Mumm, Harald

Working Paper

Softwarearchitektur eines Fahrrad-Computer-Simulators

Wismarer Diskussionspapiere, No. 17/2005

Provided in Cooperation with:
Hochschule Wismar, Wismar Business School

Suggested Citation: Mumm, Harald (2005) : Softwarearchitektur eines Fahrrad-Computer-Simulators, Wismarer Diskussionspapiere, No. 17/2005, Hochschule Wismar, Fachbereich Wirtschaft, Wismar

This Version is available at:
https://hdl.handle.net/10419/23325

Hochschule Wismar
University of Technology, Business and Design
Fachbereich Wirtschaft

Hochschule Wismar
University of Technology, Business and Design
Faculty of Business

Harald Mumm

Softwarearchitektur eines Fahrrad-Computer-Simulators

Heft 17 / 2005

W D P

Wismarer Diskussionspapiere / Wismar Discussion Papers

Der Fachbereich Wirtschaft der Hochschule Wismar, University of Technology, Business and Design bietet die Präsenzstudiengänge Betriebswirtschaft, Management sozialer Dienstleistungen, Wirtschaftsinformatik und Wirtschaftsrecht sowie die Fernstudiengänge Betriebswirtschaft, International Management, Krankenhaus-Management und Wirtschaftsinformatik an. Gegenstand der Ausbildung sind die verschiedenen Aspekte des Wirtschaftens in der Unternehmung, der modernen Verwaltungstätigkeit im sozialen Bereich, der Verbindung von angewandter Informatik und Wirtschaftswissenschaften sowie des Rechts im Bereich der Wirtschaft.
Nähere Informationen zu Studienangebot, Forschung und Ansprechpartnern finden Sie auf unserer Homepage im World Wide Web (WWW): http://www.wi.hs-wismar.de/.

Herausgeber: Prof. Dr. Jost W. Kramer
Fachbereich Wirtschaft
Hochschule Wismar
University of Technology, Business and Design
Philipp-Müller-Straße
Postfach 12 10
D – 23966 Wismar
Telefon: ++49/(0)3841/753 441
Fax: ++49/(0)3841/753 131
e-mail: j.kramer@wi.hs-wismar.de

Vertrieb: HWS-Hochschule Wismar Service GmbH
Phillipp-Müller-Straße
Postfach 12 10
23952 Wismar
Telefon:++49/(0)3841/753-574
Fax: ++49/(0)3841/753-575
e-mail: info@hws-startupfuture.de
Homepage: www.hws-startupfuture.de

ISSN 1612-0884

JEL-Klassifikation Z00

Inhaltsverzeichnis

1. Einleitung

Das Ziel dieser Arbeit ist die Darstellung der Softwarearchitektur des im Heft 6 (2005) der Wismarer Diskussionspapiere (s. [1]) vorgestellten Simulators für einen einfachen Fahrradcomputer. Diese Darstellung kann für alle diejenigen interessant sein, die Simulatoren für Geräte entwickeln, in denen Neumann-Maschinen zur Anwendung kommen. Simulatoren werden z.B. für Produktstudien eingesetzt, um Entwicklungskosten einzusparen. Neumann-Maschinen finden in fast allen modernen Produkten, wie z.B. in Chipkarten, in Telefonen, in Steuerungsteilen für Heizungsanlagen, in Digitalkameras uvm. ihren Einsatz. Im Rahmen des aktuellen Forschungsthemas der Wirtschaftsinformatik „Pervasive Computing“ werden Neumann-Maschinen zukünftig auch in jedem Füllfederhalter und in jeder Zigarette zu finden sein.

Die Software wurde unter Java 1.2 erstellt und arbeitet mit einer graphischen Bedienoberfläche. Da auch die Möglichkeit bestehen sollte, die Software über das Internet zu bedienen, wurde sie als Java-Applet realisiert.

Das Dokument `http://www.wi.hs-wismar.de/~mumm/ergoapp2/Ergo.html` beinhaltet den Aufruf des Applets.

In der Arbeit finden Diagrammtechniken der Unified Modelling Language (UML) Verwendung, die als bekannt vorausgesetzt werden. Sollte das nicht der Fall sein, wird das Buch von Perdita Stevens (s.[4]) empfohlen, in dem auch noch weitere Literaturhinweise zu finden sind.

Natürlich wird durch den Einsatz der UML mit „Kanonen auf Spatzen geschossen“, denn diese Sprache wurde für große Softwaresysteme, wie z.B. für das Gepäckleitsystem eines großen internationalen Flughafens oder für die Rettungsleitstelle einer Großtadt entwickelt. Aber ihr Einsatz ist unter dem Aspekt der Lehre trotzdem gerechtfertigt, weil sie während des Studiums der Wirtschaftsinformatik in kurzer Zeit relativ abstrakt behandelt wird und geeignete Beispiele dem Verständnis nur dienen können.

Dies Arbeit ist implementationsorientiert (bottom-up) aufgebaut. Es wird nicht der lange Weg des Entwurfes beschrieben, sondern ausgehend vom Ergebnis werden gewisse Abstraktionen erläutert. Diese Darstellungsart wurde gewählt, um die Darstellung kurz zu halten. Am Anfang steht eine Beschreibung der Pakete der Implementierung. Anschließend werden ihre Klassen und deren Beziehungen und zum Schluss einige wenige Klassen im Detail erläutert.

In [1] wird eine Uhr wie üblich als Bestandteil eines Unterbrechungsprozessors angesehen. Von dieser Sichtweise wird hier abgewichen und die Uhr als separates Gerät betrachtet. Der Grund hierfür liegt in einer fehlenden Idee für eine Implementierung der Variante aus [1].

2. Die Aufteilung der Software

Das Beispiel Fahrradcomputer hatte von Anfang an nur exemplarischen Charakter, da Computer-Chips in fast allen modernen technischen Geräten zum Einsatz kommen.

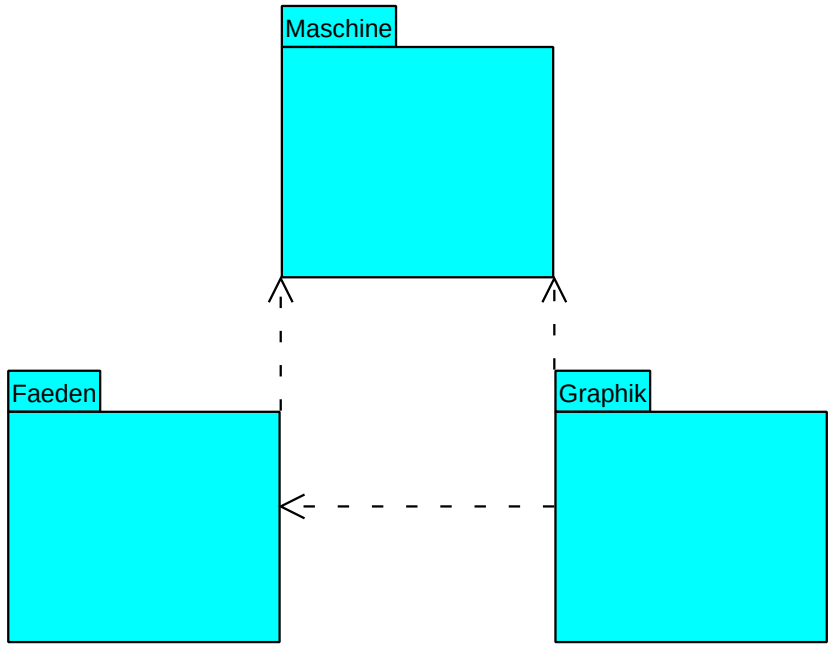
Damit die Software später auch für andere ähnlich gelagerte Beispiele genutzt werden kann, ist ihr modularer Entwurf sehr wichtig. Die Software wurde deshalb auf ca. 20 Klassen und diese wiederum auf folgende drei Pakete aufgeteilt: Graphik, Faeden[1] und (Neumann-)Maschine. Eine genaue Inhaltsbeschreibung der Pakete erfolgt im Verlaufe dieser Arbeit. Im Paket „Graphik“ sind alle Klassen der auf dem Bildschirm sichtbaren Objekte enthalten. Das Paket „Maschine“ widerspiegelt den Aufbau der Neumann-Maschine Paula (siehe [2])aus einem Kern mit Registern und einem Bus, einem Steuersignalprozessor mit einem Mikroprogrammspeicher(MS), einer arithmetisch-logischen Einheit (ALE) und einem Hauptspeicher (HS). Zur Kommunikation mit der Außenwelt dient ein Unterbrechungsprozessor (UntProz), der ebenfalls Bestandteil des Paketes „Maschine“ ist. Im Paket „Faeden“ liegen alle Threads (leichgewichtige Prozesse), die zur Simulation der Geräte benötigt werden. Im Einzelnen sind es: RunNabe für das Fahrrad, RunHerz für den Pulsfühler, RunUhr für die interne Uhr und RunPaula für die Neumann-Maschine. Im Paket „Graphik“ liegen die Klassen für die auf dem Bildschirm sichtbaren Objekte: NabePanel, HerzPanel, UhrPanel und PaulaPanel. (Unter einem Panel versteht man in Java 1.2 einen rechteckigen Bereich auf dem Bildschirm.)
Die Pfeile in der Abbildung 1 stellen keine Transportpfeile sondern Verweispfeile dar. Die Klassen aus dem Paket „Maschine“ verweisen auf keine weiteren Klassen aus anderen Paketen. Die Im- und Export-Beziehungen zwischen den Paketen werden in der Abbildung 1 veranschaulicht. Ein Pfeil, der von einem Paket auf ein anderes zeigt, symbolisiert, dass Klassen aus dem Paket, auf das der Pfeil zeigt, in Klassen aus dem Paket, von dem der Pfeil ausgeht, importiert werden.

Das Paket Maschine besteht aus sechs wichtigen Klassen und zwei Hilfsklassen, das Paket Faeden aus vier Klassen und das Paket Graphik aus acht Klassen. Alle Pakete zusammen bestehen aus ca. 880 Quelltextzeilen.

Aus der Abbildung 1 ist zu erkennen, dass im Paket „Maschine“ nur Klassen enthalten sind, die auf keine weiteren Pakete verweisen. Das ist nicht verwunderlich, handelt sich dabei doch um die Klassen, die zur Simulation der Neumann-Maschine von Hans Roeck (s.[2]) eingesetzt werden. Bei Software mit graphischer Bedienoberfläche stellt sich i. allg. immer die Frage, wie Aktionen hinter Bedienknöpfen (Buttons) oder Menueoptionen implementiert werden können, damit die Software durch die Abarbeitung dieser Aktionen nicht für längere Zeit blockiert wird. Hier werden die Aktionen in Java-Threads (leichtgewichtige Prozesse) ausgeführt. Die dazugehörigen Klassen liegen exklusiv im Paket „Faeden“. Die eigentlichen Komponenten für die Bedienung der Software wurden im Paket „Graphik“ zusammengefasst.

[1] Faeden ist die deutsche Übersetzung des englischen Wortes „threads“, womit leichtgewichtige Prozesse gemeint sind.

Abbildung 1: Die Aufteilung der Software in drei Pakete



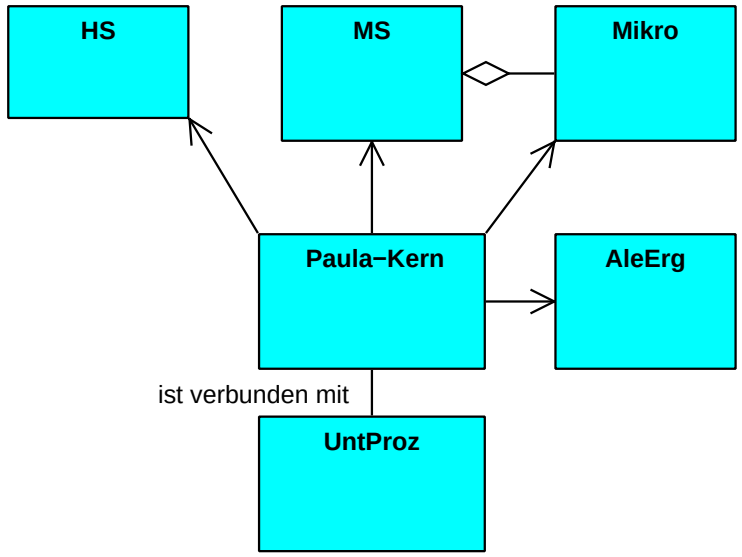
3. Das Paket Maschine

Die Maschine Paula aus [2] ist eine Mikroprogramm-gesteuerte Registermaschine u.a. mit einem Hauptspeicher und einer Arithmetisch-Logischen-Einheit (ALE). Zur Modellierung dieser Hardware werden die Klassen „HS“ (Hauptspeicher), „MS“ (Mikrospeicher), „Mikro“ (Mikroinstruktion), „Paula“ und „AleErg“ (ALE-Ergebnis) eingesetzt. Zur Kommunikation mit der Außenwelt wird der Unterbrechungsprozessor „UntProz“ verwendet.
Die einfachen Pfeile in der Abbildung 2 symbolisieren gerichtete Assoziationen der Klasse Paula zu den Klassen HS, MS, Mikro und AleErg. Die Linie zwischen den Klassen UntProz und Paula steht für eine ungerichtete Assoziation zwischen zwei Klassen.

Assoziationen stehen konzeptionell für Verbindungen beliebiger Art, die zum Austausch von Nachrichten genutzt werden. Erfolgt das Versenden von Nachrichten nur in einer Richtung, so spricht man von gerichteten Assoziationen, die graphisch durch einen Pfeil (im Gegensatz zu einer Strecke bei ungerichteten Assoziationen) dargestellt werden.

Eine Aggregation ist eine spezielle Assoziation. Diese Beziehungsart wird verwendet um anzuzeigen, dass ein Objekt einer Klasse Teil eines Objktes einer anderen Klasse ist. Man spricht auch von einer Teil-Ganzes-Beziehung. Die Raute befindet sich dabei immer am Ende des Ganzen und nicht beim Teil. Der Mikro-

Abbildung 2: Ausgewählte Klassen im Paket Maschine



speicher (MS) besteht aus einer Reihe von Mikro-Befehlen (Mikro), weshalb diese Beziehungsart zwischen diesen beiden Klassen angebracht ist.

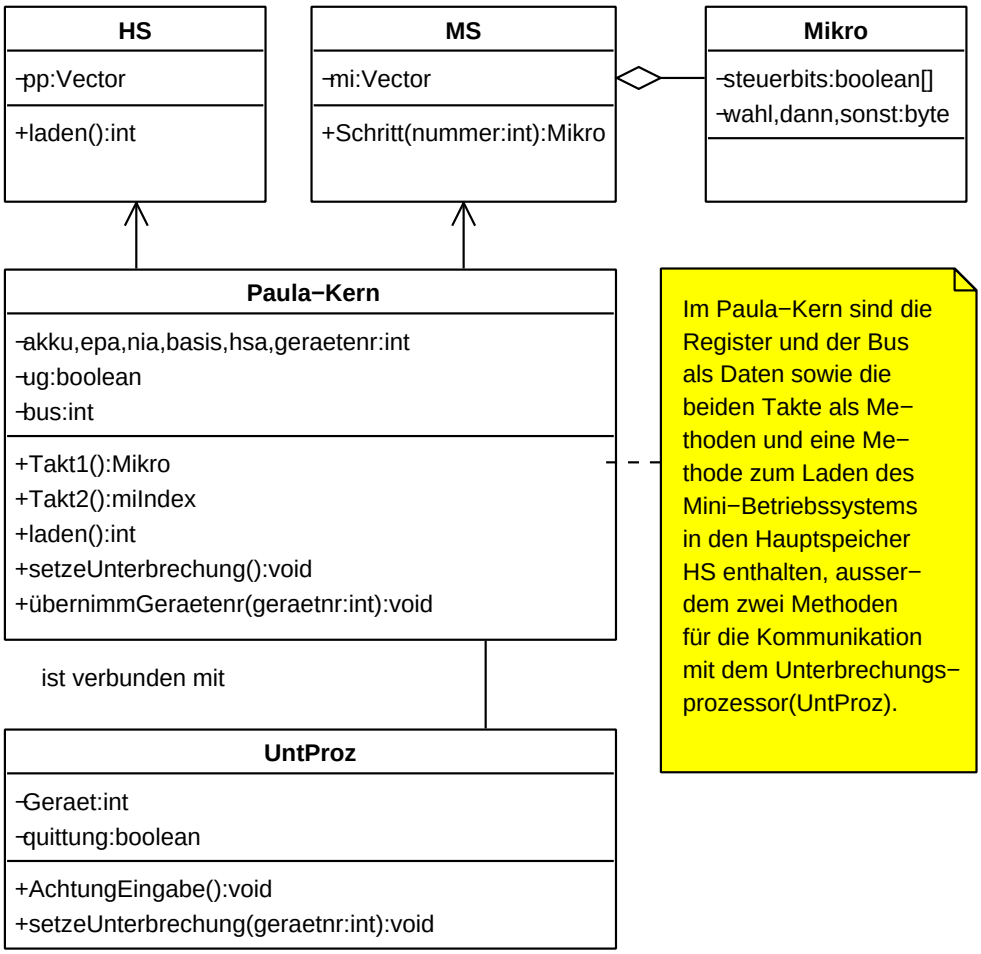
Es sollen jetzt Methoden und Instanzvariablen dieser Klassen vorgestellt werden. Die beiden wichtigsten Methoden in der Klasse „Paula-Kern“ heißen Takt1 und Takt2. Die Methode Takt1 holt die nächste Mikroinstruktion und die Methode Takt2 führt diese Mikroinstruktion dann aus und liefert die Adresse der sich anschließenden Mikroinstruktion zurück.

Einige wichtige Instanzvariablen und Methoden der Klasse Maschine sind in der folgenden Abbildung zusammengestellt worden.
Die Methode setzeUnterbrechung in der Klasse „Paula-Kern“ wird immer dann aufgerufen, wenn der Unterbechungsprozessor die Neumann-Maschine unterbrechen will. Durch die Nachricht übernimmGerätenr wird die Neummann-Maschine veranlasst, die Gerätenummer des auslösenden Gerätes vom Unterbrechungsprozessor einzulesen.

Die Geräte Uhr, Nabe und Pulsfühler senden die Nachricht setzeUnterbrechung an das Objekt der Klasse UntProz (Unterbrechungsprozessor).

Die Klasse HS (Hauptspeicher) besitzt eine pseudodynamische Instanzvariable pp zur Aufnahme der Software des Fahrradcomputers. Die Klasse MS (Mikrospeicher) besitzt eine pseudodynamische Instanzvariable mi zur Aufnahme des Mikroprogrammes. Die Objekte der Klasse „Mikro“ repräsentieren einen Mikroprogrammschritt nach [2].

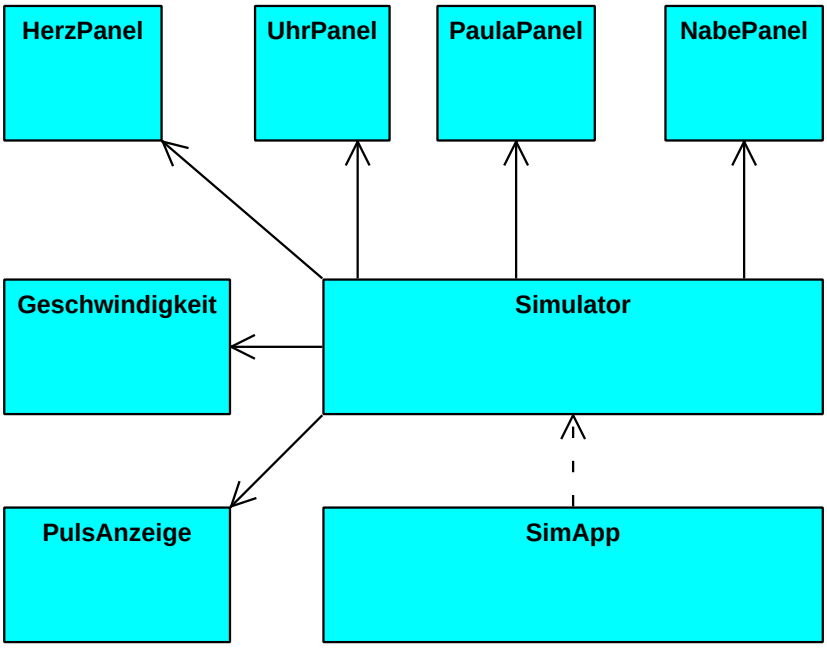
Abbildung 3: Einige Klassen im Paket Maschine im Detail



4. Das Paket Graphik

Die graphische Benutzungsoberfläche des Simulationsprogrammes wurde mit dem Java-Swing-Paket entwickelt. Die Klasse JPanel dient dabei als Basisklasse für alle graphischen Elemente. Aus der Sicht der Anwendung handelt es sich hierbei um einen rechteckigen Bereich, in dem weitere spezielle Elemente wie Schaltknöpfe, Texte sowie Ein- und Ausgabefelder platziert werden können. Die folgende Abbildung gibt einen Überblick über die Klassen des Paketes Graphik.
Die Klasse Simulator ist eine Aggregation der sechs Klassen PaulaPanel, UhrPanel, HerzPanel, NabePanel, PulsAnzeige und Geschwindigkeit. Die Klasse SimApp macht aus der Klasse Simulator ein Java-Applet. Die sechs Klassen widerspiegeln den tatsächlichen stofflichen Aufbau des Fahrradcomputers aus der eigentlichen Neumann-Maschine, dem Herzfühler, der Nabe und der Uhr sowie den beiden Anzeigen für den Puls und die Geschwindigkeit. Die Objekte der Klassen HerzPanel und NabePanel besitzen jeweils ein Textelement, zwei Knöpfe Slow

Abbildung 4: Die Klassen im Paket Graphik



und Fast zur Veränderung der Frequenz und jeweils ein Ausgabefeld zur Anzeige der gezählten Unterbrechungen.

Diese Anzeige ist deshalb besonders wichtig, weil mit ihrer Hilfe die Richtigkeit der berechneten Werte für den Puls und die Geschwindigkeit kontrolliert werden kann.

Die Uhrticks werden nur angezeigt aber ihre Frequenz ist für den Bediener nicht veränderbar.

Das Objekt der Klasse PaulaPanel enthält einen Schaltknopf zum Starten der Neumann-Maschine sowie ein Textfeld, in dem nach dem Start der Text „Paula is working!“ erscheint. Die Taktfrequenz der Neumann-Maschine kann leider nicht weiter erhöht werden, weil sie bereits aufgrund der gewählten Architektur ausgereizt ist.

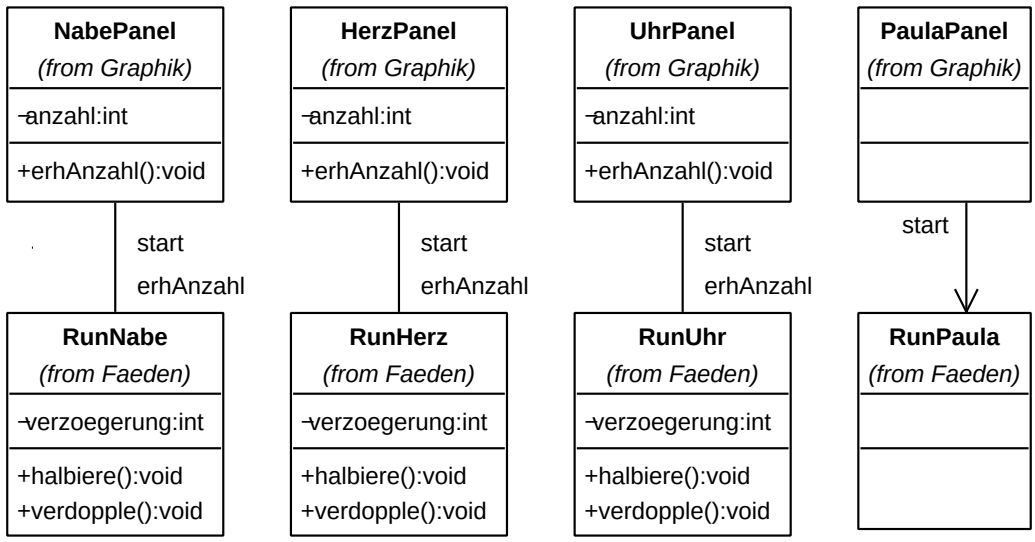
Die vier Klassen PaulaPanel, UhrPanel, HerzPanel und NabePanel stehen in einer gerichteten Assoziation zu jeweils einer Spezialisierung der Klasse JavaThread mit den Namen RunPaula, RunUhr, RunHerz und RunNabe.

Der Zugriff auf die Threads erfolgt zum Zwecke des Startens der Thread-Aktivitäten.

Nach dem Start laufen die Threads in einer Endlosschleife und sorgen sowohl für die Unterbrechungen (RunUhr, RunHerz und RunNabe) als auch für die Verarbeitung dieser Unterbrechungen (RunPaula).

Die folgende Abbildung veranschaulicht die hier vorhandenen Beziehungen zwischen Panels und Threads.

Abbildung 5: Assoziationen von Klassen aus dem Paket Graphik zu Klassen aus dem Paket Faeden



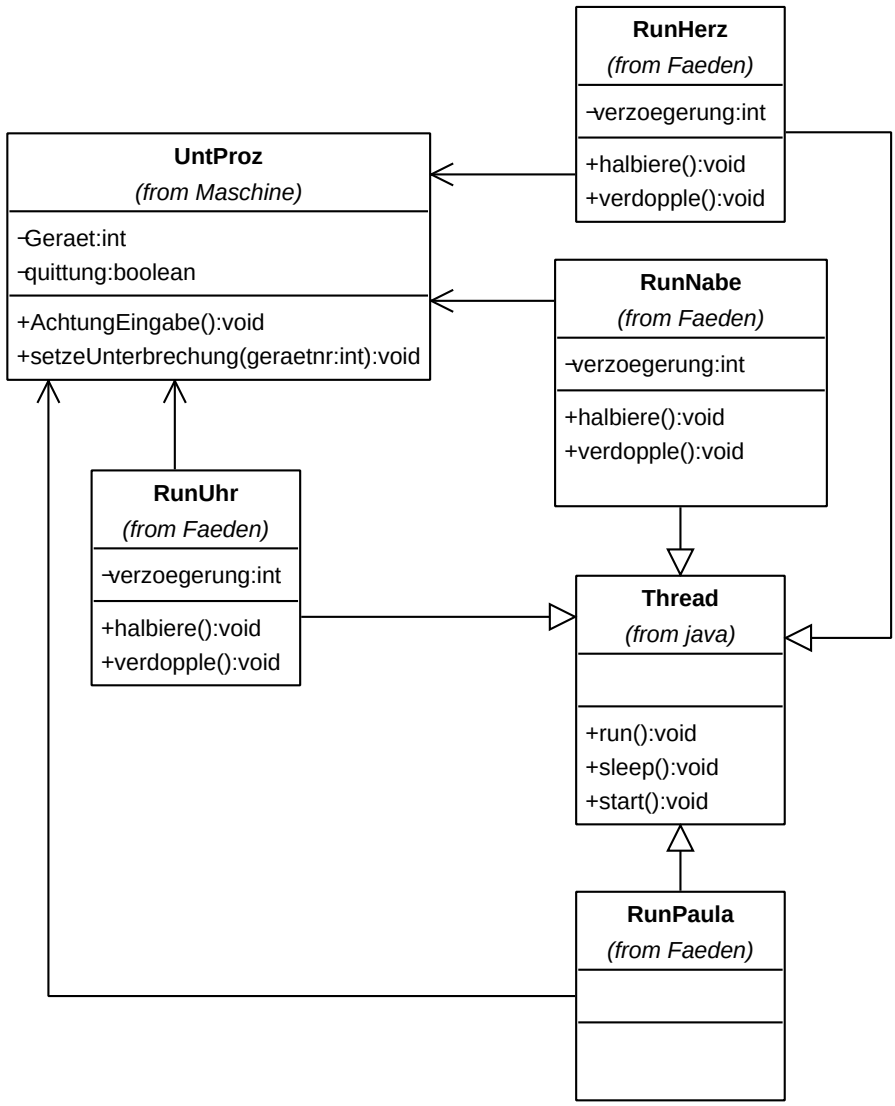
Die drei linken Assoziationen sind ungerichtet, denn nach dem Start der vier beteiligten Threads RunNabe, RunHerz, RunUhr und RunPaula durch die Panels müssen die Panels in regelmäßigen Abständen zur richtigen Anzeige der Anzahl der ausgelösten Unterbrechungen durch die Threads aktualisiert werden. Das Objekt vom Typ NabePanel startet z.B. den Thread RunNabe und das Objekt vom Typ RunNabe erhöht nach jeder simulierten Unterbrechung die Anzahl aller bisherigen Unterbrechungen vom Gerät Nabe um Eins indem es die Nachricht „erhoehe“ an das Panel NabePanel schickt.

Die Instanzvariable „verzoegerung“ in den Klassen RunNabe, RunHerz und RunUhr dient der Abspeicherung eines Wertes für die Schlafenszeit eines Threads. Die Methoden „halbiere“ und „verdoppple“ verändern diesen Wert. In einer ersten Fassung verfügte auch der Thread RunPaula über diese Instanzvariable und die beiden Methoden halbiere und verdopple. Es stellte sich aber sehr schnell heraus, dass eine Verzögerung der Neumann-Maschine unzweckmäßig ist, weil man zur Unterbrechungsbehandlung eine maximale Taktfrequenz benötigt. Ansonsten tritt schon während der Unterbrechungsbehandlung die nächste Unterbrechung auf, was zu einem „Verschlucken“ dieser Unterbrechung führen würde, weil die Neumann-Maschine während der Unterbrechungsbehandlung nicht unterbrechbar ist.

5. Das Paket Faeden

Die Geräte Pulsfühler, Nabe, Uhr und Neumann-Maschine werden durch Threads simuliert. Das ist einerseits notwendig, damit die Anwendung nicht blockiert, und andererseits zweckmäßig, weil die Geräte Uhr, Nabe und Pulsfühler nur in gewissen Abständen Unterbrechungswünsche signalisieren. Das „Schlafenlegen“ eines Threads wird über den Aufruf der Methode sleep realisiert. Die folgende Abbildung gibt einen Überblick der Klassen im Paket Faeden. Gleichzeitig werden ihr Aufbau und wichtige Beziehungen zu Klassen anderer Pakete, insbesondere UntProz aus dem Paket Maschine und Thread aus dem Paket „java.lang“, dargestellt.

Abbildung 6: Die Klassen im Paket Faeden



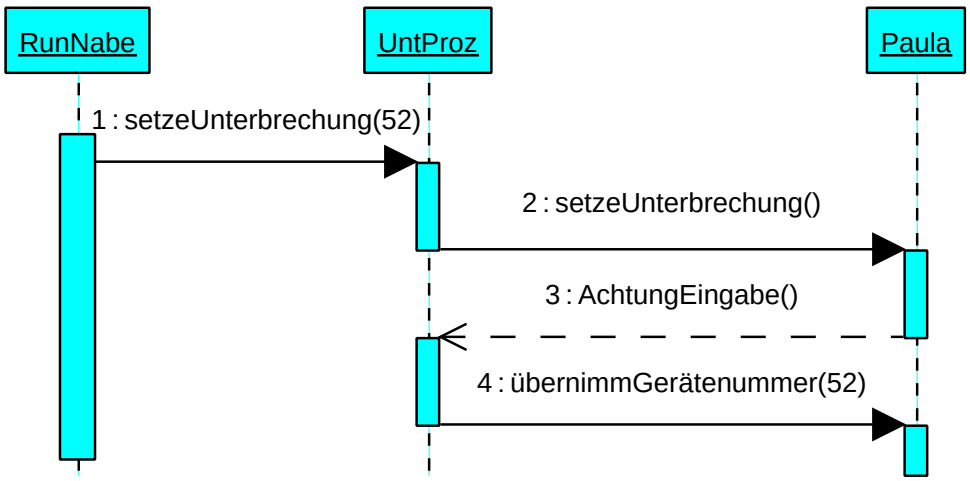
Alle Klassen des Paketes Faeden erweitern die Java-Klasse Thread.

6. Die Sequenz der Methodenaufrufe bei einer Unterbrechung

„In einem Sequenzdiagramm sind die Objekte, die an einer Kollaboration beteiligt sind, am oberen Ende einer gestrichelten Linie dargestellt. Diese Linie gibt die Zeit aus der Sicht des Objektes an: Sie ist die Lebenslinie des Objektes. Weist ein Objekt eine Aktivierung auf, wird seine Lebenslinie mit einem Rechteck bedeckt.“ (Siehe [4], S. 145)

Die Geräte Uhr, Nabe und Herzfühler lösen Unterbrechungen aus. Für sie wurden die Java-Klassen RunUhr, RunNabe und RunHerz als Erweiterung der Java-Klasse Thread definiert. Die Sequenz der Methodenaufrufe bei einer simulierten Unterbrechung wird in dem folgenden Diagramm am Beispiel des Gerätes Nabe veranschaulicht.

Abbildung 7: Die Sequenz der Methodenaufrufe bei einer Unterbrechung



In einer Endlosschleife schickt das Objekt der Klasse RunNabe nach einer gewissen Schlafenszeit (der Wert ist in der Instanzvariablen „verzoegerung“ vermerkt und wurde mit 7.500 Millisekunden initialisiert) dem Objekt vom Typ UntProz (Unterbrechungsprozessor) die Nachricht setzeUnterbrechung(52). Der Unterbrechungsprozessor signalisiert daraufhin dem Objekt vom Typ der Neumann-Maschine Paula, dass ein Unterbrechungswunsch vorliegt. Das geschieht über die Nachricht setzeUnterbrechung. Paula antwortet mit der Nachricht „AchtungEingabe“, worauf der Unterbrechungsprozessor der Neumann-Maschine durch die Nachricht übernimmGerätenummer mitteilt, welches Gerät die Unterbrechung gewünscht hat. In diesem Fall hat das Gerät Nabe völlig willkürlich die Nummer 52.

7. Die Bedienung der Software

Die Bedienung des Simulatorprogrammes für einen Fahrradcomputer ist sehr einfach, erfordert jedoch etwas Geduld. Sind der Simulatorcode und das Maschinenprogramm entweder auf der lokalen Festplatte oder (im Falle der Nutzung über das Internet) auf einer Website vorhanden, kann das Simulator-Programm bzw. Simulator-Applet gestartet werden.

Aufgrund der geringen Taktfrequenz des Fahrradcomputers musste eine sehr langsam gehende virtuelle Uhr implementiert werden: Ein Tick dieser virtuellen Uhr (eine zehntel virtuelle Sekunde) wird hier nur alle realen 7,5 Sekunden ausgelöst. Somit dauert eine virtuelle Sekunde real 75 Sekunden.

Die Uhr, der Pulsfühler und die Nabe sind nach dem Start sofort aktiv und ihre Unterbrechungswünsche werden sofort gezählt und angezeigt. Die Voreinstellung beträgt einen Herzschlag je 5 Ticks (also zwei in einer virtuellen Sekunde) und eine Nabenumdrehung je 10 Ticks (gleich einer virtuellen Sekunde).

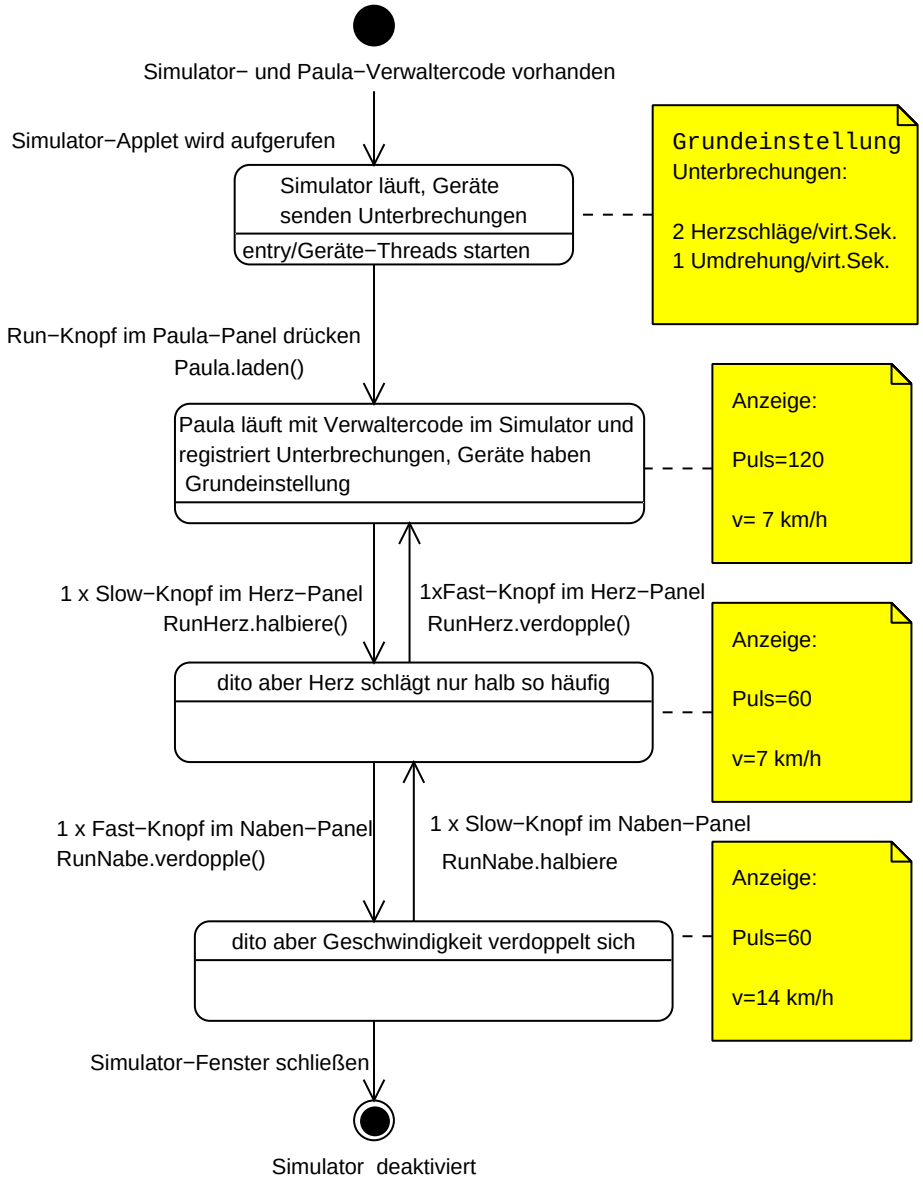
Erst nachdem der Fahrradcomputer über den Run-Knopf gestartet wurde (der Text „Paula is working!“ steht auf dem Bildschirm) werden die Unterbrechungen des Fahrradcomputers (genauer des Leerlaufprozesses) auch ausgewertet und der Puls und die Geschwindigkeit berechnet. Bei einem angenommenen Radumfang von zwei Metern kommt man durch die Voreinstellung auf eine ganzzahlige Geschwindigkeit von 7 km/h. Der Puls beträgt 120. Sollten diese beiden Werte nicht angezeigt werden, liegt es daran, dass der Fahrradcomputer zu spät gestartet wurde. Dann muss man einen Zyklus (5 Ticks) warten, bis diese beiden Werte angezeigt werden.

Zur Veränderung der Einstellungen von Herzfrequenz und Geschwindigkeit gibt es jeweils die beiden Schaltknöpfe Slow und Fast. Drückt man für den Puls einmal den Slow-Knopf, wird die Herzfrequenz halbiert, und es ergibt sich ein neuer Puls von 60. Wird anschließend für die Geschwindigkeit einmal der Fast-Knopf gedrückt, verdoppelt sich die Geschwindigkeit auf 14 km/h. Hier ist wieder etwas Geduld nötig, weil diese Änderungen erst nach der nächsten Wach-Phase der zuständigen Threads wirksam werden können. Über den jeweiligen alternativen Knopf können die Änderungen wieder zurückgenommen werden.

Die maximal mögliche Geschwindigkeit beträgt 72 km/h. Man erreicht sie durch dreimaliges Drücken des zuständigen Fast-Knopfes. In diesem Fall vergeht zwischen zwei Naben-Unterbrechungen nur ein Uhr-Tick, das entspricht 10 Umdrehungen pro Sekunde. Bei einem angenommenen Radumfang von 2 Metern ergibt das eben gerade eine Geschwindigkeit von 20 m/s oder 72 km/h.

Durch ähnliche Überlegungen kommt man auf einen maximal einstellbaren Puls von 600 durch zweimaliges Drücken des entsprechenden Fast-Knopfes.

Abbildung 8: Verschiedene Zustände des Applets

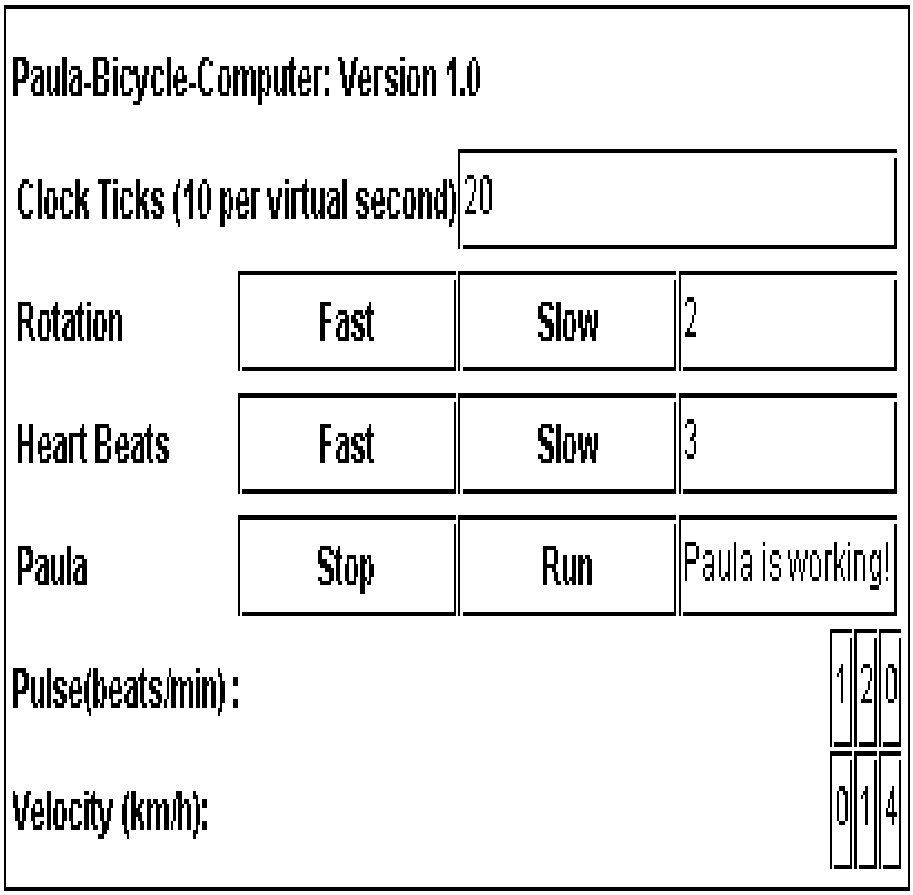


Im Zustandsdiagramm aus Abbildung 8 wird von einem geradlinigen Verlauf der Zustandsübergänge vom Startzustand bis zum Endzustand ausgegangen. Das hat hauptsächlich darstellerische Gründe und muss nicht so sein. Man kann durchaus nach einer Verringerung der Herzfrequenz auch wieder eine Steigerung der Herzfrequenz veranlassen. Das Gleiche gilt für die Geschwindigkeit. Die Implementierung der Beschleunigung und der Verzögerung wurde über eine Halbierung bzw. Verdopplung der „Schlafenszeit“ der zuständigen Threads realisiert.

8. Fazit

Zum Schluss soll der Simulator in Aktion vorgestellt werden. Der folgende Schnappschuss entstand, nachdem die Grundeinstellung für die Geschwindigkeit verdoppelt wurde.

Abbildung 9: Der Simulator in Aktion



Mit der vorgestellten Software wurde das Ziel der Bereitstellung eines erweiterbaren und kurzen Demonstrationsbeispiels für die Unterbrechungsbehandlung in einer Neumann-Maschine erreicht. Als größte Schwachstelle der erstellten Software wird die relativ geringe Abarbeitungsgeschwindigkeit der Java-Programme angesehen. Die simulierte Neumann-Maschine erreichte auf verschiedenen Rechnern maximal eine Taktfrequenz von 5.000 Hertz, was für die Unterbrechungsbehandlung selbst bei extrem geringer Frequenz der Unterbrechungen, z.B. beim Herzschlag nur 0,03, unzureichend war. Damit die angezeigten Werte von Puls und Geschwindigkeit realitätsnah werden, wurde mit virtuellen Sekunden gerechnet. Eine virtuelle Sekunde entspricht 75 realen Sekunden.

Für den Softwareentwurf wurde das UML-Werkzeug „Poseidon für UML“ in der Community Edition verwendet (siehe [3]). Beim Einsatz dieser Edition wurden überwiegend positive Erfahrungen gemacht.

Literatur

[1] **Mumm**, Harald (2005): Der vollständige Aufbau eines einfachen Fahrradcomputers, Hochschule Wismar, Fachbereich Wirtschaft 2005, WDP – Wismarer Diskussionspapiere Heft 6/2005, ISBN 3-910102-62-X.

[2] **Röck**, Hans (2004): Arbeitsmaterial Neumann-Maschine Paula, Universität Rostock, Sommersemester 2004.

[3] http://www.gentleware.com.

[5] Stevens, Perdita/Pooley, Rob (1999): UML-Softwareentwicklung mit Objekten und Komponenten, Pearson Studium 1999, ISBN 3-8273-7003-5 (Titel der Originalausgabe: Using UML: Software Engineering with Objects and Components).

Autorenangaben

Prof. Dr. rer. nat. Harald Mumm
Fachbereich Wirtschaft
Hochschule Wismar
Philipp-Müller-Straße 14
Postfach 12 10
D – 23966 Wismar
Telefon: ++49 / (0)3841 / 753 450
Fax: ++49 / (0)3841 / 753 131
E-mail: harald.mumm@wi.hs-wismar.de

WDP - Wismarer Diskussionspapiere / Wismar Discussion Papers

Heft 01/2003	Jost W. Kramer: Fortschrittsfähigkeit gefragt: Haben die Kreditgenossenschaften als Genossenschaften eine Zukunft?
Heft 02/2003	Julia Neumann-Szyszka: Einsatzmöglichkeiten der Balanced Scorecard in mittelständischen (Fertigungs-)Unternehmen
Heft 03/2003	Melanie Pippig: Möglichkeiten und Grenzen der Messung von Kundenzufriedenheit in einem Krankenhaus
Heft 04/2003	Jost W. Kramer: Entwicklung und Perspektiven der produktivgenossenschaftlichen Unternehmensform
Heft 05/2003	Jost W. Kramer: Produktivgenossenschaften als Instrument der Arbeitsmarktpolitik. Anmerkungen zum Berliner Förderungskonzept
Heft 06/2003	Herbert Neunteufel/Gottfried Rössel/Uwe Sassenberg: Das Marketingniveau in der Kunststoffbranche Westmecklenburgs
Heft 07/2003	Uwe Lämmel: Data-Mining mittels künstlicher neuronaler Netze
Heft 08/2003	Harald Mumm: Entwurf und Implementierung einer objektorientierten Programmiersprache für die Paula-Virtuelle-Maschine
Heft 09/2003	Jost W. Kramer: Optimaler Wettbewerb – Überlegungen zur Dimensionierung von Konkurrenz
Heft 10/2003	Jost W. Kramer: The Allocation of Property Rights within Registered Co-operatives in Germany
Heft 11/2003	Dietrich Nöthens/Ulrike Mauritz: IT-Sicherheit an der Hochschule Wismar
Heft 12/2003	Stefan Wissuwa: Data Mining und XML. Modularisierung und Automatisierung von Verarbeitungsschritten
Heft 13/2003	Bodo Wiegand-Hoffmeister: Optimierung der Sozialstaatlichkeit durch Grundrechtsschutz – Analyse neuerer Tendenzen der Rechtsprechung des Bundesverfassungsgerichts zu sozialen Implikationen der Grundrechte -
Heft 14/2003	Todor Nenov Todorov: Wirtschaftswachstum und Effektivität der Industrieunternehmen beim Übergang zu einer Marktwirtschaft in Bulgarien
Heft 15/2003	Robert Schediwy: Wien – Wismar – Weltkulturerbe. Grundlagen, Probleme und Perspektiven
Heft 16/2003	Jost W. Kramer: Trends und Tendenzen der Genossenschaftsentwicklung in Deutschland
Heft 01/2004	Uwe Lämmel: Der moderne Frege
Heft 02/2004	Harald Mumm: Die Wirkungsweise von Betriebssystemen am Beispiel der Tastatur-Eingabe
Heft 03/2004	Jost W. Kramer: Der Einsatz strategischer Planung in der Kirche
Heft 04/2004	Uwe Sassenberg: Stand und Möglichkeiten zur Weiterentwick-

lung des Technologietransfers an der Hochschule Wismar

Heft 05/2004 Thomas Gutteck: Umfrage zur Analyse der Kunden des Tourismuszentrum Mecklenburgische Ostseeküste GmbH

Heft 06/2004: Anette Wilhelm: Probleme und Möglichkeiten zur Bestimmung der Promotioneffizienz bei konsumentengerichteten Promotions

Heft 07/2004: Jana Otte: Personalistische Aktiengesellschaft

Heft 08/2004 Andreas Strelow: VR-Control – Einführung eines verbundeinheitlichen Gesamtbanksteuerungskonzepts in einer kleinen Kreditgenossenschaft

Heft 09/2004 Jost W. Kramer: Zur Eignung von Forschungsberichten als einem Instrument für die Messung der Forschungsaktivität

Heft 10/2004 Jost W. Kramer: Geförderte Produktivgenossenschaften als Weg aus der Arbeitslosigkeit? Das Beispiel Berlin

Heft 11/2004 Harald Mumm: Unterbrechungsgesteuerte Informationsverarbeitung

Heft 12/2004 Jost W. Kramer: Besonderheiten beim Rating von Krankenhäusern

Heft 01/2005 Michael Laske/Herbert Neunteufel: Vertrauen eine „Conditio sine qua non“ für Kooperationen?

Heft 02/2005 Nicole Uhde: Rechtspraktische Probleme bei der Zwangseinziehung von GmbH-Geschäftsanteilen – Ein Beitrag zur Gestaltung von GmbH-Satzungen

Heft 03/2005 Kathrin Kinder: Konzipierung und Einführung der Prozesskostenrechnung als eines Bestandteils des Qualitätsmanagements in der öffentlichen Verwaltung

Heft 04/2005: Ralf Bernitt: Vergabeverfahren bei öffentlich (mit)finanzierten sozialen Dienstleistungen

Heft 05/2005: Jost W. Kramer: Zur Forschungsaktivität von Professoren an Fachhochschulen, untersucht am Beispiel der Hochschule Wismar

Heft 06/2005 Harald Mumm: Der vollständige Aufbau eines einfachen Fahrradcomputers

Heft 07/2005: Melanie Pippig: Risikomanagement im Krankenhaus

Heft 08/2005: Yohanan Stryjan: The practice of social entrepreneurship: Theory and the Swedish experience

Heft 09/2005: Sebastian Müller/Gerhard Müller: Sicherheits-orientiertes Portfoliomanagement

Heft 10/2005: Jost W. Kramer: Internes Rating spezieller Kundensegmente bei den Banken in Mecklenburg-Vorpommern, unter besonderer Berücksichtigung von Nonprofit-Organisationen

Heft 11/2005: Rolf Steding: Das Treuhandrecht und das Ende der Privatisierung in Ostdeutschland – Ein Rückblick –

Heft 12/2005: Jost W. Kramer: Zur Prognose der Studierendenzahlen in Mecklenburg-Vorpommern bis 2020

Heft 13/2005: Katrin Pampel: Anforderungen an ein betriebswirtschaftliches Risikomanagement unter Berücksichtigung nationaler und internationaler Prüfungsstandards

Heft 14/2005: Rolf Steding: Konstruktionsprinzipien des Gesellschaftsrechts und seiner (Unternehmens-)Formen

Heft 15/2005: Jost W. Kramer: Unternehmensnachfolge als Ratingkriterium

Heft 16/2005: Christian Mahnke: Nachfolge durch Unternehmenskauf - Werkzeuge für die Bewertung und Finanzierung von KMU im Rahmen einer externen Nachfolge –

Heft 17/2005 Harald Mumm: Softwarearchitektur eines Fahrrad-Computer-Simulators