Seminar
Evolution von Softwaresystemen
WS 2001/2002

Ein graphbasiertes Managementsystem für dynamische Entwicklungsprozesse

(Bernhard Westfechtel)

Verfasser: Thomas Matheis
Betreuer: Carsten Görg

Motivation

Da Softwaresysteme immer komplexer werden, ziehen sie auch eine lange Entwicklungs- und Wartungsgeschichte hinter sich her. Im Laufe der Evolution eines Softwaresystems ist daher eine Vielzahl verschiedener Versionen entstanden, die mehr oder weniger strukturiert oder unstrukturiert im Unternehmen präsent sind. Da man aus Fehlern vergangener Tage auch lernen kann und eventuell auch auf ältere Versionen zurückgreifen muss, ist es für jedes Unternehmen von Vorteil, die Geschichte ihrer Softwaresysteme aufzuzeichnen, um daraus bei Bedarf wieder Nutzen zu ziehen.
Im Rahmen des Seminars Evolution von Softwaresystemen wurden Konzepte und konkrete Produkte vorgestellt, mit denen sich Evolutionen von Softwaresystemen beschreiben lassen. Das Konzept intensionaler und extensionaler Systeme, sowie die Produkte CVS und RCS nahmen Bezug auf das Konfigurationsmanagement. Auch neuere Ansätze zum Thema Evolution von Softwaresystemen, wie die Visualisierung der Evolution, Refactoring Pattern und Evolutionsabkommen wurden behandelt. In einer Fallstudie wurde die Problematik bei der Erfassung einer Evolution deutlich. Im Folgenden wird ein Managementsystem vorgestellt, in das einige Konzepte der Evolution von Softwaresystemen integriert sind. Die dafür notwendigen theoretischen Grundlagen der Graphtransformationen wurden ebenfalls im Seminar behandelt.

1 Einleitung

Um ihre Produkte am Markt konkurrenzfähig zu halten, müssen Unternehmen ihre Geschäftsprozesse ständig optimieren. Daher führt man Managementsysteme ein, deren Hauptaufgabe darin besteht, die Komplexität von Geschäftsprozessen zu verringern. Dadurch können die Entwicklungskosten gesenkt, die Entwicklungszeiten verkürzt und die Qualität der Produkte erhöht werden.
Geschäftsprozesse können in statische und dynamische Prozesse unterteilt werden. Für das Management von statischen Prozessen steht schon eine Reihe von Systemen zur Verfügung, die sich in der Praxis bewährt und Akzeptanz gefunden haben. So werden zum Beispiel Workflowmanagementsysteme für Routinetätigkeiten im Bürobereich einer Bank eingesetzt.
Während verfügbare Systeme zum Management von statischen Prozessen die Prozesse auf ganzheitlicher Ebene beschreiben, unterstützen Systeme zum Management von dynamischen Prozessen bisher nur Teilaspekte der Prozesse. Die Gründe hiefür liegen darin, dass dynamische Prozesse im Gegensatz zu den statischen einen geringen repetitiven Anteil aufweisen, schwer planbar sind und im Allgemeinen Unikate sind. Aus den genannten Gründen existieren derzeit nur wenige Systeme, die das Management von dynamischen Prozessen auf ganzheitlicher Ebene unterstützen.
Im Folgenden wird das AHEAD-System[1] behandelt, das uns einen Ansatz zum Management von dynamischen Prozessen auf ganzheitlicher Ebene liefert.

[1] AHEAD = Adaptable and Human-Centered Environment for the Management of Development Processes

2 Das AHEAD-System

Die folgende Abbildung zeigt das AHEAD-System mit seinen Komponenten im Überblick.

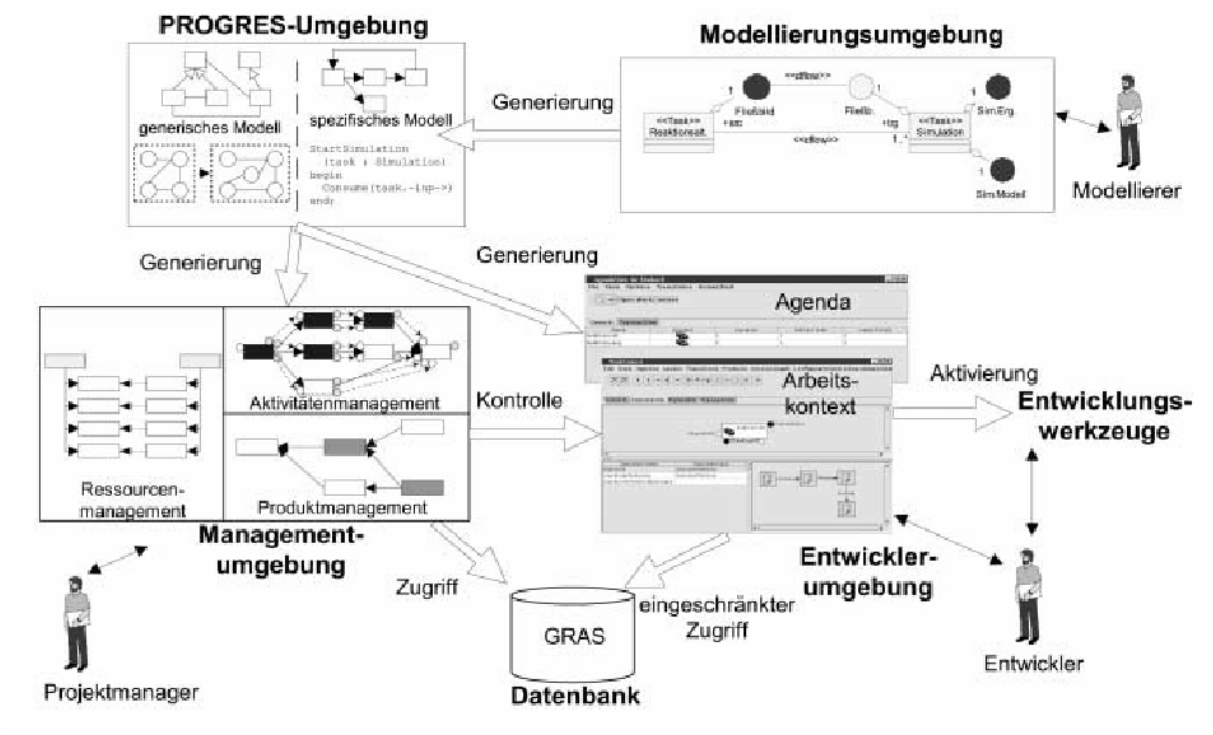

Abbildung: Das AHEAD-System im Überblick

Das AHEAD-System besteht aus einer externen und einer internen Ebene. Die externe Ebene umfasst die Managementumgebung und die Entwicklerumgebung. Sie stellt dem Anwender des AHEAD-Systems graphische Werkzeuge zur Laufzeit eines Projektes zur Verfügung. Die interne Ebene besteht aus der PROGRES-Umgebung und der Modellierungsumgebung. Hier werden die formalen Spezifikationen zur Definitionszeit eines Projektes festgelegt.

Die Managementumgebung setzt sich aus dem Aktivitätenmodell, dem Produktmodell, sowie dem Ressourcenmodell zusammen. Im Aktivitätenmodell definiert der Projektmanager dynamische Aufgabennetze, mit denen sich Entwicklertätigkeiten managen lassen. Ein Aufgabennetz besteht dabei aus atomaren und komplexen Aufgaben. Atomare Aufgaben können nicht weiter sinnvoll in Teilaufgaben zerlegt werden, während komplexe Aufgaben selbst wieder für Aufgabennetze stehen. Kontrollfluss-, Daten- und Rückgriffskanten stellen die einzelnen Aufgaben in Beziehung zueinander. Rückgriffskanten überführen das Aufgabennetz zu einem dynamischen Aufgabennetz, da durch sie Rücksprünge bzw. Zyklen erfasst werden. Kontrollflusskanten nehmen auch das Konzept des Simultaneous Engineering, das heißt die Parallelisierung von Aufgaben, auf. Das Produktmodell dient zur Verwaltung der Dokumente, die während des Projektablaufes bei der Bearbeitung der Aufgaben durch die Projektmitarbeiter entstehen. Der Projektmanager verwaltet dabei so genannte Versionsgraphen, die durch Dokument- und Konfigurationsgraphen genauer spezifiziert werden. Dokumentgraphen nehmen den Inhalt einer Dokumentversion auf, während Konfigurationsgraphen aus den Dokumentversionen gebildete Konfigurationen beschreiben. Das Ressourcenmodell bietet dem Projektmanager die Möglichkeit, die ihm zur Verfügung stehenden technischen und menschlichen Ressourcen den Aufgaben des Projektes zuzuordnen. Mit den graphischen Werkzeugen, die dem Projektmanager in der Management-

umgebung zur Verfügung gestellt werden, kann dieser sein Projekt durch Aufgabennetze beschreiben, sein Projektteam aufstellen und ein Produktmanagement vornehmen. Damit kann der Projektmanager sein Projekt planen, analysieren, steuern und überwachen.

Die Entwicklerumgebung stellt einem Entwickler eine Agenda bereit. Die Agenda zeigt dem Entwickler alle Aufgaben an, die ihm der Projektmanager zugeteilt hat. Zu jeder ihm zugeteilten Aufgabe kann der Entwickler einen Arbeitskontext aufrufen. Dieser enthält Dokumente und genauere Spezifikationen, die zur Bearbeitung der Aufgabe notwendig sind. Ferner kann der Entwickler über den Arbeitskontext auch weitere Entwicklungswerkzeuge, wie zum Beispiel Editoren oder Compiler aktivieren, um die Dokumente bzw. die Aufgabe zu bearbeiten.

Die graphischen Werkzeuge, die dem Projektmanager und den Entwicklern auf der externen Ebene zur Verfügung gestellt werden, müssen vom Anwender des AHEAD-Systems nicht implementiert werden, sondern werden aus einer formalen Spezifikation generiert. Aus diesem Grund werden zur internen Modellierung Graphen und Graphersetzungsregeln eingesetzt. Alle Managementdaten, wie zum Beispiel Aufgabennetze, Produktkonfigurationen oder Ressourcenzuteilungen werden in einheitlicher Weise als attributierte Graphen repräsentiert. Operationen, die dem Projektmanager oder dem Entwickler auf der externen Ebene durch die graphischen Werkzeuge zur Verfügung stehen, werden durch Graphersetzungsregeln formal spezifiziert. Graphen und Graphersetzungsregeln werden in der graphbasierten Datenbank GRAS gespeichert.

Die PROGRES-Umgebung stellt eine Spezifikationssprache zur Verfügung, mit der die Managementdaten erfasst werden können. Eine Spezifikation in PROGRES besteht aus einem generischen und einem spezifischen Teil. Im generischen Modell werden Spezifikationen gemacht, die von einem konkreten Anwendungsbereich unabhängig sind. Diese werden nur einmal erstellt und können dann in unterschiedlichen Domänen wieder verwendet werden. Generische Spezifikationen werden vom AHEAD-System schon mitgeliefert, müssen vom Anwender also nicht explizit neu erstellt werden. Im spezifischen Modell werden Spezifikationen definiert, die das Managementsystem an einen konkreten Anwendungsbereich anpassen. Aus der Gesamtspezifikation erzeugt ein Compiler dann Quellcode, mit dem automatisch die graphischen Werkzeuge der externen Ebene generiert werden.

Die Modellierungsumgebung bietet dem Anwender die Möglichkeit, den spezifischen Teil der PROGRES-Spezifikation nicht in der PROGRES-Spezifikationssprache zu erstellen, sondern sich der UML[2] zu bedienen. Da UML weit verbreitet ist und sich zum Standard entwickelt hat, müssen sich Domänenexperten somit nicht erst die PROGRES-Spezifikationssprache aneignen. Zur Modellierung werden dabei vor allem Klassen- und Zustandsdiagramme eingesetzt. In UML erstellte Modelle werden dann automatisch durch einen Compiler in den spezifischen Teil der PROGRES-Spezifikation übersetzt.

Für detailliertere und weitergehende Informationen über das AHEAD-System sei an dieser Stelle auf [1] verwiesen. Die theoretischen Grundlagen über Graphen und Graphersetzungs-regeln finden sich in [2].

[2] UML = Unified Modeling Language; weitere Informationen enthält [4]

3 Bewertung des AHEAD-Systems

Im Folgenden soll das AHEAD-System genauer auf seine Stärken hin untersucht sowie gegenüber anderen Systemen abgegrenzt werden:

Wie in der Einleitung schon angedeutet liegt der wesentliche Vorteil des AHEAD-Systems darin, dass es dynamische Prozesse auf ganzheitlicher Ebene unterstützt, das heißt Aktivitäten, Produkte und Ressourcen eines Projektes integriert betrachtet. Bisher verfügbare Managementsysteme für dynamische Entwicklungsprozesse tragen diesem Konzept nur wenig Rechnung, da sie lediglich Teilaspekte der dynamischen Prozesse erfassen. So stehen beispielsweise Systeme zum Produktmanagement zur Verfügung, die sich in der Praxis bewährt haben und auch weit verbreitet sind. Diese Systeme sind jedoch nur auf das Management von Produkten ausgerichtet. Als Beispiele sollen hier die Systeme RCS und CVS dienen, die auch im Rahmen des Seminars behandelt wurden und in [3] genauer erläutert werden. Die genannten Systeme erstrecken sich ausschließlich auf das Konfigurations-management und können damit im AHEAD-System dem Produktmodell der Management-umgebung zugeordnet werden. Zwar sind diese Systeme in ihrer Funktionalität wesentlich umfangreicher als das Produktmodell des AHEAD-Systems. Vom groben Einsatzbereich her gesehen genügt das Produktmodell jedoch denselben Anforderungen. Auch Projekt-managementsysteme (zum Beispiel MS Project von Microsoft) und Workflowmanagement-systeme (zum Beispiel Lotus Notes von IBM) sind in ihrer Funktionalität wesentlich komplexer als das AHEAD-System, setzen ihren Schwerpunkt jedoch nur auf die Aktivitäten eines Prozesses. Desweiteren unterstützen die meisten Managementsysteme für dynamische Prozesse nur die technische Seite. Im AHEAD-System wird durch das Ressourcenmodell der Managementumgebung auch die betriebswirtschaftliche Seite beachtet. So finden sich beispielsweise im Ressourcenmodell durch die Unterscheidung von Ist-Ressourcen und Plan-Ressourcen, sowie durch die Ermittlung der Auslastungskurven von Mitarbeitern auch Konzepte der Personalplanung wieder. An dieser Stelle sollte deutlich werden, dass das AHEAD-System Aktivitäten, Produkte und Ressourcen integriert betrachtet, und damit dynamische Prozesse auf ganzheitlicher Ebene beschreibt. Das AHEAD-System folgt somit näher dem Life-Cycle eines Prozesses als Systeme, die nur Teilaspekte eines Prozesses unterstützen.

Im Vergleich mit anderen Systemen fällt weiterhin auf, dass das AHEAD-System an unterschiedliche Anwendungsbereiche angepasst werden kann. Die graphischen Werkzeuge, die mit dem AHEAD-System generiert werden, sind nicht branchenspezifisch, sondern allgemein gehalten. Beispielsweise können somit sowohl dynamische Prozesse in der Softwareentwicklung als auch dynamische Prozesse in der Verfahrenstechnik beschrieben werden. Da sich Domänenexperten der UML bedienen können, entfällt auch die Einarbeitung in eine neue Spezifikationssprache.

Das AHEAD-System repräsentiert Daten intern als Graphen und Operationen auf diesen Graphen als Graphersetzungsregeln. Dem AHEAD-System liegt damit eine formale Spezifikation zu Grunde, was das Verständnis über die generierten Werkzeuge vereinfacht und den Aufwand zur Generierung der Werkzeuge reduziert. Viele vergleichbare Systeme basieren dagegen nicht auf einer formalen Spezifikation.

Nachfolgend werden die Schwachstellen des AHEAD-Systems näher erläutert:

Nachteilig im Vergleich zu anderen Systemen wirkt sich beim AHEAD-System aus, dass das AHEAD-System noch nicht ausreichend evaluiert wurde. Das AHEAD-System wurde bisher nur im Rahmen eines Referenzszenarios in der Verfahrenstechnik evaluiert. Zwar erwies sich das integrierte Management der dynamischen Prozesse innerhalb des Pilotprojektes als vorteilhaft. Umfangreiche Aussagen über die qualitative und quantitative Eignung des AHEAD-Systems für die Praxis lassen sich dadurch jedoch noch nicht ableiten. Nachteilig könnte sich in diesem Zusammenhang bei zukünftigen Evaluationen auch die vorausgesetzte flache Hierarchie bei den Projektbeteiligten auswirken. Das AHEAD-System unterscheidet zwischen Projektmanager und Entwickler. In kleinen Projekten mag die Einteilung in Projektleiter und Projektteam noch vorteilhaft erscheinen. Aber gerade bei größeren und eventuell noch internationalen Projekten wird sich diese festgelegte Einteilung nicht als vorteilhaft erweisen, da hier beispielsweise noch Bereichsleiter und Abteilungsleiter aufgenommen werden müssten und vielleicht auch die Notwendigkeit bestünde, das Projekt auf mehrere Unternehmen zu verteilen. In solch einem Umfeld könnte auch das AHEAD-System das Projekt nicht mehr ganzheitlich erfassen. Lösungsansätze zur Erweiterung auf ein größeres Projektbeteiligtenumfeld und zur verteilten Anwendung des AHEAD-Systems sind aufgrund der Konzeption des AHEAD-Systems auch in Zukunft nur bedingt realisierbar.

Um bei Unternehmen größere Akzeptanz zu finden, fehlt es dem AHEAD-System bislang noch an Möglichkeiten, andere Systeme zu integrieren. Der Arbeitskontext der Entwicklerumgebung erlaubt es nur Entwicklungswerkzeuge auszuführen. Viele Unternehmen verwenden aber zum Beispiel bereits Systeme wie CVS zum Produktmanagement ihrer Projekte und sind im Allgemeinen nicht bereit, diese zu Gunsten eines neuen Systems völlig abzuschaffen. Daher wäre es nötig, dem AHEAD-System weitere Schnittstellen zu bereits weit verbreiteten kommerziellen Systemen zur Verfügung zu stellen. Lösungsansätze zur besseren Integration des AHEAD-Systems sind Gegenstand laufender Untersuchungen der Entwicklergruppe des AHEAD-Systems.

Bei der Anwendung des AHEAD-Systems erweisen sich die in der Modellierungsumgebung zu erstellenden Modelle als problematisch. Zur Anwendung des AHEAD-Systems ist es notwendig, für alle projektspezifischen Managementdaten Modelle in der UML zu modellieren. Bei größeren Projekten werden die zu erstellenden Modelle sehr komplex und unübersichtlich. Aus diesem Grund ist validiertes Domänenwissen und umfangreiche Modellierungserfahrung erforderlich, das nicht von jedem vorausgesetzt werden kann. Für größere Projekte scheint das AHEAD-System auch in diesem Punkt noch ungeeignet zu sein, da der Modellierungsaufwand zu hoch ist. Die Entwicklung und Bereitstellung von Referenzmodellen könnte dieser Problematik Rechnung tragen.

4 Zusammenfassung

Mit dem AHEAD-System wurde ein Managementsystem für dynamische Prozesse vorgestellt, das Produkte, Aktivitäten und Ressourcen eines Entwicklungsprozesses integriert betrachtet. Die Stärke des Systems liegt dabei in der ganzheitlichen Sichtweise auf die Prozesse. Allerdings ist das AHEAD-System für größere Projekte für die Praxis derzeit nur bedingt anwendbar. Die Ansätze und Konzepte der integrierten Sichtweise auf Prozesse gehen jedoch deutlich über die Funktionalität kommerzieller Managementsysteme hinaus und zeichnen damit das AHEAD-System besonders aus.

Literatur

[1] Westfechtel, Bernhard: Ein graphbasiertes Managementsystem für dynamische Entwicklungsprozesse, Informatik – Forschung und Entwicklung (2001), 16

[2] Andries, Gregor: Graph Transformation for Specification and Programming, Science of Computer Programming, Vol.34, Elsevier, 1999

[3] Zeller, Andreas: Programmierwerkzeuge, dpunkt verlag, 2000

[4] Fowler, Martin: UML konzentriert, Addison Wesley, 2000