kollaborationswerkzeuge

Zunächst soll in diesem Kapitel auf die für den Softwareerstellungsprozess benötigten kollaborativen Funktionen eingegangen werden. Anschließend werden aktuelle Kollaborationsplattformen vorgestellt und analysiert.

3.1 Benötigte kollaborative Funktionen

Die Beteiligung mehrerer Stakeholder mit unterschiedlichen Interessen und Blickwinkeln führt zu speziellen Anforderungen an die Werkzeugunterstützung der verschiedenen Aktivitäten im Softwareentwicklungsprozess. Es müssen daher sowohl die einzelnen Phasen mit ihren jeweiligen Aufgaben durch entsprechende Werkzeugen unterstützt werden (vgl. Kapitel 2) als auch eine zentrale Kollaborationsplattform zur Projektkoordination bereit stehen.

Die Projektkoordination erfolgt im Open Source-Umfeld hauptsächlich über zentrale Plattformen im Netz, wie SourceForge, und mittels Standardkomponenten für Versionskontrolle (Subversion, CVS etc.) sowie Gruppenkommunikation (Mailinglisten, Issue Tracker, Foren etc.). Hersteller von kommerziellen Softwareentwicklungswerkzeugen bieten in letzter Zeit immer mehr Teamfunktionalitäten mit zum Teil zentrale Infrastruktur an und unterstützen somit vermehrt auch kollaborative, verteilte Softwareerstellungsprozesse. Im nächsten Abschnitt sollen daher aktuelle Kollaborationsplattformen vorgestellt und analysiert werden. Die Ergebnisse basieren auf der bereits weiter oben erwähnten Marktstudie des Forschungsprojekts CollaBaWü (vgl. Kapitel 2). Zwar bieten die untersuchten Plattformen teilweise auch phasenbezogene Funktionen wie bspw. Source Code Engineering an, fokussieren sich aber im Wesentlichen auf allgemeinere Projektmanagementfunktionen – z.B. Statusverfolgung. Im Sinne einer Client/Server-Architektur können sich daher arbeitsplatzbezogene Werkzeuge (CASE Tools und IDEs, vgl. Abschnitt 2) und zentrale Projektkoordinationplattformen ideal ergänzen.

3.2 Kollaborationsplattformen

In diesem Kapitel werden nun Werkzeuge untersucht, die sowohl die Phasen des Konfigurations- und Änderungsmanagements als auch des Projektmanagements unterstützen. Bis auf die Rational Suite, die nachfolgend nichtsdestotrotz aufgeführt wird, stellen diese Werkzeuge so genannte Kollaborationsplattformen, auch CSD[38]-Plattformen genannt, dar. Die Ergebnisse der Marktstudie sind in folgender Tabelle festgehalten:

Hersteller	**Tool Name**	**Version**	Versionskontrolle	Problemverfolgung	Projektverfolgung und -berichte	Projektmanagement	Wissenserfassung	Dokumenten-/Dateimanagement	Mailinglisten	Diskussionsforum	Nachrichtenfunktion	Netzbasierte Administration	E-Mail-Integration	Ingenieurmäßige Quelltextbearbeitung	Suche nach Schlüsselwörtern	Umfrage-Tools	Schnipselbibliothek	Echtzeitkommunikation	Planungswerkzeug
Intland	Codebeamer	3.7	x	x	x	x	x	x	x	x	x	x	x	x	x			x	x
CollabNet	SourceCast Ent.	4.0	x	x	x	x	x	x	x	x	x	x			x			x	
VA Software	SourceForge Ent.	4.2	x	x	x	x	x	x	x	x	x	x			x				
Open Source	GForge	4.5	x	x	x	x		x	x	x	x	x			x	x	x		
IDEs mit Erweiterungen zur Anbindung an Kollaborationsplattformen																			
Open Source (Sun)	NetBeans[39]	4.1	x					x						x					
Open Source (IBM)	Eclipse	3.1.1	x	x		x		x			x			x	x				
IBM	Rational Suite	?[40]	x	x	x	x	x	x				x		x	x				
Borland	Application Lifecycle	?[41]	x	x	x	x		x		x		x		x	x				
Microsoft	Visual Studio	2005	x	x		x		x		x		x		x	x		x		

Tabelle 6: Übersicht Kollaborationsplattformen

[38] CSD = Collaborative Software Development
[39] Vgl. http://www.netbeans.org/catalogue/ (12.12.2005)
[40] Stand 1.12.2005 lt. http://www-306.ibm.com/software/awdtools/suite/index.html (12.12.2005)
[41] Stand 9.12.2005 lt. http://www.borland.com/de/products/alm/index.html (12.12.2005)

Alle Kollaborationsplattformen liefern die Möglichkeit zur *Versionskontrolle*, d.h. zur Versionierung der im Projekt bearbeiteten Artefakte und Dateien. Diese Aufgabe übernehmen bei allen Werkzeugen externe Programme und Standardkomponenten wie z.B. Subversion[43] oder CVS[44]. Die meisten externen Programme werden von Intlands **CodeBeamer**[45] unterstützt. Lediglich Microsofts **Visual Studio**[46] und Borlands **Application Lifecycle**[47] bieten hier ein eigenes System an.

Zur *Problemverfolgung* (Issue Tracking) gehören unter anderem eine Volltextsuche oder automatische E-Mail-Benachrichtigung bei Änderungen. **Collabnet Enterprise**[48], **Sourceforge Enterprise**[49] und **CodeBeamer** bieten in dieser Kategorie die meisten Funktionen, die **Rational Suite** von IBM[50] und **GForge**[51] (der quelloffene Ableger von Sourceforge) liefern nicht ganz so viele Funktionen. Die Eclipse-basierte Entwicklungsumgebung, die mit entsprechenden Plugins[52] erweitert werden kann, liefert hierzu kaum Funktionen. Die Einbindung an Bugzilla wird von ihr allerdings sehr gut unterstützt. Generell besteht auch immer die Möglichkeit, IDEs wie Eclipse und NetBeans über das Versionierungssystem an die entsprechenden Hosting-Plattformen anzubinden.

Projektverfolgung und -berichte (Project Tracking/Reporting) beinhalten die Verwaltung des gesamten Projekts und die Verfolgung von Änderungen im Projekt. Hier liefern außer Eclipse alle Werkzeuge in etwa die gleiche Funktionalität, lediglich **CollabNet** hebt sich ein wenig durch seine Suche in Parallelprojekten hervor.

Zum *Projektmanagement* (Project Management) gehört insbesondere die Möglichkeit, individuelle Rollen zu vergeben und Aufgaben in Form von Aufgabenlisten zu verteilen. So hat z.B. ein Projektmanager andere Rechte und Möglichkeiten als ein Programmierer bzw. ein Anwender. Weiterhin gehören hierzu die Möglichkeiten, das Projekt webbasiert zu verwalten oder persönliche Aufgabenlisten für jeden der Beteiligten zu erstellen. Auch hier liefern im Wesentlichen alle Programme ähnliche Funktionen. CodeBeamer, Sourceforge, Collabnet sowie die Rational Suite liefern außerdem Möglichkeiten zur systematischen *Wissenserfassung* (Knowledge Capturing).

Zum *Dokumenten-/Dateimanagement* (Document / File Management) zählen einerseits die Versionierung von Dokumenten und andererseits der Rollen-basierte Zugriff auf diese Dateien. Diese Funktionen liefern wiederum im Wesentlichen alle betrachteten Werkzeuge und Kollaborationsplattformen. **CodeBeamer** liefert hier zusätzlich noch eine Zugriffsverfolgung und eine E-Mail Benachrichtigung, wenn ein Dokument gelesen und/oder verändert wird. Dies kann in einem kollaborativen, verteilten Szenario eine erhebliche Reduktion von Koordinations- und Kommunikationsaufwand bedeuten.

Außerdem ermöglichen es alle Programme außer den in Tabelle 6 aufgeführten IDEs, Mailinglisten einzurichten und zu verwalten. Zusätzlich bieten die reinen Kollaborationsplattformen sowie **Eclipse** die Möglichkeit, Nachrichten zu verwalten und zu publi-

---

[43] http://subversion.tigris.org/ (12.12.2005)
[44] http://www.nongnu.org/cvs/ (12.12.2005)
[45] http://www.intland.com (12.12.2005)
[46] http://msdn.microsoft.com/vstudio/products/vsts/suite/default.aspx (12.12.2005)
[47] http://www.borland.com/de/products/alm/index.html (12.12.2005)
[48] http://www.colab.net (12.12.2005)
[49] http://www.vasoftware.com/sourceforge (12.12.2005)
[50] http://www-306.ibm.com/software/awdtools/suite/index.html (12.12.2005)
[51] http://www.gforge.org (12.12.2005)
[52] http://www.eclipse-plugins.info (12.12.2005)

zieren. Dies kann z.B. durch einen RSS-Feed oder durch ein Wiki, wie beispielsweise TWiki[53], geschehen.

Mit der *netzbasierten Administration* (Web-Based Administration) bieten alle CSD-Plattformen die Möglichkeit, die Projekte über das Internet zu verwalten. **CodeBeamer** ermöglicht weiterhin mit der *E-Mail-Integration* (Email Integration) externe E-Mail-Clients wie z.B. Thunderbird in das Programm zu integrieren.

Zur *ingenieurmäßigen Quelltextbearbeitung* (Source Code Engineering) gehören z.B. Refactorings (bspw. die projektweite Umbenennung von Variablen- und Methodennamen) sowie die Möglichkeit, UML Diagramme direkt in Code bzw. Quelltextfragmente zu übersetzen. Diese Möglichkeit bieten **Eclipse, NetBeans** und **CodeBeamer**. Das Source Code Engineering bildet ebenso die Stärke der **Rational Suite**: von ihr werden alle Phasen der Projektentwicklung unterstützt. So gibt es hier z.B. automatische Tests oder eine automatische Benachrichtigung, wenn Testskripte geändert werden müssen.

Alle Werkzeuge liefern zudem die Möglichkeit, das gesamte Projekt nach *Schlüsselwörtern zu durchsuchen* (Search Code and Content). Daneben bietet **GForge** als einziges Werkzeug die Möglichkeit, strukturierte *Umfragen* (Surveys) unter den Projektbeteiligten zu starten. Zudem kann es so wie **Visual Studio** auch einzelne *Codeschnipsel* verwalten. Bei **Collabnet** und **CodeBeamer** gibt es zudem einige Funktionen zur *Echtzeitkommunikation* (Real-Time Communication), so z.B. die Möglichkeit Programme für Chats, Videotelefonie usw. einzubinden. CodeBeamer ermöglicht es immerhin, den Onlinestatus der anderen Beteiligten (Awareness) zu sehen. **Eclipse** bietet mit dem Eclipse Communication Framework (ECF)[54] eine grundlegende API zur Einbindung von Werkzeugen zur Echtzeitkommunikation – entsprechend ausgereifte Plugins müssen noch implementiert werden. **CodeBeamer** beinhaltet zusätzlich als einziges Programm einen Zeitplaner (Scheduler) zum automatischen Start von Testscripten oder zur automatischen Erzeugung der Dokumentation (z.B. im javadoc-Format).

Zusammenfassend kann konstatiert werden, dass sich CodeBeamer, Sourceforge und CollabNet sowie GForge funktional nur sehr geringfügig unterscheiden. Allerdings weist **CodeBeamer** im direkten Vergleich die größte Funktionsvielfalt auf und bietet auch ein Plugin zur Anbindung an Eclipse. Zudem können sich kommerzielle Anbieter von Kollaborationsplattformen über die mitgelieferte Dokumentation der Software sowie die Support-Dienstleistungen differenzieren. Des Weiteren müssen solche Plattformen sicherlich für einige Projekte zusätzlich angepasst werden, was nur im Fall des quelloffen erhältlichen GForge ohne Inanspruchnahme des Herstellers möglich ist.

Die Stärken der Rational Suite, die mittlerweile auch größten Teils auf eine Eclipse-Architektur aufbaut, liegen ganz klar in der Entwicklung und der automatischen Codeerzeugung sowie dem Testen. Am Wenigsten liefern in diesem Bereich derzeit noch die selbst zusammengestellten Eclipse- bzw. NetBeans-basierten Lösungen mit unterschiedlichen Plugins, allerdings gibt es hier noch sehr gute Erweiterungsmöglichkeiten.

Für eine noch detailliertere Gegenüberstellung der wichtigsten Kollaborationsplattformen sei auf Tabelle 8 und Tabelle 9 im Anhang verwiesen.

[53] http://twiki.org/ (12.12.2005)
[54] http://www.eclipse.org/ecf/ (12.12.2005)

4 Zusammenfassung

Als Fazit der Werkzeuganalyse lässt sich konstatieren, dass große Firmen (IBM, Microsoft, Borland) den kompletten Softwareentwicklungsprozess mit ihren Paketen abdecken und verstärkt kollaborative Funktionen in ihre Komplettpakete mit einbinden. Dadurch decken sie einen Großteil des Funktionsumfangs der CSD-Plattformen mit ab (siehe Rational Suite im vorangegangenen Kapitel). Kollaborationsplattfomren bieten allerdings spezielle Funktionen, die Komplettpakete nicht bieten: Webadministration von Projekten mit Hilfe von Browsern (Codebeamer fährt dabei Zweigleisig mit einer Browser-Client-Version und einer Installationsversion), sowie eine Integration von vielen verschiedenen SE-Tools.

Bedingt durch den erhöhten Interaktions- und Kommunikationsgrad haben vor allem Werkzeuge für die Anforderungsanalyse und das Debuging viele kollaborative Funktionen integriert. Es fällt zudem auf, dass kleinere Softwareentwicklungswerkzeughersteller ihre Produkte mit kollaborativen Funktionen ausbauen. So ist z.B. Trackstudio eigentlich ein Debugging Tool, stellt vom Umfang her aber schon fast eine vollwertige CSD-Plattform dar.

Eine synchrone Kommunikation[55] wird fast überhaupt nicht unterstützt. Die Ausnahme stellen hier zwei Design-Werkzeuge (Eclipse und Poseidon mit Instant Messaging) dar. Open Source-Produkte sind in den CSD-Plattformen oft durch ihre Plugin-Funktionen und Schnittstellen eingebunden (Beispiel: Eclipse bei CodeBeamer).

Bei den Testing-Tools sind erwartungsgemäß keine kollaborativen Funktionen implementiert. Es fällt zudem auf, dass die einzelnen Testing-Tools teilweise auf bestimmte Plattformen spezialisiert sind und ihre Testfunktionen daher sehr unterschiedlich sind.

[55] Synchrone kollaborationsfunktionen für SE-Werkzeuge könnten sein: Online Voting, Shared Whiteboard, VoIP, Instant Messaging

Anhang A

Name	Hersteller
CRADLE/REQ	3SL (Structured Software Systems)
ARTiSAN Real-Time Studio	ARTiSAN
TestComplete	AutomatedQA
C++ BuilderX	Borland
CaliberRM	Borland
Jbuilder	Borland
Kylix	Borland
Optimizeit	Borland
Together	Borland
Bugzilla	Bugzilla.org
TrackRecord	Compuware Corporation
JUDE	Eiwa System Management
Poseidon	Gentleware
Analyst Pro	Goda
IBM series for Software Design & Construction	IBM
Rational RequisitePro	IBM
Rational Rose XDE developer	IBM
RationalClearCase	IBM
Websphere	IBM
ArcStyler	Interactive Objects
IntelliJ IDEA	Jet Brains
TrackGear	LogiGear
MagicDraw series	MagicDraw
Visual Studio	Microsoft
ObjectiF	microTOOL
ProblemTracker	NetResults
Eclipse	Open Source
Kdevelop	Open Source
NetBeans	Open Source
TestMaker	PushToTest
Segue	Segue Software
Select solution for MDA	Select Business Solutions
TestCoverage	SemanticDesigns
TestRobot	SoftRobot
TestWorks	Software Research
CARE	SOPHIST
Enterprise Architect	Sparx Systems
SpeeDEV	SpeeDEV
Catalyze	SteelTrace
Java Studio Creator	Sun Microsystems
PowerDesigner	Sybase
TeamTrack	TeamShare
DOORS	Telelogic
TAU/Developer	Telelogic
TrackStudio	TrackStudio
ArgoUML	University of California, Irvine
Visual Paradigm	Visual Paradigm

Tabelle 7: Liste der analysierten Werkzeuge

Anhang B

	Collabnet Sourcecast	Sourceforge	Codebeamer	Rational Suite
version control	CVS, Subversion	CVS, Subversion, ClearCase	CVS, Subversion integrated SCM API for PCVS, CMSynergy, Sourcesafe, ClearCase	Rational ClearCase
issue tracking	Tracking: task, bug, Requirement, Feature request, change request,			
	Attachements to Issues			
	Real Time Reports on trends and activities			
	Automated email notification on changes			
			Association with other issuses	
	History / full text search			
	Issues Association with source code			
	Role based access permission			
	Web Accessability			
project tracking/reporting	Queries on Data of Multiple Projects			
	Customizeable Data Definitions to track			
	Dashboard			
project management	Role based permissions			
	Task Manager for Project Plans and deliverables, personalized to-Do Lists			
	Web Administration			
knowledge capturing				
Document / file management	Knowledge capturing in one Repository by document versioning			
	Role based Access			
			Access Tracker	
			Email notification on document Read/Write	
		automated document routing		
mailing lists	customizeable Mailing lists, message archive, multiple subscription types, notification on tracked issues			
discussion forum	Role based permissions, Email notification, read notification, remote submission via email, attatchments			
news	project news			
web-based administration	Administration via a Browser			
Email integration			Integration of Emailclient	
Source Code Engineering			Integration in IDEs	All Phases supported
Search Code & Content	Search Code & Content			
Surveys				
Real-Time Communication			See online status of other users	
	Integraton of tools for chat, online meetings, voice and video chat			
Scheduler			Integrated scheduler for builds, unit tests, documentation generation, external analyzer	
History			View user history with visited artifact links	

Tabelle 8: Kollaborationsplattformen en detail – Teil 1

	Gforge	Eclipse	Microsoft	Borland
version control	CVS, Subversion, Clear Case	CVS, Subversion, ClearCase	integrated	StarTeam
issue tracking	Tracking: task, bug, Requirement, Feature request, change request,			
	Attachements to Issues			
				Automated email notification on changes
				History / full text search
	Issues Association with source code			
	Role based access permission			
	Web Accessability			
project tracking/reporting	Customizeable Data Definitions to track			
				Dashboard
	Role based permissions			
project management		Task Manager for Project Plans and deliverables, personalized to-Do Lists		
	Web Administration			
knowledge capturing	Knowledge capturing in one Repository by document versioning			
Document / file management	Role based Access			
	customizeable Mailing lists, message archive, multiple subscription types, notification on tracked issues			
mailing lists	Role based permissions, Email notification, read notification, remote submission via email, attatchments			
discussion forum	project news			
news	Administration via a Browser			
web-based administration				
Email integration				
Source Code Engineering	Search Code & Content			
Search Code & Content	Surveys for users & administrators			
Surveys				
Real-Time Communication				
Scheduler				
History				

Tabelle 9: Kollaborationsplattformen en detail – Teil 2

Literaturverzeichnis

[Beyer und Holtzblatt 1995] Beyer, H.R.; Holtzblatt, K.: Apprenticing With the Customer; Communications of the ACM, 1995, 38, 45-52

[Cook und Churcher 2003] Cook, C.; Churcher, N.: An Extensible Framework for Collaborative Software Engineering; Proceedings of the 10th Asia-Pacific Software Engineering Conference (APSEC'03), IEEE, 2003, 290-30

[French und Layzel 1998] French, A.; Layzel, P.: A Study of Communication and Cooperation in Distributed Software Project Teams; IEEE Int. Conference on Software Maintenance, Bethesda, 1998, 146-155

[Geisser und Hildenbrand 2005] Geisser, M.; Hildenbrand, T.: Eine Methode zur kollaborativen Anforderungserhebung und entscheidungsunterstützenden Anforderungsanalyse - Working Paper 6/2005 - August 2005; Working Papers in Information Systems - University of Mannheim - Department of Information Systems 1, 2005

[Hood u.a. 2005] Hood, C.; Kress, A.; Stevenson, R.; Versteegen, G.; Wiebel, R.: iX-Studie zum Thema Anforderungsmanagement - Methoden und Techniken, Einführungsszenarien und Werkzeuge im Vergleich; Heise Zeitschriften Verlag, 2005

[Karolak 1998] Karolak, D.W.: Global software development: managing virtual teams and environments; Wiley-IEEE Computer Society Pr, 1998

[Lloyd u.a. 2002] Lloyd, W.J.; Rosson, M.B.; Arthur, J.D.: Effictiveness of Elicitation Techniques in Distributed Requirements Engineering; Proceedings of the IEEE Joint International Conference on Requirements Engineering (RE'02), 2002

[Rashid u.a. 2006] Rashid, A.; Behm, A.; Geisser, M.; Hildenbrand, T.: Kollaborative Softwareentwicklung – Zum Kollaborationsbegriff; Arbeitspapier fzi/2006/1 an der Universität Karlsruhe, FZI Forschungszentrum Informatik, 2006

[Schienmann 2002] Schienmann, B.: Kontinuierliches Anforderungsmanagement; Addison-Wesley, 2002

[Sommerville 2004] Sommerville, I.: Software Engineering, Addison-Wesley, 2004

[StandishGroup 2002] The Standish Group: What Are Your Requirements?, 2002

[Versteegen 2004] Versteegen, G.: Anforderungsmanagement; Springer, 2004

[Yphise 2005] Yphise: Requirements-driven application lifecycle management - Executive Volume; Yphise Software Product Assessment, 2005