

Kapitel 6 - Die Abnahme-, Einführungs-, Wartungs- und Pflegephase

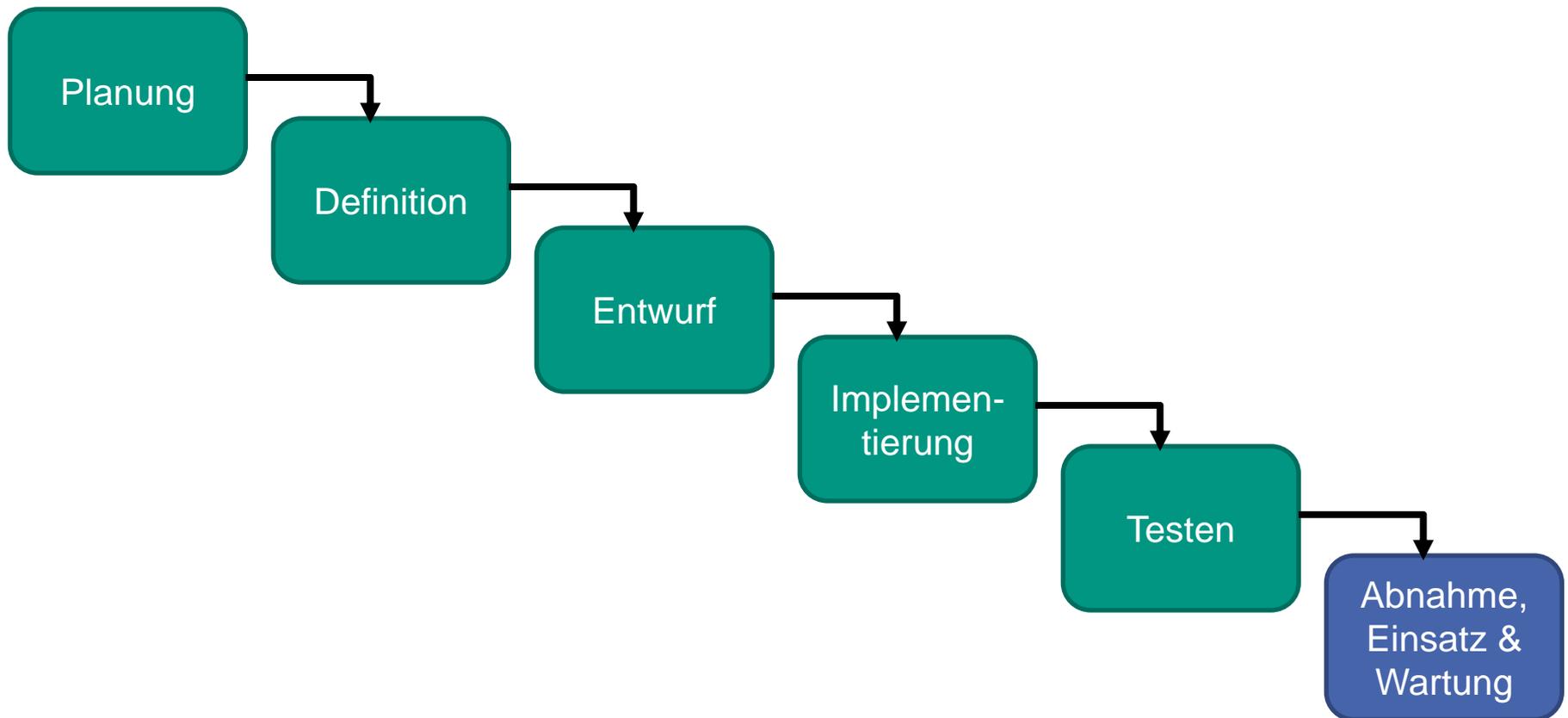
SWT I – Sommersemester 2010

Walter F. Tichy, Andreas Höfer, Korbinian Molitorisz

IPD Tichy, Fakultät für Informatik



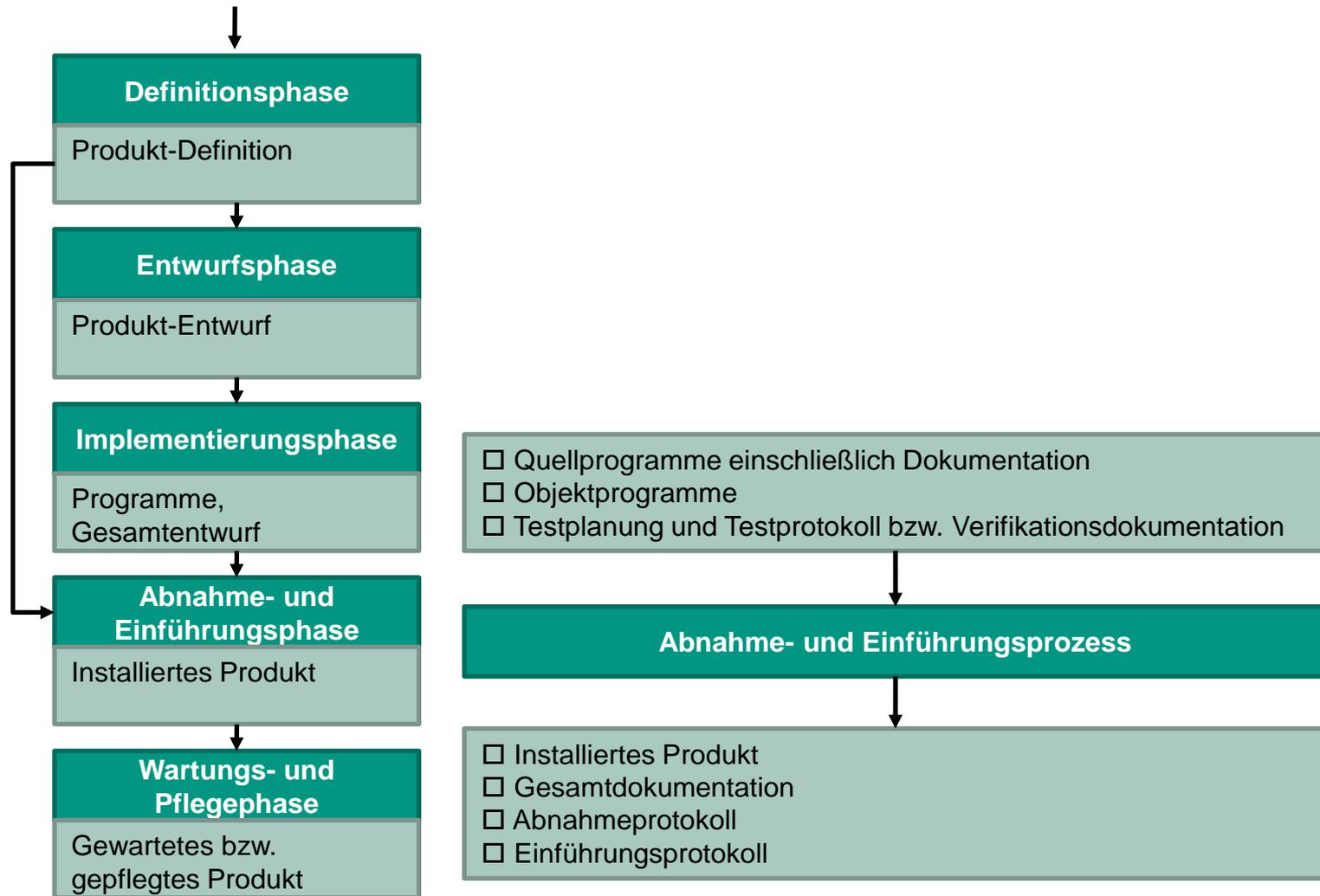
Wo sind wir gerade?



Inhalt

- Die Abnahmephase
- Die Einführungsphase
- Wartung & Pflege: Aufgaben und ihr Aufwand
- Wartung vs. Pflege
- Verbesserung der Pflege
- Verbesserung der Wartung

Die Abnahme-, Einführungs- & Wartungsphase



Die Abnahme-, Einführungs- & Wartungsphase

- Das **fertiggestellte** Gesamtprodukt wird abgenommen und beim Anwender eingeführt, d.h. in **Betrieb genommen**
- Ab diesem Zeitpunkt unterliegt das Produkt dann der **Wartung & Pflege**
- Eingebettet zwischen
 - Implementierungsphase und
 - Wartungs- & Pflegephase

Die Abnahmephase

- Die Abnahmephase: Tätigkeiten
 - Übergabe des Gesamtprodukts einschl. der gesamten **Dokumentation** an den Auftraggeber
 - Mit der Übernahme verbunden ist i. allg. ein **Abnahmetest**
 - Innerhalb einer Abnahme-Testserie ist es auch sinnvoll, **Belastungs-** oder **Stresstests** durchzuführen
 - Das Ergebnis der Abnahmephase ist ein Abnahmeprotokoll.

Die Abnahmephase

- Abnahme
 - Nach erfolgreichen Tests des Produkts durch den Auftraggeber
 - Die formale Abnahme ist die (schriftliche) Erklärung der Annahme (im juristischen Sinne) eines Produkts durch den Auftraggeber

Die Abnahmephase

- Externer Auftraggeber
 - Abnahmetest hängt auch davon ab, ob der Auftraggeber das Produkt...
 - nur nutzt, aber nicht wartet und pflegt
 - nutzt und selbst wartet und pflegt
 - Welche Alternative der Auftraggeber wählt, sollte bereits bei der Auftragsvergabe bekannt sein
 - Die für den Auftraggeber relevanten Qualitätsmerkmale hängen von der gewählten Alternative ab.

Die Abnahmephase

- **Produktnutzung**
 - Wesentliche Qualitätsmerkmale:
 - Nutzbarkeit
 - Integrität
 - Effizienz
 - Korrektheit und
 - Zuverlässigkeit

- **Wartung & Pflege**
 - Zusätzliche Merkmale:
 - Wartbarkeit
 - Testbarkeit
 - Flexibilität

Die Abnahmephase

- Abnahmetest
 - Erfüllung der Qualitätsmerkmale prüfen
 - Macht Auftraggeber Wartung & Pflege selbst, dann benötigt er...
 - die **gesamte** Entwurfs- & Implementierungsdokumentation
 - eine **sorgfältige** Einführung in die Architektur
 - Die **gesamten** Testeinrichtungen und Testfälle

Die Einführungsphase

- Tätigkeiten
 - Installation des Produkts
 - Einrichtung des Produkts in dessen Zielumgebung zum Zwecke des Betriebs
 - Schulung der Benutzer und des Betriebspersonals
 - Nach der Installation des Produkts sind die Benutzer in die Handhabung des Produkts einzuweisen
 - Inbetriebnahme des Produkts
 - Übergang zwischen Installation und Betrieb.

Die Einführungsphase

- Einführungsprotokoll
 - Alle Vorkommnisse, die in der Einführungsphase auftreten, werden festgehalten
- Einführung muss sorgfältig geplant werden
- Umfangreiche Produkteinführungen wie Innovationseinführungen zu behandeln

Die Einführungsphase

- Bei Ersatz eines existierenden Systems: Umstellung
 - Zeitplanung, u.U. mit Netzplänen
 - Wichtige Aufgabe:
 - **Umstellung** der Datenbestände
 - Manuelle Karteien und Aktenbestände müssen oft erst **aufbereitet** oder zusammengestellt werden, bevor sie für die neue Datenverwaltung erfasst werden können
 - Bei umfangreichen Beständen
 - Für die manuelle Datenerfassung muss **Zeit** eingeplant werden

Die Einführungsphase

■ Umstellung

- Das größte Problem ergibt sich bei der Übertragung „lebender“ Datenbestände z.B. Lagerdateien, Kundendaten
 - Hier muss zu einem bestimmten Zeitpunkt oder zu mehreren Zeitpunkten umgestellt werden
- Es ist zu berücksichtigen, dass zur Erstellung neuer Bestände zum Teil eigene Konversions-Programme erforderlich sind, die entwickelt werden müssen
- Außerdem ist zu überlegen, wie die Richtigkeit der erstellten Datenbestände überprüft werden kann.

Die Einführungsphase

- Inbetriebnahme auf 3 Arten möglich:
 - direkte Umstellung
 - Parallellauf
 - Versuchslauf

Die Einführungsphase

■ Direkte Umstellung

- Es wird unmittelbar von dem **alten auf das neue** System übergegangen
- Die Benutzung des alten Systems wird gestoppt, um das neue System sofort in Betrieb zu nehmen
- Für die Umstellungsarbeiten wird ein Wochenende oder eine Feiertagsperiode gewählt
- Die direkte Umstellung ohne weitere Vorkehrungen ist **risikoreich**

Die Einführungsphase

■ Parallellauf

- Die Bewegungsdaten werden sowohl im alten als auch im neuen System verarbeitet, so dass die Ergebnisse miteinander verglichen werden können
- Vorteil:
 - Man hat **Sicherheit**, falls das neue System nicht funktioniert
- Nachteile:
 - Hohe **Kosten**
 - Schwierigkeiten, die durch den Parallellauf zweier Systeme entstehen.

Die Einführungsphase

■ Versuchslauf

■ 1. Möglichkeit

- Neues System arbeitet mit **Daten aus vergangenen Perioden**, so dass die Ergebnisse bekannt sind und überprüft werden können
- In der Zeit der Versuchsläufe meldet der Benutzer Beanstandungen und Versagen
- Aktuelle Verarbeitung erfolgt im alten System

Die Einführungsphase

- 2. Möglichkeit
 - Einführung des neuen Systems **in einzelnen Stufen**, indem verschiedene Funktionsbereiche sukzessiv übernommen werden
 - Dies erleichtert auch die Versetzung und Schulung von Personal.

Die Einführungsphase

- Pilotinstallation oder Betatest
 - Wird ein Softwareprodukt für den **anonymen Markt** hergestellt dann erfolgen vor einer allgemeinen Vertriebsfreigabe eine Reihe von Pilotinstallationen bei Pilotkunden (Betatest).

Die Einführungsphase

- Ende der Produktentwicklung
 - Nach erfolgreicher Einführung des Produktes erfolgt die offizielle **Freigabe** des Produktes
 - Damit ist die **Produktentwicklung beendet**
- Abnahme- & Einführungsphase liefert folgende Ergebnisse:
 - Gesamtprodukt einschl. Gesamtdokumentation
 - Abnahmeprotokoll
 - Einführungsprotokoll.

Die Einführungsphase

- Als Grundlage für die spätere Wartung ist es nötig, alle Produkte zu archivieren
 - Das Wartungsarchiv muss von jedem Produkt verschiedene Versionen aufbewahren.
 - Das Wartungsarchiv muss Informationen über die installierten Versionen bei den einzelnen Kunden aufnehmen
 - Oder die Software identifiziert sich selbst.
 - Soll automatische Aktualisierung der Software erfolgen?

Die Wartungs- & Pflegephase

- **Wartung und Pflege**
 - Beginnt mit der **erfolgreichen** Abnahme und Einführung eines Software-Produkts

Die Wartungs- & Pflegephase

- Nach der Inbetriebnahme eines Produktes...
 - treten im täglichen Betrieb Versagen auf
 - ändern sich die Umweltbedingungen
 - neue Systemsoftware
 - neue Hardware
 - neue organisatorische Einbettung
 - entstehen neue Wünsche und Anforderungen
 - neue Funktionen
 - geänderte Benutzungsoberfläche
 - erhöhte Geschwindigkeit.

Die Wartungs- & Pflegephase

■ Alterung von Software

- Software, bei der nicht ständig Defekte behoben und Anpassungen sowohl an die Umwelt als auch an neue Anforderungen vorgenommen werden, **altert** und ist irgendwann veraltet
- Sie kann dann nicht mehr für den ursprünglich vorgesehenen Zweck eingesetzt werden
- „Software veraltet in dem Maße, wie sie mit der Wirklichkeit nicht Schritt hält“ /Sneed 83/.

Wartung & Pflege: Aufgaben und ihr Aufwand

- 4 Kategorien der Wartungs- & Pflege
 - **korrektive** Tätigkeiten (Wartung)
 - Stabilisierung / Korrektur
 - Optimierung / Leistungsverbesserung
 - **progressive** Tätigkeiten (Pflege)
 - Anpassung / Änderung
 - Erweiterung.

Wartung & Pflege: Aufgaben und ihr Aufwand

■ Stabilisierung / Korrektur

- Alle Tätigkeiten, die dazu dienen, Defekte zu beheben
- Es kann sich dabei um Defekte handeln, die bereits bei der Entwicklung in das Produkt gelangt sind, oder um solche, die bei der Wartung neu entstehen.

Wartung & Pflege: Aufgaben und ihr Aufwand

■ Stabilisierung / Korrektur

- Software-Produkte werden mit durchschnittlich 0,75% Defekten pro 100 Anweisungen freigegeben
 - 10.000 Zeilen = 75 Defekte
 - Nur ein Teil dieser Defekte wird vor der Inbetriebnahme entdeckt
 - Die meisten davon werden erst im Betrieb festgestellt
 - Die Defektbeseitigung verursacht erhebliche Kosten
 - Lokalisierung und Behebung dieser Restdefekte:
 - Wartung im engeren Sinne, obwohl eigentlich eine Restarbeit der Entwicklung.

Wartung & Pflege: Aufgaben und ihr Aufwand

■ Stabilisierung / Korrektur

- Besonders schnell vermehren sich **Wartungsdefekte**, die sogenannten *Second Level Defects*
 - Defekte, die bei Defektbehebung eingeführt wurden
 - Sie können bald die Mehrzahl der Defekte ausmachen
 - Ursache:
 - Schlechte Konstruktion und Fehleranfälligkeit des ursprünglichen Produkts
 - Mangelhafte Dokumentation
 - Mangelndes Produktverständnis bei Wartungspersonal

Wartung & Pflege: Aufgaben und ihr Aufwand

■ Beispiel

- Formales Produktmodell weder erstellt noch auf Vollständigkeit, Konsistenz und Eindeutigkeit überprüft
- Sonderfälle werden übersehen und nicht implementiert
- Freigegebenes Produkt „läuft“ solange, wie Sonderfälle nicht auftreten
- „Stürzt“ das Produkt beim 1. Sonderfall ab, dann wird die Implementierung ergänzt um einen „Rucksack“ der Art „if Sonderfall then ...“ usw.
- Programm wird daher immer unübersichtlicher und schlechter wartbar
- An unerwarteten Stellen entstehen plötzlich Folgefehler.

Wartung & Pflege: Aufgaben und ihr Aufwand

■ Faustregeln

- Auf 10 Defekte, die vor der Produktfreigabe durch Testen gefunden werden, entfällt 1 Defekt, der nach der Freigabe gefunden wird
- Es dauert die 4 bis 10-fache Zeit, um in einem umfangreichen, im Einsatz befindlichen Software-Produkt einen Defekt zu finden und zu beheben, als in einem Produkt vor oder kurz nach der Freigabe.

Wartung & Pflege: Aufgaben und ihr Aufwand

- Optimierung/Leistungsverbesserung
 - Frisch eingesetzte Software verbraucht oft mehr Zeit und Speicher, als zur Erfüllung ihrer Aufgaben zur Verfügung steht.
 - Optimierung erfolgt selten vor der ersten Freigabe
 - Sobald ein Produkt funktionsfähig ist, wird es freigegeben
 - Optimierung bleibt der Wartung vorbehalten:
 - Alle Aktivitäten um Leistung zu verbessern
 - Feinoptimierung und Reduzierung des Speicherbedarfs
 - Zum Teil sind auch Restrukturierungen erforderlich, um die Leistungsverbesserungen zu erreichen.

Wartung & Pflege: Aufgaben und ihr Aufwand

- Anpassung/Änderung
 - Anpassungen werden durch Wandlungen in der Umwelt erzwungen:
 - Änderungen in der technischen Umgebung
 - z.B. neue Systemsoftware, neue Geräte
 - Änderungen in den Benutzungsoberflächen
 - z.B. modifizierte Fenster oder Formulare, Spracheingabe
 - Änderungen in den Funktionen
 - z.B. Gesetzesänderungen, neue betriebliche Regelungen.

Wartung & Pflege: Aufgaben und ihr Aufwand

■ Erweiterungen

- Führen zu einer funktionalen Ergänzung des Produkts
- Funktionen, die bei der Erstentwicklung vorgesehen oder geplant, aber nicht implementiert wurden, werden eingebaut
- Oder es ergeben sich neue Funktionen aus den Erfordernissen des Betriebs der Software.

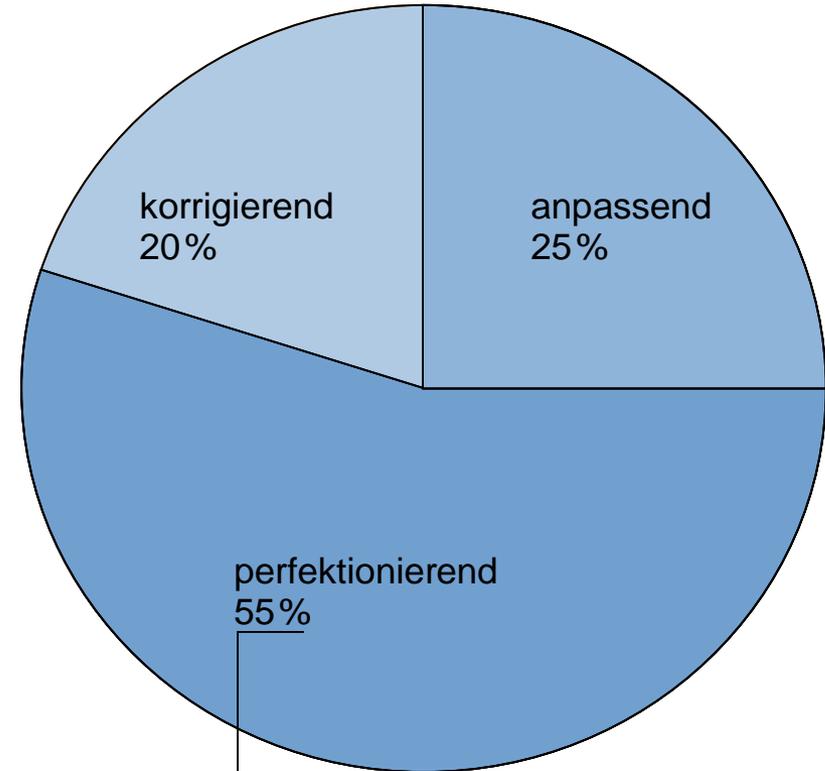
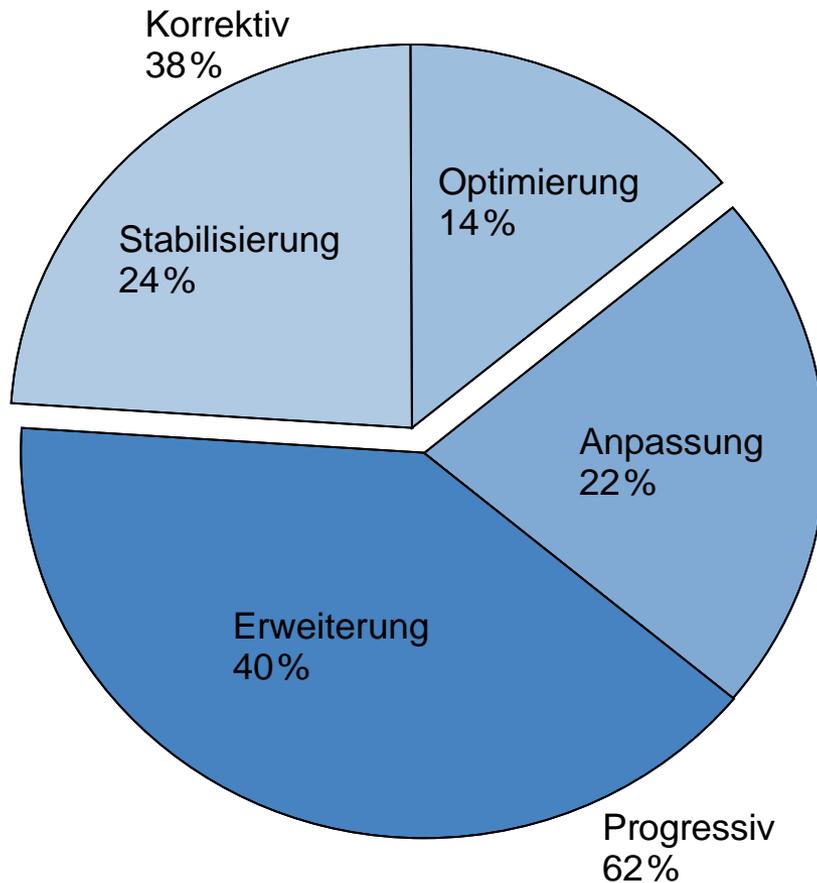
Wartung & Pflege: Aufgaben und ihr Aufwand

- Andere Klassifikation:
- Korrigierende Aktivitäten
 - Umfassen das Identifizieren und Korrigieren von
 - Software-Defekten
 - Leistungsmängeln
 - Implementierungsdefekten
 - Dazu gehören auch „Notfall-Reparaturen“, die sofort ausgeführt werden müssen, um den laufenden Betrieb aufrecht zu erhalten
 - Auch die Korrektur der Implementierung gehört zu diesen Aktivitäten, um sie den spezifizierten Produkt-Anforderungen und Leistungen anzugleichen.

Wartung & Pflege: Aufgaben und ihr Aufwand

- **Anpassende Aktivitäten**
 - Dienen dazu, die Software an die sich ändernde Produktumgebungen anzugleichen
- **Perfektionierende Aktivitäten**
 - Erhöhen die Leistung, verbessern die Kosteneffektivität, Verarbeitungseffektivität und Wartbarkeit
 - Dazu gehören auch Erweiterungen aufgrund von neuen Benutzeranforderungen
- **Der meiste Aufwand entfällt auf Anpassungen und Erweiterungen**
 - 62% bis 80%.

Wartung & Pflege: Aufgaben und ihr Aufwand



Erweiterungen für Benutzer ..	42%
Dokumentation	6%
Effizienz	4%
Sonstiges	3%

Wartung vs. Pflege

■ Wartung:

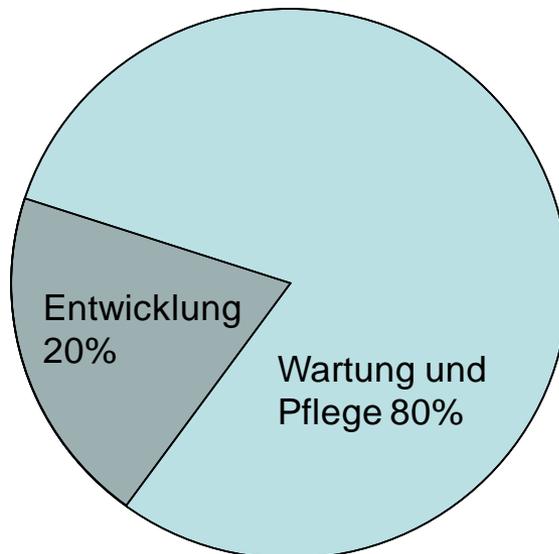
- Lokalisierung und Behebung von Defekten von in Betrieb befindlichen Software-Produkten, wenn das Versagen bekannt ist.
- Ist ereignisgesteuert, daher schwer planbar.

■ Pflege:

- Lokalisierung und Durchführung von Anpassungen, Änderungen und Erweiterungen von in Betrieb befindlichen Software, wenn die Art der gewünschten Änderungen/Erweiterungen feststeht.
- Ist planbar.

Wartung und Pflege vs. Entwicklung

- Maintenance
 - USA: Subsummierung von Wartung und Pflege
- Lebenszyklus (engl. *life cycle*) eines Produkts
 - Aufwand für ein Produkt lässt sich aufteilen in
 - den **Entwicklungsaufwand**
 - den **Wartungs- & Pflegeaufwand**



Wartung vs. Pflege

■ Faustregeln

- Der Aufwand für die Wartung & Pflege ist normalerweise **größer** als der Entwicklungsaufwand
- Der Aufwand für die Wartung & Pflege ist typischerweise um einen **Faktor von 2 bis 4 größer** als der Entwicklungsaufwand für ein umfangreiches Produkt.
- Eine solche Aufwandsverteilung bedeutet
 - Im Extremfall sind von 100 Mitarbeitern einer Software-Abteilung 80 Mitarbeiter mit der Wartung & Pflege „alter“ Software beschäftigt
 - Nur 20 Mitarbeiter entwickeln neue Software.

Verbesserung der Pflege

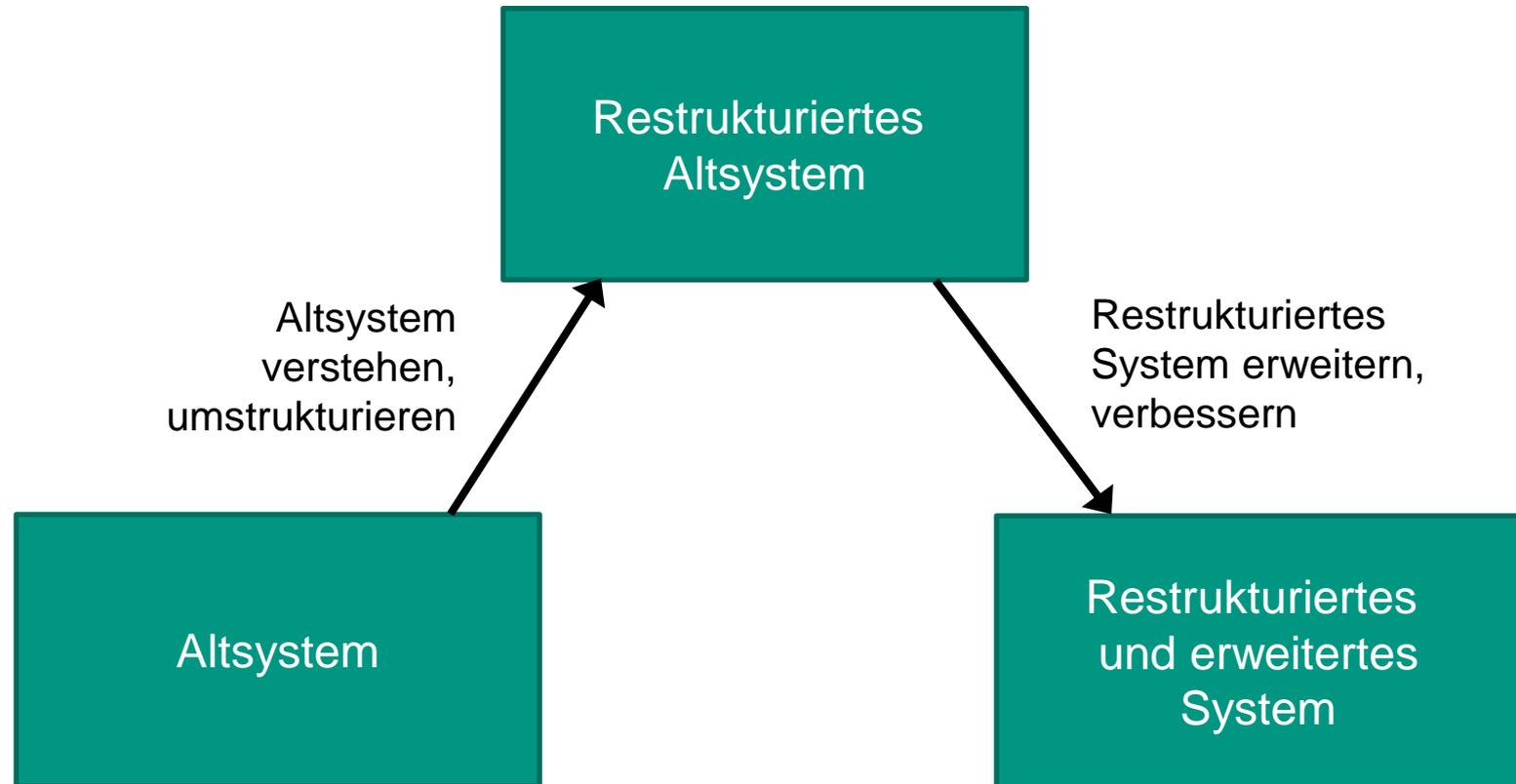
- Pflege = Weiterentwicklung
 - Anpassungen und Erweiterungen eines Produkts sind charakteristisch für **Weiterentwicklungen** bzw. für neue Versionen von Produkten
 - Es ist daher sinnvoll – abgesehen von minimalen Änderungen – alle Pflegeaktivitäten den normalen Software-Entwicklungsprozess durchlaufen zu lassen
 - Im **evolutionären** und **inkrementellen** Prozessmodell
 - Keine Pflegephase mehr, sondern Pflegeaktivitäten werden als Erstellung einer neuen **Produktversion** angesehen.

Verbesserung der Pflege

- Wartungs- & Pflegeaufwand wächst sowohl mit
 - dem Alter als auch mit
 - dem Umfang des Software-Produkts
- Umfang wächst um ca. 10% pro Jahr
 - Bereitstellung zusätzlicher Merkmale und Funktionen trägt vor allem zu diesem Zuwachs bei
 - Ältere Produkte tendieren dazu, umfangreicher und schwerer wartbar zu sein
 - Ab einem bestimmten Zeitpunkt folgende Fragen:
 - Soll weiter gewartet und gepflegt werden?
 - Soll das Produkt saniert werden?
 - Soll das Produkt durch ein neues ersetzt werden?

Software-Sanierung

- Sanierung oder Restaurierung (engl. *Re-Engineering*) von Alt-Systemen
 - Es wird immer alte Software geben
 - alte Software ist der überwiegende Teil der installierten Software
 - Bei der Sanierung erfolgt ein Verstehen der alten Software und ihre Umwandlung in eine besser wartbare Form (neuere Sprache, klarer Entwurf, modularisierte Struktur) vor der eigentlichen Änderung
 - Die Entwicklung von Hilfsmitteln, die Softwaresanierung erleichtern, ist ein wichtiges Forschungsgebiet



Änderungsverwaltung

- Erfassung und Verwaltung eingehender **Fehlermeldungen** und **Verbesserungsvorschlägen**
- Entscheidung über die Bearbeitung von Änderungsanträgen/Fehlermeldungen
 - Ablehnung/Annahme
 - Auswahl eines Lösungsvorschlags
 - Berücksichtigung der technischen und zeitlichen Auswirkungen
 - Veranlassungen der Bearbeitung.
 - Bündelung von Änderungen.

Änderungsverwaltung

- Damit Defekte behoben werden können, muss das vom Benutzer gemeldete Versagen **reproduzierbar** sein.
- Alle zur Defektbehebung relevanten Informationen werden in einer **Fehlermeldung** (engl. *bug report, problem report*) durch den Benutzer zusammengefasst.
- Relevante Informationen sind unter anderem die Eingaben, die zum Ausfall führten, die Produktversion, das verwendete Betriebssystem, etc.

Änderungsverwaltung

- Damit mehrere Benutzer (der Software) gleichzeitig Fehler melden und mehrere Entwickler die gemeldeten Fehler/Änderungswünsche bearbeiten können, bietet sich der Einsatz einer **Änderungsverfolgung** (engl. *bug tracker*) an:

Kommerziell

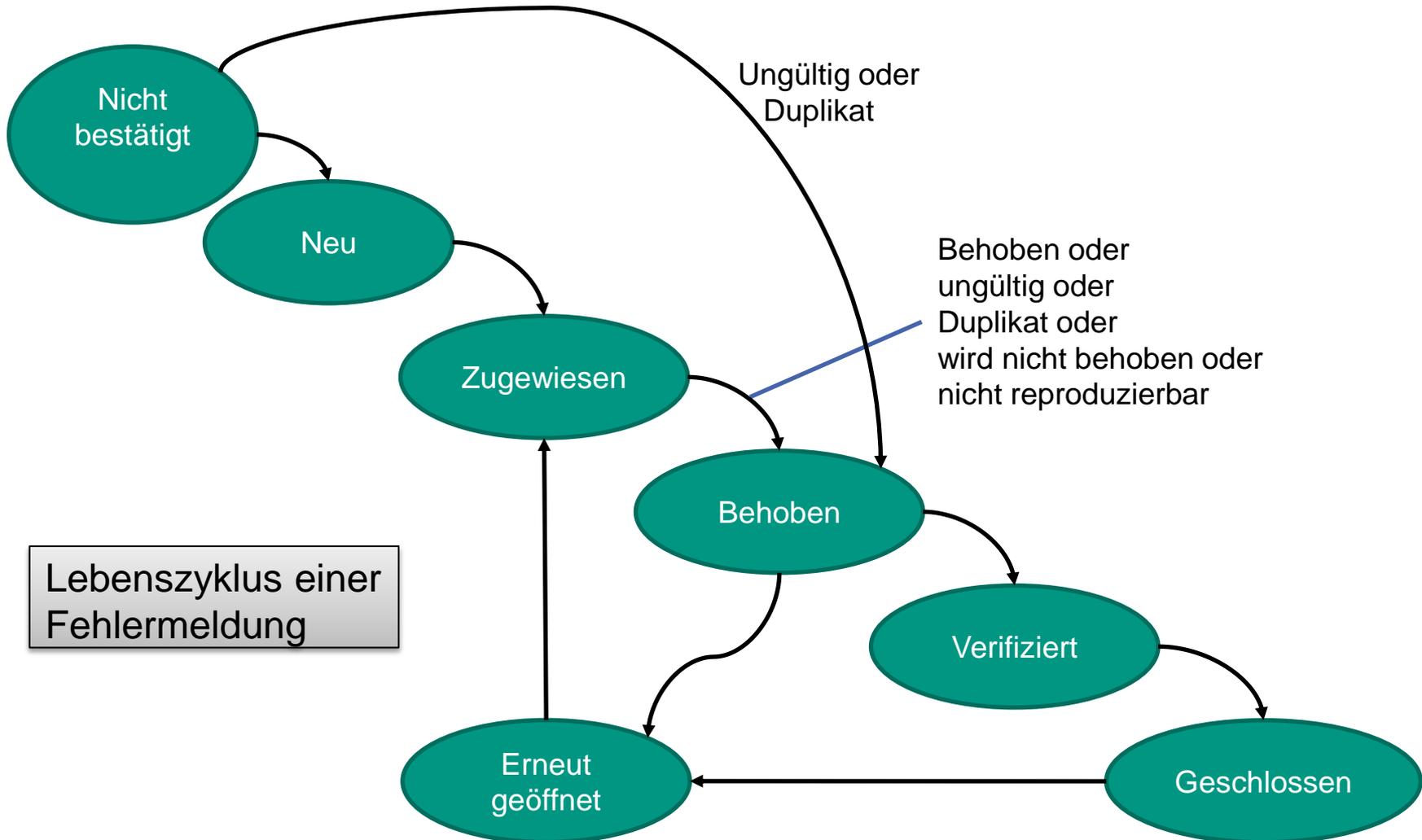
- BugZilla: <http://www.bugzilla.org/>
- JIRA: <http://www.atlassian.com/software/jira/>
- Trac: <http://trac.edgewall.org/>
- ...

Änderungsverfolgung

■ BugZilla:

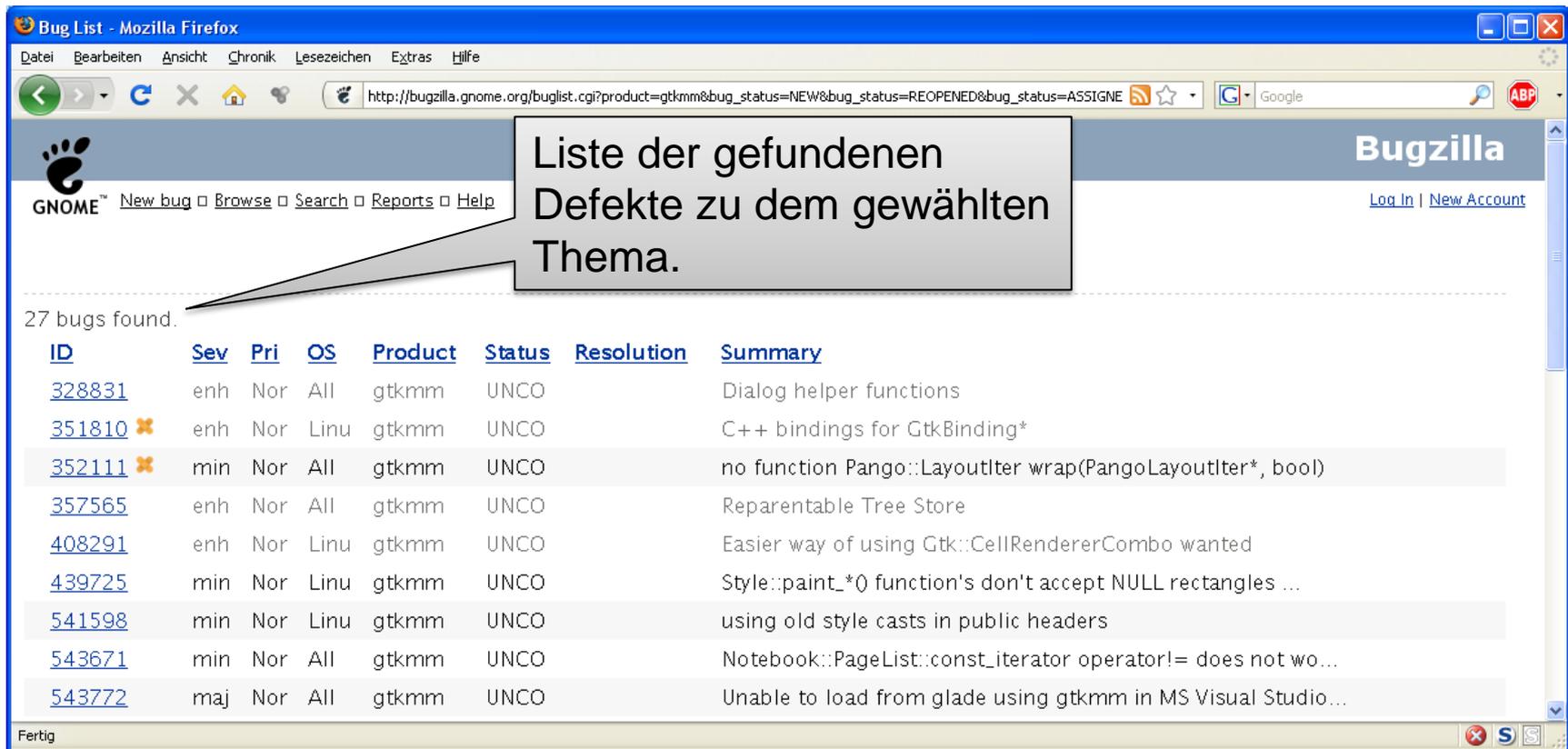
- Eine Meldung in BugZilla besteht aus:
 - Urheber
 - Beschreibung des Problems oder Änderungswunsches
 - Schweregrad
 - Priorität
 - Status (7 Zustände)
 - nicht bestätigt
 - neu
 - (einem Entwickler) zugewiesen
 - behoben
 - Behebung wurde verifiziert
 - geschlossen
 - erneut geöffnet
 - Produkt(-Version), Komponente
 - Betriebssystem

Änderungsverfolgung



Änderungsverfolgung

■ Bugzilla im Einsatz:



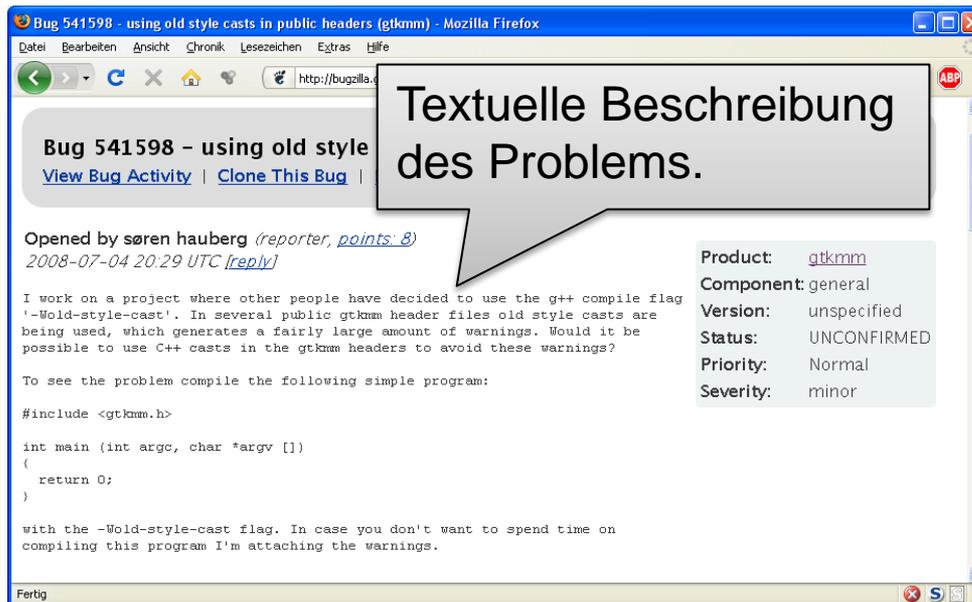
Liste der gefundenen Defekte zu dem gewählten Thema.

27 bugs found.

<u>ID</u>	<u>Sev</u>	<u>Pri</u>	<u>OS</u>	<u>Product</u>	<u>Status</u>	<u>Resolution</u>	<u>Summary</u>
328831	enh	Nor	All	gtkmm	UNCO		Dialog helper functions
351810 ✘	enh	Nor	Linu	gtkmm	UNCO		C++ bindings for GtkBinding*
352111 ✘	min	Nor	All	gtkmm	UNCO		no function Pango::LayoutIter wrap(PangoLayoutIter*, bool)
357565	enh	Nor	All	gtkmm	UNCO		Reparentable Tree Store
408291	enh	Nor	Linu	gtkmm	UNCO		Easier way of using Gtk::CellRendererCombo wanted
439725	min	Nor	Linu	gtkmm	UNCO		Style::paint_*() function's don't accept NULL rectangles ...
541598	min	Nor	Linu	gtkmm	UNCO		using old style casts in public headers
543671	min	Nor	All	gtkmm	UNCO		Notebook::PageList::const_iterator operator!= does not wo...
543772	maj	Nor	All	gtkmm	UNCO		Unable to load from glade using gtkmm in MS Visual Studio...

Änderungsverfolgung

■ Bugzilla im Einsatz:



Bug 541598 - using old style casts in public headers (gtkmm) - Mozilla Firefox

[View Bug Activity](#) | [Clone This Bug](#)

Opened by [søren hauberg](#) (reporter, [points: 8](#))
 2008-07-04 20:29 UTC [\[reply\]](#)

I work on a project where other people have decided to use the g++ compile flag '-Wold-style-cast'. In several public gtkmm header files old style casts are being used, which generates a fairly large amount of warnings. Would it be possible to use C++ casts in the gtkmm headers to avoid these warnings?

To see the problem compile the following simple program:

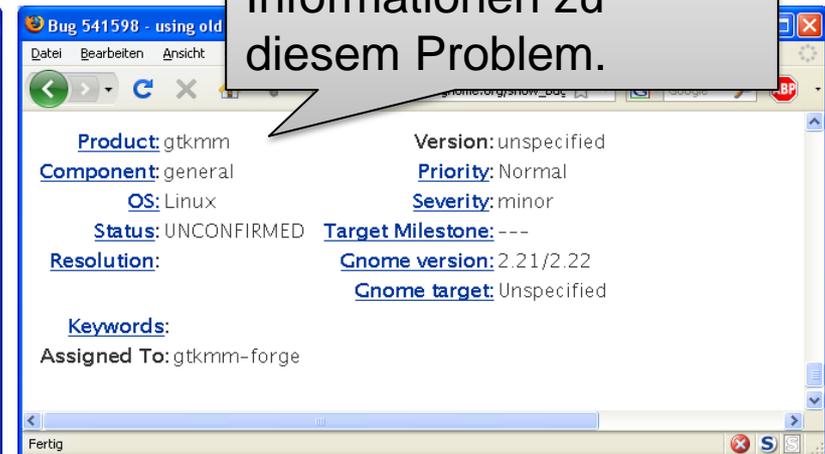
```
#include <gtkmm.h>

int main (int argc, char *argv [])
{
    return 0;
}
```

with the `-Wold-style-cast` flag. In case you don't want to spend time on compiling this program I'm attaching the warnings.

Product: [gtkmm](#)
Component: general
Version: unspecified
Status: UNCONFIRMED
Priority: Normal
Severity: minor

Textuelle Beschreibung des Problems.



Product: [gtkmm](#) **Version:** unspecified
Component: [general](#) **Priority:** [Normal](#)
OS: [Linux](#) **Severity:** [minor](#)
Status: [UNCONFIRMED](#) **Target Milestone:** [---](#)
Resolution: **Gnome version:** [2.21/2.22](#)
 Gnome target: [Unspecified](#)

Keywords:
 Assigned To: [gtkmm-forge](#)

Weitere relevanten Informationen zu diesem Problem.

Änderungsverwaltung

- Es bieten sich zusätzlich auch automatisierte Fehlermeldungen nach einem Programmabsturz an, wie es bspw. bei Programmen von Mozilla ist (<http://talkback-public.mozilla.org>)
- Dabei ergeben sich allerdings oft Bedenken hinsichtlich der gesendeten Daten (Stichwort: Datenschutz/Privatsphäre)

Organisation der Wartung

- Sollte Wartung eigenständig sein?
- Oder sollten Entwickler auch warten?

Vorteile eigenständiger Wartungsorganisation

- Klare Zuordnung der Wartungs- und Entwicklungskosten
- Entlastung der Entwickler von Wartungsaufgaben und insbesondere von paralleler Durchführung unterschiedlicher Tätigkeiten
- Qualitativ besserer Abnahmetest durch das Wartungsteam
- Besserer Kundenservice durch **Konzentration** auf die **Wartung**
- Einstellung **spezialisierter Mitarbeiter** bzw. gezielte Ausbildung der Mitarbeiter
- **Effizientere Kommunikation** zwischen den Wartungsmitarbeitern
- Höhere **Produktivität** durch Spezialisierung und zusammenhängende Produktkenntnisse.

Nachteile eigenständiger Wartungsorganisation

- Wartungsarbeiten können ein „schlechtes Image“ bekommen, wodurch die Motivation der Mitarbeiter sinkt
- Beim Übergang von der Entwicklung zur Wartung geht wertvolles Wissen über das Produkt verloren
- Koordinationsprobleme zwischen Entwicklung und Wartung, insbesondere wenn neue Produkte alte ersetzen.
- Die Entwickler müssen **nicht** die **Konsequenzen** ihrer Entwicklung tragen
- Die Wartungsmitarbeiter müssen sich aufwendig in die Systeme einarbeiten
- Eine **gleichmäßige Auslastung** der Mitarbeiter ist schwierig zu erreichen.

Organisation der Wartung

- Es gibt keine perfekte Organisation
- Kompromiss:
 - Getrennte Organisationen
 - Die Mitarbeiter „rotieren“ aber zwischen beiden Organisationseinheiten
 - Dann sehen Entwickler, was in der Wartung wichtig ist und umgekehrt.
 - Man bleibt nicht für immer in der Wartung „hängen“.
 - Der Erfolg der Wartung hängt weniger von der Softwaretechnik, sondern vor allem von der Organisation und dem Management ab.