

wissensbasiertes System

- ersten Ebene: Dialogkomponente (Anwender), Erklärungskomponente Wissenserwerbskomponente (Entwickler).
- Inferenzkomponente
- statischen und dynamischen Wissensbank.

Unternehmung

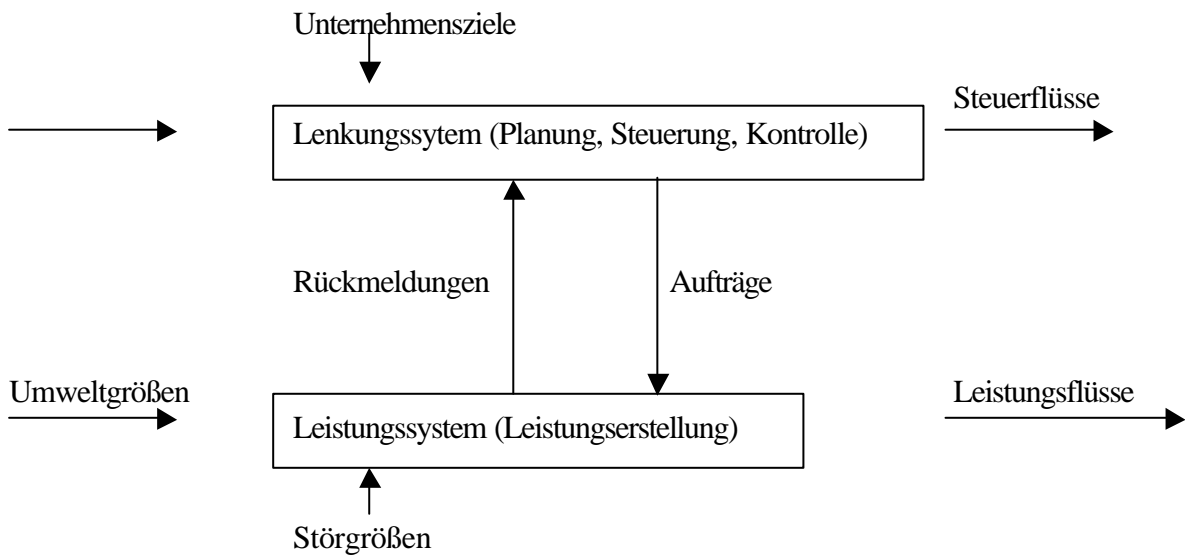
- offenes (Leistungsaustausch mit der Umwelt),
- zielgerichtetes (Unternehmensziele, das Zielsystem besteht aus Sach- und Formalzielen) und
- soziotechnisches (personelle und maschinelle Aufgabenträger) System.

Formalziele:

- kostengünstig
- Qualität
- Größe
- Aussehen
- Termineinhaltung

Sachziele:

- Entwicklung eines AWS



Informationssystem:

- Objektart Information
- Lenkungssystem und im Falle des Dienstleistungsgewerbes aus den Bereichen des Leistungssystems, die ebenfalls die Objektart Information verarbeiten.
- Aufgabenebene und der Aufgabenträgerebene des Informationssystems
- Zwischen den Aufgabenträgern Kommunikationsbeziehungen.
- Computer-Computer- oder einer Computer-Mensch-K: **Anwendungssystem**

Bei einem Anwendungssystem greift die Nutzermaschine über ein Programm aus die Basismaschine zu.

Projektablauf

- Anwendersicht : Die Aufgaben definiert. (Nutzermaschine).
- Aufgaben greifen dann über die Softwareentwicklung (Programm) auf die aus
- Entwicklersicht durchgeführte Aufgabenträgerdefinition (Basismaschine) zu.

Aufgaben

- Bausteine für die Gestaltung zielgerichteter Systeme.
- **Außensicht der Aufgabe**
 - o Aufgabenobjekt
 - o Aufgabenzielen (Sach- und Formalziele)
 - o Vor- und Nachereignissen für die Aufgabendurchführung.
- **Innensicht**
 - o Lösungsverfahren
 - o Bezug auf die aufgabenspezifischen Formalziele und durch einen Bezug auf den vorgesehenen Aufgabenträgertyp wird der Erfolg programmiert.
 - o **VOS** die Vorereignisse, die nach der Aktionensteuerung zu Nachereignissen führen
 - o **KOS** beschreibt man die Interaktion zwischen Aktionen und Aufgabenobjekt.

Der **Aufgabenträger** der Aufgabe legt den Automatisierungsgrad der Aufgabe fest. Man beschreibt diesen Grad mit (Aktionsdurchführung, Aktionssteuerung, Vorgangsauslösung).

Funktionen

- enumerativ in Tabellen
- Algorithmen
- deklarativ

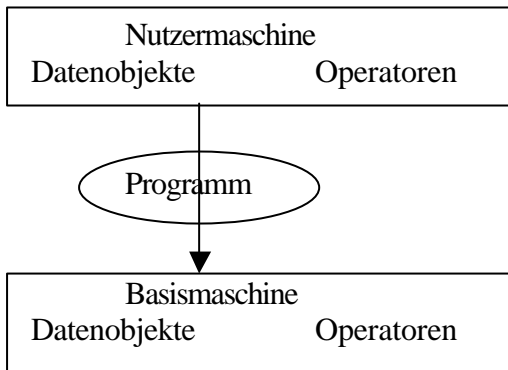
Algorithmus

- Beschreibung einer Funktion anhand von Teilfunktionen und deren Ablaufbeziehungen
- endliche viele Teilfunktionen
- Funktionswert in endlicher Zeit
- Teilfunktionen maschinell durchgeführt werden.
- Teilfunktionen, Zerlegungsbeziehungen zwischen Funktion und Teilfunktion und die Ablaufbeziehungen zwischen den Teilfunktionen stellen dabei die Komponenten eines Algorithmus dar.

Bei der **deklarativen Funktionsbeschreibung** wird ein System von Relationen R_1, R_2, \dots, R_n beschrieben, die über gemeinsame Attribute verknüpft sind. Die Beschreibung ist dabei vorvollständig und nacheindeutig. Ein Beispiel einer Relation ist:

untersteht(MitarbeiterA,Chef) :- berichtet(MitarbeiterA,Chef)

Ein **Programm** besteht aus Operatoren (Anweisungen an das Rechenwerk, Ablaufstrukturen (Anweisungen an das Steuerwerk) und Datenobjekten (im Hauptspeicher)



Die Nutzermaschine bietet dabei eine Anwendungsplattform an, während die Basismaschine für das Programm eine Systemplattform anbietet.

ADK-Methode

- Anwendungsfunktionen
- Datenverwaltung
- Kommunikationssystem
- Ziel:
 - o Flexibilität
 - o Reduzierung der Komplexität
 - o die Standardisierung
 - o Portierbarkeit

Abstrakter Datentyp

- Information Hiding (oberste Abstraktion)
- Trennung zwischen Innen- und Außensicht
- Typdefinition
- Externe Datenobjekttypen sind für Nutzer dabei nur über Operatoren zugänglich.
- Spezifikation kann getrennt von konkreten Anwendungen erfolgen.

Objekttyp

- Objektnamen
- objektinternen Speicher
- Operatoren.

Objektorientierte Programme

- Die Datenkapselung (Instanz eines ADT mit den Merkmalen Information Hiding, Trennung Außen-/Innensicht und die axiomatische Spezifikation), den Polymorphismus und die Vererbung:

```

Public class Kunde{
private int _Nr;
private String _Name;
}

```

```

public class Privatkunde extends Kunde{
private string _Anschrift;
}

```

```

public class Webservice {
private vectorwebservice = new vector();
}

```

erbt eigentlich von 2 Seiten

Programmstruktur

- greift das Hauptprogramm über ein Interface auf die verschiedenen Basismaschinen zu. (B-N-Maschine)

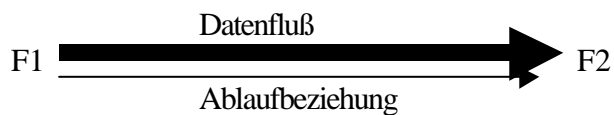
Datentypen

- Datenobjekttyp
- darauf definierten Operatoren
- Komplexe Datenobjekttypen werden in
 - o statische (homogenes
 - o heterogenes kartesischen Produkt)
 - o dynamische Datenstrukturen (lineare Liste, hierbei hat jedes Element genau einen Vorgänger und einen Nachfolger)

Verschiedene **Ablaufstrukturen** beschreiben den inneren Aufbau des Programmes.

- Sequenz (begin F1; F2; End) geht linear vor.
- Alternativen (if p then F1 else F2) wird nur ein Weg gegangen
- Wiederholungen (repeat F1 until p) werden getätigt, bis p eingetreten ist.

Die Ablaufbeziehungen eines Programmes sind dabei datenflussbedingt oder betriebsmittelbedingt.



Die Basismaschine ist eine virtuelle Maschine mit den virtuellen **Betriebsmitteln** virtueller Prozessor, virtueller Speicher und virtuelle Peripherie.

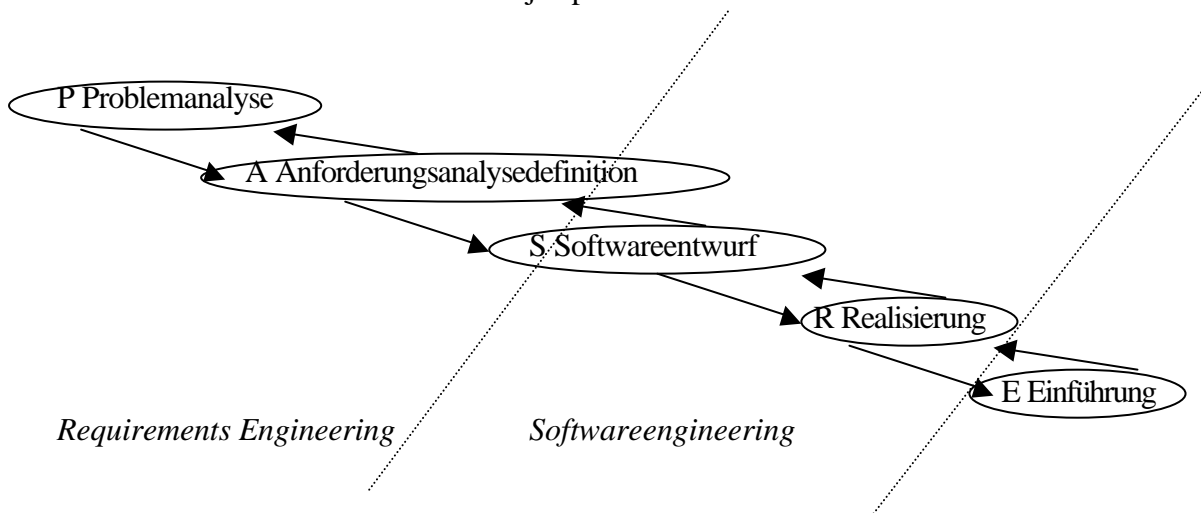
Phasenmodelle

- Entwicklungsdauer von Softwareprojekten
- zunehmende Komplexität der Softwareentwicklung
- die Realisierung in Mehrpersonenprojekten und die
- Kosten der Entwicklung

Wasserfallmodell

- sequentielle, kaskadische Struktur
- Softwareentwicklung nach ingenieurmäßigen Prinzipien
- bei jedem Phasenübergang erfolgt eine Erstellung und Abnahme von Dokumenten.
- dabei wird jede Phase durch unterschiedliche Methoden, Werkzeuge und Vorgehensweisen unterstützt.
- nach der Entwicklung erfolgt neben der Nutzung der Software auch die weitere Wartung.

Das Wasserfallmodell beschreibt 5 Projektphasen:



Kritisch:

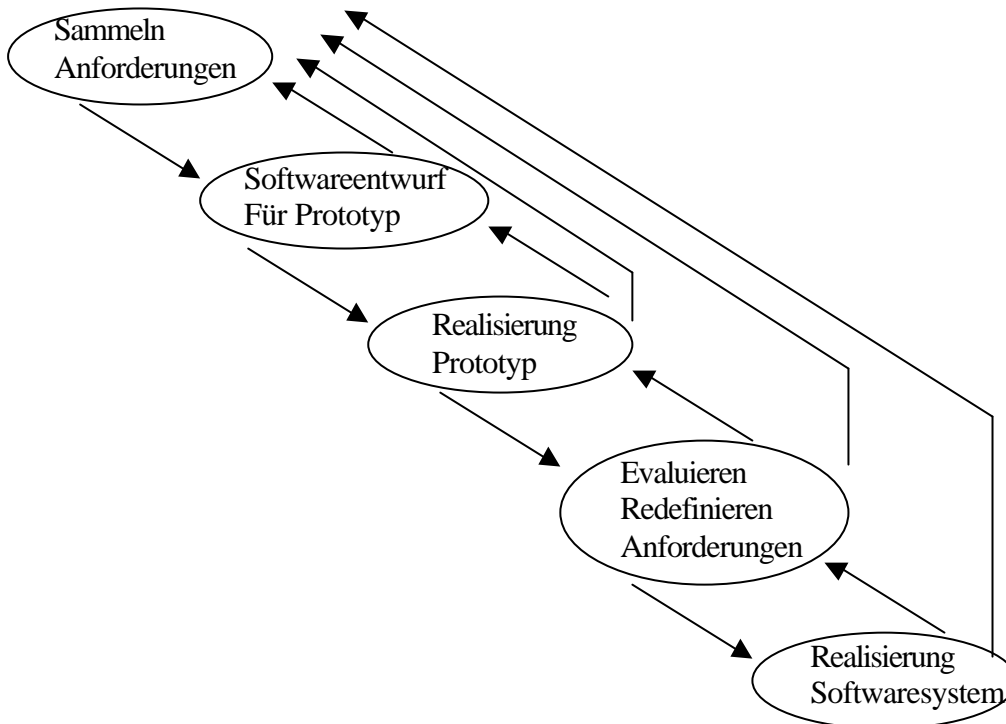
- streng sequentielle Projektablauf nicht realistisch
- Anforderungen können bei Projektbeginn nicht vollständig formuliert werden
- nur für statische, technische Systeme geeignet, nur bedingt für lebende Organisationen.

Requirement Engineering

- Problemanalyse, erfolgt eine Erfassung und Analyse der Aufgabenstellung sowie der Forderungen und Sollvorstellungen. Dies stellt eine grobe Istaufnahme und Istanalyse dar. Rahmenbedingungen und Ziele werden festgelegt und in einem Lastenheft festgehalten.
- zweiten Phase, der Anforderungsanalyse und –definition wird ein Pflichtenheft erstellt, welches eine feine Istaufnahme und Istanalyse beinhaltet.

Prototyping-Projektmodell

- Verbesserung der Anforderungsdefinition hinsichtlich Konsistenz, Detaillierungstiefe und Abstimmung mit dem Auftraggeber.
- Während Anforderungsanalysedefinition wird Vorläufer des Softwaresystems mit allen fachlichen Anforderungen erstellt.



Problematisch

- vollständige Formulierung der Anforderungen an das System benötigt
- dringende Notwendigkeit ist des weiteren eine wechselseitige Koordination zwischen Entwickler und Anwender während der Entwicklung
- Realisierbarkeit des Projektes lässt sich nicht immer garantieren (je nach Umfang des Softwareprojektes).
- die Überzeugkraft des Verkäufers ist bei diesem Phasenmodell in der Akquisitionsphase gefragt.

Arten von Prototypen

- Demonstrationsprototyp dient der Kundenacquisition
- der Prototyp im engeren Sinne veranschaulicht spezielle Aspekte
- Labormuster dient der Demonstration der technischen Machbarkeit des Projektes
- das Pilotsystem umfasst den Kern des Produktes

Vorteile des Prototyping

- Reduzierung des Entwicklungsrisikos
- wiederverwendet
- schnell erstellt werden
- starke Rückkopplung zwischen dem Endbenutzer und dem Auftraggeber statt.

Nachteile

- der höhere Entwicklungsaufwand,
- die schlechte Dokumentation der Prototypen,
- die unbekanntes Grenzen des Prototyps
- Gefahr eines Wegwerfprototypen

Spiralmodell

- Minimierung des Entwicklungsrisikos
- vier Phasen durchlaufen wieder zu Phase eins
- ersten Phase die Ziele, Anforderungen und Alternativen festgelegt
- zweite Phase bewertet diese Alternativen durch Prototypen, Risikoanalysen, Simulationen und Benchmarks.
- dritten Phase wird ein phasenbezogenes Produkt wie beim Wasserfallmodell erstellt.
- vierte Phase dient der Planung der nächsten Lösungsschritte.

- risikogetriebenes Modell.
- Jede Spirale stellt einen iterativen Zyklus durch dieselben Schritte dar. Die Ziele für jeden Zyklus werden dabei aus den Ergebnissen des letzten Zyklus abgeleitet.

Vorteile

- Überprüfung und eventuelle erneute Festlegung des Prozessablaufs in Abhängigkeit von den Risiken.
- variabel und wird nicht für die gesamte Entwicklung festgelegt
- andere Prozessmodelle integriert werden
- Fehler und ungeeignete Alternativen werden dabei frühzeitig eliminiert.
- sehr flexibel
- Wiederverwendbarkeit der Software.

Kritisch

- sehr hohen Managementaufwand benötigt.
- Management und die Identifizierung von Risiken ist dabei sehr problematisch.
- für kleine und mittlere Projekte nicht geeignet.

Projektmanagement

- Planung, Steuerung und Kontrolle der durchzuführenden Aufgaben
- Definition der Projektaufgaben gemäß dem Phasenmodell und der geplanten Produktstruktur (Projektstrukturplan).
- Ermittlung der zeitlichen Abhängigkeiten zwischen den Aufgaben anhand von logischen Abhängigkeiten (Netzplan: Vorgang, Meilenstein und zeitliche Abhängigkeit)
- Ermittlung der Ressourcen bezüglich Qualifikationsprofil und Quantität (Personengebörge).

Am Ende der Projektplanung erfolgt der **AZRF-Plan** (Aufgaben, Zeit, Ressourcen und Finanzen). Er verfeinert, konkretisiert und ergänzt das ausgewählte Prozessmodell.

Kriterien der Aufgabenabgrenzung

- Aufgabenergebnisse klar definierbar und prüfbar sein müssen
- Aufgaben müssen von den vorgesehenen Aufgabenträgern durchgeführt werden
- Leistungsbeziehungen zwischen den Aufgaben vollständig und konsistent sein.

Projektplanung und Projektsteuerung

- beauftragt die Aufgabenpakete
- zeitliche und fachlich inhaltliche Kontrolle der Aufgabenergebnisse statt
- Projektdaten müssen aktualisiert werden
- Berichte und Analysen erstellt
- Informationen an alle Projektbeteiligten weitergegeben.

Projektleiter

- Kopf
- technischen, wirtschaftlichen und organisatorischen Probleme lösen
- engen Kontakt zum Nutzer und Auftraggeber
- Termineinhaltung und die Wahrung der Vertragsrechte und Vertragspflichten garantieren.

Pflichtenheft:

- Zielbestimmung/ Qualität
- Produkteinsatz
- Produktumgebung
- Produktfunktion
- Produktdaten
- Benutzeroberfläche
- Entwicklungsumgebung

Qualitätsmanagement

- produktorientiert und prozessorientiert

Qualitätssicherung

- konstruktiven und Analytischen Maßnahmen
- Konstruktiven Qualitätssicherungsmaßnahmen wird im vorhinein durch das Setzen von Standards, Meilensteinen und festen Rollen für Teammitglieder ein bestimmtes Niveau gehalten. Technische Maßnahmen und organisatorische Maßnahmen werden ergriffen.
- analytischen Qualitätssicherungsmaßnahmen erfolgt eine Codeinspektion, ein Modultest und ein Systemtest zur Überprüfung. Analysierende und testende Verfahren dienen der Analyse.

Qualitätsmerkmale

- Brauchbarkeit (Funktionstüchtigkeit, Betriebstüchtigkeit, Benutzungsfreundlichkeit),
- Pflegbarkeit (Transparenz, Änderbarkeit, Erweiterbarkeit)
- Anpassbarkeit (Portabilität, Wiederverwendbarkeit, Integrationsfähigkeit)

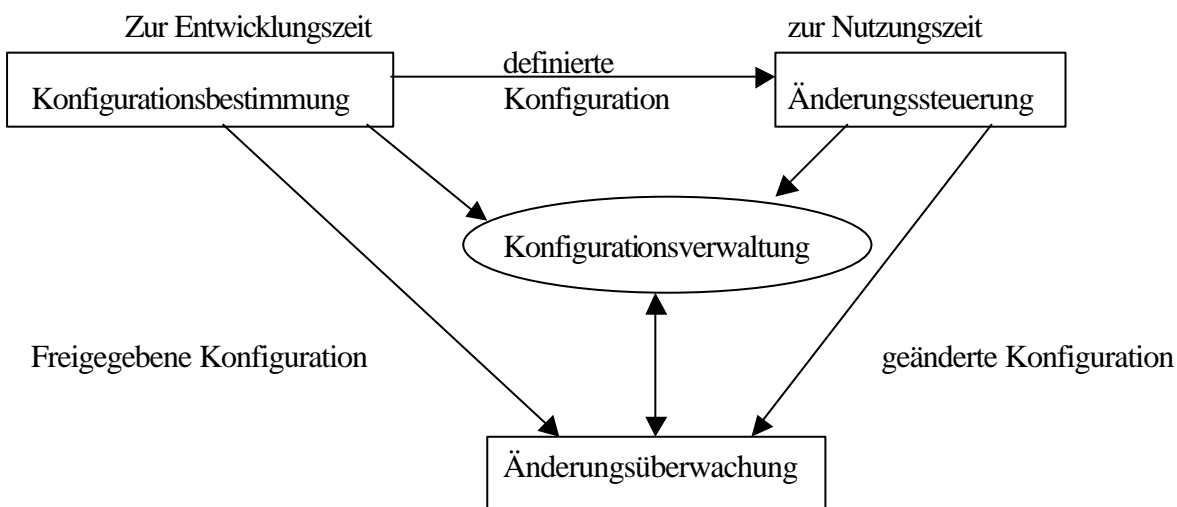
Analytische Qualitätssicherungsmaßnahmen

- Formalprüfung, also die Stichprobenweise Prüfung auf Einhaltung der Regeln gemäß QS-Plan oder Projekthandbuch.
- Zwischenproduktprüfung beinhaltet die Prüfung auf Einhaltung formaler Vorgaben, externer und interner Konsistenz, die Methode dabei ist der Walkthrough
-

Qualitätssicherungsmaßnahmen

- **Initialisierung** (Q-Merkmale, Q-Maße, QS-Plan, Testplan),
- **Qualitätssicherungsplan** (Produktbezogen)(Q-anforderungen, Standards und Richtlinien, Organisation der QS, Festlegung von Baselines, Methoden, Werkzeuge)
- **Testplan** (Technische Produktprüfung) (Auflisten der Testobjekte, Rahmen der Testdokumentation, Testvorbereitung, Testdurchführung, Testauswertung).
- Mängelliste
 - o Prüfung,
 - o Folgemaßnahmen
 - o verwendeten Projektwerkzeuge

Das **Konfigurationsmanagement** dient der Ordnung der Vielfalt von Objekten.



Konfigurationsmanagementaktivitäten

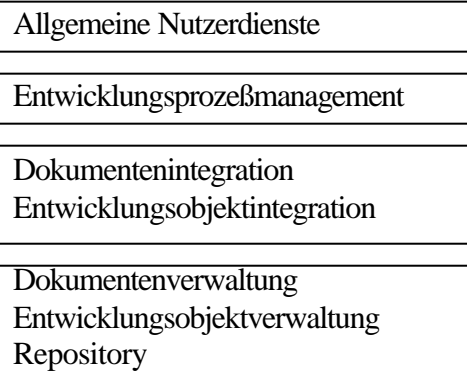
- Die **KM-Initialisierung** macht aus dem Input Projekthandbuch, Projektplan und KM-Plan wird durch die Initialisierung das Output Baseline-Festlegungen, Teamorganisation, Kontrollgremien und Änderungsgremien.
- Bei der **Konfigurationsbestimmung** erfolgt aus dem Input Fachkonzept, DV-Konzept und den SW-Anforderungen das Output Konfigurationslisten.
- Bei der **Änderungssteuerung** erfolgt aus dem Input Problemmeldung, Änderung, SW-Anforderungen und Baselineprodukte das Outputprodukt Durchführungsantrag und Ablehnungsbescheid.
- Die **Konfigurationsverwaltung** benutzt die Input neues Produkt, zugriffsrechte und alle Produkte. Als Output werden zugriffsrechte gewährt und Backupband.

Das Konfigurationsmanagement hat **Probleme** bezüglich häufige Änderungen, dem Beheben von Fehlern, der Nachvollziehbarkeit und der Vollständigkeit.

Das **Konfigurationsidentifikationsdokument** hat eine Definitionsphase, Entwurfsphase, Einführungsphase und eine Pflegephase.

Die **Aufgaben** des Konfigurationsmanagement beinhalten die Planung des Konfigurationsmanagements (Konfigurationsmanagementplan, Produktbibliothek), die Produkt- und Konfigurationsverwaltung (Archivierung der Konfiguration, Datensicherung) und das Änderungsmanagement (Verwaltung, Entscheidung, Bearbeitung veranlassen, Information aller Betroffenen).

CASE: Computer Aided Software Engineering.



Allgemeine Ziele

- erhöhen der Produktivität
- Verbesserung der Qualität
- Verbesserung des Entwicklungsmanagement
- Entwicklungstechnische Ziele (Methodenbasiert entwickeln, Qualität der Dokumentation, Wiederverwendbarkeit, Schwachstellenanalyse).
- wirtschaftlichen Ziele (Effektivität, Effizienz, Entwicklungszeit, Wartungsaufwand).
- organisatorische Ziele (Unterstützung des Prozessmodells, Standardisierung, Kontrolle der Soll/Ist-Zustände).

Komponenten eines CASE-Systems sind

- allgemeine Nutzer-Dienste, wie Präsentations- und Entwicklungswerkzeuge, Verwaltung und Bearbeitung.

- Bei dem Entwicklungsprozessmanagement erfolgt die Definition und der Aufruf von Entwicklungsschritten.
- Die Dokumentenverwaltung basiert auf einem Datenbanksystem.
- Die Dokumentenintegration erlaubt den Datenaustausch und die Verwaltung.
- Werkzeuge wie Upper CASE, Lower CASE und I-CASE.

Architekturmerkmale von AWS:

Client-Server-Systeme

1. Zentrale Architektur KAD
 1. K: Browser
 2. A: Webserver
 3. D: Daten
2. Dezentrale Kommunikation (Fat Server) K-AD
 - a. Nutzung von GUI
 - b. Probleme:
 - i. Kapazität des Kommunikationskanals K-A
 - ii. Leistung von A und D
3. Dezentrale KA-D (Fat Client)
 - a. Vorteile:
 - i. hohe Rechnerleistung und
 - ii. verteilte Datenhaltung
 - b. Probleme:
 - i. Kapazität des Kommunikationskanals
 - ii. Leistung von D
4. Dezentrale Kommunikation KA1-A2D
 - a. Vorteile:
 - i. Zuordnung vollautomatisierter Aufgaben zu A2
 - ii. Verwendbarkeit standardisierter Protokolle
 - iii. Hohe Freiheitsgrade für die Aufteilung
5. **Merkmale von Client-Server-Systemen:**
 - a. Dienste: Server stellen den Clients Dienste zur Verfügung, diese werden über ein vereinbartes Protokoll aufgerufen
 - b. Gemeinsame Ressourcen: Server kann mehrere Clients bedienen
 - c. Standorttransparenz: Ort des Serverprozeß ist unbekannt
 - d. Skalierbarkeit: Hinzufügen und Entfernen von Clients

GUI – Graphical User Interface:

1. Non gui Clients:
 - a. Minimum an menschlicher Interaktion ohne Bedarf für Multitasking
 - b. Minimum an menschlicher Interaktion mit Bedarf für Multitasking
2. gui Clients:
 - a. Dialoge: AWS

Transaktionskonzepte :

1. Klassifikation von Lenkungssystemen nach der Art der Lenkung (zeitdiskret/zeitkontinuierlich)
2. Gestaltung von Transaktionen nach dem a-v-dk-Konzept
 - a. Sachziel der Transaktion: Leistungsaustausch zwischen Sender und Empfänger
 - b. Formalziel: sind transaktionsspezifisch, z.B. Kostenminimierung
 - c. 3 Phasen:
 - i. Anbahnung: Transaktion kann rückgängig gemacht werden
 - ii. Vereinbarung: Transaktion muß bei Abbruch kompensiert werden
 - iii. Durchführung/ Kontrolle: Transaktion muß bei Abbruch kompensiert werden
3. ACID-Prinzip:
 - a. Atomicity: T ist nicht Unterbrechbar
 - b. Consistency: T wahrt die Konsistenz der Datenbasis
 - c. Isolation: mehrere T laufen ohne gegenseitige Beeinflussung ab
 - d. Durability: Änderungen sind dauerhaft

Datenintegration

- starke Rückkopplung (gemeinsamer Speicher der Prozessoren)
- schwache Rückkopplung (globales Kommunikationssystem mit Prozessor-Speicher) sein.

Die **Diskurswelt** ist der Ausschnitt der betrieblichen Realwelt, der zu modellieren ist. Die Diskurswelt ist ein offenes, soziotechnisches und zielgerichtetes System.

Bei der Modellierung erfolgen **folgende Schritte**:

1. Systemabgrenzung
2. Erstellen Gesamtmodell der Diskurswelt
3. Automatisierungsentscheidung
4. Gestaltung von AWS

1.