Informatisches Modellieren

Softwareentwicklung im Informatikunterricht

Dr. Henry Herper

Otto-von-Guericke-Universität Magdeburg

Institut für Simulation und Graphik

Lisa-Weiterbildung - Aug. 2002
(angepasst Aug. 2004)

Informatische Modellierung

„Im Informatikunterricht bedeutet „Modellierung“ im wesentlichen die Abgrenzung eines für den jeweiligen Zweck relevanten Ausschnittes der Erfahrungswelt, die Herausarbeitung seiner wichtigen Merkmale unter Vernachlässigung der unwichtigen sowie seiner Beschreibung und Strukturierung mithilfe spezieller Techniken aus der Informatik.“

/*Gesellschaft für Informatik (GI) e.V.:* Empfehlungen für ein Gesamtkonzept zur Informatischen Bildung an allgemein bildenden Schulen, Bonn, 2000/

Modell und Modellieren im Informatikunterricht

In der Informatik ist ein Modell ein von Subjekten durch Abstraktion geschaffenes Abbild eines Originals, meistens aus der realen Welt. Ziel des Modellierens ist es, Probleme aus der "realen Welt" mit Informatik-Werkzeugen zu bearbeiten sowie konkrete Informatiksysteme zu durchdringen oder zu erzeugen.

Der Einsatz dieser Informatiksysteme wirkt auf die reale Welt zurück.

(Arbeitsgruppe Modellbildung und Simulation - Königsteiner Gespräche 2001)

Eigenschaften von Software

- ✓ Software ist nicht sinnlich wahrnehmbar
- ✓ Software besteht aus Sprache
- ✓ Software ist digital
- ✓ ***Software ist fehlerhaft***
- ✓ große Softwaresysteme sind komplex
- ✓ große Softwaresysteme bestehen aus umfangreichen Texten
- ✓ große Softwaresysteme verfestigen Sichtweisen

Programmieren

„Programmieren bedeutet das Lösen von Problemen der realen Welt mit Hilfe des Computers. Dazu gehören alle Tätigkeiten, die vom umgangssprachlich formulierten Problem zum präzisen, formalen Programm führen.“

/Pagel, Six; Software Engineering; 1994/

geringe Programmkomplexität ⇨ „Programmieren im Kleinen“

Programmieren

Programmierung ist die Algorithmierung eines Problems und die Umsetzung der Lösung dieses Problems in ein Programm, d.h. in eine Notation, die von der konkreten Anlage, auf der das Problem behandelt werden soll, "verstanden" wird. Die Anlage kann das Programm als eine Anweisungsfolge interpretieren, die die Problemdaten verarbeitet.

© Bibliographisches Institut & F.A. Brockhaus AG Mannheim und paetec Gesellschaft für Bildung und Technik mbH Berlin. Alle Rechte vorbehalten. www.schuelerlexikon.de

Struktur der Rahmenrichtlinie (Entwurf 01/03)

SJG 10	SJG 11	SJG 12
Grundlagen der Informationstechnik Projektarbeit unter Nutzung von Standardsoftware	Algorithmen-strukturen und ihre Implementierung	Wahlthema
Informatik und Gesellschaft Computer-Netzwerke	Datenstrukturen Informatisches Modellieren	Projektarbeit zur Software-entwicklung

Informatisches Modellieren in der Rahmenrichtlinie (Entwurf 01/03)

SJG 10	SJG 11	SJG 12
Grundlagen der Informationstechnik **Projektarbeit unter Nutzung von Standardsoftware**	Algorithmen-strukturen und ihre Implementierung	Wahlthema
Informatik und Gesellschaft Computer-Netzwerke	Datenstrukturen **Informatisches Modellieren**	**Projektarbeit zur Software-entwicklung**

Informatisches Modellieren in den RRL

SJG 11/2 - Zeitumfang: 16 Stunden

Ziele:

- Festigung der Fähigkeiten und Fertigkeiten im Bereich der Algorithmierung
- Festigung der Programmierfertigkeiten
- Erlernen der Grundlagen der Softwaretechnik
- Analyse von komplexen Informatiksystemen

Informatisches Modellieren in den RRL

Vorbemerkung/Qualifikation

Die im Umgang mit Algorithmierung und Programmiersprache erlernten Fähigkeiten und Fertigkeiten werden an einem Beispiel gefestigt und dabei wird ein vollständiger Softwareentwicklungszyklus (Software life cycle) durchlaufen.

Die Komplexität des Beispiels ist so zu wählen, dass eine Realisierung in der zur Verfügung stehenden Zeit möglich ist.

Fragen bei der Softwareentwicklung

- Wie konzipiert, analysiert und integriert man Gesamtsysteme, um die gewünschten Eigenschaften zu erhalten?
- Welche Hilfsmittel gibt es und wie werden sie verwendet?
- Wie integriert man Hardware- und Softwarearchitekturen?
- Wie kommuniziert der Mensch über geeignete Graphikoberflächen und Schnittstellen mit dem Gesamtgebilde?
- Welches sind Funktionen, die der Mensch optimal ausführen kann und welche Aufgaben kann das System autonom erledigen?

/Paul Levi, Ulrich Rembold; Einführung in die Informatik; Carl Hanser Verlag München Wien 2003/

Software Engineering

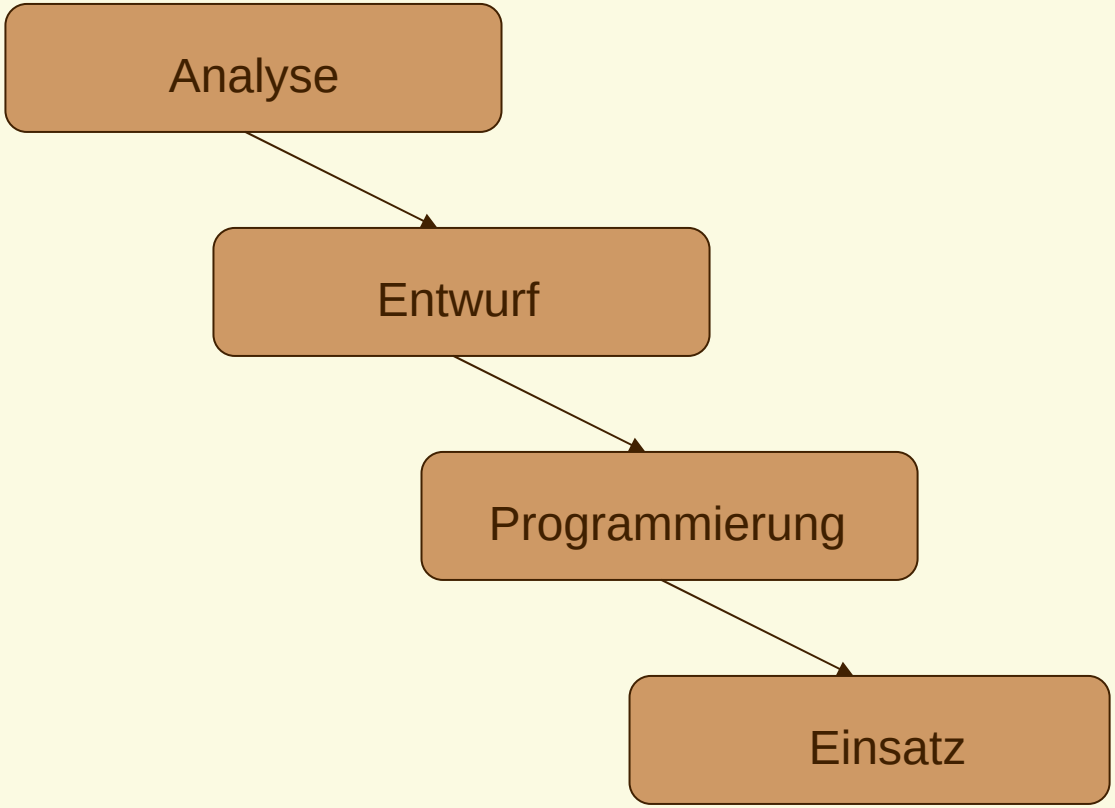
Aufgaben des Softwareentwicklers:

Umsetzung einer Idee in ein reales Produkt mit Hilfe von wissenschaftlichen Erkenntnissen und unter Verwendung von Werkzeugen der Informationsverarbeitung.

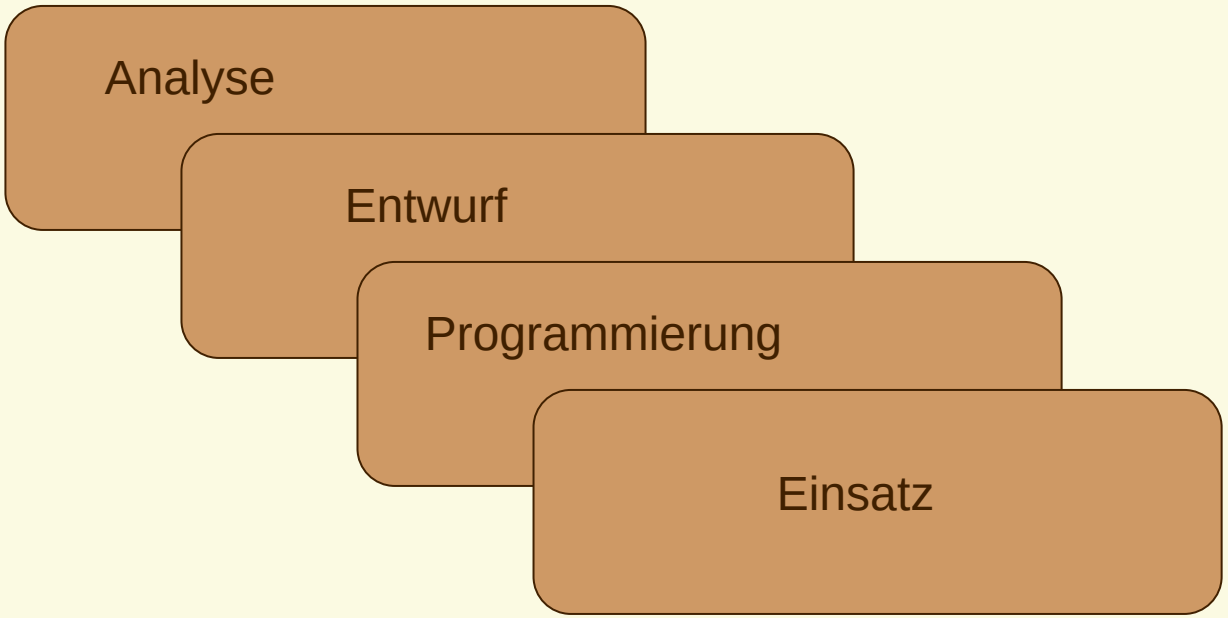
➔Der Softwareentwickler bzw. das Entwicklungsteam muss über Kompetenzen auf dem Gebiet der Konstruktionslehre, Produktionstechnik, Qualitätssicherung und Projektorganisation für Software verfügen.

Softwareentwicklung ist (bisher) nicht automatisierbar ➔ Softwareentwicklung erfordert Kreativität

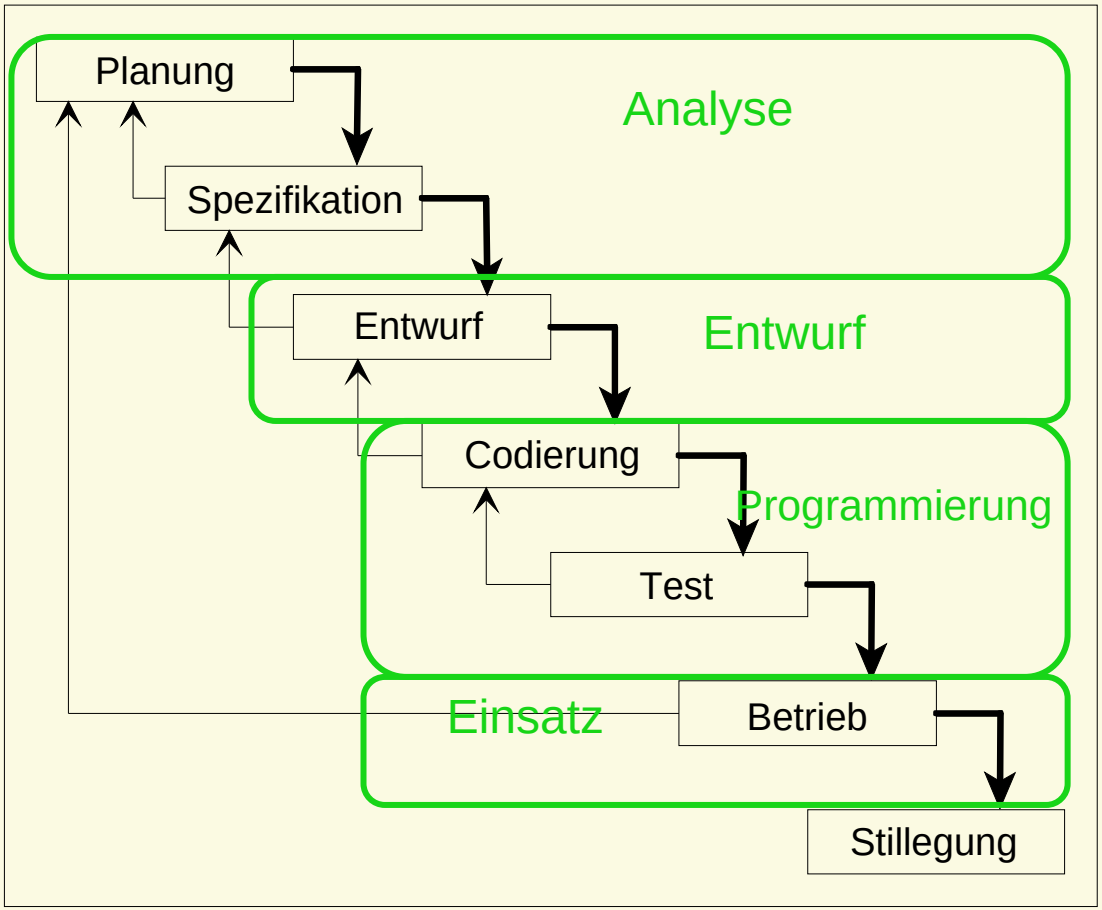
Softwarelebenszyklus - klassisches Wasserfallmodell



Softwarelebenszyklus – heutiges Wasserfallmodell



Softwarelebenszyklus – Wasserfallmodell



Softwarelebenszyklus - Schulnotation

Programmierung
(Arbeitsschritte der Programmentwicklung)

Problemanalyse
Untersuchung der Algorithmierbarkeit des Problems; Erfassung der Problemdaten und der Werkzeuge, die das informationsverarbeitende System für die Behandlung anbietet

↓

Algorithmierung
Erarbeitung der Programmstruktur im Rahmen der durch die Programmumgebung vorgegebenen Bedingungen

↓

Codierung
Schaffung einer Programmniederschrift, die auf dem Computer abgearbeitet werden kann
(Programmierung im engeren Sinne)

↓

Programmtest
Prüfung der korrekten Arbeitsweise des Programms an Hand ausgewählter, wohlüberlegter Daten (Trockentest zum Prüfen der Steuerstrukturen; Konzipierung einer Testumgebung für Programmteile)

↓

Dokumentation
Beschreibung der Leistungsfähigkeit des Programms; Vermittlung der Handhabung und Wartung einem potentiellen Anwender

© Bibliographisches Institut & F.A. Brockhaus AG Mannheim und paetec Gesellschaft für Bildung und Technik mbH Berlin. Alle Rechte vorbehalten. www.schuelerlexikon.de

1. Finden einer geeigneten Problem-/ Aufgabenstellung

Anforderungen an eine Aufgabenstellung:

- Systembeschreibung in einer für die Schüler verständlichen Form
- innerhalb einer begrenzten Bearbeitungszeit lösbar
- mit den bisher vermittelten Anweisungen der Programmiersprache implementierbar
- Ergebnisse aus der Erfahrungshorizont der Schüler verifizierbar und validierbar
- Lösung der Aufgabe schafft „Mehrwert“

1. Finden einer geeigneten Problem-/ Aufgabenstellung

Zu festigende Fertigkeiten in der Programmierung

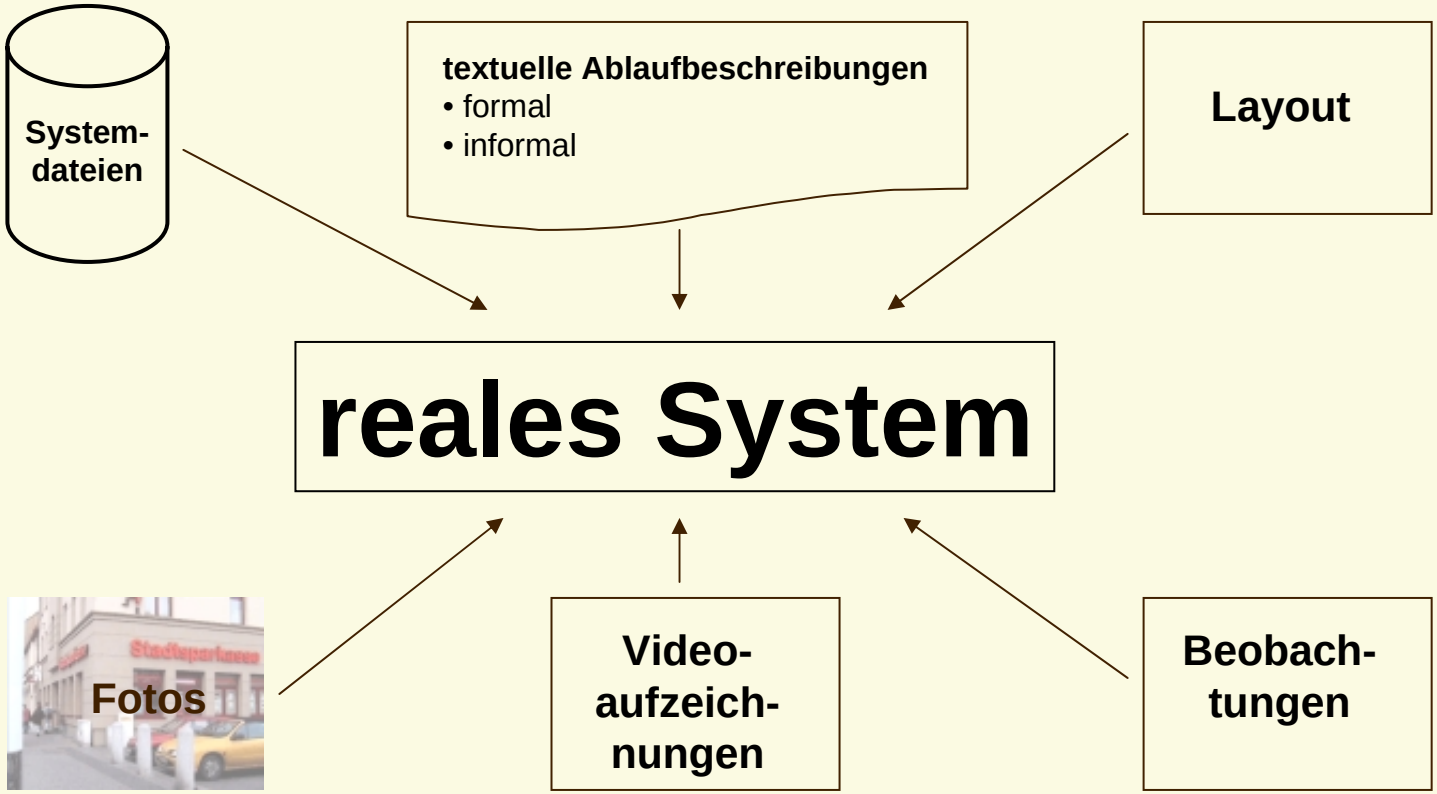
- Struktur, Aufbau und Modularisierung von Programmen
- Datentypen und ihre zweckmäßige Verwendung
- Unterprogrammkonzepte und Parameterübergabe
- Kontrollstrukturen
- Datensicherung in Dateien
- Gestalten von Benutzungsoberflächen

1. Finden einer geeigneten Problem-/ Aufgabenstellung

Aspekte von Datenschutz und Datensicherheit

- Urheber-/Eigentumsrecht an Daten
- Datenschutzbestimmungen bei der Datenerfassung
- Zugriffsrechte auf Daten
- Sicherung der Projektdaten

Multimediale Systembeschreibung



Beispiel für eine textuelle Aufgabenstellung

Bei den folgenden Aufgaben kommt es darauf an, das Vorgehen vom Problem zu seiner Lösung zu konzipieren. Es sind die Arbeitsschritte bei der Programmentwicklung inhaltlich zu beschreiben, und zwar einerseits so genau und vollständig wie möglich, andererseits anschaulich durch den Entwurf von Dialogmenüs, Erarbeitung von Testdaten u.s.w.

.....

b) Die für den Chemieunterricht zur Verfügung stehenden Chemikalien sollen in einer Datensammlung erfasst werden.

Hinweise: Ist eine Unterteilung der Sammlung sinnvoll? Entwerfen Sie die Struktur der Datensätze (Attribute der Teile des Datensatzes)! Sind Summenformeln aufzunehmen und wie müssen diese codiert werden? Welche Programmierwerkzeuge sollen genutzt werden? Erarbeiten Sie Konzepte für die Bereitstellung der Eingabedaten sowie für die Verwaltung der Datei und ihren Schutz vor unberechtigtem Zugriff und Testrahmen und Testdaten für eine Minimalversion!

/Informatik bis zum Abitur, Paetec Verlag, S. 238/

Systemverständnis als Grundlage der informatischen Modellbildung

Systemverständnis ist z.B. die Grundlage für:

- die Wahl eines geeigneten Abstraktionsgrades,
- die Abgrenzung der Aufgabenstellung (Umweltschnittstellen),
- die Erstellung der formalen Algorithmen,
- die Verifikation und Validierung des Modells und
- die Entwicklung einer geeigneten Dokumentation.

Aufgabenklassen im Informatikunterricht

Datenbanken/
Datenverarbeitung/
Datenverwaltung

Messen/
Steuern/
Regeln

Komponenten von
Informatiksystemen

Modellbildung
und Simulation

Anwendungsgebiete von Aufgaben

- Aufgaben im Unterrichtseinsatz
  - ➔ kleine abgeschlossene Beispiele zur Demonstration bestimmter Funktionalitäten bzw. Sprachelemente
- Hausaufgaben
  - ➔ Festigung der Fähigkeiten und Fertigkeiten durch kleine Anwendungsaufgaben
- Klausur- und Prüfungsaufgaben
  - ➔ Aufgaben oder Teile von Aufgaben, die in einer relativ kurzen Bearbeitungszeit durch die Schülerinnen und Schüler selbständig zu bearbeiten sind
- Projekt- und Belegaufgaben
  - ➔ Aufgaben mit höherer Komplexität, die eine Bearbeitung über mehrere Unterrichtseinheiten erfordern

2. Problemformulierung, Problemanalyse und Entwurf

Die Schülerinnen und Schüler

- sind in der Lage, aus einer Aufgabenstellung heraus die Ziele und Anforderungen für ein Informatiksystem abzuleiten,
- erkennen das Informatiksystem als Abbild (Modell) eines realen Systems,
- können die erforderlichen Daten bestimmen und in der geforderten Genauigkeit erfassen,

Informatisches Modellieren – Problemformulierung und Problemanalyse

Inhalte	Hinweise zum Unterricht
– **Problemformulierung und Problemanalyse** • Ergebnis: Anforderungsdefinition	– vollständiges und eindeutiges Erfassen des realen Systems bzw. Problems und aller wichtigen Umgebungsbedingungen (Systemeigenschaften, Pflichtenheft)

Analysephase - Formen der Ist-Analyse

Formen:

- **Benutzeranalyse** (Was erwartet der Benutzer vom System? Welche Fähigkeiten hat der Benutzer?)
- **Prozessanalyse** (Welche Aktivitäten hat das System auszuführen und welche Eigenschaften haben diese?)
- **Datenanalyse** (Welche Daten stehen über das System zur Verfügung? Wie ist Art, Struktur und Volumen der Daten?)
- **Kommunikationsanalyse** (Wie bewegen sich die Informationen im System?)
- **Organisationsanalyse** (Wie sind die zu lösenden Aufgaben in das Gesamtsystem einbezogen?)

Analysephase - Methoden der Ist-Analyse

Methoden:

- **Beobachtung** (Sammeln von Erkenntnissen über das System, Auswertung von verfügbaren Dokumenten)
- **Interviews** (Gespräche mit Systemexperten/ zukünftigen Anwendern)
- **schriftliche Befragung** (Beantwortung vorgefertigter, möglichst eindeutiger Fragebögen)
- **Untersuchung** (gezielte Ermittlung detaillierter Angaben)

Aspekte des Datenschutzes sind zu beachten!

nach /HORN97/

Analysephase - Analyse des Soll-Zustandes (Systemspezifikation)

Kategorien von Anforderungen:

- **funktionelle Anforderungen** (Welche Leistungen sind bereitzustellen? Wie ist das System durch den Nutzer zu bedienen?)
- **qualitative Anforderungen** (Welche Qualitätsfaktoren sind bei der Entwicklung des Systems und der Dokumentation zu beachten?)
- **operationelle** Anforderungen (Welche quantitative Leistung ist bei der praktischen Nutzung zu erbringen?)

Zur Festlegung dieser Anforderungen ist eine Zerlegung des Gesamtsystems sowie die Festlegung der Systemgrenzen notwendig. Unterschiedliche Zerlegungsstrategien sind möglich.

Analysephase - Analyse des Soll-Zustandes

➔ **Lastenheft (Aufgabenstellung)**

mit:

- Beschreibung des zukünftigen Systems,
- Beschreibung der Zielsetzung,
- Beschreibung der Umweltschnittstellen sowie
- Randbedingungen.

Entwurfsphase - Projektvorbereitung

- **Umgebungsmodell**

Die Beziehungen des Softwaresystems zu seiner Umgebung werden dargestellt.

- **Ereignistabelle**

Es werden alle Ereignisse erfasst, auf die das System reagieren soll. Weiterhin wird festgelegt, welche Daten das Ereignis auslösen und wie das System auf dieses Ereignis zu reagieren hat.

- **Datenflussdiagramme**

Im Datenflussdiagramm werden die Aktivitäten zur Verarbeitung der Daten als Prozesse dargestellt. Externe Kommunikationspartner werden als Quellen und Senken dargestellt. Das **Kontextdiagramm** ist ein spezielles Datenflussdiagramm zur Darstellung der Beziehungen des Systems mit seiner Umwelt.

(Kontrollflüsse zur Steuerung des Ablaufes werden nicht dargestellt)

Entwurfsphase – Projektvorbereitung - Ereignistabelle

Ereignistabelle Schulbibliothek (Beispiele)

Ereignis	Datenfluss zum System	Antwort des Systems
Erfassen eines Buches	Buch-Eintragsmeldung	Bestätigung der Eintragung, Aktualisieren des Bestandes
Suchen eines Buches	Suchanfrage mit Abfragedaten	Liste der gefundenen Bücher, Fehlmeldung
Drucken von Mahnungen	Anforderung der aktuellen Mahnungen	Ausgabe der Liste und der Mahnscheine

Entwurfsphase – Projektvorbereitung - Datenflussdiagramm

- für die Behandlung jedes Ereignisses aus der Ereignistabelle wird im Datenflussdiagramm genau ein Prozess vorgesehen, der für die Reaktion des Systems auf dieses Ereignis vollständig zuständig ist
- Datenflüsse werden in ankommende und zu erzeugende Datenflüsse unterteilt
- die Massenspeicher des Systems werden ergänzt und mit den darauf zugreifenden Prozessen und Datenflüssen verbunden

Entwurfsphase - Projektvorbereitung

Datenkatalog
Beschreibt den Inhalt und die Struktur aller Datenflüsse und Speicher des Datenflussdiagramms in einer Tabelle.

Endgültiges Verhaltensmodell
Dieses Verhaltensmodell ist ein vollständiges, exaktes und konsistentes Modell zur Beschreibung der tatsächlichen funktionalen Anforderungen an das System.

Entwurfsphase – Projektvorbereitung - Datenkatalog

Beispiel: Datensatz für ein Buch

Autor1, Autor2	je 25 Zeichen	n * String[25]
Herausgeber	25 Zeichen	String[25]
Titel	60 Zeichen	String[60]
Verlag	30 Zeichen	String[30]
Erscheinungsjahr	ganzzahlig	integer
Auflage	ganzzahlig	integer
Band	ganzzahlig	integer
Sprache	20 Zeichen	String[20]
Heft	ganzzahlig	integer
Deskriptoren	Feld mit 5 x 20 Zeichen	array[1..5] of String[20]
Preis	Dezimalzahl	real
Signatur	20 Zeichen	String[20]
ISBN – Nummer	20 Zeichen	String[20]

Entwurfsphase – Projektvorbereitung

➔ Pflichtenheft

juristisch verbindliche Formulierung des Vertrages zwischen dem Auftraggeber und dem Softwareentwicklungsunternehmen zur Herstellung und Abnahme des Softwaresystems

- mit detaillierter Planung des Projektumfangs (Zeitdauer, Kosten und Personal- und Materialbedarf)
- Haftung
- Infrastruktur

3. Entwurfs- und Implementierungsphase

Die Schülerinnen und Schüler

- können unter Anleitung ein komplexes Problem in Teilprobleme zerlegen und die Funktionen dieser Bausteine und ihre Schnittstellen implementieren sowie die Teillösungen zu einer Gesamtlösung zusammenfassen.

Informatisches Modellieren - Entwurf

Inhalte	Hinweise zum Unterricht
– **Entwurf** • Ergebnis: Modellbildung • Ergebnis: Spezifikation	– Beschreibung des zu erzeugenden Systems mit informalen und formalen Mitteln und Erzeugung eines abstrakten Modells – Zerlegung des komplexen Gesamtsystems in überschaubare Teilprobleme (Module) und ihre Bearbeitung in Gruppen – Zusammenführen dieser Teilprobleme durch Beschreibung der Modulfunktionen und einer eindeutigen Schnittstellenbeschreibung sowie Aufzeigen vorhandener Modulabhängigkeiten

UML - Unified Modeling Language

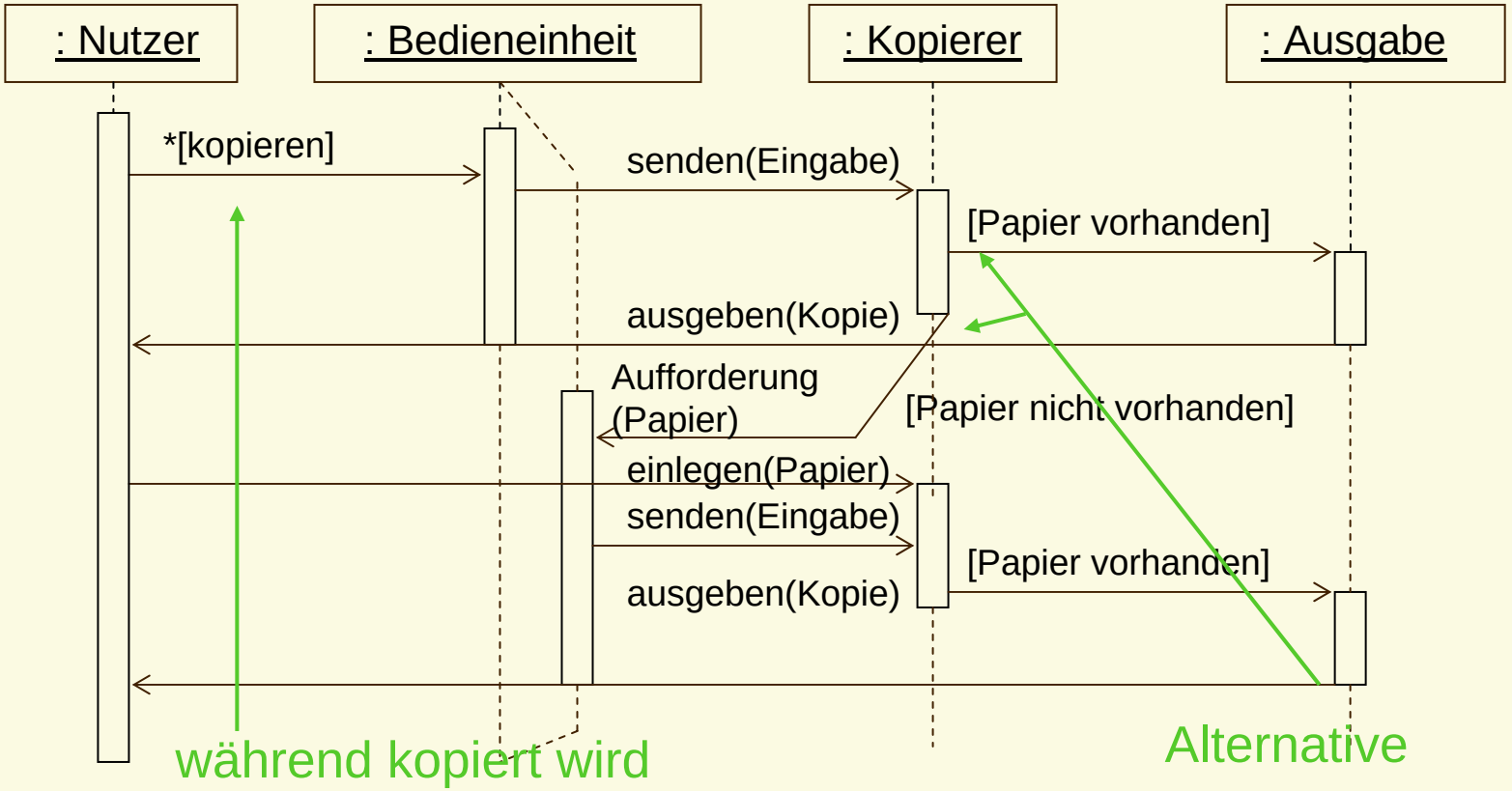
UML – Unified Modeling Language

- mit UML wurde eine einheitliche Notation für unterschiedliche Einsatzgebiete geschaffen (z.B. Datenbankanwendungen, Echtzeitsysteme, Graphikprogramme, Workflow-Anwendungen)
- Modellierungssprache , die basierend auf Diagrammen eine definierte Systembeschreibung realisiert
- **Ziel: visuelle Modellierung zur Unterstützung der Entwicklung von Softwaresystemen**

Ein UML-Modell sagt aus, ***was*** ein Modell tun soll, aber ***nicht, wie*.**

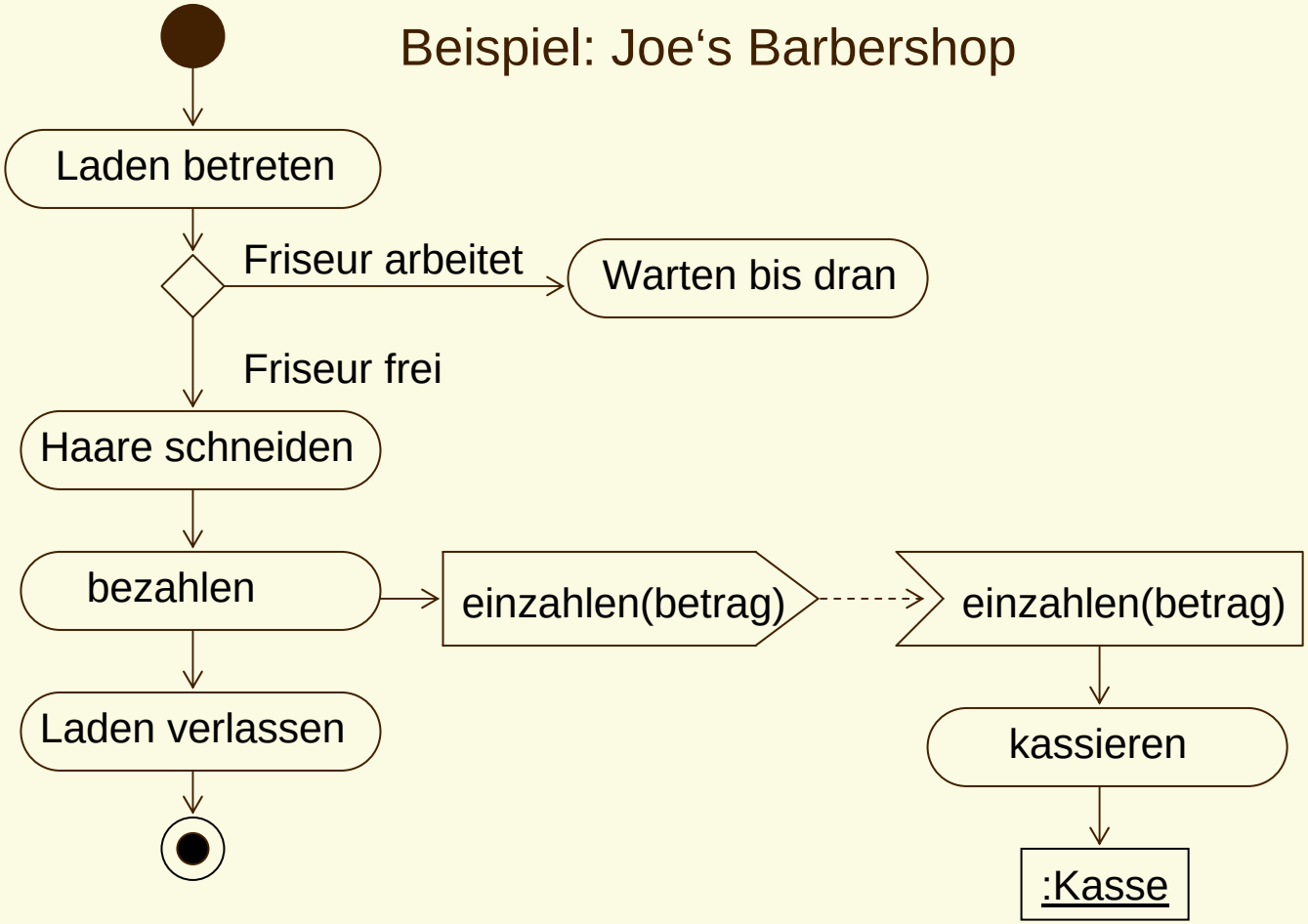
UML – Sequenzdiagrammme - Kontrollstrukturen

In Sequenzdiagrammen können Verzweigungen und Schleifen verwendet werden.



UML – Aktivitätsdiagramme

Beispiel: Joe‘s Barbershop



Entwurfsphase - Prototyping

In der Softwareentwicklung wird unter **Prototyping** die Entwicklung funktionsfähiger Software verstanden, deren Zweckbestimmung es ist, bestimmte Eigenschaften oder Komponenten eines Softwaresystems auf ihre Realisierbarkeit zu überprüfen. Andere Eigenschaften können abstrahiert oder weggelassen werden.

Prototyping schafft die **Kommunikationsbasis** für alle beteiligten Gruppen, vermittelt **experimentelle und praktische Erfahrungen** für die Auswahl von Designalternativen und ist eine **dynamische Beschreibung** des sich entwickelnden Softwaresystems.

Entwurfsphase - Prototyping

Vorteile:

- frühzeitiger Kontakt zwischen Entwicklern und Anwendern, auf einer Basis, die der späteren Nutzungssituation nahe kommt
- Anwender können frühzeitig die Nutzung und Handhabung des Systems erlernen
- Unvollständige und inkonsistente Spezifikationen können sehr schnell erkannt werden
- das System kann frühzeitig der Managementebene vorgestellt werden
- es steht schnell ein experimentelles Modell zur Verfügung, mit dem die Softwarequalität und Zuverlässigkeit verbessert werden kann

Nachteile:

- Prototyp kann sich in seinem Verhalten signifikant vom späteren System unterscheiden
- Auftraggeber verwechselt Prototyp mit endgültigem System

Entwurfsphase - Implementationsmodell

Ziel: Planung der prozessorinternen Struktur (Algorithmen)

Methode: Modularisierung

Als **Modul** wird allgemein ein Programmbaustein bezeichnet, das wenigstens die folgenden Eigenschaften hat:

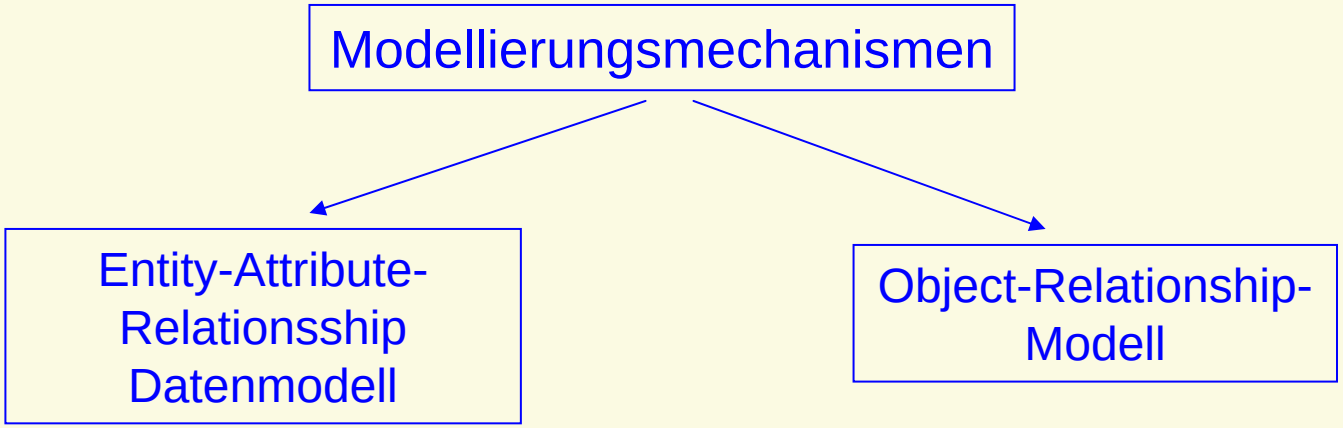
- Es besitzt eine eindeutig definierte Schnittstelle (Interface). Die Kommunikation des Moduls mit seiner Umgebung erfolgt ausschließlich über diese Schnittstelle.
- Ein Modul realisiert eine oder mehrere in sich geschlossene Aufgaben, die über seine Schnittstelle als Leistungen des Moduls (Funktionen, Datentypen, Konstanten) von anderer Programmteilen genutzt werden können.
- Auf welche Weise (Algorithmus, Datenstruktur) die Leistungen durch das Modul erbracht werden, bleibt das vollständige Geheimnis des Moduls (Geheimnisprinzip).
- Es ist bei Beibehaltung der Schnittstelle rückwirkungsfrei austauschbar.
- Es ist prinzipiell unabhängig von anderen Programmteilen erarbeitbar und auf Korrektheit prüfbar.

/HORN97/

Entwurfsphase - Implementationsmodell

Datenmodellierung:

Die Datenmodellierung ermöglicht eine qualitative und quantitative Beschreibung der Informationen.



Entwurfsphase - Implementationsmodell

Datenmodellierung - Schichtenmodell:

1. externe Schema (1..n) – beschreiben die Datenbank aus der Sicht des Nutzers
2. konzeptionelles Schema – Logisches Datenmodell, unabhängig vom Betriebssystem und der rechnerinternen Realisierung. Es wird der logische Aufbau der Datensätze und der Datei definiert. Zugriffsinformationen sind enthalten.
3. internes Schema – beschreibt die physikalische Architektur der Daten auf einer bestimmten Speicherarchitektur
4. Datei

Entwurfsphase - Implementationsmodell

Prozessmodellierung:

Prozess (nach DIN6620): „Der Prozess ist die Umwandlung und/oder der Transport von Materie, Energie und/oder Information. Die Zustandsgrößen technischer Prozesse können mit technischen Größen gemessen und/oder geregelt werden.“

Prozessormodell:

Als Prozessor wird eine aktive Komponente eines Systems bezeichnet, die Einrichtungen zur Aufnahme von Informationen von einem Ereignis zur Verarbeitung und ggf. Speicherung sowie zur Weiterleitung einer Antwort besitzt.

Das Implementationsmodell legt fest, welcher Prozessor welche Algorithmen zu bearbeiten hat.

Entwurfsphase - Implementationsmodell

Modellierung der Benutzerschnittstelle:

In der Regel verfügt jedes Datenverarbeitungssystem über eine Benutzerschnittstelle. Die Gestaltung ist von der geplanten Anwendergruppe abhängig.

mögliche Fragestellungen:

Welche Informationen müssen dem Nutzer zur Verfügung gestellt werden?

Welche Handlungsfähigkeit hat der Nutzer?

Wie reagiert das System auf Fehleingeben?

Wie wird die Nutzerunterstützung realisiert?

Entwurfsphase - Implementationsmodell

Datenabstraktion:
Es wird eine Abbildung der logischen Struktur auf eine geeignete physische Struktur innerhalb des Moduls vorgenommen. Gegebenenfalls werden Objekte mit der entsprechenden Kapselung geschaffen.

Funktionale Abstraktion:
Alle auszuführenden Aktionen werden dahingehend untersucht, ob sie in gleicher oder ähnlicher Form an verschiedenen Stellen des Systems benötigt werden oder ob sie sich aus Aktionen zusammensetzen lassen, die weniger spezialisiert sind. Funktionsmodule haben kein Gedächtnis. Es werden ausschließlich Funktionen bereitgestellt, die Eingangsdaten in die entsprechenden Ausgangsdaten transformieren.

Entwurfsphase - Modulspezifikation

Mit der Modulspezifikation wird eine genaue Beschreibung eines jeden Moduls vorgenommen. Darin enthalten ist:

- die Beschreibung des Modulkopfes,
- eine vollständige Definition der Schnittstellen, getrennt nach Import und Export,
- der Modulname,
- eine kurze verbale Beschreibung der Modulfunktionalität,
- der Name des Bearbeiters des Moduls und
- das Datum der letzten Änderung.

➔ **Definition der notwendigen Algorithmen in einer entsprechenden Darstellungsform**

Entwurfsphase - Modulspezifikation

Die Beschreibung der Anforderungen an die Eingangsdaten erfolgt durch die **Vorbedingung P(V)**. Eine Beschreibung des Zustandes der Ausgabedaten nach Verlassen des Blockes erfolgt durch die **Nachbedingung Q(V)**.

Entwurfsphase - Konstruktion von Algorithmen

Top-Down-Entwicklung und schrittweise Verfeinerung

Ausgehend von einer eindeutigen Spezifikation des Problems werden die Ebenen der Hierarchie stufenweise realisiert. Dazu werden die anstehenden Probleme schrittweise in Teilprobleme mit immer feinerem Detailliertheitsgrad zerlegt, bis man auf der untersten Ebene angelangt ist. Auf dieser Ebene lassen sich die algorithmischen Aktionen unmittelbar in einer dem Rechner verständlichen Programmiersprache ausdrücken.

(Dijkstra)

Entwurfsphase - Entwurf von Algorithmen

Die **schrittweise Verfeinerung** bis zur konkreten algorithmischen Lösung von Problemen und Teilproblemen bis auf das Niveau von Pseudocode und Programmiersprache wird als *„Programmieren im Kleinen“* bezeichnet.

Verfahren der schrittweisen Verfeinerung:

1. grobe Spezifikation des Algorithmus (durch Vor- und Nachbedingung)
2. Definition der algorithmischen Aktionen, die zur Erfüllung der Nachbedingung auszuführen sind, Spezifikation der Zwischenzustände, Deklaration der zugehörigen Daten
3. interaktive Verfeinerung der Beschreibung von Aktionen, bis diese durch entsprechende Konstrukte einer Programmiersprache ausgedrückt werden können

Entwurfsphase - Entwurf von Algorithmen

Simultane Konstruktion und Verifikation von Algorithmen

1. Formulierung von Zusicherungen
2. Verifikation der logischen Übergänge zwischen Berechnungszuständen

Verifikation:

Verifikation [lat.], die Überprüfung einer (logisch- analyt. oder empir.) Aussage auf ihre Richtigkeit; **verifizieren**, durch Überprüfen die Richtigkeit einer Hypothese bestätigen.- Ggs. Falsifikation.

Informatisches Modellieren - Implementierung

Inhalte	Hinweise zum Unterricht
– Implementierung • Ergebnis: lauffähiges dokumentiertes Programm	– Programm entspricht dem Ein-/ Ausgabeverhalten der Anforde-rungsdefinition

Programmierung/Implementierung

Implementierung:

korrekte Umsetzung der Modulspezifikation in Code für eine Basismaschine.

Basismaschine:

zur Verfügung stehende Hardware und Software (Betriebssystem, Entwicklungsumgebung, Compiler, Datenbanken)

Kontext des Gesamtsystems beachten:

1. Sinnvolle Reihenfolge der Implementierung der Moduln
2. Einheitliche Richtlinien für die Programmierung der einzelnen Moduln

Informatisches Modellieren – Funktions- und Leistungsüberprüfung

Inhalte	**Hinweise zum Unterricht**
– Funktions- und Leistungsüberprüfung • Ergebnisanalyse und Bewertung • Ergebnis: modifiziertes Programm	– Verifikation und Testen der Einzelmodule – Integrationstest – Installationstest – Laufzeit- und Speicherverhalten

Programmierung/Implementierung - Test

Die Testphase hat die Aufgabe, die **Zuverlässigkeit** und **Korrektheit** der Software nachzuwei-sen.

Dies wird durch Fehlerbeseitigung und Vergleich von Soll- und Isteigenschaften realisiert. Es wird die Verifikation und Validierung durchgeführt.

Programmierung/Implementierung - Validierung

Von SARGENT wird die Validierung in drei Stufen eingeteilt.

1. DATENVALIDIERUNG

Die geschaffene Datenbasis muss den sicheren Übergang von einer Stufe des Modells zur nächsten gewährleisten.

2. VALIDIERUNG DES ABSTRAKTEN MODELLS

In dieser Stufe der Modellvalidierung wird festgelegt, dass die Theorie und Grundlagen sowie die mathematischen, logischen und kausalen Zusammenhänge für das gesamte Modell sowie für jedes Untermodell für die beabsichtigte Nutzung richtig sind. Die Korrektheit der Abstraktion muss überprüft werden.

3. WIRKSAMKEITSVALIDIERUNG

Diese Stufe der Modellvalidierung sichert, dass das Modell die Problemstellung für den geforderten Einsatzzweck richtig wiedergibt.

VALIDIERUNG bezieht sich immer auf ein konkretes System und kann nicht allgemein durchgeführt werden.

Programmierung/Implementierung - Testphase

*** Regeln für die Durchführung der Testphase** (nach Rembold):

1. Die Tests müssen so ausgelegt werden, dass **jede Konsequenz einmal durchlaufen** wird.

2. Es müssen die **oberen und unteren Grenzwerte** erprobt werden, und zwar Werte, die gerade noch gültig sind und Werte, die gerade nicht mehr gültig sind.

3. Es müssen **repräsentative Daten** aus den Gültigkeitsbereichen erprobt werden.

4. Es müssen **Testläufe** mit ungültigen Daten durchgeführt werden.

Programmierung/Implementierung - Integration

- die entwickelten und getesteten Moduln werden zu einem Gesamtsystem zusammengefügt
- bei der parallelen Entwicklung von Moduln ist die Verwendung von Testrahmen möglich

Arbeit unter Laborbedingungen

Programmierung/Implementierung - Installation

- das lauffähige Gesamtsystem wird in die Softwareumgebung des Anwenders integriert
- die für die Arbeit des Systems notwendigen Datenbestände müssen bereitgestellt werden

Arbeit in der realen Systemumgebung

➔ ***Abnahmetest***

3. Entwurfs- und Implementierungsphase

Die Schülerinnen und Schüler

- sind in der Lage, eine projektbegleitende Dokumentation zu erstellen,
- begreifen den Softwarelebenszyklus als iterativen Prozess und erkennen dessen wirtschaftliche Bedeutung.

Informatisches Modellieren – Installation und Abnahme / Dokumentation

Inhalte	Hinweise zum Unterricht
– **Installation und Abnahme**	– Ergebnis: fertiges Softwareprodukt entsprechend der Anforderungsdefinition
– **Dokumentation**	– die Dokumentation ist über den gesamten Zeitraum dieser Unterrichtseinheit zu führen – die Dokumentation kann als Hilfesystem erzeugt werden

Programmierung/Implementierung - Dokumentation

Bestandteile der Systemdokumentation

- Spezifikation (vollständige Ergebnisse der Phase Analyse)
- Systemarchitektur (vollständige Ergebnisse der Phase Entwurf)
- Programmentwurf (Entwurfsunterlagen wie Ablaufpläne, Struktogramme, Datenflusspläne)
- kommentierte Modulquelltexte
- Testplan und Ergebnisprotokolle
- Inhalt und Ergebnis des Abnahmetests

Programmierung/Implementierung - Dokumentation

Bestandteile der Anwenderdokumentation

- Bedienungsanleitung (kurze Beschreibung der Funktion des Systems für den Endnutzer)
- Installationsanleitung (Vorgehensweise bei der Installation des Systems auf der gewünschten Hardware)
- Referenzhandbuch (ausführliche Anleitung für die Benutzung des Systems, detaillierte Informationen über Bedingungen und Abläufe im System)
- Administrationshandbuch (notwendige Unterlagen für den Systemverantwortlichen, z.B. Verwaltung der Nutzer und Zugriffsrechte, Inhalte und Verwendung von Logfiles)

⇨ Viele Dokumentationsteile werden heute in Hilfesysteme integriert.

Einsatz/Wartung

Wartung:
Als **Wartung** (maintenance) eines Softwaresystems werden alle nach dem Abschluss der Entwicklung vorgenommenen Arbeiten bezeichnet, die eine Modifikation oder Korrektur des Systems bewirken.
/Horn97/

Aspekte des informatischen Modellierens

- ✓**Ziele des zu erstellenden Informatiksystems herausarbeiten**
- ✓**Phasen des Modellbildungsprozesses**
  - ✓**Modell-Erzeugung**
    - ✓interdisziplinäre Problemanalyse
    - ✓heuristisches Vorgehen in der Modellbildung
    - ✓Abstraktion und Reduktion (z.B. ADT)
    - ✓(Formale) Beschreibungsformen für die relevanten Daten und Prozesse
    - ✓...
  - ✓**Modell-Umsetzung**
    - ✓Operationalisierbarkeit, Automatisierbarkeit, Implementierung
    - ✓Interaktion Mensch und Modell
    - ✓...
  - ✓**Modell-Kritik**
    - ✓Einbezug von Optimierungsaspekten (Verbesserung, Anpassung)
    - ✓interdisziplinäre Validierung im soziotechnischen Kontext
    - ✓Reflexion von Simulations- und Visualisierungsergebnissen
    - ✓...
  - ✓**Modelldokumentation**
- ✓**Iteration des Modellbildungsprozesses**

(Arbeitsgruppe Modellbildung und Simulation - Königsteiner Gespräche 2001)