
Grundlagen Software Engineering

Wiederverwendung

Inhalt

- ☐ Zur Problematik
- ☐ Wiederverwendbarkeit und Wiederverwendung
- ☐ Technik
- ☐ Organisation
- ☐ Management
- ☐ Kosten/Nutzen der Wiederverwendung
- ☐ Einführung der Wiederverwendung

Zur Problematik

☐ Problematik

- Bei jeder Software-Entwicklung wird in der Regel das »Rad« mehrmals neu erfunden
- Luxus, der auf Dauer nicht bezahlt werden kann

☐ /Weber 92/

- »Die Wiederverwendbarkeit von Produktkonzepten, von Produkten, aber auch von Verfahrensweisen ist das zentrale technologische Konzept in den hochentwickelten westlichen Industrien
- Mit der Wiederverwendung wird sowohl eine Kostensenkung als auch eine Qualitätsverbesserung der Produkte angestrebt.«

Zur Problematik

- »Diese Vorgehensweise, bei der darauf verzichtet wird, existierende Produkte als Komponenten zu verwenden und für jedes Produkt und jede zu entwickelnde Komponente die Marktchancen vor ihrer Entwicklung abzuschätzen, ist – so kann man es wohl sagen – industrielle Steinzeit.«

☐ »Golden Rule of Reuseability:

- *Before you can reuse something, you need to*
- *1st find it*
- *2nd know what it does*
- *3rd know how to reuse it.*«

Zur Problematik

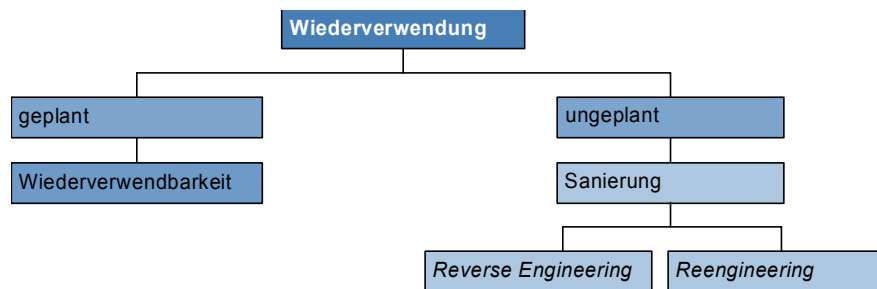
- ☐ Ziele der Wiederverwendung
 - Signifikante Produktivitätserhöhung
 - Qualitätsverbesserung der Produkte
 - Verkürzung der Entwicklungszeit
 - Reduzierung der Kosten

Wiederverwendbarkeit & Wiederverwendung

- ☐ Wiederverwendbarkeit (*reuseability*)
 - Erstellen und Bereitstellen wiederverwendbarer Software
- ☐ Wiederverwendung (*reuse*)
 - Einsatz wiederverwendbarer Software
 - Hochgradig wiederverwendbare Software-Komponenten erleichtern die Wiederverwendung dieser Komponenten
- ☐ Voraussetzungen: Optimales Zusammenwirken von
 - Technik
 - Organisation
 - Management

Wiederverwendbarkeit & Wiederverwendung

☐ Geplante vs. ungeplante Wiederverwendung



Wiederverwendbarkeit & Wiederverwendung

☐ Heute: Ungeplante Wiederverwendung

- Vorhandene Software-Systeme
 - müssen aufgrund geänderter Hardware- und Systemsoftware-Plattformen und/oder zusätzlicher Produkthanforderungen neu erstellt werden
- Ursprüngliche Systeme
 - nicht so konzipiert, um später in anderen Kontexten wiederverwendet zu werden
- Einsatz von *Reverse Engineering*, *Reengineering* und Sanierung
 - um Software-Komponenten für Wiederverwendung zu identifizieren und aufzubereiten

Wiederverwendbarkeit & Wiederverwendung

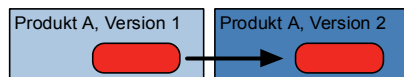
☐ Geplante Wiederverwendung

- Software-Komponenten von vornherein so konzipieren, dass sie später gut wiederverwendet werden können
- Gute Wiederverwendbarkeit bedeutet, dass die Software-Komponenten
 - einen hohen Allgemeingrad besitzen
 - qualitativ hochwertig sind
 - gut dokumentiert sind
- Software-Komponente
 - Jedes explizite, physikalisch vorhandene Arbeitsergebnis, das erstellt wurde
 - Jedes Arbeitsergebnis, das eine Problemlösung für spätere Projekte leicht verfügbar macht

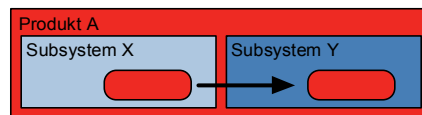
Wiederverwendbarkeit & Wiederverwendung

☐ 4 Typen der Wiederverwendung

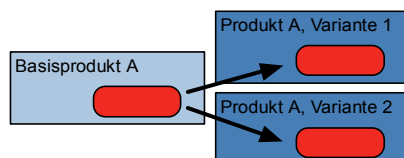
Wiederverwendung bei Versionsentwicklung



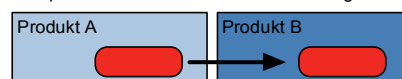
Intraproduktorientierte Wiederverwendung



Wiederverwendung bei Variantenentwicklung



Interproduktorientierte Wiederverwendung



Legende:



Wiederverwendbarkeit & Wiederverwendung

- ☐ Wiederverwendung bei Versionsentwicklung und Variantenentwicklung
 - Bereits gut erschlossen
 - Speziell, wenn Objektorientierung eingesetzt wird
- ☐ Intraproduktorientierte und interproduktorientierte Wiederverwendung
 - Findet bisher kaum statt
- ☐ Ebenen der Wiederverwendbarkeit
 - Produkt-Definition
 - Produkt-Entwurf
 - Produkt-Implementierung
 - Heute: Vorwiegend Wiederverwendung von Quellcode-Komponenten

Wiederverwendbarkeit & Wiederverwendung

- ☐ Differenzierung nach Anwendungsbereich
 - Vertikale Wiederverwendung
 - In der gleichen Anwendungsdomäne
 - Horizontale Wiederverwendung
 - In verschiedenen Anwendungsbereichen
 - Beispiel: Wissenschaftl. Unterprogramm-Bibliotheken
 - Beispiel: Bibliotheken mit Fundamentalklassen
- ☐ Art der Wiederverwendung
 - *white-box*-Wiederverwendung
 - Die Komponenten werden vom Wiederverwender modifiziert, adaptiert und neu getestet.
 - *black-box*-Wiederverwendung
 - Die Komponenten werden nicht geändert

Technik

- Wiederverwendung in Abhängigkeit vom Abstraktionsniveau
 - Funktionale Abstraktion
(Funktionen, Prozeduren, Makros)
 - Datenabstraktion
(abstrakte Datenobjekte und abstrakte Datentypen)
 - Klassen mit Vererbung und Polymorphismus
 - Zusätzlich können alle 3 Abstraktionen auch noch generisch sein
(außer abstrakte Datenobjekte)

Technik

- Funktionale Abstraktion
 - + Gut geeignet für einige ausgewählte Anwendungsbereiche wie
mathematische Bibliotheken
 - Abstraktionsniveau zu gering
 - Funktionaler Blickwinkel nicht allgemein genug
 - Trennung von Wert / Zustand und Bearbeitung
 - Parametermechanismus zu unflexibel
 - Generische, funktionale Abstraktion wird nur von wenigen
Programmiersprachen unterstützt

Technik

☐ Datenabstraktion

- Attribute und Operationen zu einer Einheit zusammengefasst (Kapselung und Geheimnisprinzip)
- Dadurch ein Nachteil der funktionalen Abstraktion aufgehoben
- + Gut geeignet für viele Anwendungsgebiete, insbesondere für fundamentale Datenstrukturen wie Keller, Warteschlange, Liste usw.
- + Durch Typparametrisierung hoher Allgemeingrad und gute Anpassbarkeit möglich
- + Leicht verständlich
- Nur anwendbar bei streng typisierten Sprachen
- Nicht so allgemein wie die Vererbung

Technik

☐ Datenabstraktion

- Trotz der Nachteile umfangreiche Wiederverwendbarkeitsbibliotheken
- Beispiel: Ada-Bibliothek von /Booch 87/
- Viele Klassenbibliotheken benutzen ebenfalls keine Vererbung, sondern verwenden nur die Möglichkeiten generischer Klassenkonzepte

☐ Objektorientierte Komponenten

- Die besten Möglichkeiten, eine hohe Wiederverwendbarkeit zu erreichen
- Es kommt jedoch darauf an, diese Möglichkeiten richtig zu nutzen
- Ein falscher Einsatz kann zum Gegenteil führen

Technik

- ☐ Objektorientierte Komponenten
 - + Vererbung erlaubt Spezialisierung durch *black-box*-Wiederverwendung, Originalklasse muss nicht geändert werden
 - + Der Anteil neuen Codes minimal, wenn zusätzliche Eigenschaften hinzugefügt werden
 - Die systeminhärente Komplexität wird zum großen Teil auf die Dynamik des Systems verlagert
 - Unübersichtliche Vererbungshierarchien, wenn die Vererbung über viele Ebenen geht
- ☐ Wiederverwendbarkeit größerer Einheiten
 - Rahmen (*frameworks*)
 - Analyse- und Entwurfsmuster (*patterns*)
 - Halbfabrikate (*componentware*)

Organisation

- ☐ Wiederverwendung
 - nicht automatisch durch Einsatz einer geeigneten Technik
 - muss organisiert werden
- ☐ Organisation muss umfassen
 - Aufbau, Einrichtung und Betrieb eines Wiederverwendbarkeitsarchivs
 - Evolutionäre Verbesserung der Wiederverwendung
 - Einbettung der Wiederverwendung in das Prozessmodell

Organisation

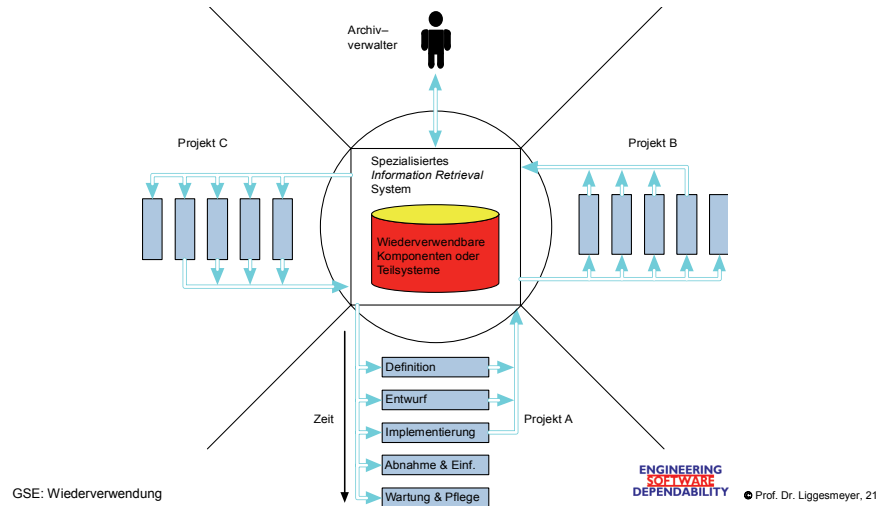
- Wiederverwendbarkeitsarchiv
 - Wiederverwendung scheitert heute oft daran, dass bereits vorhandene Komponenten nicht schnell, sicher und bequem wiedergefunden werden können
 - Beispiel
 - Software-Entwickler möchte Software wiederverwenden
 - Frage an Kollegen nach entsprechend entwickelter Software
 - Erinnerung an ein ähnliches Teilproblem, vor 2 Jahren in einem anderen Projekt
 - Hinweis auf bestimmte Datei in Bandsicherung des damaligen Projekts
 - Suche der Bandsicherung im Archiv

Organisation

- Nach längerem Suchen Auffinden einer Datei ähnlichen Namens
- Nur kommentierter Quellcode ohne weitere Dokumentation
- Bei Quellcodeübersetzung: damalige Compilerversion nicht mehr aufzutreiben
- Deprimiert wendet sich der Mitarbeiter einer eigenen Problemlösung zu; das Rad wird neu erfunden
- Anforderungen: Wiederverwendbarkeitsarchiv
 - Projektübergreifend
 - Leicht zugänglich
 - Jeder Mitarbeiter muss jederzeit von seinem Arbeitsplatz über Netz darauf zugreifen können

Organisation

□ Konzept eines Wiederverwendbarkeitearchivs



Organisation

- Wiederverwendbarkeits-Archiv kann sein
 - eigenständiges Archiv
 - mit oder ohne Anbindung an eine CASE-Umgebung
 - integriert in eine wiederverwendungsorientierte CASE-Umgebung
- Zum Wiederfinden muss man vorher konstruktiv etwas tun
 - Wiederverwendbare Komponenten und Teilsysteme müssen klassifiziert werden
- Klassifikation
 - Forschung: Viele Klassifikationssysteme für Ablage und Suche wiederverwendbarer Komponenten
 - Wichtiges Verfahren: Facettenklassifikation

Organisation

- Dienste einer Archivverwaltung
 - Aufbau der Klassifikationssysteme
 - Klassifizierung und Wiederauffinden der Komponenten und Teilsysteme
 - Export von Komponenten in andere Verwaltungssysteme
 - Import aus anderen Systemen
 - Berichte über die Klassifizierungssysteme und über die Komponenten mit ihren Klassifikationen
 - Nicht nur Verweise auf Quellen, sondern Quellen selbst speichern
 - Ist z.B. das OOA-Modell gefunden, dann muss das grafische OOA-Modell zugreifbar sein

Organisation

- Anforderungen an wiederverwendungs-orientierte CASE-Umgebungen
 - Unterstützung der Montage von Anwendungen aus Komponenten
 - *black-box*-Wiederverwendung
 - Verwaltung einer »nutzt«-Beziehung zwischen Komponente und nutzender Anwendung
 - Dadurch können Änderungen an der Originalkomponente der nutzenden Anwendungen mitgeteilt werden

Organisation

- *white-box*-Wiederverwendung
 - Jede nutzende Anwendung muss zusätzlich ihre eigene Version aus der Originalkomponente ableiten und weiterentwickeln können
 - Nutzende Anwendung benötigt Testumgebung der Komponente, wie Testrahmen, Testfälle, Testprotokolle
- Online-Verwaltung aller Ergebnisse aller Werkzeuge während des gesamten Entwicklungszyklus
 - Alle müssen mit ihren Beziehungen untereinander manipuliert werden können
- Konfigurations- und Änderungsverwaltung der wiederverwendbaren Komponenten

Organisation

- ☐ Reifegradstufen der Wiederverwendung
- Stufe 1: Ad hoc-Wiederverwendung
 - Wiederverwendung unsystematisch
 - Gelegentlich, in Abhängigkeit von der Arbeitsweise der Projektmitarbeiter, unkoordiniert und in der Regel undokumentiert
 - Regel: kopieren, anpassen und auseinanderlaufen lassen
 - Gemeinsame Versionsführung mit dem Original findet nicht statt
 - Wartung und Fehlerbehebung mehrfach
 - Großteil heutiger Wiederverwendung ist sicher in dieser Weise
 - Immer noch besser als überhaupt keine Wiederverwendung
 - Einfach und nur mit kurzfristigem Nutzen
 - Nutzen wird wieder aufgehoben, wenn sich zu viele nicht dokumentierte Kopien unkontrolliert verbreiten und die Software unwartbar machen

Organisation

Stufe 2: Wiederverwendung verfügbarer Software

- Durch Umsetzung geeigneter Maßnahmen systematisiert
- Grundlagen für Entwicklung wiederverwendbarer Komponenten werden geschaffen
- Verfügbare und wiederverwendbare Software muss gesammelt und strukturiert dokumentiert werden

Organisation

- Systemteile, die unterschiedlicher Änderungshäufigkeit unterliegen, werden lokalisiert
 - Varianten und Versionen entsprechender Systeme können effizienter entwickelt werden
- Verfahrensbasierter Zukauf von im Markt verfügbarer Software
 - Software-Entwicklung wird gezielt unterstützt
- Erstellung von modular aufgebauten Software-Systemen
 - Grundlage für die Wiederverwendung von Komponenten dieser Systeme
- Marktübliche und interne Standards werden eingehalten
 - portable Systeme

Organisation

Stufe 3: Entwicklung für Wiederverwendung

- Software-Entwicklung orientiert sich an Anwendungsdomänen
- Geeignete Infrastruktur für wiederverwendungsorientierte Entwicklung wird aufgebaut
- Software-Komponenten werden erstellt
 - Nicht im Hinblick auf ihren Einsatz in einem Produkt
 - Sondern auf ihre künftigen Verwendungsmöglichkeiten
- Software-Komponenten mit Wiederverwendungspotential werden als inkrementell erweiterbare Halbfabrikate erstellt

Organisation

- Funktionalität dieser Software kann ohne Eingriff in den Quellcode ausgebaut werden
- Inkrementell erweiterbare Software erleichtert Erstellung von Versionen und Varianten
- Erstellung von Halbfabrikaten kann projektunabhängig oder abgetrennt von anwendungsspezifischen Projektarbeiten geschehen
- Halbfabrikate auf der Grundlage einer Analyse der Anwendungsdomäne
 - Ziel: Domänenmodell, das eine generelle und damit wiederverwendbare Architektur für die Produkte der untersuchten Anwendungsdomäne darstellt

Organisation

- Weitere notwendige Aktivitäten
 - Einrichtung eines Wiederverwendbarkeits-Archivs
 - Durchführung einer wiederverwendungsspezifischen Aus- und Weiterbildung
 - Schaffung einer entsprechenden innerbetrieblichen Kommunikationsstruktur (*Bulletin Boards* usw.)
 - Bereitstellung von Richtlinien für standardisierte Auswahl, Bewertung und Anpassung von Komponenten

Organisation

Stufe 4: Verwendung von Domänenmodellen und statistische Steuerung des Prozesses

- Entwicklung von Anwendungen mit Domänenmodellen
- Steuerung der wiederverwendungsorientierten Software-Entwicklung auf statistischer Basis
- Für jede wohldefinierte Anwendungsdomäne wird ein Modell bereitgestellt
 - Das Modell soll eine generelle und damit wiederverwendbare Architektur für die Produkte der untersuchten Anwendungsdomäne beschreiben
- Analyse der jeweiligen Domäne erlaubt eine Integration in den Software-Entwicklungsprozess

Organisation

- Jedes Domänenmodell besteht aus
 - Architektur
 - Menge integrierter Komponenten, den Halbfabrikaten, aus eigenem und kommerziell verfügbarem Bestand
- Schwerpunkt der Software-Entwicklung
 - Modellierung von Anwendungen
 - Nicht Programmierung
- Anwendungen entstehen durch das Montieren von Halbfabrikaten
- Anwendungsentwickler entwickelt nur die anwendungsspezifischen Teile

Organisation

- Folgende Aktivitäten außerdem
 - Komponentenkreislauf für Entwicklung, Zertifizierung und Benutzung von Komponenten wird installiert
 - Wiederverwendbarkeitsorientierte CASE-Umgebungen werden eingerichtet
 - Projektverlauf wird durch Kennzahlen und Kosten/Nutzen-Modelle statistisch gesteuert
 - Kennzahlen werden aus Messungen ermittelt

Organisation

Stufe 5: Organisationsweite Ausrichtung auf Wiederverwendung

- Aktivitäten aller Unternehmensbereiche vollständig auf Wiederverwendung ausgerichtet
- Anwendungen werden wie in Stufe 4 durch Montage von bereits vorhandenen, weitgehend standardisierten Halbfabrikaten erstellt
- Neu: auch nicht softwareentwickelnde Unternehmensbereiche handeln wiederverwendungsorientiert
- Alle Verkaufsaktivitäten werden unter Berücksichtigung verfügbarer Komponenten ausgeübt.

Organisation

- Jede Software-Komponente wird als Vermögenswert des Unternehmens betrachtet
- Höhe der Vermögenswerte und Lebenszykluslänge dieser Komponenten finden Eingang in die
 - Entscheidungen der Software-Erstellung
 - strategischen Entscheidungen des Unternehmens

Organisation

□ Wiederverwendungsstufen

Stufe	Wiederverwendung in Prozent	Anforderungen
5 Bereichsbezogene Wiederverwendung	80 – 100%	Bereichsanalysen und -architekturen
4 Konsequente Wiederverwendung	50 – 70%	Bibliotheken, Prozesse, Metriken, Training
3 Planbare Wiederverwendung	30 – 40%	Zustimmung und Unterstützung vom Management, Motivationsprogramme, Bibliotheken
2 Ausschichten von Altanwendungen	10 – 50%	Glück und Wartungsprobleme
1 Keine oder ad hoc-Wiederverwendung	< 20%	Keine organisatorischen Änderungen

Organisation

□ 3-stufiges Reifegradmodell

- Praxiserfahrungen im Banken- und Versicherungsbereich

Stufe I: Wartbarkeit

- Wiederverwendung innerhalb eines einzelnen Projekts

Stufe II: Ausgewogenheit

- Wiederverwendung innerhalb ähnlicher Projekte

Stufe III: Standardisierung

- Grundlage für bereichsübergreifende Wiederverwendung

Organisation

- Wiederverwendungsorientiertes Prozessmodell
 - Notwendige Voraussetzung für Etablierung der Wiederverwendung
 - Alle Entwicklungsphasen:
 - Geeignete Komponenten werden gesucht
 - Neue wiederverwendbare Komponenten werden klassifiziert und im Archiv abgelegt
 - Wiederverwendbare Komponenten einsetzbar in
 - Definitionsphase
 - Entwurfsphase
 - Implementierungsphase
 - Intern erstellt oder extern zugekauft

Organisation

- Ständige Beobachtung des Softwaremarktes erforderlich
- Tätigkeiten beim Kauf von Komponenten
 - Auflistung der heutigen und zukünftigen Anforderungen an die eigenen Anwendungen
 - Prüfung
 - Komponenten ausreichend parametrisiert, um sie an wechselnde Erfordernisse anpassen zu können
 - Können die Komponenten mit CASE- und *Reengineering*-Werkzeugen evaluiert, angepasst und gewartet werden
 - Klärung der rechtlichen Aspekte der Nutzung der Komponenten

Management

- ☐ Wandel der Managementtätigkeiten
 - Software-Manager erhält die Aufgabe, ein Software-Produkt zu entwickeln
 - Heute
 - Kernfrage: »Wie löse ich das Problem mit meinen Ressourcen?«
 - Prognose
 - Kernfrage: »Wo hat bereits jemand ein ähnliches Problem gelöst, und wie bekomme ich die Problemlösung?«
 - Haupttätigkeit des Managements in Zukunft
 - Sicherstellung der Wiederverwendung von Komponenten

Management

- ☐ Wiederverwendbarkeit
 - stellt heute kein primär technisches Problem dar
 - ist ein Organisations- und Managementproblem
 - Wer sich heute dazu entscheidet, mit der Wiederverwendung von Software zu beginnen, kann morgen damit anfangen
- ☐ Wiederverwendbarkeitskultur
 - Aufgabe des Software-Managements
 - Bereitstellung einer geeigneten organisatorischen Umgebung
 - Etablierung einer Wiederverwendbarkeits-Kultur

Management

- Beispiel
 - Mitarbeiter entwickelt Software-Komponente im Rahmen eines Projekts
 - Während der Entwicklung hat er Ideen zur Verallgemeinerung der Komponente für den Einsatz in anderen Kontexten
 - Unter hohem Termindruck wird nur eine projektspezifische Lösung entwickelt
 - Szenario zeigt:
 - In fast allen Software-Unternehmen fehlt der Anreiz für Mitarbeiter, allgemeinere Lösungen zu entwickeln
 - Daher: Entwicklung von Belohnungssystemen für
 - Bereitstellung wiederverwendbarer Komponenten
 - Wiederverwendung dieser Komponenten

Management

- Beispiel
 - Nippon Novel zahlt jedem Software-Ingenieur 5 Cent pro Codezeile, wenn er eine Komponente in das Wiederverwendbarkeitsarchiv einbringt oder eine Komponente wiederverwendet
 - Der Entwickler einer wiederverwendbaren Komponente erhält für jede Wiederverwendung zusätzlich 1 Cent pro Codezeile
- Beantwortung folgender Fragen vom Management
 - Wer zahlt für die Entwicklung einer wiederverwendbaren Komponente?
 - Wer übernimmt die Wartungsverpflichtung?
 - Was gewinnt derjenige, der eine wiederverwendbare Komponente zur Verfügung stellt?

Kosten/Nutzen der Wiederverwendung

- ☐ Kosten/Nutzen der Wiederverwendung
- ☐ Qualitative Aussagen
 - Faustregel: »Formel 3«
 - Software muss *dreimal entwickelt* werden, bevor sie wirklich wiederverwendbar entwickelt werden kann
 - Bevor die »Früchte der Wiederverwendung geerntet« werden können, muss Software *dreimal wiederverwendet* werden
 - *break-even*-Punkt bei ca. 3 Wiederverwendungen einer gezielt für die Wiederverwendung entwickelten Komponente
 - Danach können die um 30% bis 50 % höheren Entwicklungskosten wieder hereingespielt werden

Kosten/Nutzen der Wiederverwendung

- 60% höhere Erstellungskosten:
 - 25% für zusätzliche Verallgemeinerung
 - z.B. Parametrisierung, Entwurf, *review*
 - 15% für zusätzliche Dokumentation
 - 10% für zusätzliches Testen
 - 5% für Archivablage und Wartung

Kosten/Nutzen der Wiederverwendung

- Wiederverwendung und Gesamtproduktivität

	0%	25%	50%
Gesamtzeit in MM	81,5	45	32
Anzahl der Mitarbeiter	8	6	5
Kosten je Zeile	40,74	22,5	16
Zeilen pro MM	165	263	370
Ersparnis	0%	45%	61%

- Es wurden 3 reale Anwendungen mit 10.000 Zeilen COBOL-Code untersucht.

Kosten/Nutzen der Wiederverwendung

- Wiederverwendung und Gesamtproduktivität

	0%	10%	30%	50%	80%
Gesamtzeit in MM	200	176	131	89	33
Ersparnis	0%	12%	34%	56%	84%

- Es wurden Objective-C-Programme mit 20.000 - 30.000 Zeilen untersucht

□ Kosten / Nutzen-Verhältnis bei Klassenbibliotheken

- Bibliotheksbestand wirkt sich gravierend auf die Kosten / Nutzen-Relation aus
- Sind am Anfang keine Bibliotheksklassen verfügbar, dann wird die Kosten / Nutzen-Relation erst für das 3. Projekt < 1

Kosten/Nutzen der Wiederverwendung

- Nutzen überwiegt erst nach Ende des 6. Projekts, wenn
 - Größe einer Bibliothek, beginnend mit 10 Klassen, kontinuierlich wächst, und
 - Entwicklungskosten für wiederverwendbare Klassen doppelt so hoch wie für projektspezifische Klassen sind
- Potentieller Wiederverwender hat höheren Aufwand
 - Er muss eine Komponente suchen, finden, verstehen, installieren, testen, parametrisieren und integrieren
 - Gewählte Komponente kann ungeeignet sein
- Aufwand kann sich lohnen, da Entwickler den Eigenentwicklungsaufwand oft stark unterschätzen

Kosten/Nutzen der Wiederverwendung

- Japanische Erfahrungen
 - Nicht primär eine Frage der Technik
 - Seit Ende der 80er Jahre hat Wiederverwendung eine hohe Priorität
 - Ergebnisse durch straffe Organisation und strikte Einhaltung von Methoden
 - NEC
 - 6,7 fache Produktivitätssteigerung
 - 2,8 fache Qualitätsverbesserung im Bereich betriebswirtschaftlicher Anwendungen
 - Erreicht durch
 - Identifikation und Standardisierung von 32 logischen Schablonen und 130 gemeinsamen Algorithmen
 - Strikte Anwendung ihrer CASE-Umgebung

Kosten/Nutzen der Wiederverwendung

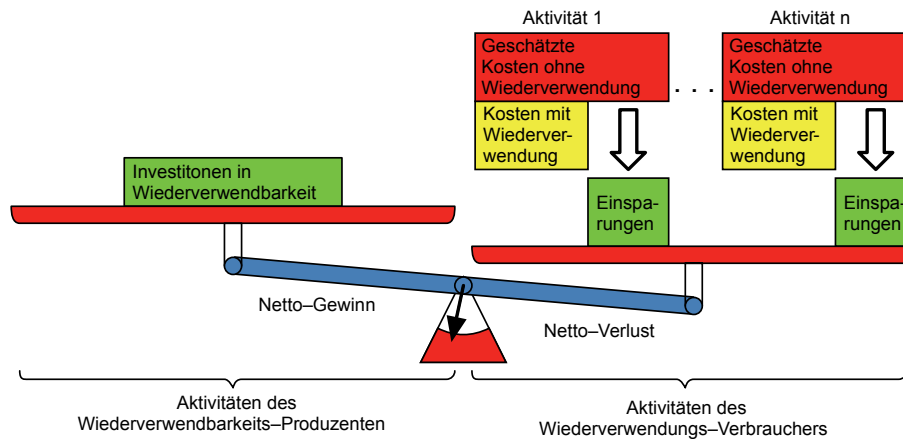
- Fujitsu
 - Nach Aufbau eines Wiederverwendbarkeitsarchivs Termintreue 70%
 - Vorher nur 20%
- Amerikanische Erfahrungen
 - Raytheon
 - Analyse und überarbeiteter Entwurf von 5000 im Einsatz befindlichen COBOL-Programmen
 - 50%-ige Produktionssteigerung in 6 Jahren
 - In neuen Systemen durchschnittlich 60% Code wiederverwendet
 - Unabhängige amerikanische Studie
 - > 70% kürzere Lieferzeiten durch Wiederverwendung

Kosten/Nutzen der Wiederverwendung

- Hewlett-Packard
 - Projekt 1
 - Fehlerreduktion: 51%
 - Produktivitätssteigerung: 57%
 - Projekt 2
 - Fehlerreduktion: 24%
 - Produktivitätssteigerung: 40%
 - Reduktion der Entwicklungszeit: 42%
- Quantitative Aussagen
 - Jede wiederverwendbare Software-Komponente muss als betrieblicher Vermögenswert (*asset*) angesehen werden
 - Dann kann Wiederverwendung wie jede andere finanzielle Investition kalkuliert werden

Kosten/Nutzen der Wiederverwendung

□ Kosten-/Nutzen-Relation der Wiederverwendung



GSE: Wiederverwendung

ENGINEERING
SOFTWARE
DEPENDABILITY ● Prof. Dr. Liggesmeyer, 53

Kosten/Nutzen der Wiederverwendung

- Eine Investition in die Wiederverwendung ist kosteneffektiv, wenn $K < N$
 - K = Gesamtkosten für die Investition
 - N = Einsparungen
- Nutzen N

$$N = \sum_{i=1}^k n_i = \sum_{i=1}^k (o_i - m_i)$$

- n_i : Einsparungen für die Aktivität i
- o_i : geschätzten Koste für die Aktivität i ohne Wiederverwendung
- m_i : Kosten für i mit Wiederverwendung
- k : Anzahl der Aktivitäten, die durch die entsprechende Investition tangiert sind

GSE: Wiederverwendung

ENGINEERING
SOFTWARE
DEPENDABILITY ● Prof. Dr. Liggesmeyer, 54

Kosten/Nutzen der Wiederverwendung

- Die Kosten/Nutzen-Relation, auch *Return-On-Investment*-Relation genannt, ergibt sich zu

$$R = \frac{N}{K}$$

- $R < 1$ ist bedeutet Verlust
- $R \gg 1$ bedeutet guten Gewinn
- 3 mögliche Strategien damit $R \gg 1$
 - Anteil der Wiederverwendung erhöhen
 - Durchschnittliche Kosten der Wiederverwendung reduzieren
 - Investitionskosten, die nötig sind, um einen Wiederverwendbarkeitsnutzen zu erzielen, reduzieren

Kosten/Nutzen der Wiederverwendung

- Metriken zum Messen der Wiederverwendung:
6 Kategorien
 - Metriken zur Kosten/Nutzen-Analyse
 - Metriken zur Einordnung in ein Reifegradmodell
 - Metriken zum Grad der Wiederverwendung
 - Metriken zur Berechnung der Wiederverwendungshindernisse
 - Metriken zur Schätzung der Wiederverwendbarkeit einer Komponente
 - Metriken zur Benutzung des Wiederverwendbarkeitsarchivs

Kosten/Nutzen der Wiederverwendung

- Metriken zur Einordnung in ein Reifegradmodell
 - Sind im strengen Sinne keine Metriken
 - Es handelt sich vielmehr um Merkmale, die einer Organisationseinheit helfen, den momentanen Reifegrad festzustellen
- Metriken zum Grad der Wiederverwendung
 - Werden benutzt, um den prozentualen Wiederverwendungsanteil zu schätzen und seine Entwicklung über die Zeit zu verfolgen:

$$\frac{\text{Anteil wiederverwendeter Komponenten}}{\text{Gesamtanzahl der Komponenten}}$$

Kosten/Nutzen der Wiederverwendung

- Metriken zur Berechnung der Wiederverwendungshindernisse
 - Ermitteln die Misserfolgsk Faktoren beim Versuch, Komponenten wiederzuverwenden
 - Misserfolgsk Faktoren sind
 - Anzahl Versuche, eine Komponente wiederzuverwenden
 - Komponente existiert nicht
 - Komponente ist nicht verfügbar
 - Komponente wurde nicht gefunden
 - Komponente wurde nicht verstanden
 - Komponente ist nicht gültig
 - Komponente kann nicht integriert werden

Kosten/Nutzen der Wiederverwendung

- Jedem Misserfolgsfaktor sind Misserfolgsursachen zugeordnet
 - Dem Faktor »Anzahl Versuche, eine Komponente wiederzuverwenden« sind z.B. folgende Ursachen zugeordnet
 - Ressourcenbeschränkungen
 - Kein Anreiz für die Wiederverwendung
 - Fehlende Ausbildung
- Verwendung einer solchen Metrik
 - Eine Organisation sammelt Daten über Misserfolgsfaktoren und -ursachen
 - Sie benutzt diese Informationen, um die Wiederverwendungsaktivitäten zu verbessern

Kosten/Nutzen der Wiederverwendung

- Metriken zur Schätzung der Wiederverwendbarkeit einer Komponente
 - Messen Attribute einer Komponente, die Indikatoren für ihre potentielle Wiederverwendbarkeit sind
 - Schwierigkeiten entstehen dadurch, dass manche Attribute vom Typ der wiederverwendbaren Komponente und der Programmiersprache abhängen

Kosten/Nutzen der Wiederverwendung

- Beispiel
 - Für eine hohe *black-box*-Wiederverwendbarkeit von Ada-Komponenten sind folgende Attribute gute Indikatoren
 - Weniger Aufrufe pro Quellcodezeile
 - Weniger Ein-/Ausgabe-Parameter pro Quellcodezeile
 - Weniger Lese-/Schreibanweisungen pro Zeile
 - Mehr Kommentare im Verhältnis zum Code
 - Mehr Hilfsfunktionen pro Quellcodezeile
 - Weniger Quellcodezeilen

Kosten/Nutzen der Wiederverwendung

- Metriken zur Benutzung des Wiederverwendbarkeitsarchivs
 - Güte des Klassifikationsschemas
 - Kosten
 - Sucheffektivität
 - Verständlichkeit
 - Güte der Komponenten
 - Anzahl der Wiederverwendungen innerhalb von 3 Monaten
 - Beurteilungen über wiederverwendete Komponenten
 - Benutzung des Wiederverwendbarkeitsarchivs
 - Systemverfügbarkeit
 - Anzahl der Benutzer
 - Ausgeführte Archivfunktionen
 - Anzahl verfügbarer Komponenten usw.

Einführung der Wiederverwendung

- Vorgehensweise
 - Einführung der Wiederverwendung
 - Typische Innovationseinführung mit allen damit verbundenen Merkmalen
 - Orientierung an einem Reifegradmodell der Wiederverwendung
 - 1. Aufgabe: Ermittlung der Ist-Stufe
 - Anschl. Planung, wie man schrittweise die jeweils nächste Stufe erreicht
 - Realistische und messbare Ziele definieren
 - Zeit um Wiederverwendungsgrad von 20% zu erreichen
 - Erfahrungsgemäß mindestens 1 Jahr

Einführung der Wiederverwendung

- Anforderungen an wiederverwendbare Komponenten dürfen anfangs nicht zu hoch sein
 - Dürfen in ein Archiv nur getestete, einsatzfertige Komponenten, dann trauen sich Entwickler nicht, ihre Software anzubieten
- Lösung
 - Archiv teilen in
 - geprüfte Komponenten
 - bereitgestellte, aber noch zu überprüfende und zu komplettierende Komponenten
- Einführung von Wiederverwendung verbunden mit Investitionen und Veränderungen im Unternehmen

Einführung der Wiederverwendung

- Dauerhafte Praktizierung der Wiederverwendung
 - muss in die Unternehmenskultur einfließen
 - muss täglich geübt werden
- Wandel der Entwicklungstätigkeit
 - Vom schreibenden Entwickler zum lesenden, evaluierenden und kreativ komponierenden Software-Architekten
- Entwickler
 - zeigen deutliche Ressentiments gegen Software, die sie nicht selbst entwickelt haben
 - befürchten schlechtere Qualität und fühlen sich in ihrem kreativen Selbstverständnis getroffen.

Einführung der Wiederverwendung

- ☐ Hindernisse bei der Einführung der Wiederverwendung
 - Ökonomisch
 - Fehlendes *Commitment*
 - Unklare Geschäftsstrategie
 - Investitionshöhe
 - Aufwandsgeschäft
 - Fehlende Nutzungs- und Verwertungsrechte
 - Organisatorisch
 - Im Prozess nicht vorgesehen
 - Verantwortung nicht zugewiesen
 - Fehlender Katalysator
 - Fehlende Infrastruktur

Einführung der Wiederverwendung

- Soziologisch
 - *Not-invented-here*-Syndrom
 - Widerstand gegen Veränderungen
 - Existenzängste »*Re-Use* ist ein Job-Killer«
 - Selbstverständnis des Entwicklers / geändertes Rollenbild
- Technisch
 - Fehlende Erfahrung mit praktischen Anwendungen
 - Mangelndes *Know-how*
 - Schwächen im *Software-Engineering*-Prozess
 - Fehlende Werkzeuge