Objektorientierte Systementwicklung mit der Unified Modeling Language (UML)

(Dr. Markus Nüttgens, Dipl.-Hdl. Michael Hoffmann, Dipl.-Inform. Thomas Feld, Institut für Wirtschaftsinformatik (IWi), Universität des Saarlandes, Saarbrücken)

Vorgehensmodelle für die objektorientierte Systementwicklung

Im Kontext der objektorientierten Systementwicklung werden in der Literatur primär evolutionäre Vorgehensmodelle diskutiert (vgl. Boehm, B. W., A Spiral Model of Software Development and Enhancement, S. 64, Henderson-Sellers, B., Edwards, J. M., The Object-Oriented System Life Cycle, S. 152, Meyer, B., From Structured Programming to Object-Oriented Design, S.19-39). Dies ist u.a. in den Grundlagen des objektorientierten Paradigmas begründet, wonach Objekte in sich abgeschlossene und eigenständig existente Subsysteme darstellen. Aufgrund der Definition einer internen und exterenen Objektstruktur können somit Systeme mit einer skalierbaren Größe entwickelt werden. Bei der evolutionären Vorgehensweise endet jeder Zyklus mit einem ablauffähigen Softwaresystem. Dies wird erreicht, indem aus Projektzielen Entwicklungsergebnisse abgeleitet werden, deren Einsatz, zumindest dv-technisch, auch isoliert möglich ist. Die einzelnen Teilsysteme können dadurch schneller zur Verfügung gestellt und früher auf ihre Einsetzbarkeit hin getestet werden. Die Weiterentwicklung beinhaltet einerseits Verbesserungen, die sich durch den Praxistest ergeben, und andererseits die Realisierung weiterer Teilsysteme. So ist es möglich, bei großen Projekten durch Prototypenentwicklungen schon relativ früh Ergebnisse zu präsentieren und Fehlentwicklungen zu vermeiden.

Entwicklung und Beschreibung eines Vorgehensmodells

Nachfolgend wird am Beispiel der Unified Modeling Language (UML) (vgl. Rumbaugh, J., Jacobsen, I., Booch, G., Unified Modeling Language Documentation Set 1.1) ein Vorgehensmodell zur objektorientierten Systementwicklung aufgezeigt. Im Umfang der UML ist bisher kein Vorgehensmodell explizit definiert. In Abbildung 1 ist daher ein grobes Vorgehensmodell dokumentiert, welches ein mögliches Vorgehen im Rahmen der objektorientierten Systementwicklung beschreibt.

Grundlage für die objektorientierte Systementwicklung ist in der Regel ein optimiertes Geschäftsprozeßmodell (vgl., Oestereich, B., Objektorientierte Softwareentwicklung mit der Unified Modeling Language, S. 85, Yourdon, E. et al., Mainstream Objects, S. 71 ff.).

Die Teilprozesse der objektorientierten Systementwicklung (Analyse, Design und Montage) beschreiben einen Zyklus mit Rückkopplungen zwischen den einzelnen Elementen des Vorgehensmodells. Ist ein objektorientiertes Entwicklungsprojekt freigegeben, erfolgt zunächst eine objektorientierte Analyse. Daran schließt sich das objektorientierte Design an. Treten beim objektorientierten Design Widersprüche oder Probleme auf, die in dieser Phase gelöst werden können, erfolgt eine Rückdelegation zur objektorientierten Analyse.

Nüttgens, M.; Hoffmann, M.; Feld, T.: Objektorientierte Systementwicklung mit der Unified Modeling Language (UML), in: Scheer, A.-W.: ARIS - Modellierungsmethoden, Metamodelle, Anwendungen, 3. Aufl., Berlin et. al. 1998. (http://www.iwi.uni-sb.de/nuettgens/Aris-uml-vgm/Aris-uml-vgm.ps)

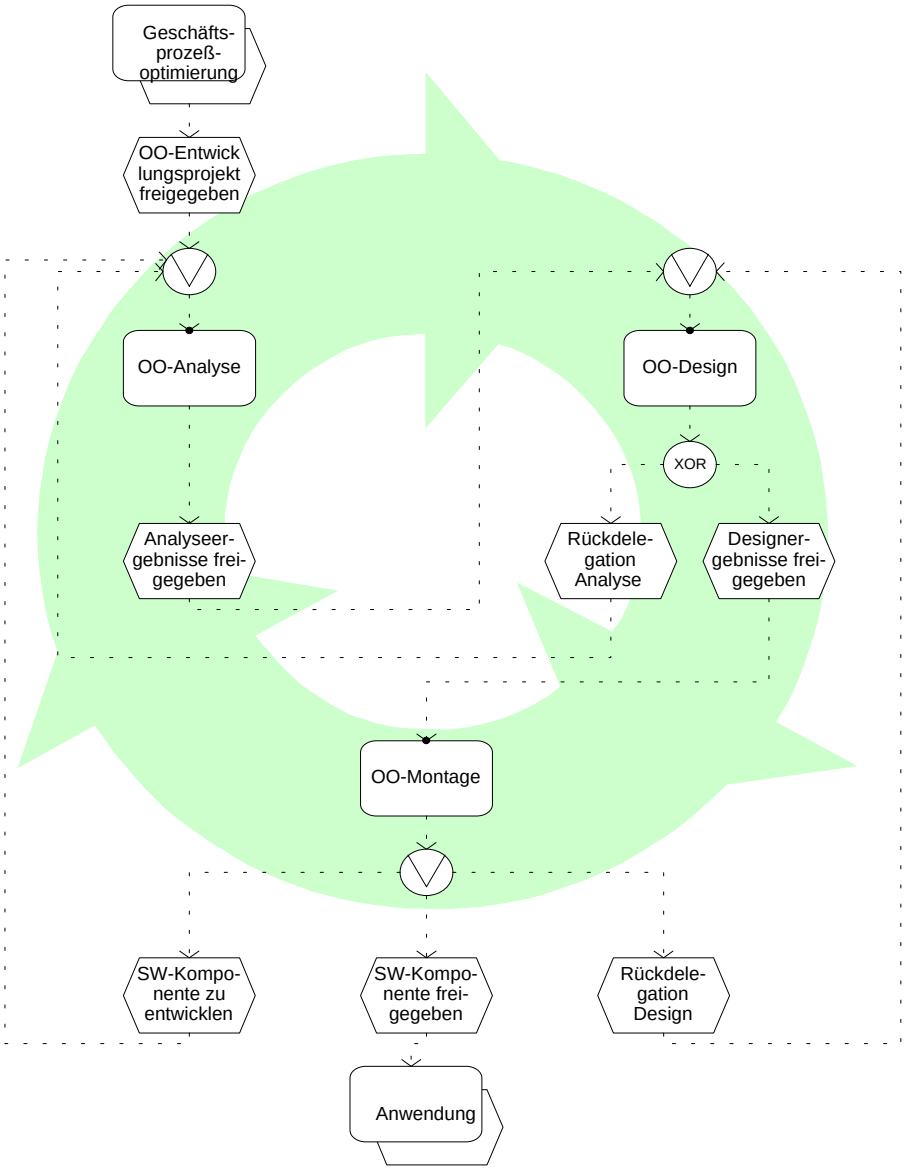

Abbildung 1: GrobesVorgehensmodell für die objektorientierte Systementwicklung

Sind die Designergebnisse freigegeben, bilden diese den Input für die objektorientierte Montage. Die Design-Modelle werden mit Hilfe von Code-Generatoren weitgehend automatisch implementiert. Treten bei der Code-Generierung Fehler auf, die ihren Ursprung im Design haben, erfolgt eine Rückdelegation. Wird die entwickelte Software-Komponente für den Einsatz freigegeben, beginnt der operative Einsatz. Hier wird der Übergang von der Build-Time zur Run-Time vollzogen. Ist eine Überarbeitung der Software-Komponete erforderlich bzw. sind weitere Softwarekomponenten zu entwickeln, wird mit dem Eintritt in die objektorientierte Analyse der Zyklus erneut durchlaufen. Im folgenden werden die Teilprozesse des Vorgehensmodells auf einer detaillierteren Abstaktionsstufe beschrieben.

Objektorientierte Analyse

Abbildung 2 beschreibt den objektorientierten Analyseprozeß auf der Grundlage eines Bausteins für phasenbezogene Vorgehensmodelle. (vgl. Nüttgens, M., Koordiniert–dezentrales Informationsmanagement, S. 223)

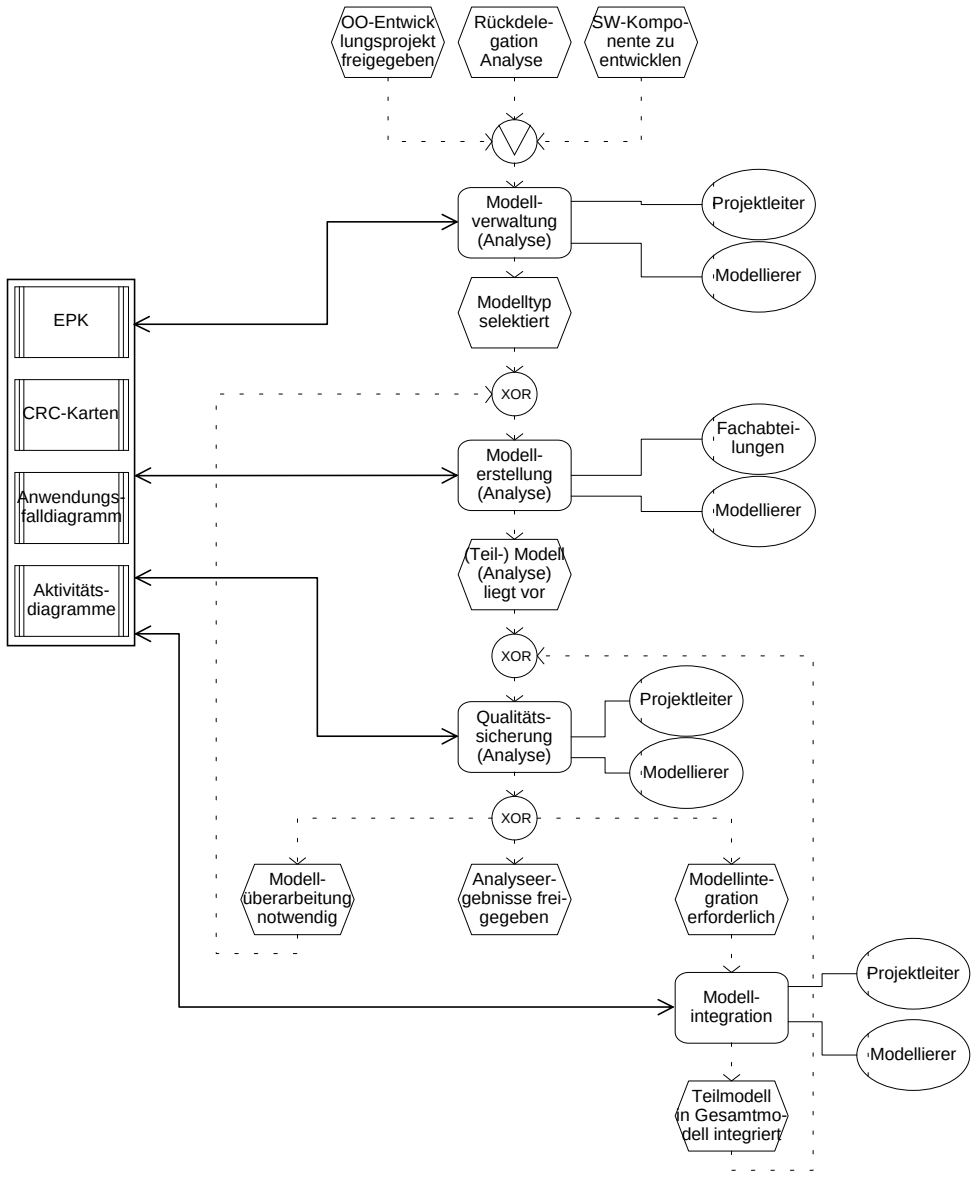

Abbildung 2: Vorgehensmodell objektorientierte Analyse

Im ersten Schritt der Modellverwaltung werden die Modelltypen zur Systemanalyse von dem Projektleiter in Absprache mit den Modellierern ausgewählt. Relevante UML-Modelltypen sind CRC-Karten (class-responsibilities-collaborators), Anwendungsfalldiagramme (use case diagram) und Aktivitätsdiagramme (activity diagram). Diese können auf der Grundlage bereits erhobender Geschäftsprozeßmodelle, beispielsweise Ereignisgesteuerter Prozeßketten (EPK), erstellt werden.

Anwendungsfalldiagramme werden im Rahmen von UML schwerpunktmäßig zur ersten Erhebung von Organisationsszenarien eingesetzt. Diese können aus einzelnen Funktionsbausteinen eines EPK-Models abgeleitet und dann in der entsprechenden UML-Notation detailliert beschrieben werden. Hierbei kann die Unterstützung von Funktionen durch eine Anwendungskomponente einen ersten Anhaltspunkt zur Strukturierung der Anwendungsfälle geben. Aktivitätsdiagramme können aus den Kontrollflußinformationen eines EPK-Modells abgeleitet und um die Beschreibung konkreter Objektzustände ergänzt werden. Aktivitätsdiagramme bilden auch die Grundlage zur Ableitung objektorientierter Workflow-Management-Modelle.

Nach erfolgter Qualitätssicherung müssen (Teil-)Modelle in ein Gesamtmodell integriert und fehlerhafte Modelle überarbeitet werden. Vollständige und fehlerfreie UML-Modelle dienen dann als Grundlage für das objektorientierte Design.

Objektorientiertes Design

In Abbildung 3 ist das Vorgehensmodell zum objektorientierten Design in Analogie zum Vorgehensmodell zur objektorientierten Analyse dargestellt. Als UML-Modelltypen werden hierbei Zustandsdiagramme (state diagram) , Sequenzdiagramme (sequence diagram), Klassendiagramme (class diagram), Kollaborationsdiagramme (collaboration diagram) oder detaillierte Aktivitätsdiagramme (activity diagram) eingesetzt.

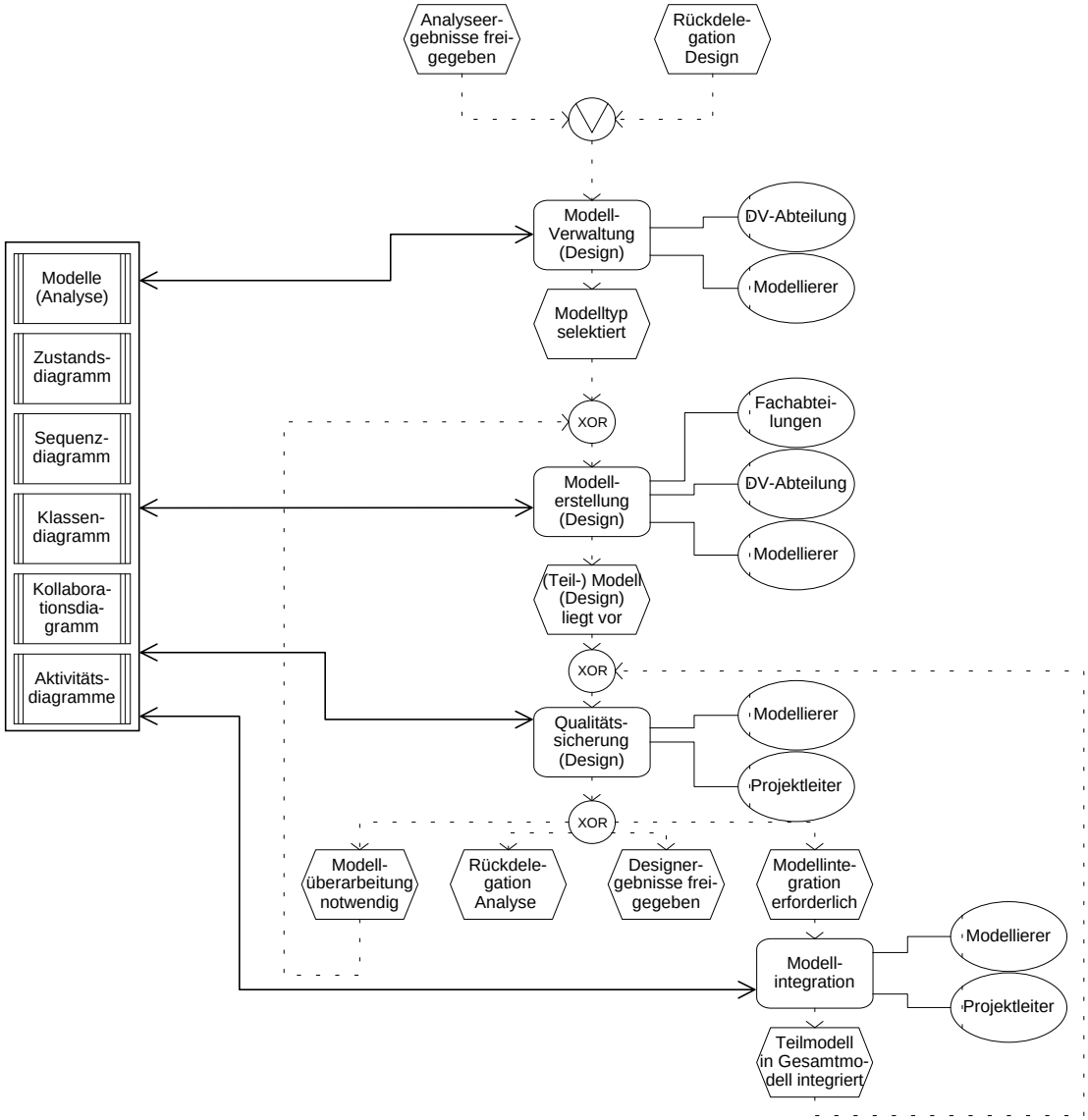

Abbildung 3: Vorgehensmodell objektorientiertes Design

Besondere Bedeutung kommt hierbei dem Entwurf der Klassendiagramme zu. Erste Anhaltspunkte zum Entwurf der Klassendiagramme können analog zur objektorientierten Analyse auf der Grundlage von Geschäftsprozeßmodellen erfolgen. So lassen sich beispielsweise Funktionsbausteine und Input-Outputdaten der EPK-Modelle objektbezogen detailliern und zu den

relevanten Geschäftsobjektklassen zugeordnen. Sollte bereits ein strukturiertes Datenmodell beispielsweise in der Form eines Entity-Realstionship-Modells (ERM) vorliegen, können auch hieraus bereits die wesentlichen Klassen und ihre Strukturbeziehungen abgeleitet werden.

Objektorientierte Montage

Abbildung 4 beschreibt das Vorgehensmodell zur objektorientierten Montage. Bei der Implementierung der objektorientierten Designmodelle wird eine weitgehende Automatisierung durch den Einsatz objektorientierter Code-Generatoren angestrebt. Teilweise muß auch zusätzlicher Programmcode, zum Beispiel zur Implementierung von Methoden, erzeugt werden. Hierbei kann, wenn vorhanden, auf bestehende Klassenbibliotheken zurückgegriffen werden. Der weitere Prozeßverlauf verläuft analog zum Vorgehensmodell des objektorientierten Designs bzw. der objektorientierten Analyse.

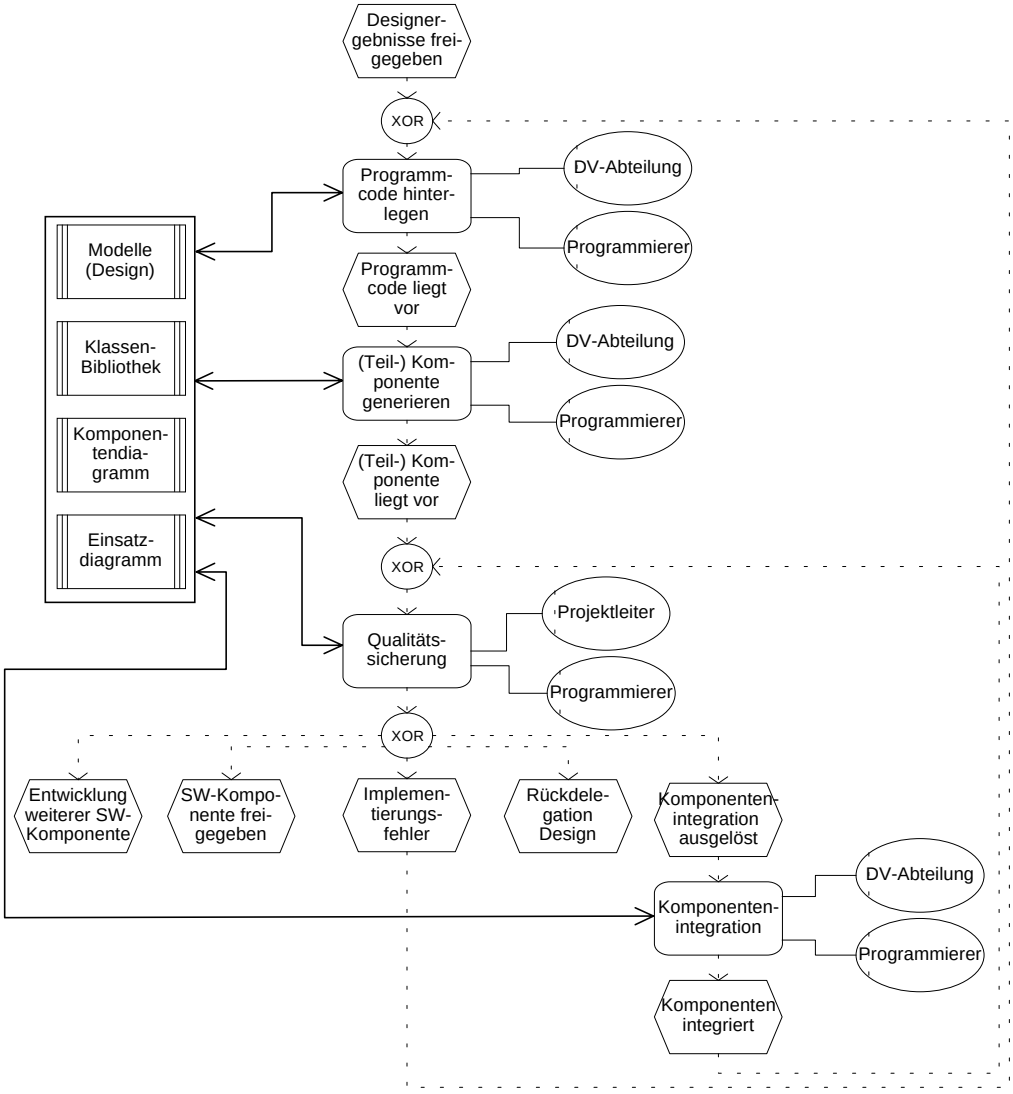

Abbildung 4: Vorgehensmodell objektorientierte Montage

Nach erfolgreicher Integration von (Teil-)Komponenten kann sowohl die Entwicklung weiterer (Teil-)Komponenten angestoßen werden als auch die Anwendung der bereits entwickelten und integrierten (Teil-)Komponenten erfolgen.

Ausblick

Sowohl die primär strukturierte als auch die primär objektorientierte Systementwicklung basiert auf dem gemeinsamen Verständnis, in einem ersten Schritt innerhalb des Diskursbereiches Geschäftsprozese auf der Grundlage betriebswirtschaftlich-organisatorischer Zielvorgaben zu optimieren. Derart optimierte Geschäftsprozesse bilden in der Form ablauforganisatorischer Szenarien den fachlichen Ausgangspunkt zur Systementwicklung. Hierzu hat sich mit der Methode der Ereignisgesteuerten Prozeßkette (EPK) eine sowohl in der Theorie als auch in der Praxis akzeptierte Methode für betriebswirtschaftliche Fachinhalte etabliert. Bisher fehlt es aber an konkreten methodischen Erweiterungen, ablauforganisatorisch und funktional strukturierte EPK-Modelle in objektorientierte Analyse- und Designmodelle zu überführen. (Hoffmann, W., Scheer, A.-W., Hoffmann, M., Überführung strukturierter Modellierungsmethoden in die Object Modeling Technique (OMT), Bungert, W., Heß, H., Objektorientierte Geschäftsprozeßmodellierung, Scheer, A.-W., Nüttgens, M., Zimmermann, V., Objektorientierte Ereignisgesteuerte Prozeßkette (oEPK)).

Vorgehensmodelle können einen wesentlichen Beitrag zur Entwicklung integrativer Ansätze leisten, indem der Verwendungszweck und Nutzen der Modelle im Kontext eines durchgängigen Systementwicklungsprozesses offengelegt und dokumentiert wird.

Literatur

Bungert, W., Heß, H., Objektorientierte Geschäftsprozeßmodellierung. In: Information Management 10 (1995) 1, S. 52-63.

Boehm, B.W., A Spiral Model of Software Development and Enhancement, In: IEEE Computer, 21 (1988) 5, S. 61-72.

Henderson-Sellers, B.; Edwards, J.M., The Object-Oriented System Life Cycle, In: Communications of the ACM, 33 (1990) 9, S. 142-159.

Hoffmann, W., Scheer, A.-W., Hoffmann, M., Überführung strukturierter Modellierungsmethoden in die Object Modeling Technique (OMT), In: Scheer, A.-W. (Hrsg.) Veröffentlichungen des Institutes für Wirtschaftsinformatik (IWi), Saarbrücken 1995.

Meyer, B., From Structured Programming to Object oriented Design: The Road to Eiffel. In: Structured Programming, 10 (1989) 1, S. 19-39.

Nüttgens, M., Koordiniert–dezentrales Informationsmanagement: Rahmenkonzept – Koordiantionsmodelle – Werkzeug-Shell, Wiesbaden 1995.

Oestereich, B., Objektorientierte Softwareentwicklung mit der Unified Modeling Language, 3. Aufl., München, Wien 1997.

Rumbaugh, J, Jacobsen, I., Booch, G., Unified Modeling Language Documentation Set 1.1, Rational Software Coperation (Hrsg.), Santa Clara 1997, http://www.rational.com/uml/documentation.html vom 27.11.1997.

Scheer, A.-W., Nüttgens, M., Zimmermann, V., Objektorientierte Ereignisgesteuerte Prozeßkette (oEPK) – Methode und Anwendung, In: Scheer, A.-W. (Hrsg.) Veröffentlichungen des Institutes für Wirtschaftsinformatik (IWi), Saarbrücken 1997.

Yourdon, E. et al., Mainstream Objects, München et al. 1996.