7 Leitbilder für die Lehre des Systementwurfs: ein Studienmodell der Informatik-Systemtechnik

Walter Lang, Klaus Quibeldey-Cirkel, Hans Wojtkowiak

Lehrstuhl für Technische Informatik, Universität Siegen, 57068 Siegen
e-mail: quibeldey@sieis.informatik.uni-siegen.de

Zusammenfassung: Die „Halbwertszeit" des Informatikwissens wird auf 3–4 Jahre geschätzt, technologieneutrales Wissen währt im allgemeinen länger [6]. Damit einher geht die Diskussion um Langzeit-Anforderungen an die berufliche Qualifizierung der Informatik-Studierenden. Es werden Stimmen laut, die eine Revision der Lehrinhalte alle 5–6 Jahre fordern [13]. Die curricularen Forderungen reichen von Reformansätzen über die Gründung neuer Studiengänge bis zur Umstrukturierung der Kern-Informatik in eine *Technik-Wissenschaft* [11] oder *System-Informatik* [14]. Parallel sind IEEE-Bestrebungen im Gange, das Curriculum einer fächerübergreifenden Ingenieurdisziplin zu erstellen: ECBS, „Engineering of Computer-Based Systems", übertragen *Informatik-Systemtechnik* [23]. Ziel ist der industrielle Systementwurf aus analogen, digitalen und mechanischen Komponenten.

Unser Beitrag beschreibt den Weg zu einem *mittelfristig* realisierbaren ECBS-Curriculum: Wir stellen ein *Studienmodell* vor, das eine berufsqualifizierende Vertiefung im Systementwurf bietet. Die Konzeption des Studienmodells orientiert sich an Leitbildern, die wir hier näher erläutern.

Schlüsselworte: Entwurfslehre, Systementwurf, ECBS, Informatik-Systemtechnik, Studienmodell, Projektlabor, Qualität der Lehre

1 Lehre des Systementwurfs

Edward Yourdon hat es in „Decline and Fall of the American Programmer" [25] vorweggenommen: Das klassische Berufsbild des Informatikers verblaßt; *Outsourcing* — die Verlagerung der Programmier-Dienstleistungen in „Billig-Lohnländer" — macht den Haus-Informatiker wirtschaftlich obsolet: „Man kann [...] sagen, daß die bei uns überwiegende Produktion von Prozedural-Informatikern (>90 % der Absolventen mit Praktischer Informatik als Schwerpunkt) inzwischen sehr bedenklich geworden ist" [14, R. Hartenstein]. Das Anforderungsprofil der Wirtschaft hat sich radikal gewandelt: Gesucht wird der *System-Informatiker*, „der ein Problem jeweils gleichrangig kompetent (a) in Software, (b) in Hardware und (c) in einer Kombination aus beiden lösen kann". Gerade der Systementwurf von *Embedded Systems* verlangt den Mix aus Soft- und Hardware-Kompetenzen.

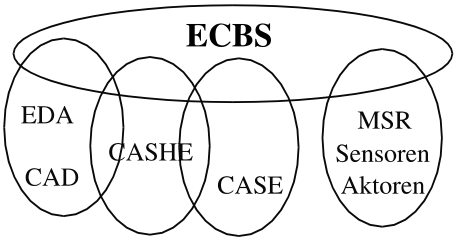
Didaktisches Ziel einer jeden Ausbildung ist der *wettbewerbsfähige* Absolvent. Wettbewerbsfähigkeit wird stets neu definiert, je nach Stand der Technik und den Bedürfnissen des Marktes: *technology push — market pull*. Praxis-Adäquanz sollte also im Mittelpunkt der curricularen Diskussion zur Informatik-Ausbildung stehen.

1.1 Entwurfslehre im großen

Die curriculare Standortbestimmung der Entwurfslehre findet derzeit in einem *didaktisch großen* Rahmen statt: eigenständige wissenschaftliche Studiengänge stehen in der Diskussion. Dabei zeichnen sich 3 Zielkategorien ab: 1. *inter*-disziplinär im Überschneidungsbereich der ingenieurgemäßen Informatik (VLSI-Entwurf und Software-Engineering) mit den klassischen Ingenieurwissenschaften, 2. *intra*-disziplinär im Bereich der Technischen Informatik und 3. *disziplinär* im Gesamtbereich der klassischen Informatik. Die wichtigsten Niederschläge dieser Diskussion werden kurz skizziert:

1. IEEE-Initiative ECBS „Engineering of Computer-Based Systems“

   Industrielle Systeme verbinden heute logistische mit technischen Funktionen auf der Grundlage mikroelektronischer und mechanischer Komponenten. Die Informatik liefert die Methoden, Werkzeuge und Techniken für die *Integration* der Komponenten. Beispiele für diese Systemklasse sind *rechnergestützte* Telekommunikations- und Flugleitsysteme. Als Sammelbegriff wird hierfür „Informatiksysteme“ vorgeschlagen [23]. Die Ingenieurdisziplin, die sich mit der Lehre von Informatiksystemen befassen soll, heißt „Informatik-Systemtechnik“ (deutsche Übertragung von ECBS). Bild 1 zeigt die interdisziplinären Überschneidungen zwischen den technischen Entwurfsfeldern der Informatik und den klassischen MSR-Techniken (Messen, Stellen und Regeln):



ECBS	Engineering of Computer-Based Systems
CAD EDA	Computer-Aided Design Electronic Design Automation
CASE	Computer-Aided Software Engineering
CASHE (Codesign)	Computer-Aided Software/Hardware Engineering

Bild 1: Disziplinen des Systementwurfs

2. Studiengänge der Technischen Informatik

   Die Technische Informatik versteht sich primär als „Technik-Wissenschaft“ [11] mit einem Bekenntnis zur ingenieurgemäßen Vorgehensweise. Dieses Selbverständnis spiegelt sich in den Bezeichnungen der Studiengänge wider: Beispiele sind die „Ingenieur-Informatik“ an der Universität Dortmund, die „Techno-Informatik“ an der Universität Kaiserslautern oder die „Informationstechnik“ an der Universität Paderborn [14].

3. Neustrukturierung der Kern-Informatik

   In [14] schlägt Klaus Waldschmidt einen umfassenden *alleinigen* Studiengang „System-Informatik“ vor, angelehnt an den angelsächsischen Begriff „Computational Science“. Jedes Teilgebiet im Bild 2 wird gleichermaßen in Theorie, Praxis und Anwendung vertreten. Ein fundiertes mathematisches Grundstudium wird vorausgesetzt.

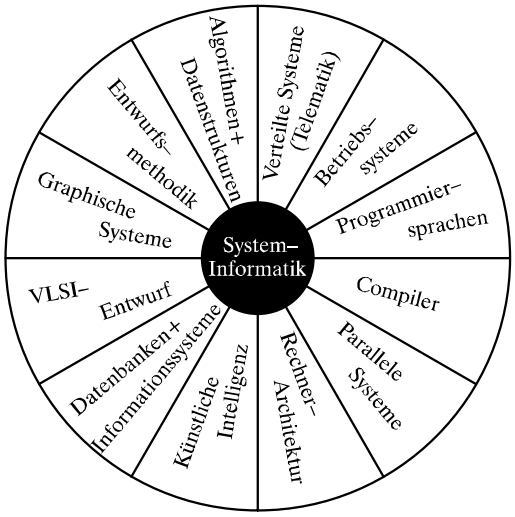


Bild 2: Vorschlag zur Neustrukturierung

Der von uns verfolgte Weg zu einem *mittelfristig* realisierbaren ECBS-Curriculum heißt *Studienmodell*. Es definiert einen *didaktisch kleinen* Rahmen *innerhalb* eines etablierten Informatik-Studiengangs.

1.2 Entwurfslehre im kleinen

Neue Studiengänge sind verwaltungstechnisch und finanziell zu aufwendig und in der Umsetzung zu langwierig. Aus dem Ergebnisbericht des 1. Treffens der Hochschullehrer für Technische Informatik zur eigenen Standortbestimmung:

„ ... Dennoch wurde vor zu vielen Reformansätzen gewarnt. Man muß die Mängel bei der Informatik abstellen, nicht neue Richtungen konstruieren, die die Gefahr bieten, nach außen hin wie Luftschlösser zu wirken." [14]

Das steht im Einklang mit den Empfehlungen des Deutschen Stifterverbandes zur Qualifikation von Hochschulabsolventen:

„ ... keine weiteren spezialisierten Studiengänge mehr einzurichten, bestehende Studiengänge zu straffen und dabei auch die Qualifikationsanforderungen der Wirtschaft zu berücksichtigen." [15]

Oder, um es auf den Punkt zu bringen, das Hochschulrahmengesetz fordert geradezu die ständige *Studienreform* in § 8:

„Hochschulen haben [...] Inhalte und Formen des Studiums im Hinblick auf die Entwicklung der Wissenschaft (und Kunst), die Bedürfnisse der beruflichen Praxis und die notwendigen Veränderungen in der Berufswelt zu überprüfen und weiterzuentwickeln." [16]

Und § 10 betont, daß Studiengänge zu einem *berufsqualifizierenden* Abschluß führen. Unserer Erfahrung nach kann man mit neuen Studiengängen nicht flexibel und schnell genug auf veränderte Berufsbilder reagieren. Anders verhält es sich dagegen mit Studienmodellen: Hier sind zeitgemäße berufsqualifizierende Vertiefungsrichtungen relativ einfach aus bestehenden und neuen Lehrveranstaltungen konfigurierbar (*thematischer* Fächerkatalog). Im folgenden beschreiben wir diesen Weg am Beispiel des Studienmodells „Informatik-Systemtechnik".

Die Siegener Situation

Traditionell besteht eine enge Verzahnung zwischen Informatik und Elektrotechnik: Bild 3. Die Siegener Situation ist durch den *gemeinsamen* Fachbereich beider Disziplinen gekennzeichnet. Allerdings existiert kein eigenständiger Studiengang Informatik; angeboten wird allein der Studiengang Technische Informatik, der mit dem Grad „Diplom-Ingenieur" abschließt. Die Verzahnung beider Disziplinen findet ihre Ausprägung vor allem in der Mikro-Systemtechnik. Ein Beispiel für die dabei entstehende Hardware-Software-Synergie ist das von uns als Posterbeitrag zum E.I.S.-Workshop vorgestellte Laborvehikel des „anwendungsspezifischen Steuerungsrechners" [10]. Anhand dieses Vehikels lehren wir Rechner-Architektur und Compilerbau im Kontext der Automatisierungstechnik (erst die konkrete Anwendung zeigt die Leistungsfähigkeit einer Rechner-Architektur, und eine Architektur ist nur so gut wie ihr Compiler).

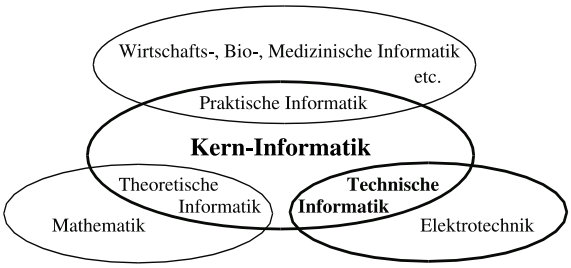


Bild 3: Klassische Informatikbereiche

Der Studiengang Technische Informatik ist somit prädestiniert für Vertiefungsrichtungen mit mikro-systemtechnischen Bezügen. Bisweilen ist es den Studierenden überlassen, welche Wahlpflichtfächer sie miteinander kombinieren. Die Erfahrung zeigt aber, daß die Studierenden in ihren Wahlmöglichkeiten vielfach überfordert sind. Es fehlt das Wissen um die curricularen Querbezüge, um Studienschwerpunkte *praxisadäquat* bilden zu können. Das Konzept des Studienmodells kann hier die nötige Orientierung leisten. Im folgenden werden wir den curricularen Rahmen — das Studienmodell „Informatik-Systemtechnik" (IST) — näher erläutern.

1.3 Informatik-Systemtechnik

Die Veranstaltungsformen „Projektgruppen" und „Informatik-Seminare" bieten einen *flexiblen* und *integrativen* Vertiefungsrahmen innerhalb eines etablierten Studiengangs. Gerade die Informatik-Seminare werden in ihrer curricularen *Bindungs-*

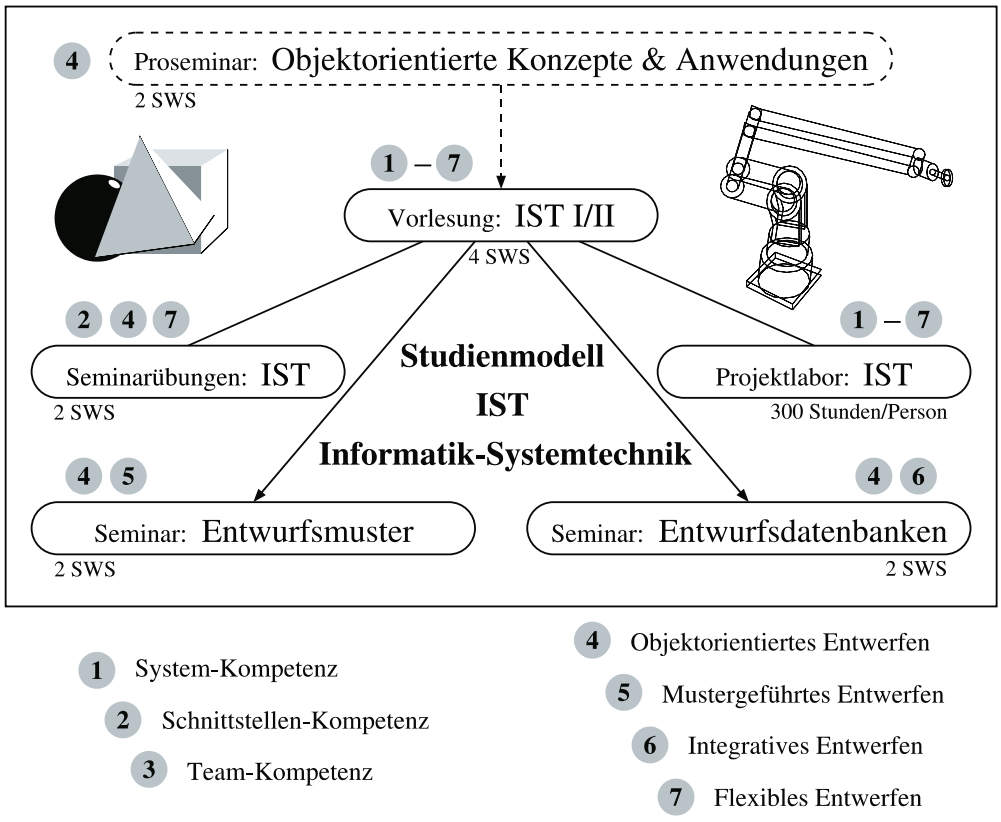


Bild 4: Die Leitbilder im Studienmodell

funktion meist unterschätzt. Die jederzeit aktualisierbaren Themenkataloge und „Reader“ (Sammlung ausgewählter Fachaufsätze) können als Katalysatoren für Interessenschwerpunkte fungieren: Die Seminarthemen und die Seminarliteratur motivieren viele Teilnehmer zum Besuch weiterführender Lehrveranstaltungen. Beispiele unserer Erfahrung: Das Seminar „Entwurfsdatenbanken“ motiviert die Vertiefung in den Fächern „Informationssysteme“ und „Software-Entwicklungsumgebungen“; das Seminar „Entwurfsmuster“ motiviert die Vertiefung im Fach „Programmiermethodik“. Analoges gilt für das „Projektlabor“: Aus dieser Laborform stammen derzeit die meisten unserer Diplomanden.

Inhaltlich vereinen wir die Informatik-Systemtechnik mit den objektorientierten Themen. Die engen Bezüge zwischen dem Objekt-Paradigma und der ECBS-Ingenieurdisziplin wurden in [21] herausgearbeitet. Unser Studienmodell umfaßt derzeit die folgenden Veranstaltungen (Bild 4):

- Integrierte Lehrveranstaltung IST I/II

  Die Lehrveranstaltung umfaßt Vorlesung, Übungen und Projektgruppen und ist Bestandteil des Wahlpflicht-Fächerkatalogs für alle Studienrichtungen der Elektrotechnik und der Technischen Informatik. Die Themen der Vorträge und Seminarübungen nennt Tabelle 1.

- Projektlabor

  Unter diesem Laborkonzept organisieren wir die Projektgruppen zur „Informatik-Systemtechnik“. In unserem Posterbeitrag zum E.I.S.-Workshop werden Form und Inhalt des Projektlabors detailliert beschrieben [10]. Das Konzept „Projektgruppe“ — als Äquivalent zur Studienarbeit (300 Stunden) —

„Informatik-Systemtechnik“ IST I		Seminarübungen: Software-Entwurf	
1.	Überblick: Objektorientierter Systementwurf [20]		
2.	Entwurfsparadigmen in der Informatik	1.	Organisation der Projektgruppen
3.	Entwurfskomplexität		
4.	Komplexitätsbewältigung	2.	Übungen zur „Entwurfsfreiheit“
5.	Software-Industrialisierung		
6.	OOx: Abstrahieren, Teilen, Kommunizieren	3.	OOAD: Booch-Methode [3]
7.	OOAD: Grundlagen der Analyse- & Designmethoden		
8.	OOP: Definition & Einordnung der Objekt-Sprachen	4.	Booch-Methode: Notation I
9.	Kognitive Aspekte: Entwerfen als Problemlösen		
10.	Vom „Tripel des Objekts“: Struktur & Verhalten & Constraints	5.	Booch-Methode: Notation II
11.	Managementaspekte: Techniktransfer & Projektorganisation		
12.	Von einer „Wissenschaft des Entwerfens“	6.	OOAD: Entwurfsbeispiel
13.	Rückblick: Fragenkatalog & Lehr-Evaluation		
Projektziel in der vorlesungsfreien Zeit: OOAD-Modell des Informatiksystems (Gebäude-Leittechnik)			

„Informatik-Systemtechnik“ IST II		Seminarübungen: Hardware-Entwurf	
1.	Überblick: Leitbilder der Informatik-Systemtechnik		
2.	Ökonomie des Entwerfens: Wiederverwendung	1.	Grundlagen zum Prozessorentwurf
3.	Wiederverwendbares Expertenwissen: Entwurfsmuster		
4.	Wiederverwendbare Infrastrukturen: CAD-Frameworks	2.	Entwurf integrierter HW-Systeme
5.	Das Für und Wider formaler Methoden		
6.	Formalismen I: Constraint-Systeme	3.	Grundlagen zum Compilerbau
7.	Formalismen II: Hardware-Software-Codesign		
8.	Formalismen III: Feldprogrammierbare Hardwaresysteme	4.	CAD-Werkzeuge: Fallstudie I
9.	Berufsbild „System-Informatiker“		
10.	Rückblick: Fragenkatalog & Lehr-Evaluation	5.	CAD-Werkzeuge: Fallstudie II
Projektziel in der vorlesungsfreien Zeit: Programmierung des Informatiksystems auf UPCS-Basis [10]			

Tabelle 1: Lehr- und Übungsziele der IST-Vorlesung

wurde erstmals und treffend in der Dortmunder Diplom-Prüfungsordnung zum Studiengang „Ingenieur-Informatik“ charakterisiert:

> „(1) Eine Projektgruppe dient der Vermittlung typischer Arbeitsmethoden des Informatik-Ingenieurwesens bei der Bearbeitung umfangreicher Problemstellungen. Sie bereitet auf das in der späteren Berufspraxis wichtige arbeitsteilige Vorgehen vor. Zugleich werden in einer Projektgruppe Methoden der Ingenieur-Informatik angewendet und bestehende Kenntnisse vertieft.
> (2) Eine Projektgruppe vereinigt die Lehrveranstaltungsformen eines Seminars, einer Spezialvorlesung, eines Fortgeschrittenenpraktikums und eines Kolloquiums ... “ [14]

Das Aufgabenspektrum unserer Projektgruppen stammt aus der regionalen Industrie: von Anlagenbauern und Ingenieurbüros, die den Entwurf von Informatiksystemen betreiben. Die Projektziele liegen primär in den frühen Phasen des objektorientierten Entwurfs: Systemanalyse und -design (gegenwärtig im Bereich der Gebäude-Automation). Es wird aber grundsätzlich der gesamte Entwicklungszyklus durchlaufen: vom Kundenauftrag bis zum validierten Prototypen.

- Proseminar „Objektorientierte Konzepte & Anwendungen“

  Objektorientierte Themen durchdringen alle Entwurfsphasen — von der Spezifikation bis zur Programmierung komplexer Systeme. Mehr noch: selbst die *kognitive* Phase der Ideenfindung orientiert sich am Objekt. Das Proseminar analysiert, erläutert und bewertet das Spektrum der objektorientierten Mo-

delle und Methoden. Technische, wirtschaftliche und menschliche Gründe motivieren den objektorientierten Systementwurf [20]. Tabelle 2 nennt den Kanon der Referatsthemen:

„Objektorientierte Konzepte & Anwendungen“
Objektorientierung: Historie & Einordnung
Abstraktion: Begriffe — Klassen — Objekte
Ressourcen-Teilung: Ausprägung & Vererbung
Kommunikation: Polymorphie & Message passing
Objektorientierung im Software-Engineering
Systemanalyse nach Coad/Yourdon
Systementwurf nach Coad/Yourdon
Programmierung in Eiffel: Zusicherungskonzept
Programmierung in C++: hybrid vs. homogen
Persistente Objekte: Datenbank-Managementsysteme
Wiederverwendung: Entwurfsmuster & Frameworks
Objektorientierung: ein Grundprinzip der Informatik?

Tabelle 2: Themen des Proseminars

- Seminar „Entwurfsmuster“

  Technische Entwurfsmuster sind Beschreibung und Vorlage zugleich: sie unterstützen den Entwurfs*prozeß* und dokumentieren die Struktur des Entwurfs*produkts*. Sie erlauben den Entwurf komplexer Systeme auf einer hohen Erfahrungsstufe, ohne daß der Entwerfer diese Erfahrung selbst gemacht haben muß. Entwurfsmuster sind bewährte Problem-Lösungs-Paare in einem bestimmten technischen Kontext. Sie sind nicht die Lösung selbst, sondern zeigen erprobte Lösungsstrategien auf. Die Teilnehmer diskutieren Vorschläge aus der Gebäude-Architektur [1] (hier dienten Entwurfsmuster erstmals als Design-Philosophie), aus der Softwaretechnik [5] (als Fortsetzung der objektorientierten Wiederverwendung auf einer hohen Wertschöpfungsstufe) und aus dem Schaltungsentwurf (hier haben Entwurfsmuster im *zellbasierten* Chipentwurf immer schon die dominante Rolle gespielt). Tabelle 3 zeigt einen Ausschnitt aus dem derzeitigen Themenkatalog:

- Seminar „Entwurfsdatenbanken“

  Der moderne Chip-Entwurf ist äußerst umfangreich an heterogenen Daten: Komplexe

„Entwurfsmuster“
Entwurfsmuster: Begriffsbestimmung
Architektonische Entwurfsmuster [1]
Entwurfsmuster & Software-Wiederverwendung
Entwurfsmuster-Kataloge [8]
Mustergeführtes Entwerfen & OOAD
Dokumentation mit Entwurfsmustern
Hardware-Entwurfsmuster an Beispielen
Software-Entwurfsmuster an Beispielen

Tabelle 3: Themenauszug des Seminars

Entwurfsobjekte werden auf verschiedenen Abstraktionsebenen (von der Systemspezifikation bis zur Transistorebene) und unter verschiedenen Sichten (Verhalten, Struktur, Physik) im Team entworfen und einheitlich verwaltet. Die *Integrität* der Entwurfsdaten ist das zentrale Problem. Der Ansatz über eine relationale Datenmodellierung erwies sich als ineffizient. Objektorientierte Datenmodelle suchen den Kompromiß zwischen der flexiblen Formulierung der Integritätsbedingungen und der Laufzeiteffizienz der Datenhaltung. Die Teilnehmer diskutieren aktuelle Forschungsergebnisse zum datenbankbasierten Soft- und Hardware-Entwurf [19]. Auch hier dient ein Themenausschnitt zur Verdeutlichung der Lehrinhalte: Tabelle 4.

„Entwurfsdatenbanken“
Semantische Datenmodellierung
Informationsmodellierung mit Express
Objektorientierte Datenbanken am Beispiel
Kooperativer Entwurf: Transaktionen & Versionen
CAx-Entwurfsdatenbanken
VHDL-Entwurfsverwaltung
VLSI-Synthesesysteme an Beispielen
Software-Entwicklungsumgebungen
Kooperative Autorensysteme

Tabelle 4: Themenauszug des Seminars

1.4 Zur Qualität der Lehre

Zum Thema „gute Lehre“ und „effektive Lehre“ siehe [16]. Wir teilen die dort vertretene Meinung, „daß häufig gute Lehre auf Medieneinsatz und die bessere Präsentation beliebiger Inhalte reduziert wird. Traditionelle Lehre wird oft fortgeschrieben, Forschungsergebnisse über studentisches Lernen werden nicht aufgenommen“. Mit unserem Studienmodell „Informatik-Systemtechnik“ wollen wir primär die Studierenden auf die *autonome* Bewältigung professioneller und personeller Probleme vorbereiten. Auf die Überwindung des *Praxisschocks* [17] sind die gruppendynamischen Prozesse im Projektlabor ausgerichtet.

Das Lernen von Fakten ist dem „Verstehen durch Üben“ von Methoden und Techniken nachgeordnet (*I see — I forget, I hear — I remember; I do — I understand*). Die berufsnahe Verknüpfung und Erprobung des Gelernten steht im Vordergrund. Im folgenden werden einige methodische und organisatorische Maßnahmen aufgelistet, die von den Studierenden als besonders positiv gewertet wurden:

- Aufzeigen von Berufsperspektiven
  (Arbeitsmarktstudien; Kooperation mit Ingenieurbüros: von der Projektplanung bis zum Prototypen)
- Vortragsstil *Kaleidoskop* [16]
  (lebendig-bunte Folge verständlicher, in sich abgeschlossener Einzelvorlesungen im Stil der amerikanischen *Lecture*)
- „Reader“ für alle Lehrveranstaltungen
  (Sammlung einschlägiger Fachaufsätze zu den Einzelvorträgen)
- „Handouts“
  (Kopien der Vortragsfolien, ausgearbeitete Positionspapiere usw. gegen den studentischen „Mitschreib-Zwang“)
- „Semesterapparat“
  (speziell für die Lehrveranstaltung zusammengestellte Präsenzliteratur)
- „Projektlabor“ zu IST I/II
  (für die Projektgruppen reservierte Arbeitsräume mit OOx- und CAx-Instrumentarium)
- Fragenkataloge zu IST I/II
  (zur Prüfungsvorbereitung)

Trotz aller Maßnahmen, um den studentischen Lernprozeß zu unterstützen, bedarf es didaktisch mehr: Ein Studienmodell sollte einen *Grundgedankengang* haben, einen *didaktischen Leitfaden*, der sich durch die Lehrveranstaltungen und deren Sitzungen zieht:

> „Auf diese Weise werden die einzelnen Wissensgebiete nicht nur additiv aneinandergereiht. Sie bilden vielmehr einen sinnvollen Zusammenhang, der, wenn er transparant wird, die Einsicht der Studierenden in die Sache fördert.“ [16]

In diesem Sinne orientieren wir die Lehrinhalte des Systementwurfs an fächerübergreifenden Leitbildern.

2 Die 7 Leitbilder

Bild 4 nennt die Leitbilder und zeigt ihre Einordnung im Studienmodell „Informatik-Systemtechnik“. Wir unterteilen sie entsprechend ihrer Herkunft in (a) Leitbilder der Wirtschaftspraxis und (b) Leitbilder der universitären Forschung. Das Bild 4 suggeriert *gelehrte* und *geprobte* Leitbilder: *individuell* gelehrt werden sie alle, *gemeinsam* geprobt werden sie nur im Projektlabor.

2.1 Leitbilder der Praxis

Leitbild: „System-Kompetenz“

Es gibt keine isolierten Systeme, jedes System steht im *Kontext* anderer. Der Entwerfer hat stets das „eingebettete“ Ziel vor Augen: die geforderte Systemfunktion, die vorgesehene Umgebung und die zu gestaltende Schnittstelle zwischen System

und Umgebung. Die Systemfunktion besitzt technische und wirtschaftliche Aspekte: die Machbarkeit und den unternehmerischen Gewinn. Die Systemumgebung hat im allgemeinen ökologische Aspekte: die Verträglichkeit der Technikfolgen. Und die Systemschnittstelle hat ergonomische Aspekte: die Akzeptanz durch den Benutzer. Der Systementwerfer denkt und handelt im Kontext von Funktion, Umgebung und Schnittstelle. Die ganzheitliche Systemsicht entscheidet über die *Qualität* sowohl des Entwurfsprozesses als auch des Entwurfsprodukts [4].

Welche Entwürfe setzen System-Kompetenz voraus? Informatiksysteme [23]. Das sind *rechnergestützte* komplexe und hybride Artefakte. Sie sind komplex, da sie sich aus verteilten Systemen zusammensetzen, die wiederum verteilte Systeme sein können. Sie sind hybrid, da sie sowohl Software als auch digitale und analoge Hardware umfassen: Rechner und Peripherie, Datenbanken, Sensoren und Aktoren. Die Systemteile können geographisch verteilt oder lokal eng gekoppelt sein. Beispiele sind Leitsysteme für Gebäudekomplexe oder Fertigungssteuerungen.

System-Kompetenz entwickelt sich erst in langjähriger Berufserfahrung. Das Bewußtsein hierfür zu schaffen, ist aber Aufgabe der Ingenieurausbildung. Das Ideal ist der „systemdenkende" Generalist; die Sicht des Modul-Spezialisten ist für den Systementwurf unzureichend [24].

Leitbild: „Schnittstellen-Kompetenz"

Der Systementwurf ist stets *mehrphasig*, auch wenn die „Entwurfsautomatisierung" die späten Phasen transparent hält. Die Bedeutung der Phasenschnittstellen verschiebt sich in Richtung der *Spezifikation*: Das *Lastenheft* spezifiziert die Systemanforderungen und die Gebrauchsfälle. Es ist das Ergebnis der Anwendungsmodellierung und die vertragliche Schnittstelle zum Auftraggeber. Das *Pflichtenheft* spezifiziert Art und Umfang der technischen Realisierung. Es ist das Ergebnis des Hardware-Software-(Co-)Designs und eventuell die vertragliche Schnittstelle zu einem Dienstleister oder Zulieferer. Die Phasen der technischen Realisierung erfolgen in möglichst flexiblen und wiederverwendbaren Entwurfsmedien: in Entwurfsdatenbanken und programmierbaren Hardware-Strukturen.

Schnittstellen sind stets Orte des Methoden- und Werkzeugwechsels. Schnittstellen-Kompetenz ist die *Synergie* aus Methoden- und Werkzeug-Kompetenz. Das Wissen um die Spezifikation der Schnittstellen — formale Hard-und Software-Beschreibungen, Austauschformate, Datenbankschemata — zeichnet den Systementwerfer vor dem Modulentwerfer aus. Der Systementwerfer orientiert sich schnittstellenübergreifend: er nimmt die Belange der späten Phasen vorweg und führt sie als *Korrektiv* in die frühen Phasen zurück. Antizipation und Rückkopplung von Entwurfsentscheidungen bestimmen den Entwicklungszyklus. In allen Phasen des Systementwurfs treten *Modellmonopole* [17] auf, die es *dialogisch* im Team zu überwinden gilt.

Leitbild: „Team-Kompetenz"

Neben den technischen Schnittstellen beeinflussen besonders die zwischenmenschlichen „Schnittstellen" die Entwurfsqualität und -produktivität: Ein arbeitsteiliger Prozeß setzt kooperative und kommunikative Fähigkeiten voraus — *soft skills*. Die soziale Kompetenz des Systementwerfers ist so wichtig wie seine fachliche. Kundenbefragungen, Projektbesprechungen, Delegation von Aufgaben, Auswahlgespräche mit Dienstleistern und Zulieferern, Präsentation von Ergebnissen und andere „zwischenmenschliche Situationen" im Systementwurf fordern die *nicht-technische* Kompetenz [4, 7, 15, 17].

Persönlichkeitstraining liegt außerhalb der Ingenieurausbildung. Das Bewußtsein für Defizite und erste Erfahrungen mit gruppendynamischen Prozessen sind aber durchaus vermittelbar: Selbstorganisierte studentische „Miniprojekte" in Kooperation mit Ingenieurbüros (von der Projektplanung bis zum Prototypen) sind hier das didaktische Vehikel. Die Bedeutung von *Projektgruppen* ist in den Lehrplänen der Informatik bereits fest verankert. Übungen und Praktika zum Systementwurf sollten grundsätzlich in Projektgruppen organisiert werden.

2.2 Leitbilder der Forschung

Leitbild: „Objektorientiertes Entwerfen“

Die Didaktik komplexer Entwürfe fordert eine begriffliche und zugleich technische *Grundeinheit des Entwerfens*. Gefordert ist die Kapselung von Struktur, Verhalten und Beschränkung. In diesem dreifachen Aspekt der Kapselung korrespondiert der *zellbasierte* Hardware-Entwurf mit dem *objektorientierten* Software-Entwurf. Beide Strategien zu integrieren, würde den hybriden Systementwurf aus Soft- und Hardware-Komponenten methodisch und didaktisch vereinheitlichen [9]. Die Objektorientierung führt die zellbasierte Entwurfsstrategie konzeptionell weiter: Sie durchdringt alle Phasen des Systementwurfs — von der Spezifikation bis zur Realisierung [20].

Leitbild: „Mustergeführtes Entwerfen“

Es gibt nur wenige Universal-Strategien für den technischen Entwurf — Entwurfsmuster zählen zu den bedeutendsten. Wie kann der unerfahrene Entwerfer die Erfahrung des Experten nutzen? Wie läßt sich Erfahrung (= erprobtes Wissen) für die Wiederverwendung dokumentieren? Durch Muster. Musterorientierte Entwurfsstrategien gibt es seit langem für den Gebäude-Entwurf und die Städteplanung [1], seit kurzem für die Software-Technik [5,8] und demnächst für den Schaltungsentwurf. Erfahrungstransfer und Erfahrungsersatz sind durch *hypertext*-gestützte „Musterkataloge“ systematisierbar.

Überdies fördert der mustergeführte Entwurf die Motivation des Laien-Entwerfers: Entwurfsmuster als praxiserprobte Einheiten aus Problemstellung und Lösungsansatz verkürzen die Lernphase und schaffen didaktisch wichtige Erfolgserlebnisse. Als *Extrakt* wissenschaftlicher Arbeiten (technischer Diplomarbeiten und Dissertationen) gewähren Entwurfsmuster das schnelle Erfassen des Forschungsstands einer Fachgruppe.

Leitbild: „Integratives Entwerfen“

Ein pragmatisches Konzept, die Entwurfskomplexität zu reduzieren, heißt *CAD-Framework*: Standardisierte Infrastrukturen für Entwurfswerkzeuge, die auf einem *gemeinsamen* Datenbestand operieren, gewährleisten „Integrität durch Integration“. Integratives Entwerfen rückt 2 Facetten der *Informationsverarbeitung* in den Mittelpunkt: 1. die Datenrepräsentation, das Modellieren und Schematisieren der Entwurfsinformation, und 2. die Datenverwaltung, das konsistente Fortschreiben und Abfragen des Datenbestands. Damit gehen zwei Gegenstände der Modellierung einher: *deskriptive* Austauschformate und *prozedurale* Werkzeugschnittstellen. Der deskriptive Aspekt umfaßt die statischen Strukturen der Entwurfsobjekte. Hier bietet die Informationsmodellierung (zum Beispiel in der Objekt-Sprache Express) die Strukturmittel, um den Informationsgehalt semi-formal zu beschreiben. Der prozedurale Aspekt zielt auf die Interaktion zwischen den Entwurfswerkzeugen. Da Transferdienste auf der Datei-Ebene zu „grobkörnig“ sind, müssen die Werkzeuge über *semantische Datenmodelle* integriert werden.

Der Systementwerfer sollte also die Probleme *jenseits* der Entwurfswerkzeuge gut kennen [18]. Dies setzt *datenbanktechnisches* Wissen voraus: Management der Entwurfstransaktionen und Versionierung der Entwurfsobjekte.

Leitbild: „Flexibles Entwerfen“

Redesigns sind prinzipiell unvermeidbar, denn innovative komplexe Entwürfe folgen dem Prinzip von Versuch und Irrtum. Redesigns sind aber kostspielig, wenn *masken*programmierbare Schaltungen (ASIC) eingesetzt werden. *Feld*programmierbare Schaltungen (FPGA) schaffen hier Abhilfe: Sie sind das Hardware-Pendant zum ausführbaren Zielcode eines Rechnerprogramms. Kommen weitere „weiche“ Hardware-Strukturen hinzu, wie programmierbare Analogschaltungen und Verbindungselemente, so steht ein *durchgängig* feldprogrammierbares Zielmedium zur Verfügung. Flexibles Entwerfen ist dann

von der Systemebene bis zur Hardware-Implementierung möglich: „Rapid Prototyping“ analog zur Software-Technik [2, 22].

2.3 Evaluation der Leitbilder

Die Leitbilder 1 bis 3 im Bild 4 entspringen zahlreichen Arbeitsmarktstudien für Informatik-Absolventen (vor allem [4, 7]). Die Leitbilder 4 und 6 waren Gegenstand eines BMFT-Projekts [19] und die Leitbilder 5 und 7 sind aktueller Forschungsgegenstand unserer Fachgruppe. Der reguläre Vorlesungszyklus „Informatik-Systemtechnik“ soll im Wintersemester 1996/97 beginnen. Als Vorläufer wird seit dem Wintersemester 1994/95 die Lehrveranstaltung OOS I/II „Objektorientierter Systementwurf“ angeboten [21], wobei das Laborkonzept „Projektlabor“ bislang auf die „Projektgruppe“ reduziert ist, und somit das in [10] beschriebene IST-Werkzeugszenario noch nicht zum Tragen kommt.

Die OOS-Vorträge und Seminarübungen wurden von den Studierenden anhand der Fragebögen aus [16] evaluiert. Im Vordergrund der Bewertung stand nicht so sehr die Präsentation der Lehrinhalte, sondern vielmehr die *Motivation* für die aktive Gestaltung des eigenen Studiums. Tenor der Evaluation: eine Mehrheit hat das Konzept des Studienmodells angenommen und will die aufgezeigte Vertiefungsrichtung „Informatik-Systemtechnik“ weiter verfolgen.

Literatur

[1] Alexander, C. et al.: A Pattern Language. Oxford University Press, New York, 1977

[2] Boemo, E. et al.: Field-Programmable Logic in Education: a Case Study. In: [12], S. 452–457

[3] Booch, G.: Object-Oriented Analysis and Design: with Applications. Benjamin/Cummings Publishing Company, Redwood City, Kalifornien, 2. Aufl., 1994

[4] Brodbeck, F. C.; Frese, M. (Hrsg.): Produktivität und Qualität in Software-Projekten: Psychologische Analyse und Optimierung von Arbeitsprozessen in der Software-Entwicklung. Oldenbourg-Verlag, München, 1994

[5] Coplien, J. O.: Software-Entwurfsmuster. Objekt-Spektrum, Jg. 1, H. 6, 1995, S. 52–58

[6] Clauser, C.: Kritische Analyse eines Paradigmenwechsels in der Software-Industrie — dargestellt am Beispiel des objektorientierten Ansatzes. Diplomarbeit, Universität Mannheim, 1995

[7] Dostal, W.: Berufsbilder in der Informatik. In: Informatik-Spektrum, Jg. 18, H. 3, 1995, S. 152–162

[8] Gamma, E. et al.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Reading, 1995

[9] Glunz, W.: Hardware-Entwurf auf abstrakten Ebenen unter Verwendung von Methoden aus dem Software-Entwurf. Dissertation, Paderborn, 1994

[10] Klose, B. et al.: E.I.S.-Labore: eine Siegener Chronik. Posterbeitrag zum 7. E.I.S.-Workshop, TU Chemnitz-Zwickau, 1995

[11] Luft, A. L.: Informatik als Technik-Wissenschaft: Eine Orientierungshilfe für das Informatik-Studium. BI-Wissenschaftsverlag, Mannheim, 1988

[12] Luk, W.; Moore, W. R. (Hrsg.): More FPGAs. Abingdon EE&CS Books, Oxford GB, 1994

[13] N. N.: Interview: Professor Peter Lockemann und Dr. Wilhelm Denz zur Situation der Informatik-Ausbildung: Den stabilen Arbeitsplatz wird es nicht mehr geben. Informatik-Spektrum, Jg. 18, H. 2, 1995, S. 111–113

[14] N. N.: Unterlagen zum 1. Treffen der Hochschullehrer mit dem Arbeitsschwerpunkt Technische Informatik. Schloß Dagstuhl, 21.–22. April 1994

[15] N. N.: Qualifikation von Hochschulabsolventen. In: Forschung & Lehre, 8/1995, S. 466

[16] N. N.: Handbuch Hochschullehre: Informationen und Handreichungen aus der Praxis für die Hochschullehre. Raabe-Fachverlag, Loseblatt-Ausgabe, Bonn, 1994

[17] Pasch, J.: Software-Entwicklung im Team: Mehr Qualität durch das dialogische Prinzip bei der Projektarbeit. Springer-Verlag, Berlin et al., 1994

[18] Quibeldey-Cirkel, K.: CAD-Frameworks: Die Probleme jenseits der Entwurfswerkzeuge. In: Mikroelektronik, Jg. 7, H. 2, 1993, S. 72–76

[19] Quibeldey-Cirkel, K.: CAD-Frameworks: Ergebnisse des DASSY-Projekts. In: CAD-CAM Report, Jg. 12, H. 8, 1993, S. 16–23

[20] Quibeldey-Cirkel, K.: Das Objekt-Paradigma in der Informatik. Teubner-Verlag, Stuttgart, 1994

[21] Quibeldey-Cirkel, K.: Quo vadis, Informatik? Aspekte einer objektorientierten Entwurfslehre. In: Objekt-Spektrum, Jg. 2, H. 1, 1995, S. 30–36

[22] Schubert, E. et al.: The Use of FPGAs for Educational Purposes in VLSI Microprocessor Design. In: [12], S. 458–465

[23] Schweizer, G.; Thomé, B.: Informatik-Systemtechnik (ECBS): Gedanken zu einer Disziplin. Informatik-Spektrum, Jg. 16, 1993, S. 215–221

[24] Weinberg, G.: Systemdenken und Software-Qualität. Hanser-Verlag, München, 1994

[25] Yourdon, E.: Decline and Fall of the American Programmer. Prentice Hall, 1992