8 Übertragung von Vorgehensmodellen aus der Informatik und Informationswissenschaft als Handlungsanleitung und Orientierungshilfe

Die meisten Universitätsfachbereiche befassen sich nicht direkt mit Informations- und Kommunikationstechnologien und besitzen daher zum Teil kaum Grundlagenwissen für den Umgang mit Computern. In der Informatik und Informationswissenschaft existieren dagegen insbesondere für die Entwicklung von Anwendungsprogrammen spezielle Vorgehensmodelle wie das Lebenszyklus-Phasenmodell, das objektorientierte Modell und das Prototyping, die den Integrationsprozeß in anderen Fächern unterstützen können. Ein Beweis für die Notwendigkeit solcher Modelle ist, daß im Projekt VetMedia im Verlauf von mehreren Jahren durch Experimentieren ein eigenes implizites Vorgehensmodell gefunden worden ist, das zwar nicht schriftlich fixiert und bewußt als Modell eingesetzt wird, dem aber unbewußt gefolgt wird. Unter dem Begriff „Modell“ versteht man im allgemeinen:[344]

- ein Muster, Vorbild und Entwurf für einen zu erstellenden Gegenstand oder einen zukünftigen Ablauf
- eine Vereinfachung, die eine Untersuchung oder Erforschung eines Gegenstandes bzw. Ablaufes erleichtert oder erst möglich macht

8.1 Vorgehensmodelle in der Informatik

Ein Vorgehens- bzw. Prozeßmodell ist ein allgemeiner Entwicklungsplan, der festlegt, welche Aktivitäten beim Entwickeln eines Software-Produktes wie und in welcher Abfolge durchgeführt werden.[345] Ein solches Modell erläutert u.a. die durchzuführenden Aktivitäten wie „Anforderungen ermitteln“, „Software-Architektur entwerfen“ und „Programmcode schreiben“. In der Regel werden die verschiedenen Aktivitäten zu Phasen wie z.B. Problemanalyse, Definition, Entwurf, Implementierung, Einführung und Wartung zusammengefaßt. Jede Phase wird durch die Erstellung von einem oder mehreren Dokumenten abgeschlossen, bevor mit der nächsten Phase begonnen wird. Die Definitionsphase wird z.B. durch die Erstellung eines Pflichtenheftes abgeschlossen. Durch die Festlegung von solchen Phasenabschluß-Kriterien wird der Fortschritt einer Entwicklung kontrollierbar. Die Phasen werden in einer Ablaufstruktur geordnet, die je nach Modell z.B. linear, iterativ oder zyklisch sein kann. Insgesamt stellen Vorgehensmodelle allgemeines Handlungswissen für die Software-Entwicklung zur Verfügung, so daß man bei konkreten Projekten nach diesen Modellen vorgehen kann.

8.2 Unbekanntheit von Vorgehensmodellen außerhalb der Informatik

Obwohl diese Modelle in der Informatik anerkannt sind, werden sie bei der Entwicklung von Multi-/Hypermedia-CD-ROM- und Internet-Lernanwendungen für das Studium der Tiermedizin kaum eingesetzt. Dies trifft vermutlich auch für andere Fächer mit geringer Computer-Erfahrung wie z.B. Kunstgeschichte zu. Gründe dafür sind, daß diese Modelle bei den Entwicklern, d.h. Hochschullehrern und Studenten, weitgehend unbekannt sind und daß die Verwendung nicht zwingend notwendig ist, weil die Anwendungen hauptsächlich aus Dokumenten bestehen, deren Erstellung durch Entwicklungswerkzeuge wie z.B. ToolBook® intuitiv ist und weil die Vorgehensweise

[344] Vgl. Duden Fremdwörterbuch, 1974, S. 470.
[345] Vgl. Balzert, 1998, S. 135.

durch Experimentieren gefunden werden kann. So sind z.B. die in dieser Arbeit beschriebenen Anwendungen „Brunstzyklus beim Rind“, „Tiergeburtshilfe“ und „Rund- und Bandwürmer bei Hund und Katze“ ohne den Einsatz von Vorgehensmodellen erstellt worden.

8.3 Probleme bei der Software-Entwicklung ohne die Verwendung von Vorgehensmodellen

Nach Balzert soll grundsätzlich jede Software-Entwicklung im Rahmen eines Vorgehens- bzw. Prozeß-Modells erfolgen[346], weil die Entwicklung ohne Modelle nach dem „Code&Fix“-Prinzip mit den Arbeitsschritten

1. Schreibe ein Programm
2. Finde und behebe die Fehler in dem Programm

u.a. folgende Nachteile besitzt:

- Zur Behebung von Fehlern wird das Programm umstrukturiert, so daß weitere Fehlerbehebungen immer aufwendiger werden. Dies führt zu der Erkenntnis, daß eine Entwurfsphase vor der Programmierung notwendig ist, in der u.a. der Programmaufbau systematisch geplant wird.
- Selbst weitgehend fehlerfreie Software wird vom Endbenutzer oft nicht akzeptiert, wenn es keinen Bedarf dafür gibt. Dies führt zu der Erkenntnis, daß eine Definitionsphase vor dem Entwurf benötigt wird, in der die Benutzeranforderungen ermittelt werden.
- Programm-Fehler sind schwierig zu finden, wenn Tests nicht systematisch vorbereitet und durchgeführt werden. Dies führt zu der Erkenntnis, daß eine separate Testphase notwendig ist.

Zusammengenommen ergibt sich aus diesen Erkenntnissen das klassische Lebenszyklus-Phasenmodell, aus dem sich die meisten anderen Vorgehensmodelle ableiten lassen. Dieses Modell wird in Abbildung 40 als eine Synthese der Darstellungen von Balzert sowie Kimm, Koch, Simonsmeier und Tontsch wiedergegeben. Die Bezeichnung „Lebenszyklus“ stammt daher, daß dieses Modell den Entwicklungsprozeß vom Beginn der Erstellung eines Produktes bis zum Ende seiner Verwendung beschreibt. Im Hinblick auf die Integration neuer Technologien an Hochschulen ist dabei insbesondere die sogenannte „Einführungsphase“ von Bedeutung, die aus folgenden Tätigkeiten besteht:[347]

- Installation, d.h. Einrichtung des Software-Produktes in der Zielumgebung zum Zweck des Betriebes
- Schulung der Benutzer und des Betriebspersonals zur Einweisung in die Handhabung des Produktes
- Inbetriebnahme des Produktes, d.h. der Übergang zwischen Installation und Betrieb

Dieser Begriff der „Einführung“ zielt in erster Linie auf die Integration eines einzelnen eigenentwickelten Software-Produktes wie z.B. einer Lernanwendung und weniger auf die flächendeckende Einführung von Computer-Technologien in das Studium.

---

[346] Vgl. Balzert, 1998, S. 98.
[347] Vgl. Balzert, 1996, S. 964.

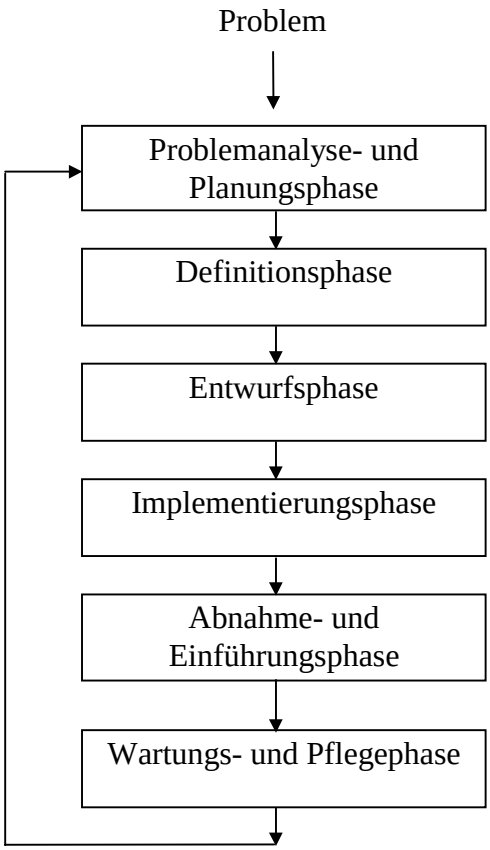

Abbildung 40: Lebenszyklus-Phasenmodell[348]

Kimm, Koch, Simmonsmeier, Tontsch geben als Grund für den Einsatz von Vorgehensmodellen an, daß die Programmentwicklung ohne Modelle mitte der sechsiger Jahre in einem solchen Maß zu fehlerhaften und kaum wartbaren Programmen geführt hat, daß man von der sogenannten „Software-Krise“ zu sprechen begann.[349] Zur Lösung der Softwarekrise in der Informatik ist u.a. das Software-Engineering bzw. die Software-Technik als wissenschaftliche Disziplin entwickelt worden.

Die Begriffe „Software-Engineering“ und „Software-Technik“ werden in der Literatur weitgehend synonym verwendet. Software-Technik ist die zielorientierte Bereitstellung und systematische Verwendung von Prinzipien, Methoden, Konzepten, Notationen und Werkzeugen für die arbeitsteilige, ingenieursmäßige Entwicklung und Anwendung von umfangreichen Software-Systemen.[350] In der Software-Technik stellen u.a. Vorgehens- bzw. Prozeßmodelle eine Hauptmethode für die systematische Entwicklung von Programmen dar.

Der Informatiker Boles meint, daß Vorgehensmodelle der Software-Entwicklung, die aufgrund der Softwarekrise in den 60er Jahren entstanden sind, in der Multimedia-Praxis heutzutage kaum Verwendung finden, wie eine Befragung von einmal 40[351] und einmal 650[352] Multimedia-Software-Herstellern ergeben hat. Boles befürchtet daher, daß

[348] Vgl. Balzert, 1996, S. 42 und Kimm, Koch, Simonsmeier, Tontsch, 1979, S. 19.
[349] Vgl. Kimm, Koch, Simmonsmeier, Tontsch, 1979, S. 12-14.
[350] Vgl. Balzert, 1996, S. 36.
[351] Vgl. Sawhney, 1995.
[352] Vgl. Hitzges, Laich, 1995.

man bereits in wenigen Jahren analog zum Begriff der Softwarekrise von der Multimedia-Krise bzw. WWW-Krise sprechen wird.[353]

Nach Sander wird das Fehlen einer systematischen Vorgehensweise bei der Entwicklung von Multi-/Hypermedia-Anwendungen dadurch verdeckt, daß diese Anwendungen selten eine Größenordnung erreichen, die eine solche Vorgehensweise erforderlich macht.[354]

8.4 Vorgehensmodelle speziell für die Entwicklung von Lernanwendungen

Die folgenden Vorgehensmodelle unterstützen insbesondere die Erstellung von Lernanwendungen und werden daher in speziellen Abschnitten detailliert erläutert:

- Phasenkonzept der Teachwareentwicklung von Bodendorf
- Modell der Lernprogramm-Entwicklung von Steppi
- DIALEKT-Entwicklungsmodell für hypermediale Lernsysteme

8.4.1 Phasenkonzept der Teachwareentwicklung von Bodendorf

Für die Entwicklung von Lernanwendungen schlägt Bodendorf ein am Wasserfall-Modell orientiertes Vorgehensmodell vor, das als „Phasenkonzept der Teachwareentwicklung“ bezeichnet wird.[355] Die folgende Abbildung zeigt, daß bei der Entwicklung nach diesem Modell schrittweise von allgemeinen, nichttechnischen Spezifikationsphasen zu immer konkreteren Realisierungsphasen vorgegangen wird, wobei die Ergebnisse einer Phase wie bei einem Wasserfall in die jeweils nächste Phase „fallen“.

Zielanalyse
In der Zielanalyse wird u.a. eine Bedarfs- und Zielgruppenanalyse durchgeführt, um zu ermitteln, welche Kenntnisse und Fähigkeiten durch die zu erstellende Lernsoftware vermittelt werden sollen.

Lösungskonszept
Aufbauend auf den ermittelten Lernzielen und Lerninhalten wird in der zweiten Phase ein Lösungskonzept bzw. Pflichtenheft erstellt, das u.a. den Typ des zu entwickelnden Programms wie z.B. Tutorium, Simulation oder Wissenstest zur Erreichung der in der Zielanalyse ermittelten Anforderungen festlegt. Weiterhin werden die für die Entwicklung benötigten Ressourcen an Personal, Hardware, Software und Geld definiert und ein Projektplan aufgestellt.

Pädagogisches Design
Das pädagogische Design umfaßt u.a. die Festlegung einer Lehrstrategie, die Modularisierung, d.h. Aufteilung des Lehrstoffes in einzelne Informationseinheiten, und die Bestimmung der inhaltlichen Abhängigkeiten zwischen einzelnen Modulen. Das Ergebnis besteht in einem zeitlichen Ablaufplan der Lektionen in Form eines

---

[353] Vgl. Boles, D.: Erstellung multimedialer Dokumente und Anwendungen - Verfahren und Werkzeuge, in: Workshop "Software-Engineering für Multimedia-Systeme" im Rahmen der GI"97-Jahrestagung, Aachen, 26. September 1997, 09/97, URL: http://www-is.informatik.uni-oldenburg.de/~dibo/paper/gi97-mmse/main.html, Stand: 24.02.98.

[354] Vgl. Sander, 1997, S. 233.

[355] Vgl. Bodendorf, 1990, S. 75-78.

Drehbuches, das die Vorgaben des Pflichtenheftes in konkrete Lehrelemente und -abläufe umsetzt.

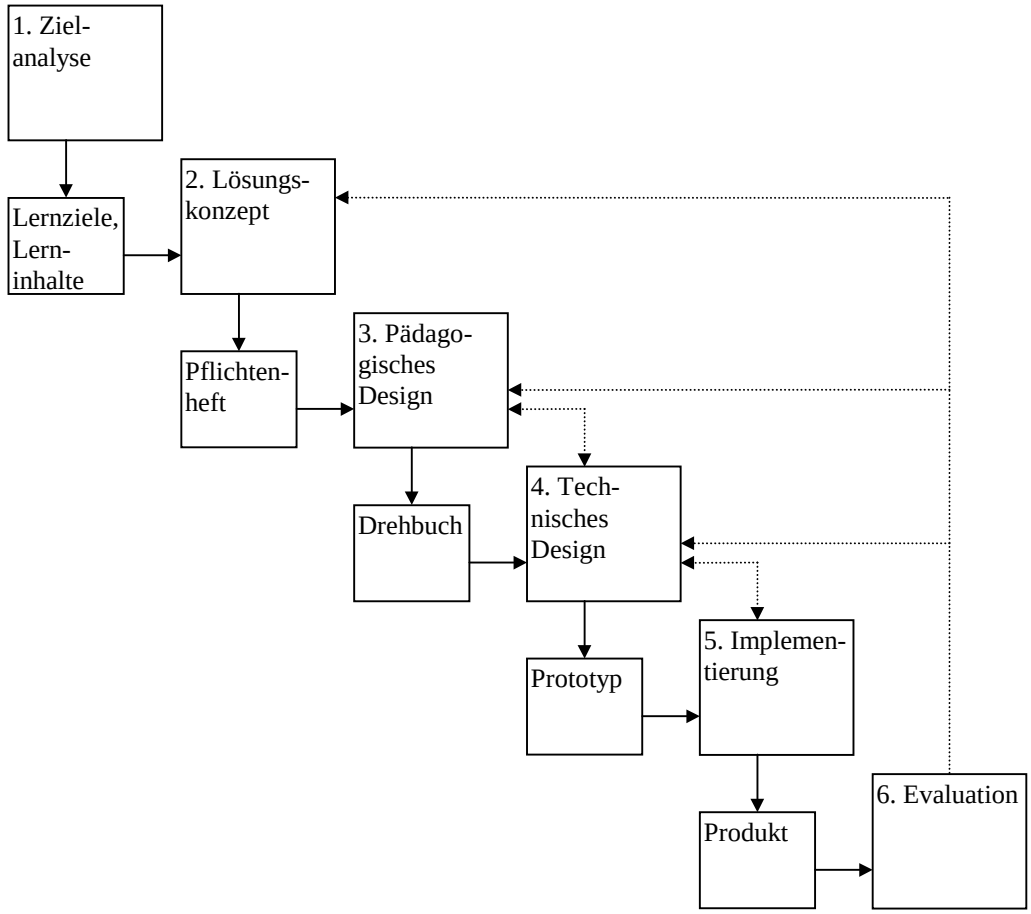

Abbildung 41: Phasenkonzept der Teachwareentwicklung nach Bodendorf

Technisches Design
In dieser Phase wird das Drehbuch in ein technisches Design umgesetzt. Dazu werden u.a. die zu verwendende Hard- und Software sowie die vorhandenen und zu erstellenden Medien definiert und ein Entwurf der Benutzerschnittstelle angefertigt. Das Endprodukt dieser Phase besteht aus einem Prototypen, der das pädagogische Design des vorhergehenden Entwicklungsschrittes widerspiegelt.

Implementierung
In der Implementierungsphase wird der Prototyp zu einem einsatzfähigen Produkt weiterentwickelt.

Evaluation
Die anschließende Evaluation gibt Aufschluß über die Akzeptanz, den Lernerfolg und die Nutzung des Lernsystems. Die Ergebnisse der Evaluation dienen als Grundlage für eine eventuelle Überarbeitung und Verbesserung des Programmes.

Durch die gestrichelten Pfeile in der Abbildung des Phasenmodells wird angedeutet, daß beim Entwicklungsprozeß ähnlich wie im Wasserfall-Modell jederzeit wieder auf die vorangegangenen Stufen zurückgesprungen werden kann, wenn dort Aufgaben wiederholt werden müssen.

8.4.2 Modell der Lernprogramm-Entwicklung von Steppi

Steppi beschreibt das folgende Modell der Lernprogramm-Entwicklung, das den Entwicklungsprozeß in drei Abschnitte unterteilt, die jeweils jeweils aus mehreren Entwicklungsphasen bestehen.[356] Diese drei Abschnitte sind die Lernprogramm-Planung, die Lernprogramm-Erstellung und die Lernprogramm-Erprobung. Das Modell beschreibt in erster Linie die professionelle Erstellung von Lernanwendungen z.B. in einer Software-Firma, denn am Anfang einer Entwicklung steht der Auftrag eines Kunden und am Ende der Vertrieb des fertigen Programms.

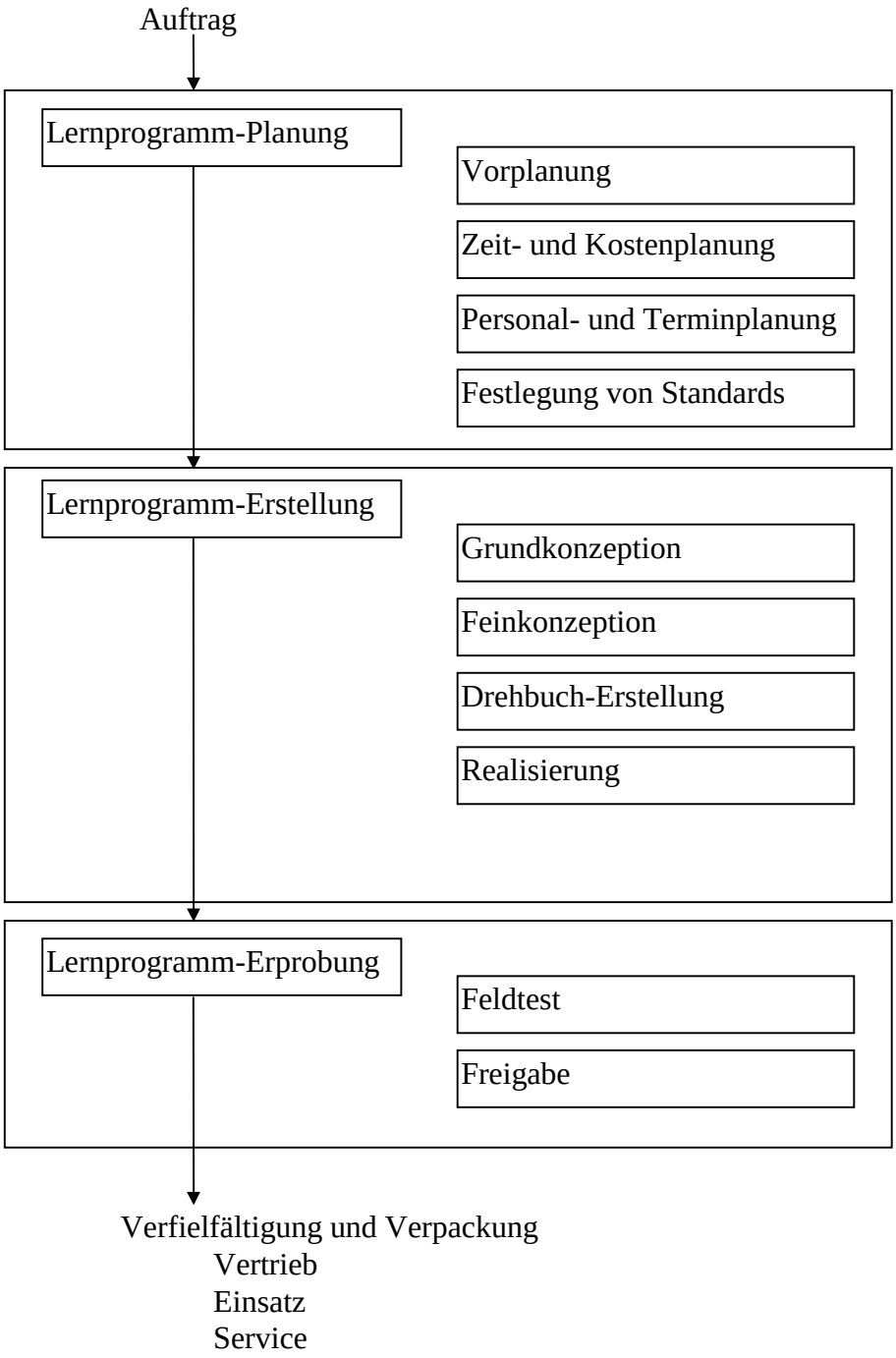

Abbildung 42: Modell der Lernprogramm-Entwicklung nach Steppi

[356] Vgl. Steppi, 1990, S. 128-189.

Im Rahmen der Vorplanung wird u.a. geklärt, ob die Voraussetzungen für die Entwicklung und den Einsatz von computerbasierten Lernanwendungen erfüllt sind, z.B. ob sich die geplanten Ausbildungsziele überhaupt durch ein Computer-Lernsystem erreichen lassen oder ob herkömmliche Unterrichtsformen wie Seminare besser geeignet sind. Der zweite Schritt besteht in der Zeit- und Kostenplanung, die auf einer Aufwandsschätzung für den gesamten Entwicklungsprozeß beruht. Die Durchführung dieser Zeit- und Kostenplanung ist für unerfahrene Hochschullehrer und Studenten kaum möglich, weil sie die Dauer und den Aufwand für die einzelnen Aktivitäten während der Entwicklung kaum beurteilen können. Nach Steppi basiert das Modell auf dem Einsatz erfahrener Fachautoren, geübter Programmierer und Graphiker.[357] Im dritten Schritt der Lernprogrammplanung erfolgt u.a. die Zusammenstellung des Entwicklungsteams und die Benennung der Projektleitung. Zum Abschluß der Planung werden u.a. zur Rationalisierung der Entwicklungsarbeit und zur Sicherung eines einheitlichen Erscheinungsbildes sogenannte Standards festgelegt. Mit Standards meint Steppi einen einheitlichen Programmrahmen, der für die Entwicklung verschiedener Lernanwendungen eingesetzt wird und z.B. Programmablauf, Benutzerführung und Funktionen in allen diesen Programmen festlegt.

In der Grundkonzeption werden die Lerninhalte definiert und die Grundstruktur des Programms festgelegt, die bei Steppi im wesentlichen einen systemgeführten, linearen Lernablauf darstellt. Die Feinkonzeption umfaßt die Ausarbeitung der Präsentation von Lernstoff, der Interaktionsfunktionen und der Erfolgskontrollen in jedem einzelnen Lernsschritt. Das Drehbuch ist die detailllierte Arbeitsanweisung zur Realisierung eines Lernprogramms, die in der Realisierungsphase in ein ablauffähiges Programm umgesetzt wird. Zum Abschluß der Lernprogramm-Erstellung erfolgt ein Test der technischen Funktionen des Programms.

Im Rahmen der Programm-Erprobung wird u.a. mit Hilfe von Feldtests die Akzeptanz des Lernprogramms überprüft, das schließlich für die Vervielfältigung und den Vertrieb freigegeben wird.

8.4.3 DIALEKT-Entwicklungsmodell für hypermediale Lernsysteme

Das DIALEKT (Digitale Interaktive Lektionen)-Entwicklungsmodell ist in den Wirtschaftswissenschaften u.a. von Apostolopoulos, Geukes und Zimmermann entwickelt worden.[358] Im Gegensatz zu den Vorgehensmodellen von Bodendorf und Steppi, die die Entwicklung von Lernanwendungen im allgemeinen beschreiben, zielt das in Abbildung 43 dargestellte DIALEKT-Modell speziell auf die Entwicklung hypermedialer Lernsysteme.

Apostolopoulos, Geukes und Zimmermann gehen davon aus, daß Lernen am Computer auf einer wirklichkeitsnahen Handlung bzw. „Story“ basiert.[359] Ein Beispiel für eine solche Handlung in der Anwendung „ODI (Optical Distortion Inc.)“ ist die Aufgabe, ein Marketingkonzept für eine fiktive Firma zu entwerfen, die Kontaktlinsen für Hühner verkaufen will.[360] Diese Handlung zieht sich als „roter Faden“ bzw. Hauptnavigationshilfe durch die gesamte Anwendung und wird am Anfang der

---

[357] Vgl. Steppi, 1990, S. 128.
[358] Vgl. Apostolopoulos, Geukes, Zimmermann, 1996.
[359] Vgl. Apostolopoulos, Geukes, Zimmermann, 1996, S. 2.
[360] Vgl. Dialekt (Digitale Interaktive Lektionen)-Projekt, Freien Universität Berlin, URL: http://dialekt.cedis.fu-berlin.de/dialekt.cfm?seite=produkte%5Codi%5Codi.cfm, Stand: 13.01.99.

Entwicklung festgelegt. Der gesamte Entwicklungsprozeß gliedert sich in die Hauptphasen „Konzeption“, „Medienproduktion“, „Programmierung“ und „Test“, die in mehreren Zyklen durchlaufen werden, bis die Entwicklung abgeschlossen ist.

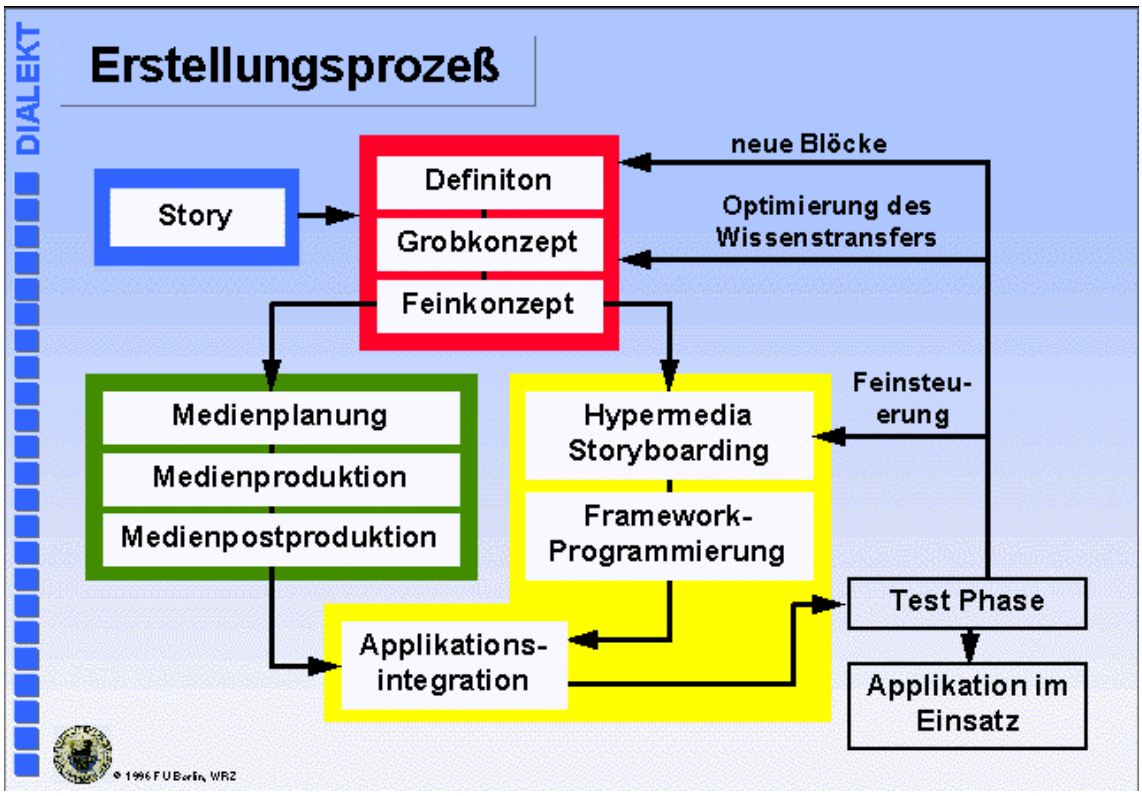

Abbildung 43: DIALEKT-Entwicklungsmodell

In der Konzeptionsphase erfolgt die Definition der Projektziele, die Beschreibung der technischen und organisatorischen Rahmenbedingungen sowie eine erste Analyse von Zielgruppen und Lernzielen. Im Grobkonzept wird eine grobe Stukturierung und Abgrenzung des in der Anwendung abzubildenden Wissens vorgenommen. Weiterhin wird die Lehrstrategie festgelegt und die zu vermittelnden Lerninhalte in einzelne Informationseinheiten aufgeteilt. Das Feinkonzept enthält eine detaillierte Beschreibung von Präsentations- und Interaktionsideen, die von den Programmierern als Vorlage für die Implementierung verwendet werden. Die Ergebnisse der Konzeptionsphase fliessen in Form der Medienplanung in die Medienproduktion und in Form eines Drehbuchs bzw. Hypermedia-Storyboards in die Programmierung ein.

Die Medienproduktion umfaßt u.a. die Erstellung von Bildern, Graphiken, Animationen und Videos sowie deren Nachbearbeitung bzw. Postproduktion wie z.B. Filmschnitt und -digitalisierung. In der Programmierungsphase werden die unterschiedlichen Medien zu Bildschirmseiten bzw. „Frames“ zusammengesetzt und die Interaktion und Navigation implementiert. In der Testphase wird u.a. die Programm-Bedienbarkeit und die Qualität der Wissensdarstellung evaluiert und gegebenenfalls verbessert.

8.4.4 Fazit

Die Modelle von Bodendorf, Steppi und Apostolopoulos et al. sind nur zum Teil die Erstellung von Lernanwendungen im Hochschulstudium geeignet, weil diese Modelle u.a. den Einsatz von professionellen Programmierern und Graphikern voraussetzen. Dagegen wird die Entwicklung z.B. im Projekt VetMedia in Berlin normalerweise mit

unerfahrenen Doktoranden durchgeführt. Diese können am Anfang eines Projektes in der Regel kaum ein pädagogisches Design, ein Feinkonzept oder Drehbuch erstellen, wie dies von den anderen Modellen vorgesehen wird. Deshalb ist die Einarbeitung der Doktoranden z.B. in die Bedienung der Entwicklungswerkzeuge wie HTML ein Hauptbestandteil des Entwicklungsprozesses. Diese Einarbeitungsphase kommt in den oben genannten drei Modellen nicht vor.

8.5 Explizite Darstellung des impliziten VetMedia-Vorgehensmodells

An dem in dieser Fallstudie untersuchten Fachbereich ist im Projekt VetMedia im Verlauf von mehreren Jahren durch Experimentieren ein implizites Vorgehensmodell für die Entwicklung von Multi-/Hypermedia-CD-ROM- und Internet-Anwendungen gefunden worden, dem unbewußt gefolgt wird und das sich auch in anderen Hochschulprojekten einsetzen läßt. Um das implizite VetMedia-Modell z.B. zur Kommunikation zwischen den Entwicklern über den Stand eines Entwicklungsprozesses verwenden zu können, muß es explizit beschrieben werden. Die Aufgabe dieser expliziten Darstellung liegt u.a. darin, die zum Teil unbenannten Elemente des Entwicklungsvorganges zu benennen, erläutern und systematisch zu gliedern. Die Bezeichnungen dieser Elemente werden im folgenden fettgedruckt dargestellt, um sie zu kennzeichnen, einzuführen und zu fokusieren. Die explizite Darstellung des impliziten VetMedia-Vorgehensmodells besteht im wesentlichen aus den in Abbildung 44 dargestellten Phasen und Arbeitsschritten.

8.5.1 Planung

Nach diesem Modell wird die Entwicklung von Anwendungsprogrammen normalerweise durch die Idee eines Hochschullehrers oder Doktoranden oder den Auftrag eines Wirtschaftsunternehmens initiiert. So ist z.B. die Anwendung „Internetbasiertes Vorlesungsskript Tiergeburtshilfe“ aufgrund der Idee entstanden, dieses Programm zur Vermittlung von Grundlagenwissen einzusetzen, um dadurch mehr Zeit für die Diskussion aktueller Fragestellungen im Unterricht zu schaffen.

Nach dem Lehrbuch „Software-Engineering“ von Kimm, Koch, Simonsmeier und Tontsch[361] bildet ein konkretes **Problem** bei der Durchführung einer bestimmten Aufgabe den Ausgangspunkt für die Entwicklung eines Computer-Programms. Ziel dieser Entwicklung ist es, ein Programm zu erstellen, mit dem sich das Problem lösen und die jeweilige Aufgabe durchführen läßt. Die erste Phase des von Kimm, Koch, Simonsmeier und Tontsch beschriebenen Software-Lebenszyklus-Modells besteht deshalb u.a. aus einer **Problemanalyse**, in der das zu lösende Problem vollständig und eindeutig beschrieben wird. Aus dieser Beschreibung werden detaillierte **Anforderungen** für die Entwicklung eines Programms abgeleitet.

Die Entwicklung von Anwendungen an Hochschulen basiert nicht in erster Linie auf Problemen, deren Lösung durch den Einsatz von Multi-/Hypermedia- und Internet-Anwendungen dringend erforderlich ist, so daß die Anwendungen demzufolge unbedingt von Hochschullehrern und Studenten benutzt werden. Unerfahrene Anwender z.B. in der Tiermedizin können nur zum Teil Probleme und Anforderungen formulieren, u.a. weil ihnen die Möglichkeiten der Computer-Technologien weitgehend unbekannt sind und weil sie nicht wissen, daß die Anwendungen Probleme lösen können. Die

[361] Vgl. Kimm, Koch, Simonsmeier, Tontsch, 1979, S. 19.

Entwicklung von Anwendungen dient daher in erster Linie dazu, die Möglichkeiten der Technologien kennenzulernen.

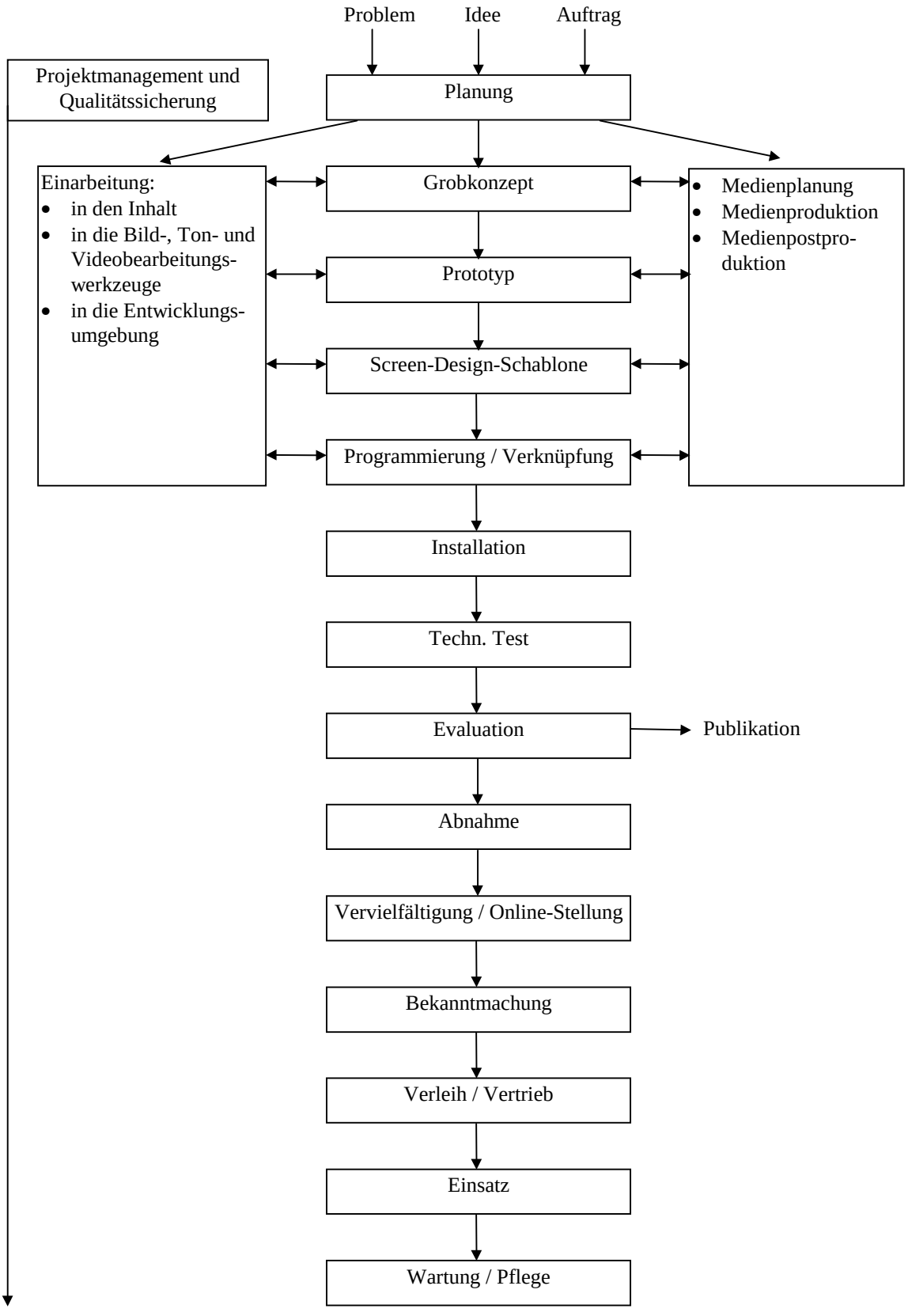

Abbildung 44: Phasen und Arbeitsschritte des impliziten VetMedia-Modells

Während der Planungsphase des Entwicklungsprozesses werden im VetMedia-Projekt normalerweise verschiedene Dokumente erstellt, für die es keine feststehenden Bezeichnungen gibt, sondern die von Fall zu Fall als **Expose**, **Projektplan**, **Konzept**, **Projektidee** oder **Projektvorschlag** bezeichnet werden. Auch gibt es keine Vorgaben hinsichtlich Inhalt und Struktur dieser Dokumente, so daß sie von Projekt zu Projekt unterschiedliche Informationen enthalten, die hier am Beispiel der Projektidee für die Anwendung „Tiergeburtshilfe“ wiedergegeben werden:

- **Titel bzw. Thema**: Internetbasiertes Vorlesungsskript Tiergeburtshilfe
- **Zielsetzung**: Bereitstellung des Inhalts der Vorlesung Tiergeburtshilfe im Internet zur Vor- und Nachbereitung der Lehrveranstaltung sowie zur Prüfungsvorbereitung
- **Zielgruppe**: Studenten der Tiermedizin vor allem an der Freien Universität Berlin, die an der Vorlesung im fünften Semester und an der praktischen Übung im neunten Semester teilnehmen
- **Zielplattform:** Die Anwendung wird für die Benutzung im Internet entwickelt, so daß sie auf PCs, Macintosh-Rechner und UNIX-Workstations abgerufen werden kann. Für Benutzer ohne Internet-Zugang wird die Anwendung zusätzlich auf CD-ROM übertragen.
- **Projektdauer**: etwa ein Jahr
- **Hard- und Software**: Pentium I-PC, WWW-Server wie z.B. Microsoft Internet Information Server®, WWW-Browser wie z.B. Netscape Navigator®, HTML-Editor wie z.B. Softquad HoTMetaL®
- **Mitarbeiter**:
    - Doktorand der Tiermedizin für die Darstellung des Inhalts in HTML
    - Informationswissenschaftler für die Programmierung von Java-Applets und die Verwaltung des WWW-Servers
    - Mediendesigner für die Erstellung von Graphiken für die Benutzeroberfläche
- **Erklärung zum Verbleib von Urheberrechten** bei dem Leiter des VetMedia-Projektes bzw. der Freien Universität Berlin

In den Vorgehensmodellen für die Software-Entwicklung in der Informatik wird als Ergebnis einer Entwicklungsphase normalerweise die Fertigstellung von bestimmten Dokumenten wie **Lastenheft**, **Projektplan** und **Durchführbarkeitsuntersuchung** verlangt, bevor mit den Aktivitäten in der nächsten Phase begonnen werden kann. Für den Aufbau, den Inhalt und zum Teil sogar die Gestaltung dieser Dokumente gibt es vorgegebene Richtlinien. So werden z.B. in dem V-Modell, einem Vorgehensmodell, dessen Verwendung bei Software-Entwicklungen für die Bundesbehörden und die Bundeswehr vorgeschrieben ist, auf etwa 90 DIN A4-Seiten die Gliederung, der Inhalt und das Seitenlayout von mehr als 30 verschiedenen Dokumenten beschrieben, die im Verlauf einer Software-Entwicklung nach diesem Modell zu erstellen sind.[362] Dadurch soll eine standardisierte Abwicklung von Software-Entwicklungsprojekten und die Kontrolle des Projektfortschrittes an den Phasenabschlüssen ermöglicht werden. So können Probleme rechtzeitig erkannt und durch Eingreifen der fristgerechte und erfolgreiche Abschluß des Entwicklungsprozesses sichergestellt werden.[363] Balzert sieht bei dieser dokumentengesteuerten Software-Entwicklung die Gefahr des Entstehens einer „Software-Bürokratie“[364], in der die Erstellung der Dokumente wichtiger wird als die Erstellung des Programms.

---

[362] Vgl. Der Bundesminister des Inneren, 1995.
[363] Vgl. z.B. Kimm/Koch/Simonsmeier/Tontsch, 1979, S. 18-19 und Balzert, 1996, S. 39.
[364] Vgl. Balzert, 1998, S. 113.

Eine so umfangreiche Dokumentation des Entwicklungsprozesses wie in dem V-Modell ist für Projekte im Studium normalerweiser nicht notwendig, weil die zu entwickelnden Anwendungen nicht so komplex sind wie Programme für die Bundesbehörden oder die Bundeswehr. Trotzdem sind standardisierte Vorlagen für die Dokumentation des Entwicklungsprozesses auch an Hochschulen eine Unterstützung, weil sie bewirken, daß Programmanforderungen wie z.B. der darzustellende fachliche Inhalt sowie die zu programmierenden Funktionen schriftlich fixiert werden. Die genaue Beschreibung solcher Programmanforderungen ist insbesondere bei der Zusammenarbeit mit Unternehmen aus der Wirtschaft notwendig, weil es bei unklaren Anforderungsdefinitionen zu Streitigkeiten kommen kann, ob eine bestimmte Funktion zum ursprünglich vereinbarten Leistungsumfang gehört oder extra zu bezahlen ist. Als Vorlage für die Beschreibung von Programmanforderungen kann z.B. das von Balzert vorgeschlagene Pflichtenheft dienen, das in dem „Lehrbuch der Software-Technik“ ausführlich erläutert wird.[365]

8.5.2 Einarbeitung

In dieser Phase arbeiten sich die als Entwickler beschäftigten Studenten bzw. Doktoranden in den jeweiligen darzustellenden fachlichen Inhalt wie z.B. das Thema Tiergeburtshilfe und die Bedienung der Entwicklungswerkzeuge wie z.B. das Autorensystem Multimedia ToolBook®, den HTML-Editor HoTMetaL® von Softquad oder das Bildbearbeitungsprogramm Adobe Photoshop® ein. Die Einarbeitung in den fachlichen Inhalt ist notwendig, weil die Doktoranden zwar normalerweise im Studium Lehrveranstaltungen zu dem jeweiligen Thema besucht haben, aber nicht genügend Wissen besitzen, um eine Multi-/Hypermedia- oder Internet-Anwendung zu diesem Thema zu erstellen. Zum Einstieg in den Inhalt werden Lehrbücher gelesen, Kurse besucht und die Hochschullehrer des jeweiligen Fachgebietes befragt. Die Einarbeitung in die Bedienung der Entwicklungswerkzeuge ist notwendig, weil Doktoranden in der Tiermedizin normalerweise kaum Erfahrung mit diesen Werkzeugen besitzen. Der Einstieg in die Verwendung der Werkzeuge erfolgt meist durch eine praktische Einführung am Rechner z.B. durch einen Informationswissenschaftler oder einen Tiermediziner mit entsprechender Erfahrung, durch den Besuch von Kursen, durch die Lektüre von Einführungs- und Benutzerhandbüchern sowie durch praktisches Üben am Rechner. Eine weitere Methode zur Einarbeitung ist die Anregung durch Beispiele aus anderen Anwendungen. Der Einarbeitungsprozeß dauert in der Regel bis zur Fertigstellung der Anwendung, weil die Doktoranden während des gesamten Projektes neue Erfahrungen sammeln.

8.5.3 Grobkonzept

In dieser Phase wird für den fachlichen Inhalt der zu entwickelnden Anwendung ein **Grobkonzept** erstellt, das in der Regel aus der Auflistung von verschiedenen Themen und Unterthemen besteht, deren hierarchische Struktur in einem Diagramm graphisch dargestellt wird. Abbildung 45 zeigt ein Grobkonzept am Beispiel der Anwendung „Tiergeburtshilfe“, das u.a. die Themen der gleichnamigen Vorlesung enthält und im Verlauf der Entwicklung durch Unterthemen ergänzt wird, wie dies beim Thema „Geburtshilflicher Untersuchungsgang“ angedeutet ist.

---

[365] Vgl. Balzert, 1996, S. 106-109.

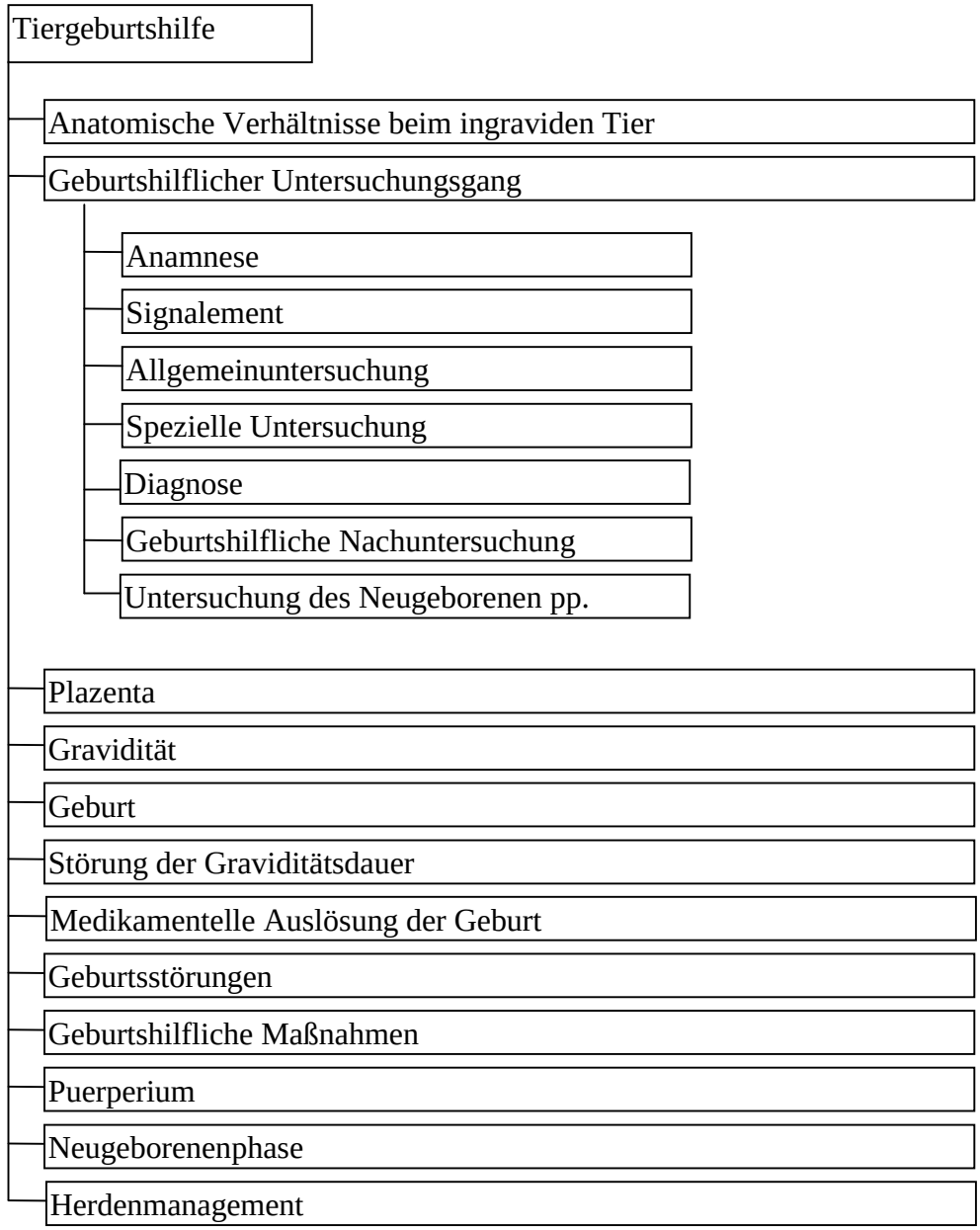

Abbildung 45: Grobkonzept für den Inhalt der Internet-Anwendung „Tiergeburtshilfe“

In der Informatik wird u.a. das objektorientierte Modell verwendet, um die in einem Weltausschnitt vorkommenden Objekte, ihre Eigenschaften und Verhaltensweisen und Beziehungen zueinander im Computer abzubilden. Dieser Weltausschnitt wird auch als **Miniwelt** bezeichnet. Ein Beispiel für eine Miniwelt ist die Tiergeburtshilfe. Ein **Objekt** im Sinne des objektorientierten Modells ist ein Formalismus, mit dem sich ein Objekt aus der realen Welt wie z.B. der geburtshilfliche Untersuchungsgang direkt durch ein Objekt „Geburtshilflicher Untersuchungsgang“ im Rechner darstellen läßt. Mit dem objektorientierten Modell können sowohl die Eigenschaften als auch die Verhaltensweisen bzw. Methoden eines realen Objektes abgebildet werden. So besitzt z.B. das Objekt „Geburtshilflicher Untersuchungsgang“ das **Attribut** „Ziel“, das darin besteht, die Ursachen einer eventuellen Geburtsstörung zu erkennen, eine klare Diagnose zu formulieren und eine optimale Behandlung durchzuführen. Weiterhin besitzt das Objekt „Geburtshilflicher Untersuchungsgang“ u.a. die **Methoden** „Anamanese“, „Signalement“ und „Allgemeinuntersuchung“, die beschreiben, wie der Untersuchungsgang auszuführen ist. Ein weiteres Objekt in der Miniwelt

„Tiergeburtshilfe“ ist die Vorlesung zu diesem Thema. Die Objekte „Vorlesung Tiergeburtshilfe“ und „Geburtshilflicher Untersuchungsgang“ stehen in einer Beziehung zueinander, weil der Untersuchungsgang ein Thema der Vorlesung ist. Diese Beziehung zwischen den beiden Objekten wird z.B. durch das Attribut „Thema“ in dem Objekt „Vorlesung Tiergeburtshilfe“ dargestellt, wobei diese Eigenschaft u.a. den Wert „Geburtshilflicher Untersuchungsgang annehmen kann. Objekte mit gleichen Eigenschaften können zu sogenannten **Klassen** wie z.B. der Klasse „Vorlesung“ zusammgengefaßt und im Rechner dargestellt werden. Dadurch müssen Attribute und Methoden, die verschiedenen Objekten gemeinsam sind, nur einmal in der Klasse und nicht für jedes Objekt einzeln repräsentiert werden. So braucht z.B. das Attribut „Thema“, nur einmal in der Klasse „Vorlesung“ dargestellt zu werden und gilt dann automatisch für alle Vorlesungsobjekte. Auf diese Weise läßt sich mit Hilfe von Objekten ein Ausschnitt aus der Welt der Tiermedizin detailliert im Rechner repräsentieren.

Ein Hauptvorteil des objektorientierten Modells ist die durchgängige Anwendbarkeit des Modells in den Phasen Analyse, Entwurf und Implementierung des Software-Lebenszyklus durch die Methoden der objektorientierten Analyse (OOA)[366], des objektorientierten Design (OOD)[367] und der objektorientierter Programmierung (OOP)[368], die z.B. von Coad und Yourdon beschrieben werden. Das objektorientierte Modell ist eines der wichtigsten Modelle der Informatik in den neunziger Jahren, weil die meisten neuen Entwicklungswerkzeuge wie z.B. die Programmiersprache Java™, die CORBA (Common Object Request Broker)-Spezifikation für die Kommunikation zwischen objektorientierten Programmen auf verschiedenen Rechnern und das objektrelationale Datenbankmanagementsystem Oracle 8® auf diesem Modell beruhen. Wenn das objektorientierte Modell das vorherrschende Paradigma in der Informatik darstellt, ist wahrscheinlich auch in der Tiermedizin und anderen Fächern die Verwendung des objektorientierten Modells für die Analyse, den Entwurf und die Programmierung von Lernanwendungen zu empfehlen.

8.5.4 Prototyp

In den meisten Vorgehensmodellen für die Multi-/Hypermedia-Entwicklung wie z.B. dem DIALEKT-Modell[369], dem Phasenmodell der Multimedia-Entwicklung bei der Firma Pixelpark[370] und dem Modell des CBT-Entwicklungsproesses bei Steppi[371] folgt auf die Grobkonzeptphase eine Feinkonzeptphase, in der die Gestaltung, der Inhalt und die Funktionen der Anwendung für jede einzelne Bildschirmseite in einem Drehbuch detailliert beschrieben werden, bevor mit der Implementierung begonnen wird. Dagegen wird im VetMedia-Vorgehensmodell normalerweise nach dem Grobkonzept zunächst ein **Prototyp** der Anwendung erstellt. Das Prototyping bezeichnet in der Informatik u.a. die Erstellung von ablauffähigen Vorversionen des zukünftigen Produkts vor der Implementierungsphase.[372] Auf Grund der mangelnden Erfahrung von Studenten und Doktoranden bei der Entwicklung von Multi-/Hypermedia- und Internet-Anwendungen ist die Erstellung eines detaillierten Feinkonzeptes meist nicht möglich, da die

366 Vgl. z.B. Coad, Yourdon, 1991.
367 Vgl. z.B. Coad, Yourdon, 1991b.
368 Vgl. z.B. Coad, Nicola, 1993.
369 Vgl. Apostolopulos, Geukes, Zimmermann, 1996, S. 10.
370 Vgl. Degen, 1996, S. 34.
371 Vgl. Steppi, 1990, S. 129.
372 Vgl. Balzert, 1998, S. 115.

Möglichkeiten von Entwicklungsumgebungen wie z.B. HTML weitgehend unbekannt sind und daher auch kaum umsetzbare Konzepte erstellt werden können. Stattdessen muß diese Erfahrung mit der Entwicklungsumgebung u.a. durch die Entwicklung eines Prototypen gewonnen werden. Diese Prototypen dienen beispielsweise der Kommunikation zwischen den Entwicklern und Auftraggebern aus der Wirtschaft vor allem bei der Auswahl unterschiedlicher Gestaltungsvarianten für die Oberfläche. Mit Hilfe von Autorensystemen wie Macromedia Authorware® sowie HTML-Editoren wie Netscape Pagecomposer® können Prototypen innerhalb von Tagen erstellt werden. Bröhl und Dröschl nennen u.a. folgende weitere Zielsetzungen des Prototyping:[373]

- Unterstützung, Präzisierung und Korrektur der Anforderungsanalyse
- Durchführung von Realisierbarkeitsuntersuchungen
- Präzisierung und Verifikation des Entwurfs
- Schaffung einer Grundlage für die Kommunikation der Entwickler mit den Anwendern
- Beschaffung von Informationen über die Leistungsdaten und Hardware-Anforderungen eines geplanten Systems
- Beschaffung von Informationen über die Eignung von Software-Entwicklungsumgebungen für die Durchführung eines bestimmten Projektes

Zu den Nachteilen des Prototyping gehören u.a. die oben im Zusammenhang mit dem „Code & Fix“-Modell beschriebenen Probleme wie z.B. der chaotische, weil nicht systematisch geplante Programmaufbau.

Bei der Entwicklung eines Prototypen wird zunächst jedes Thema aus dem Grobkonzept durch eine Bildschirmseite in einem Autorensystem bzw. durch ein HTML-Dokument in einer Internet-Anwendung repräsentiert. Diese Bildschirmseiten bzw. HTML-Dokumente werden mit Texten und Bildern gefüllt und durch Hypertext-Verknüpfungen miteinander verbunden.

8.5.5 „Screen Design“-Schablone

Für die Benutzeroberfläche des Prototypen werden normalerweise drei bis fünf **Layout**- bzw. „**Screen Design**“-Varianten von einem Mediendesigner entworfen, von denen eine Variante für die endgültige Gestaltung ausgewählt wird. Dieses Layout dient dann als **Schablone** für die Plazierung von Texten, Photos, Graphiken und Videos sowie von Navigationssymbolen wie z.B. „zum Hauptmenü“. Mit Hilfe einer solchen Schablone können Fachwissenschaftler weitgehend unabhängig von Programmierern und Designern beliebig viele Seiten der gleichen Art erstellen, die sie dann jeweils mit Inhalten füllen. Die Gestaltung der Schablonen wird in der Regel so angelegt, daß sie eine Einheit von Form und Inhalt ausdrückt. So ist z.B. für eine Anwendung zum Thema „Rund- und Bandwürmer bei Hund und Katze“ eine „Screen Design“-Schablone entworfen worden, die das Praxisschild eines Tierarztes und eine Klingelleiste an der Hauswand darstellt, weil die Behandlung von Hunden und Katzen mit Rund- und Bandwürmern in der Regel bei Kleintierärzten erfolgt, an deren Haustür solche Schilder zu finden sind. Abbildung 46 zeigt die Schablone, in die Titel bzw. Überschrift, Inhalt und Verknüpfungen zu anderen Themen bei der Erstellung einer Bildschirmseite eingefügt werden.

---

[373] Vgl. Stary, 1996, S. 230; Bröhl, Dröschl, 1995, S. 100 und Der Bundesminister des Inneren, 1992, S. 3-9.

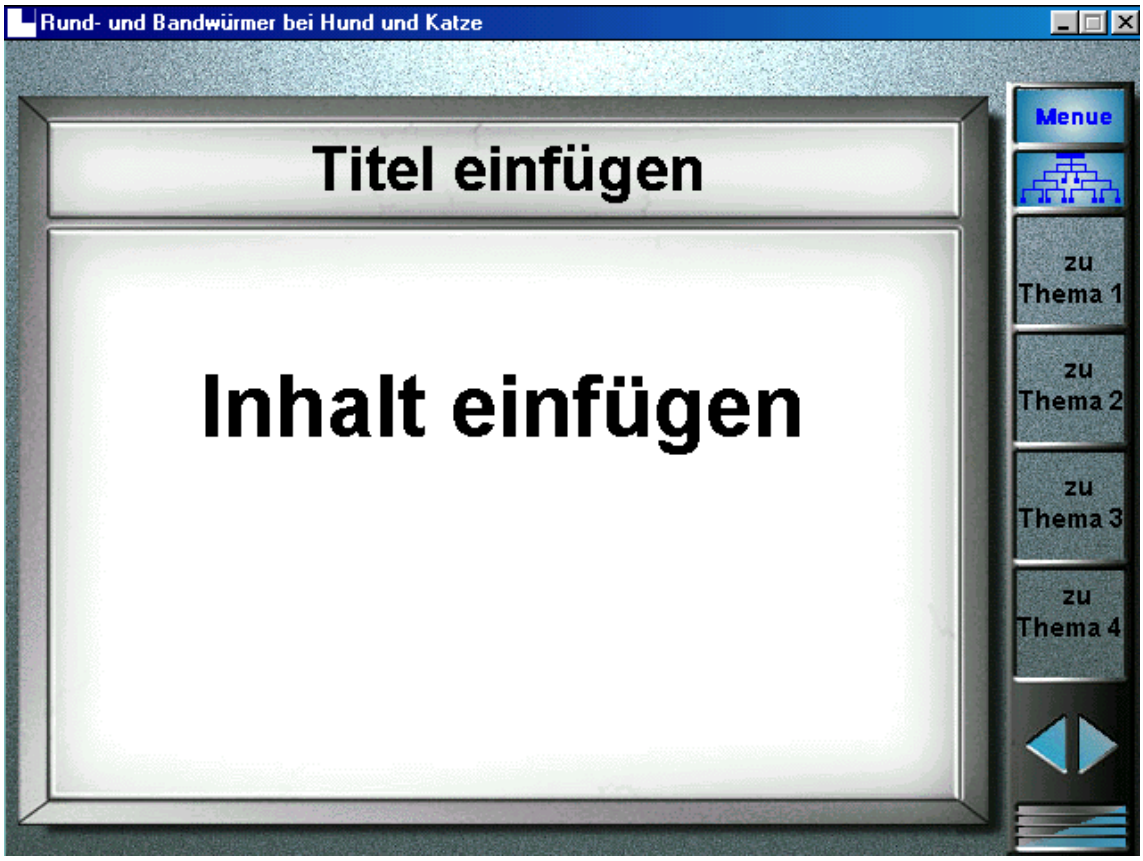

Abbildung 46: „Screen Design“-Schablone für die Anwendung „Rund- und Bandwürmer bei Hund und Katze“

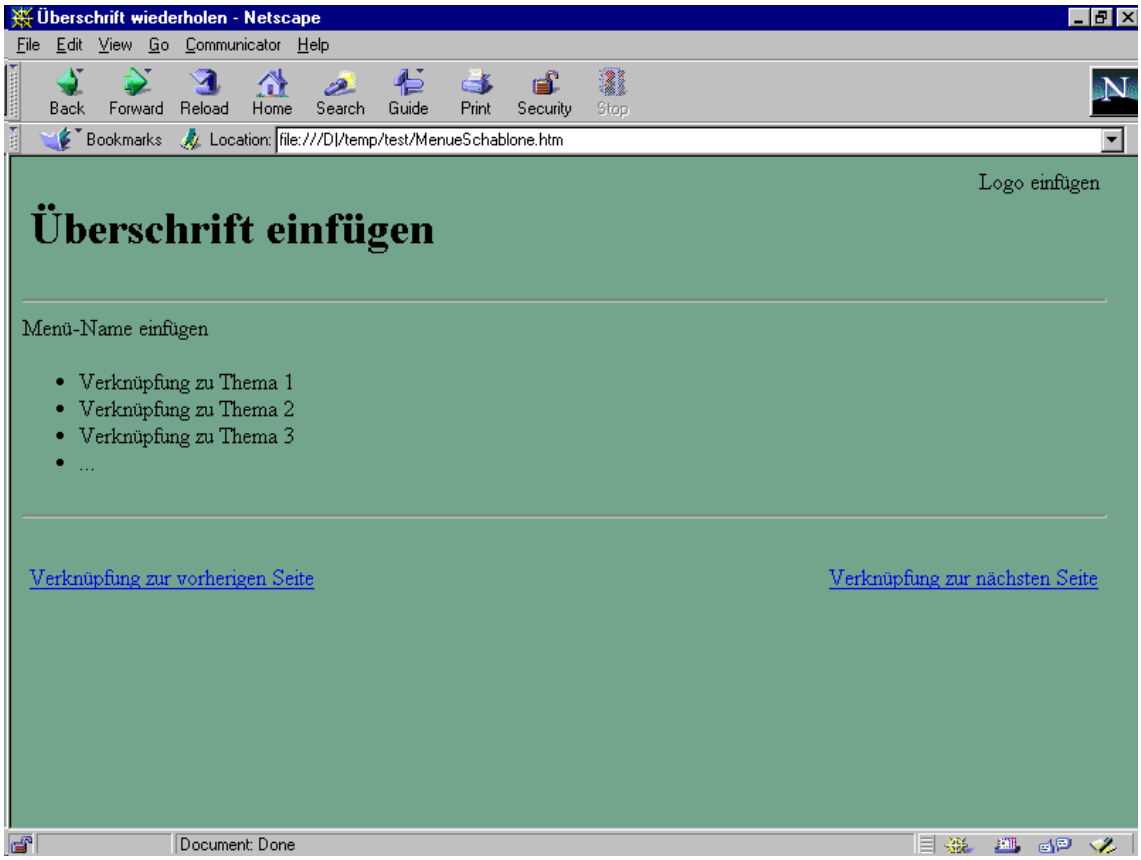

Abbildung 47: HTML-Schablone für die Erstellung von Menü-Dokumenten in der Anwendung „Tiergeburtshilfe“

Abbildung 48: HTML-Schablone für die Erstellung von Inhalts-Dokumenten

Auch für Internet-Anwendungen wie das „Vorlesungsskript Tiergeburtshilfe“ lassen sich solche Schablonen in Form von HTML-Dokumenten erstellen. Die Schablone in Abbildung 47 dient der Erstellung von Menü-Dokumenten zur Auswahl von Themen. Eine weitere Schablone in Abbildung 48 wird zur Erstellung von Inhalts-Dokumenten mit fachlichen Informationen benutzt. Die Verwendung dieser Schablonen zur Erstellung von HTML-Dokumenten umfaßt u.a. folgende Schritte:

1. Laden der Schablone in einen HTML-Editor wie z.B. HoTMetaL®
2. Ersetzen der Platzhalter wie z.B. „Verknüpfung zum Hauptmenü“, „Überschrift einfügen“ und „Inhalt einfügen“ mit den anwendungsspezifischen Angaben
3. Abspeichern der ausgefüllten Schablone unter einem neuen Dateinamen

Mit Hilfe der oben gezeigten Schablonen für Menü- und Inhaltsseiten läßt sich z.B. die folgende hierarchische Anwendungsstruktur aufbauen:

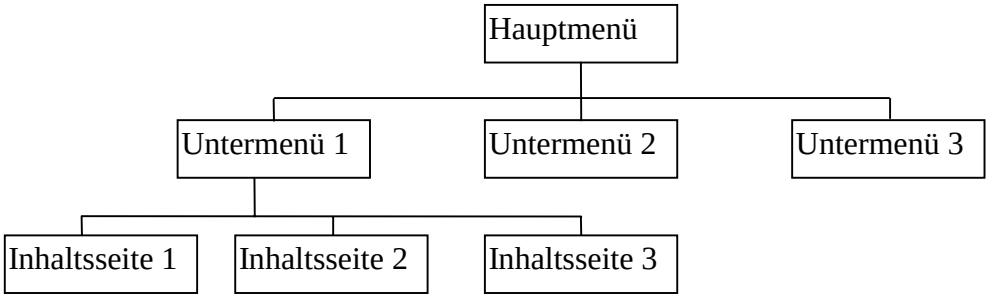

Abbildung 49: Aufbau einer Anwendung durch zwei Arten von Schablonen

In Form von sogenannten „templates“ werden solche Schablonen z.B. auch vom CLIVE-(Computer-based Learning in Veterinary Education)-Konsortium in Großbritannien entwickelt.[374]

Wenn die Texte, Bilder und Videosequenzen von Lernanwendungen in einer Datenbank gespeichert und diese Inhalte zur Laufzeit in die „Screen Design“-Schablonen eingelesen werden, lassen sich Bildschirmseiten in CD-ROM-Anwendungen bzw. HTML-Dokumente in Internet-Anwendungen dynamisch erstellen. Der Autor hat diese Vorgehensweise u.a. am Beispiel einer ToolBook®-Anwendung zum Thema „Brunstzyklus beim Rind” auf der Grundlage des relationalen Datenbankmanagementsystems Microsoft Access®[375] sowie am Beispiel der Internet-Anwendung „Bayovac Website“ auf der Grundlage des relationalen Datenbankmanagments Oracle® untersucht. Der Vorteil dieser Verfahrensweise liegt u.a. darin, daß die Inhalte aus der Datenbank und das in den Schablonen festgelegte Layout getrennt werden. So läßt sich einerseits das Layout in den Schablonen ändern, ohne daß hunderte von HTML-Seiten editiert werden müssen. Die Änderung des Layouts von Web-Seiten kann z.B. durch Änderungen der „corporate identity“ eines Fachbereichs oder durch neue Moden und Zeitgeist-Trends notwendig werden. Andererseits lassen sich die Informationen in der Datenbank mit Hilfe von Web-Formularen bequem ändern, ohne daß man HTML-Dateien aufrufen muß, in denen die Informationen zwischen Layout-Anweisungen kaum aufzufinden sind. Ein Nachteil der Verwendung von Datenbanken zur Erstellung von Anwendungen ist die Verzögerung des Seitenaufbaus, weil die Seiten erst zur Laufzeit zusammengesetzt werden. Mit Hilfe von Werkzeugen wie z.B. ColdFusion® von Allaire Corp. können Internet-Anwendungen erstellt werden, bei denen die HTML-Dokumente einmal aus der Datenbank erzeugt werden und dann als statische Seiten angezeigt werden. Nur wenn sich der Inhalt der Datenbank ändert, werden neue HTML-Seiten generiert. Auf diese Weise läßt sich die Flexibilität einer Datenbank mit der Geschwindigkeit der Darstellung von normalen HTML-Dateien kombinieren.

8.5.6 Medienplanung, Medienproduktion und Medienpostproduktion

In der Phase „**Medienplanung, Medienproduktion und Medienpostproduktion**“ werden u.a. folgende Aktivitäten zur Planung, Erstellung und Nachbearbeitung von Texten, Bildern, Audio-, Video- und Animationssequenzen durchgeführt:

- Recherche und Beschaffung von vorhandenem Material
- Sichtung des vorhandenen Materials
- Beurteilung der Verwendbarkeit des vorhandenen Materials z.B. im Hinblick auf die Beachtung von Urheberrechten
- Auflistung noch fehlenden bzw. zu produzierenden Materials
- Erstellung von Texten
- Bildproduktion, Bilddigitalisierung und Bildbearbeitung
- Erstellung von Bedienungs- und Navigationselementen
- Audioproduktion, Audiodigitalisierung und Audiobearbeitung
- Videoproduktion, Videodigitalisierung und Videobearbeitung
- Erstellung von Animationssequenzen

---

[374] Vgl. Abschnitt 6.1.1und CLIVE (Computer-Assisted Learning in Veterinary Education) URL: http://www.clive.ed.ac.uk/web/queries/catalog/cattyp.idc?TYP_id=OT, Stand: 13.05.98.

[375] Vgl. Hallmann, Heuwieser, 1995.

Für die Erstellung von Audio- und Videosequenzen wird in der Regel eine spezielle Ausstattung benötigt wie z.B. ein Tonstudio mit einem Sprecher für die Aufzeichnung von gesprochenen Texten und ein Computer mit einer Karte für die Digitalisierung und Komprimierung von Videosequenzen. Eine entsprechende Ausstattung steht normalerweise an den Universitäts-Medienzentren zur Verfügung. Die Anschaffung eines speziellen Arbeitsplatzrechners für die Digitalisierung von AVI- und MPEG (Motin Picture Expert Group)-Videosequenzen, der u.a. mit 256 MB Arbeitsspeicher, 27 GB Festplattenspeicher, einer Miro DC 30 Plus Videokarte und Adobe Premiere® für den Videoschnitt ausgerüstet ist, kostet etwa 20.000 DM.

Obwohl zu Beginn der Phase „Medienplanung, -produktion und -postproduktion“ die eine Liste der für die Anwendungsentwicklung benötigten Materialien erstellt wird, ist diese Auflistung normalerweise unvollständig, weil sich aus dem Einbau der ersten fertiggestellten Medien in den Prototypen immer wieder Ideen für die Erstellung neuer Medien ergeben, so daß diese Phase fast bis zur Fertigstellung des Programms dauern kann.

8.5.7 Anwendungsprogrammierung und -verknüpfung

In dieser Phase werden die erstellten Texte, Bilder, Audio-, Video- und Animationssequenzen in die zuvor erstellte Layout-Schablone eingebaut und die einzelnen Bildschirmseiten durch Hypertext-Verknüpfungen miteinander verbunden. Weiterhin werden Interaktionselemente wie z.B. mit der Maus zu erforschende Graphiken programmiert und der Prototyp auf diese Weise sukzessiv zum vollständigen Anwendungsprogramm weiterentwickelt.

Zwischen korrespondierenden Informationen in verschiedenen Anwendungen können durch Hypertext-Verknüpfungen fächerübergreifende Zusammenhänge hergestellt werden. Ein Beispiel dafür ist die geplante Verknüpfung von Informationen zum Thema „Beckenanatomie“ in einem Programm zur angewandten Anatomie des Rindes mit Informationen zur Bedeutung der Beckenanatomie bei der Geburt von Tieren in einem Programm zur Tiergeburtshilfe. Durch diese Verknüpfungen können z.B. die Studenten, die sich im klinischen Studienabschnitt mit der Tiergeburtshilfe beschäftigen, ihr vorklinisches Wissen zur Anatomie des Rindes auffrischen. Die Studenten im vorklinischen Studienabschnitt lernen auf diese Weise die praktischen Anwendungsmöglichkeiten des anatomischen Grundwissens in der Tiergeburtshilfe kennen und werden dadurch bei der Aneigung dieses Wissens motiviert.

Ein Problem bei der Herstellung dieser fächerübergreifenden Zusammenhänge ist die Auffindung der korrespondierenden Informationen in den verschiedenen Anwendungen, weil dies voraussetzt, daß man den Inhalt der einzelnen Anwendungen kennt, was nicht immer der Fall ist. Davis, Hall, Heath, Hill und Wilkins beschreiben eine mögliche Lösung dieses Problems im Rahmen des Microcosm-Systems durch sogenannte „Text Retrieval Links“[376]. Diese Hypertext-Verknüpfungen werden zur Laufzeit dynamisch berechnet, indem mit Hilfe von Information Retrieval-Funktionen beim Anklicken eines Begriffs korrespondierende Dokumente gesucht und aufgelistet werden, zu denen der Benutzer dann navigieren kann.

[376] Vgl. Davis, Hall, Heath, Hill, Wilkins, 1992.

8.5.8 Installation

Bei der Entwicklung von Multi-/Hypermedia-CD-ROM-Anwendungen wird in dieser Phase ein „Setup“-Programm erstellt, das die Installation der Anwendung unterstützt. Damit der Benutzer den Installationsvorgang Schritt für Schritt nachvollziehen kann, wird eine Installationsanleitung geschrieben, die jede vom Benutzer durchzuführende Eingabe erläutert. Schließlich werden die Anwendung und das „Setup“-Programm mit Hilfe von CD-Recording-Software wie z.B. Gear® von Elektroson auf eine CD-ROM übertragen. Mit Hilfe dieser CD-ROM wird die Anwendung für die nachfolgende Evaluation auf verschiedenen Rechnern installiert.

Bei der Entwicklung von Internet-Anwendungen ist die Erstellung eines „Setup“-Programms nicht notwendig, weil diese Anwendungen nur einmal auf einen WWW-Server übertragen und von dort aus weltweit über das Internet abgerufen werden.

8.5.9 Technischer Test

Ziel des Anwendungstests ist die Auffindung und Beseitigung von technischen Problemen und inhaltlichen Fehlern in einer Anwendung, wie die Beseitigung von sogenannten „blinden“ Hypertext-Verknüpfungen, die beim Anklicken zur Anzeige der Fehlermeldung „HTTP/1.0 404 Objekt nicht gefunden“ führen, weil das Zieldokument nicht an dem angegebenen Ort gefunden wurde. Der Test von selbstentwickelten Programmen kann z.B. mit Studenten im Computer-Arbeitsraum durchgeführt werden. Die Studenten notieren die von ihnen gefundenen Fehler in einem Protokoll, damit sie später von den Entwicklern nachvollzogen und behoben werden können.

8.5.10 Evaluation

Wenn die Anwendungsentwicklung im Rahmen eines Forschungsprojektes wie z.B. einer Dissertation erfolgt, findet normalerweise nach der Fertigstellung des Programms eine Evaluation statt, in der u.a. die Effektivität und Akzeptanz der Anwendung bei den Studenten untersucht wird. Dabei werden zum Teil auch die Anforderungen der Studenten an das Programm erhoben. So untersucht z.B. Regula in der Evaluation der CD-ROM-Anwendung „Brunstzyklus beim Rind“, ob die Studenten dieses Programm zur Vor- oder Nachbereitung von Vorlesungen, Kursen oder Praktika einsetzen würden.[377] Diese Ermittlung der Programmanforderungen bei den Benutzern soll nach dem Lebenszyklus-Vorgehensmodell aus der Informatik bereits in der Definitionsphase am Anfang der Entwicklung durchgeführt werden, damit sich die Entwicklung an den tatsächlichen Bedürfnissen der Anwender orientieren kann. In der Tiermedizin wird die Anforderungsdefinition nach der Implementierung durchgeführt, weil Multimedia-Lernanwendungen noch weitgehend unbekannt sind. Am Beispiel einer vorliegenden Anwendung können die Studenten und Hochschullehrer beurteilen, ob das Programm für sie nützlich ist.

Diese Vorgehensweise ähnelt dem **evolutionären Software-Entwicklungsmodell**, in dem aufgrund der Anforderungen eines Auftraggebers der Kern eines zu entwickelnden Produktes definiert wird. Nur dieser Produktkern wird entworfen und implementiert.[378] Der Auftraggeber sammelt Erfahrungen mit dem Einsatz dieses Produktkerns und

[377] Vgl. Regula, 1997, S. 118.
[378] Vgl. Balzert, 1998, S. 120-122.

ermittelt daraus Anforderungen für eine erweiterte Version. Der Produktkern wird anschließend um die neuen Anforderungen ergänzt. Die neue Produktversion wird eingesetzt, und auf Grund der dabei gewonnenen Erfahrungen werden wiederum neue Anforderungen aufgestellt. D.h. das Software-Produkt wird allmählich und stufenweise entwickelt, gesteuert durch die Erfahrungen, die der Auftraggeber und die Benutzer mit dem Produkt machen. Allerdings können im Tiermedizinstudium die Ergebnisse der Evaluation kaum noch zur Verbesserung der Anwendung genutzt werden, weil im Unterschied zum evolutionären Software-Modell nach der Evaluation keine neue Versionen des Programms erstellt werden, sondern die Doktoranden ihre Dissertationen über das Projekt schreiben und anschließend die Universität verlassen.

Wenn die Hochschullehrer und Studenten in der Tiermedizin verschiedene Modell-Anwendungen kennengelernt haben, können in Zukunft wahrscheinlich Anforderungen an neue Anwendungen wie in der Informatik üblich am Anfang der Entwicklung ermittelt werden.

8.5.11 Publikation

Im Hochschulprojekten werden die entwickelten Programme und Evaluationsergebnisse in der Regel in wissenschaftlichen Zeitschriften publiziert, auf Kongressen vorgetragen und in den entsprechenden Tagungsbänden veröffentlicht. Beispiele für solche Publikationen aus dem VetMedia-Projekt sind u.a.:

Heuwieser, W.; Regula, G.; Hallmann, T.; Schimmelpfennig, K. (1997):
Computer based training of dairy reproduction: A comparison between a tutorial and a problem oriented approach, in: Proceedings of the 30th AABP Meeting, September 18-20 1997, Montreal, Canada, 1997

Rother, M.; Heuwieser, W.; Hallmann, T. (1999):
Erfahrungen mit einem Internet-basierten Vorlesungsskript zum Fachgebiet Tiergeburtshilfe, in: Tierärztliche Praxis, Februar 1999, Jahrgang 27 (G), S. 9-15

Steens, R.; Heuwieser, W.; Hallmann, T.; Schein, E.; Parthier, H.; Schaper, R. (1997):
Presentation of an interactive, multimedia training program to teach helmintology in dogs and cats, in: Proceedings of the 16th International Conference of the World Association for the Advancement of Veterinary Parasitology (WAAVP), 10-15 August 1997, Sun City, South Africa, p. 79

8.5.12 Abnahme

Bei Auftragsproduktionen z.B. für Pharmazie-Unternehmen umfaßt die Entwicklung u.a. auch eine Abnahmephase, in der die Übergabe der fertigen Anwendung an den Auftraggeber erfolgt, der das Programm daraufhin testet, ob es seine Anforderungen erfüllt. Bei erfolgreicher Abnahme erklärt der Auftraggeber die Annahme des Programms, das damit in sein Eigentum übergeht.

8.5.13 Vervielfältigung bzw. Online-Stellung

Die Verteilung der entwickelten Anwendungen kann u.a. auf CD-ROM, im Internet oder auf beiden Medien erfolgen. Zur Verteilung von CD-ROM-Anwendungen können

entweder CD-Rohlinge einzeln mit einem CD-Recorder beschrieben werden, oder es wird eine Firma mit der Vervielfältigung beauftragt. Weiterhin wird ein CD-Cover erstellt und vervielfältigt, das u.a. die Installationsanleitung und die Hard- und Software-Voraussetzungen für den Einsatz der Anwendung beschreibt. Bei Internet-Anwendungen entfällt normalerweise die Vervielfältigung und die Cover-Erstellung, weil die Anwendungen über das WWW verteilt werden. Um eine Internet-Anwendung auf einem WWW-Server zugänglich zu machen, wird sie in ein öffentlich zugängliches Verzeichnis kopiert und durch Hypertext-Verknüpfungen z.B. mit der Begrüßungsseite des jeweiligen WWW-Servers verknüpft, so daß sie von dort aus aufgerufen werden kann, ohne daß man die genaue URL (Uniform Resource Locator)-Adresse kennt.

8.5.14 Bekanntmachung

Da CD-ROM- und Internet-Anwendungen an Hochschulen ein vergleichsweise neues Medium sind, müssen normalerweise Studenten und Hochschullehrer auf diese Programme aufmerksam gemacht werden, damit sie auch benutzt werden. Zur Bekanntmachung eignen sich Ankündigungen in Vorlesungen und Aushänge am „schwarzen Brett“. Um Internet-Anwendungen auch außerhalb der eigenen Hochschule bekannt zu machen, können die entsprechenden URLs in fachspezifische internationale Sammlungen von Internet-Adressen eingefügt werden. In der umfangreichsten tiermedizinischen Adressen-Sammlung NetVet steht für das Einfügen von neuen URLs eine Web-Seite mit einem speziellen Eingabeformular bereit.[379]

8.5.15 Vertrieb und Verleih

Die Abgabe und Ausleihe der entwickelten Programme an Studenten umfaßt u.a. Aufgaben wie die Entgegennahme und Abrechnung von Einnahmen, die Versendung von CDs mit der Post an Empfänger in anderen Städten sowie die Kontrolle der rechtzeitigen Rückgabe von ausgeliehenen CDs. Diese Aufgaben können zum Teil von der Fachbereichsbibliothek übernommen werden, die z.B. Mahnungen bei verspäteter Rückgabe maschinell erstellen kann. Bei der Entwicklung von Anwendungen in Zusammenarbeit mit der Wirtschaft übernehmen normalerweise die Marketing- und Vertriebsabteilungen in den Unternehmen die Verteilung.

8.5.16 Einsatz

Der Einsatz von Lernanwendungen umfaßt deren praktische Verwendung zur Unterstützung des Studiums in Computer-Arbeitsräumen und bei den Studenten zuhaus sowie in Zukunft auch in Lehrveranstaltungen, wenn die Hörsäle mit Computern, Netzwerk-Zugängen und Videoprojektoren ausgestattet werden. Eine ausführliche Diskusion der Einsatzmöglichkeiten für Computer-Anwendungen unter den Rahmenbedingungen der geltenden Studienordnung findet sich in Kapitel 12.

8.5.17 Wartung- und Pflege

Nach Balzert umfaßt die **Wartung** eines Software-Produktes die Lokalisierung und Behebung von Fehlerursachen sowie die Optimierung des Leistungsverhaltens nach der

[379] Vgl. „NetVet - Veterinary Resources“, Washington University, URL: http://netvet.wustl.edu/newurl.htm, Stand: 11.05.98.

Inbetriebnahme.[380] Die **Pflege** besteht dagegen aus der Änderung eines Software-Produktes durch Anpassung an geänderte Anforderungen wie die Erweiterung durch funktionale Ergänzungen oder die Anpassung an neue Umgebungen wie die Verwendung im Internet. So ist z.B. eine Anpaßung von CD-ROM-Anwendungen wie „Brunstzyklus beim Rind“ notwendig, damit diese Programme auch im Internet laufen. Wenn Anwendungsprogramme nicht an neue Umgebungen angepaßt werden, besteht die Gefahr, daß sie veralten und schließlich nicht mehr benutzt werden. Die Bereitstellung von Personal für diese Anpassungsarbeiten ist ein in Hochschulprojekten weitgehend ungelöstes Problem, weil es sich dabei normalerweise nicht um Leistungen handelt, die als Diplom bzw. Dissertation anerkannt werden können, und weil diese Arbeiten aufgrund des Umfangs in der Regel nicht von Studenten nebenbei erledigt werden können. Nach Balzert ist der Aufwand für die Wartung und Pflege von Software größer als der Entwicklungsaufwand und umfaßt zwischen 67 und 80 Prozent des Gesamtaufwandes.[381]

8.5.18 Projektmanagement

Nach Balzert umfaßt das **Management** von Software-Projekten alle Aufgaben, um die Aktivitäten von Mitarbeiter zu planen und zu kontrollieren, damit ein Ziel erreicht wird, das durch die Mitarbeiter allein nicht erreicht werden kann.[382] Wie Abbildung 44 zeigt, ist das Projektmanagement eine den gesamten Entwicklungsprozeß begleitende Tätigkeit. Zu den Zielen des Software-Managements zählt Grady u.a.:[383]

- Maximierung der Kunden- bzw. Benutzerzufriedenheit
- Minimierung des Aufwandes und der Zeit für die Software-Erstellung
- Minimierung von Programmfehlern

Nach Haag ist der Projektmanager z.B. für das Einhalten von Terminen und Richtlinien, die Organisation des Projektes, die Motivation der Mitarbeiter und die Kommunikation mit dem Auftraggeber verantwortlich.[384] Thayer nennt u.a. folgende Management-Aktivitäten in Software-Entwicklungsprojekten[385], die auch im Hochschulprojekten vorkommen und dort zum Teil von Professoren ausgeführt werden, die zugleich Projekt-Leiter sind:

Planungsaktivitäten:
- Ziele setzen
- Strategien und Taktiken entwickeln
- Termine festlegen
- Entscheidungen treffen
- Vorgehensweisen auswählen und Regeln festlegen
- zukünftige Situationen vorhersehen
- Budgets vorbreiten

Organisationsaktivitäten:
- Identifizieren und Gruppieren der zu erledigenden Aufgaben

---

380 Vgl. Balzert, 1996, S. 975-976.
381 Vgl. Balzert, 1996, S. 970.
382 Vgl. Balzert, 1998, S. 6.
383 Vgl. Grady, 1992, p. 22.
384 Vgl. Haag, 1995, S. 94.
385 Vgl. Thayer, 1990, S. 17.

- Auswahl und Etablierung organisatorischer Strukturen
- Festlegung von Verantwortungsbereichen und disziplinarischen Vollmachten
- Festlegen von Qualifikationsprofilen für Positionen

Personalaktivitäten:
- Positionen besetzen
- Neues Personal einstellen und integrieren
- Aus- und Weiterbildung von Mitarbeitern
- Personalentwicklung planen
- Mitarbeiter beurteilen
- Mitarbeiter bezahlen
- Mitarbeiter versetzen oder entlassen

Leitungsaktivitäten:
- Mitarbeiter führen und beaufsichtigen
- Kompetenz delegieren
- Mitarbeiter motivieren
- Aktivitäten koordinieren
- Kommunikation unterstützen
- Konflikte lösen
- Innovationen einführen

Kontrollaktivitäten:
- Prozeß- und Produktstandards entwickeln und festlegen
- Berichts- und Kontrollwesen etablieren
- Prozesse und Produkte vermessen
- Korrekturaktivitäten initiieren
- Loben und Tadeln

8.5.19 Qualitätssicherung

Qualitätssicherung gehört neben der Software-Entwicklung und dem Software-Management zu den Hauptaufgaben der Software-Technik.[386] Die Qualitätssicherung beginnt nicht erst nach der Fertigstellung einer Anwendung, sondern ist eine den gesamten Entwicklungsprozeß begleitende Tätigkeit, damit Fehler zum frühestmöglichen Zeitpunkt erkannt und behoben werden können. Auf diese Weise werden Folgefehler vermieden und der Aufwand für die Fehlerkorrektur minimiert. Im VetMedia-Projekt erfolgt die Qualitätssicherung bisher überwiegend in Form einer Begutachtung der erstellten Anwendungen durch den Projektleiter und gegebenenfalls zusätzlicher Experten für das jeweilige Fachgebiet. Weiterhin werden Programmtests mit Studenten und Tierärzten durchgeführt.

Nach dem „Lehrbuch der Software-Technik“ von Balzert umfaßt die Qualitätsicherung alle geplanten und systematischen Tätigkeiten, um den Nachweis zu erbringen, daß die Qualitätsanforderungen an ein Software-Produkt erfüllt sind.[387] In der DIN ISO-Norm 9126 wird Software-Qualität als die Gesamtheit der Merkmale und Merkmalswerte eines Software-Produktes definiert, die sich auf dessen Eignung beziehen, festgelegte

[386] Vgl. Balzert, 1998, S. 253ff.
[387] Vgl. Balzert, 1998, S. 299.

oder vorausgesetzte Erfordernisse zu erfüllen. Die Norm nennt die folgenden sechs Qualitätsmerkmale für Software-Produkte:[388]

- **Funktionalität**, d.h. das Vorhandensein von zuvor festgelegten Eigenschaften, die die definierten Anforderungen erfüllen
- **Zuverlässigkeit**, d.h. die Fähigkeit der Software ihr Leistungsniveau unter festgelegten Bedingungen über einen bestimmten Zeitraum zu bewahren
- **Benutzbarkeit**, d.h. der Aufwand, der zur Benutzung erforderlich ist, und die individuelle Beurteilung der Benutzung durch eine festgelegte Benutzergruppe
- **Effizienz**, d.h. das Verhältnis zwischen dem Leistungsniveau der Software und dem Umfang der eingesetzten Betriebsmittel wie z.B. Arbeitsspeicher
- **Änderbarkeit**, d.h. der Aufwand der zur Durchführung von Korrekturen, Verbesserungen und Anpassungen an die Umgebung und an neue Anforderungen notwendig ist
- **Übertragbarkeit**, d.h. die Eignung der Software von einer Hard- oder Software-Umgebung in eine andere Umgebung übertragen zu werden

Damit diese Qualitätsmerkmale auf jede Art von Software angewendet werden können, sind sie allgemein gehalten. Um die Qualität und Akzeptanz von Multi-/Hypermedia- und Internet-Anwendungen für die Hochschulausbildung sicherzustellen, wird ein spezielleres Beurteilungssystem benötigt. Ein Beispiel dafür sind die „Qualitätskriterien für Elektronische Publikationen in der Medizin“, die am Institut für Medizin-Informatik an der Universität Freiburg von Schulz, Auhuber, Schrader und Klar für die Humanmedizin erarbeitet worden sind. Dieser Kriterienkatalog läßt sich auch in anderen Fächern als Checkliste für die Begutachtung von Lernanwendungen verwenden, weil er von den humanmedizinischen Inhalten weitgehend unabhängig ist.[389] Die aufgelisteten Kriterien ermöglichen eine Beurteilung der Qualität von Lernanwendungen u.a. in den Bereichen Inhalt, Technik und Design. Fünf Beispiele für Qualitätskriterien sind:

- Für die Bedienung sind keine DV-technischen Spezialkenntnisse erforderlich.
- In Internet-Anwendungen sind große Bilddateien vor dem Herunterladen als „Thumbnails“ einsehbar.
- Bei tutoriellen Systemen erscheinen die Bildschirmfenster niemals „vollgeschrieben“. Die Regel „Ein Thema - ein Textfenster“, ist weitgehend realisiert und Scrollen von Text wird weitgehend vermieden.
- Filmähnliche Vor- und Abspanne sind abschaltbar und überspringbar.
- Die zu einer definierten Internet-basierten Publikation gehörigen Seiten heben sich in ihrem Layout klar sichtbar von anderen, nicht zugehörigen Seiten ab, um die Gefahr unbemerkten Verlassens der Publikation zu verringern.

8.5.20 Rollen bei der Anwendungsentwicklung

Die Entwicklung von Anwendungsprogrammen kann normalerweise nicht durch einen einzelnen Mitarbeiter durchgeführt werden, da er zuviel Zeit für die Fertigstellung benötigen würde und in der Regel auch nicht alle erforderlichen Qualifikationen besitzt. Daher muß eine Gruppe von Mitarbeitern mit unterschiedlichen Qualifikationen gemeinsam als Team an dieser Aufgabe arbeiten. Die Erstellung von Multi-

[388] Vgl. DIN ISO 9126, 30.9.91, S. 3ff.
[389] Vgl. Abteilung Medizinische Informatik, Universität Freiburg, URL: http://www.imbi.uni-freiburg.de/medinf/cbt_qk.htm, Stand: 07.02.99.

/Hypermedia-Anwendungen erfordert nach dem sogenannten „7M-Modell“ von Sander, Kolthof und Scheer u.a. Wissen aus den folgenden Bereichen:[390]

- **Medien-Authoring**: Festlegung der Inhalte, die in eine Multimedia-Anwendung aufgenommen werden, sowie die Beschaffung dieser Informationen und ihre Formulierung in einer geeigneten Form
- **Medien-Didaktik**: Gestaltung der Anwendung nach pädagogischen Erkenntnissen z.B. als Tutorium, Informationssystem oder Simulation und Bestimmung einer Kombination des Anwendungseinsatzes mit herkömmlichen Lehrfomen wie Vorlesungen, Übungen und Seminaren
- **Medien-Psychologie**: Auswahl von Medien wie Texten, Bildern und Videos im Hinblick auf die Vermittlung von bestimmten Informationen
- **Medien-Design**: Erstellung einer intuitiv verständlichen Benutzeroberfläche und einer logisch aufgebauten Bedienerführung
- **Medien-Technik**: Produktion u.a. von Bildern, Audio- und Videosequenzen
- **Medien-Engineering**: Erstellung und Implementierung eines Programmierkonzeptes zur Integration der erstellten Medien
- **Medien-Management**: Planung, Steuerung und Kontrolle der Entwicklungsaufgaben

Nach Erfahrungen aus dem VetMedia-Projekt lassen sich die für die Entwicklung notwendigen Erfahrungen, Kenntnisse und Fähigkeiten vor allem durch die in Tabelle 1 angegebenen Rollen beschreiben. Eine Person übernimmt in der Regel mehrere Rollen, weil normalerweise nicht genügend Mittel zur Verfügung stehen, um für jede Rolle einen speziellen Mitarbeiter zu beschäftigen.

Person:	**Rolle:**
Professor	• Projektmanager • Inhaltsexperte • Vertriebsexperte
Doktorand	• Inhaltsexperte • Audioproduzent • Videoproduzent • Vertriebsexperte
Mediendesigner	• Graphik-Designer
Informationswissenschaftler	• Programmierer • System- und Netzwerkverwaltung • Audioproduzent • Videoproduzent
Student	• Tester

Tabelle 1: Personen und ihre Rollen bei der Anwendungsentwicklung

8.5.21 Zusammenarbeit im Team

Die Autoren DeMarco und Lister berichten, daß etwa 15 bis 25 Prozent der von ihnen untersuchten Software-Entwicklungen scheitern, wobei die Ursache in der Mehrheit der untersuchten Projekte nicht in technischen, sondern in personenbezogenen Problemen liegt:

[390] Vgl. Sander, Kolthof, Scheer, 1996.

„*Die größten Probleme unserer Arbeit sind keine technologischen Probleme, sondern soziologische Probleme*.“[391]

Nach DeMarco und Lister lassen sich erfolgreiche Software-Entwicklungen vor allem auf gute menschliche Zusammenarbeit zurückführen. Damit Fachleute in einem Team erfolgreich zusammenarbeiten, müssen sie gut geführt werden. Dazu benötigt man teamorientierte Manager und teamfähige Mitarbeiter, die nach Balzert unter Bezug auf DeMarco, Lister und Berkel u.a. folgende Aufgaben und Eigenschaften besitzen:[392]

teamfähiger Manager:
- Team auf gemeinsames Ziel ausrichten
- Strategische Richtlinien vorgeben, aber keine taktischen Richtlinien
- Kompetenz bei Mitarbeitern anerkennen
- Freiheit und Verantwortung für bestimmte Aufgaben an Mitarbeiter übertragen
- Vertrauensvorschuß gewähren
- Teams sich selbst bilden lassen oder Mitspracherecht bei der Zusammensetzung einräumen

teamfähiger Mitarbeiter:
- Positive Einstellung zur Teamarbeit
- Kritik- und Konfliktttoleranz
- Gegenseitige Anerkennung und Respektierung der fachlichen Qualifikation und persönlichen Integrität
- Partnerschaftliches Verhalten
- Fähigkeit, widersprüchliche und voneinander abweichende Informationen zu verarbeiten
- Bereitschaft, sich voll im Team zu engagieren
- Bereitschaft, mit sich selbst zufrieden zu sein

Eine mögliche Quelle für Konflikte ist der Interessengegensatz zwischen Mediendesign und Programmierung. So steht für den Designer in der Regel die ansprechende Gestaltung der Benutzeroberfläche im Vordergrund. Dagegen legt der Programmierer normalerweise vor allem Wert auf die Ablaufgeschwindigkeit und die Änderbarkeit von Anwendungen. Diese Ziele laufen einander teilweise entgegen, wenn z.B. hochauflösende Graphiken die Ladezeiten einer Internet-Anwendung erhöhen. Die Lösung solcher Interessesenkonflikte liegt u.a. darin, einen Kompromiß zu treffen, der sich an den jeweiligen Anwendungsanforderungen wie Bedienbarkeit, Effizienz und Änderbarkeit orientiert.

[391] DeMarco, Lister, 1991, S. 5.
[392] Vgl. DeMarco, Lister, 1991, S. 141ff und Berkel, 1984, S. 29f.