Teilprojekt D1

Anwendungssystem Gebäude

Leiter

Prof. Dr. Gerhard Zimmermann	Dr. Bernd Schürmann
geb. 18.5.1939	geb. 22.8.1959
Universität Kaiserslautern	Universität Kaiserslautern
Postfach 3049	Postfach 3049
67653 Kaiserslautern	67653 Kaiserslautern
Tel.: 0631/205-2628	Tel.: 0631/205-3069
Fax: 0631/205-2162	Fax: 0631/205-2162
zimmerma@informatik.uni-kl.de	schuerma@informatik.uni-kl.de

Mitarbeiter

Ergänzungsausstattung:

Dipl. Inform. S. Queins
Dipl. Inform. J. P. Riegel (seit 1.8.1999)
Dipl. Inform. M. Schütze (bis 30.4.1999)

Grundausstattung:

Dipl. Inform. J. P. Riegel (bis 1.8.1999)
Dipl. Inform. A. Metzger (seit 1.12.1998)
Dipl. Inform. T. Tetteroo (seit 1.8.1999)

1 Kurzfassung der Aufgabenstellung

Primär dient dieses Teilprojekt als Infrastrukturprojekt für die B-Projekte. Es spielt die Rollen des Auftraggebers und, zusammen mit D2, des Experten im Anwendungsgebiet „Gebäudeautomation“. Desweiteren stellt D1 dem SFB Testumgebungen zur Verfügung. Daraus ergeben sich vier Arbeitsschwerpunkte:

Die Rolle des Experten wird durch die **Anwendungsfeldanalyse** erarbeitet. Dazu ist Wissen über das gesteuerte System, also die Physik der Gebäude und die Funktion der Installationen, und Wissen über Steuerungen notwendig. Unser Ziel ist es, das Wissen so zu strukturieren und als „Lexikon“ abzulegen, dass es automatisch oder interaktiv in allen Phasen des Software-Entwicklungsprozesses eingesetzt werden kann, ohne auf Anwendungsexperten zurückgreifen zu müssen.

In der Rolle des Auftraggebers werden **Szenarien und Problembeschreibungen** erstellt und gesammelt, die den Anwendungsbereich möglichst weit abdecken und vor allem auch nichtfunktionale Anforderungen enthalten.

Für **Prototyping und Test** von Steuerungssystemen werden **Gebäudesimulatoren** je nach Problembeschreibung in vielen Varianten benötigt. Deshalb wird ein System von Generatoren entwickelt, das für die statischen Komponenten die bereits vorhandenen Generatoren und für die dynamischen Komponenten generative Entwurfsmuster (ähnlich Design-Patterns) und neue Generatoren verwendet, um aus Problembeschreibung und Lexikon automatisch die richtige Variante zu erzeugen.

Simulatoren abstrahieren von der Realität. Ein endgültiger Test der Steuerungssysteme ist deshalb nur an einem realen Gebäude möglich. Außerdem müssen die Simulatoren getestet werden. Deshalb werden Teile von Gebäuden als **Testfeld** ausgebaut, das mit Sensoren, Aktuatoren, Prozessoren und Kommunikationstechnik ausgestattet wird.

Als zusätzliche Aufgabe wurde das ursprünglich in B9 beantragte **Requirements-Engineering** mit übernommen, da diese Phase im Entwicklungsprozess gerade bei komplexen eingebetteten Systemen besonders wichtig ist und effizient bearbeitet werden muss.

2 Entwicklung des Teilprojekts in der Förderperiode

Bei der Bewilligung des Teilprojektes D1 für den Berichtszeitraum wurde zur Auflage gemacht, Teile des nicht bewilligten Teilprojektes B9 mit zu übernehmen. Daraus ergab sich außer den in D1 geplanten Arbeitsschwerpunkten A bis D ein weiterer, den wir hier Arbeitsschwerpunkt E: Requirements-Engineering nennen. Zu Beginn der Periode gab es im SFB keine Gruppe, die sich mit dieser wichtigen Phase des Entwicklungsprozesses beschäftigt hat. Für Experimente mit späteren Phasen war es aber unbedingt notwendig, hier entsprechende domänenspezifische Lösungen zu finden und in Fallstudien Beispiele von System-Requirements bereitzustellen.

Dieser zusätzliche Arbeitsschwerpunkt hat sich als sehr fruchtbar herausgestellt. Allerdings mussten die Arbeiten in den Arbeitsschwerpunkten A und B entsprechend reduziert werden, was durchaus gerechtfertigt erscheint. Die Arbeitsschwerpunkte C und D wurden wie geplant durchgeführt.

Personell gab es Änderungen durch die Promotion von Herrn Schütze, auf dessen Stelle Herr Riegel aus der Grundausstattung nachrückte. In der Grundausstattung waren nicht immer beide Stellen besetzt, bedingt durch den allgemeinen Mangel an geeigneten Kandidaten in den schwachen Jahrgängen.

3 Durchgeführte Arbeiten und Ergebnisse

3.1 Arbeitsschwerpunkt A: Anwendungsfeldanalyse

Ein *Global-Dictionary* genanntes Hypertext-Dokument wurde kontinuierlich mit Wissen aus der Anwendungsdomäne gefüllt. Es besteht zur Zeit aus etwa 300 Objekten, die in 6 Kategorien geordnet und zusätzlich über Generalisierungs-, Aggregations-, Material- und Topologie-Relationen verknüpft sind. Navigiert wird über Hyperlinks. Auf diese Weise sind die professionellen englischen Bezeichnungen mit Erläuterungen, physikalischen Zusammenhängen und Standardwerten leicht aufzufinden, wenn ein Schlagwort in der logischen Nähe des Objektes bekannt ist. Da das Global-Dictionary Wissen über die gesteuerte Strecke mit Wissen über Steuerung und Automation verknüpft, ist seine Struktur ideal geeignet, um Artefakte für die Wiederverwendung abzulegen und zu finden. Benutzt wurde das Global Dictionary vor allem zur Erstellung von Auszügen für spezielle Projekte, den *Project-Dictionaries.*

3.2 Arbeitsschwerpunkt B: Szenarien und Problembeschreibungen

In einer umfangreichen Studie wurde als Szenario eine existierende Heizungs-, Klima- und Lichtsteuerung in einem Großraumbüro (Intelligent Workplace, CMU) mit individueller Arbeitsplatzsteuerung analysiert und modelliert. Die untersuchten Steuerungen waren das Metasys-System von Johnson Controls für Klima und Heizung und EIB von Siemens für Licht. Die Klimatisierung der einzelnen Arbeitsplätze wurde durch Personal Environment Module von Johnson Controls ermöglicht. Zur Beeinflussung des natürlichen Lichteinfalls wurden drehbare Spiegel und Jalousetten gesteuert. Speziell zur Modellierung wurden eine Reihe von Ansätzen erprobt, die schließlich zu der im Arbeitsschwerpunkt E beschriebenen Analysemethode auf der Basis einer angepassten Modellarchitektur geführt haben [13], [21].

Da die analysierte Steuerung für den futuristischen Intelligent Workplace (IW) sehr konservativ ist, wurde zusammen mit Prof. Mahdavi (CMU) als gemeinsames Projekt eine Dissertation an der CMU initiiert, die den technischen Möglichkeiten des IW entsprechende Automationsszenarien entwickeln soll.

Als Infrastrukturprojekt wurden Problembeschreibungen für Fallstudien erstellt und es wurde in der Rolle des Kunden und des Domänenexperten (zusammen mit D2) Hilfestellung gegeben. Die Problemstellung für eine Lichtsteuerung wurde ein Art Benchmark für einen internationalen Vergleich verschiedener Methoden zum Requirements-Engineering (siehe auch [28]). Eine Erweiterung dieser Problembeschreibung zur Temperatursteuerung wurde Grundlage der im Querschnittsprojekt Q3 durchgeführten Domänenanalyse [19].

3.3 Arbeitsschwerpunkt C: Gebäudesimulation

Ziel dieses Arbeitsschwerpunktes ist es, eine Umgebung bereitzustellen mit der es einfach und schnell möglich ist, angepasste Gebäudesimulatoren zu erzeugen. Als Modellierungsnotation werden dazu neben Klassendiagrammen vor allem domänenspezifische Entwurfsmuster (Pattern) verwendet. Bereits in der ersten Förderungsperiode entstanden die Grundideen und erste Werkzeuge. Die folgende Aufzählung zeigt die wesentlichen Neuerungen dieser Periode:

- Implementierung weiterer Werkzeuge.
- Strukturierung des Pattern-Kataloges. Neu hinzugekommen sind vor allem die Abschnitte Software-Architektur und Kontrollfluss.
- Typisierung der Pattern-Schnittstellen und Aufstellung von Regeln zur Anwendung der Entwurfsmuster.
- Erprobung der Entwurfsmethode und Überprüfung der Simulatoren.

Im Rahmen von AP-C1 wurden neue Entwurfsmuster entwickelt und Regeln und Typen für bestehende Pattern eingeführt. AP-C2 beschäftigte sich vor allem mit der Integration verschiedener Komponenten und Modelle. Die Erprobung der Entwurfsmethode für Gebäudesimulatoren fand im Rahmen von AP-C3 statt.

Arbeitspaket AP-C1: Modelle

Der aktuelle Pattern-Katalog [29] besteht aus 41 Pattern aus den Bereichen Software-Architektur, Kontrollfluss, Benutzungsschnittstelle und Gebäudesimulation. Etwa die Hälfte der Entwurfsmuster wurde während dieser Förderungsperiode entworfen. Diese Muster können zur Modellierung von Software-Komponenten verwendet werden. Dazu werden sie instanziiert und an das zu entwerfende Komponentenmodell formal angebunden [26]. So dienen die Pattern zum einen zur Festlegung der Funktionalität eines oder mehrerer Objekte und sie können darüber hinaus als Bindeglied zwischen verschiedenen Modellen und Notationen eingesetzt werden (siehe auch AP-C2). Generatoren überführen die Modelle in fertig ablauffähigen Programmcode [24], [25]. Die entwurfsmusterbasierte Modellierung von Komponenten in der Domäne Gebäudesimulation wurde von Martin Schütze in seiner Dissertation [12] zusammengefasst und erläutert.

Um den Umgang mit den Pattern zu vereinfachen und den Entwurfsspielraum sinnvoll einzuschränken, wurde eine Typisierung der Pattern-Schnittstelle eingeführt [6]. Dadurch wird die Anbindung der Pattern einfacher, da unsinnige Bindungspunkte und falsche Pattern-Kombinationen von vorn herein verboten sind. Neben der Typisierung der anbindbaren Methoden und Attribute können mögliche Bindungspunkte auch durch Regeln weiter eingeschränkt werden. Damit ist es möglich, bestimmte Objektstrukturen zu fordern, um ein Pattern anbinden zu können. Innerhalb dieser Förderungsperiode entstand ein Editor zur Anwendung der Pattern [1]. Dieser Editor wurde zudem um das Konzept der Typen und Regeln erweitert, sodass nur gültige Pattern-Bindungen erstellt werden können [6].

Entwurfsmuster können weiterhin zur Modellierung der Architektur einer Software-Komponente benutzt werden. Einige Architekturstile (z.B. Pipes&Filters, Model-View-Controller) können durch Pattern ausgedrückt werden und so explizit in die Modellierung einfließen [20]. Somit wird eine schrittweise Verfeinerung der Komponentenmodelle unterstützt, indem zunächst Architekturmerkmale modelliert werden und diese dann (mit demselben Beschreibungsmittel) später konkretisiert werden.

Arbeitspaket AP-C2: Synthese

Ein wichtiger Aspekt in der Modellierung von Gebäudesimulatoren ist die klare Definition und Verwendung von Schnittstellen zwischen einzelnen Teilen. Dieses sind sowohl Schnittstellen innerhalb eines Modells (z.B. Verteilung, Kopplung unterschiedlicher Effekte), als auch externe Schnittstellen (Kommunikation mit der Hardware, Anbindung weiterer Modellnotationen). Zur Schnittstellendefinition zwischen Objekten dienen Pattern der Kategorie Software-Architektur. Diese helfen zum einen, unabhängig von der Objektinstanziierung zu werden (Patterns Traversal und Follow-Relation [29]) und zum anderen, Datenkanäle und Datenfluss zu modellieren (Pattern Pipes&Filters [7]). Die Pipes&Filters Pattern bieten darüberhinaus auch eine einfache Möglichkeit zur Verteilung eines Simulators auf mehrere Rechner. Die dabei auftretende Kommunikation geschieht mit dem in [30] beschriebenen socket-basierten Mechanismus.

Im Rahmen einer Diplomarbeit [8] wurde die Integration Endlicher Automaten in Klassendiagramme mit Pattern untersucht. Dabei ging es darum, Kontrollfluss und Funktionalität einzelner Objekttypen mit Endlichen Automaten (z.B. in SDL) zu beschreiben und diese dann an ein Klassendiagramm zu binden. Die dabei entstandenen Pattern können benutzt werden, um verschiedene Abarbeitungsstrategien der Endlichen Automaten auszuwählen.

Gegen Ende dieser Antragsperiode wurden Untersuchungen angefangen, welche Auswirkungen die Kopplung eines Simulators mit realer Hardware (Testfeld) hat. Damit bei gemischter HW/SW-Simulation möglichst wenige gegenseitige Beeinflussungen auftreten, ist eine gute Partitionierung zwischen Simulator und Gebäudeinstallation nötig. Diese hängt im wesentlichen vom betrachteten Szenario und den zu messenden Größen ab.

Arbeitspaket AP-C3: Erprobung

In der Antragsperiode wurden zahlreiche Fallstudien durchgeführt, um die beschriebene entwurfsmusterbasierte Modellierungsmethode zu verbessern und um die mathematische Genauigkeit der Simulatoren zu überprüfen. Die Modellierungsexperimente dienten in erster Linie zur Verbesserung des Pattern-Kataloges, indem neue Pattern definiert wurden und bestehende Pattern verfeinert und korrigiert wurden. Die Parameter der einzelnen Simulationsobjekte wurden durch Vergleichsmessungen mit dem realen Testraum abgestimmt. Zum Vergleich des entwurfsmusterbasierten Ansatzes mit der SDL Modellierungsmethode wurde ein Simulator zur Wärmesimulation in SDL modelliert (siehe auch 3.5). Diese Arbeiten sind noch nicht abgeschlossen, zeigen aber bereits, dass der von SDT automatisch generierte C-Code die thermische Simulation von Gebäuden mit etwa 1000 Räumen in Realzeit zulässt.

In weiteren Untersuchungen wurde festgestellt, dass die Geschwindigkeit der generierten Simulatoren im Wesentlichen von der Zahl der vorkommenden Prozesswechsel abhängt. Die Zeit zur eigentlichen Berechnung der Simulationsformeln ist relativ klein gegenüber der Event-Verarbeitung und den damit verbundenen Prozesswechseln. Ein Optimierungsansatz ist es also, die Prozesswechsel beim Simulator möglichst gering zu halten. Dieses kann durch Delegation der Berechnungen an aggregierte Objekte geschehen. Dabei dürfen jedoch keine Zeitgrenzen verletzt werden, die die Simulationsergebnisse (durch zu seltene Berechnungen) falsch werden lassen. Die Anzahl der in Realzeit simulierbaren Räume (mit 6 Prozessen pro Raum) liegt bei ca. 200. Wird das gesamte Haus nur als ein Prozess gerechnet, lassen sich ca. 1000 Räume in Realzeit simulieren.

3.4 Arbeitsschwerpunkt D: Testfeld

Das Testfeld wurde um drei weitere Räume und zwei Flursegmente erweitert, womit ein größerer Bereich von Szenarien zur Validierung von Steuerungssystem-Prototypen und Simulatoren abgedeckt wird (siehe AP-D1). Bei der Installation zusätzlicher Komponenten wurde die bisher eingesetzte sternförmige Verdrahtung aufgegeben und durch eine Kommunikationshierarchie ersetzt. Dabei sind die einzelnen Ebenen in dieser Hierarchie über unterschiedlich leistungsfähige und skalierbare Kontroller (Prozessoren) gekoppelt (siehe AP-D2). Dieses Testfeld mit ca. 115 Sensoren und 55 Aktuatoren wurde zur Durchführung von Experimenten genutzt, bei denen D1 Hilfestellung leistete (siehe AP-D3).

Arbeitspaket AP-D1: Sensoren und Aktuatoren

Es hat sich herausgestellt, dass zur Beurteilung unterschiedlicher Steuerungstrategien für Gebäudeautomationssysteme Experimente parallel zueinander durchgeführt werden müssen. Externe Einflüsse wie z.B. das Wetter sind nicht beeinflussbar und deshalb sind Vergleichsexperimente, die zu unterschiedlichen Zeitpunkten durchgeführt werden, schwieriger zu beurteilen. Daher wurde die initiale Installation [13], die aus einem Computerlabor bestand, das von Studenten genutzt wurde, zunächst um einen weiteren solchen Raum ergänzt. Anschließend wurden zwei Mitarbeiterbüros, die sich nicht im Aufbau aber in der Nutzung von den Laboren unterscheiden, zum Testfeld hinzugenommen.

In jedem der Räume stehen Helligkeits- und Temperatursensoren, Bewegungsmelder und Fenster- und Türkontakte für Messungen und Eingaben in das Steuerungssystem zur Verfügung. Die Heizkörperventile können durch elektrische Stellantriebe gesteuert werden. Im Notlaufbetrieb sorgt ein lokaler PD-Regler für deren Ansteuerung. Die Raumbeleuchtung wurde so modifiziert, dass sie sich elektronisch schalten und dimmen lässt. Das Schalten der Lampen bleibt durch den Einsatz von Stromstoßrelais unabhängig vom Automationssystem. Zur Abdunklung der Räume wurden diese mit Innenjalousien ausgestattet, deren Lamellen sich motorisch bewegen lassen. Für einen typischen Raum kann man so auf 22 Sensoren und 12 Aktuatoren zugreifen. [30]

In den zwei Flursegmenten wurden 18 Bewegungsmelder montiert, welche Bewegungen im Flur und vor den Zugangstüren erfassen. Die Beleuchtung der Flursegmente kann über Pulsgeneratoren geschaltet werden. An den Außenwänden des Gebäudes wurden Helligkeitssensoren angebracht, mit deren Daten sich die in einen einzelnen Raum potenziell einfallende Helligkeit ermitteln lässt. Diesen wurden speziell auch für die Ausführung der in Querschnittsprojekt Q1 entstandenen Prototypen eingesetzt.

Der Einsatz von kostengünstigen Dallas 1-Wire-Sensoren wurde zu Gunsten einer Eigenentwicklung [2] aufgegeben, da wegen des aufwändigen Busprotokolls eine Ankopplung der Dallas-Bausteine über ein FPGA [3] nötig war.

Arbeitspaket AP-D2: Kontroller und Kommunikation

Es wurde die in Abbildung 1 gezeigte Kommunikationshierarchie realisiert. Auf unterster Ebene werden Sensoren und Aktuatoren über einfache Messdatenverbindungen an Mikrokontroller (μC) angekoppelt. Diese sind in einer Area (Raum oder Flursegment) untereinander und mit einem wesentlich leistungsfähigerem Embedded-PC über einen CAN-Bus vernetzt, um eine echtzeitfähige Kommunikation zu gewährleisten. Die Embedded-PCs aller Areas kommunizieren über Ethernet und besitzen eine Verbindung zu einem wiederum leistungsfähigeren Standard-PC. Das gesamte System lässt sich mit dem Uni-LAN und dem Internet verbinden, um auch von extern auf das Testfeld zugreifen zu können.

Für die Anbindung der Sensorik und Aktuatorik an die 8051-kompatiblen Mikrokontroller wurde ein modulares Design realisiert. Dabei werden die vom Mikrokontroller verfügbaren I/O-Pins so gruppiert, dass vorgefertigte I/O-Module angeschlossen werden können. Die Konfiguration erfolgt über den CAN-Bus ohne Hardwareeingriff und erlaubt so ein hohes Maß an Flexibilität für die eventuell in Experimenten notwendigen Umbauarbeiten und die daraus folgenden Änderungen in der Konfiguration. Neben der Umwandlung der Messwerte in physikalische Größen und der Kommunikation dieser Daten über den CAN-Bus laufen auf den Mikrokontrollern rudimentäre Steuerungsalgorithmen, die einen Minimalbetrieb der Installation gewährleisten [2]. Für Experimente während der Design-Phase bieten die Kontroller noch genügend Ressourcen für kleinere Komponenten des Steuerungssystems. Der Einsatz von SLIOs (Serial Linked I/O, [5]) zur Anbindung von Sensoren und Aktuatoren über CAN wurde wegen des eingeschränkten Adressbereichs nicht weitergeführt.

Die Installation wurde so verwirklicht, dass die Kopplung der einzelnen Areas auf drei unterschiedliche Arten erfolgen kann:

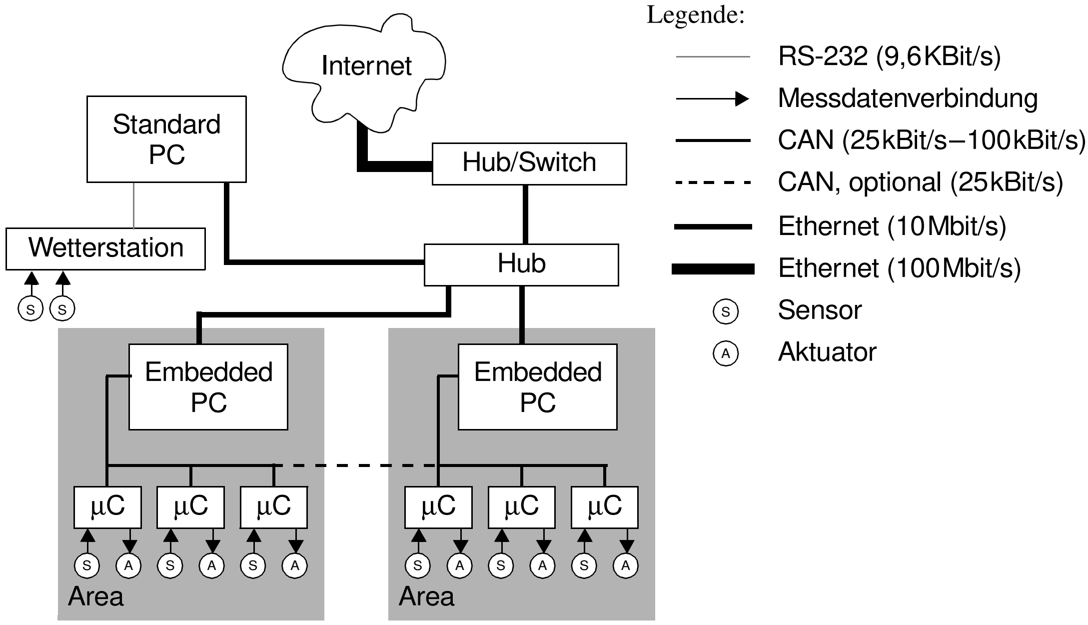

Abbildung 1: Kommunikationshierarchie

- Die einzelnen CAN-Bussegmente können zu einem großen Segment zusammengeschaltet werden, was die direkte Kommunikation aller Komponenten am Bus ermöglicht (in Abb. 1 durch die gestrichelte Verbindung zwischen den µC angedeutet).
- Die Embedded-PCs können eingesetzt werden, um CAN $\leftrightarrow$ Ethernet-Router zu realisieren. Um das Routing und eine verbindungsorientierte Kommunikation der Komponenten zu ermöglichen, soll noch in dieser Förderungsperiode ein CAN-Protokoll aus Teilprojekt B4 eingesetzt werden. Dazu wurden Vorarbeiten für ein echtzeitfähiges Kommunikationssubsystem [10], [27] und einen CAN $\leftrightarrow$ CAN-Router [11] geleistet.
- Der auf den Embedded-PCs vorhandene TCP/IP-Stack kann verwendet werden, um über eine Socket-Schnittstelle das für den Datenkonzentrator und den Simulator genutzte Protokoll [30] zur Verfügung zu stellen.

Auf den Embedded-PCs können in der Analyse oder Design entstandene Prototypen ausgeführt werden. In diesen Prototypen werden typischerweise komplexere Steuerungsaufgaben realisiert. Eine Kopplung zwischen einzelnen Prototyp-Komponenten auf unterschiedlichen Rechnern erfolgt über TCP/IP-Socket-Verbindungen.

Der Standard-PC wird zur Aufzeichnung von Messdaten und zum Anschluss der Wetterstation genutzt. Für die Administrierung des Systems existiert ein Java-Applet, das die Konfigurationsdaten sämtlicher Sensoren und Aktuatoren verwaltet.

Arbeitspaket AP-D3: Experimente

Zur Validierung des in Querschnittsprojekt Q1 spezifizierten Gebäudeautomationssystems wurde dieses um die nicht ausgestatten Räume reduziert und im Testfeld als Prototyp ausgeführt. Der Schritt von der Entwicklungsplattform (Sun-, bzw. HP-Workstation) zur Zielplattform (Embedded-PC) gestaltete sich im ersten Schritt sehr einfach. Es musste lediglich der erzeugte C-Code neu übersetzt und einige handgeschriebene C-Funktionen angepasst werden. Die Verteilung des Prototyps auf die Embedded-PCs war dagegen aufwändiger. Zunächst mussten sinnvolle Partitionen festgelegt und dann die Teilsysteme neu generiert werden, wobei für die Kopplung über Sockets, bzw. CAN, eine Kodierung und Dekodierung von Signalen erfolgen musste. Desweiteren wurde für die Socket-Kommunikation eine zentrale Komponente entwickelt, die alle Signale entgegennimmt und an das entsprechende Teilsystem weiterreicht. Es hat sich gezeigt, dass die einzelnen Schritte sehr mühsam sind und für effizientes Prototyping unbedingt durch Werkzeuge unterstützt werden müssen.

Als weiteres Experiment wurde ein Praktikum durchgeführt, in welchem Studenten ein Steuerungssystem modellierten. Nachdem das System mit Hilfe einer Simulation des Gebäudes getestet worden war, wurde ein realer Raum des Testfeldes zur Validierung genutzt.

3.5 Arbeitsschwerpunkt E: Requirements-Engineering

Wir haben an der Entwicklung einer Methode gearbeitet, die es erlaubt, aus einer gegebenen, informellen Problembeschreibung formale Anforderungen herzuleiten. Ziel des Ansatzes ist es, schnell ein ausführbares Modell zu erhalten, das dann, als Prototyp, sowohl zur Absprache mit dem Kunden, als auch zur Überprüfung der durchgeführten Entwicklungsschritte dient. Es dient weiterhin als Anforderungsdokument für die nachfolgenden Entwicklungsschritte. Bei der Entwicklung der Methode haben wir Wert auf die Effizienz der Entwicklung und eine verständliche und vollständige Formulierung der Anforderungen gelegt.

Die speziellen Eigenschaften der Methode sind in der folgenden Aufzählung genannt und werden im Weiteren detaillierter dargestellt. Die dabei referenzierten Fallstudien werden am Ende dieses Abschnitts kurz vorgestellt. Eine vollständige Beschreibung der Methode findet sich in [14], Kapitel 4.3.

- domänenspezifische Modellarchitektur
- spezielle informelle Beschreibung
- schrittweise Überführung und Verfeinerung in eine formale Beschreibung mit Inspektion und Prototyping
- Wiederverwendung von Domänenwissen und Entwicklungsprodukten
- Definition einer Metrik zur Bewertung der Wiederverwendung
- Unterstützung des Prototypings
- Analyse der Modellierung und Betrachtung der nichtfunktionalen Eigenschaften (speziell die Zeitanforderungen)
- Formalisierung des Ansatzes durch eine Entwurfssystematik

Die Entwicklungsschritte und -produkte sind in Abbildung 2 zusammengefasst.

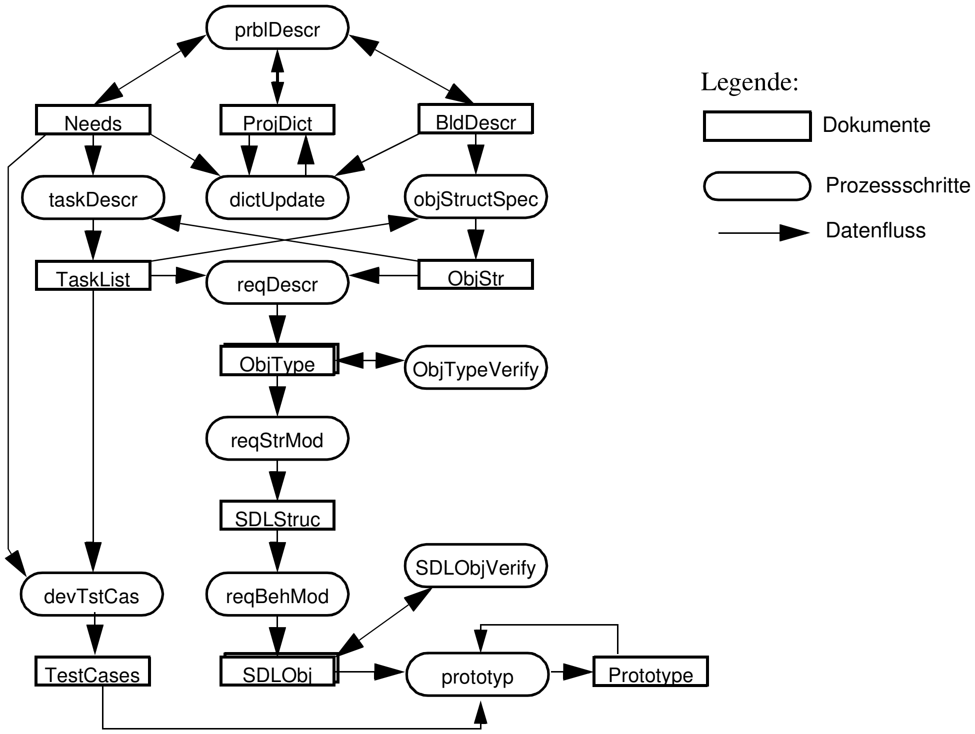

Abbildung 2: Prozessmodell

Zentraler Bestandteil ist die frühe Ermittlung einer Systemstruktur. Dies wird durch die Spezialisierung auf die Domäne Gebäudautomation ermöglicht, da sich eine sinnvolle Struktur für ein Steuerungssystem leicht aus der Struktur des Gebäudes ableiten lässt. Die Systemstruktur wird durch Aggregationsbeziehungen beschrieben, bei denen die einzelnen aggregierenden Typen nicht nur strukturierenden Charakter haben. Vielmehr werden ihnen, ähnlich einer Organisationshierarchie, Aufgaben (Tasks) zugewiesen [15]. Die Ermittlung der Tasks geschieht aus den in der Problembeschreibung gegebenen Anforderungen (Needs). Dabei kann ein Need durch mehrere Tasks bearbeitet werden, bzw. kann ein Task mehrere Needs realisieren. Die Tasks werden den einzelnen Objekttypen aus der Systemstruktur zugewiesen, woraus die Notwendigkeit der Zerlegung der Needs in einzelne Tasks folgt. Die Tasks werden informell beschrieben. Sie definieren, was ein Objekttyp im einzelnen zu leisten hat. Wie die einzelnen Tasks gelöst werden und welche Kommunikation sie dazu benötigen, wird als Strategie informell modelliert. Daraus ergeben sich auch die Schnittstellen zwischen den Objekttypen. Anschließend werden dann die Objekttypen in eine formale Beschreibung überführt. Zur Zeit haben wir als formale Beschreibungstechnik SDL gewählt, wobei unsere Methode nicht unbedingt auf diese Technik eingeschränkt ist.

Für die einzelnen Entwicklungsschritte sind Überprüfungen vorgesehen, die entweder durch eine Inspektion (speziell der informellen Dokumente) und durch Prototyping durchgeführt werden. Für die Inspektion wurden typabhängige Checklisten erstellt. Diese Checklisten wurden im Laufe der Entwicklung der Methode verfeinert bzw. erweitert. Als Beispiel sei die Fallstudie CS3/1 erwähnt ([16], Kapitel 5). Hier wurden die gefundenen Defekte in verschiedene Klassen eingeteilt. Die Konsequenz aus dieser Auswertung war die Verbesserung der Checklisten, um die häufig gemachten Fehler früher zu erkennen. Mit dieser Änderung und der Einführung eines Project-Dictionary konnten in der nächsten Fallstudie z.B. die Fehler bzgl. inkonsistenter Signaldefinition um ca. 70% reduziert werden.

In den einzelnen Fallstudien wurden neben der objekttypspezifischen Erfassung des Modellierungsaufwands die Anzahl und der Typ der Fehler erfasst. Daraus lassen sich dann die benötigten Aussagen bzgl. Fehlerausbreitung, Wiederverwendung und bei erneuten Projektdurchläufen bzgl Verbesserungen in der Methode ableiten.

Von der Wiederverwendung (Reuse) wird bei allen Entwicklungsschritten Gebrauch gemacht. Dazu werden die in früheren Projekten entwickelten Produkte in eine Reuse-Bibliothek eingebracht und so dem aktuellen Projekt zur Verfügung gestellt. Um den Nutzen einer speziellen Wiederverwendung beurteilen zu können, muss der Aufwand zur Neuerstellung eines Artefakts mit dem Aufwand zur Anpassung eines passenden Reuse-Artefakts verglichen werden. Um diesen abschätzen zu können, wird auf die Erfahrung aus früheren Wiederverwendungen zurückgegriffen. Desweiteren können wir am Ende eines Projekts Aussagen über den Nutzen von Wiederverwendung bezogen auf das gesamte Projekt machen. Dazu haben wir die in [32] vorgestellte Metrik auf unsere Modellierungsmethodik angepasst (siehe auch [14], Kapitel 7.2). Für die Fallstudie CS3/2 ergab sich eine Produktivitätssteigerung von 1,22 durch konsequente Wiederverwendung.

Ein zentrales Produkt unserer Methode ist das Project-Dictionary, in dem in höchstem Maße von Wiederverwendung Gebrauch gemacht wird. Es dient zur Kommunikation der Entwickler untereinander. In ihm werden Begriffe und objektübergreifende Informationen abgelegt. Initial wird es mit Daten aus dem projektübergreifenden Lexikon zur Verfügung gestellt und im Laufe der Entwicklung um die projektspezifischen Einträge erweitert. In der Fallstudie CS3/2 enthielt das Project-Dictionary am Ende des Projekts ca. 280 Einträge.

Die Erstellung eines Prototyps wird stark von den in der Entwicklungsumgebung SDT von Telelogic integrierten Generatoren unterstützt. Diese überführen ein SDL-Modell in ablauffähigen C-Code. Wie Abbildung 3 zeigt, stellen wir zur Anbindung der Prototypen an die Testumgebung (bei uns i. Allg. ein reales Gebäude oder eine Simulation des Gebäudes) die generische Komponente ProtoCtrl zur Verfügung. Diese ermöglicht die Verbindung entweder über eine TCP/IP-Socket-Schnittstelle oder, zur Kommunikation mit bereits installierten Hardware-Komponenten, über eine CAN-Schnittstelle. Der Transport der physikalischen Werte von bzw. zu den Sensor- und Aktuatorkomponenten innerhalb des Steuerungssystems erfolgt durch SDL Remote-Procedure-Calls. Die Kommunikation zur Umgebung wird durch die Java Komponente ControlSI [9] bzw. durch deren Erweiterung Interlink [31] ermöglicht.

Neben der Ermittlung der funktionalen Anforderungen wird Wert auf die nichtfunktionalen Anforderungen gelegt. In Bezug auf Zuverlässigkeit werden die Anforderungen betrachtet, die bereits vom Kunden definiert wurden. Diese lassen sich dann auch

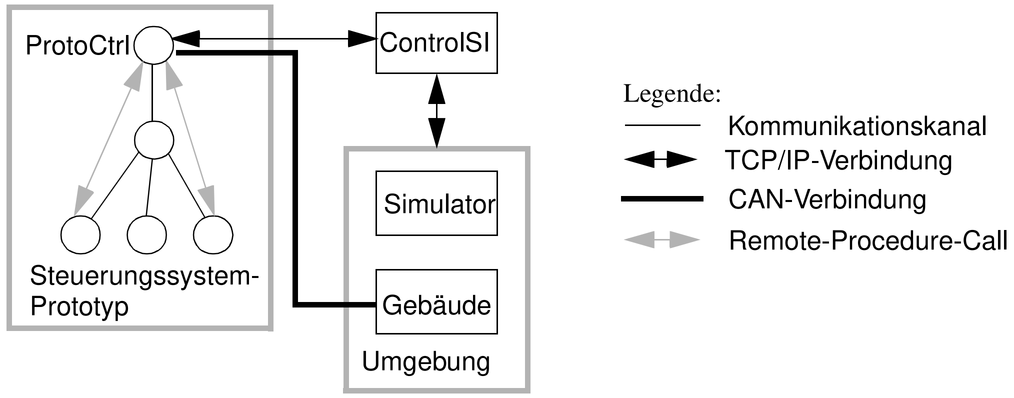

Abbildung 3: Kopplung eines Prototyp an die Umgebung

mit Hilfe des Prototypings überprüfen. Dies gilt nicht für die Zeitanforderungen, da bei unserem Ansatz für das Prototyping in dieser frühen Phase gerade bezüglich der Kommunikation sehr starke Vereinfachungen gemacht werden. Deswegen werden die Zeitanforderungen im SDL-Modell nicht explizit modelliert, sie werden vielmehr aus den SDL-Modellen hergeleitet. Dies betrifft die Zeitspanne, die nach Auslösen eines Event (z.B. die Aktion eines Benutzers) benötigt wird, bis eine Reaktion an einem Aktuator zu beobachten ist. Ausgehend von dem Event wird eine Kette von Signalen bzw. aufgerufenen Prozessen gebildet, die ausgehend von diesem Event bis hin zur Aktuatoränderung im System entsteht. Die Ermittlung dieser Kette geschieht mittels einer dynamischen Analyse [17], bei der sogenannte Trace-Files untersucht werden. Diese Trace-Files werden beim Ablauf des Prototyps erstellt und beinhalten die Kommunikation der Prozesse untereinander. Der Signalkette kann dann eine Zeit zugeordnet werden, innerhalb derer sie abgearbeitet werden muss.

Die dynamische Analyse dient neben der Ermittlung der Zeitanforderungen auch zur Bewertung der SDL-Modelle und wird auch im weiteren Entwurf, speziell der Designphase verwendet [17].

Um die Planung eines Projekts zu unterstützen, wurden verschiedene Abhängigkeiten zwischen den Objekten und ihrem Entwicklungsaufwand untersucht. Es hat sich herausgestellt, dass eine Abschätzung der Komplexität der Objekttypen stark mit dem benötigten Aufwand korreliert ([16], [14], Kapitel 7.1). Die Komplexität eines Objettyps errechnet sich aus der Summe der Komplexitäten seiner Tasks und der Anzahl seiner aggregierten Komponenten. Für die Fallstudie CS3/2 ergab sich eine Korrelation von Komplexität zu dem benötigten Modellierungsaufwand von 0,82 im Schnitt.

Zur Zeit wird an der Systematisierung unserer Methode gearbeitet. Wir haben begonnen, die Entwicklungsprodukte, Teilprozesse und andere benötigte Daten abstrakt zu beschreiben. Dazu haben wir zunächst ein die Produktdaten beschreibendes UML-Modell erstellt, von dem wir dann eine Abstraktion bilden. In diesem abstrakten Modell werden Versionierungsbeziehungen und Zusammenhänge zwischen Produkten, Teilprozessen und Werkzeugen beschrieben [18]. Das abstrakte Modell soll jedoch nicht nur zur Beschreibung der Methode des Requirements-Engineering dienen. Vielmehr soll die Möglichkeit bestehen, alle in der weiteren Entwicklung eingesetzten Prozesse in dieses Modell zu integrieren, um die Schnittstelle zwischen diesen Prozessen zu vereinfachen.

Zur Entwicklung und Erprobung unserer Methode wurde eine Reihe von Fallstudien durchgeführt. Diese sind chronologisch mit einer kurzen Beschreibung aufgelistet:

- **IntelligentWorkplace**: An der Carnegie Mellon University wurden Teile der Steuerung des dort entstehenden Intelligent Workplace modelliert (siehe Arbeitsschwerpunkt B).
- **CS3/1**: Für das Gebäude 32, 4. Stock wurde eine Lichtsteuerung im Rahmen eines Querschnittprojektes realisiert. Zum ersten Mal wurde ein größeres Projekt mit mehreren Personen durchgeführt.
- **CS3/2**: Die Problemstellung von CS3/1 wurde um eine Heizungssteuerung erweitert. Die Erfahrungen aus CS3/1 und die Einführung von Reuse wurden zur Prozessverbesserung eingesetzt. Die resultierenden Veränderungen am Prozess wurden anhand eines erneuten Projektdurchlaufes bewertet.
- **CSD1**: Es wurde die Problembeschreibung von CS3/1 noch einmal bearbeitet, jedoch wurde das in CS3/2 entwickelte System als Ausgangspunkt genommen und daraus die Temperatursteuerungskomponente entfernt. Gezeigt wurde dadurch die Anpassbarkeit der von uns entwickelten Produkte. Im Vergleich dazu wurde in **Light32** dieselbe Problemstellung ohne Wiederverwendung des gesamten CS3/2-Entwurfs bearbeitet. Ähnlich wie in CS3/2 wurde lediglich Wiederverwendung auf Objekttypebene betrieben [28].

Neben den aufgelisteten Fallstudien haben wir eine Reihe kleinerer Projekte durchgeführt. Wir haben einen alternativen Ansatz zur Modellierung der Systemstruktur erprobt [4] und unsere Methode bei der Modellierung und Generierung verschiedener Simulatoren eingesetzt. Gerade hier hat sich die Notwendigkeit gezeigt, neue Beschreibungstechniken wie Petri Netze oder Message-Sequence-Charts in die Methode zu integrieren. Desweiteren haben wir die Änderbarkeit unserer Entwürfe untersucht. Dazu haben wir bestehende Entwürfe bzgl. ihrer Funktionalität und ihrer räumlichen Struktur variiert.

Als Zusammenfassung wird in der folgenden Tabelle eine Überblick über vier Fallstudien gegeben (siehe auch [28]). Es werden die Komplexität des Gesamtprojekts (einheitenlos) und der Gesamtaufwand in Minuten angegeben. Die letzten beiden Zeilen geben den Aufwand pro kleinster Komplexitätseinheit und die Produktivität, normiert auf CS3/1, an.

Vergleich Komplexität und Aufwand	CS3/1	CS3/2	Light32	CSD1
Gesamtkomplexität	688	1826	1190	1190
Gesamtaufwand (min)	19000	29000	17100	1600
Aufwand / Komplexität (min)	27,6	15,9	14,4	1,34
Relative Produktivität	1	1,73	1,92	20,6

4 Zusammenarbeit mit anderen Teilprojekten

Eine enge Zusammenarbeit bestand durch die gemeinsame Arbeit in den Querschnittsprojekten Q1 und Q3 mit fast allen anderen Teilprojekten. Insbesondere gab es gemeinsame Vorarbeiten mit A1 und B1 für die Fallstudien, über die in Q1 berichtet wird (siehe auch [16]). Im Rahmen dieses Querschittsprojekts wurde ebenfalls in Zusammenarbeit mit B10 die Struktur des prozessunterstützenden Project-Dictionarys entworfen. Mit D2 gab es sehr enge Kontakte auf dem Gebiet der Regelung und Steuerung und bei wärmetechnischen Fragen. Die Zusammenarbeit mit C1 hat sich aus der ersten Fallstudie „Beschreibungstechniken im praktischen Einsatz“ der vorangegangenen Antragsperiode fortgesetzt und zu einem von C1 realisierten Gebäudesimulator geführt, der in mehreren Fallstudien zum Testen eingesetzt wurde. Mit B4 wurden Absprachen über das Kommunikationssystem des Testfelds getroffen, um die in B4 entwickelten Kommunikationsprotokolle einsetzen zu können und B4 Experimente zu ermöglichen.

Eine neue Kooperation mit A3 wird zu einer ersten Implementierung unseres Produktmodells der Methode zum Requirements-Engineering in der Erfahrungsdatenbank führen (siehe auch [18]).

Im Querschnittsprojekt Q3 beteiligte sich D1 an der Entwicklung eines Domänenmodells. D1 übernahm die Rolle des Kunden im Querschnittsprojekt Q4 und arbeitete mit an der Integration des dort verwendeten Prozesses mit der in D1 entwickelten Methode zum Requirements-Engineering.

5 Eigene Veröffentlichungen im Berichtszeitraum

Projektarbeiten

[1] R. Boos. *PEditor.* 1998

[2] T. Disper. *Gebäudeautomation mit CAN auf Basis des Philips P8xC592*. 1999

[3] C. Buch. *TRANSdALL - Entwurf eines Interface-Bausteins zum Transfer des Dallas 1-Wire Bus an eine 8051 kompatible CPU*. 1997

[4] K. Haller. *Development of a Lighting Control System in SDL.* 1999

[5] A. Metzger. *Entwurf einer C Bibliothek zur Ansteuerung von Philips P82C150 SLIO Bausteinen auf Basis des Philips P8xC592 Mikrokontrollers unter dem Echtzeitbetriebssystem KEIL RTX-51.* 1997

Diplomarbeiten

[6] R. Boos. *Erweiterung des Pattern-Konzeptes von PsiGene um Typen und Regeln.* 1999

[7] C. Kaesling. *Domänenspezifischer Software-Architektur-Baukasten mit Design-Patterns.* 1998

[8] B. Noll. *Pattern-Basierte Anwendung Endlicher Automaten in PSiGene*. 1999

[9] A. Metzger. *Tool Supported Prototyping for the Specification and the Design of Building Automation Systems*. 1998

[10] M. Schneider. *Entwurf und Implementierung eines echtzeitfähigen Kommunikationssubsystems für CAN*. 1998

[11] B. Weidmann. *Entwurf und Implementierung eines CAN-Bus-Routers.* 1997

Dissertationen

[12] M. Schütze. *Eine musterbasierte Methode zur domänenspezifischen Modellierung und Generierung von Softwarekomponenten.* Universität Kaiserslautern. Veröffentlicht im Shaker Verlag. 1999

SFB-Berichte

[13] A. Metzger, S. Queins, B. Schürmann. *Installation eines Testraums zur Gebäudeautomation und deren Schnittstelle*. SFB 501-Bericht Nr. 13/96. 1996

[14] S. Queins, G. Zimmermann. *A First Iteration of a Reuse-Driven, Domain-Specific System Requirements Analysis Process*. SFB 501-Bericht Nr. 13/99. 1999

[15] G. Zimmermann. *A Domain Specific Model Architecture for Complex Embedded Systems: A Building Automation Case Study.* SFB 501-Bericht Nr. 10/98. 1998

[16] R. Feldmann, J. Münch, S. Queins, S. Vorwieger, G. Zimmerman. *Baselining a Domain-Specific Software Development Process*. SFB 501-Bericht Nr. 2/99. 1999

[17] S. Queins, B. Schürmann, T. Tetteroo. *Bewertung des dynamischen Verhaltens von SDL-Modellen*. SFB 501-Bericht Nr.9/99. 1999

[18] W. Mahnke, U. Marder, S. Queins, N. Ritter, B. Schürmann. I*ntegrierte Produkt- und Erfahrungsdatenverwaltung*. SFB 501-Bericht Nr. 8/00. 2000

[19] L. Geyer, M. Kronenburg, R. Merz, B. Schürmann. *Domain Modeling in the SFB 501*. SFB 501-Bericht 07/00. 2000

Vorträge / Tagungsbeiträge

[20] J.P. Riegel, C. Kaesling, M. Schütze. *Modeling Software Architecture Using Domain-Specific Patterns.* First Working IFIP Conference on Software Architecture (WICSA 1). Kluwer Academic Publishers. 1999

[21] G. Zimmermann. *Domain Specific Model Architecture for Complex Embedded Systems: A Building Automation Case Study.* H. Franke, B. Kleinjohann, J. Sztipanovits: Adaptation and Evolution in Embedded Information Systems. Seminar No. 98441, Report No. 226. Dagstuhl. 1998

[22] G. Zimmermann. *Modeling embedded digital controllers.* New Information Technologies in Science, Education, Telecommunications and Business (IT+SE'98). Yalta-Gurzuff, Ukraine. 1998

[23] B. Schürmann. *Structure and Design of Building Automation Systems.* New Information Technologies in Science, Education, Telecommunications and Business (IT+SE'98). Yalta-Gurzuff, Ukraine. 1998

[24] M. Schütze, J.P. Riegel, G. Zimmermann. *Pattern-Based Application Generator for Building Simulation.* ESEC 97. Zurich (CH). 1997

[25] F. Heister, J.P. Riegel, M. Schütze, S. Schulz, G. Zimmermann. *Pattern-Based Code Generation for Well-Defined Application Domains.* EuroPLoP '97. Siemens Technical Report 120/SW1/FB. Munich (D) / Zurich (CH). 1997

Zeitschriften / Monografien

[26] M. Schütze, J.P. Riegel, G. Zimmermann. P*SiGene - A Pattern-Based Component Generator for Building Simulation.* Journal Theory and Practice of Object Systems (TAPOS). Vol. 5, No. 2. 1999

[27] J.P. Riegel, M. Schneider. *Dynamic Resource Management for the Controller Area Network.* Erscheint in: CAN Newsletter 6/00, CAN in Automation, Nürnberg, 2000

[28] S. Queins, G. Zimmermann. *A Reuse-Driven, Domain-Specific System Requirements Analysis Process*. Special Issue on Requirements Engineering: The Light Control Case Study of the Journal of Universal Software Engineering. 1999

Online-Dokumente

[29] J.P. Riegel. *PSiGene Pattern Catalog.* 2000
http://wwwagz.informatik.uni-kl.de/projects/psigene.html.

[30] A. Metzger. *Dokumentation der Testinstallation für Gebäudeautomationssysteme*. 2000
http://wwwagz.informatik.uni-kl.de/publications/publications.2.dir/Testraum/paper.ps.gz

[31] A. Metzger. *An Interlink of Building Control System Prototypes and the Lighting Simulation Lumina*. 1999

http://wwwagz.informatik.uni-kl.de/publications/publications.2.dir/Met99.0/paper.ps.gz

6 Fremdveröffentlichungen

[32] W. Frakes, C. Terry. *Software Reuse: Metrics and Models*. ACM Computing Surveys. Vol. 28, No. 2. 1996. S. 415–435.

[33] I. Jacobson, G. Booch, J. Rumbaugh. *The Unified Software Development Process*. Addison-Wesley. 1999

Teilprojekt D2

Modellbasierte Entwicklung wiederverwendbarer Regelungsalgorithmen

Leiter

Prof. Dr.-Ing. habil. L. Litz
geb. 30.8.1949
Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern
Tel.: 0631/205-4050
Fax: 0631/205-4462
litz@eit.uni-kl.de

Mitarbeiter

Ergänzungsausstattung:

Dipl. Ing. J. Schäfer (01.06.98 - 01.5.00)

Dr. rer. nat., Dipl. Phys. R. Merz (seit 01.06.98)

Grundausstattung:

Dipl. Ing. P. Denu

Dipl. Ing. A. Diehl

Dipl. Ing. H. Voos

Techniker M. Behr (bis 8.6.1998), S. Becker (seit 1.7.1999)

1 Kurzfassung der Aufgabenstellung

Das Teilprojekt D2 befasst sich mit der *modellbasierten Entwicklung von Regelungs- und Steuerungsalgorithmen*. Hierbei wurden insbesondere wiederverwendungsunterstützende Konzepte beim Entwurf von Steuerungs- und Regelungsalgorithmen sowie der zugrundeliegenden mathematischen Modelle des zu beeinflussenden technisch physikalischen Prozesses untersucht. Das Ziel war neben der Einbindung der Entwurfsmethodik in den allgemeinen Softwareentwicklungsprozess sowohl eine Erhöhung der Entwurfseffizienz wie auch der Qualität. Einen wesentlichen Anteil innerhalb des Teilprojektes D2 nehmen die Validierungsphasen ein. Sowohl die mathematischen Modelle als auch die Algorithmen sollen auf Komponentenebene, aber auch innerhalb exemplarischer Konfigurationen getestet werden. Als Referenz sollen mittels kommerzieller Gebäudesimulatoren (z.B. TRNSYS) erstellte Simulationsmodelle sowie konkrete mit der entsprechenden Sensorik ausgestattete Testobjekte dienen. Die Idee hierbei ist, durch ausschließliche Anwendung validierter Komponenten aussagekräftige Gesamtmodelle erzeugen zu können.

Neben den eigenen Forschungszielen übernimmt das Teilprojekt D2 innerhalb des SFBs die Rolle des *Domänenexperten*. Dies bedeutet, es unterstützt mit Expertenwissen aus der Anwendungsdomäne Gebäudeautomation, aber auch aus der Steuerungs- und Regelungstechnik. Zum einen erfolgt dies durch die direkte Integration beratender Mitarbeiter in die Teams experimenteller Entwicklungsprojekte, zum anderen durch den Ausbau domänenspezifischer Einträge der Erfahrungsdatenbank. Neben der Entwicklung geeigneter Selektionsmechanismen zur Wiederauffindung des erforderlichen Wissens ist eine gezielte Aufbereitung notwendig, welche die allgemeine Nutzbarkeit ermöglicht.

2 Entwicklung des Teilprojekts in der Förderperiode

Dies ist die erste Förderperiode des Teilprojektes D2. Zum Zeitpunkt der Erstellung dieses Berichtes beträgt die bisherige Gesamtbearbeitungsdauer knapp 2 Jahre, da die beiden Mitarbeiter der Ergänzungsausstattung erst zum 1.6.1998 eingestellt werden konnten. Herr Schäfer, verantwortlich für die Algorithmenentwicklung schied aus dem Teilprojekt zum 1.5.2000 aus. Zusätzlich unterstützt wurde D2 im Bereich formaler Spezifikationen von Steuerungsalgorithmen durch Frau Weng (seit Oktober 99).

D2 beteiligte sich als Infrastrukturprojekt intensiv an den Querschnittsprojekten. Noch im Juni 1998 übernahm D2 im Rahmen des Baseline-Experiments QP 1 *(Prototyping eines Gebäudeautomationssystems)* die Experten- und Kundenrolle. D2 stand beratend bei QP 2 *(Entwurf eines Gebäudeautomationssystems)* zur Verfügung und übernahm in der Fallstudie *RECS (Requirements Engineering Case Study: Light Control)* wieder die Kunden- und Expertenrolle. Eine detaillierte Aufbereitung und Analyse der Anwendungsdomäne folgte im Querschnittsprojekt QP 3 (*Entwicklung eines Domänenmodells*). Es zeigte sich, dass der projektierte zeitliche Umfang der Beteiligung an SFB übergreifenden Querschnittsprojekten von 8h/Woche speziell im Falle von Infrastrukturprojekten weitaus zu niedrig angesetzt war. Jedoch rechtfertigt die Effizienz der teilprojektübergreifenden Querschnittsprojekte den erhöhten Aufwand vollständig. Gerade im Falle von D2 konnte so die Integration des Expertenwissens in den allgemeinen Softwareentwicklungsprozess unter realistischen Bedingungen untersucht werden und praktikable Lösungen gefunden werden.

3 Durchgeführte Arbeiten und Ergebnisse

Die modellbasierte Entwicklung von Regelungs- und Steuerungsalgorithmen erfolgt in den Ingenieurwissenschaften durch Nutzung hochentwickelter, mathematischer Simulationsumgebungen ([24]-[25], [30]-[31]). Diese verfügen über ein umfangreiches Repertoire numerischer Funktionen zum Regelungsdesign, effiziente Simulatoren und Codegeneratoren sowie integrierte Tools zur Analyse und Visualisierung. Direkt kompatible Hardware-in-the-Loop-Anbindungen zum schnellen Prototyping sind meist enthalten. Einige verfügen auch über Bibliotheksverwaltungen zur Unterstützung der Wiederverwendung implementierter Komponenten oder kompletter Modelle. Es existieren jedoch starke konzeptionelle Unterschiede der Simulationssysteme.

So war zunächst die Auswahl einer geeigneten Simulations- und Entwicklungsumgebung zu treffen, die es erlaubt generische, wiederverwendungsunterstützende Entwurfsmethoden in die Modell- und Algorithmenentwicklung einzubringen. In den Ingenieurwissenschaften etablierte Blockschaltbildsimulatoren mit starrer Aufteilung aller Schnittstellen in Ein- bzw. Ausgänge sowie mangelnder Unterstützung von Vererbungs- und Aggregationshierarchien erwiesen sich als ungeeignet.

Anders ist dies bei Systemen zur objektorientierten mathematischen Modellbildung, wie der Simulationsumgebung Dymola. Es zeigt sich, dass gerade die Methode der objektorientierten mathematischen Modellbildung zur Beschreibung technischer Prozesse, charakterisiert durch Klassendefinitionen, Vererbungs- und Aggregationshierarchien, Komponentenkapselung sowie nicht berechnungskausaler Modellierung eine ideale Basis zur Realisierung wiederverwendungsunterstützender Entwurfsverfahren darstellt ([23], [26]-[29]) .

Wesentlich für die Vorgehensweise ist die Kompatibilität der mathematischen Modelle sowie der Algorithmen. Beide müssen zur simulativen Validierung innerhalb eines Gesamtsimulationsmodelles zusammengefügt werden. Abhängig vom Simulationsergebnis muss eine schnelle Modifikation der Reglerauswahl, der Reglerparametrierung und eventuell auch des Detaillierungsgrades des mathematischen Modells erfolgen können.

So wurde sowohl die Bibliothek mathematischer Modelle zur Simulation thermischen Gebäudeverhaltens (3.1 Arbeitsschwerpunkt A) wie auch die hierzu kompatible Bibliothek von Steuerungs- und Regelungsalgorithmen (3.2 Arbeitsschwerpunkt B) in Dymola spezifiziert und implementiert.

3.1 Arbeitsschwerpunkt A

Der Algorithmenentwurf basiert auf einer detaillierten mathematischen Modellierung des dynamischen Verhaltens der zu beeinflussenden Strecke. Dies unterstützt bei der Analyse der Problemstellung, der Detektion der Wirkungszusammenhänge und Verkopplungen und ermöglicht so die Festlegung der eigentlichen Regelkreise. Darüberhinaus erlaubt die Modellierung die Parametrierung der Algorithmen und die Durchführung simulativer Tests zur Überprüfung der Eignung für die konkrete Problemstellung. Auch langsame thermische Prozesse und ihre Beeinflussung, die experimentell nur in langwierigen Messungen verfolgbar wären, können so simulativ in überschaubaren Zeiten untersucht werden.

Startpunkt des Arbeitsschwerpunktes A war die gezielte Auswahl der relevanten physikalischen Prozesse durch Analyse der Anwendungsdomäne „thermisches Gebäudeverhalten“. Es folgte die Untergliederung in Teilprozesse synchron mit der Definition von Schnittstellen. Zur Beschreibung der Teilprozesse wurden Komponentenmodelle entworfen. Transparenz und eine klare Strukturierung wurde erzeugt durch Aufbau einer Aggregationshierarchie, die sich an der echten Strukturierung der physikalischen Realität orientiert. Unterstützt wird dies durch die graphische Implementierung der Komponentenrelationen in hierarchischen Objektdiagrammen.

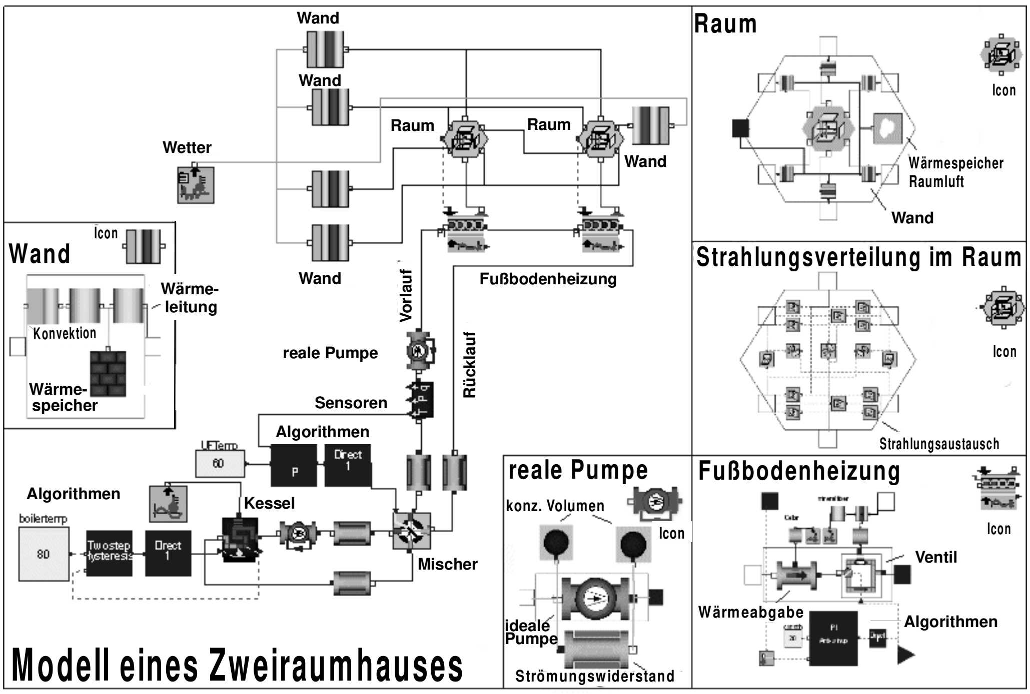

Abbildung 1: Simulationsmodell eines fußbodenbeheizten Zweiraumhauses mit einigen exemplarisch ausgewählten Komponentenmodellen

Innerhalb der Komponentenmodelle wurden physikalische Gleichungen durch Bilanzierung von Erhaltungsgrößen oder auch durch Nutzung phänomenologischer Gesetze erhalten. Beim Entwurf jeder Komponente war abzuwägen, ob diese auf mehrfach nutzbares Basiswissen, beispielsweise physikalische Konstanten und Stoffeigenschaften oder Basisgleichungen zurückgreift, was die Nutzung einer Vererbungshierarchie nahelegt. Die neuen Komponenten verfügen so bei ihrer Definition und Einstellung in die Vererbungshierarchie unmittelbar über die notwendigen physikalischen Grundeigenschaften. Es sind nur noch ihre spezifischen Eigenheiten zu implementieren.

Die Komposition der Komponentenmodelle zum Gesamtsimulationsmodell wurde für exemplarische Konfigurationen typischer Gebäudeteile oder Gebäude vorgenommen. Hierbei erfolgte eine Konzentration auf die verschiedenen physikalisch relevanten Aspekte. Schwerpunkte beim hydraulischen System waren die Wärmeerzeugung im Kessel, das Fließverhalten des Heizungswassers und die Verteilung der Wärme im Rohrleitungssystem, das dynamische Verhalten des Mischventils sowie die Wärmeabgabe von Fußbodenheizung und Radiatoren. Auf Seite der thermischen Speicher wurde die Wärmespeicherfähigkeit sowie der Wärmedurchgang durch bautechnische Komponenten und der Einfluss der Wärmestrahlung betrachtet. Es folgte die Entwicklung von Komponenten zur Wettersimulation und ihre Anbindung an das Hausmodell. Ergänzt zum vollständigen simulationsfähigen Regelkreismodell wurden die Gesamtsimulationsmodelle durch Einbindung von Algorithmenbausteinen aus Arbeitsschwerpunkt B.

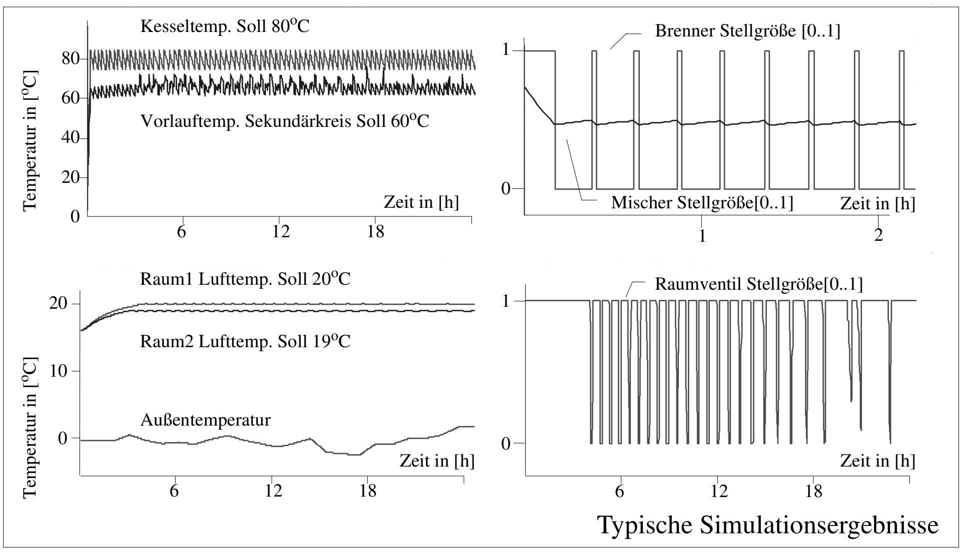

Abbildung 2: Simulationsergebnisse der Regelgrößen sowie Stellgrößen des fußbodenbeheizten Zweiraumhauses aus Abb. 1.

Ausgehend von einfachen, auf einen Aspekt konzentrierten Modellen wurden zunehmend komplexere Gebäudemodelle erstellt (Abb. 1) und simuliert (Abb. 2). Einfache Modelle bilden beispielsweise den Wärmedurchgang durch eine Wand bei festen

Innen- und Außentemperaturen nach. Komplexere Modelle beschreiben Einraum-, Zweiraum- oder Zehnraumhäuser, mit Radiator- und Fußbodenheizung, mehreren Heizkreisen, mischergesteuerten Vorlauftemperaturen, und detaillierten Wettermodellen. Auch die komplexeren Modelle orientieren sich an gebräuchlichen Installationen (Abb. 3).

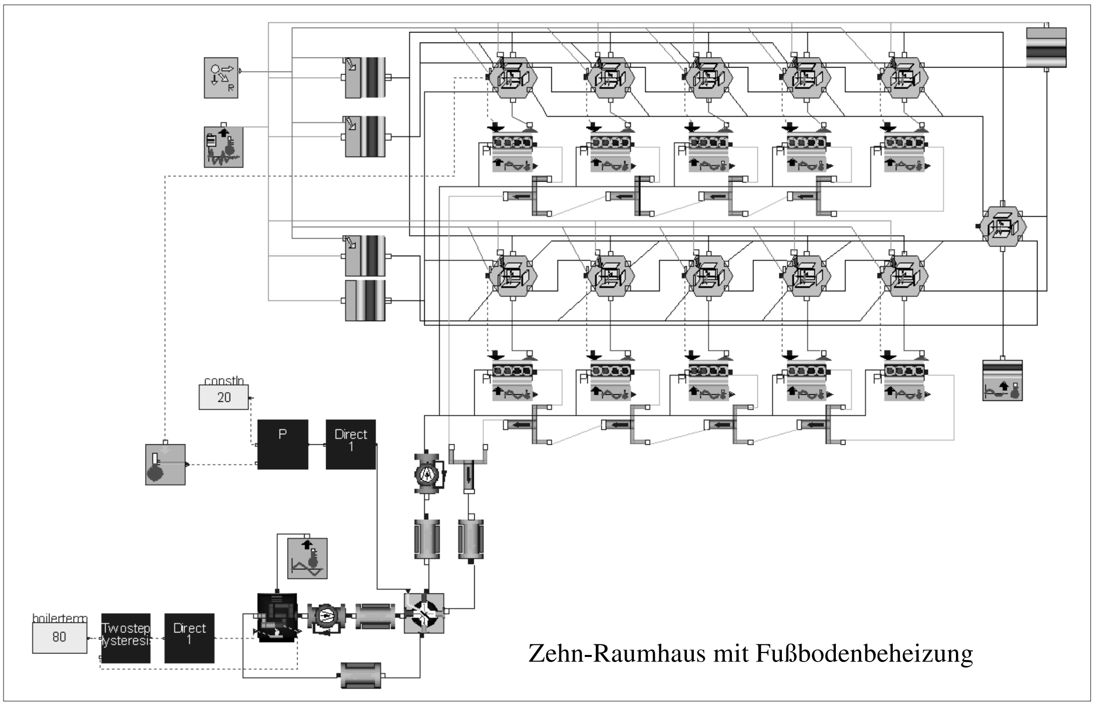

Abbildung 3: Simulationsmodell eines fußbodenbeheizten Zehn-Raumhauses mit hydraulischer Anlage.

Logische Validierungen durch Plausibilitätsbetrachtungen der Ergebnisse, bilanzierende Kontrollrechnungen sowie exemplarische simulative Validierungen wurden angewandt um die Realitätsnähe der mathematischen Modelle zunächst auf Komponentenebene, dann auch für die ausgewählten instantiierten Beispiele zu überprüfen (Abb. 4).

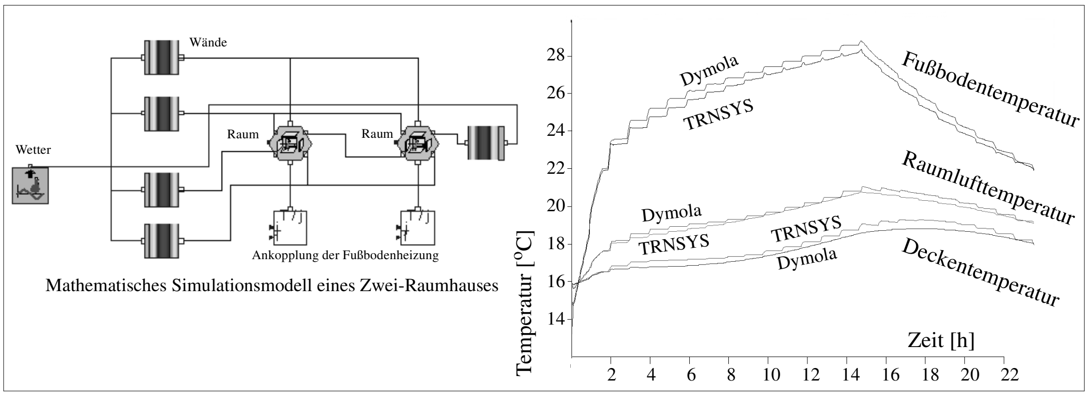

Abbildung 4: Simulative Validierung der mathematischen Modelle in Dymola zur Wärmeverteilung innerhalb eines Raumes durch Vergleich der Resultate mit dem kommerziellen Gebäudesimulator TRNSYS.

Die so validierten Modellkomponenten, wie auch alle ihre Basisklassen wurden ergänzt um Selektionsmerkmale betreffend ihren möglichen Verwendungszweck, Regeln zur Adaption sowie zur Komposition am Einsatzort und in eine Modellbibliothek eingestellt. So kann durch Selektion, Adaption und Komposition für den konkreten Anwendungsfall, generisch ein Gesamtmodell instantiiert werden.

In gleicher Weise werden auch die fertigen, komplexen Modelle in einer Modellbibliothek aufgenommen. Trotz ihrer vollständigen Instantiierung eignen sie sich zur Wiederverwendung. Neben dem exemplarischen Charakter der gespeicherten Modelle kann ein Großteil möglicher neuer Anwendungsfälle durch Komponentenaustausch bzw. -ergänzung der bestehenden Modelle abgedeckt werden. Der modulare Charakter sowie die klare Aggregationshierarchie der entwickelten Modelle unterstützt die einfache Adaption der bestehenden Gesamtmodelle in neuen Einsatzfällen.

Darüberhinaus stellen die Gesamtsimulationsmodelle, ergänzt um die Algorithmenbausteine aus Arbeitsschwerpunkt B lauffähige Regelkreismodelle dar. Sie verfügen damit schon über geeignete Algorithmen bzw. im Falle komplexerer Modelle auch über komplette Regelungskonzepte. Das Auffinden eines geeigneten Simulationsmodells führt damit letzendlich schon zu einem geeigneten Vorschlag bezüglich des Regelungskonzeptes.

3.2 Arbeitsschwerpunkt B

Basierend auf der mathematischen Streckenmodellierung wurden die eigentlichen Regelungsalgorithmen entworfen. Ziel war es mit möglichst wenigen Algorithmen ein möglichst großes Anwendungsspektrum abzudecken. Eine sorgfältige Auswahl der in die Bibliothek einzustellenden Algorithmen war daher unabdingbar. Zunächst wurden Algorithmen auf Komponentenebene, die Teilaspekte und lokale Probleme beherrschen, entwickelt. Mit wachsender Komplexität der mathematischen Modelle aus Arbeitsschwerpunkt A erfolgte dann der Übergang zu komplexeren Regelstrategien, die hierarchisch organisiert, elementare Algorithmen aggregieren.

In gleicher Form wie im Falle der Bibliothek mathematischer Komponentenmodelle existiert eine hierarchisch strukturierte Algorithmenbibliothek. Diese besteht aus Sortimenten von Grundbausteinen zur Sollwertberechnung, Sensor-, Aktorverarbeitung sowie den eigentlichen Reglerkernen (P, PI, PD, PID und Zwei- bzw. Drei-Punkt-Reglern). Zwei exemplarische Tests eines PI-Bausteins sind in Abb. 5 dargestellt.

Diese Bausteine sind ähnlich den Modellkomponenten anwendungsfertig vorkonfektioniert zur Komposition neuer Gesamtalgorithmen. Durch Aufbau mehrfach verschleifter Regelkreise lassen sich durch dieses modulare Konzept auch anspruchsvollere Regelungsaufgaben lösen. Gerade verkoppelte Prozesse mit stark unterschiedlicher Eigendynamik, wie sie im Bereich thermischen Gebäudeverhaltens häufig auftreten, erfordern dies.

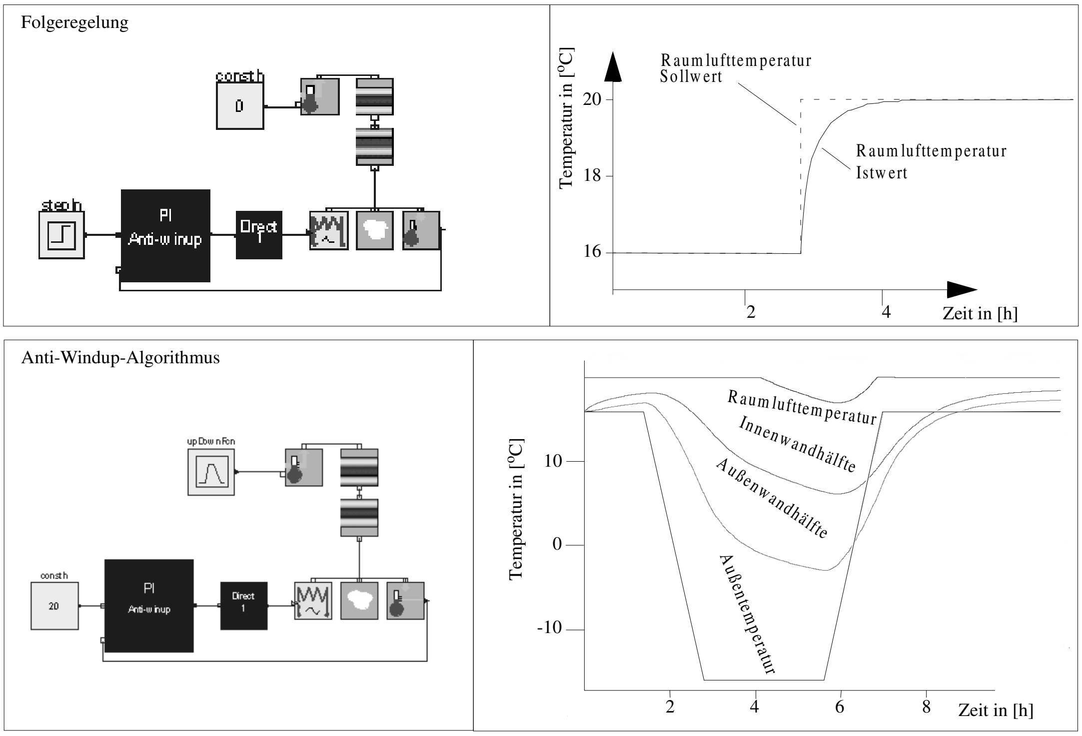

Abbildung 5: Test eines PI-Reglerbausteines mit Anti-Windup-Algorithmus. Der Regler geht auf die maximal mögliche Stellgröße, verbleibt auf diesem Wert, reagiert jedoch korrekt beim erneuten Ansteigen der Temperatur.

Die generische Entwurfsmethodik basiert auf der Modularität des Konzeptes begleitet von dem sukzessiven, prozessbegleitenden Ausbau der Algorithmenbibliothek. Das zur Verfügung stehende Spektrum gespeicherter, validierter Algorithmenkomponenten sowie beispielhafter komplexer Algorithmen soll auch weiterhin stetig zunehmen. Die Algorithmenbibliothek beinhaltet neben den eigentlichen Algorithmenbausteinen Regeln zu ihrer Auswahl, ihrem Einsatzgebiet und ihrer Verknüpfung.

Darüberhinaus existieren konkrete Anwendungsbeispiele durch die in Arbeitsschwerpunkt A schon aufgeführten Bibliothekseinträge der Modellbibliothek mit bereits vollständigen, entwickelten Modellen. Die hier abgelegten Modelle entsprechen simulationsfähigen Regelkreismodellen. Sie bestehen aus instantiierten Gesamtsimulationsmodellen des zu beeinflussenden technisch physikalischen Prozesses, der sogenannten Strecke, sowie den zugehörigen Algorithmen. Sie sind simulativ und exemplarisch experimentell validiert. Sie repräsentieren typische Anwendungsbeispiele von Regelungs- und Steuerungsalgorithmen, die zur Orientierung bei einer Neuentwicklung dienen können, die aber auch durch Modifikation und Adaption an eine neue Aufgabenstellung anzupassen sind. Beispiele unterschiedlichen Detaillierungsgrades und Komplexität wurden entwickelt. Die Bibliothek wurde projektbegleitend durch Aufnahme neuer Modelle erweitert. Die graphische Implementierung sowie der modulare Aufbau der mathematischen Modelle ermöglichen es dem Domänenexperten sofort ein intuitives Bild sowohl der beschriebenen Strecke, als auch der angewandten Lösungskonzepte zu erhalten. Zudem erläutern Kommentare die Charakteristika der durch die Modelle mathematisch beschriebenen, realen technisch physikalischen Prozesse bzw. ihrer Charakteristika.

Alle bislang beschriebenen Bibliotheken enthalten in einer konkreten Simulationssprache implementierte, durch Kommentare ergänzte mathematische Beschreibungen. Sie unterstützen vorwiegend den Domänenexperten beim Entwurf der Steuerungs- und Regelungsalgorithmen (siehe. Abb. 6), da zumindest eine gewisse Erfahrung mit dem mathematischen Simulationstool, sowie dem Begriff des Regelkreises zum Verständnis unabdingbar ist. Neben diesen in mathematischer Form spezifizierten Modell- und Algorithmenbibliotheken entwickelt D2 als Essenz formal spezifizierte, projektübergreifend nutzbare Regelungs- und Steuerungsalgorithmen. Beim Design wurde großen Wert darauf gelegt, dass die Algorithmen auch für Softwareentwickler nutzbar sind und direkt in allgemeine Softwareentwicklungsprozesse einfließen können. Diese zweite Kategorie teilprojekteigener Ergebnisse ist die eigentliche Schnittstelle zum SFB. Sie umfasst einsatzbereite, implementierte Algorithmen die vom Softwareentwickler in das Gesamtsystem integriert werden können. Diese werden letzendlich aus den formal spezifizierten mathematischen Beschreibungen der Simulationssprachen generiert. Bis zum Abschluss der Förderungsperiode soll die automatische Umsetzung vollständig möglich sein. Der Experte liefert dann letztendlich aus der ihm übergebenen Anforderungsbeschreibung der regelungstechnischen Aufgabe sowie der Beschreibung des technisch physikalischen Prozesses integrierbare black-box-artige Codefragmente. Insbesondere bei komplexen regelungstechnischen Aufgaben zur Steuerung großer Systeme, die über die Anwendung einiger weniger Grundbausteine hinausgehen, ist eine stärkere Integration in den allgemeinen Entwicklungsprozess mit der Folge einer aus Sicht der Softwareentwickler erhöhten Transparenz wünschenswert. Während auf niedrigen automatisierungstechnischen Ebenen die Bausteinanwendung adäquat ist und den Softwareentwickler von numerischen, mathematischen Überlegungen befreit, sind auf höheren Ebenen andere Konzepte erforderlich. Daher erarbeitete das Teilprojekt D2 eine ergänzende Methodik.

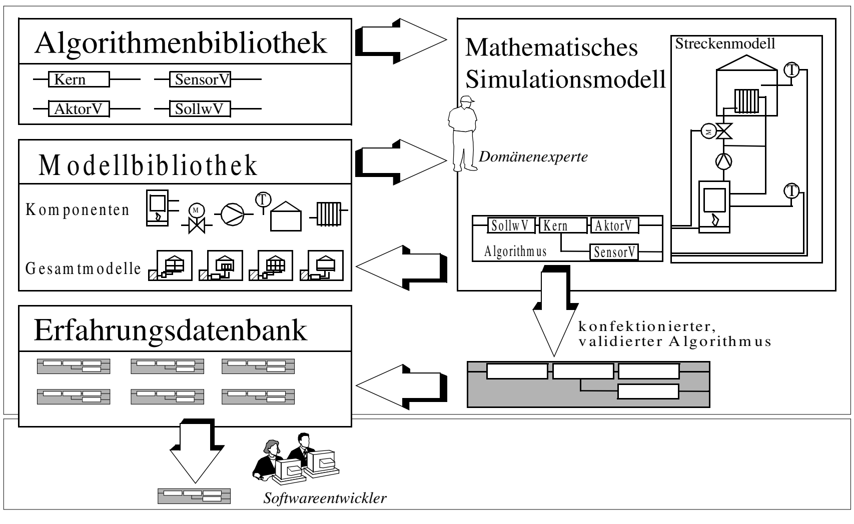

Abbildung 6: Teilprojekteigene Bibliotheksstruktur und Integration im Gesamt-SFB

Übergeordnete Strategien lassen sich auf einer höheren Abstraktionsebene leicht spezifizieren und können im Prinzip als Erweiterung der initialen Anforderungsbeschreibung aufgefasst werden. Ein Beispiel hierfür ist die außentemperaturabhängige Nachführung der Vorlauftemperatur entsprechend einer Heizkennlinie, ergänzt durch thermostatische Einzelraumregelung. Verborgen im mathematischen Simulationsmodell oder im implementierten Codefragment bleibt diese Strategie dem Softwareentwickler bei Fehlersuche oder Notwendigkeit der Modifikation ebenfalls verborgen. Eingeflossen in einer frühen Phase des allgemeinen Softwareentwicklungsprozesses als zusätzliche Anforderung aus Expertensicht erfolgt die Implementierung transparent als Teil des Gesamtsystems. Das Teilprojekt erarbeitet daher zusätzlich einen Katalog höherer Strategien, Taktiken und Konzepte, die formal spezifiziert in einer eigenen Untereinheit der Erfahrungsdatenbank zur Verfügung gestellt werden sollen.

Von Bedeutung sind hierbei insbesondere Selektionsmaße zum Auffinden der Einträge in der Erfahrungsdatenbank. Das Auffinden geeigneter Lösungen basiert auf einer formalen Definition der Eigenschaft *'geeignet'*. Eine Lösung bzw. auch Komponente ist geeignet, falls sie ursprünglich aufgrund *'ähnlicher'* Anforderungen entwickelt wurde. Als Ähnlichkeitsmaßdient dient zunächst die hierarchische Ebenenstruktur der Automatisierungstechnik. Auf niedrigen Ebenen findet man mathematische Beschreibung, auf höheren Ebenen Ergänzungen der initialen Anforderungen aus Expertensicht. Kriterien der Detailauswahl folgen dann aufgrund der Kenntnis der Art und Anzahl von Aktoren und Sensoren, den Charakteristika der konkreten Strecke sowie der Anforderungsdokumente.

4 Zusammenarbeit mit anderen Teilprojekten

Als Infrastruktur- und Serviceprojekt ergab sich eine Vielzahl von Kooperationen mit anderen Teilprojekten.

D2 übernahm im Rahmen des Querschnittprojektes *(QP 1: Prototyping eines Gebäudeautomationssystems)* in einer Doppelbesetzung, zunächst gemeinsam mit D1 die Kunden- und zugleich Expertenrolle. Es war hierbei für die Erstellung von Problembeschreibungen, dem Entwurf und Design geeigneter Steuerungs- und Regelungsalgorithmen zur Integration in das Gesamtsystem, sowie für die Entwicklung realistischer Testszenarien verantwortlich. Gerade die Interaktion mit den Softwareentwicklern innerhalb des Teams erwies sich als idealer Startpunkt des Teilprojektes zur Definition der Expertenrolle und ihrer Aufgaben innerhalb des SFBs. Wann welches Wissen wo in den Entwicklungsprozess einfließen muss, trat am konkreten Beispiel direkt in Erscheinung. Es zeigte sich hierbei auch die primäre Notwendigkeit eine gemeinsame Sprache zu finden, aber auch eine klare Abgrenzung der Tätigkeitsgebiete zwischen Domänenexperte und Softwareentwicklern festzulegen.

Begleitend zu der Teamarbeit wurde in einer Analysephase das notwendige Expertenwissen bezüglich der bauphysikalischen Zusammenhänge, der technischen Installationen sowie Vorschriften und DIN-Normen zusammengestellt. Die Selektion des notwendigen Wissens sowie die allgemeinverständliche Aufbereitung schloss sich den Recherchen an. Dieses Wissen floß dann in das zusammen mit B10 („Anwendungsentwicklung mit vorkonfektionierten Softwaresystemen"), C1 („Formale Beschreibungstechniken") und D1 („Anwendungssystem Gebäude") durchgeführte Querschnittsprojekt QP3 (*QP 3: Entwicklung eines Domänenmodells*) ein. Dieses Domänenmodell soll als Referenzmodell bei der Erstellung von Systemfamilien dienen, es stellt aber auch einen Schritt zur Entwicklung eines generischen Anforderungsdokumentes dar. Eine weitere Detaillierung wird sich anschließen. So kann das Domänenmodell letztendlich dazu dienen, detailliertes Expertenwissen zu speichern und bei der Instanziierung eines konkreten Anforderungsdokumentes den Domänenexperten mit allen im konkreten Anwendungsfall regelungstechnisch relevanten Sachverhalten zu versorgen.

D2 beteiligte sich als Kunde und beratender Domänenexperte an der von den Teilprojekten B4 („Generische Kommunikationssysteme") und C1 („Formale Beschreibungstechniken") durchgeführten Fallstudie *„Requirements Engineering Case Study: Light Control"*. D2 erstellte die initiale, informelle Problembeschreibung und beantwortete in der Kundenrolle alle Fragen, die bei der Erstellung der formalen Anforderungsspezifikation auftraten. Gerade hier war die Integration von D2 als Domänenexperte notwendig, um regulierend ein unbeabsichtigtes Divergieren des erforderlichen Lösungsaufwandes zu vermeiden und der Studie eine realitätsnahe Basis zu geben. Ergebnisse der Studie werden in einem Sonderheft des Journals of Universal Computer Science, Springer Heidelberg veröffentlicht werden. Zudem beteiligte sich D2 als regulärer Teilnehmer an der Fallstudie mittels eines eigenen methodenorientierten Beitrags zur formalen Spezifikation des Steuerungsalgorithmus durch interpretierte Petri-Netze *(submitted to J.UCS, special issue: „Requirements Engineering Case Study Light Control").*

Ein wesentlicher Schwerpunkt des Teilprojektes D2 ist die Nutzbarmachung der teilprojekteigenen Ergebnisse bzw. Produkte. Die Einlagerung der teilprojekteigenen Ergebnisse in die Gesamterfahrungsdatenbank des SFB wurde zusammen mit A1 („SE-Labor") und A3 („Objektrelationale Datenbanktechnologie") erarbeitet. Die Einlagerungen spiegeln die zwei Kategorien der Ergebnisse wieder. Während die zur Simulation geeigneten mathematischen Beschreibungen vor allem von den Domänenexperten genutzt und wiederverwendet werden, werden die vorkonfektionierten Algorithmen von den Softwareentwicklern als fertige Module in Softwarelösungen eingesetzt. Der Schwerpunkt lag daher insbesondere bei der Ausarbeitung charakterisierender Attribute zur Selektion gerade der vorkonfektionierten Algorithmen. Hinzu kommen die formal spezifizierten Regeln und Strategien.

5 Eigene Veröffentlichungen im Berichtszeitraum

Projektarbeiten

[1] C. Anwander: *Simulation des thermischen Verhaltens eines Niedrigenergiehauses mit dem Gebäudesimulator TRNSYS*; Projektarbeit, Fachbereich Elektro- und Informationstechnik, Universität Kaiserslautern, Januar 2000

[2] M. Grünewald: *Entwicklung eines Industrie PID-Reglers unter vergleichender Anwendung der Tools SDT*, Matlab/Simulink und Dymola, Juli 2000

[3] F. Felgner: *Objektorientierte mathematische Modellierung der Hydraulik von Heizungsanlagen,* September 2000.

Diplomarbeiten

[4] O. Seel: *Objektorientierte mathematische Modellbildung*; Diplomarbeit, Fachbereich Elektro- und Informationstechnik, Universität Kaiserslautern, Juni 2000

[5] E. Sitompul: *Entwicklung von generischen Regelungsalgorithmen zur Hausbeheizung unter kombinierter Anwendung von Dymola und Matlab,* September 2000

Dissertationen: keine im Berichtszeitraum

SFB-Berichte

[6] R. Merz und L. Litz: *Anwendung generischer Methoden bei der objektorientierten Modellierung großer, komplexer dynamischer Systeme*, Technical Report SFB 501- 07/99, Fachbereich Elektro- und Informationstechnik, Universität Kaiserslautern, 1999

[7] J. Schäfer und L. Litz: *Objektorientierte Entwicklung von Regelungs- und Steuerungsalgorithmen für die Anwendungsdomäne Gebäudeautomation unter den Aspekten Wiederverwendbarkeit und Generik*, Technical Report SFB 501-010/99, Fachbereich Elektro- und Informationstechnik, Universität Kaiserslautern, 1999

Vorträge: keine im Berichtszeitraum

Tagungsbeiträge

[8] A. Diehl, M. Becker: *Ein dezentrales, modulares Konzept zur Automatisierung kältetechnischer Anlagen.* In: Tagungsband GMA-Kongreß'98 Mess- und Automatisierungstechnik *"Neue Entwicklungen Technologie Anwendungen"*, Ludwigsburg, Juni 1998. VDI Berichte 1397, S. .395-402, VDI-Verlag, Düsseldorf, 1998

[9] A. Diehl, L. Litz: *Object-Oriented Top-Down Modeling For Dynamic Simulation Of Copression Plants*. In the Proceedings of the 20th International Congress of Refrigeration, IIR/IIF, Sydney, 1999, Paper Code: 064

[10] A. Diehl: *Dynamic Simulation of Refrigeration Plants Based on a Object-Oriented Top-Down Model.* In: Proceedings of Modelling and Control Conference 2000, MIC-2000, Innsbruck, 2000, Paper Code: 306-021

[11] G. Frey und L. Litz: *Entwurf und formale Verifikation von Steuerungen mit interpretierten Petri-Netzen*, In: Tagungsband GMA-Kongreß'98 Mess- und Automatisierungstechnik *Neue Entwicklungen Technologie Anwendungen*, Ludwigsburg, Juni 1998. VDI Berichte 1397, S. 291-298, VDI-Verlag, Düsseldorf, 1998.

[12] G. Frey: *Simulation of Hybrid Systems based on Interpreted Petri Nets*, Proceedings of the IEE International Conference on Simulation - Innovation Through Simulation, York, 30.09. - 02.10.98, S. 168-175.

[13] G. Frey und L. Litz: Verification and Validation of Control Algorithms by Coupling of Interpreted Petri Nets, Proceedings of the IEEE SMC'98, San Diego, 11.-14. 10.98, Band 1, S. 7-12

[14] J. Schäfer: *Reuse- and Object-Oriented Development of Standard Algorithms* IFAC Workshop on Digital Control: *Past, Present and Future of PID Control* 5.-7.4.2000 Vapor Universitari, Terassa, Spain 2000

[15] H. Voos, R. Habtom, L. Litz: *Development of an Engineering Tool for the Design and Optimization of Knowledge-based Control Structures*. In Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering SEKE'99, Kaiserslautern 1999, S. 327-331

[16] X. Weng, L. Litz: *Verification of logic control design using SIPN and model checking - Method and case study.* Erscheint in: Proceedings of the American Control Conference ACC 2000, Chicago, 28.-30. Juni 2000.

Zeitschriften / Monografien

[17] M. Becker und A. Diehl: *Optimierung von Kälteanlagen*. In: *ki Luft- und Kältetechnik* 5(1998), C. F. Müller Verlag, Hüthig GmbH, Heidelberg, S.262-263

[18] M. Becker, A. Diehl, R. Habtom: *Modellierung einer kältetechnischen Anlage mit rekurrenten Neuronalen Netzen.* atp - Automatisierungstechnische Praxis 9/99, R. Oldenbourg Verlag, S. 22-31.

[19] A. Diehl, M. Becker, L. Litz: *Die Automatisierung in der Kältetechnik - Gestern und heute.* In: ki Luft- und Kältetechnik 2(1999); C.F.Müller Verlag, Hüthig GmbH, Heidelberg; S.101 - 106.

[20] A. Diehl: *Technische und wirtschaftliche Optimierung von Kälteanlagen. In: ki Luft- und Kältetechnik* 5(1999); C.F.Müller Verlag, Hüthig GmbH, Heidelberg; S.296 - 297

[21] R. Merz und L. Litz: *Objektorientierte mathematische Modellierung: Generische Methoden bei komplexen dynamischen Systemen*, Informatik Spektrum 2(2000), Springer Verlag Heidelberg, S. 90-99

[22] L. Litz und G. Frey: *Methoden und Werkzeuge zum industriellen Steuerungsentwurf - Historie, Stand, Ausblick,* at-Automatisierungstechnik 47(1999)4, R. Oldenbourg Verlag, S. 145-156

6 Fremdveröffentlichungen

[23] P. Beater: Objektorientierte Modellierung physikalischer Systeme Teil 10. at1, A1-A5 (2000)

[24] *Dymola.* Lund/Schweden: Fa. Dynasim, 1999; *http://www.dynasim.se*

[25] J. Hoffmann: *Matlab* und *Simulink*. Bonn: Addison -Wesley 1998

[26] M. Otter: Objektorientierte Modellierung physikalischer Systeme Teil 1-5. at1, A1-A4(1999); at2, A5-A8(1999); at3, A9-A12(1999); at4, A13-A16(1999); at5, A17-A20 (1999)

[27] M. Otter, B. Bachmann: Objektorientierte Modellierung physikalischer Systeme Teil 6. at6, A21-A24(1999)

[28] M. Otter, H. Elmqvist, L. Mattsson und S.E. Mattsson: *Objektorientierte Modellierung physikalischer Systeme* Teil 7 + 8. at9, A25-A28(1999); at11, A29-A32(1999)

[29] M. Otter, C. Schlege: *Objektorientierte Modellierung physikalischer Systeme* Teil 9. at12, A33-A36(1999)

[30] *Simulink*: *Matlab/Simulink*, Natick/USA MathWorks Inc.1999; *http://www.Mathworks.com*

[31] *Smile*; *http://buran.fb10.tu-berlin.de/Energietechnik/EVT_KT/smile/*
Smile-Usertag GMD first, Berlin: 16. April, 1999; *http://www.first.gmd.smile*