SOFTWARE CONSTRUCTION

RWTH AACHEN
RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

Diplomarbeit

Analyse von Softwareentwicklungs- und Wartungsprozessen auf Basis von Change Request Daten

– Analysis of Software Development and Maintenance Processes based on Change Request Data –

von

— Georg Richlofsky —

Vorgelegt der:	Fakultät für Mathematik, Informatik und Naturwissenschaften der Rheinisch-Westfälischen Technischen Hochschule Aachen im Juni 2009
Angefertigt am:	Lehr- und Forschungsgebiet Informatik 3 Prof. Dr. rer. nat. Horst Lichter
Gutachter:	Prof. Dr. rer. nat. Horst Lichter Prof. Dr.-Ing. Manfred Nagl
Betreuer:	Dipl.-Inform. Holger Schackmann

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Inhalt

Zur Entwicklung und Wartung von großen Software Systemen sind heutzutage Change-Request-Management Systeme ein unverzichtbares Hilfsmittel. Mithilfe der Change-Request-Management Systeme ist es möglich, die eingehenden Erweiterungswünsche und Fehlermeldungen systematisch zu erfassen, zu bearbeiten und zu verwalten.

Die Firma KISTERS verwendet unternehmensweit standardisierte Softwareentwicklungs- und Wartungsprozesse zur Entwicklung ihrer Softwareprodukte. Zur Unterstützung dieser Prozesse wird das Change-Request-Management System Bugzilla verwendet. Die Software Bugzilla ist ein integrierter Bestandteil der Entwicklungsprozesse. Die Arbeitsabläufe zur Umsetzung einer Erweiterung und Behebung einer Fehlermeldung werden mithilfe der Change-Requests dokumentiert und gesteuert. Die gesammelten Change-Request Daten erlauben folglich eine Analyse der gelebten Prozesse.

Aufgrund des Volumens an gesammelten Change-Requests bedarf es einer werkzeugunterstützten Auswertung. Hierzu wurde die Software BugzillaMetrics am Lehr- und Forschungsgebiet 3 in Kooperation mit der Firma KISTERS entwickelt. Die Software BugzillaMetrics erlaubt es, Metriken flexibel zur Auswertung der Change-Request Daten zu definieren, so dass einzelne Prozessattribute auf einer höheren Abstraktionsebene sichtbar gemacht werden können.

Mithilfe der Software BugzillaMetrics können die verschiedenen Softwareentwicklungs- und Wartungsprozesse verfolgt und überwacht werden. Ebenso lassen sich mögliche Potentiale zur Prozessverbesserung identifizieren und der Erfolg der anschließenden Prozessverbesserungsmaßnahmen demonstrieren. Hierzu müssen sowohl die Ziele der einzelnen Messungen definiert, als auch die Metriken validiert werden.

1.1 Aufgabenstellung

In dieser Arbeit sollen die Softwareentwicklungs- und Wartungsprozesse auf Basis von Change-Request Daten analysiert werden. Der Schwerpunkt der Analyse liegt auf dem Sichtbarmachen der Eigenschaften von Softwareentwicklungs- und Wartungsprozessen. Hierzu werden im Folgenden die Softwareentwicklungs- und Wartungsprozesse der Firma KISTERS anhand der gesammelten Change-Request Daten betrachtet.

In einem ersten Schritt soll geklärt werden, welche möglichen geforderten Qualitätseigenschaften an den Prozess bestehen (Bsp. Termineinhaltung). Dabei ist der Entwicklungskontext der Softwareprodukte der Firma KISTERS zu berücksichtigen.

Anschließend sollen diese Qualitätseigenschaften anhand der gesammelten Change-Request Daten mithilfe von BugzillaMetrics untersucht werden. Hierzu ist ein exploratives Vorgehen notwendig, bei dem Metriken zur Bewertung der Qualitätseigenschaften entwickelt werden. Die definierten Metriken sind anhand der gesammelten Change-Request Daten zu validieren und gegebenenfalls zu verfeinern.

Abschließend ist zu diskutieren, inwiefern die entwickelten Metriken nützliche Aussagen bezüglich der Qualitätseigenschaften zulassen.

1.2 Gliederung

Das zweite Kapitel beginnt mit einer Einführung in die Softwareentwicklungs- und Wartungsprozesse. Anschließend werden die Grundlagen bezüglich der Messung von Software und Softwaremetriken dargestellt. Im folgenden Abschnitt wird das Konzept des bidirektionalen Qualitätsmodells zur Analyse von Prozessen vorgestellt. Abgeschlossen wird dieses Kapitel mit einer Diskussion, wie die konkreten bidirektionalen Qualitätsmodelle zur Analyse genutzt werden können.

In Kapitel 3 wird das Umfeld der Betrachtung der Analyse beschrieben. Im ersten Abschnitt wird hierzu der KISTERS Softwareentwicklungs- und Wartungsprozess dokumentiert. Anschließend werden das Wekzeug Bugzilla und seine KISTERS spezifischen Besonderheiten erläutert. Abgeschlossen wird dieses Kapitel mit einer kurzen Beschreibung der zur Auswertung verwendeten Software BugzillaMetrics.

Das Kapitel 4 beschreibt das zur Analyse verwendete bidirektionale Qualitätsmodell. Dazu werden zuerst die möglichen Informationsbedürfnisse der Firma KISTERS gesammelt. Anhand der gesammelten Informationsbedürfnisse werden im folgenden Abschnitt die Qualitätseigenschaften abgeleitet. Abschließend wird die Beziehung zwischen den Qualitätsmerkmalen und den Qualitätseigenschaften anhand der Qualitätsindikatoren erläutert.

Das Kapitel 5 dient der Vorbereitung der Analyse. Im ersten Abschnitt werden verschiedene Aspekte im Bezug auf die Analyse der Projekte und gelebten Prozesse erörtert. Der zweite Abschnitt gibt Hinweise zur Analyse mit einem bidirektionalen Modell. Danach werden Besonderheiten und gefundene Fehler des Werkzeugs BugzillaMetrics behandelt.

Die gesammelten Change-Requests werden anhand der explorativ entwickelten Metriken im Kapitel 6 analysiert. Hierzu wird in einem ersten Schritt die gewählte Vorgehensweise zur Analyse erläutert. Im Anschluss werden die ermittelten Ergebnisse der Metriken dargestellt.

In Kapitel 7 werden die ermittelten Ergebnisse der Metriken unter Berücksichtigung der Qualitätsindikatoren hinsichtlich der Qualitätseigenschaften des vorgestellten bidirektionalen Qualitätsmodells diskutiert. Die Qualitätsindikatoren werden hierbei auf ihre Aussagekraft hin geprüft.

Abschließend wird in Kapitel 8 die Arbeit zusammengefasst und ein Ausblick auf weitere Arbeiten gegeben.

2 Grundlagen

Inhalt

2.1 Softwareentwicklungs- und Wartungsprozesse

Die Entwicklung von Software ist ein komplexer Vorgang, bei dem die Anforderungen von Anwendern in ein fertiges Softwareprodukt transformiert werden. Um Software erfolgreich zu entwickeln, bedarf es neben der Implementierung der Software einer Vielzahl von Aktivitäten, die in Prozessen gruppiert werden. Der Ablauf der *Prozesse* wird mithilfe von *Vorgehensmodellen* beschrieben.

2.1.1 Prozess

Für den Begriff des Prozesses gibt es eine Vielzahl verschiedener Definitionen. Die verschiedenen Definitionen betonen unterschiedliche Aspekte von Prozessen. In dieser Arbeit wird die weitverbreitete Definition der ISO 9000:2000 verwendet:

Definition: process - set of interrelated or interacting activities which transforms inputs into outputs [ISO 9000:2000]

Demnach ist ein Prozess eine Abfolge von in gegenseitiger Beziehung stehenden oder interagierenden Aktivitäten, welche Eingaben in Ausgaben transformieren. Die Aktivitäten eines Prozesses sind eine Liste von Aktionen, die durchgeführt werden um die Ergebnisse des Prozesses zu erzielen. Einen Prozess kann man folglich durch die Gruppierung von weiteren Prozessen oder die Zerlegung in weitere Prozesse beschreiben. Häufig spricht man in diesem Fall auch von Prozessgruppen (engl. process group). Viele Definitionen weisen auf die Notwendigkeit eines Ziels bei der Definition eines Prozesses hin:

Definition: process - a series of actions bringing about a result (PMI, Guide to PMBOK)

Um die Gesamtheit aller Aktivitäten bei der Softwareentwicklung zusammenzufassen, spricht man auch von **dem** Softwareprozess. Häufig werden die Begriffe Prozess und Projekt verwechselt. Ein Projekt ist die zeitlich begrenzte Anstrengung mit dem Ziel, bestimmte Resultate zu erstellen [nach IEEE1490]. Jedem Projekt liegt ein Prozess zugrunde. Ein solcher Prozess kann unbewusst gewählt, oder auch formal beschrieben und definiert sein. Die Beschreibung erfolgt entsprechend dem Abstraktionsgrad des Prozesses in natürlicher Sprache, in strukturierter Form oder durch verschiedene Diagrammtypen, wie Ablaufgraphen, welche die Interaktion zwischen den Aktivitäten verdeutlicht, oder durch Rollenmodelle, welche die *Verantwortlichkeiten* für Aktivitäten innerhalb des Prozesses darstellen. Die Literatur enthält eine Vielzahl von Vorlagen zur Beschreibung von Prozessen.

2.1.2 Vorgehens- und Prozessmodell

Der Ablauf von Softwareprozessen wird anhand von Vorgehens- oder Prozessmodellen beschrieben. Den Kern der Prozessmodelle bilden die Vorgehensmodelle. Ein Vorgehensmodell beschreibt eine Sicht auf den Prozess und betrachtet den Prozess somit unter ausgewählten Gesichtspunkten. Die Prozessmodelle erweitern diese Betrachtung um Aspekte, wie die Organisationsstruktur sowie Vorgaben für das Projektmanagement, die Qualitätssicherung, die Konfigurationsverwaltung und die Dokumentation [LL07]. Die Trennung zwischen Vorgehens- und Prozessmodell ist häufig unscharf. Daher wird im Folgenden, wie oftmals in der Literatur, kein Unterschied zwischen Vorgehens- und Prozessmodell gemacht.

Ein bekanntes Vorgehensmodell ist das **Wasserfallmodell**. Das Wasserfallmodell betrachtet die Aktivitäten und deren Verknüpfung während des Entwicklungsprozesses. Der Ablauf des Softwareentwicklungsprozesses wird als eine Abfolge von Aktivitäten beschrieben, die über ihre produzierten Teilergebnisse miteinander verknüpft werden. Aufgrund der Tatsache, dass die Teilergebnisse der verschiedenen Aktivitäten fehlerhaft sein können, erlaubt das Wasserfallmodell in vorangegangene Aktivitäten zurückzukehren, um die erkannten Fehler zu korrigieren. Durch die Rückkehr in vorherige Aktivitäten entstehen Zyklen. Die Reihenfolge der Aktivitäten des Wasserfallmodells ist fest definiert als Analysieren, Entwerfen, Codieren, Testen, Installieren und Warten.

Die iterative- und inkrementelle Software-Entwicklung sind Repräsentanten der Klasse der nichtlinearen Vorgehensmodelle. Während die linearen Vorgehensmodelle, wie das Wasserfallmodell, Zyklen in ihrem Ablauf vermeiden, sind die Zyklen ein elementarer Bestandteil bei nichtlinearen Vorgehensmodellen. Im Folgenden werden verschiedene nichtlineare Modelle vorgestellt.

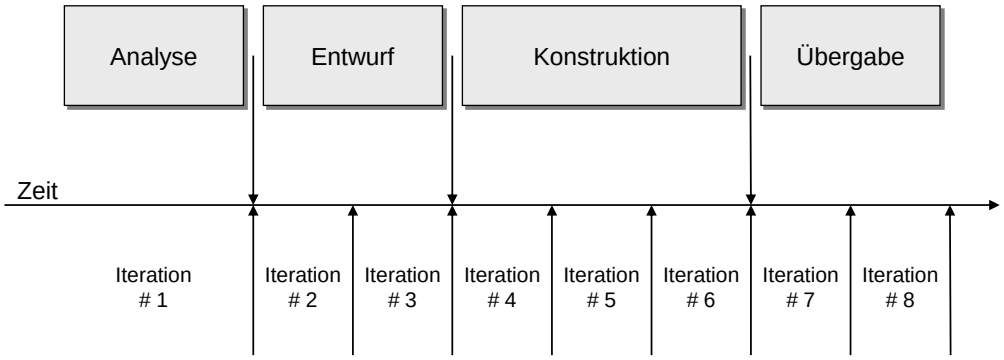


Abbildung 2.1: Abfolge von Iterationen unter Berücksichtigung der Schwerpunkte der Aktivitäten (nach [Wal01])

Definition: Iterative Software-Entwicklung - Software wird in mehreren geplanten und kontrolliert durchgeführten Iterationsschritten entwickelt. Ziel dabei ist, dass in jedem Iterationsschritt - beginnend bei der zweiten Iteration - das vorhandene System auf der Basis der im Einsatz erkannten Mängel korrigiert und verbessert wird. Bei jedem Iterationsschritt werden die charakteristischen Tätigkeiten Analysieren, Entwerfen, Codieren und Testen durchgeführt. [LL07]

Das iterative Software-Entwicklungsmodell ist besonders für die Entwicklung von Softwareprodukten geeignet, bei denen die tatsächlichen Anforderungen nicht vollständig erhoben werden können oder der Einsatz der Software die Anforderungen verändert. Das Softwareprojekt wird in diesem Modell in mehreren Zyklen entwickelt. Das Ziel eines jeden Zyklus ist es, ein lauffähiges Softwareprodukt zu entwickeln, so dass die bisher gesammelten Erfahrungen und neuen Anforderungen an das Softwareprodukt in der Entwicklung berücksichtigt werden können.

In jedem Zyklus werden die charakteristischen Tätigkeiten der Softwareentwicklung durchlaufen. Typischerweise finden die verschiedenen Tätigkeiten in unterschiedlichen Iterationen in ungleicher Intensität statt. In den ersten Iterationen liegt der Schwerpunkt auf der Tätigkeit der Analyse. Mit zunehmender Anzahl von Iterationen verschiebt sich der Schwerpunkt auf die Folgeaktivitäten (siehe Abb. 2.1). Die Ergebnisse der vorhergehenden Iteration werden in den folgenden Iterationen wiederverwendet.

Die inkrementelle Software-Entwicklung entwickelt ein Softwareprodukt in Ausbaustufen. Nachdem im ersten Zyklus das Kernsystem entwickelt wurde, erweitert jede neue Ausbaustufe das bestehende System um neue Funktionen. Dieses Modell eignet sich besonders für Projekte ohne einen definierten Endstand, wie zum Beispiel Betriebssysteme.

Definition: Inkrementelle Software-Entwicklung - Das zu entwickelnde System bleibt in seinem Gesamtumfang offen; es wird in Ausbaustufen realisiert. Die erste Stufe ist das Kernsystem. Jede Ausbaustufe erweitert das vorhande-

ne System und wird in einem eigenen Projekt erstellt. Mit der Bereitstellung einer Erweiterung ist in aller Regel (wie bei der iterativen Entwicklung) eine Verbesserung der alten Komponenten verbunden. [LL07]

Die iterative Software-Entwicklung und die inkrementelle Software-Entwicklung unterscheiden sich hinsichtlich ihrer Absichten. Das Ziel der inkrementellen Software-Entwicklung ist es, mit jeder Ausbaustufe die Funktionalität zu erweitern. Bei der iterativen Software-Entwicklung versucht man mithilfe der Zyklen das tatsächliche Ziel besser zu erreichen.

Häufig wird neben der Betrachtung des eigentlichen Entwicklungsprozesses der gesamte Lebenslauf (engl. life-cycle) des Softwareprodukts betrachtet. Das Modell des **Software-Lebenszyklus** beschreibt den Lebenslauf von dem Konzept bis hin zur Stilllegung der Software. Der Software-Lebenslauf umfasst üblicherweise: Analyse, Spezifikation, Entwurf, Implementierung, Test, Integration, Abnahme, Betrieb und Wartung sowie Stilllegungsphase. Dabei ist es nicht verboten, dass die Phasen einander überlappen oder in iterativer Weise ablaufen [IEEE 610.12:1990]. Insbesondere lässt sich die Betrachtung des Lebenslaufes um die Softwarenentwicklung unterstützenden Prozesse, wie der Dokumentation und Konfiguration, oder die Wartungsphase, erweitern [ISO 12207].

2.1.3 Software-Wartungsprozess

Nachdem die entwickelte Software an den Kunden ausgeliefert wurde, beginnt die Phase der Software-Wartung. Bei der anschließenden Nutzung der Software durch den Kunden werden Fehler entdeckt, die nicht während der Softwareentwicklung gefunden wurden. Des Weiteren ändert sich die Umgebung der Software, so dass eine Erweiterung oder Anpassung an der Software notwendig wird. Daher wird die Software fortlaufend verändert, um für den Kunden nutzbar zu bleiben.

Der **Wartungsprozess** wird genutzt, wenn der Code oder ein Dokument eines Softwareprodukts aufgrund eines Problems, einer notwendigen Erweiterung oder einer Anpassung verändert wird. Dabei ist die existierende Software unter Wahrung deren Integrität zu verändern (nach [IEEE 12207-1996]).

	Korrektur	Erweiterung
Proaktive	Prävention	Verbesserung
Reaktive	Korrektur	Anpassung

Tabelle 2.1: Kategorien der Wartung (nach [SWE04])

Die Wartungsmaßnahmen lassen sich in verschiedene Kategorien unterteilen. Dabei wird zum einen unterschieden, ob die Software erweitert oder korrigiert wird und zum anderen, ob die Wartung proaktiv oder reaktiv erfolgt (siehe Tabelle 2.1). Hierdurch entstehen die folgenden vier verschiedenen Kategorien der Wartung:

- *Adaptive Wartung* (engl. adaptive maintenance): Die Software wird angepasst, um neue oder sich ändernde Anforderungen der Umgebung zu erfüllen.

- *Korrektive Wartung* (engl. corrective maintenance): Die Software wird verändert, um bekannte Fehler zu beheben.

- *Verbessernde Wartung* (engl. perfective maintenance): Die Software wird verändert, um die Ausprägung bestimmter Software-Qualitätsmerkmale zu verbessern.

- *Präventive Wartung* (engl. preventive maintenance): Die Software wird mit dem Ziel verändert, bestehende Defekte und latente Fehler zu beheben, bevor diese zum Versagen der Software führen.

Während die Kategorien der adaptiven und korrektiven Wartung eindeutig sind, erfordert die Kategorie der Verbesserung eine präzisere Definition [LL07]. Die verbessernde Wartung erhöht demnach den Nutzen des Systems, indem sie ein Merkmal, welches bereits die Spezifikation erfüllt, verändert.

Besonders kritisch wird die Kategorie der präventiven Wartung in der Literatur diskutiert [LL07] [SP01]. In der Diskussion wird die Betrachtung ausschließlich auf die Software selbst beschränkt. Viele der Definitionen von Wartung erweitern jedoch die Betrachtung auf das gesamte (Software-)Produkt, so dass der Systementwicklungskontext zu berücksichtigen ist. So referenziert die Definition des Wartungsbegriffes der IEEE 610-12:1990 die präventive Wartung ausschließlich im Systemkontext.

Der Wartungsprozess (nach ISO 14764:2006) umfasst verschiedene Aktivitäten (siehe Abb. 2.2):

- Prozess Implementierung

- Analyse von Problemen und Modifikationen

- Implementierung der Modifikation

- Review der Wartung/Abnahme

- Migration

- Stilllegung der Software

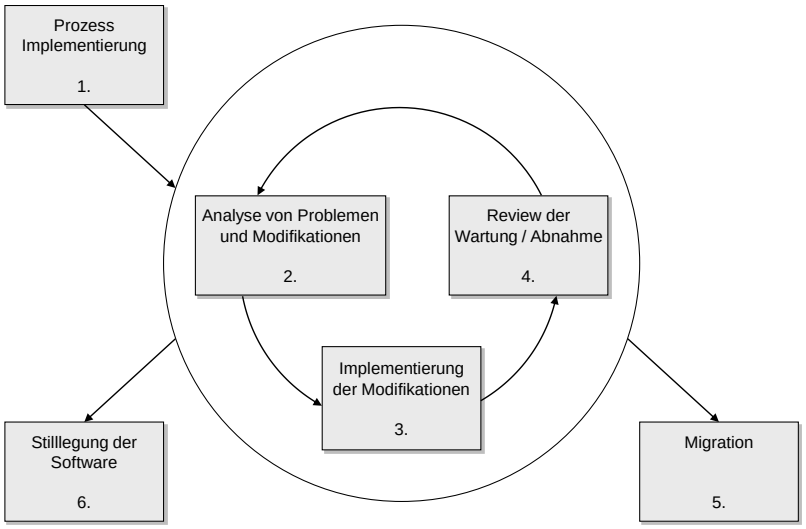


Abbildung 2.2: Aktivitäten des Wartungprozesses (nach [ISO06])

Die Aktivitäten des Wartungsprozesses sind denen der Software-Entwicklung ähnlich. Folglich empfiehlt die ISO 14764 auch auf ähnliche Entwicklungsprozesse zurückzugreifen. Dennoch umfasst der Wartungsprozess einige besondere Aktivitäten.

Zuerst müssen die eingehenden Änderungswünsche und Problemmeldungen erfasst und analysiert werden. Aufgrund der Ergebnisse der Analyse wird entschieden, ob die Anfrage akzeptiert oder abgelehnt wird. Eine akzeptierte Anfrage wird anschließend implementiert. Abschließend werden das Ergebnis einer Anfrage und ihre Änderung geprüft. Die Anfrage wird geschlossen, falls ihr Initiator die Veränderung akzeptiert.

Der Wartungsprozess umfasst vielfach zusätzliche Aktivitäten, um Anwender im Umgang mit dem Softwareprodukt zu unterstützen (**Support-Prozess**).

Für die Annahme und Verwaltung von Anfragen empfiehlt sich die Verwendung eines **Change-Request-Management-Systems**. Die Änderungswünsche und Fehlermeldungen in einem solchen System bezeichnet man als „**Change-Requests**“. Ein Change-Request enthält neben der eigentlichen Anfrage eine Vielzahl von Informationen, wie zum Beispiel den aktuellen Status der Anfrage, den verantwortlichen Bearbeiter, die Dringlichkeit, oder die Version des Softwareprodukts. Die Change-Request-Management-Systeme werden je nach Spezialisierung oder ihrer historischen Herkunft auch Bug-Tracking-Systeme oder Issue-Tracker genannt. Dementsprechend werden die Change-Requests auch als Bug-Reports, Issues oder Cases bezeichnet.

2.2 Messung und Softwaremetriken

Die Messung (engl. measurement) und Softwaremetriken sind von zentraler Bedeutung für das Software-Engineering. Denn erst mithilfe von Messungen können quantitative Aussagen über Prozesse und Produkte gemacht werden. Dementsprechend vielfältig sind auch die Ziele, mit denen die Messungen ausgeführt werden: Verstehen, Falsifizieren, Begreifen, Lernen, Beweisen, Validieren, Management, Kontrolle, Verbesserung, Exploration, Untersuchungen, Zertifikation [ED07].

Die im Software-Engineering betrachteten Objekte sind sehr unterschiedlich. Neben dem Produkt der Software, werden auch abstrakte Konzepte wie Prozesse oder konkrete Gegenstände wie Ressourcen betrachtet. Die unterschiedlichen Objekte werden unter dem Begriff der Entitäten subsummiert.

Im Rahmen der Messung wird einer Entität eine Zahl oder ein Symbol zugewiesen, welches ein Attribut der Entität beschreibt. Die Messung einer Entität erfolgt durch Anwendung einer Metrik:

Defintion: software quality metric:
(1) A quantitative measure of the degree to which an item possesses a given quality attribut.
(2) A function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given quality attribute. [IEEE 610.12]

Die *software quality metric* ist eine Abbildung eines Attributs einer Entität auf eine skalare oder vektorielle Größe. Anstelle des Begriffes „software quality metric“ wird kürzer der Begriff „Metrik“ als Synonym gebraucht. Zu beachten ist, dass die hier verwendete Definition „software quality metric“ **nicht** der mathematischen Definition einer Metrik entspricht.

Die ermittelten Messdaten lassen sich zur Auswertung durch verschiedene (Rechen-)Operationen manipulieren. Welche Operationen und Vergleiche bei der Untersuchung von Messwerten gültig sind, ist abhängig von der Beziehung zwischen den Merkmalsausprägungen und den ihnen zugewiesenen Werten. Diese Beziehung wird mittels Skalentypen beschrieben (siehe Tab. 2.2). Die gebräuchlichsten fünf Skalentypen sind: Nominalskala, Ordinalskala, Intervallskala, Rationalskala und Absolutskala

- Nominalskala: Unterscheidung von Entitäten
- Ordinalskala: Anordnung der Entitäten
- Intervallskala: Verhältnis der Entitäten
- Rationalskala: Verhältnis von Messwerten
- Absolutskala: Messwerte

Skala	Logische und mathematische Operationen	zulässige Transformationen
Nominal	=, !=	bijektive Abbildungen
Ordinal	=, !=, <, >	jede streng monotone Abbildung
Interval	=, !=, <, >, +, -	$g(x) = a * b + c$
Rational	=, !=, <, >, +, -, *, /	$g(x) = a * x$
Absolut	=, !=, <, >, +, -, *(, /)	$g(x) = x$

Tabelle 2.2: Skalen

In der Praxis kommt es durch die Kombination und Untersuchung von Messergebnissen zu Skalenübergängen. Entsprechend können bei einem Übergang in eine „schwächere“ Skala nur deren schwächere logische und mathematische Operationen bei der Untersuchung verwendet werden. Die Absolutskala erlaubt eigentlich nicht die Division, in der Praxis wird sie jedoch häufig in die Rationalskala umgedeutet. Hierzu erweitert man den Zahlenbereich von den natürlichen Zahlen auf die reellen Zahlen und erlaubt zudem die Divisionsoperation.

2.3 Qualitätsmodelle

Viele verschiedene Faktoren beeinflussen die Qualität der Softwareprodukte. So besteht heutzutage der allgemeine Konsens, dass die Prozessqualität über die Projektqualität indirekt auf die Entwicklungs- und Wartungsqualität des Softwareprodukts einwirkt. Eine hohe Prozessqualität soll hierbei die Produktion von qualitativ hochwertiger Software ermöglichen (vgl. [ISO 9126] [LL07]).

Im Allgemeinen wird der Qualitätsbegriff unabhängig von dem Gegenstand definiert:

Definition: Qualität - Gesamtheit von Eigenschaften und Merkmalen eines Produktes oder einer Tätigkeit, die sich auf die Eignung zur Erfüllung gegebener Erfordernisse beziehen.

[...]

Anmerkung 2: Ein Produkt ist zum Beispiel jede Art von Waren, Rohstoffen, aber auch der Inhalt von Konzepten und Entwürfen. Eine Tätigkeit ist zum Beispiel jede Art von Dienstleistung, aber auch ein maschineller Arbeitsablauf wie ein Verfahren oder ein Prozess.
DIN 55350-11:1995-08 (Auszug) nach [LL07]

Der Begriff des Produkts wird in der gewählten Definition bewusst offen gehalten. Hierdurch können die unterschiedlichen Objekte des Software-Engineerings betrachtet werden. Dies gilt insbesondere für Prozesse, Projekte, Produkte und Dokumente.

Die Definition setzt implizit durch die „gegebenen Erfordernisse“ voraus, dass die Qualität immer in einem bestimmten Kontext betrachtet werden muss. Dieser Kontext kann sehr unterschiedlich sein, so dass die Vorstellung über die Qualität ohne gegebenen Kontext stark divergieren kann. Die ISO 9000:2000 bemerkt hierzu: „*Like beauty, everyone may have his or her idea of what quality is*“ [SSM06].

Mithilfe von Qualitäten beschreibt man die gewünschten Eigenschaften des Produkts. Folglich werden Produkte mittels der zuvor individuell festgelegten Qualitäten bewertet. Die gegebene Definition der Qualität enthält jedoch keine konkreten Hinweise, wie Produkte zu bewerten sind. Um die Bewertung eines Produkts durchzuführen, verwendet man in der Praxis ein **Qualitätsmodell**. In einem solchen Qualitätsmodell können die geforderten Qualitäten in der Regel nicht direkt bewertet werden. Beispielsweise ist die Qualität der Planungssicherheit in einem Prozess zu allgemein gefasst und somit für die praktische Anwendung ungeeignet. Daher zerlegt man die Qualitäten solange in weitere relevante Qualitätseigenschaften, bis man sie bewerten kann. Eine Möglichkeit zur Zergliederung der Planungssicherheit ist die Aufspaltung in Aufwand und Termineinhaltung. Bei der Zerlegung ist es durchaus erlaubt, dass mehrere Qualitätseigenschaften gemeinsame Teileigenschaften besitzen. Die einzelnen Qualitätseigenschaften dürfen sich zueinander in einer Konkurrenzbeziehung befinden, so dass eine Gewichtung der konkurrierenden Qualitätseigenschaften notwendig ist, um Zielkonflikte zu vermeiden. Im Allgemeinen strebt man eine disjunkte Zerlegung der Qualitätseigenschaften an. Denn durch die nichtdisjunkte Zerlegung entstehen zusätzlich Interdependenzen, die bei einer Gewichtung zu berücksichtigen sind.

Man definiert ein Qualitätsmodell als eine Menge von Qualitätseigenschaften und deren Beziehung zueinander. Üblicherweise besitzt ein so erstelltes Qualitätsmodell eine hierarchische Struktur und wird als Baum oder azyklischer Graph dargestellt (siehe Abb. 2.3). Die Menge der Qualitätseigenschaften bildet die Grundlage um Anforderungen an die Qualität zu formulieren und die Qualität zu evaluieren [ISO9216].

Ein weitverbreitetes Qualitätsmodell zur Bewertung von Qualität eines Softwareprodukts ist die ISO 9216. In diesem Modell werden aus verschiedenen Sichten unterschiedliche Qualitätsmodelle entwickelt. Die interne Qualität betrachtet die Qualität des Softwareprodukts aus der Perspektive des Entwicklers. Aus der Sicht des Anwenders der Software wird die externe Qualität abgeleitet. Die Nutzungsqualität (quality-in-use) ist ein separates Qualitätsmodell, welches die Software in einer bestimmten Umgebung aus der Sicht des Nutzers darstellt. Die Definition der verschiedenen Qualitätsmodelle unterstreicht die Abhängigkeit vom Kontext und der Perspektive des Betrachters bei der Analyse und Bewertung. In diesen Modellen werden keine Angaben gemacht, wie die einzelnen Qualitätseigenschaften zu messen sind. Die zur Norm gehörenden technischen Berichte (technical reports) ergänzen die entsprechenden Qualitätsmodelle beispielhaft um Metriken für die Qualitätseigenschaft auf der untersten

Ebene der Zerlegung. Durch Anwendung der Metriken wird gemessen, inwiefern das Softwareprodukt den „gegebenen Erfordernissen“ entspricht.

2.3.1 Bidirektionale Qualitätsmodelle

Die Qualitätsmodelle beschreiben die Menge der Qualitätseigenschaften und ihre Beziehung untereinander. Dabei werden abstrakte Qualitäten solange in Qualitätseigenschaften zerlegt, bis diese für eine praktische Anwendung geeignet sind. Um die Qualitätsmodelle für die Bewertung und Analyse (von Prozessen) nutzbar zu machen, müssen diese weiter detailliert werden. Hierzu wird das im Folgenden beschriebene Konzept des bidirektionalen Qualitätsmodells verwendet (vgl. [SL08]).

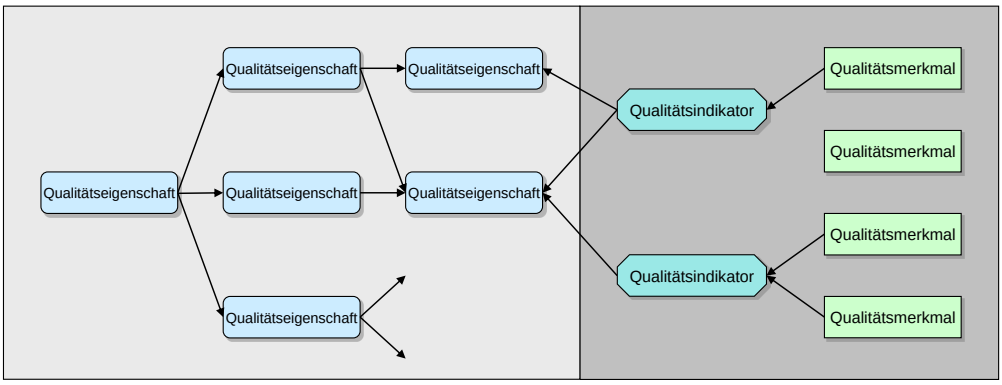


Abbildung 2.3: Konzept des bidirektionalen Qualitätsmodells (nach [SL08])

Das bidirektionale Qualitätsmodell besteht aus zwei verschiedenen Sichten (siehe Abb. 2.3) mit unterschiedlichen Stoßrichtungen. In der ersten Sicht werden, ausgehend von den Anforderungen, die Qualitäten festgelegt. Die Qualitäten werden dann anschließend dem Top-Down Ansatz folgend weiter konkretisiert, indem sie in relevante Qualitätseigenschaften zerlegt werden. Dies geschieht sukzessiv bis die Qualitätseigenschaften entsprechend feingranular zerlegt sind und handhabbar werden. Die **Qualitätseigenschaft** (engl. quality-attribute) ist die zu einer Entität in qualitativer Hinsicht, zugehörige charakteristische (Teil-)Beschaffenheit (vgl. [SSM06]). Beispielsweise lässt sich die „innere Prozessqualität“ in die Qualitätseigenschaften „Know-How-Gewinn“ oder „Projektklima“ zerlegen, welche sich offensichtlich immer noch nicht objektiv bewerten lassen. Um der Qualitätseigenschaft eine objektive Bewertung zuzuweisen, bedarf es der zweiten Sicht in dem bidirektionalen Qualitätsmodell.

Die zweite Sicht entsteht unabhängig von der Ersten. Hierzu betrachtet man die unterscheidbaren und objektiven Merkmale einer Entität. Ein **Qualitätsmerkmal** (engl. quality-property) beschreibt ein (Teil-)Charakteristikum, welches zur Unterscheidung der betrachteten Entität herangezogen werden kann (nach [SSM06]). Die Ausprägung des Qualitätsmerkmals lässt sich durch entsprechende Hilfsmittel automatisch bestimmen. Ein Qualitätsmerkmal für die

Betrachtung von Prozessen ist die Anzahl der „Fire-Fighting“ Aktivitäten in einer Periode. Die Anzahl der „Fire-Fighting“ Aktivitäten lässt sich als die Anzahl der Change-Requests mit höchster Priorität im Change-Request-Managment beschreiben und ihre Anzahl automatisch auslesen.

Die Qualitätsmerkmale werden nun in einer Bottom-Up Vorgehensweise zu den Qualitätsindikatoren zusammengefasst. Die **Qualitätsindikatoren** (engl. quality-indicators) bestehen aus einer Menge von Qualitätsmerkmalen und einer Menge von Regeln, wie die Qualitätsmerkmale bezüglich einer Qualitätseigenschaft zu interpretieren sind. Mithilfe der Qualitätsindikatoren ist es nun möglich, den Zusammenhang zwischen den nicht messbaren Qualitätseigenschaften und den objektiv bestimmbaren Qualitätsmerkmalen zu beschreiben („quantify a quality“). In einem Qualitätsindikator wird den gemessenen Werten der Qualitätsmerkmale eine Bedeutung hinsichtlich der Qualitätseigenschaften zugewiesen. Es wird ebenso der Umgang mit untypischen Ausprägungen und statistischen Ausreißern beschrieben. Der Einfluss der verschiedenen Qualitätsindikatoren auf eine Qualitätseigenschaft wird in gewichteter Form vorgenommen. Hierzu muss die Qualitätseigenschaft weit genug zerlegt sein (operationalisiert sein), um den Einfluss des Qualitätsmerkmals kausal begründet nachvollziehen zu können. Der Qualitätsindikator füllt somit die Lücke zwischen den subjektiv bestimmten Qualitätserwartungen und den objektiv nachvollziehbaren Qualitätsmerkmalen.

Der Vorteil von bidirektionalen Qualitätsmodellen gegenüber den unidirektionalen Qualitätsmodellen ist die Möglichkeit, der zuvor modellierten Qualitätserwartung eine objektiv *nachvollziehbare* Bewertung zuzuweisen (vgl. [ISO9126]). Durch die Berücksichtigung der nicht-technischen Sichtweise ist es im Gegensatz zu reinen Metrikkatalogen (vgl. [KJ08]) möglich, mithilfe der Qualitätsindikatoren die Beziehung zwischen den durch die Metriken erfassten Merkmalen und den Qualitätserwartungen herzustellen. Durch die Unabhängigkeit der Elemente Qualitätsindikator und Qualitätsmerkmal ist es möglich, die beiden Sichten wiederzuverwenden und evolutionär weiterzuentwickeln. Die durch die Qualitätseigenschaften definierte Sicht lässt sich somit innerhalb von Unternehmen für unterschiedliche Prozesstypen, wie das V-Modell 97 oder Agilen-Methoden, weiter ausprägen. Die technische Sicht kann um zusätzliche Techniken erweitert werden, beispielsweise lässt sich der Zugriff auf Change-Request-Management Systeme oder Version-Management Systeme automatisieren, oder um Schnittstellen zu den ERP-Systemen erweitern.

2.4 Vermessung von Qualität mithilfe eines Qualitätsmodells

Die Qualitätsindikatoren beschreiben eine kausal nachvollziehbare Beziehung zwischen der Menge der Qualitätsmerkmale und den Qualitätseigenschaften. Um die betrachteten Gegenstände bezüglich ihrer Qualität zu unterscheiden, zu

analysieren und Entscheidungen über sie vorzubereiten, wird der Qualitätseigenschaft anhand von Regeln ein Wert oder Symbol zugewiesen, wodurch ein direkter Vergleich der einzelnen Qualitätseigenschaften auf der untersten Ebene möglich wird. Das bidirektionale Qualitätsmodell erlaubt jedoch auch einen Vergleich von Qualitätseigenschaften auf einer höheren Ebene bis hin zur Wurzel durch die Bildung von Indexzahlen.

Zur Analyse eines Gegenstandes im bidirektionalen Qualitätsmodell wird zuerst die Ausprägung seiner Merkmale bestimmt. Diese werden unabhängig von ihrem Kontext erfasst und können als eine Funktion in Abhängigkeit von der Entität $f(E)$ bestimmt werden. Die Ausprägung der Merkmale wird entsprechend der drei Schritte *Abstraktion*, *Quantifizierung* und *Rückübertragung* bestimmt (vgl. [SSM06]). Im ersten Schritt der Abstraktion wird von der Entität ein Modell gebildet, welches von dem konkreten Merkmal abstrahiert. Hierzu betrachtet man beispielsweise die Change-Requests, welche in einen bestimmten Zustand gewechselt sind. Anschließend erfolgt die eigentliche Vermessung der Entität, das heißt das Modell wird auf ein Symbol oder eine Zahl abgebildet. So beträgt die Anzahl der Change-Requests, welche in den Zustand REOPENED wechselten, x Stück. Im abschließenden Schritt wird der Messwert auf das betrachtete Merkmal zurück übertragen und erlaubt nun auch statistische Aussagen der Quantifizierung, wie der Prozess A besitzt x Change-Requests, welche im Betrachtungszeitraum in den Zustand REOPENED wechselten. Neben der Quantifizierung des Merkmals der Entität können Vergleiche der Art x ist kleiner y ($x < y$) angestellt und auf die Entitäten zurück übertragen werden. Welche statistischen Aussagen sinnvoll getroffen werden können, hängt von der zugrunde liegenden Skala ab (siehe 2.2 Messung und Softwaremetriken). Somit ist ein Vergleich bezüglich des Abstands verschiedener Reifegrade (Ordinalskala) eines Prozesses im CMMI nicht sinnvoll.

Der Qualitätsindikator beschreibt die Beziehung zwischen den zuvor vermessenen Qualitätsmerkmalen und den abstrakten Qualitätseigenschaften. Häufig müssen bei der Betrachtung einer Qualitätseigenschaft mehrere Merkmale berücksichtigt werden. Daher unterscheidet man bei der Messung von Merkmalen zwischen der **direkten Messung** (engl. direct measure) und der **indirekten Messung** (engl. indirect measure). Die direkte Messung ist unabhängig von anderen Messungen und erfasst eine Information bezüglich eines bestimmten Qualitätsmerkmals. Als Beispiele für direkte Messungen findet man im Software-Engineering:

- Anzahl der Codezeilen (lines of code),
- Anzahl der gefundenen Fehler im Test,
- Dauer der Entwicklung (in Personen-Monate),
- Länge des Zeitintervalls für den Test eines Change-Requests (in Stunden),
- Anzahl der Bearbeiterwechsel bis zum Beginn der Implementierung, oder
- Alter des Change-Requests in Tagen.

Mithilfe der direkten Messung ist es nicht möglich, eine Qualitätseigenschaft unter Berücksichtigung mehrerer Merkmale zu untersuchen. Jedoch bilden die direkten Messungen die Grundlage für die indirekten Messungen. Eine indirekte Messung ist eine Funktion von zwei oder mehreren direkten Messungen und fasst verschiedene Merkmale einer Entität oder dasselbe Merkmal von verschiedenen Entitäten zu einer Information zusammen, das heißt die Information der direkten Messungen wird ergänzt. So erlaubt die direkte Messung der Anzahl der Change-Requests mit der höchsten Priorität zwar einen Vergleich zwischen Prozessen, jedoch wird nicht die Gesamtzahl der Change-Requests des einzelnen Prozesses ausreichend berücksichtigt. Daher wird bei der Normalisierung die Zahl der Change-Requests mit der höchsten Priorität um den Aspekt der Gesamtzahl der Change-Requests des Prozesses ergänzt. Eine einfache Transformation, die keinerlei Information der eigentlichen direkten Messung hinzufügt, wie die Anwendung der Quadratwurzel auf die Messung, gilt daher nicht als indirekte Messung (vgl. [FP98]). In der Literatur werden häufig statt der Begriffe der direkten und indirekten Messung die Begriffe Basismessung (engl. base measure) und abgeleitete Messung (engl. derived measure) verwendet (vgl. [ISO 15939]).

Die bei der Messung ermittelten Werte bezüglich der Qualitätsmerkmale werden anschließend mittels des Qualitätsindikators auf die Qualitätseigenschaften abgebildet. Dies geschieht entsprechend den zuvor festgelegten Regeln, welche den Zusammenhang zwischen den quantitativ gemessenen Werten und der qualitativen Ausprägung der Qualitätseigenschaften herstellten. Hierzu werden die gemessenen Werte mittels eines Vergleiches auf eine qualitative Wertung abgebildet. Für die qualitative Ausprägung verwendet man in der Praxis verschiedene Darstellungen.

Die einfachste Form ist die Abbildung auf die booleschen Werte wahr (=1) und falsch (=0). Die Darstellung findet Anwendung bei Fragen, die mit Ja oder Nein zu beantworten sind. Hierzu zählen beispielsweise Fragen, ob bestimmte Tätigkeiten oder Aktivitäten innerhalb eines Prozesses stattgefunden haben beziehungsweise ausgeübt wurden, bestimmte Merkmalsausprägungen im Betrachtungszeitraum aufgetreten sind, oder Grenzen unter- oder überschritten wurden. So lässt sich beispielsweise darstellen, ob es in einem Prozess zu „Fire-Fighting“ Aktivitäten gekommen ist, das heißt es existieren hoch priorisierte Change-Requests innerhalb des Betrachtungszeitraums. Ein weiteres Beispiel ist die Frage, ob Deadlines im Betrachtungszeitraum verletzt wurden.

Die Darstellung der qualitativen Wertung durch die zwei Zustände wahr und falsch ist häufig zu grobgranular. Daher verwendet man zur Darstellung der qualitativen Wertung vielfach das Intervall $[0, 1]$ oder $[-1, 1]$ zusammen mit der Rationalskala. Nun lassen sich neben der „absoluten“ Güte auch Unterschiede zwischen den gemessen Merkmalsausprägungen berücksichtigen, wie zum Beispiel der Anteil der „Fire-Fighting“ Aktivitäten innerhalb der verschiedenen Prozesse.

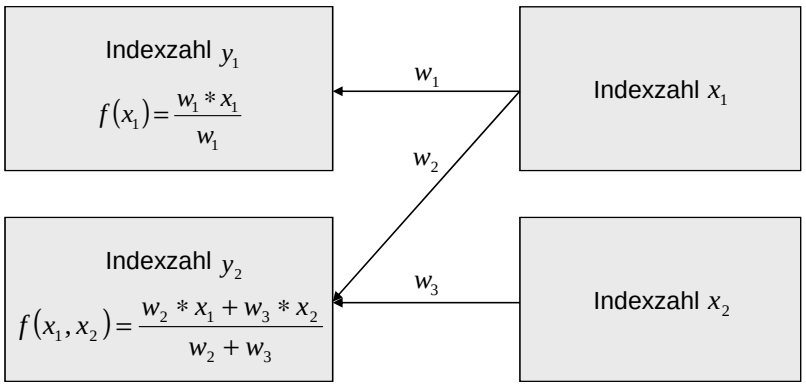


Abbildung 2.4: Beispiel: Gewichtung von Indexzahlen

Die so festgestellte qualitative Ausprägung des Merkmals lässt sich nun in gewichteter Form auf die Qualitätseigenschaft übertragen, so dass diese nun eine qualitative Wertung in Form eines Skalars besitzt. Diese Wertung lässt sich als Indexzahl der Qualitätsausprägung interpretieren:

Eine **Indexzahl** ist eine kollektive Kenngröße für eine Vielzahl unter einem bestimmten Gesichtspunkt zusammenfassbarer [...] [Einzelphänomene], die durch eine geeignete Kombination individueller [...] [Einzelmerkmale] entsteht. (aus [SSM06])

Die durch die Qualitätsindikatoren bestimmten Qualitätseigenschaften besitzen nun eine qualitative Wertung, welche durch eine Indexzahl dargestellt wird. Die Indexzahlen sind im Gegensatz zu den Ausprägungen der Qualitätsmerkmale dimensionslos. Hierdurch wird die Trennung der beiden Sichten im bidirektionalen Qualitätsmodell gewährleistet und die weitere Analyse der Qualitätseigenschaften kann nun unabhängig von der technischen Sicht erfolgen.

Das Konzept der Indexzahlen kann auf die gesamte Sicht der Qualitätseigenschaften übertragen werden. Hierdurch lassen sich auch Qualitätseigenschaften, die in keiner direkten Relation zu den Qualitätsindikatoren stehen, qualitativ bestimmen. Diese qualitative Bewertung der übergeordneten Qualitätseigenschaften lässt sich durch die weitere Aggregation von Indexzahlen ermitteln. Häufig geschieht dies durch die gewichtete Berücksichtigung der einzelnen Indexzahlen (siehe Abb. 2.4). Die Gewichtung repräsentiert die relative Bedeutung der Indexzahl im Bezug auf die ihr übergeordnete Qualitätseigenschaft und ist in Abhängigkeit der Qualitätsziele zu wählen. Hierdurch lässt sich der Einfluss der verschiedenen Qualitätseigenschaften priorisieren.

Zur Aggregation der Indexzahlen sind neben der gewichteten Berücksichtigung der einzelnen Indexzahlen andere Berechnungen möglich. Vielfach ist es von Interesse, ob ein Betrachtungsgegenstand eine Menge von Qualitätseigenschaften besitzt. Entsprechend interpretiert man die Indexzahlen als boolesche Werte und aggregiert die einzelnen Indexzahlen durch eine Konjunktion. So lässt sich beispielsweise ermitteln, ob in einem Prozess alle vorgeschriebenen Tätigkeiten

durchgeführt wurden. Fordert man dagegen, dass nur eine Tätigkeit aus einer Menge von alternativen Tätigkeiten durchgeführt wird, so bietet sich die Disjunktion an.

Die Verwendung von Indexzahlen erlaubt die aggregierte Darstellung der Information, wodurch ein effizienter Vergleich zwischen der Ausprägung der Qualitäten ermöglicht wird. Erlaubt man beispielsweise drei verschiedene qualitative Ausprägungen (besser, neutral, schlechter), so sind bei einem Vergleich von 7 Indexzahlen schon $3^7 = 2187$ mögliche Ausprägungen zu berücksichtigen (vgl. Ampeltest [Küt06]). Jedoch verlieren die Indexzahlen mit steigendem Abstraktionsgrad auch an Information. Während die Beziehung zwischen Qualitätseigenschaft und Indexzahl bei der Abbildung durch die Qualitätsindikatoren leicht nachvollziehbar ist, wird dies mit zunehmendem Abstraktionsgrad der Qualitätseigenschaften erschwert.

2.4.1 Bestimmung der Qualität durch empirische Vergleiche

Die Qualitätsindikatoren beschreiben anhand von Regeln, wie die ermittelten Werte der Qualitätsmerkmale auf die qualitative Interpretation der Qualitätseigenschaften abzubilden sind. Diese Regeln beinhalten verschiedene Arten von Vergleichen um die Ausprägung der Qualität zu bestimmen. Zum Vergleich wird daher ein Referenzwert benötigt, der auf verschiedene Weisen gewählt werden kann.

Als Referenzwert kann eine feste **Schranke** bezüglich der quantitativen Merkmalsausprägung gewählt werden. Beispielsweise wird das Überschreiten des Grenzwerts der Schranke als negativ bewertet und entsprechend auf die Indexzahl abgebildet. Die Abbildung kann sowohl auf boolesche Werte als auch auf eine stärkere Skala erfolgen (vgl. 2.2 Messung und Softwaremetriken). Im Software-Engineering interpretiert man das Überschreiten einer bestimmten Zeilenzahl des Quellcodes einer Methode als eine negative Auswirkung auf die Qualitätseigenschaft Lesbarkeit. Für die Betrachtung von Prozessen lässt sich das Verhältnis der verletzten Deadlines bezüglich aller Deadlines als Schranke wählen. Überschreitet dieses Verhältnis einen gewissen Schwellenwert, ist eine genaue Prüfung des Sachverhalts vorzunehmen und gegebenenfalls entsprechende Maßnahmen zu ergreifen. Neben dem Referenzwert wird in der Regel noch eine Toleranzgrenze bestimmt, damit nur relevante Abweichungen der Messung zu einer negativen Qualitätsausprägung führen. Die nicht relevanten Abweichungen entstehen beispielsweise durch Messungenauigkeiten bei direkten Messungen oder Rundungsfehler bei indirekten Messungen. Die Festlegung der Toleranzgrenzen geschieht häufig auf Basis von Erfahrungswerten oder mittels statistischer Methoden (vgl. [KJ08]).

In der Praxis ist es oft ein Problem, die Werte für die Referenzwerte und ihrer zugehörigen Toleranzen zu bestimmen. Häufig sind noch gar keine Referenzwerte aufgrund der geringen Erfahrung mit dem Betrachtungsgegenstand vorhanden,

weshalb diese Werte dann eher von ideellen Vorstellungen abgeleitet werden, welche für die praktische Anwendung ungeeignet sind. Daher bietet sich der Vergleich mit einem realen **Referenzgegenstand** an. Ein solcher Referenzgegenstand für die Analyse von Prozessen ist beispielsweise ein ähnliches Projekt welches nach demselben Prozess ausgeführt wird oder ein ähnliches Projekt aus einer empirischen Datenbank. Die Merkmalsausprägung des Betrachtungsgegenstandes wird nun mit dem vom Referenzgegenstand abgeleiteten Referenzwert verglichen. Der Qualitätsindikator beschreibt nun, wie diese Ausprägung auf die entsprechende Qualitätseigenschaft zu interpretieren ist. Dieser Ansatz setzt implizit voraus, dass ein entsprechender geeigneter Vergleichsgegenstand existiert, der hinsichtlich der Qualitätseigenschaften und Qualitätsmerkmale vergleichbar ist. Des Weiteren müssen auch hier die Toleranzgrenzen bestimmt werden.

Dieser Ansatz lässt sich in natürlicher Weise erweitern, indem anstelle eines festen Referenzgegenstandes eine Menge von Referenzgegenständen betrachtet wird. Hierzu wird zu jedem Betrachtungszeitpunkt der Referenzwert für den Vergleich anhand der in einer Datenbank hinterlegten **empirischen Daten** bestimmt (vgl. [SSM06]). Die Referenzgegenstände sollten hinsichtlich der geforderten Qualitätseigenschaften vergleichbar sein und dieselben Qualitätsmerkmale besitzen. Zudem wird gefordert, dass ein Vergleich zwischen den Qualitätsmerkmalen unabhängig von Volumeneffekten, wie zum Beispiel der Anzahl der LOC oder Projektgröße, möglich ist. Hierzu sind die gemessenen Qualitätsmerkmale mit geeigneten Größen zu normalisieren. Das Ziel der Normalisierung ist die Betrachtung eines Qualitätsmerkmals, ohne dass dieser Vergleich von der „Größe/Umfang“ des Gegenstandes dominiert wird. Der Vergleich eines Prozesses mit x Änderungsanforderungen mit einem Prozess mit 2x Änderungsanforderungen sollte hinsichtlich eines bestimmten Qualitätsmerkmals unabhängig von der Anzahl der Änderungsanforderungen möglich sein.

Für die normierten Daten werden im Allgemeinen die folgenden statistischen Kenngrößen erhoben:

- Minimum: Die kleinste Merkmalsausprägung innerhalb der Beobachtungen
- Unteres Quartil: 25% der Beobachtungen sind kleiner,
  75% der Beobachtungen sind größer
- Median (Quartil 2): 50% der Beobachtungen sind kleiner,
  50% der Beobachtungen sind größer
- Oberes Quartil: 75% der Beobachtungen sind kleiner,
  25% der Beobachtungen sind größer
- Maximum: Die größte Merkmalsausprägung innerhalb der Beobachtungen

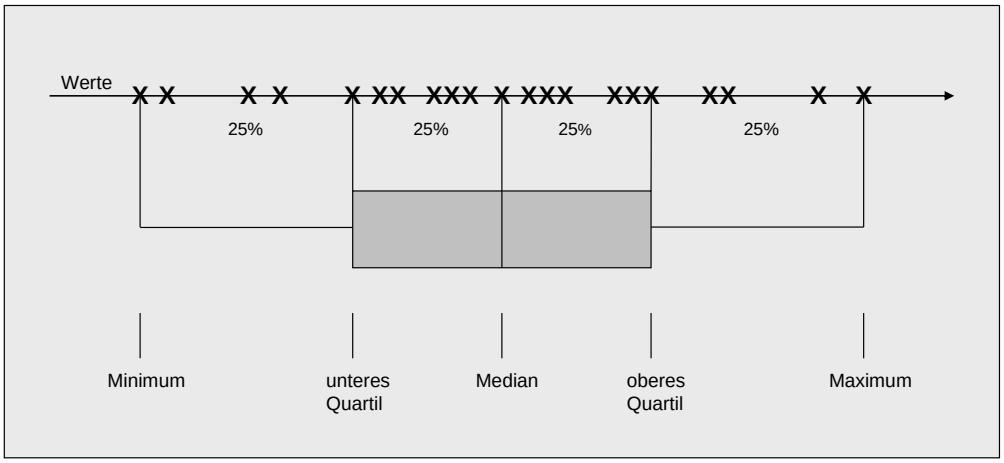


Abbildung 2.5: Boxplot (nach [SSM06])

Die statistischen Kenngrößen lassen sich anschaulich mit Boxplots darstellen (siehe Abb. 2.5). Der Boxplot benutzt Quartil zur graphischen Darstellung von Lagemaßen und hebt potentielle Ausreißer hervor.

Als Lagemaß ist der Median dem arithmetischen Mittel in der Betrachtung vorzuziehen, da dieser robust gegenüber Ausreißer reagiert. Daneben wird häufig das „Winsorized mean“ betrachtet, welches im Gegensatz zum arithmetischen Mittel weniger sensitiv auf Ausreißer reagiert und mehr Information bezüglich der Stichprobe nutzt. Hierzu werden die k kleinsten Beobachtungen jeweils durch die k+1 kleinste Beobachtung ersetzt und, analog für die größten Betrachtungen, die l größten Beobachtungen jeweils durch die l-1 größte Beobachtung ersetzt und abschließend durch die Anzahl der Beobachtungen dividiert.

Beispiel:

j	1	2	3	4	5	6	7
k	1	1	2	5	7	24	100

Median $= x_{\frac{7+1}{2}} = 5$

arithmetische Mittel $= \frac{1+1+2+5+7+24+100}{7} = \frac{140}{7} = 20$

Winsorized Mean $= \frac{1+1+2+5+7+24+24}{7} = \frac{65}{7} = 9,285$

Die so ermittelten Kenngrößen werden nun als Referenzwerte für die Vergleichsbasis verwendet. Dabei gilt als Prämisse, dass die Merkmalsausprägungen, die schlechter als der für den Median ermittelten Wert sind, Verbesserungspotential besitzen. Entsprechend teilt der Median die Beobachtungen in zwei Äquivalenzklassen: besser und schlechter. Die Äquivalenzklassen sind entsprechend der Interpretation der qualitativen Merkmalsausprägung zu wählen. Folglich ist das Maximum als der qualitativ beste beziehungsweise schlechteste Wert zu interpretieren. Der Median als Grenzwert ist plausibel, da die Hälfte der Referenzgegenstände eine qualitativ gleiche oder bessere Merkmalsausprägung erreicht hat.

Die zuvor bestimmten statistischen Kenngrößen erlauben eine weitere Verfeinerung der Äquivalenzklassen in besser, neutral und schlechter. Hierzu zerlegt man die Messwerte in drei Klassen: Minimum bis zum unteren Quartil, unteres Quartil bis zum oberen Quartil, und oberes Quartil bis zum Maximum. Der Betrachtungsgegenstand lässt sich nun durch den Vergleich mit den Referenzgegenständen in eine der Klassen mit der dazugehörigen Interpretation zuordnen. Sollen zudem noch Entwicklungstendenzen berücksichtig werden, bietet es sich an, die Beobachtungen zwischen dem unteren und dem oberen Quartil in weitere Klassen zu unterteilen, beispielsweise in unteres Quartil bis Median und Median bis oberes Quartil.

Der Vorteil der Verwendung von empirischen Daten aus der Datensammlung ist die Praxistauglichkeit, da durch die Verwendung der statistischen Kennzahlen realistische und realisierbare Vergleichswerte als Vergleichsbasis herangezogen werden. Diese Vergleichswerte lassen sich zudem aus der empirischen Datenbank automatisch bestimmen. Hingegen problematisch für die Umsetzung ist die Verfügbarkeit der empirischen Daten. Diese müssen in entsprechender Anzahl und Qualität verfügbar sein.

3 Umfeld der Betrachtung

Inhalt

Bei der Analyse von Softwareentwicklungs- und Wartungsprozessen muss insbesondere das Umfeld der Betrachtung berücksichtigt werden. Dazu werden im Folgenden zuerst die KISTERS-Prozesse dokumentiert. Anschließend wird der in die Prozesse integrierte KISTERS-Bugzilla mit seinen Besonderheiten erläutert. Abschließend wird eine Einführung in die Verwendung des Werkzeugs BugzillaMetrics gegeben, welches im Rahmen dieser Arbeit zur Auswertung der Change-Request Daten genutzt wird.

3.1 Der KISTERS Softwareentwicklungs- und Wartungsprozess

Das Unternehmen KISTERS gliedert seine Geschäftsprozesse in die drei verschiedenen Bereiche: Führungsprozesse, operative Prozesse und unterstützende Prozesse. Die Softwareentwicklung stellt die Haupttätigkeit des Geschäftsbereichs der Umweltinformatik dar und ist in die operativen Prozesse eingegliedert. In der Softwareentwicklung werden Softwareprodukte entwickelt und IT-Projekte realisiert [SSM06].

Der Produktentwicklungsprozess wird durch die eingehenden Anforderungen angestoßen. Die eingehenden Anforderungen werden zuerst erfasst und anschließend analysiert. In regelmäßigen Abständen findet der Planungs-Subprozess statt, in dessen Verlauf die Anforderungen um Steuerinformationen, wie Zeitaufwand und geplanter Bearbeitungstermin, ergänzt werden. Die Bearbeitung der Anforderung wird entsprechend dem zuvor festgelegten Bearbeitungstermin durchgeführt. Danach werden die bearbeiteten Anforderungen im Test-Subprozess geprüft, so dass über die Freigabe entschieden werden kann. Bei einer positiven Entscheidung hinsichtlich der Freigabe wird das neue Release durch den Auslieferungsprozess an den Kunden verteilt.

Die eingehenden Anforderungen werden kontinuierlich im Subprozess „**Anforderungserfassung**“ gesammelt. Eine Anforderung kann die Erweiterung der

Funktionalität, die Änderung bestehender Funktionalität oder die Behebung eines Fehlers sein. Angestoßen wird dieser Prozess durch die eingehenden Kundenanforderungen von internen und externen Kunden. Die verschiedenen eingehenden Anforderungen werden in einheitlicher Form dokumentiert. Das Ergebnis dieses Subprozesses ist eine erfasste Anforderung.

Über die erfassten Anforderungen wird in der „**Anforderungsanalyse**“ entschieden. Zuerst werden sie auf Vollständigkeit geprüft, Risiken und Chancen analysiert, das betreffende Release identifiziert und Interdependenzen hinsichtlich des Projekts beziehungsweise des Produkts geprüft. Anschließend wird hinsichtlich der weiteren Bearbeitung entschieden. Hierzu unterscheidet man zwischen den folgenden Möglichkeiten:

- Die Anforderung wird abgelehnt und dokumentiert. Gegebenenfalls ist eine Mitteilung an den Reporter zu versenden, damit dieser den Kunden informieren kann.
- Die Anforderung bleibt offen und wird mit einem Datum zur Wiedervorlage als „zurückgestellte Anforderung“ dokumentiert. Diese durchläuft bei Wiedervorlage den Prozess erneut. Ebenso wie bei der abgelehnten Anforderung ist der Kunde zu informieren.
- Die Anforderung betrifft die Produktentwicklung beziehungsweise das Customizing und wird folglich als „analysierte Anforderung“ gekennzeichnet. Die „analysierte Anforderung“ stößt den Subprozess der Releaseplanung an. Bei Anforderungen hinsichtlich der Produktentwicklung ist die Variabilität der Anforderung zu dokumentieren.

Das Ergebnis der „Anforderungsanalyse“ ist die Analyse der erfassten Anforderungen hinsichtlich der weiteren Bearbeitung.

Die analysierte Anforderung wird hinsichtlich ihrer Realisierung in der „**Release-Planung**“ zuerst grob kategorisiert und in den Entwicklungsplan eingepflegt. Die in dem Entwicklungsplan eingepflegten Anforderungen werden durch den Systemdesigner weiter zerlegt und aus der Systemsicht beschrieben. Zusätzlich wird die Anforderung um die Aspekte Aufwandsabschätzung, Risikomanagement, Zeitplanung und Ressourcenplanung ergänzt. Die so bearbeitete Anforderung wird auch Task genannt. Einer Task wird unter Berücksichtigung von Inter- und Intra-Produkt Dependenzen ein Bearbeitungstermin zugewiesen und der Zeitplan für das Produkt beziehungsweise Projekt aktualisiert. Für die dringlichen Anforderungen wird dieser Subprozess in verkürzter Form mit reduzierten Planungstätigkeiten und Abstimmung durchgeführt.

Die gesammelten Tasks für ein Release werden im Subprozess „**Task-Bearbeitung**“ realisiert. Erfordert die Task eine Überarbeitung der Architektur, ist diese in der Architektur-Beschreibung zu dokumentieren. Anschließend werden die Module und die zugehörigen Testfälle entwickelt oder überarbeitet. Eine Task gilt als erfolgreich umgesetzt, falls der Entwicklertest bestanden wurde. Werden

im Entwicklertest Fehler gefunden, sind diese durch Rücksprung in die entsprechende Tätigkeit zu beheben. Nach der erfolgreichen Umsetzung der Task sind ihre Wechselwirkungen hinsichtlich anderer Entwicklungszweige und Anforderungen zu dokumentieren. Zusätzlich wird für die Task die tatsächlich geänderte beziehungsweise die neue Funktionalität beschrieben. Die bearbeitete Anforderung erfordert eine Aktualisierung der begleitenden Dokumentation, wie Anwenderdokumentation, Schulungsunterlagen oder Releasenotes. Die dringenden Tasks durchlaufen diesen Subprozess in reduzierter Form, was insbesondere die Dokumentation betrifft. Demzufolge wird gefordert, die entsprechenden Schritte im Subprozess nachzuholen. Das Ergebnis des Subprozesses nach der Abschlusskontrolle sind die „bearbeiteten Anforderungen“.

Die „bearbeiteten Anforderungen“ werden im Subprozess „**Test**“ durch manuelle oder automatische Tests geprüft. Deren Ergebnisse werden dabei im Testprotokoll dokumentiert. Die „bearbeiteten Anforderungen“, die erfolgreich getestet wurden, werden als „getestete Anforderungen“ bezeichnet. Für die gefundenen Fehler wird zwischen den nicht erfolgreich getesteten Anforderungen und den neu gefundenen Fehlern unterschieden. Die nicht erfolgreich getesteten Anforderungen werden zurückgesetzt zu „erfasste Anforderungen“ und erneut der Anforderungsanalyse unterzogen. Die neu gefundenen Fehler werden als „analysierte Anforderungen“ erfasst und stoßen den Subprozess der Releaseplanung an. Die Entscheidung über die Freigabeempfehlung des Release geschieht auf Basis der Testprotokolle und wird im Testbericht dokumentiert.

Die Prozesse „**Auslieferung**“ und „**Abnahme**“ schließen die Entwicklung eines Projekts beziehungsweise das Release eines Produkts ab. Die neuen Anforderungen werden kontinuierlich im Subprozess „Anforderungserfassung“ gesammelt.

Im Produktentwicklungsprozess wird zwischen dem **Customizing** und der **Produktweiterentwicklung** unterschieden. Im Customizing werden die kundenspezifischen Anforderungen in Projekten entwickelt. Die Produktanforderungen werden hingegen in der Produktweiterentwicklung realisiert und in Form von Releases an den Kunden ausgeliefert. Unabhängig von der Art der Anforderungen werden die Subprozesse Releaseplanung, Taskbearbeitung, Test, Freigabe und Auslieferung durchlaufen. In der Releaseplanung sind insbesondere die Abhängigkeiten zwischen den einzelnen Anforderungen im Hinblick auf Produktentwicklung und Customizing (Intra-Produkt) sowie den Inter-Produkt Dependenzen zu untersuchen.

Neben dem Ablauf des Produktentwicklungsprozesses und seiner Ergebnisse werden in ergänzenden Dokumenten die Rollen und Verantwortlichkeiten für den Prozess, seine Aktivitäten und Ergebnisse festgelegt.

Der von dem Unternehmen KISTERS verwendete Softwareentwicklungsprozess ist kompatibel zum V-Modell gestaltet. Hierzu wurde das generische V-Modell entsprechend den Bedürfnissen von KISTERS mithilfe des Tailoring angepasst und ausgestaltet. Das auf KISTERS angepasste V-Modell lässt sich in vier Sub-

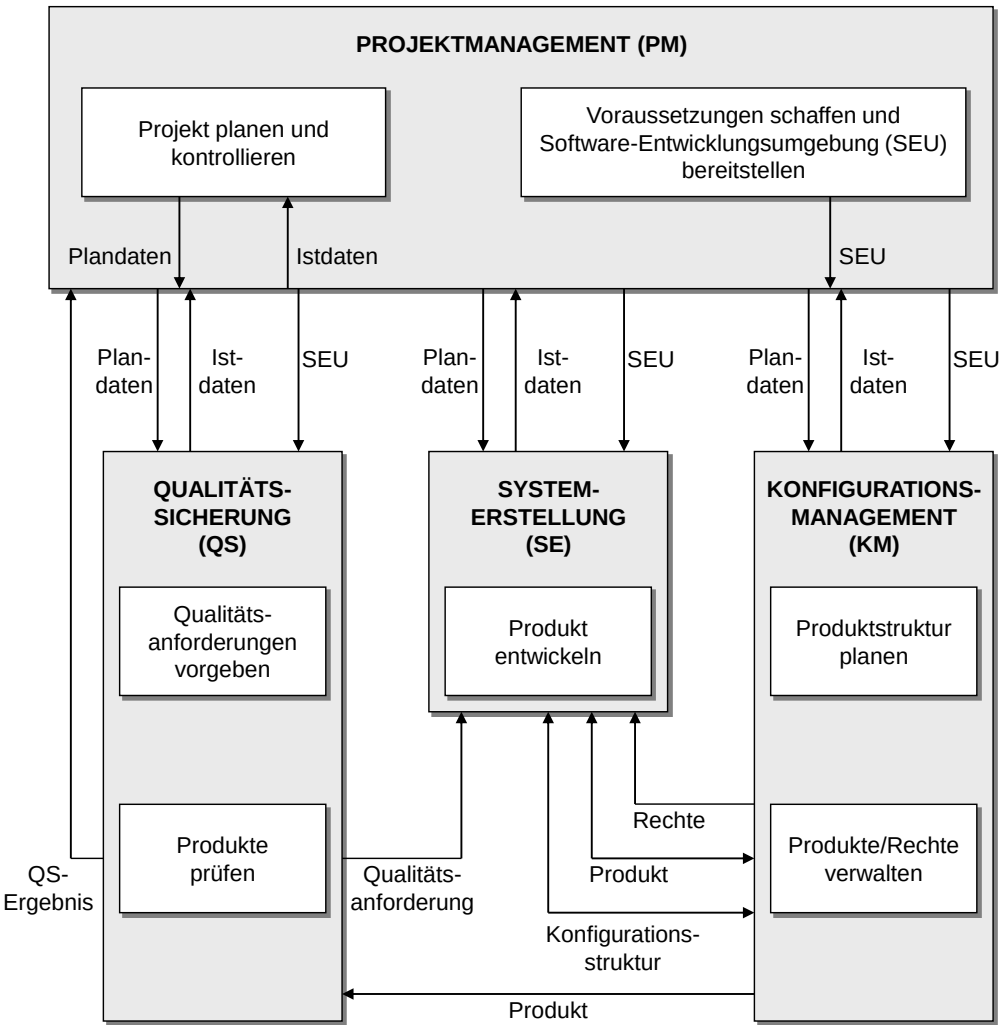


Abbildung 3.1: V-Modell 97 (nach [Drö98])

prozesse aufteilen (siehe Abb. 3.1): Projektmanagement (**PM**), Systemerstellung (**SE**), Qualitätssicherung (**QS**) und Konfigurationsmanagement (**KM**). Die Anforderungen hinsichtlich des Detaillierungsgrades der Subprozesse und der Umfang der benötigten Dokumentation sind entsprechend der Projektgröße festgelegt [SSM06]. Als Umsetzungsstrategie für den Prozess wird den Empfehlungen des V-Modells [Drö98] folgend eine inkrementelle und iterative Vorgehensweise gewählt [KIS08]. Das Produkt wird in einer Folge von Release-Zyklen in inkrementeller Weise entwickelt (siehe 2.1.2 Vorgehens- und Prozessmodell). Die verschiedenen Release-Zyklen dürfen sich zeitlich hinsichtlich ihrer Tätigkeiten überschneiden. Innerhalb des Release-Zyklus werden die für die Softwareentwicklung typischen Tätigkeiten in iterativer Art und Weise in unterschiedlicher Intensität durchgeführt. Hierzu untergliedert KISTERS die Entwicklung in die Phasen: Planung, Entwurf & Architektur, Implementierung, Rollout-Phase und Support-Phase. Die in den verschiedenen Phasen erstellten Dokumente werden im Konfigurationsmanagement verwaltet und von der Qualitätssicherung fort-

laufend geprüft. Hierzu wird schon während der Implementierungsphase der erstellte Quelltext kontinuierlich für das geplante Release integriert.

3.2 KISTERS-Bugzilla

Die Umsetzung des Softwareentwicklungs- und Wartungsprozesses wird bei KISTERS durch die Verwendung einer individuell angepassten Bugzilla-Installation unterstützt. Die Software Bugzilla ist ein Change-Request-Management-System (vgl. 2.1 Softwareentwicklungs- und Wartungsprozesse) aus dem Open-Source Bereich, welches zur Verwaltung von Änderungswünschen und Fehlermeldungen dient [Buga]. Bei KISTERS wird Bugzilla zur Erfassung der Anforderungen und zur Dokumentation des Ablaufs der Bearbeitung der Anforderungen genutzt. Hierzu erlaubt Bugzilla das Einstellen neuer Anforderungen, die Suche innerhalb der Anforderungen, die Verfolgung von Anforderungen im Prozess, und die Bearbeitung der Anforderungen selbst. Die in Bugzilla enthaltenen Anforderungen werden als Datenquelle zur Planung der Produkte, der Releases und der Tasks verwendet.

Die in KISTERS-Prozess beschriebenen Anforderungen werden auf die Einträge innerhalb des KISTERS-Bugzilla abgebildet. Diese Einträge des KISTERS-Bugzillas werden neutral als **Cases** bezeichnet und sind die verwalteten Change-Requests eines Change-Request-Management-Systems.

Die eingehenden Anforderungen der Softwareentwicklungs- und Wartungsprozesse werden in KISTERS-Bugzilla erfasst. Ein Case ist folglich das Ergebnis des Prozesses „Anforderungserfassung“. Hierzu wird die eingehende Anforderung als Case (vgl. [Sch06]) in der Software Bugzilla dokumentiert. Der Case entspricht somit der „erfassten Anforderung". Bei der Dokumentation einer eingehenden Anforderung als Case wird dieser zusätzlich zu seiner Beschreibung um einen kurzen Titel ergänzt.

Um zwischen den verschiedenen eingehenden Anforderungen zu unterscheiden, werden die Cases in **Bugs**, **Enhancements** und **SupportCalls** unterteilt. Eine eingehende Anforderung zur Behebung eines Softwarefehlers wird als Bug gekennzeichnet. Mit Enhancements werden Cases bezeichnet, welche die bestehende Funktionalität der Software erweitern. Ein SupportCall beschreibt eine Nutzeranfrage und beinhaltet eventuell die Änderung bestehender Funktionalität der Software (z.B. Konfiguration). Die Unterscheidung für die einzelne Anforderung wird im Case durch den **Schweregrad (engl. severity)** dokumentiert.

Neben der Art der Anforderung dokumentiert der Schweregrad auch, in welchen Umfang die Anforderung den Kunden betrifft (siehe Tab. 3.1), wie zum Beispiel die erwartete Nützlichkeit einer Erweiterung oder das Ausmaß der Störung von Geschäftsprozessen des Kunden aufgrund eines Software-Defekts. Die **Priorität** hingegen beschreibt die Bedeutung oder Dringlichkeit der Anfrage aus der

Severity	Bug	Enhancement	SupportCall
1	B1_Critical	E1_EnhanceCritical	S1_Critical
	Tool unusable, destructive	Tool not “fit for purpose“	Customer completely stuck
2	B2_Major	E2_EnhanceOblig	S2_Major
	Tool partially unusable	Tool incomplete	Cust. severely constrained
3	B3_Normal	E3_EnhanceNormal	S3_Normal
	Tool largely usable	Valuable enhancement	Cust. slightly constrained
4	B4_Minor	E4_EnhanceNice	S4_Minor
	Minor cosmetics	Just nice to have	Completely uncritical

Tabelle 3.1: KISTERS Schweregrad (nach [Sch08])

Priority	Product	Support
1	Needs immediate action, fix before you go home!	Resolve before you go home!
2	Fix for next patch to customer	Get back to customer in very short time (~1bd)
3	Fix for next regular release if possible	Get back to customer in short time (~1bw)
4	Fix when no higher priority open	Don´t forget to get back to customer

Tabelle 3.2: KISTERS Priorität (nach [Sch08])

Sicht von KISTERS (siehe Tab. 3.2). Die Cases mit der höchsten Priorität P1 beschreiben Notfälle und sind sofort zu bearbeiten, d.h. der Case durchläuft die weiteren Subprozesse in reduzierter Form.

Zur Identifizierung der Cases wird ein eindeutiger Bezeichner hinzugefügt. Die Beziehung zwischen dem Case und den betroffenen Softwareprodukten wird durch den Produktnamen, die Komponente und die Version erfasst. Zur Kommunikation mit dem Kunden werden der Kundenname und dessen Kontaktinformationen mit verwaltet.

Die verschiedenen Verantwortlichkeiten bei der Bearbeitung des Cases innerhalb des Prozess werden in unterschiedlichen Feldern, wie zum Beispiel „`reporter`“ oder „`assignee`“, dokumentiert. Der aktuelle Zustand der Bearbeitung der Anforderung wird im Zustandsfeld fortgeschrieben. Hierzu definiert KISTERS verschiedene Zustände und die erlaubten Zustandsübergänge in Form eines Work-

from \ to	NEW	REOPENED	ASSIGNED	WAITCUSTINFO	PROCESSING	RESOLVED	VERIFIED	SHIPPED	CLOSED
NEW	I		X	X	X	X			S
REOPENED	X		X	X	X	X			S
ASSIGNED	X			X	X	X			S
WAITCUSTINFO	X		X		X	X			S
PROCESSING	X		X	X		X			S
RESOLVED		X					X	X	X
VERIFIED		X						X	X
SHIPPED		X							X
CLOSED		X							F

I = Initial Status F = Final Status

X = Transition possible for all cases

S = Transition possible only for support cases (severity S*)

Tabelle 3.3: KISTERS Workflow (nach [Sch08]))

flows (siehe Tab. 3.3), die dazugehörige Semantik ist in der entsprechenden Arbeitsanleitung hinterlegt (vgl. [Sch06]).

Die erfassten Anforderungen werden auf den Zustand NEW abgebildet. Infolge der Anforderungsanalyse wechseln abgelehnte und zurückgestellte Anforderungen in den Zustand RESOLVED mir Resolution INVALID, WONTFIX oder LATER. Die akzeptierten Anforderungen werden im Planungsprozess um ihr Bearbeitungsdatum (als `targetMilestone`) und gegebenenfalls um eine Deadline ergänzt. Zudem wird der verantwortliche Bearbeiter über das Assignee-Feld zugewiesen. Die erfolgreiche Bearbeitung einer Anforderung im Subprozess „Task-Bearbeitung“ wird mit dem Zustandswechsel auf RESOLVED mit Resolution FIXED abgebildet. Die im Testprozess erfolgreich geprüften Anforderungen werden zu getesteten Anforderungen, deren Zustand im Case mit VERIFIED beschrieben wird. Infolge eines nicht erfolgreichen Tests wird die bearbeitete Anforderung wieder zu einer erfassten Anforderung. Jedoch wird ihr Zustand im Case zur besseren Unterscheidung mit REOPENED charakterisiert. Die erfolgreiche Auslieferung der Anforderung wird durch den Zustandsübergang nach CLOSED gekennzeichnet. Die Software KISTERS-Bugzilla definiert im Workflow weitere Zustände um insbesondere die Bearbeitung des Cases im Task-Bearbeitungsprozess zu beschreiben und die Kommunikation mit dem Kunden explizit darzulegen.

Eine Unterscheidung zwischen den abgelehnten, zurückgestellten und bearbeiteten Anforderungen anhand des Zustands RESOLVED ist nicht möglich. Hierzu wird die Art der Bearbeitung des Cases über das Feld Resolution angezeigt. Die zurückgestellten Anforderungen werden durch die Resolution LATER, REMIND oder WORKSFORME gekennzeichnet. Die Resolutionen INVALID und WONTFIX dokumentieren abgelehnte Anforderungen. Mit der DUPLICATE Resolution wird dokumentiert, dass der Case bereits existiert. Die Bearbeitung der Anforderung findet im ursprünglichen Case statt.

Viele Bearbeitungsschritte des Cases werden über Kommentare dokumentiert. Dokumente und Dateien werden dem Case als Attachement hinzugefügt. Zusätzlich lassen sich projektspezifische Informationen mithilfe von Flags dokumentieren.

3.3 BugzillaMetrics

BugzillaMetrics ist ein web-basierendes Werkzeug, welches die Analyse und Evaluation von Change-Request Daten unterstützt. Hierzu erlaubt es über sein Web-Frontend Anfragen zu spezifizieren und zu speichern. Mithilfe der Anfragen wird die Berechnung angestoßen und ihre Ergebnisse lassen sich in verschiedenen Darstellungsformen ausgeben.

3.3.1 Spezifikation von Metriken

Die Spezifikation der Metrik beschreibt, wie die Informationen des Change-Request-Management-Systems ausgewertet werden. Hierzu wird von den konkreten Informationen des einzelnen Change-Request mithilfe vordefinierter Auswertungsverfahren abstrahiert. Dabei berücksichtigt BugzillaMetrics den zeitlichen Verlauf der Bearbeitung des Change-Requests anhand dessen gespeicherter Historie. Statt dem Begriff des Change-Request wird in BugzillaMetrics üblicherweise der Begriff Case verwendet.

Im „**base filter**“ der Spezifikation wird die Menge der betrachteten Cases festgelegt, so dass beispielsweise nur ausgewählte Produkte oder ausschließlich hochpriorisierte SupportCalls betrachtet werden. Die „**evaluation time period**“ legt den Betrachtungszeitraum der Auswertung fest, in dem Start- und Endzeitpunkt vorgegeben werden. Die „**evaluation period granularity**“ unterteilt den Betrachtungszeitraum zur Auswertung anhand von Zeitintervallen. Hierzu kann sowohl auf die vordefinierten Zeitintervalle wie Jahr, Monat oder Tag, als auch auf individuell festgelegte Intervalle zurückgegriffen werden. Die „**grouping parameters**“ unterteilen die Menge der Cases in unterschiedliche Partitionen in der Auswertung.

Die Merkmalsausprägung eines Cases wird in BugzillaMetrics durch Event- und State-Filter beschrieben. Mithilfe der **Event-Filter** wird festgelegt, welche Ereignisse während der Berechnung betrachtet werden. Der am häufigsten verwendete Event-Filter ist der Transitionsfilter, welcher beispielsweise Zustandsübergänge oder Änderungen des Bearbeiters erfasst. Ebenso lassen sich die in Bugzilla neu eingestellten Cases (`created`) oder alle Cases am Ende eines Betrachtungszeitraums (`endOfTimeIntervall`) ermitteln. Die **State-Filter** prüfen, ob ein Case bestimmte Merkmalsausprägungen aufweist. Die Event- und State-Filter lassen sich mithilfe der booleschen Operationen zu komplexeren Filtern verknüpfen.

Für die zuvor festgelegten Auswertungszeiträume wird für jeden Case anhand der definierten Berechnungsvorschrift ein „**Case Value**“ bestimmt. Die Spezifikation der Berechnungsvorschrift erfolgt auf Basis eines „**Value Calculators**“. BugzillaMetrics besitzt die folgenden vier verschiedenen Value Calculators:

- **CountEventsCalculator**
  Dieser Value Calculator zählt das Auftreten der in der Berechnungsvorschrift spezifizierten Ereignisse für einen Case. Dem Case wird entsprechend des spezifizierten Gewichts (engl. Weight) ein numerischer Wert zugewiesen. Mithilfe von CountEventsCalculators lassen sich eine Vielzahl von Metriken spezifizieren: die Anzahl der neu erstellten Cases eines Zeitraums, die Anzahl der Cases mit Nacharbeiten in einer Periode, die Anzahl der Cases, die ihre Deadline verletzt haben, der verbleibende Arbeitsaufwand aller Cases im Zustand NEW, oder die Schätzgüte hinsichtlich des Aufwands der Cases bei Zustandsübergang nach RESOLVED.

- **CountUntilCalculator**
  Der CountUntilCalculator zählt das Auftreten der in der Count-Komponente mithilfe eines Event-Filters beschriebenen Ereignisse bis zum Auftreten des Until-Ereignisses. Mittels des CountUntilCalculator lässt sich beispielsweise die Anzahl der Zustandswechsel bis zum Zustandsübergang nach RESOLVED bestimmen. Ebenso lässt sich prüfen, wie häufig der Schweregrad des Cases bis zum Ende seiner Bearbeitung eskaliert wurde.

- **IntervalLengthCalculator**
  Die Zeit in Tagen zwischen zwei Ereignissen wird mithilfe des IntervalLengthCalculators bestimmt. Hierzu werden ein Startereignis und ein Endereignis spezifiziert. Zusätzlich lässt sich noch spezifizieren, wann ein Case Value erzeugt wird. Dies ist entweder jedes Mal, wenn das Startereignis auftritt, oder nur beim ersten beziehungsweise dem letzten Auftreten des Startereignisses der Fall. Man nutzt den IntervalLengthCalculator unter anderem um die Zeitspanne zwischen der Erstellung eines Cases und seiner erfolgreichen Bearbeitung zu bestimmen.

- **ResidenceTimeCalculator**
  Für jedes der festgelegten Ereignisse wird die Verweildauer in Tagen eines Cases in einem bestimmten Zustand bestimmt. Analog zum IntervalLengthCalculator lässt sich spezifizieren, wann ein Case Value generiert wird. Die Verwendung des ResidenceTimeCalculators erlaubt es, die Zeit zu ermitteln, in der ein Case im Zustand PROCESSING verweilt hat, bis dieser als erfolgreich bearbeitet gekennzeichnet wurde.

Der IntervalLengthCalculator und ResidenceTimeCalculator sind zwei unterschiedliche Konzepte, die auseinander zu halten sind. Der IntervalLengthCalculator berechnet die Zeitspanne von einem Startereignis bis zu einem Endereignis. Der ResidenceTimeCalculator berechnet bei jedem Auftreten eines bestimmten Ereignisses die Verweildauer in einem bestimmten Zustand. Die Zeitspanne lässt sich eindeutig einem Start- und Endereignis zuweisen. Die Verweildauer wird jedoch jedem auftretenden Ereignis zugewiesen.

Für die ermittelten „Case Values“ der einzelnen Auswertungszeiträume lassen sich verschiedene Auswertungen spezifizieren. Man unterscheidet zwischen der Detailauswertung und der Auswertungsspezifikation. In der **Detailauswertung** werden für die zuvor festgelegten Value Calculators die Menge der im Auswertungszeitraum ermittelten Case Values und die dazugehörigen Cases-Identifier gespeichert. Die **Auswertungsvorschrift** spezifiziert, wie die ermittelten Werte aus der Berechnungsvorschrift in die Auswertung zu übernehmen sind. Für eine Berechnungsvorschrift sind dies beispielsweise

- die Summe aller Case Values eines Auswertungszeitraums,
- die Anzahl der Case Values,
- die Anzahl der betrachteten Cases,
- der größte Case Value des Beobachtungszeitraums oder
- der Median der Case Values im Betrachtungszeitraum.

Die aus den Berechnungsvorschiften übernommenen Werte lassen sich in der Auswertungsvorschrift kombinieren und transformieren.

3.3.2 Web-Frontend

Im Web-Frontend sind einige Anfragen vordefiniert, um den Einstieg in die Benutzung von BugzillaMetrics zu erleichtern. Neben den vordefinierten Anfragen wird die Spezifikation eigener Metriken durch den Benutzer unterstützt. Die selbst spezifizierten Metriken lassen sich speichern und weiter bearbeiten. Die vordefinierten Anfragen können als Ausgangspunkt für individuell angepasste Metriken verwendet werden. Hierzu wählt der Benutzer eine geeignete Berechnungsvorschrift und spezifiziert die gewünschte Metrik. Die so erzeugte

Metrik wird in BugzillaMetrics als ein XML-Dokument verwaltet, welches die Spezifikation der Metrik in einer deklarativen Sprache abbildet. Dem erfahrenen Nutzer wird der Zugriff auf dieses Dokument erlaubt, und gestattet weitere mächtigere Möglichkeiten zur Spezifikation von Metriken. So können vor allem die Berechnungen gezielter gesteuert werden, wie beispielsweise die Steuerung des IntervallFilters. Zudem bietet die deklarative Beschreibung den Zugriff auf Event-Filter und Manipulationsoptionen, die nicht im Web-Frontend definiert sind. Jedoch ist eine anschließende Manipulation über die Assistenten zur Festlegung der Berechnungsvorschrift nicht mehr möglich.

Nach erfolgreicher Spezifikation der Metrik lässt sich ihr Berechnungsvorgang anstoßen. Zuerst wird die syntaktische Korrektheit des XML-Dokuments, welches die Berechnungsvorschrift enthält, geprüft. Das Ergebnis der anschließenden Berechnung wird zur graphischen Analyse als Chart ausgegeben. Dieser kann wahlweise in Linie oder in gestapelter Form dargestellt werden. Neben der graphischen Darstellung der Berechnung erlaubt BugzillaMetrics zur weiteren Analyse die Ausgabe in tabellarischer Form. Das Ergebnis der Berechnung lässt sich ebenfalls wahlweise als csv-Tabelle (comma-seperate-values) oder Excel-Tabelle exportieren. Die mächtigste Form der Ausgabe ist ein XML-Dokument, welches die Spezifikation der Metrik, die Ergebnisse und die Detailinformationen der Berechnungen beinhaltet.

3.3.3 Integration von Bugzilla in den Messungsprozess (nach ISO15939)

Zur Analyse von Prozessen mithilfe von BugzillaMetrics bietet sich ein Vorgehen entsprechend dem in der ISO 15939 beschriebenen Measurement Prozesses (dt. Messungsprozess) an. Der Measurement Prozess adaptiert hierzu den Plan-Do-Check-Act Zyklus, wobei die Schnittstellen zur Aktion nur angedeutet werden. Hierzu werden die einzelnen Phasen des Plan-Do-Check-Act Zyklus auf die Aktivitäten des Prozesses abgebildet. Im Messungsprozess werden folgende Aktivitäten beschrieben und durch Aufgaben (engl. Tasks) weiter detailliert:

- Aufbau und Unterhaltung des Engagement (engl. Commitment) zur Messung
- Planung des Messungsprozesses
- Durchführung des Messungsprozesses
- Evaluation des Messungsprozesses

Die Aktivität Aufbau und Unterhaltung des Engagements zur Messung fordert die Unterstützung für den Messungsprozess ein und reserviert die benötigten Ressourcen. Die Verknüpfungen zur Act-Phase sind nur angedeutet durch die Schnittstellen zu den technischen und den Management-Prozessen. In der Plan-

Phase wird zuerst das Ziel der Messung bestimmt. Die Do-Phase ist die konkrete Messung, deren Ergebnisse in der Check-Phase analysiert werden. Die Analyse bezieht sich sowohl auf die Ergebnisse der Messung, als auch auf den Messungsprozess und Messvorgang selbst. Die Erkenntnisse aus der Check-Phase werden in der Act-Phase in konkrete Aktionen umgesetzt. Der Plan-Do-Check-Act Zyklus wird zyklisch ausgeführt, wodurch insbesondere in den frühen Iterationen das Verhalten des Messgegenstandes durch reproduzierbare Ergebnisse stabilisiert werden soll (vgl. Deming-Cycle [ED07]). Die Ergebnisse der einzelnen Phasen können in den folgenden Iterationen des Zyklus wiederverwendet werden, wie beispielsweise die definierten Geschäftsziele und die daraus resultierenden Fragestellungen oder die Metriken zur Messung.

Im Measurement-Prozess sind Rücksprünge zwischen den einzelnen Phasen nicht definiert. Wurde in der Plan-Phase eine Fragestellung definiert, die sich in der Do-Phase als Metrik nicht operationalisieren lässt, muss der Grund dafür in der Check-Phase analysiert und dokumentiert werden. In der folgenden Iteration kann nun unter Berücksichtigung der Erkenntnisse der Check-Phase beispielsweise eine alternative Fragestellung entwickelt oder die Metrik zur Messung angepasst werden. Indem keine Rücksprünge innerhalb des Zyklus definiert sind, müssen die Ergebnisse der Check-Phase am Ende des Zyklus evaluiert werden. Die fortlaufende Evaluation erlaubt eine kontinuierliche Verbesserung des Messungsprozesses und die Gefahr einer Manipulation der Messergebnisse durch eine Veränderung der Messvorschrift wird abgemildert. Es ist nicht erlaubt zwischen Messung und Analyse ohne Dokumentation zu iterieren, bis die manipulierten Messergebnisse vorliegen. Somit lässt sich anhand der Dokumentation eine mögliche Manipulation nachvollziehen.

Bei der Analyse von Prozessen unterstützt die Software BugzillaMetrics als Messungs- und Auswertungssoftware insbesondere die Phasen Do und Check. Die Do-Phase wird in BugzillaMetrics durch die Definition von Metriken zur Messung innerhalb des Datenbestandes des Change-Request-Management Systems unterstützt. Die Ergebnisse lassen sich für die Check-Phase visualisieren oder für andere Analyse-Software bereitstellen. Mithilfe der Darstellung in tabellarischer Form ist eine erste Evaluation möglich. Einen tiefergehenden Einblick ermöglichen jedoch erst die Detailinformationen. Aufgrund der automatisierten Auswertung können kurze Iterationszyklen gewährleistet werden.

4 KISTERS Qualitätsmodell

Inhalt

Im Folgenden wird ein Qualitätsmodell zur Analyse von Prozessqualitäten des Softwareentwicklungs- und Wartungsprozesses der Firma KISTERS entwickelt. Hierzu werden entsprechend dem vorgestellten bidirektionalen Qualitätsmodell die beiden verschiedenen Sichtweisen ausgeprägt (vgl. 2.3.1 Bidirektionale Qualitätsmodelle). Zuerst werden mögliche Informationsbedürfnisse identifiziert und deren Beziehung untereinander durch Qualitätseigenschaften dargestellt. Dabei sind die KISTERS-spezifischen Prozesse zu berücksichtigen. Anschließend wird die Sichtweise der Qualitätsmerkmale und Qualitätsindikatoren modelliert. Die Qualitätsmerkmale der Prozesse werden mithilfe der aufgezeichneten Change-Request Daten in Bugzilla erfasst.

4.1 Informationsbedürfnisse

Die Qualitätseigenschaften und ihre Beziehungen untereinander werden ausgehend von den **Informationsbedürfnissen** modelliert. Hierzu werden die Informationsbedürfnisse aus den unterschiedlichen Zielen und Zielvorstellungen der Organisation abgeleitet (vgl. [SL08]). Diese Sichtweise wird im Allgemeinen um Geschäftsziele, rechtliche Vorschriften sowie Produkt- und/oder Projektziele ergänzt (vgl. [ISO 15939]). Hierzu werden im Folgenden aus der Literatur verschiedene Ziele und Zielvorstellungen für die Softwareentwicklungs- und Wartungsprozesse abgeleitet und um KISTERS spezifische Zielvorgaben erweitert.

Das Capability Maturity Model (CMM) und **Capability Maturity Model Integrated** (CMMI) ist ein Modell zur Bewertung der Prozessreife einer Organisation hinsichtlich ihrer Systementwicklungsprozesse und kann Voraussetzung zur Bewerbung um Aufträge sein. Die Bewertung der Prozessreife erfolgt durch ein Assessment, infolge dessen der Reifegrad bestimmt wird. Zur Bewertung der Prozessreife wird der gesamte Prozess in verschiedene Prozessbereiche unterteilt. Für diese Prozessbereiche wird dann geprüft, ob die vorgeschriebenen Tätigkeiten ausgeführt werden beziehungsweise relevant/anwendbar sind (vgl. [KJ08]). Die Reifegrade sind in aufsteigender Folge von 1 („initial“) bis 5 („optimizing“)

angeordnet. Mit zunehmendem Reifegrad werden die Anforderungen verschärft und die Anzahl der betrachteten Prozessbereiche nimmt zu. Das CMMI wurde entwickelt um unabhängig vom konkreten Prozessmodell den Reifegrad zu bestimmen und mögliche Schwächen zu identifizieren. Hierzu werden in den Prozessbereichen Vorgehensweisen und Tätigkeiten zusammengefasst, die nach den Erfahrungen notwendig sind, um Entwicklungsprojekte erfolgreich durchzuführen (vgl. [LL07]). Neben dem CMMI hat sich die ISO 15504 / SPICE im Software-Engineering zur Bewertung von Softwareentwicklungsprozessen im europäischen Raum etabliert. Durch dieses Bewertungsverfahren versucht man die verschiedenen Verfahren zur Bewertung zu harmonisieren, und ergänzt den Standard zusätzlich um ein Modell zur Prozessverbesserung.

Aus der betriebswirtschaftlichen Sicht werden Prozesse mithilfe von **Kennzahlensystemen** analysiert (siehe [Küt07]). Der Schwerpunkt dieser Betrachtungsweise liegt im Controlling. Hierzu werden hauptsächlich Kennzahlen zu den Aspekten Kosten, Zeit und Qualität erfasst. Die einzelnen Kennzahlen werden differenziert, entsprechend des gemessenen Merkmals betrachtet und den einzelnen organisatorischen Einheiten zugeordnet. Die Softwareentwicklungs- und Wartungsprozesse werden in der Literatur nicht diskutiert. Jedoch haben sich verschiedene Kennzahlensysteme etabliert, deren Fokus auf den *unterstützenden Prozessen* (vgl. 2.1.2 Vorgehens- und Prozessmodell), wie dem Management der Anforderungen, und der Bereitstellung von IT-Diensten liegt.

Die unterstützenden Prozesse, wie Änderungsmanagement oder Konfigurationsmanagement, werden im Rahmen des IT-Service-Managements betrachtet. Die **IT Infrastructure Library** (ITIL) definiert verschiedene good/best practices, von denen verschiedene Qualitätsziele abgeleitet werden können (vgl. [vB04]). Beispielsweise lassen sich die Ziele der ITIL-Prozesse „Incident-Management“ und „Problem-Management“ auf die Bearbeitung der Anfragen übertragen.

Bei der Modellierung von Qualitätseigenschaften sind insbesondere die **KISTERS** spezifischen Zielvorgaben zu berücksichtigen. Für das Qualitätsmanagementsystem werden im Qualitätsmanagement-Handbuch **generische Ziele** für die Unternehmensprozesse in Form von Qualitätsgrundsätzen formuliert (siehe [SSM06]). Als explizite Verbesserungsziele für die Software-Entwicklung werden als Qualitätsziele die „Termintreue“ und „Planungsgenauigkeit“ operationalisiert (siehe [KIS09]). Die „**Termintreue**“ der Projektabwicklung soll durch verschiedene Maßnahmen, wie eine effiziente Terminplanung und Steuerung, die Optimierung des Mitarbeitereinsatzes sowie Maßnahmen des Projekt-Controllings, verbessert werden. Zur Ermittlung der Kennzahl wird die tatsächliche und geplante Projektlaufzeit genutzt. Die „**Planungsgenauigkeit**“ soll mithilfe der Maßnahmen „Angebotskalkulation optimieren“ und „Mitarbeiterschulungen“ verbessert werden. Als Kennzahl wird das prozentuale Verhältnis zwischen tatsächlichen und geplanten Arbeitstagen verwendet. Zur Ermittlung dieser Qualitätsziele werden die entsprechenden Projektabschlussberichte als Datenquelle verwendet. Die „**Kundenzufriedenheit**“ wird anhand von Fragebögen ermittelt und soll durch Schulungsmaßnahmen der Anwender verbessert werden.

4.2 Qualitätseigenschaften

Ausgehend von den verschiedenen Zielen und Zielvorgaben für den Softwareentwicklungs- und Wartungsprozess lassen sich die Qualitätseigenschaften und ihre Beziehungen untereinander modellieren. Im Folgenden werden ausgewählte Informationsbedürfnisse der Firma KISTERS sowie die daraus abgeleiteten Fragestellungen zu den Qualitätseigenschaften zusammengefasst und erläutert.

Die Qualitätseigenschaft **Management der Anforderungen** betrachtet den Umgang mit Änderungsanfragen und Fehlermeldungen. Das Ziel ist es, ein gemeinsames Verständnis zwischen dem Kunden und dem Auftragnehmer bezüglich der Anforderungen zu erzeugen. Dazu müssen neben den technischen Anforderungen, wie der Spezifikation der Änderung, insbesondere die nichttechnischen Aspekte, wie Dringlichkeit oder Deadlines, geklärt werden. Die erfassten Anforderungen bilden die Grundlage für die Planung, Schätzung und Durchführung des Softwareprojekts (vgl. [Wal01]). Hieraus lassen sich die folgenden Fragestellungen ableiten:

- Findet in der Betrachtungsperiode vordringlich Weiterentwicklung oder Fehlerbehebung statt?
- Wie hoch ist der Anteil der Projektentwicklung am Entwicklungsaufkommen?
- Wie zuverlässig ist eine erste Klassifikation der Cases hinsichtlich ihres Typs?
- Kann parallele oder doppelte Arbeit vermieden werden?

Die Planungstätigkeiten des Softwareentwicklungs- und Wartungsprozesses sind von elementarer Bedeutung für die Realisierung von IT-Produkten und IT-Projekten. Im Rahmen der **Planung** müssen *angemessene* Pläne zur Umsetzung der Anforderungen entwickelt werden. Hierzu werden der Umfang und Aufwand für die zu leistenden Arbeiten mit hinreichender Genauigkeit geschätzt. Die Ergebnisse der Planung müssen dokumentiert werden, damit die Umsetzung entsprechend der freigegebenen Planung erfolgt (vgl. [Wal01]). Die Planung wird hinsichtlich der Aspekte Zeit und Aufwand betrachtet:

- Werden Termine geplant?
- Werden die geplanten Termine eingehalten?
- Wird der benötigte Zeitaufwand zur Umsetzung der Cases geplant?
- Ist die Schätzung des Zeitaufwands hinreichend genau?
- Wie groß ist die Planungsreichweite?

Die Qualitätseigenschaft **Überwachung und Verfolgung des Fortschritts** beschreibt in welchem Umfang der tatsächliche Fortschritt bei der Umsetzung der Anforderungen sichtbar gemacht wird. Dies ist die Voraussetzung um durch den Vergleich von geplanten und erreichten Zielgrößen relevante Abweichungen zu erkennen und gegebenenfalls Korrekturmaßnahmen einzuleiten (vgl. [Wal01]).

- Wird der Entwicklungsfortschritt der Bearbeitung dokumentiert?
- Wie ist die Granularität der Cases?

Die Informationsbedürfnisse hinsichtlich des zeitlichen Aspekts der Bearbeitungsdauer der einzelnen Anforderungen werden in der Qualitätseigenschaft **Lösungsgeschwindigkeit** zusammengefasst. Die Bearbeitungsdauer wird für die verschiedenen Phasen des Softwareentwicklungsprozesses wie folgt aufgeteilt:

- Wie lange dauert es bis zum Abschluss der Bearbeitung?
  Bearbeitungszeit (time to solution)
- Wie lange dauert es bis zum Schließen des Cases?
  Lösungszeit (time to customer accept)

Im Rahmen der **Software-Qualitätssicherung** wird die im Softwareentwicklungs- und Wartungsprozess erzeugte Qualität der bearbeiteten Artefakte sichtbar gemacht. Diese umfasst neben den gefundenen Software-Fehlern die ungeplanten Nacharbeiten aufgrund einer nicht erfolgreichen Umsetzung von Anforderungen.

- In welchem Umfang fallen ungeplante Nacharbeiten an?
- Wie lange ist die Bearbeitungsdauer der ungeplanten Nacharbeiten?
- Wie sensitiv ist die Fehlersuche?

Das **Software-Konfigurationsmanagement** überwacht den Zugriff auf die bearbeiteten Artefakte der Anforderungen und stellt die Konsistenz/Integrität während des gesamten Software-Lebenszyklus hindurch sicher. Die verschiedenen bearbeiteten Artefakte lassen sich zu jedem Zeitpunkt eindeutig identifizieren und ältere Zustände rekonstruieren. Zur systematischen Umsetzung von Änderungsanforderungen und Fehlermeldungen des Kunden erlaubt das Konfigurationsmanagement, die Software-Konfiguration des an den Kunden gelieferten Produkts eindeutig zu bestimmen.

- Wird der `targetMilestone` für den Case dokumentiert?
- Werden die umgesetzten Anforderungen mit `releaseNotes` dokumentiert?

In Softwareentwicklungs- und Wartungsprozessen müssen eine Vielzahl verschiedener Anspruchsgruppen (engl. *stakeholder*) koordiniert werden. Die Anspruchsgruppen umfassen im Allgemeinen die Entwickler, das Management sowie die

internen und externen Kunden. Die Effizienz der Koordinationsprozesse wird durch den Kommunikationsaufwand zwischen den Anspruchsberechtigten bestimmt. Insbesondere die Koordination zwischen dem Kunden und dem Unternehmen ist von entscheidender Bedeutung um die Kundenbedürfnisse erfolgreich in der Software umzusetzen. Zusammengefasst wird dies in der Qualitätseigenschaft **Koordination der beteiligten Anspruchsgruppen**.

- Wie hoch ist der Koordinationsaufwand bei der Bearbeitung der Cases?
- Wieviel Reibung entsteht in dem Koordinationsprozess mit dem Kunden?

Die SupportCall-Cases werden separat innerhalb der Qualitätseigenschaft **Supportprozess** betrachtet. Im Gegensatz zu den Änderungsanforderungen und Fehlermeldungen betreffen die SupportCalls den Kunden und nicht das Produkt. Die SupportCalls sind jedoch im Kontext des Wartungsprozesses als Tätigkeiten hinsichtlich des Softwareprodukts zu interpretieren.

- Wie viele SupportCalls können direkt gelöst werden?

Während der **Fire-Fighting Aktivitäten** steht die zeitnahe Umsetzung der Anforderungen im Vordergrund, wodurch der Softwareentwicklungs- und Wartungsprozess gegebenenfalls in verkürzter Form durchlaufen wird.

- Wie hoch ist der Anteil der Fire-Fighting Aktivitäten im Prozess?
- Wie lange dauert die Lösung der Cases hinsichtlich der Priorität P1?

Die **Kundenzufriedenheit** fasst in einer Qualitätseigenschaft die Anforderungen des Kunden an die Softwareentwicklungs- und Wartungsprozesse zusammen. Hierzu vergleicht der Kunde die tatsächlichen Ergebnisse des Prozesses mit seiner subjektiven Leistungserwartung. Da der Kunde im Allgemeinen nur eine begrenzte Sicht auf die Bearbeitung seiner Anforderungen besitzt, konzentriert man sich auf die Endresultate der Umsetzung.

- Wie zuverlässig ist der Software-Anbieter hinsichtlich der Termin-Absprachen?
- Ändert sich die Kommunikationsstruktur zum Kunden?

Das Beziehungsgeflecht der einzelnen Qualitätseigenschaften lässt sich anhand des nachfolgenden azyklischen Graphen anschaulich darstellen (siehe Abb. 4.1 und 4.2). Diese Darstellung wird um die in Kapitel 4.3 beschriebenen Qualitätsindikatoren ergänzt, so dass der Bezug zwischen den einzelnen Qualitätsindikatoren und den Qualitätseigenschaften nachvollziehbar wird.

Eine Gliederung der Qualitätsmerkmale hinsichtlich verschiedener Rollen im Softwareentwicklungs- und Wartungsprozess lässt sich anhand der in KISTERS-Bugzilla definierten Sichten ableiten. In KISTERS-Bugzilla werden vier verschiedene Ansichten definiert: Customer View, Developer View, QA View und

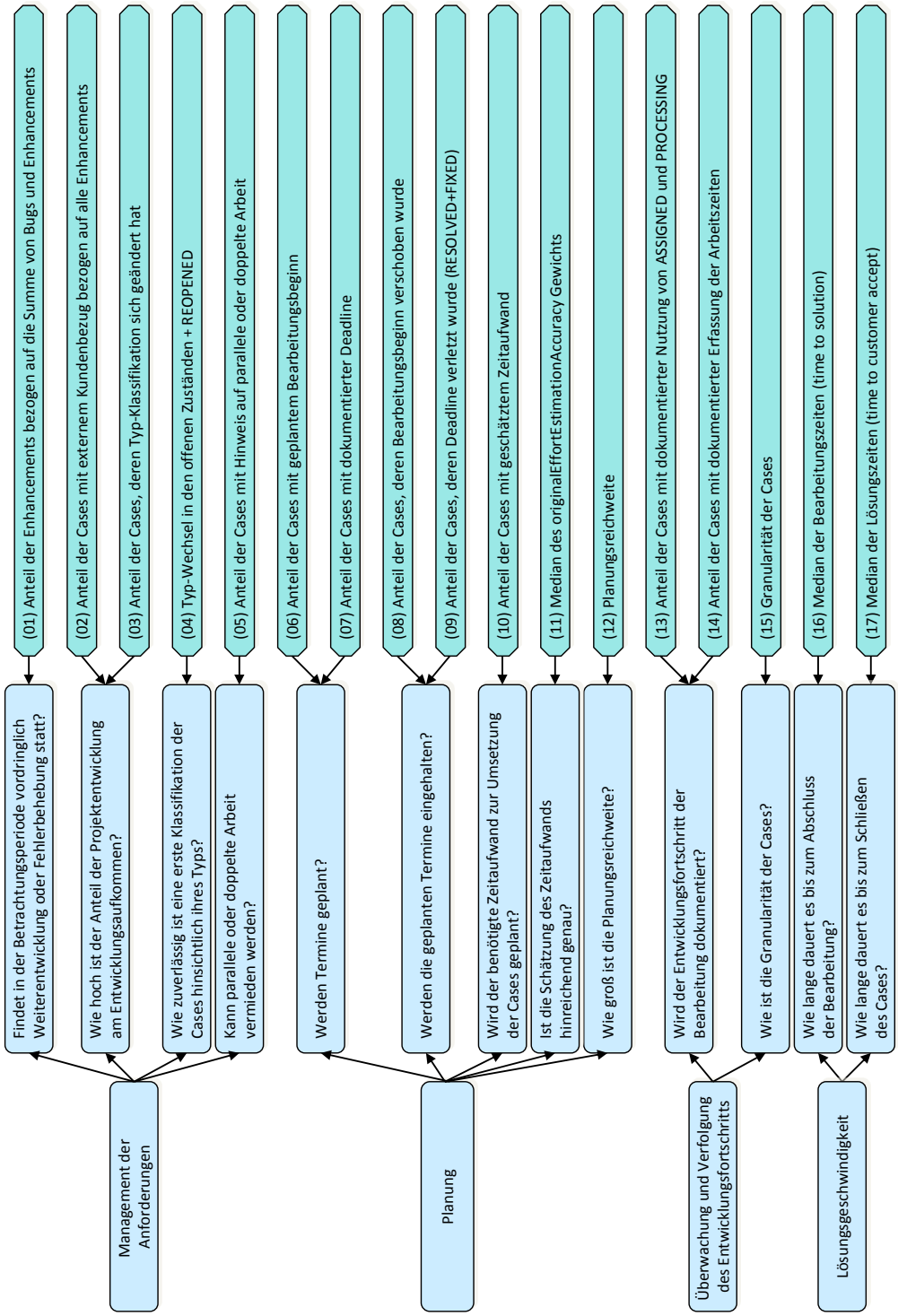


Abbildung 4.1: KISTERS bidirektionales Qualitätsmodell (Teil 1)

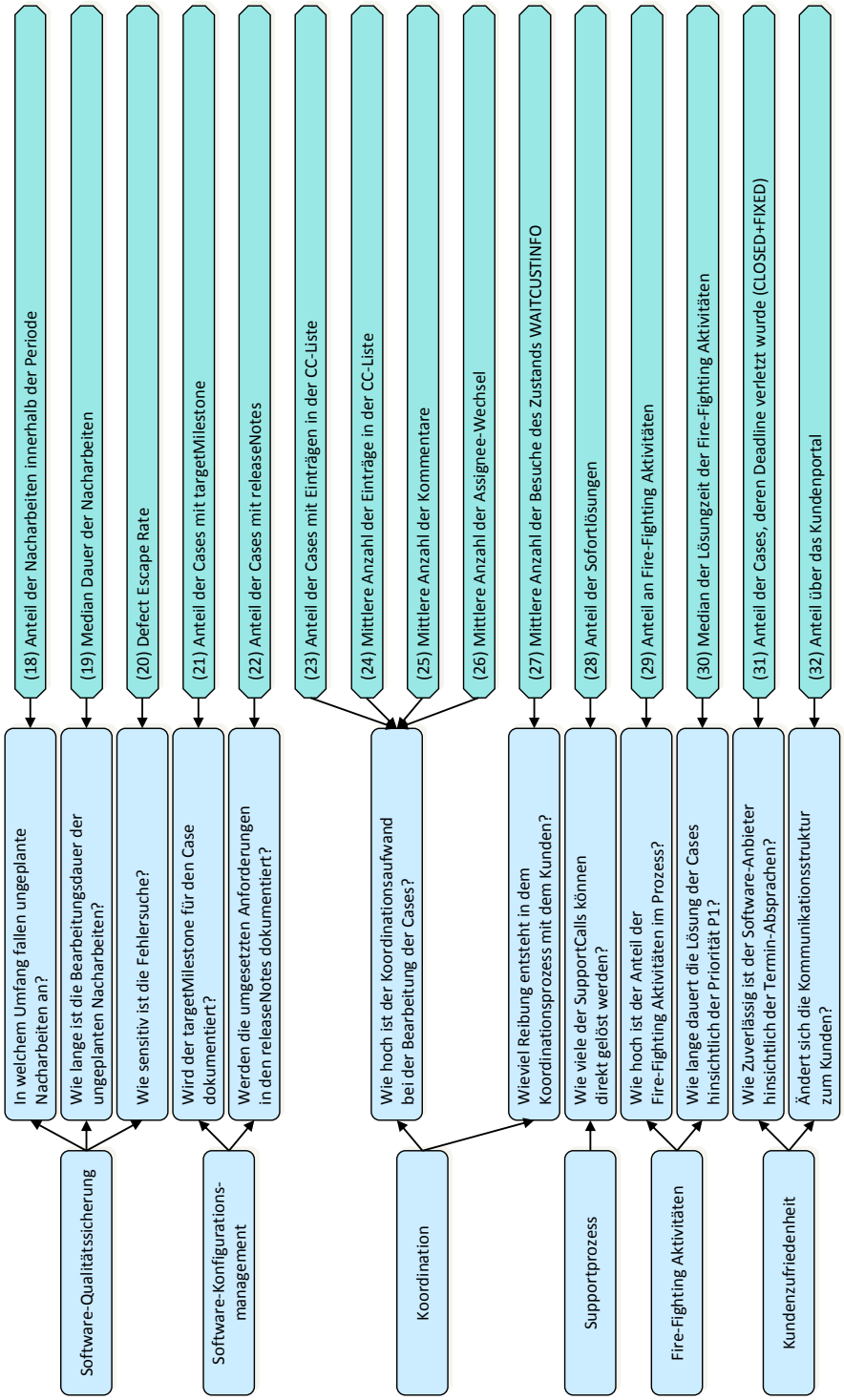


Abbildung 4.2: KISTERS bidirektionales Qualitätsmodell (Teil 2)

Product Overview. Diese Sichten gliedern den Entwicklungsprozess unter Berücksichtigung der Zustände. Hierzu werden die fein granularen Zustände zu Zustandsmengen aggregiert. Beispielsweise definiert der Customer View (dt. Kunden-Sicht) die drei Zustandsmengen ToDo (=NEW, ASSIGNED, RESOLVED), Under Construction (=PROCESSING, WAITCUSTINFO, RESOLVED) und Solved (=VERIFIED, SHIPPED, CLOSED). Die verschiedenen Sichten spiegeln die Informationsbedürfnisse einzelner Anspruchsgruppen wider und sind somit zur Definition von Qualitätseigenschaften geeignet. Jedoch überschneiden sich die Informationsbedürfnisse der verschiedenen Anspruchsgruppen, wodurch innerhalb der Qualitätseigenschaften ein kompliziertes Beziehungsgeflecht entsteht. Das oben vorgestellte Qualitätsmodell für KISTERS lässt sich durch eine unterschiedliche Gewichtung der Indexkennzahl im Modell (vgl. Kapitel 2.4) entsprechend den Informationsbedürfnissen für die verschiedenen Anspruchsgruppen ausprägen. Daher wird im Folgenden auf eine Untergliederung entsprechend der Rollen verzichtet.

4.3 Qualitätsindikatoren und Qualitätsmerkmale

Die Qualitätsmerkmale und Qualitätsindikatoren bilden die technische Stoßrichtung bei der Erstellung eines Qualitätsmodells. Nachdem mithilfe der Qualitätseigenschaften die Anforderungen an die Qualität des Softwareentwicklungs- und Wartungsprozesses modelliert wurden, lässt sich nun mithilfe der Qualitätsindikatoren der Einfluss der Qualitätsmerkmale auf die Qualitätseigenschaften darstellen. **Die Qualitätsmerkmale der Prozesse werden anhand der in KISTERS-Bugzilla aufgezeichneten Change-Request Daten von BugzillaMetrics erfasst.** Folglich lassen sich nur die in den Change-Requests sichtbar gemachten Qualitätsmerkmale erfassen. Im Folgenden wird mithilfe der Qualitätsindikatoren die Beziehung zwischen dem erfassten Qualitätsmerkmal und der modellierten Qualitätseigenschaften dargestellt.

Die Qualitätsindikatoren werden anhand des folgenden Schemas vorgestellt. Die Beschreibung beginnt mit dem **Titel** des Qualitätsindikators, der einen Ausblick auf das betrachtete Qualitätsmerkmal gibt, sowie einer **eindeutigen Identifikationsnummer**. Gefolgt von der **Definition** wie das betrachtete Qualitätsmerkmal zu erfassen ist. Abschließend wird der **Bezug** zwischen dem betrachteten Qualitätsmerkmal und den Qualitätseigenschaften dargestellt. Hierzu wird die Relevanz des Qualitätsmerkmals im Bezug auf die betrachtete Qualitätseigenschaft begründet und eine mögliche Interpretation hinsichtlich der Qualitätseigenschaft gegeben.

(01) Anteil der Enhancements bezogen auf die Summe von Bugs und Enhancements

Diese Metrik ermittelt den Anteil der Enhancements bezogen auf die Summe von Enhancements und Bugs jeweils beim Zustandsübergang nach RESOL-

VED+FIXED. Der Anteil lässt sich zum einem anhand der Anzahl der Cases zum anderen anhand des tatsächlichen dokumentierten Arbeitsaufwands bestimmen.

Management der Anforderungen: Findet in der Betrachtungsperiode vordringlich Weiterentwicklung oder Fehlerbehebung statt?

Die Cases werden anhand des ihnen zugewiesenen Typs unterschieden. Betrachtet man die Cases vom Typ Enhancements, so besitzen diese einen Weiterentwicklungscharakter, wohingegen die Bugs die Fehlerbehebung beschreiben. Der Zustandsübergang eines Cases nach RESOLVED+FIXED dokumentiert das Ende seiner Bearbeitung durch die Entwicklerrolle. Hierdurch ist der Zeitbezug zum Betrachtungszeitraum gegeben. Wird der Anteil auf Basis der Anzahl der Cases bestimmt, werden unter Umständen die individuellen Unterschiede zwischen den Cases nicht ausreichend berücksichtigt. Daher ist eine Betrachtung auf Basis des tatsächlichen dokumentierten Arbeitsaufwands vorzuziehen. Voraussetzung ist jedoch, dass der Arbeitsaufwand für die einzelnen Cases korrekt erfasst und dokumentiert wird. Beide Alternativen berücksichtigen nicht, dass Cases innerhalb der Betrachtungsperiode mehrfach in den Zustand RESOLVED+FIXED wechseln können. Dies kann insbesondere bei der Ermittlung auf Basis des Arbeitsaufwands das Ergebnis verfälschen.

(02) Anteil der Cases mit externem Kundenbezug bezogen auf alle Enhancements

In dieser Metrik wird der Anteil der Enhancements mit dokumentiertem externem Kundenbezug bezogen auf alle Enhancements jeweils beim Zustandsübergang nach RESOLVED+FIXED ermittelt. Ein dokumentierter externer Kundenbezug liegt vor, wenn das Feld `Company` weder leer bleibt noch mit einem der Werte KISTERS, HYDSTRA oder INTERN belegt ist. Der Anteil kann sowohl auf Basis der Anzahl der Cases als auch auf Basis des tatsächlich dokumentierten Arbeitsaufwands ermittelt werden.

Management der Anforderungen: Wie hoch ist der Anteil der Projektentwicklung am Entwicklungsaufkommen?

Ein Entwicklungsziel von KISTERS ist es, Standard-Produkte zu entwickeln. Die Enhancements mit einem dokumentierten externen Kundenbezug können als Hinweis auf Projektentwicklung interpretiert werden. Berücksichtigt werden nur Enhancements mit der Resolution FIXED, so dass nur das tatsächliche Entwicklungsaufkommen der Betrachtungsperiode erfasst wird. Andere Resolutions (z.B. INVALID) haben im Allgemeinen keinen direkten Einfluss auf das Entwicklungsaufkommen oder besitzen keinen direkten Zeitbezug zur Betrachtungsperiode (z.B. Resolution DUPLICATE).

(03) Anteil der Cases, deren Typ-Klassifikation sich geändert hat

Es wird der Anteil der Cases, deren Typ-Klassifikation sich bis zum ersten Zu-

standsübergang nach RESOLVED geändert hat, im Verhältnis zu allen Cases mit erstem Zustandsübergang nach RESOLVED in der Betrachtungsperiode bestimmt.

Management der Anforderungen: Wie zuverlässig ist eine erste Klassifikation der Cases hinsichtlich ihres Typs?

Die Cases werden entsprechend ihres Charakters als Kundenanfrage, Änderungsanforderung oder Fehlermeldung klassifiziert. Wird die Klassifikation des Typs vor dem ersten Zustandsübergang nach RESOLVED gewechselt kann, dies Verzögerungen bei der Analyse und Bearbeitung des Cases bedeuten. Daher sollte frühzeitig ein Konsens mit dem Kunden über den Charakter der Anfrage erreicht werden. Durch die Betrachtung des Zustandsübergangs nach RESOLVED ist der Zeitbezug gewährleistet.

(04) Typ-Wechsel in den offenen Zuständen + REOPENED

Die einzelnen Anteile der Wechsel hinsichtlich des Typs (Bug, Enhancement und SupportCall) werden, bezogen auf alle Typ-Wechsel des Betrachtungszeitraums, in den Zuständen NEW, ASSIGNED, PROCESSING, WAITCUSTINFO und REOPENED bestimmt.

Management der Anforderungen: Wie zuverlässig ist eine erste Klassifikation der Cases hinsichtlich ihres Typs?

Diese Metrik erlaubt eine detaillierte Betrachtung der Typ-Wechsel in den ausgewählten Zuständen. Hierdurch lassen sich die Wechselbeziehungen aufzeigen. Im Gegensatz zu (03) werden die Typ-Wechsel in den Zuständen unabhängig von einem Besuch des Zustands RESOLVED betrachtet. Mithilfe dieser Metrik ist es möglich, Schwerpunkte innerhalb der verschiedenen Typ-Wechsel zu identifizieren.

(05) Anteil der Cases mit Hinweis auf parallele oder doppelte Arbeit

Diese Metrik ermittelt den Anteil der Cases, die mindestens einen Zustand aus der Menge {ASSIGNED, PROCESSING} besucht haben, bevor ihre Resolution erstmalig auf DUPLICATE gesetzt wurde. Bezogen auf die Menge aller Cases, deren Resolution zum ersten Mal auf DUPLICATE gesetzt wurde.

Management der Anforderungen: Kann parallele oder doppelte Arbeit vermieden werden?

Mit der Resolution DUPLICATE wird dokumentiert, dass eine ähnliche Anforderung bereits in Bugzilla erfasst wurde. Der Zustand PROCESSING dokumentiert aktive Tätigkeit zur Lösung des Cases. Hingegen ist ein Besuch des Zustands ASSIGNED nur ein möglicher Hinweis auf aktive Tätigkeiten, da die Dokumentation der aktiven Bearbeitung im Zustand PROCESSING aufgrund des Dokumentationsaufwands vermieden wird. Folglich kann ein Besuch eines

der beiden Zustände als Hinweis auf die aktive Tätigkeit hinsichtlich der Lösung des Cases ausreichen.

(06) Anteil der Cases mit geplantem Bearbeitungsbeginn

Zur Ermittlung dieser Metrik wird der Anteil der Cases bestimmt, welche einen Hinweis auf einen möglichen dokumentierten geplanten Bearbeitungsbeginn haben. Der geplante Bearbeitungsbeginn wird im Feld `targetMilestone` als „KW xx“ eingetragen, wobei xx durch die geplante Kalenderwoche zu ersetzen ist. Der Anteil wird ermittelt anhand der Cases mit Hinweise auf geplanten Bearbeitungsbeginn bis zum ihrem ersten Zustandsübergang nach RESOLVED +FIXED, bezogen auf alle Cases, deren erster Zustandsübergang nach RESOLVED+FIXED innerhalb der Betrachtungsperiode liegt.

Planung: Werden Termine geplant?

Die Planung des Bearbeitungstermins bildet die Grundlage zum Vergleich zwischen tatsächlichem und geplantem Fortschritt. Sie ist notwendig, um den Entwicklungsprozess zu koordinieren. Dies umfasst sowohl die Koordination des Zugriffs auf die Ressourcen als auch die Koordination der Entwicklungstätigkeiten bei der Realisierung der verschiedenen Funktionen des Software-Produkts. Die Metrik berücksichtigt nur die dokumentierten Bearbeitungstermine. Ohne Dokumentation des Bearbeitungstermins lässt sich die Umsetzung des Terminplans, ebenso wie die Schätzgüte hinsichtlich der Termineinhaltung, nicht überprüfen.

(07) Anteil der Cases mit dokumentierter Deadline

Die Metrik erlaubt die Bestimmung der Cases, die eine dokumentierte Deadline besitzen. Eine gültige Deadline besitzt das Datumsformat yyyy-mm-dd. Betrachtet wird der Anteil der Cases mit dokumentierter Deadline beim Zustandsübergang nach RESOLVED+FIXED bezogen auf alle Cases mit Zustandsübergang nach RESOLVED+FIXED innerhalb der Betrachtungsperiode. Alle in dieser Metrik erfassten Cases werden jeweils nur einmal gezählt (`countUnique`).

Planung: Werden Termine geplant?

Die Deadlines bestimmen explizit den Termin an dem aus Kundensicht eine Lösung des Cases zur Verfügung steht. Der Termin soll folglich so gewählt sein, dass alle Entwicklungsaktivitäten erfolgreich durchgeführt werden können. Hierzu sollte ein Konsens mit dem Kunden gefunden werden. Folglich ist eine Deadline eine verbindliche Zusage an den Kunden. Es gilt eine Überschreitung der Deadline zu vermeiden.

(08) Anteil der Cases, deren Bearbeitungsbeginn verschoben wurde

Diese Metrik ermittelt den Anteil der Cases, deren geplanter Bearbeitungsbeginn verschoben wurde. Der geplante Bearbeitungsbeginn wird als Kalenderwo-

che im `targetMilestone` dokumentiert. Der Anteil lässt sich anhand der Cases mit Hinweis auf einen verschobenen Bearbeitungsbeginn bei ihrem ersten Zustandsübergang nach RESOLVED+FIXED bezogen auf alle Cases mit erstem Zustandsübergang nach RESOLVED+FIXED ermitteln.

Planung: Werden die geplanten Termine eingehalten?

Das Vertrauen in die Ergebnisse der Planung hängt maßgeblich davon ab, ob realistische Pläne erstellt werden. Das häufige Neuplanen bedeutet sowohl für den Planungsverantwortlichen als auch für den betroffenen Entwickler neben einem Vertrauensverlust, zusätzlichen Aufwand. Der dokumentierte Bearbeitungsbeginn ist ein geplanter Termin. Wird der dokumentierte Bearbeitungsbeginn verschoben, kann der ursprünglich geplante Termin nicht eingehalten werden. Ein Hinweis auf einen verschobenen Bearbeitungsbeginn ist jedoch auch positiv zu bewerten, da offensichtlich der Regelkreislauf des Projektmanagements eingeschlossen wurde.

(09) Anteil der Cases, deren Deadline verletzt wurde (RESOLVED +FIXED)

Betrachtet wird der Anteil der Cases, deren Deadline bei Zustandsübergang nach RESOLVED+FIXED verletzt ist bezogen auf alle Cases die beim Zustandsübergang nach RESOLVED+FIXED eine gültige Deadline besitzen. Alle in dieser Metrik erfassten Cases werden jeweils nur einmal gezählt (`countUnique`).

Planung: Werden die geplanten Termine eingehalten?

Bei der Planung von Deadlines sind alle Bearbeitungsschritte der Entwicklung zu berücksichtigen. Werden Deadlines während der Bearbeitung verletzt, so entsteht dies möglicherweise aufgrund von Defiziten im Planungsprozess oder fehlenden Entwicklungskapazitäten.

(10) Anteil der Cases mit geschätztem Zeitaufwand

Der geschätzte Zeitaufwand wird im Feld `Estimated Hours` dokumentiert. Diese Metrik ermittelt den Anteil der Cases mit dokumentiertem geschätztem Zeitaufwand beim Zustandsübergang nach RESOLVED+FIXED bezogen auf alle Cases mit einem Zustandsübergang nach RESOLVED+FIXED. Die Cases werden in dieser Metrik innerhalb der einzelnen Anteile jeweils einmal gezählt (`countUnique`).

Planung: Wird der benötigte Zeitaufwand zur Umsetzung der Cases geplant?

Die Planung der Cases mit geschätztem Aufwand erlaubt eine bessere Übersicht über den geplanten Ressourceneinsatz. Bei der Planung der Cases ohne geschätzten Aufwand bleibt unter Umständen der mögliche Ressourceneinsatz unberücksichtigt.

(11) Median des `originalEffortEstimationAccuracy`-Gewichts

Die Metrik wird anhand des Gewichts `originalEffortEstimationAccuracy` von BugzillaMetrics [QMe] ermittelt. Das Gewicht vergleicht den geschätzten geplanten Zeitaufwand (`Estimated Hours`) eines Cases mit dessen dokumentiertem Zeitaufwand (`Hours Worked`). Die Ausprägungen werden auf das Intervall von 1 (absolute Übereinstimmung) bis 0 (keine Übereinstimmung) abgebildet. Beim Zustandsübergang nach RESOLVED+FIXED wird der Median für das Gewicht `originalEffortEstimationAccuracy` der Cases mit geschätztem Aufwand gebildet. Das Gewicht differenziert nicht zwischen dem Über- und Unterschätzen des geplanten Zeitaufwands.

Planung: Ist die Schätzung des Zeitaufwands hinreichend genau?

Um eine realistische Planung hinsichtlich des Zeitaufwands zu erhalten, muss der geplante mit dem tatsächlichen Zeitaufwand fortlaufend verglichen werden. Die Differenzen zwischen geschätztem und tatsächlichem Zeitaufwand sollten im Rahmen der Verbesserung des Planungsprozesses verringert werden. Eine ausreichend genaue Schätzung des geplanten Zeitaufwands ist die Voraussetzung dafür, den Entwicklungsfortschritt anhand des tatsächlichen geleisteten Zeitaufwands zu bestimmen.

(12) Planungsreichweite

Die Planungsreichweite wird im Folgenden definiert als die Summe des verbleibenden Zeitaufwands (`remaining Workload`) aller Cases in den Zuständen NEW, ASSIGNED, PROCESSING, WAITCUSTINFO und REOPENED am Ende des Betrachtungsintervalls dividiert durch die Summe des erfassten tatsächlichen Zeitaufwands beim Zustandsübergang von NEW, ASSIGNED, PROCESSING und WAITCUSTINFO nach RESOLVED+FIXED. Der Zustandsübergang von REOPENED nach RESOLVED+FIXED wird nicht berücksichtigt, da ansonsten der geleistete Aufwand zu sehr verfälscht wird. Denn bei einem Zustandswechsel von REOPENED wird erneut der gesamte geleistete Aufwand angesetzt.

Planung: Wie groß ist die Planungsreichweite?

Die Planungsreichweite beschreibt, in welchem Zeitraum der verbleibende geschätzte Zeitaufwand unter der ceteris paribus Annahme bearbeitet werden kann. Unter der Voraussetzung, dass der geschätzte und der tatsächliche Zeitaufwand dokumentiert werden, kann die Planungsreichweite Hinweise für die Planung der nächsten Perioden geben.

(13) Anteil der Cases mit dokumentierter Nutzung von ASSIGNED und PROCESSING

Betrachtet wird der Anteil der Cases mit Hinweis auf mindestens einen Besuch der Zustandsmenge {ASSIGNED, PROCESSING} bis zum ersten Zustands-

übergang nach RESOLVED+FIXED bezogen auf alle Cases mit erstem Zustandsübergang nach RESOLVED+FIXED in der Betrachtungsperiode.

Überwachung und Verfolgung des Entwicklungsfortschritts: Wird der Entwicklungsfortschritt der Bearbeitung dokumentiert?

Zur Überwachung des Entwicklungsfortschritts müssen die geplanten mit den tatsächlichen Größen verglichen werden. Dazu sollte der Entwicklungsfortschritt anhand der Zustandsübergänge nach ASSIGNED und PROCESSING dokumentiert werden. Jedoch sind die beschriebenen Zustandsübergänge nur ein Hinweis auf die Bearbeitung des Cases, welche mögliche Entwicklungsaktivität in Bugzilla abbilden.

(14) Anteil der Cases mit dokumentierter Erfassung der Arbeitszeiten

Zur Ermittlung dieser Metrik werden alle Cases beim Zustandsübergang nach RESOLVED+FIXED ermittelt, deren geschätzter Zeitaufwand ungleich Null ist. Anhand dieser Cases wird der Anteil der Cases mit dokumentiertem Zeitaufwand bestimmt.

Überwachung und Verfolgung des Entwicklungsfortschritts: Wird der Entwicklungsfortschritt der Bearbeitung dokumentiert?

Durch die Dokumentation der Arbeitsstunden in Bugzilla kann der mögliche aktuelle Entwicklungsfortschritt der Cases nachvollzogen werden. Hierzu bedarf es einer plausiblen Schätzung des geplanten Zeitaufwands und einer angemessenen Erfassung des Zeitaufwands. Wird der Entwicklungsfortschritt nur anhand der Zustandsübergänge dokumentiert, sind nur sehr grobe Aussagen möglich.

(15) Granularität der Cases

Diese Metrik erfasst den geschätzten Zeitaufwand von den Zuständen NEW, WAITCUSTINFO, ASSIGNED und PROCESSING beim Zustandsübergang nach RESOLVED+FIXED. Anschließend wird der Median der geschätzten Zeitaufwände der Cases für den Betrachtungszeitraum ermittelt.

Überwachung und Verfolgung des Entwicklungsfortschritts: Wie ist die Granularität der Cases?

Die Granularität der Cases wird anhand der geschätzten Stunden ermittelt. Der geplante Aufwand beschreibt den Umfang der benötigten Tätigkeiten zur Realisierung des Cases. Die einzelnen Cases sollten hinsichtlich des Umfangs der Tätigkeiten vergleichbar sein, da ansonsten ein Vergleich auf Basis zwischen den Cases ohne Berücksichtigung des Aufwands nur unzureichende Aussagen ermöglicht. Wird die Granularität der Cases zu klein gewählt, entsteht unnötiger Dokumentationsaufwand. Die Cases, deren Umfang zu groß gewählt wurde, sind in der Verfolgung durch das Risikomanagement aufwendiger.

(16) Median der Bearbeitungszeiten (time to solution)

Diese Metrik ermittelt das Zeitintervall vom Einstellen des Cases in Bugzilla bis zum ersten Zustandsübergang nach RESOLVED+FIXED. Anschließend wird der Median der Zeitintervalle für den Betrachtungszeitraum gebildet.

Lösungsgeschwindigkeit: Wie lange dauert es bis zum Abschluss der Bearbeitung?

Die Zeitspanne bis zur erfolgreichen Bearbeitung eines Cases bestimmt maßgeblich die Lösungszeit im Bearbeitungsprozess, das heißt die Zeitspanne von der Erfassung des Cases bis zur Dokumentation seiner Bearbeitung durch den Zustandswechsel nach RESOLVED+FIXED. Die Bearbeitungszeit darf nicht mit dem erfassten Zeitaufwand (`Hours Worked`) verwechselt werden.

(17) Median der Lösungszeiten (time to customer accept)

Diese Metrik ermittelt das Zeitintervall vom Einstellen des Cases in Bugzilla bis zum ersten Zustandsübergang nach CLOSED+FIXED. Anschließend wird der Median der Zeitintervalle für den Betrachtungszeitraum gebildet.

Lösungsgeschwindigkeit: Wie lange dauert es bis zum Schließen des Cases?

Die gesamte Zeitspanne zwischen der Erfassung und dem Schließen des Cases bestimmt die Lösungszeit. Bei einem Case vom Typ Bug bedeutet eine kürzere Lösungszeit, dass die gestörte Funktionalität dem Kunden schnell wieder zur Verfügung steht. Je kürzer die Lösungszeit bei Cases des Typs Enhancement ist, desto früher kann der Kunde die neue beziehungsweise geänderte Funktionalität der Software nutzen.

(18) Anteil der Nacharbeiten innerhalb der Periode

Der Anteil wird anhand der Zustandswechsel von REOPENED in einen anderen Zustand bezogen auf alle Zustandswechsel nach RESOLVED+FIXED ermittelt. Hierzu werden nur Cases betrachtet, deren Verweildauer in REOPENED größer als 1 Stunde war.

Software-Qualitätssicherung: In welchem Umfang fallen ungeplante Nacharbeiten an?

Im Rahmen der Software-Qualitätssicherung wird die Qualität der einzelnen in der Softwareentwicklung erstellten Artefakte sichtbar gemacht. Für die gefundenen Defizite werden Fehlermeldungen eingestellt oder Nacharbeiten angeordnet. Ein hoher Anteil an Nacharbeiten kann ein Hinweis auf Probleme während der Entwicklung sein. Hierzu sollte im Rahmen der Qualitätssicherung untersucht werden, ob die Defizite prozessinhärent sind oder individuelle Gründe vorliegen. Die Verweildauer von mindestens einer Stunde wurde empirisch bestimmt (vgl. 7 Diskussion) und hat technische Ursachen.

(19) Median Dauer der Nacharbeiten

Im Folgenden wird das Zeitintervall zwischen dem Zustandswechsel nach REOPENED und dem Verlassen des Zustands REOPENED betrachtet. Zur Ermittlung der Ausprägung wird der Median der Zeitintervalle innerhalb des Betrachtungszeitraums gebildet.

Software-Qualitätssicherung: Wie lange ist die Bearbeitungsdauer der ungeplanten Nacharbeiten?

Die Nacharbeiten sind ungeplante Tätigkeiten bei der Bearbeitung eines Cases. Im Allgemeinen ist der hierfür notwendige Aufwand nicht im Budget geplant. Zudem entsteht durch die Nacharbeiten eine Verzögerung im Entwicklungsprozess. Hierdurch wird die Lösungsgeschwindigkeit innerhalb des Prozesses reduziert. Die Anzahl und Dauer dieser Verzögerungen sind zu minimieren. Die Dauer der Nacharbeiten ist ein Hinweis auf den ungeplanten Mehraufwand durch unzureichende Lösungen der Cases.

(20) Defect Escape Rate

Die Defect Escape Rate ist das Verhältnis der eingehenden Fehlermeldungen mit externem Unternehmensbezug zu allen eingehenden Fehlermeldungen des Betrachtungszeitraums. Ein dokumentierter externer Kundenbezug liegt vor, wenn das Feld `Company` weder leer bleibt noch mit einem der Werte KISTERS, HYDSTRA oder INTERN belegt ist. Zudem wird das Feld `Company` als fixed-Field in der Metrik gekennzeichnet (siehe [QMe]).

Software-Qualitätssicherung: Wie sensitiv ist die Fehlersuche?

Die Prüfung der Artefakte in der Software-Qualitätssicherung erfordert fehlersensitive Prüf- und Testverfahren. Ein niedriger Anteil bei der Defect Escape Rate kann ein Hinweis auf die Verwendung von fehlersensitiven Prüf- und Testverfahren sein. Jedoch werden nur die dokumentierten Bugs betrachtet. Die nicht dokumentierten Bugs, welche insbesondere zu Beginn von neuentwickelten Produkten auftreten, können das Ergebnis verfälschen. Der Zeitbezug ist gegeben, da nur die neu eingestellten Bugs der Periode betrachtet werden. Die Cases, welche unter einer anderen Typ-Klassifikation eingestellt werden, bleiben unberücksichtigt.

(21) Anteil der Cases mit `targetMilestone`

Der Anteil der Cases mit gültigem `targetMilestone` wird beim ersten Zustandsübergang nach CLOSED+FIXED ermittelt. Ein `targetMilestone` wird als gesetzt interpretiert, falls weder „---“ noch ein Hinweis auf Kalenderwoche „KW“ vorliegt. Zur Berechnung des Anteils werden die Cases mit einem Hinweis auf einen gültigen `targetMilestone` bezogen auf alle Cases mit erstem Zustandswechsel nach CLOSED+FIXED ermittelt.

Software-Konfigurationsmanagement: Wird der `targetMilestone` für den Case dokumentiert?

Spätestens wenn ein Case erfolgreich ausgeliefert wird, ist der Releasename im `targetMilestone` des Software-Produkts zu dokumentieren. Hierdurch wird die Verfolgung des Cases im Rahmen des Konfigurationsmanagements ermöglicht. Zusätzlich erfolgt häufig eine produktspezifische Dokumentation mittels Flags.

(22) Anteil der Cases mit `releaseNotes`

Diese Metrik ermittelt den Anteil der Cases mit Hinweis auf gesetzte `releaseNotes` beim ersten Zustandsübergang nach CLOSED+FIXED bezogen auf alle Cases mit erstem Zustandsübergang nach CLOSED+FIXED. Hierzu werden sowohl die deutschsprachigen als auch die englischsprachigen `releaseNotes` berücksichtigt.

Software-Konfigurationsmanagement: Werden die umgesetzten Anforderungen in den `releaseNotes` dokumentiert?

Die `releaseNotes` beschreiben aus Sicht des Kunden die durch den Case realisierte Funktionalität. Die `releaseNotes` eines jeden Releases werden gesammelt und dem Kunden bereitgestellt. Funktionalität, die nicht über die `releaseNotes` dokumentiert wird, ist gegebenenfalls für den Kunde nicht sichtbar. Eine einheitliche Verwendung während des Betrachtungszeitraums ist eine Voraussetzung zur Verwendung dieser Metrik (vgl. 7 Diskussion).

(23) Anteil der Cases mit Einträgen in der CC-Liste

In dieser Metrik wird die Anzahl der Einträge in der CC-Liste des Cases mithilfe des Gewichts `ccCount` ermittelt (siehe [QMe]). Der Anteil der Cases mit Einträgen beim Zustandsübergang nach RESOLVED+FIXED wird bezogen auf alle Zustandsübergänge nach RESOLVED+FIXED bestimmt.

Koordination: Wie hoch ist der Koordinationsaufwand bei der Bearbeitung der Cases?

Bei Änderungen des Cases werden zusätzlich zum reporter, assignee und qaContact, die in der CC-Liste eingetragenen Personen per E-Mail informiert. Folglich sind die Personen auf der CC-Liste bei der Koordination zu berücksichtigen.

(24) Mittlere Anzahl der Einträge in der CC-Liste

Der Mittelwert (arithmetisches Mittel) bezogen auf die Anzahl der Einträge in der CC-Liste der Cases wird beim Zustandsübergang nach RESOLVED+FIXED ermittelt. Hierzu wird das Gewicht `ccCount` von BugzillaMetrics verwendet (siehe [QMe]).

Koordination: Wie hoch ist der Koordinationsaufwand bei der Bearbeitung der Cases?

Die bei den CC's eingetragenen Personen werden bei Änderungen des Cases informiert. Folglich besitzen sie ein Informationsbedürfnis hinsichtlich des Cases. Ist der Mittelwert der Anzahl der Anspruchspersonen hoch, so ist dies ein Hinweis auf einen erhöhten Koordinationsaufwand.

(25) Mittlere Anzahl der Kommentare

Mithilfe des `commentCount` Gewichts von BugzillaMetrics lässt sich die Anzahl der Kommentare ermitteln (siehe [QMe]). Diese Metrik ermittelt den Mittelwert (arithmetisches Mittel) der Anzahl aller Kommentare der Cases mit Zustandsübergang nach RESOLVED+FIXED innerhalb der Betrachtungsperiode.

Koordination: Wie hoch ist der Koordinationsaufwand bei der Bearbeitung der Cases?

Mithilfe der Kommentare wird ebenfalls die Bearbeitung der Cases dokumentiert. Die Kommunikation über den Case soll nicht mittels der Kommentare erfolgen. Stattdessen sollen durch die Kommentare die Ergebnisse dokumentiert werden. Folglich sollten die verschiedenen Anspruchspersonen die Kommentare mitverfolgen. Ein hoher Umfang an Kommentaren ist ein Hinweis auf ein hohes Maß an koordinierenden Tätigkeiten.

(26) Mittlere Anzahl der Assignee-Wechsel

In dieser Metrik wird die mittlere Anzahl an Assignee-Wechseln bis zum ersten Zustandsübergang nach RESOLVED+FIXED ermittelt. Hierzu wird das arithmetische Mittel aus der Summe der Assignee-Wechsel bezogen auf die Anzahl der betrachten Cases gebildet.

Koordination: Wie hoch ist der Koordinationsaufwand bei der Bearbeitung der Cases?

Der Entwickler übernimmt die Bearbeitung indem der Case auf ASSIGNED gesetzt wird. Die Assignee-Wechsel dokumentieren den Wechsel des zuständigen Entwicklers beziehungsweise die Delegation eines Cases. Durch häufige Wechsel der Assignee entsteht zusätzlicher Koordinationsaufwand hinsichtlich der Bearbeitung der Cases.

(27) Mittlere Anzahl der Besuche des Zustands WAITCUSTINFO

Die mittlere Anzahl der Besuche des Zustands WAITCUSTINFO wird ermittelt, indem die Anzahl der Besuche von WAITCUSTINFO bis zum ersten Zustandsübergang nach RESOLVED+FIXED ermittelt wird. Anschließend wird das arithmetische Mittel aus der Summe der Besuche von WAITCUSTINFO und der Anzahl der betrachteten Cases mit erstem Zustandsübergang nach RESOLVED+FIXED in der Betrachtungsperiode ermittelt.

Koordination: Wieviel Reibung entsteht in dem Koordinationsprozess mit dem Kunden?

Durch den Zustandsübergang nach WaitCustInfo wird dokumentiert, dass zur Umsetzung des Cases zusätzliche Informationen beim Kunden angefordert wurden. Dies dokumentiert die Notwendigkeit zur Koordination zwischen der Anspruchsgruppe Kunde und der Organisation KISTERS.

(28) Anteil der Sofortlösungen

Im Folgenden wird der Anteil der Sofortlösungen bestimmt. Hierzu wird die Anzahl der SupportCalls ohne weitere Zustandsbesuche bis zum ersten Zustandsübergang nach CLOSED+FIXED bezogen auf alle SupportCalls mit erstem Zustandsübergang nach CLOSED+FIXED ermittelt.

Supportprozess: Wie viele der SupportCalls können direkt gelöst werden?

Die SupportCalls dürfen im Gegensatz zu den Bugs und Enhancements direkt als „CLOSED“ klassifiziert werden. Ein hoher Anteil deutet auf eine effiziente Kommunikation mit dem Kunden hin, was nur durch ein hohes Maß an Kundenverständnis möglich ist. Jedoch kann ein hoher Anteil ebenfalls eine schlechte Dokumentation der Umsetzung der Cases bedeuten.

(29) Anteil an Fire-Fighting Aktivitäten

Zur Bestimmung des Anteils an Fire-Fighting Aktivitäten wird als Normalisierungsgröße die Anzahl der Cases mit Zustandswechsel nach RESOLVED +FIXED gezählt (`countUnique`). Die Anzahl der Cases (`countUnique`) mit Fire-Fighting Aktivitäten wird bei ihrem Zustandsübergang von P1 in eine niedrigere Priorität innerhalb der Zustände NEW, WAITCUSTINFO, ASSIGEND, PROCESSING oder REOPENED erfasst, sowie alle Cases mit der Priorität P1 mit Zustandsübergang nach RESOLVED+FIXED. Hierdurch kann der Anteil der Fire-Fighting Aktivitäten mit dem (32) Median der Lösungszeit der Fire-Fighting Aktivitäten in Beziehung gesetzt werden.

Fire-Fighting Aktivitäten: Wie hoch ist der Anteil der Fire-Fighting Aktivitäten im Prozess?

Die Cases mit der Priorität „P1“ werden unter der Fire-Fighting Aktivität zusammengefasst. Die Bearbeitung aller anderen Cases wird zugunsten der Fire-Fighting Aktivitäten zurückgestellt. Folglich muss die laufende Bearbeitung anderer Cases unterbrochen werden. Ein hoher Anteil an Fire-Fighting Aktivitäten bewirkt häufige Unterbrechungen und Wechselzeiten zwischen den Cases. Zudem finden die Fire-Fighting Aktivitäten ungeplant statt und besitzen aufgrund der ASAP (as soon as possible) Bearbeitung einen deutlich höheren Ressourceneinsatz.

(30) Median der Lösungzeit der Fire-Fighting Aktivitäten

Zur Bestimmung der Zeitintervalle der Lösungszeiten der Fire-Fighting Aktivitäten wird der Zeitraum gemessen, während der sich ein Case mit Priorität P1 in den Zuständen NEW, WAITCUSTINFO, ASSIGNED, PROCESSING und REOPENED befunden hat. Anschließend wird der Median der Zeitintervalle innerhalb des Betrachtungszeitraums bestimmt.

Fire-Fighting Aktivität: Wie lange dauert die Lösung der Cases hinsichtlich der Priorität P1?

Während der Fire-Fighting Aktivität muss gegebenenfalls die Bearbeitung anderer Cases zugunsten des Notfalls unterbrochen werden. Daher kann die Bearbeitungszeit anderer Cases beeinträchtigt werden. Folglich ist die Dauer bis zur Umsetzung der Fire-Fighting Aktivität zu minimieren!

(31) Anteil der Cases, deren Deadline verletzt wurde (CLOSED +FIXED)

Betrachtet wird der Anteil der Cases, deren Deadline bei Zustandsübergang nach CLOSED+FIXED verletzt ist bezogen auf alle Cases, die beim Zustandsübergang nach CLOSED+FIXED eine gültige Deadline besitzen. Alle in dieser Metrik erfassten Cases werden jeweils nur einmal gezählt (`countUnique`).

Kundenzufriedenheit: Wie zuverlässig ist der Software-Anbieter hinsichtlich der Termin-Absprachen?

Der Termin bezüglich der Deadlines ist gemeinsam mit dem Kunden zu ermitteln. Folglich erwartet der Kunde die Einhaltung der Deadline. Für den Kunden sind im Allgemeinen nur die verletzten Deadlines vor dem Zustand CLOSED+FIXED relevant. Die Auslieferung eines Cases in einem Release wird nach Bestätigung des (internen beziehungsweise externen) Kunden durch Zustandswechsel nach CLOSED+FIXED dokumentiert. Folglich entspricht diese Sichtweise der Leistungserwartung des Kunden.

(32) Anteil über das Kundenportal

Die Metrik bestimmt den Anteil der SupportCalls, welche über das Kundenportal eingestellt wurden. Hierzu wird der Anteil der eingestellten SupportCalls ohne E-Mail Adresse von KISTERS bezogen auf alle eingestellten SupportCalls im Betrachtungszeitraum ermittelt.

Kundenzufriedenheit: Ändert sich die Kommunikationsstruktur zum Kunden?

Die Kunden der Firma KISTERS besitzen die Möglichkeit, über das Kundenportal im Internet Cases einzustellen und deren Bearbeitung online zu verfolgen. Ein hoher Anteil zeigt den Wandel der Kommunikations- und Koordinationsstrukturen. Die Wechselwirkungen mit den bestehenden Prozessen sind zu beobachten.

5 Hinweise zur Analyse

Inhalt

Bei der Analyse der Softwareentwicklungs- und Wartungsprozesse anhand von Change-Request Daten sind verschiedene Aspekte zu berücksichtigen. Diese Aspekte umfassen den Kontext der Prozesse, gegeben durch das Projekt und den gelebten Prozess. Des Weiteren sind Besonderheiten im Umgang mit einem bidirektionalen Qualitätsmodell und den verwendeten Analyseverfahren zu beachten. Abschließend werden Hinweise zur Verwendung von BugzillaMetrics gegeben sowie im Rahmen der Diplomarbeit aufgetretene Fehler und realisierte Erweiterungen beschrieben.

5.1 Projekt und gelebter Prozess

Der Ablauf der Entwicklung und Wartung lässt sich mithilfe von Prozess- und Vorgehensmodellen beschreiben. Die Analyse von Softwareentwicklungs- und Wartungsprozessen muss sich jedoch an den **gelebten** Prozessen orientieren (vgl. [ER03]). Hierzu bietet sich eine Analyse anhand der im Change-Request-Management-System erfassten Daten an.

Ein vorgegebener Standardprozess dient als Modell für den Ablauf, der für die konkrete Entwicklung anzupassen ist. Die Anforderungen hinsichtlich des Prozesses sind durch den Kontext des Entwicklungsprojekts beziehungsweise der Produktentwicklung bestimmt. Jedoch ändert sich der ursprüngliche Kontext im zeitlichen Verlauf. Beispielsweise wird aus Budgetrestriktionen der Umfang eines Entwicklungsprojekts verkleinert, oder das ursprüngliche Entwicklungsprojekt bildet aufgrund seines Erfolgs die Grundlage für die Produktentwicklung. Zusätzlich können in den einzelnen Entwicklungsprojekten individuelle Standards und Vorgehensweisen entstehen. Folglich wird sich der gelebte Prozess den geänderten Anforderungen anpassen. Die Erfahrung mit den Prozessen der Entwicklungsprojekte oder der Produktentwicklung können im Rahmen der Prozessverbesserung zu Anpassungen am Standardprozess führen, wie zum Beispiel zu einem neuen Schema zum Priorisieren von Anforderungen. Bei der Analyse sind die Veränderungen der Prozesse ausreichend zu berücksichtigen. Jedoch werden

die Veränderungen im gelebten Prozess häufig nicht bewusst wahrgenommen, wodurch sich der „kurze“ Dienstweg unter Umständen zum gebräuchlichen Ablauf entwickelt. Daher werden viele der Anpassungen des gelebten Prozesses nicht dokumentiert.

Des Weiteren wird die Verwendung des Prozesses von verschiedenen Personen oder Organisationsstrukturen unterschiedlich interpretiert. Eine unterschiedliche Interpretation des Prozesses kann eine uneinheitliche oder falsche Verwendung der Prozesse zur Folge haben. Wird der Prozess von den beteiligten Personen nicht verstanden, so kann dies gegebenenfalls zur Nichtnutzung einzelner vorgeschriebener Prozessaktivitäten und -tätigkeiten führen. Zudem können Vorbehalte gegenüber der Verwendung eines definierten Prozesses bestehen, so dass nicht ausgeschlossen werden kann, dass die Prozesse vorsätzlich falsch oder nicht genutzt werden.

5.2 Analyse mit einem bidirektionalen Qualitätsmodell

Die Grundlage zur Ausprägung eines bidirektionalen Qualitätsmodells zur Analyse von Softwareentwicklungs- und Wartungsprozessen bilden die Informationsbedürfnisse der Firma oder Organisation. Demgemäß sind die definierten Qualitätseigenschaften aus der Begriffswelt der Organisation oder dem unternehmensweiten Sprachgebrauch (engl. corporate language) entnommen und vor diesem Hintergrund zu interpretieren. Ein uneinheitlicher Sprachgebrauch innerhalb der verschiedenen Einheiten der Organisation oder Firma erschwert die Kommunikation zwischen den verschiedenen Anspruchsgruppen bei der Definition und Analyse mit einem bidirektionalen Qualitätsmodell. Im Rahmen der Entwicklung eines Qualitätsmodells besteht sogar die Chance, einen einheitlichen Sprachgebrauch zu etablieren, um Kommunikationsstörungen oder sogar -barrieren zu vermeiden. Durch den einheitlichen Sprachgebrauch wird ein Konsens hinsichtlich der zuvor subjektiven Qualitätserwartung gefördert (vgl. 2.3 Qualitätsmodelle).

Bei der Ausprägung eines bidirektionalen Qualitätsmodells für die Analyse von Prozessen ist eine isolierte Betrachtung der einzelnen Qualitätseigenschaften nicht immer möglich. So macht die Betrachtung des Anteils der verletzten Deadlines nur Sinn, wenn die Verwendung von Deadlines dokumentiert und geplant wird. Durch die Verknüpfung der einzelnen Elemente entsteht ein Beziehungsgeflecht, welches die Interpretation erschweren kann.

Als Qualitätsmerkmale sind nur Prozessattribute geeignet, die durch direkte oder indirekte Messung zu ermitteln sind. Insbesondere ist die Definition der Qualitätsmerkmale bei der Analyse anhand von Change-Requests auf die erfassten Merkmale beschränkt. In den Change-Requests nicht erfasste Daten, wie die geplante Auslieferung einer Anforderung in dem Produkt, sind der Analyse nicht zugänglich. Ebenso sind Informationen in natürlicher Sprache, wie in

den Kommentaren, selten automatisiert auswertbar, da sie in unstrukturierter oder nicht standardisierter Form vorliegen. Die Kommentare werden jedoch bei der Bearbeitung einer Anforderung häufig zur Dokumentation in einem Change-Request genutzt, da sie in ihrer Verwendung am flexibelsten sind.

Die Qualitätsindikatoren beschreiben, wie die ermittelten quantitativen Ausprägungen der Qualitätsmerkmale im Hinblick auf Qualitätseigenschaften zu interpretieren sind. Die Anpassungen an den Prozess bedingen, dass sich die Interpretation hinsichtlich der Qualitätsmerkmale ändern kann. Dies ist bei der Wahl des Betrachtungszeitraums und in den Qualitätsindikatoren zu berücksichtigen. Beispielsweise wurde bei KISTERS die Klassifikation hinsichtlich der Art der Anforderung von Bugs und Enhancements um die SupportCalls erweitert. Die Anpassungen an den Prozessen können unterschiedliche Auswirkungen auf die im Change-Request-Management-System gespeicherten Daten haben. Es kann zu strukturellen Änderungen der Change-Request (Bsp. neue Felder) oder einer geänderten Semantik eines Attributs des Change-Requests (Bsp. Bearbeitungstermin im Feld `targetMilestones`) kommen sowie zu Kombinationen aus beiden (Bsp. neuer Zustand „unconfirmed“ und geänderte Semantik bezüglich „new“). Bei den Anpassungen ist zudem zu differenzieren, ob diese an einem Stichtag vorgenommen werden oder eine kontinuierliche Umstellung erfolgt. Ein Qualitätsindikator, der die Change-Request mit höchster Dringlichkeit (P1) betrachtet, muss berücksichtigen, dass sich trotz strukturell gleicher Inhalte die Semantik im zeitlichen Verlauf geändert haben kann.

Einzelne Merkmale in den konkreten Prozessen sind unter Umständen nicht ausgeprägt, weil die entsprechenden Aktivitäten und Tätigkeiten im Betrachtungszeitraum nicht stattgefunden haben. Betrachtet ein Indikator ein solches Merkmal, muss ebenso der fehlenden Ausprägung eine qualitative Bewertung zugewiesen werden.

Die ermittelten Ausprägungen der Qualitätsmerkmale lassen sich mithilfe von Vergleichen auf Indexkennzahlen abbilden (vgl. 2.3 Qualitätsmodelle). Dabei besitzt der Umfang der Datenbasis unterschiedliche Einflüsse auf die Ergebnisse der Vergleiche. Insbesondere eine geringe Datenbasis kann zu extrem volatilen Ergebnissen führen. Die empirischen Vergleiche benötigen eine ausreichend große Datenbasis mit ähnlichen Prozessen um sinnvolle Aussagen zu ermöglichen.

Bei der Abbildung auf Indexkennzahlen und deren Aggregation können sich die unterschiedlichen qualitativen Ausprägungen untereinander kompensieren. Betrachtet man beispielsweise die Aggregation zweier Indexzahlen mittels der Bildung des Mittelwerts auf dem Intervall $[-1, 1]$, so lässt sich eine durchschnittliche Bewertung ($\frac{0+0}{2} = 0$) nicht mehr von einer „*auffälligen*“ Bewertung ($\frac{-1+1}{2} = 0$) unterscheiden.

Verschiedene Aspekte sind bei der Wahl des Betrachtungszeitraums zu berücksichtigen. Zum einen sollten innerhalb des Betrachtungszeitraums möglichst geringe Anpassungen am Prozess stattgefunden haben und der Prozess in seiner

Ausführung stabil sein. Zum anderen muss der Betrachtungszeitraum entsprechend groß gewählt werden um den Informationsbedürfnissen zu entsprechen.

5.3 BugzillaMetrics

Im folgenden Abschnitt werden Hinweise zur Verwendung von Bugzilla gegeben. Des Weiteren werden einige der im Rahmen der Diplomarbeit aufgetretenen Fehler und Verbesserungen exemplarisch erläutert und um mögliche Erweiterungen des Werkzeugs BugzillaMetrics ergänzt.

Das Volumen der Cases für den Betrachtungszeitraum der Bugzilla Datenbank ist zu groß um eine manuelle Analyse durchzuführen. Daher wird zur automatisierten Analyse im Folgenden die Software BugzillaMetrics verwendet. Die Software BugzillaMetrics ermittelt anhand vorgegebener oder vom Benutzer selbst definierter Metriken auf Basis der Informationen der Bugzilla Cases die Ausprägung für die Qualitätsmerkmale der Prozesse (vgl. 3.3 BugzillaMetrics). Folglich lassen sich nur die Qualitätsmerkmale betrachten, die mittels einer Metrik beschrieben werden können.

Die State- und Eventfilter erfassen die Merkmalsausprägungen der Cases. In einem Event-Filter lassen sich immer nur einzelne Ereignisse, zum Beispiel ein Zustandswechsel, betrachten. Werden mehrere Attribute eines Cases im Rahmen seiner Bearbeitung gleichzeitig geändert, so werden diese als Folge von Ereignissen in der Bugzilla Datenbank erfasst. Beispielsweise wird ein Zustandswechsel der Cases nach RESOLVED mit der Resolution FIXED als zwei getrennte Ereignisse durch BugzillaMetrics interpretiert. Im Bugzilla-Workflow ist es nur beim Zustandsübergang nach RESOLVED (bzw. CLOSED) möglich, die Resolution zu ändern. Aufgrund des gegebenen Prozesskontexts ist es möglich, statt der zwei getrennten Ereignisse alternativ den **Zustandsübergang** der Resolution **im Zustand** RESOLVED zu betrachten. Im Allgemeinen ist eine Auswertung, ob sich mehrere Merkmale eines Cases gleichzeitig geändert haben, nicht möglich. Eine mögliche Erweiterung des Werkzeugs BugzillaMetrics hinsichtlich der gleichzeitigen Änderung mehrerer Felder bei der Bearbeitung eines Cases ist insbesondere zur Betrachtung des Zustandsübergangs nach RESOLVED sinnvoll.

Bei der Verwendung von **State-Filtern**, die nicht direkte Inhalte der Datenbank repräsentieren, sind insbesondere deren semantische Besonderheiten zu berücksichtigen. So besitzt das Feld `beyondDeadline` im Zustandsfilter auch für die Cases ohne Deadline eine Merkmalsausprägung. Folglich führt eine Betrachtung des Anteils der Cases, welche die Deadline verletzt haben, unter Umständen zu falschen Interpretationen bezüglich des Umgangs mit Deadlines. Ein weiteres Beispiel ist die Besonderheit zur Analyse der Bugzilla-Kommentare mithilfe der Felder `Comment` und `Comments`.

Die Spezifikation einer Berechnungsvorschrift einer Metrik erfolgt anhand eines **Value Calculator**. In BugzillaMetrics sind bisher vier verschiedene Value Calculators definiert: IntervalLengthCalculator, CountUntilCalculator, ResidenceTimeCalculator und CountEventsCalculator.

Der **IntervalLengthCalculator** bestimmt die Zeitintervalle zwischen Ereignissen eines Cases. Hierzu werden Start- und Endereignis mithilfe von Event-Filtern spezifiziert. Die Case Values werden jeweils für die Endereignisse bestimmt. Der Event-Filter des Startereignisses kann im Allgemeinen mit mehreren Ereignissen übereinstimmen. Folglich liegen unter Umständen mehrere Startereignisse vor einem Endereignis. In der ursprünglichen Implementierung wurde für jedes Startereignis das Zeitintervall bis zum nächsten Endereignis ermittelt. Infolge dessen werden zu einem Endereignis mehrere Startereignisse in Beziehung gesetzt. Im Allgemeinen wird mit dem IntervalLengthCalculator jedoch die Beziehung zwischen einem Start- und einem Endereignis modelliert. Beispielsweise gibt es unter Umständen, aufgrund von Nacharbeiten (REOPENED), mehrere Zustandsübergänge nach RESOLVED+FIXED bis zum endgültigen Zustandswechsel nach CLOSED+FIXED. Daher wurde die Semantik dieses Value Calculators angepasst, so dass nur noch das kürzeste Zeitintervall zwischen mehreren Start- und dem Endereignis berücksichtig wird. Zudem wurde dieser Value Calculator um die Möglichkeit erweitert für das erste Endereignis, alle Endereignisse oder nur das letzten Endereignis eines Auswertungszeitraums ein Case Value zu ermitteln.

Der **CountUntilCalculator** beginnt die Zählung der spezifizierten Ereignisse immer ausgehend von dem Ereignis „`created`“. Eine Erweiterung der ursprünglichen Implementierung wäre sinnvoll, so dass mithilfe anderer Startereignisse, die Zählung beginnt. Somit könnten insbesondere Cases mit Nacharbeiten weiter untersucht werden.

Der **ResidenceTimeCalculator** ermittelt zu jedem festgelegten Ereignis die Verweildauer in Tagen. Treten diese Ereignisse mehrfach innerhalb der Betrachtungsperiode auf, wird jeweils die gesamte Verweildauer (V) bis zu einem Ereignis (X) bestimmt. Im Beispiel (siehe Abb. 5.1) werden im zweiten Betrachtungszeitraum zwei Case Values für einen betrachteten Case erstellt. Die Summe der ermittelten Case Values für den Case ist offensichtlich größer als die gesamte Verweildauer des Cases im Zustand V innerhalb des zweiten Auswertungszeitraums. Dies ist unter Umständen nicht die vom Benutzer erwartete Semantik. Daher ist es nun möglich, nur beim ersten oder letzten auftretenden Ereignis innerhalb des Auswertungszeitraums die Verweildauer des Cases in einem bestimmten Zustand zu ermitteln.

Der **CountEventsCalculator** erlaubt die Spezifikation von weights, welche die Ermittlung der Case Values beeinflussen. Insbesondere im Hinblick auf die Auswertung wäre es jedoch wünschenswert, wenn statt einem Wert **vektorielle Größen** betrachtet werden könnten. Eine Erweiterung um vektorielle Größen hätte zudem den Vorteil, dass auch die anderen Value Calculators um Gewichte

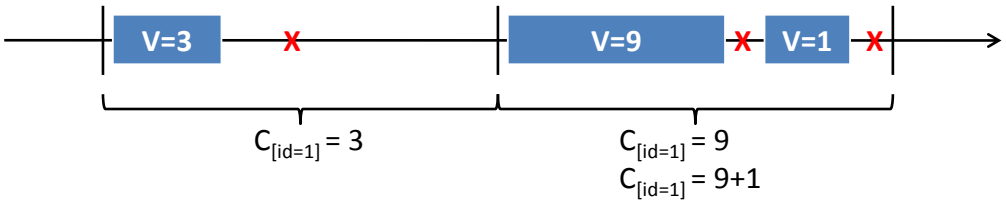


Abbildung 5.1: Beispiel: ResidenceTimeCalculator

ergänzt werden könnten. Beispielsweise wäre eine gleichzeitige Betrachtung von Zeitintervallen und dokumentiertem Zeitaufwand anhand der Detailauswertung möglich.

Bei der zeitlichen Betrachtung berücksichtigen weder der IntervalLengthCalculator noch der ResidenceTimeCalculator die Unterschiede zwischen Arbeitstagen und arbeitsfreien Tagen. Beispielsweise besitzt ein Freitagnacht über das Kundenportal eingestellter Case schon eine Verweildauer von mehreren Tagen ohne Aktivität im Zustand NEW. Dies gilt es bei der Analyse durch die Wahl angemessener Zeitschranken zu berücksichtigen.

Die ermittelten Ergebnisse sind auf ihre Plausibilität hin zu prüfen. Zudem sollten stichprobenartig einzelne Cases zur Validation untersucht werden. Im Verlauf der Diplomarbeit konnten hierdurch Bugs, wie eine fehlerhafte Implementierung des IntervalLengthCalculators oder des Medians, identifiziert und behoben werden.

6 Analyse

Inhalt

Im Folgenden werden die Softwareentwicklungs- und Wartungsprozesse der Firma KISTERS anhand des zuvor entwickelten Qualitätsmodells (siehe 4 KISTERS Qualitätsmodell) analysiert. Die Auswertungssoftware BugzillaMetrics erfasst hierzu die Ausprägung der Qualitätsmerkmale anhand der Informationen aus der Bugzilla Datenbank. Anschließend werden die ermittelten Merkmalsausprägungen entsprechend der Qualitätsindikatoren im Hinblick auf die ausgewählten Qualitätseigenschaften analysiert.

Die Ergebnisse der Analyse betreffen sensible Geschäftsbereiche der Firma KISTERS. Folglich werden die Namen der betrachteten Produkte anonymisiert widergegeben. Sind Ergebnisse der Auswertung unkenntlich zu machen, so wird dies im Text dokumentiert.

6.1 Vorgehensweise bei der Analyse

Die folgende Analyse umfasst einen Betrachtungszeitraum von drei Jahren. Diese Zeitspanne wurde unter Berücksichtigung der Änderungen an den Prozessen gewählt. Der Starttermin berücksichtigt, dass das System zur Klassifikation des Schweregrads umgestellt wurde. Zusätzlich wurde ein angemessener Zeitraum zur Umsetzung der Änderung innerhalb der einzelnen Produkte ergänzt. Folglich sind die betrachteten Prozesse innerhalb des Betrachtungszeitraums hinsichtlich Änderungen und Anpassungen relativ stabil. Um Entwicklungstendenzen zu erkennen und zu veranschaulichen, wird der Betrachtungszeitraum von drei Jahren zur Auswertung in sechs Halbjahre unterteilt.

Bei der Analyse von Softwareentwicklungs- und Wartungsprozessen mithilfe von Change-Request Daten stellt die Quantität der Datenbasis eine besondere Herausforderung dar. Eine manuelle Auswertung der Change-Request Daten ist praktisch unmöglich, daher bedarf es einer leistungsfähigen Werkzeugunterstützung um die Merkmale der Change-Requests automatisiert zu ermitteln. Hierzu wird im Folgenden BugzillaMetrics (siehe [Bugb]) verwendet. Zur Auswertung wird unterstützend Microsoft Excel 2007 SP2 genutzt.

Zur Analyse anhand von empirischen Vergleichen (vgl. 2.4.1 Bestimmung der Qualität durch empirische Vergleiche) wird eine breite Datenbasis benötigt. Sind nicht ausreichend Daten vorhanden, lässt sich die gegebene Datenbasis durch geeignete Datenquellen um zusätzliche Datensätze ergänzen (vgl. [KMS07] [Küt05]). Dies ist im Gegensatz zu betriebswirtschaftlichen Kennzahlen, wie sie für Geschäftsprozesse erhoben werden, für Softwareentwicklungs- und Wartungsprozesse problematisch. Denn obwohl es Prozess- und Vorgehensmodelle (vgl. 2.1.2 Vorgehens- und Prozessmodell) gibt, die abstrakt Ablauf, Rollen, Aktivitäten und Tätigkeiten der Entwicklung und Wartung von Softwareprodukten beschreiben, sind die Prozess- und Vorgehensmodelle individuell durch Tailoring auszuprägen. Daher ist es problematisch, die Datenbasis um vergleichbare Daten zu ergänzen.

Die Anzahl der in KISTERS-Bugzilla gesammelten Change-Requests im Betrachtungszeitraum verhindert, dass die Produkte anhand eines empirischen Vergleiches betrachtet werden können; denn die einzelnen Produkte unterscheiden sich signifikant hinsichtlich der Anzahl ihrer neu eingestellten Cases.

In der Tabelle 6.1 wird die Anzahl der neu eingestellten Cases pro Halbjahr betrachtet. Zur Anonymisierung wird die Anzahl der einzelnen Cases, mit der Anzahl der neueingestellten Cases des Produkts Blau im ersten Halbjahr der Analyse normalisiert. Man sieht deutlich, dass sich die meisten der neu eingestellten Cases auf das Produkt Blau beziehen.

Das Gesamtvolumen **aller neu eingestellten Cases** innerhalb des Betrachtungszeitraums der Produkte Rot und Grün entspricht 31% bzw. 15% **aller neu eingestellten Cases** des Produkts Blau. Betrachtet man die Produkte A bis G so entspricht deren Gesamtvolumen 3% bis 8% des Produkts Blau. Zudem ist die absolute Anzahl der neueingestellten Cases der Produkte A bis G, so gering, dass ein empirischer Vergleich aufgrund der Anzahl der Change-Requests nicht sinnvoll ist. So ergibt sich zum Beispiel der Sachverhalt, dass die Verletzung von 25% der Deadlines für ein Produkt mit vier Cases anders zu bewerten ist als für ein Produkt mit 100 Cases. Die Cases sind zudem entsprechend ihres Typs (Bug, Enhancement oder SupportCall) zu unterscheiden, so dass sich die Situation hinsichtlich des Datenumfangs weiterverschärft. Folglich ist der Datenumfang für eine Analyse anhand eines empirischen Vergleiches zu gering.

Ein Vergleich der verschiedenen Produkte hinsichtlich eines Referenzprodukts ist ebenfalls aufgrund des unterschiedlichen Datenvolumens ungeeignet. Zudem muss bei der Wahl des Referenzprodukts der Charakter der Entwicklung ausreichend berücksichtigt werden. Das Produkt Grün repräsentiert innerhalb des Betrachtungszeitraums eine Neuentwicklung, die sich schon im Kundentest befindet. Folglich werden insbesondere in den ersten Auswertungszeiträumen typischerweise noch sehr wenige Change-Request Daten erfasst. Der Charakter des Produkts Blau lässt sich als Weiterentwicklung beschreiben. Entsprechend bleibt die Anzahl der Bugs und Enhancements stabil. Hingegen ist der Charakter des Produkts Rot zunehmend durch Wartungsaufgaben geprägt, was in

Produkt	1. Halbjahr	2. Halbjahr	3. Halbjahr	4. Halbjahr	5. Halbjahr	6. Halbjahr
Blau	100%	119%	157%	152%	190%	189%
Rot	51%	53%	49%	49%	39%	34%
Grün	5%	16%	8%	31%	36%	36%
A	14%	10%	10%	5%	16%	13%
B	1%	3%	7%	4%	10%	10%
C	4%	8%	11%	9%	11%	10%
D	13%	12%	10%	12%	8%	9%
E	5%	2%	4%	7%	4%	5%
F	4%	8%	7%	4%	4%	5%
G	13%	10%	13%	13%	5%	4%

Tabelle 6.1: Anzahl der neu eingestellten Cases für die einzelnen Produkte pro Halbjahr; normalisiert mit der Anzahl der neu eingestellten Cases des Produkts Blau im ersten Halbjahr

dem Rückgang der Anzahl der Enhancements zu beobachten ist. Die anderen Produkte sind aufgrund ihres zu geringen Datenvolumens für die Betrachtung nicht geeignet.

Aufgrund der Unterschiede hinsichtlich des Datenumfangs werden im Folgenden nur die drei Produkte Blau, Rot und Grün betrachtet und aufgrund ihres verschiedenen Charakters hinsichtlich der Entwicklung gesondert im Hinblick auf ihre Entwicklungstendenzen analysiert.

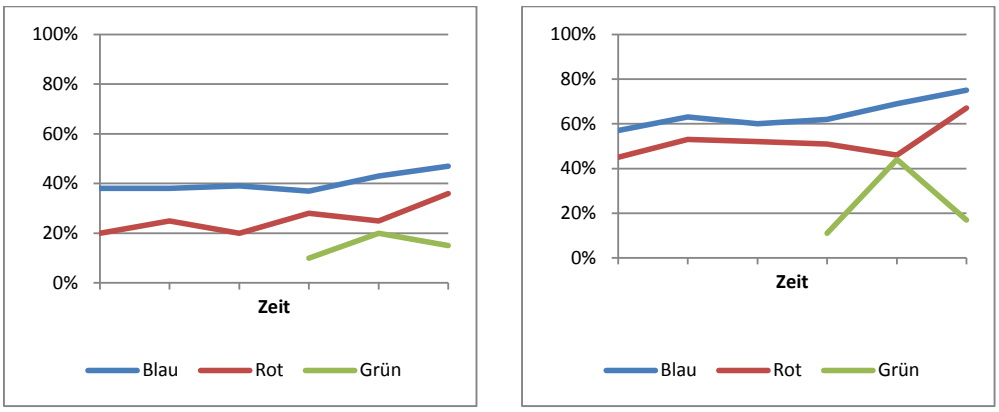
6.2 Analyse anhand des bidirektionalen Qualitätsmodells

Die Analyse für die Produkte Blau, Rot und Grün erfolgt anhand des in Kapitel 4 entwickelten bidirektionalen Qualitätsmodells im Hinblick auf die individuellen Entwicklungstendenzen. Hierzu werden im Folgenden die Ergebnisse der einzelnen Qualitätsindikatoren ausgewertet und anschließend deren Interpretation im Kapitel 7 diskutiert. Die Beschreibung der Qualitätsindikatoren und ihre Beziehung zu den Qualitätseigenschaften wurde im Kapitel 4.3 erläutert. Die nachfolgende Auswertung der Qualitätsindikatoren erfolgt anhand ihrer Beziehung zu den einzelnen Qualitätseigenschaften. Die Qualitätsindikatoren, welche sich auf mehr als eine Qualitätseigenschaft beziehen, werden nur einmal aufgeführt.

6.2.1 Management der Anforderungen

Im Rahmen des Managements der Anforderungen wird versucht ein gemeinsames Verständnis bezüglich der technischen und nicht-technischen Anforderungen zwischen Kunden und Auftraggebern zu erreichen.

Einen informativen Charakter besitzt der Qualitätsindikator (01), welcher den Anteil der Enhancements beim Zustandsübergang nach RESOLVED+FIXED bezogen auf Enhancements und Bugs ermittelt. Der Anteil diesbezüglich wird im Folgenden zum einen anhand der Anzahl der Cases und zum anderen basierend auf dem erfassten Zeitaufwand bestimmt. In der Abbildung 6.1 (a) wird der Anteil anhand der Anzahl der Cases bestimmt. Die Produkte Blau und Rot besitzen einen Seitwärtstrend mit einer leicht ansteigenden Aufwärtstendenz. Betrachtet man die Entwicklung bezüglich des erfassten Zeitaufwands (siehe Abb. 6.1 (b)), so zeigen sowohl das Produkt Blau als auch das Produkt Rot einen Aufwärtstrend.



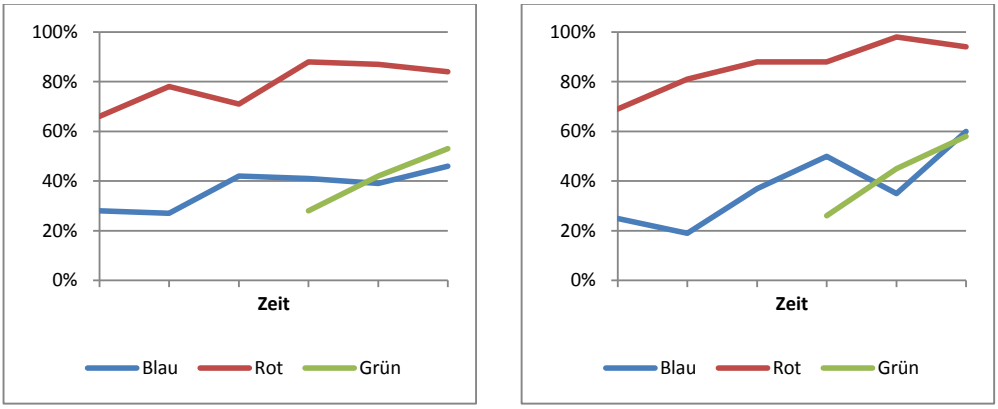
(a) Anzahl der Cases

(b) erfasster Zeitaufwand

Abbildung 6.1: Anteil der Enhancements bezogen auf die Summe von Bugs und Enhancements

Der Qualitätsindikator (02) bestimmt den Anteil der Cases vom Typ Enhancement mit Hinweis auf externe Auftraggeber beim Zustandsübergang nach RESOLVED+FIXED. Dieser lässt sich zum einem anhand der Anzahl der Cases und zum anderen mittels des geleisteten Aufwands ermitteln (siehe Abb. 6.2). Der Anteil der Enhancements von externen Auftraggebern beim Zustandsübergang nach RESOLVED+FIXED steigt bei allen drei Produkten im Betrachtungszeitraum unabhängig von der Metrik.

Die korrekte Typisierung ist für die Klassifikation und Betrachtung der einzelnen Cases von elementarer Bedeutung. Der Anteil der Cases mit Typ-Wechsel beim ersten Zustandsübergang nach RESOLVED wird im Qualitätsindikator (03) gemessen (siehe Abb. 6.3). Betrachtet man den Anteil der Cases mit Typ-Wechsel beim Zustandsübergang nach RESOLVED, erkennt man hinsichtlich der Anteile bei den Produkten Blau und Grün einen deutlichen Aufwärtstrend. Der Anteil des Produkts Rot besitzt einen Seitwärtstrend.



(a) Anzahl der Cases (b) erfasster Zeitaufwand

Abbildung 6.2: Anteil der Cases mit externem Kundebezug bezogen auf alle Enhancements

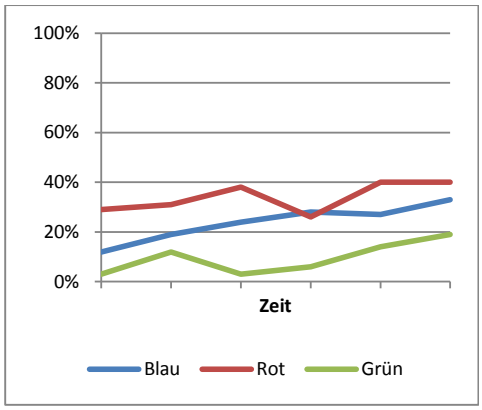
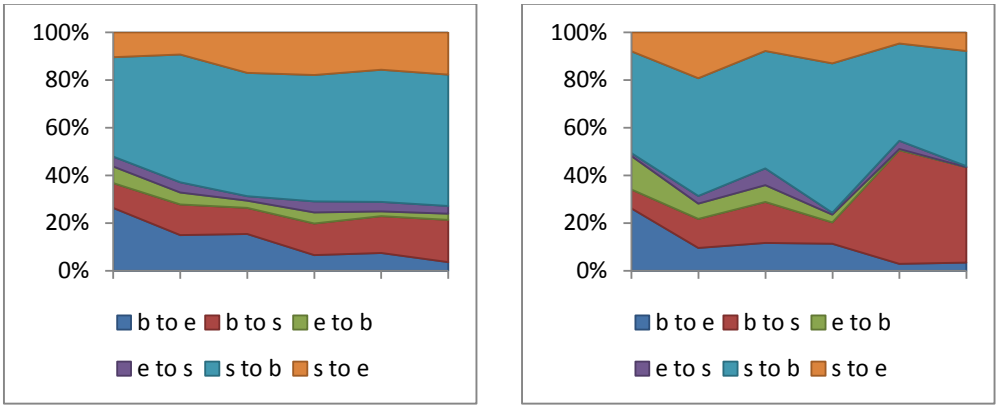


Abbildung 6.3: Anteil der Cases, deren Typ-Klassifikation sich geändert hat

Mithilfe des Qualitätsindikators (04) werden die Wechsel zwischen den Typ Klassifikationen Bug, Enhancement und SupportCall betrachtet. Hierzu wurden die Wechsel zwischen den einzelnen Klassifikationen in den Zuständen NEW, WAITCUSTINFO, ASSIGNED, PROCESSING und REOPENED ermittelt. In Abbildung 6.4 sind für die Produkte Blau und Rot die Anteile der einzelnen Wechsel der Typ Klassifikation bezogen auf die Gesamtzahl der Cases mit geänderter Typ Klassifikation dargestellt. Hinsichtlich der Produkte Blau und Rot fällt der hohe Anteil an Typ-Wechseln von SupportCall nach Bug auf. Im Betrachtungszeitraum nimmt der Anteil der Typ-Wechsel von Bug nach Enhancement für die Produkte Blau und Rot ab. In den beiden letzten Betrachtungsperioden von Rot steigt der Anteil der Cases, deren Klassifikation sich von Bug nach SupportCall ändert.



(a) Produkt Blau (b) Produkt Rot

Abbildung 6.4: Typ-Wechsel in den offenen Zuständen + REOPENED

Einen Hinweis auf dokumentierten Mehraufwand durch parallele oder doppelte Bearbeitung gibt der Anteil der Cases, welche die Zustandsmengen ASSIGNED, PROCESSING besucht haben, bevor sie als DUPLICATE gekennzeichnet wurden, an. Der Anteil hinsichtlich des Produkts Blau besitzt eine Aufwärtstendenz (siehe Abb. 6.5). Hingegen besitzt das Produkt Rot eine volatile Seitwärtsbewegung.

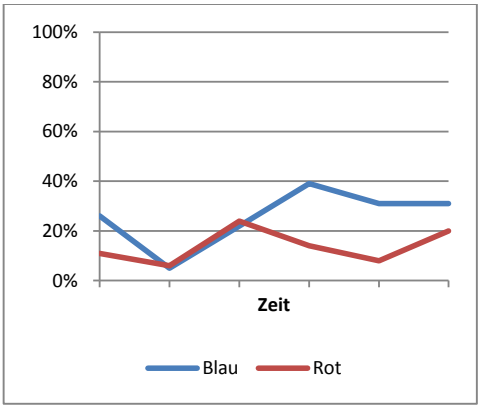


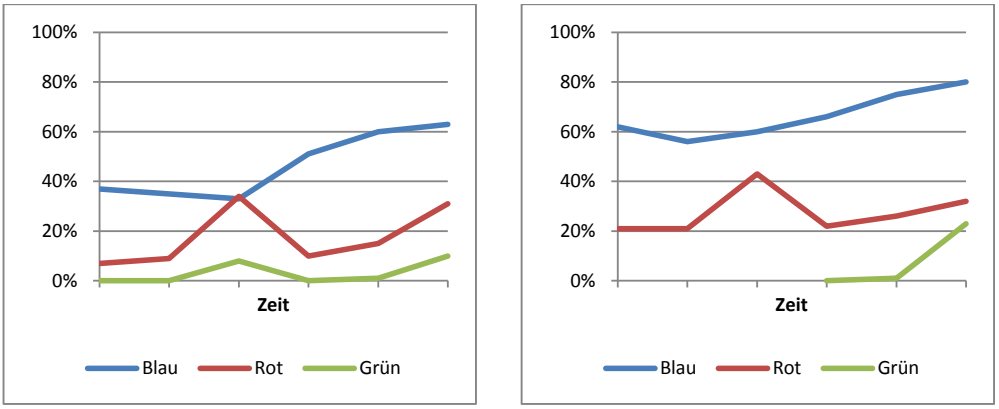
Abbildung 6.5: Anteil der Cases mit Hinweis auf parallele oder doppelte Arbeit

6.2.2 Planung

Im Rahmen der Planungstätigkeiten werden in Bugzilla sowohl zeitliche Aspekte als auch Arbeitsaufwände dokumentiert. Diese sind hinsichtlich des Anteils an Planungsaktivität und der Genauigkeit der Planungsaktivität zu untersuchen.

Zuerst soll der zeitliche Aspekt untersucht werden. Hierzu werden die Qualitätsindikatoren (06) Anteil der Cases, deren Bearbeitungsbeginn geplant wurde, (siehe Abb. 6.6), und (07) Anteil der Cases mit dokumentierter Deadline (siehe

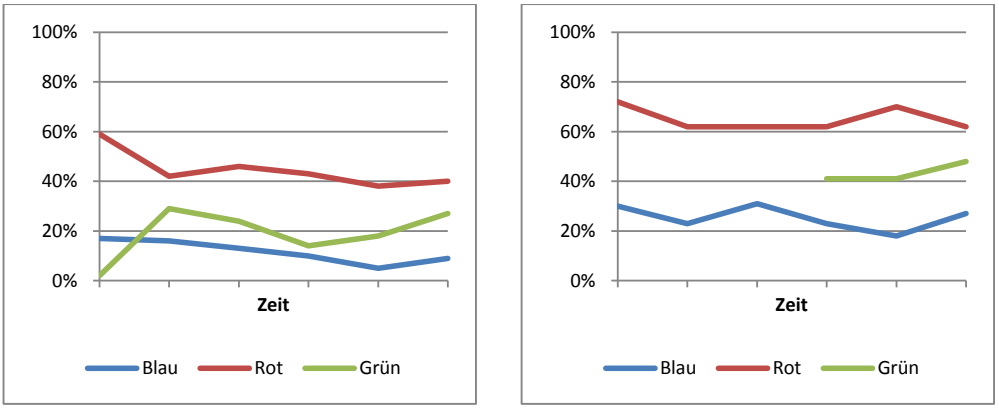
Abb. 6.7) betrachtet. Der Beginn der Bearbeitung wird als Kalenderwoche im Case festgehalten. Der Anteil der Cases mit geplantem Bearbeitungsbeginn zeigt für die Bugs des Produkts Blau im Betrachtungszeitraum einen deutlichen Aufwärtstrend. Ebenso zeigt der geplante Anteil der Bugs des Produkts Rot einen Aufwärtstrend, mit einem Zwischenhoch für den dritten Betrachtungszeitraum. Für das Produkt Grün werden kaum Bearbeitungstermine dokumentiert. Für die Enhancements des Produkts Blau steigt der Anteil der dokumentierten Bearbeitungstermine. Bezogen auf die Enhancements des Produkts Rot bleibt der Anteil stabil.



(a) Bugs (b) Enhancements

Abbildung 6.6: Anteil der Cases mit geplantem Bearbeitungsbeginn

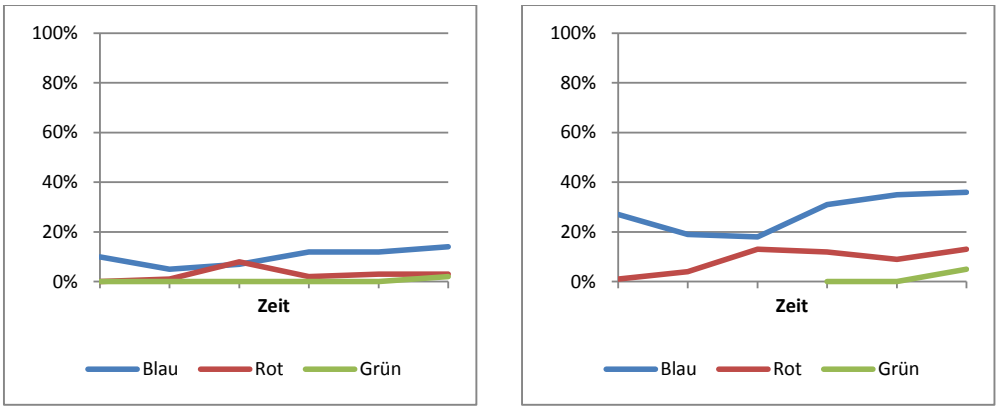
Die Auswertung hinsichtlich des Qualitätsindikators (07) Anteil der Cases mit geplanter Deadline ist in Abbildung 6.7 dargestellt. Betracht man die Cases der Typen Bug und Enhancement, so zeigt der Anteil für alle drei betrachteten Produkte einen Seitwärtstrend auf unterschiedlichen Niveaus, wobei die Cases des Typs Enhancement ein deutlich höheres Niveau aufweisen.



(a) Bugs (b) Enhancements

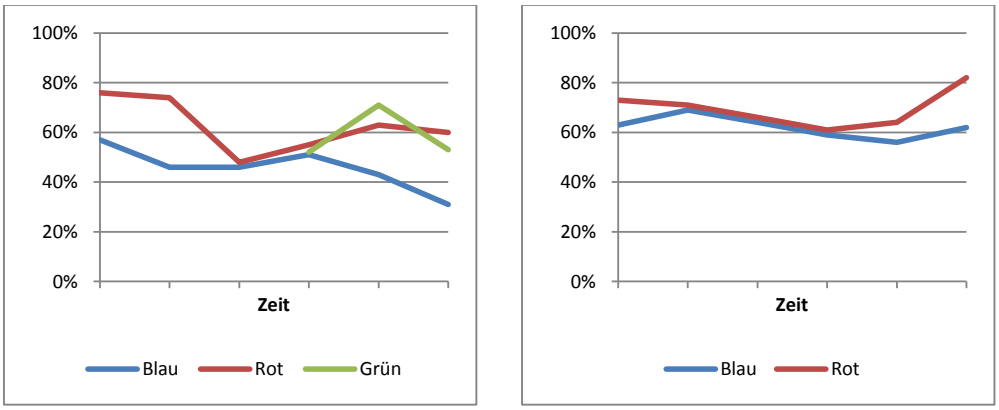
Abbildung 6.7: Anteil der Cases mit dokumentierter Deadline

Neben dem Planungsumfang ist die Planungsgenauigkeit zu berücksichtigen. Hierzu bestimmt der Qualitätsindikator (08) den Anteil der Cases, deren geplanter Bearbeitungsbeginn verschoben wurde (siehe Abb. 6.8). Für die Cases vom Typ Bug werden Änderungen hinsichtlich des Bearbeitungstermins nur in geringem Umfang dokumentiert. Der Anteil der Cases vom Typ Bug und Enhancement zeigt für die einzelnen Produkte einen Seitwärtstrend.



(a) Bugs (b) Enhancements

Abbildung 6.8: Anteil der Cases, deren Bearbeitungsbeginn verschoben wurde



(a) Bugs (b) Enhancements

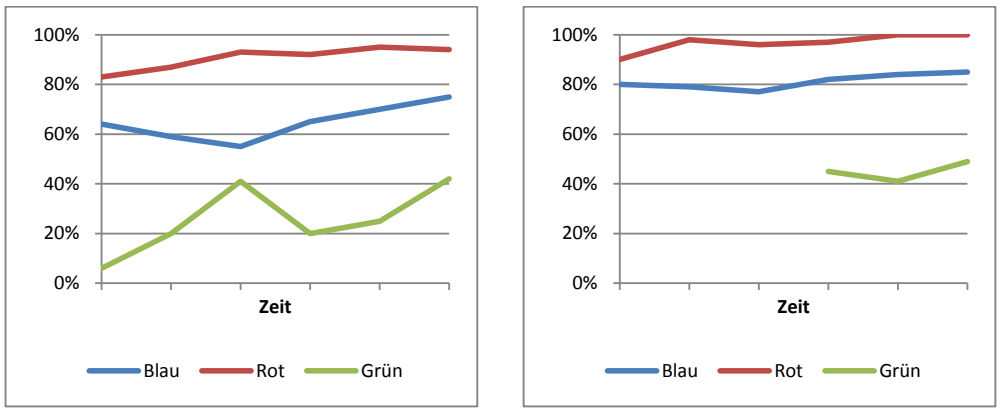
Abbildung 6.9: Anteil der Cases, deren Deadline verletzt wurde (RESOLVED +FIXED)

Bei der Planung einer Deadline muss ausreichend Zeit für alle Aktivitäten des Softwareentwicklungs- und Wartungsprozesses berücksichtigt werden. Der im Folgenden betrachtete Qualitätsindikator (09) betrachtet den Anteil der Cases der verletzten Deadlines beim Zustandsübergang nach RESOLVED+FIXED im Bezug auf alle Cases mit einer Deadline (siehe Abb. 6.9). Bezogen auf die Cases vom Typ Bug zeigen die Produkte Blau und Rot einen deutlichen Abwärtstrend. Aufgrund des geringen Volumens ist für das Produkt Grün nur eine Einordnung hinsichtlich der Bugs möglich. Die betrachteten Anteile des Produkts Grün sind

denen des Produkts Rot ähnlich. Für die Cases vom Typ Enhancement zeigen sowohl die Anteile des Produkts Blau als auch die des Produkts Rot einen Seitwärtstrend, wobei am Ende des Betrachtungszeitraums ein Anstieg für das Produkt Rot zu beobachten ist.

Des Weiteren lässt sich die Planungstätigkeit bezüglich des geplanten und geleisteten Zeitaufwands analysieren. Dazu wird zuerst der Qualitätsindikator (10) Anteil der Cases mit geschätztem Zeitaufwand und anschließend der Qualitätsindikator (11) die Genauigkeit der Schätzung unter Berücksichtigung des dokumentierten geleisteten Zeitaufwands betrachtet.

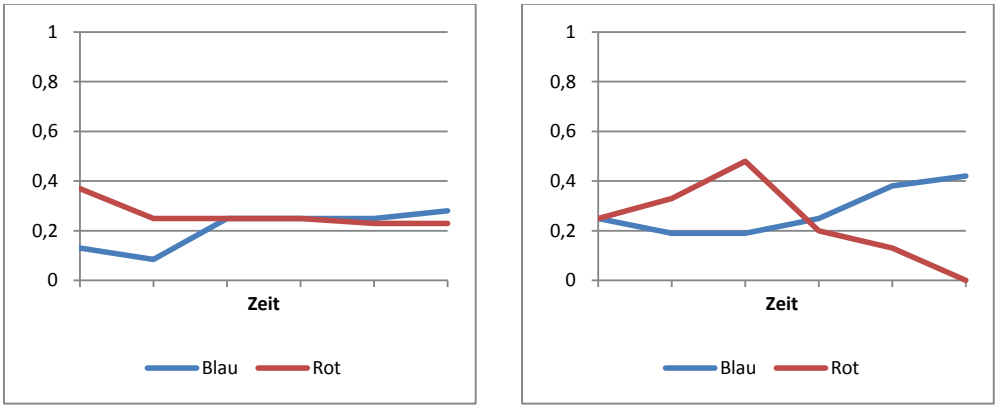
Der Anteil der Cases vom Typ Bug mit geschätztem Zeitaufwand steigt beim Produkt Blau im Betrachtungszeitraum an, während sich beim Produkt Rot der Trend auf hohem Niveau verfestigt (siehe Abb. 6.10). Das Produkt Grün besitzt einen deutlichen Aufwärtstrend, der jedoch volatil ist. Die Entwicklungstendenz der Cases vom Typ Enhancement entspricht einem Seitwärtstrend für die verschiedenen betrachteten Produkte. Für das Produkt Rot ist der Anteil in den letzten Betrachtungsperioden auf höchstem Niveau. Der Anteil der geplanten Enhancements bezüglich des Aufwands ist im Allgemeinen höher als bei den Bugs.



(a) Bugs (b) Enhancements

Abbildung 6.10: Anteil der Cases mit geschätztem Zeitaufwand

Zur Betrachtung der Planungsgenauigkeit hinsichtlich des geschätzten Zeitaufwands ist der Median des Gewichts `originalEffortEstimationAccuracy` (vgl. Beschreibung des Qualitätsindikators (11)) zu betrachten. Folglich bedeutet eine Merkmalsausprägung von 1 in der Abbildung 6.11, dass die Schätzung mit dem geleisteten Aufwand übereinstimmt. Für die Bugs zeigt sich ein positiver Entwicklungstrend für das Produkt Blau und ein Seitwärtstrend für das Produkt Rot. Der positive Entwicklungstrend des Produkts Blau zeigt sich ebenfalls bezüglich der Cases vom Typ Enhancement. Auffällig ist der uneinheitliche Trend bei der Betrachtung der Enhancements des Produkts Rot. Nach einem positiven Entwicklungstrend bis zur Mitte des Betrachtungszeitraums setzt eine deutliche Trendumkehr ein.

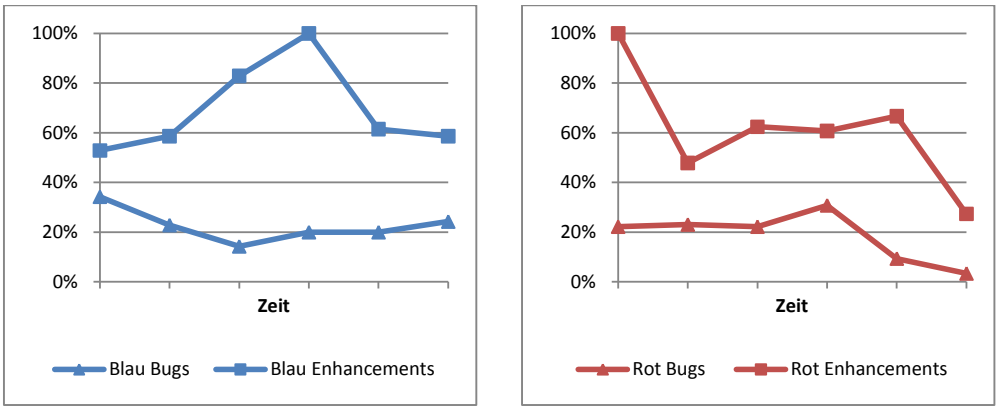


(a) Bugs (b) Enhancements

Abbildung 6.11: Median des `originalEffortEstimationAccuracy` Gewichts

Der Qualitätsindikator (12) betrachtet das Verhältnis des verbleibenden Aufwands aller noch offenen Cases zum erledigten Aufwand einer Periode. Eine Ausprägung von 1 bedeutet, dass der verbleibende Aufwand der eingestellten Cases bei ceteris paribus Annahme in einer Periode erledigt werden kann. Folglich bedeutet eine Ausprägung kleiner dem Wert 1, dass der verbleibende Aufwand der eingestellten Cases in weniger als einer Periode bearbeitet werden kann. Dieser Qualitätsindikator besitzt einen rein informativen Charakter über den dokumentierten, geschätzten verbleibenden Aufwand und gibt einen Hinweis auf noch ausstehende Arbeiten.

Die Ergebnisse dieser Metrik sind in **anonymisierter** Form wiederzugeben (siehe Abb. 6.12), so dass die zeitlichen Aspekte unkenntlich gemacht werden. Als Normalisierungsgröße wurde für die jeweiligen Produkte der größte beobachtete Wert genommen.



(a) Produkt Blau (b) Produkt Rot

Abbildung 6.12: Anonymisierte Darstellung des Qualitätsindikators (12)

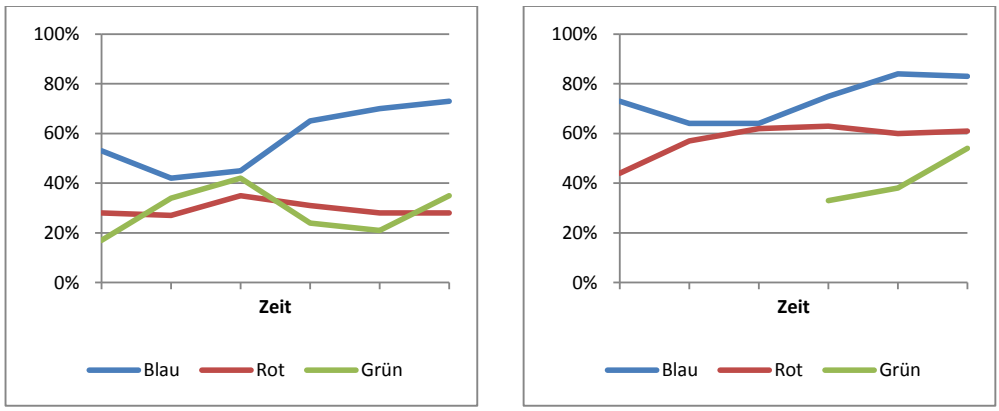
Für die Bugs des Produkts Blau und Rot besitzt der Qualitätsindikator (12) eine fallende Tendenz. Ebenso weist der Qualitätsindikator für die Enhancements des Produkts Rot eine fallende Tendenz auf, wohingegen ein volatiler uneinheitlicher Trend bezüglich des Produkts Blau zu erkennen ist.

6.2.3 Überwachung und Verfolgung des Entwicklungsfortschritts

Der Projektfortschritt im Projekt muss zur Überwachung und Verfolgung sichtbar gemacht werden. Ansonsten ist nur eine unzureichende Steuerung des Projekts möglich, da der Regelungskreis zur Projektsteuerung nicht geschlossen werden kann.

Der Entwicklungsfortschritt lässt sich in Bugzilla anhand der einzelnen Cases nachvollziehen. Dies geschieht in Bugzilla sowohl durch die dokumentierten Zustandsübergänge als auch durch die Erfassung der geleisteten Arbeitsstunden. Die Bearbeitung des Cases durch einen Entwickler wird dokumentiert, indem mindestens ein Zustand aus der Menge {ASSIGNED, PROCESSING} besucht wird. Die Entwicklung des Cases wird durch den Zustandswechsel nach RESOLVED als abgeschlossen dokumentiert. Diesbezüglich lassen sich der Qualitätsindikator (13) Anteil der Cases mit dokumentierter Nutzung von ASSIGNED und PROCESSING und (14) Anteil der Case mit dokumentierter Erfassung der Arbeitszeiten auswerten.

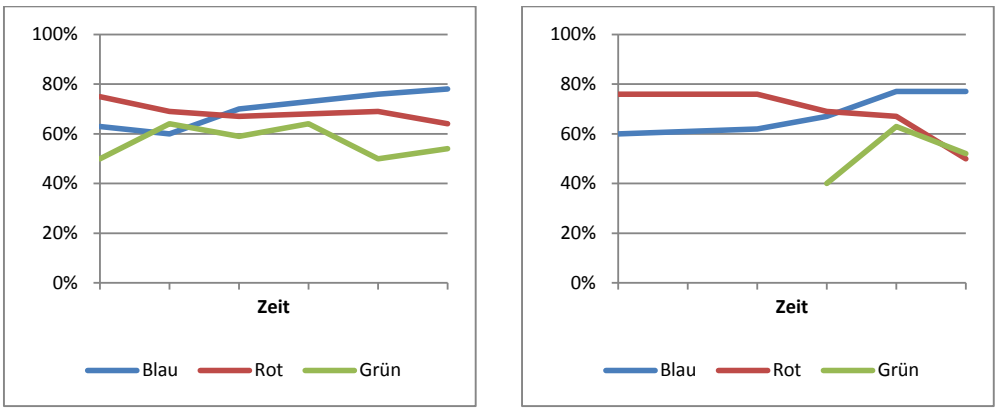
Die Abbildung 6.13 zeigt die Auswertung des Qualitätsindikators (13) getrennt für a) Bugs und b) Enhancements. Betrachtet man die Bugs, so besitzt das Produkt Blau einen steigenden Anteil hinsichtlich der Cases mit dokumentierter Zustandsnutzung. Das Produkt Rot verbleibt auf gleichem Niveau. Ebenfalls besitzt das Produkt Grün einen volatilen Seitwärtstrend. Im Betrachtungszeitraum besitzen die Enhancements des Produkts Rot einen Seitwärtstrend. Der Seitwärtstrend für das Produkt Blau ist volatiler.



(a) Bugs (b) Enhancements

Abbildung 6.13: Anteil der Cases mit dokumentierter Nutzung von ASSIGNED und PROCESSING

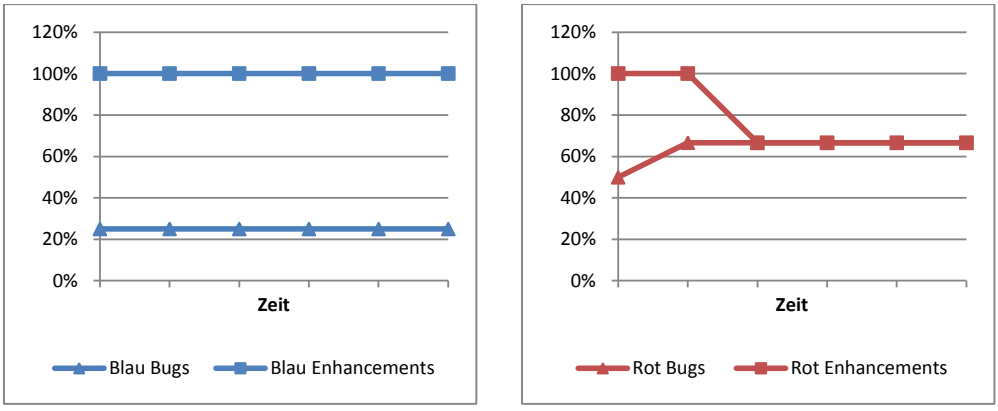
Während die Dokumentation anhand der Zustände aus der Menge {ASSIGNED, PROCESSING} einen Überblick der aktuellen Entwicklungsaktivitäten bietet, lässt sich der Entwicklungsfortschritt anhand des geleisteten Arbeitsaufwands überwachen. Um den Entwicklungsfortschritt quantifizieren zu können, bedarf es des geschätzten Entwicklungsaufwands. Daher wird nur der Anteil der Cases mit dokumentiertem geschätztem Arbeitsaufwand berücksichtigt (siehe Abb. 6.14). Im Produkt Blau steigt der Anteil der Bugs, die den geleisteten Arbeitsaufwand dokumentieren. Hingegen fällt der Anteil für das Projekt Rot bezogen auf die Bugs im Betrachtungszeitraum. Das Produkt Grün besitzt einen uneinheitlichen Trend. Bezüglich der Enhancements Cases zeigen sich ähnliche Trends wie für die Produkte Blau und Rot.



(a) Bugs (b) Enhancements

Abbildung 6.14: Anteil der Cases mit dokumentierter Erfassung der Arbeitszeiten

Um Unterschiede hinsichtlich der Granularität des geschätzten Zeitaufwands der Cases zu bestimmen, lässt sich der Qualitätsindikator (15) Granularität der Cases analysieren. Dieser bestimmt den Median des geschätzten Zeitaufwands für die Cases innerhalb des Betrachtungszeitraums.



(a) Produkt Blau (b) Produkt Rot

Abbildung 6.15: Anonymisierte Darstellung des Qualitätsindikators (15)

Die ermittelten Ergebnisse sind in **anonymisierter** Form dargestellt. Hierzu wurden die ermittelten Ergebnisse der jeweiligen Betrachtung der Produkte mit dem größten Ergebnis für das betrachtete Produkt normalisiert.

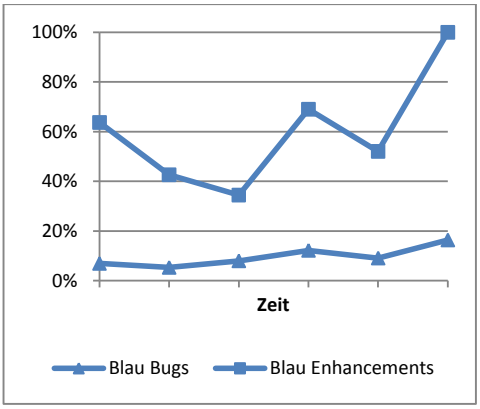
Offensichtlich besteht zwischen den Bugs und den Enhancements des Produkts Blau ein deutlicher Unterschied hinsichtlich des geschätzten Zeitaufwands (siehe Abb. 6.15). Der Median des geschätzte Zeitaufwands für die Bugs und Enhancements des Produkts Rot gleicht sich innerhalb der ersten Perioden des Betrachtungszeitraums an.

6.2.4 Lösungsgeschwindigkeit

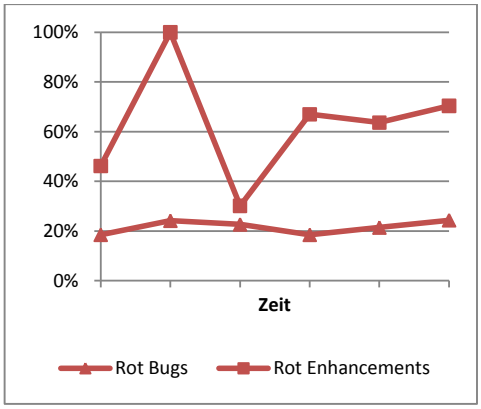
Im Rahmen der Lösungsgeschwindigkeit wird die Zeit zwischen dem Einstellen des Cases bis zu seiner Lösung beziehungsweise der Abnahme durch den Kunden ermittelt. Dementsprechend werden die Qualitätsindikatoren (16) TimeToSolution und (17) TimeToCustAccept ausgewertet. Die Lösungsgeschwindigkeit ist unabhängig von dem getrennt erfassten geleisteten Aufwand zu betrachten.

Im Folgenden müssen sowohl der Qualitätsindikator (16) als auch der Qualitätsindikator (17) in **anonymisierter** Form wiedergeben werden, so dass die zeitlichen Aspekte unkenntlich gemacht werden. Die ermittelten Ergebnisse werden anhand des größten beobachteten Wertes des jeweiligen Produkts normalisiert.

Der Qualitätsindikator (16) TimeToSolution bestimmt den Median des Zeitintervalls zwischen dem Einstellen der Cases und deren ersten Zustandsübergang nach RESOLVED+FIXED. In Abbildung 6.16 werden in **anonymisierter** Form die ermittelten Ergebnisse für Bugs und Enhancements der Cases mit der Priorität P2 getrennt nach Produkten dargestellt.



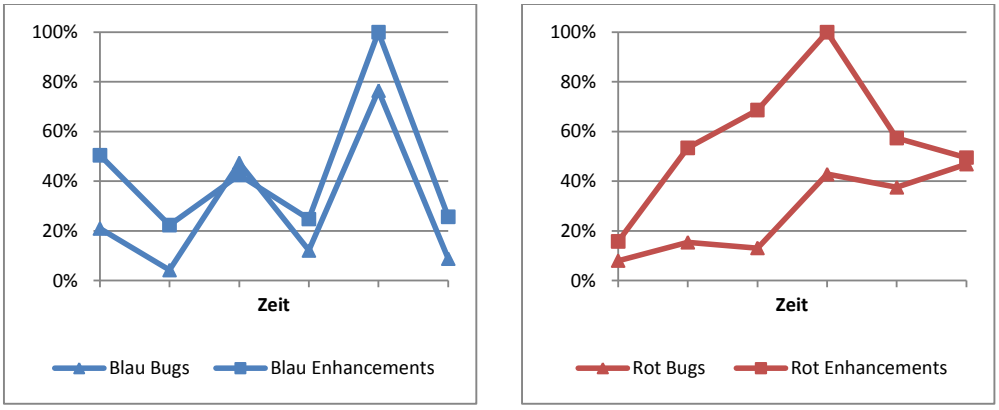
(a) Produkt Blau



(b) Produkt Rot

Abbildung 6.16: Anonymisierte Darstellung des Qualitätsindikators (16)

Während der Qualitätsindikator (16) den Median der Zeitintervalle bis zum ersten Zustandsübergang nach RESOLVED+FIXED bestimmt, wird der Qualitätsindikator (17) anhand des ersten Zustandsübergangs nach CLOSED+FIXED ermittelt. Hierzu bietet sich ebenfalls die Betrachtung der **anonymisierten** Darstellung von Bugs und Enhancements getrennt nach Produkten an (siehe Abb. 6.17).



(a) Produkt Blau (b) Produkt Rot

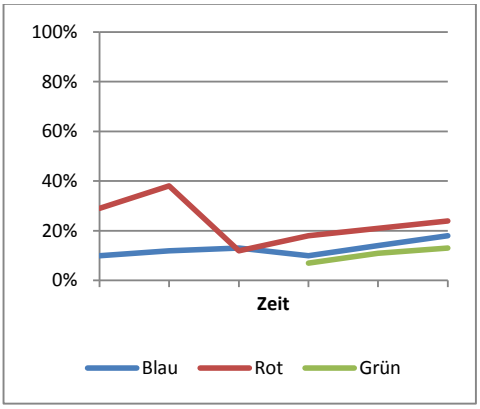
Abbildung 6.17: Anonymisierte Darstellung des Qualitätsindikators (17)

6.2.5 Software-Qualitätssicherung

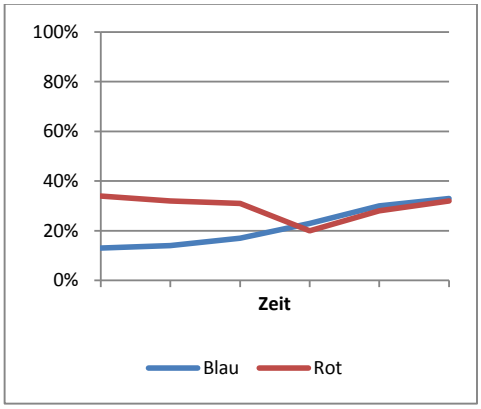
In der Software-Qualitätssicherung werden die einzelnen Artefakte der Softwareentwicklung fortlaufend geprüft und deren Qualität dokumentiert. Dies geschieht, indem gefundene Fehler dokumentiert oder Nacharbeiten angeordnet werden.

In Abbildung 6.18 wird die Anzahl der Zustandswechsel nach REOPENED mit mehr als einer Stunde Verweildauer, normalisiert mit den eindeutigen Zustandswechseln nach RESOLVED+FIXED für die Cases mit der Priorität P2 dargestellt. Die Cases vom Typ Bug weisen ab der dritten Periode einen Seitwärtstrend mit leicht ansteigenden Anteilen auf. Hinsichtlich der Enhancements des Produkts Blau zeigt sich ein Aufwärtstrend. Bei den Enhancements des Produkts Rot wird das Niveau gehalten.

Der Qualitätsindikator (19) bestimmt den Median der Zeitintervalle zwischen dem Zustandsübergang nach REOPENED und dem Verlassen des Zustands REOPENED (siehe Abb. 6.19). Zusätzlich wurden nur Cases berücksichtigt, deren Verweildauer im Zustand REOPENED mindestens eine Stunde beträgt (siehe 7 Diskussion). Im Folgenden wurden nur Cases mit der Priorität P2 betrachtet.

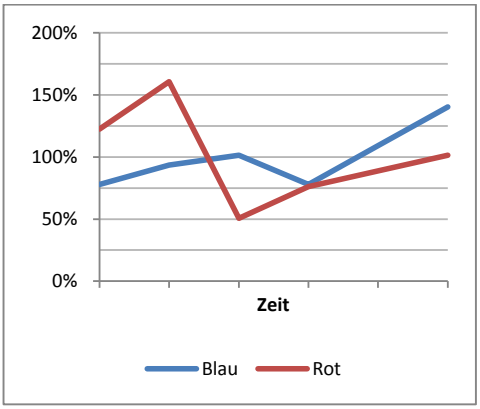


(a) Bugs

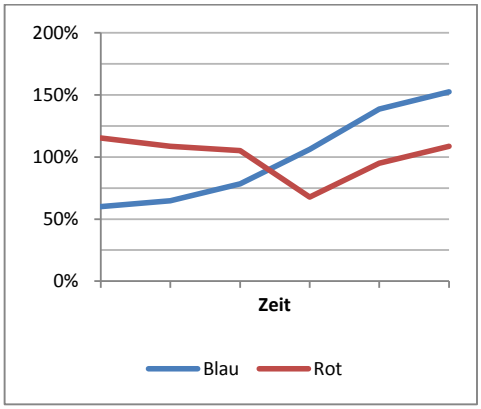


(b) Enhancements

Abbildung 6.18: Anteil der Nacharbeiten innerhalb der Perioden



(a) Bugs



(b) Enhancements

Abbildung 6.19: Anonymisierte Darstellung des Qualitätsindikators (19)

Die Darstellung der Ergebnisse des Qualitätsindikators (19) geschieht in **anonymisierter** Form. Hierzu werden die Ergebnisse der einzelnen Produkte anhand ihres über den Betrachtungszeitraum berechneten arithmetischen Mittelwerts normalisiert.

Für das Produkt Blau zeigt sich in den letzten Betrachtungsperioden ein Aufwärtstrend. Die Enhancements des Produkts Rot besitzen bezüglich des Median einen Seitwärtstrend.

Die Defect Escape Rate bestimmt den Anteil der gemeldeten Fehler mit dokumentiertem externem Unternehmensbezug im Verhältnis zu allen gemeldeten Fehlern einer Periode. In Abbildung 6.20 wird hierzu der Qualitätsindikator (20) betrachtet. Offensichtlich wird der überwiegende Anteil der Fehlermeldungen ohne Hinweis auf einen externen Unternehmensbezug eingestellt.

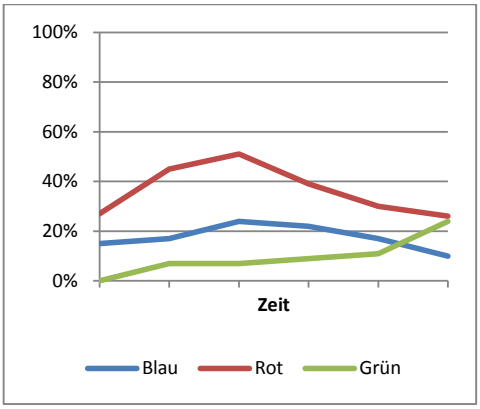
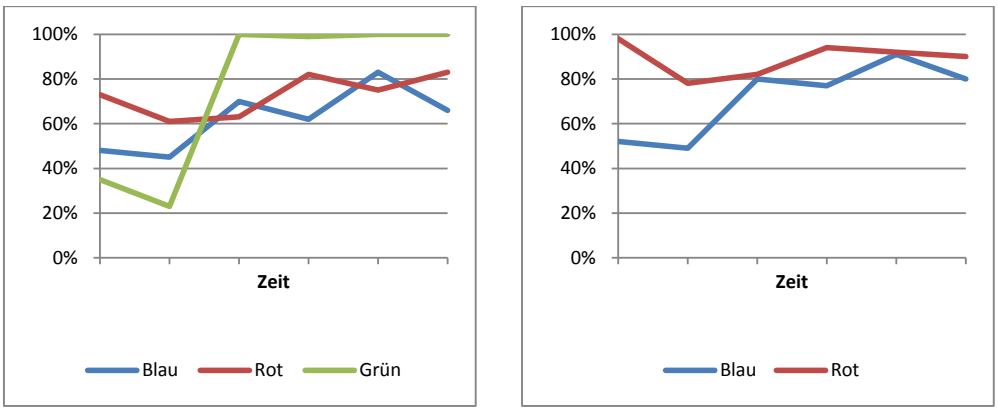


Abbildung 6.20: Defect Escape Rate

6.2.6 Software-Konfigurationsmanagement

Die verschiedenen Artefakte der Softwareentwicklung sind dem Konfigurationsmanagement unterstellt. Dieses erlaubt eine systematische Kontrolle der Änderungen an der Konfiguration und dokumentiert die Konfiguration der an den Kunden gelieferten Produkte.

Im `targetMilestone` soll nach der Bearbeitung das Release dokumentiert werden. Die Abbildung 6.21 betrachtet die Auswertung hinsichtlich des Qualitätsindikators (21) Anteil der Cases mit `targetMilestone` bei Zustandsübergang nach CLOSED+FIXED. Insbesondere in den späteren Betrachtungszeiträumen wird der überwiegende Anteil der Cases mit einem `targetMilestone` dokumentiert. Die Anteile der Enhancements liegen über denen der Bugs.

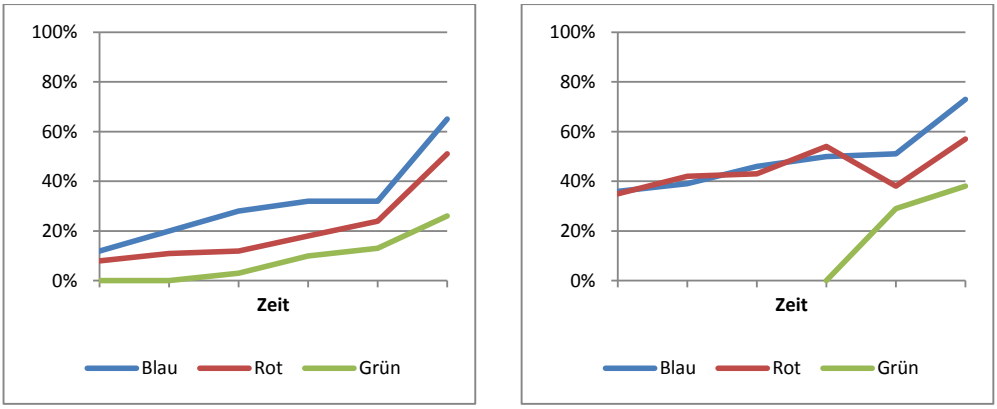


(a) Bugs (b) Enhancements

Abbildung 6.21: Anteil der Cases mit `targetMilestone`

Die Änderungen am ausgelieferten Produkt werden mittels der `releaseNotes` dokumentiert. Mithilfe des Qualitätsindikators (22) lässt sich der Anteil der Cases mit `releaseNotes` beim ersten Zustandsübergang nach CLOSED+FIXED

bestimmen (siehe Abb. 6.22). Betrachtet man die Bugs und Enhancements, so steigt der Anteil aller Produkte innerhalb des Betrachtungszeitraums. Das Niveau der Enhancements liegt im Allgemeinen höher als das der Bugs. Insbesondere im letzten Betrachtungszeitraum ist ein Anstieg zu erkennen.



(a) Bugs (b) Enhancements

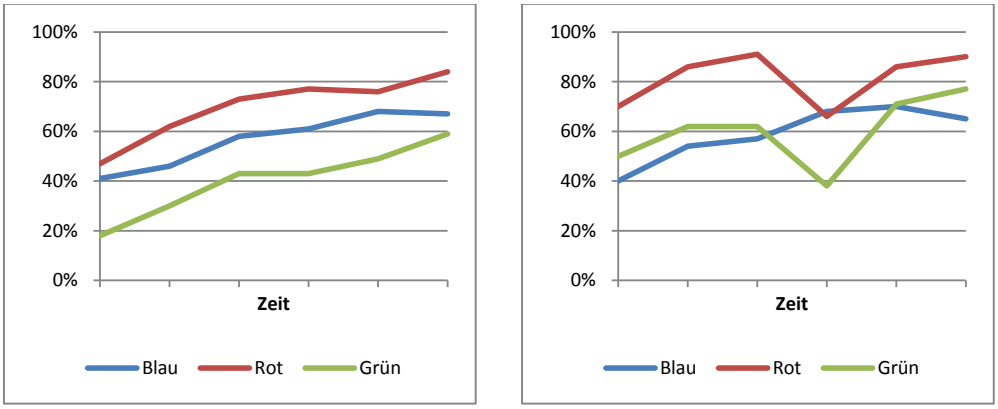
Abbildung 6.22: Anteil der Cases mit `releaseNotes`

6.2.7 Koordination der beteiligten Anspruchsgruppen

Zur Realisierung von IT-Projekten und IT-Produkten müssen die verschiedenen Bedürfnisse der Anspruchsgruppen koordiniert werden. Einige Attribute der Cases geben Hinweise auf die Effizienz der verwendeten Koordinationsprozesse.

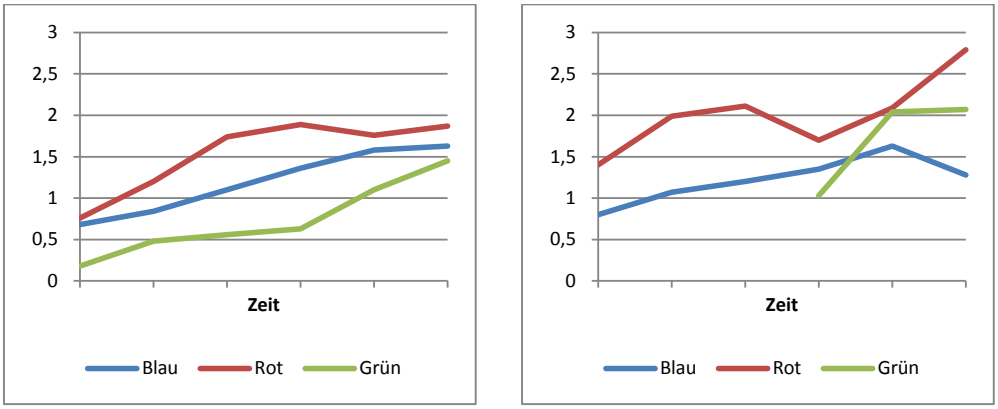
In der CC-Liste sind die Personen registriert, die neben dem reporter, assignee und qaContract über Änderungen an dem Case automatisch informiert werden. Der Qualitätsindikator (23) Anteil der Cases mit Einträgen in der CC-Liste wird im Folgenden getrennt für Bugs und Enhancements betrachtet (siehe Abb. 6.23). Der Anteil der Cases vom Typ Bug steigt für alle Produkte im Betrachtungszeitraum. Betrachtet man die Enhancements, steigt ebenfalls der Anteil der Cases mit Einträgen in der CC-Liste. Jedoch sinkt der Anteil zum vierten Betrachtungszeitpunkt für die Produkte Rot und Grün deutlich ab. Ausgehend von unterschiedlichen Startniveaus kommt es zu einer Annäherung von Bugs und Enhancements.

Neben der Betrachtung, ob eine Koordination mithilfe der CC-Liste stattfindet, gibt die Anzahl der Einträge in der CC-Liste einen Hinweis auf den benötigten Koordinationsaufwand. Diesbezüglich betrachtet der Qualitätsindikator (24) die mittlere Anzahl der Einträge in der CC-Liste (siehe Abb. 6.24). Sowohl für die Bugs als auch für die Enhancements zeigt sich ein Aufwärtstrend hinsichtlich der mittleren Anzahl der Einträge in der CC-Liste.



(a) Bugs (b) Enhancements

Abbildung 6.23: Anteil der Cases mit Einträgen in der CC-Liste



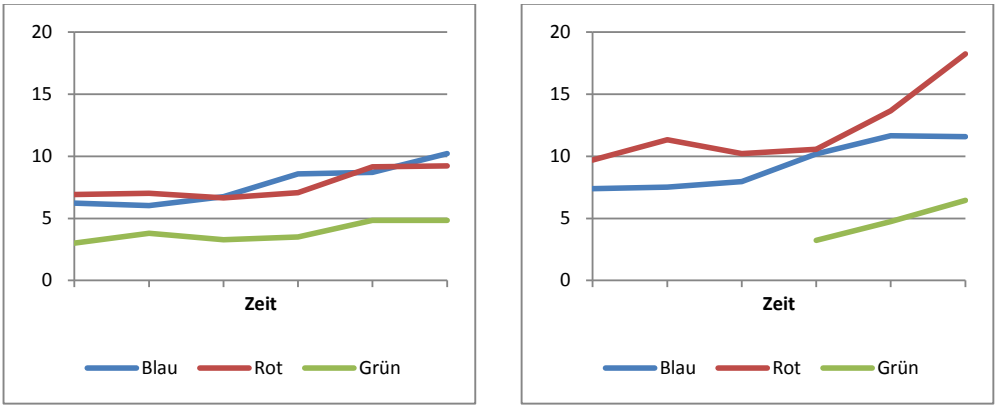
(a) Bugs (b) Enhancements

Abbildung 6.24: Mittlere Anzahl der Einträge in der CC-Liste

Anhand der beiden Qualitätsindikatoren (23) und (24) zeigt sich, dass durch den Mechanismus CC-Liste der Koordinationsaufwand hinsichtlich der Bearbeitung der Cases im Betrachtungszeitraum zunimmt.

Die Kommentare eines Cases sollen nur relevante Informationen über den Case enthalten oder diese ergänzen (vgl. [Sch08]). Diskussionen über den Case sollen nicht in Bugzilla, sondern über Skype, E-Mail, Telefon oder im Meeting stattfinden und deren Ergebnisse als Kommentar dokumentiert werden. Zu diesem Sachverhalt wird die mittlere Anzahl der Kommentare im Qualitätsindikator (25) betrachtet (siehe Abb. 6.25). Für Cases vom Typ Bug und vom Typ Enhancement steigt die mittlere Anzahl an Kommentaren im Betrachtungszeitraum an. In den Enhancements werden im Allgemeinen mehr Kommentare zur Dokumentation genutzt.

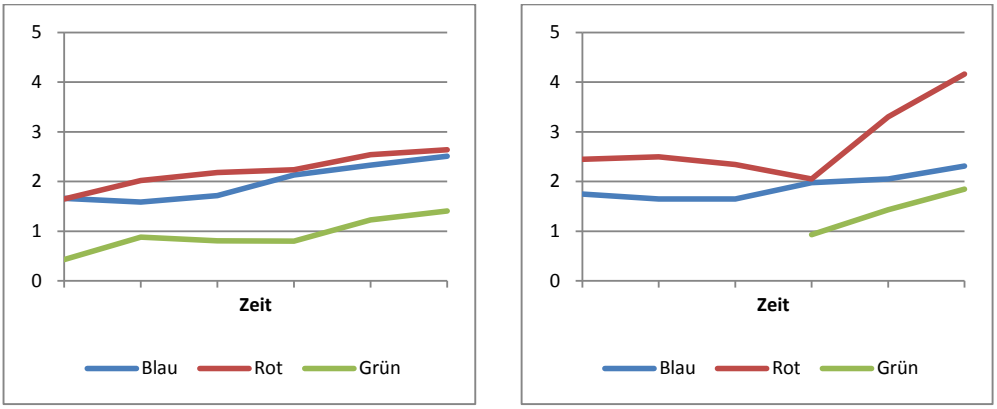
Die Metrik zur Bestimmung der mittleren Anzahl der Assignee-Wechsel wird durch den Qualitätsindikator (26) beschrieben. Die Messergebnisse werden in



(a) Bugs (b) Enhancements

Abbildung 6.25: Mittlere Anzahl der Kommentare

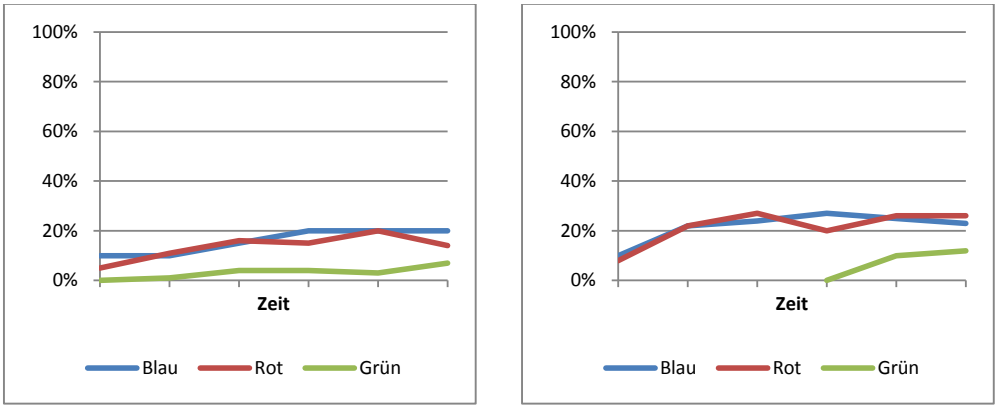
der Abbildung 6.26 veranschaulicht. Die mittlere Anzahl der Assignee-Wechsel steigt sowohl für die Bugs als auch für die Enhancements. Im Allgemeinen ist zu beobachten, dass Enhancements bis zu ihrer Realisierung im Mittel mehr Assignee-Wechsel benötigen als Bugs. Aufällig ist der Aufwärtstrend ab der vierten Betrachtungsperiode für die Enhancements des Produkts Rot.



(a) Bugs (b) Enhancements

Abbildung 6.26: Mittlere Anzahl der Assignee-Wechsel

Durch die Nutzung des Zustands WAITCUSTINFO wird dokumentiert, dass der Bearbeiter weitere Informationen angefordert hat und auf diese wartet, bevor er die Bearbeitung des Cases fortsetzen kann. Der Qualitätsindikator (27) bestimmt diesbezüglich den Anteil der Cases, welche den Zustand WAITCUSTINFO besucht haben (siehe Abb. 6.27) und betrachtet somit die Koordinationsschnittstelle zum Kunden beziehungsweise Anwender. Bei den Produkten Blau und Rot liegt der Anteil für die Cases vom Typ Bug bei 10% bis 20% und der Cases vom Typ Enhancement nach einem anfänglichen Anstieg bei 20% bis 30%. Der Anteil bei den Bugs des Produkts Grün weist einen Seitwärtstrend auf niedrigem Niveau auf, und für die Enhancements des Produkts Grün ist ein Aufwärtstrend im Betrachtungszeitraum zu beobachten.



(a) Bugs (b) Enhancements

Abbildung 6.27: Mittlere Anzahl der Besuche des Zustands WAITCUSTINFO

6.2.8 Supportprozess

Die SupportCalls sind Kundenanfragen, welche weder Änderungsanfragen noch Fehlermeldungen repräsentieren.

Aufgrund der direkten Kundenbeziehung steht eine möglichst schnelle Bearbeitung der Cases im Vordergrund. Folglich erlaubt der von KISTERS definierte Workflow einen direkten Zustandsübergang von den Zuständen {NEW, ASSIGNED, PROCESSING, WAITCUSTINFO} in den Zustand CLOSED. Zu dieser Fragestellung lässt sich der Qualitätsindikator (28) Anteil der Sofortlösungen der SupportCalls heranziehen (siehe Abb. 6.28). Der Anteil bezüglich der Sofortlösungen bleibt über den Betrachtungszeitraum stabil, wobei Niveauunterschiede hinsichtlich der einzelnen Produkte zu beobachten sind.

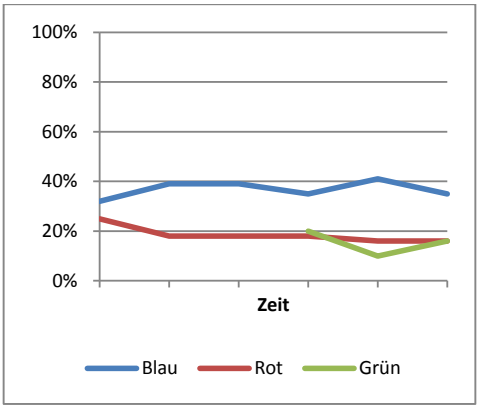
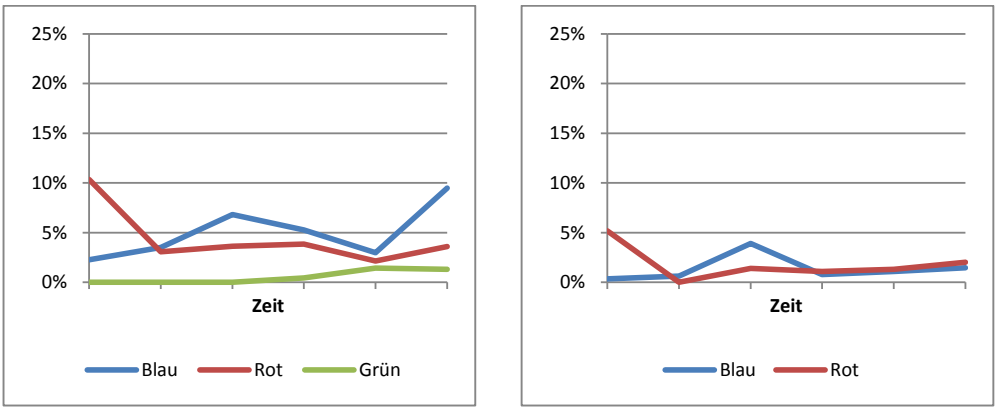


Abbildung 6.28: Anteil der Sofortlösungen

6.2.9 Fire-Fighting („P1“)

Die Fire-Fighting Aktivitäten sollen möglichst vermieden werden, da sie den Gesamtprozessablauf stören. Alle anderen Entwicklungs- und Wartungsaktivitäten sind gegebenenfalls zu unterbrechen und die verfügbaren Ressourcen der Lösung des P1-Cases unterzuordnen.

Aufgrund möglicher Kosten und störender Einflüsse sollte der Anteil der Fire-Fighting Aktivitäten möglichst gering sein. Diesen Aspekt der Entwicklungs- und Wartungsprozesse erfasst der Qualitätsindikator (29) Anteil der Fire-Fighting Aktivitäten (siehe Abb. 6.29). Der Anteil der Cases mit Fire-Fighting Aktivitäten ist bei den Enhancements niedriger als bei den Bugs.



(a) Bugs (b) Enhancements

Abbildung 6.29: Anteil an Fire-Fighting Aktivitäten

Neben der Häufigkeit der Fire-Fighting Aktivitäten ist die Dauer der eigentlichen Fire-Fighting Aktivität während der Bearbeitung von besonderem Interesse. Im Qualitätsindikator (30) wird hierzu der Median der Zeitspannen ermittelt, während sich der Case in der Zustandsmenge {NEW, ASSIGNED, PROCESSING, WAITCUSTINFO, REOPENED} mit der Priorität P1 aufhält.

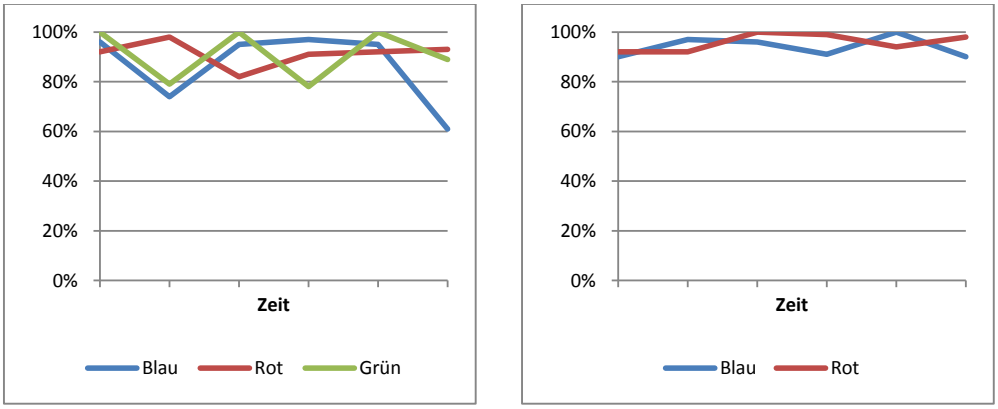
Die Interpretation des Qualitätsindikators (30) erfolgt anhand einer absoluten Zeitschranke, daher ist eine um den zeitlichen Aspekt **anonymisierte** Darstellung nicht sinnvoll. Aus diesem Grund wird im Folgenden auf die Darstellung verzichtet.

Die Mehrheit aller Zeitspannen der Cases vom Typ Bug hält sich innerhalb der Zustandsmenge mit Priorität P1 unterhalb eines halben Tages auf. Im Bezug auf die Enhancements liegt die Mehrheit aller Zeitspannen unterhalb eines Tages.

6.2.10 Kundenzufriedenheit

Die Prozessqualität kann aus Kundensicht nur unter Zuhilfenahme der für ihn sichtbaren Prozessartefakte bestimmt werden. Hierbei vergleicht der Kunde seine subjektive Leistungserwartung mit den tatsächlichen Ergebnissen.

Insbesondere die Einhaltung der mit dem Kunden vereinbarten Deadlines gibt einen Hinweis auf die Zuverlässigkeit hinsichtlich der Terminabsprachen. Der Anteil der verletzten Deadlines bezogen auf alle Cases mit Deadline beim Zustandsübergang nach CLOSED+FIXED (siehe Abb. 6.31) wird diesbezüglich im Qualitätsindikator (31) betrachtet. Der Anteil der verletzten Deadlines beschreibt sowohl für Bugs als auch Enhancements einen Seitwärtstrend. Im letzten Betrachtungszeitraum ist ein geringer werdender Anteil verletzter Deadlines für die Bugs des Produkts Blau zu erkennen.



(a) Bugs (b) Enhancements

Abbildung 6.31: Anteil der Cases, deren Deadline verletzt wurde (CLOSED+FIXED)

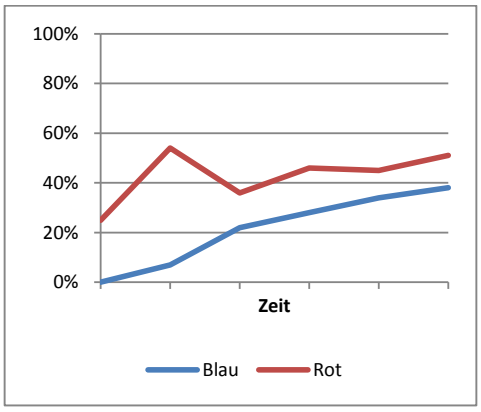


Abbildung 6.32: Anteil über das Kundenportal

Zur Betrachtung, ob sich die Kommunikationsstruktur mit dem Kunden durch die Einführung des Support Portals geändert hat, gibt der Anteil der eingehenden SupportCalls mit einem externen Reporter einen Hinweis. Hierzu wird im Folgenden der Qualitätsindikator (32) betrachtet (siehe Abb. 6.32). Der Anteil der SupportCalls mit einem externen Reporter hat für die Produkte Blau und Rot eine steigende Tendenz.

7 Diskussion

Inhalt

In der folgenden Diskussion werden die Metrikergebnisse aus Kapitel 6 hinsichtlich des im Kapitel 4 vorgestellten bidirektionalen Qualitätsmodells erörtert. Dies geschieht anhand ausgewählter Fragen zu den einzelnen Qualitätseigenschaften (siehe 4.2) unter Berücksichtigung der Qualitätsindikatoren (siehe 4.3). Die verschiedenen betrachteten Qualitätsindikatoren lassen sich durchgängig anhand ihrer Identifikationsnummer zuordnen.

7.1 Management der Anforderungen

7.1.1 Findet in der Betrachtungsperiode vordringlich Weiterentwicklung oder Fehlerbehebung statt?

Diese Fragestellung wird im Qualitätsindikator (01) bezogen auf den erfassten Zeitaufwand und der Anzahl der Cases betrachtet. Im Bezug auf den erfassten Zeitaufwand zur Bearbeitung der Cases zeigt sich für das Produkt Blau ein Anstieg von 57% auf 75% und für das Produkt Rot ein Anstieg von 45% auf 67% hinsichtlich des erfassten Zeitaufwands, der zur Weiterentwicklung aufgewendet wird. Folglich wird der erfasste Zeitaufwand beider Produkte vordringlich zur Weiterentwicklung genutzt.

Als eine Alternative zum erfassten Zeitaufwand kann die Anzahl der Cases betrachtet werden. Hierzu wird die implizite Prämisse gemacht, dass die Granu-

larität der betrachteten Cases hinsichtlich des Zeitaufwands ungefähr gleich ist. Aufgrund der unterschiedlichen Granularität (15) und dem hohen Anteil an dokumentierten Zeitaufwand (14) ist eine Betrachtung auf Basis des dokumentiertem Zeitaufwands vorzuziehen. Der Qualitätsindikator beobachtet jedoch nur die anteilige Entwicklung im Betrachtungszeitraum, nicht jedoch das absolute Entwicklungsvolumen. Folglich sind nur Aussagen über die Anteile der Entwicklung des jeweiligen Betrachtungszeitraums zulässig.

7.1.2 Wie hoch ist der Anteil der Projektentwicklung am Entwicklungsaufkommen?

Ein hoher Anteil an Individualentwicklung kann möglicherweise die Weiterentwicklung des Standardprodukts behindern. Der Qualitätsindikator (02) interpretiert den Anteil der Enhancements mit einem externen Auftraggeber als einen Hinweis auf Projektentwicklung. Hierzu werden im Folgenden die Anteile anhand des erfassten Zeitaufwands betrachtet.

Die Mehrheit der bearbeiteten Enhancements des Produkts Rot besitzt einen externen Auftraggeber und der Anteil der Cases steigt im Betrachtungszeitraum auf bis zu 98%. Betrachtet man das Produkt Blau so steigt der Anteil der Enhancements mit externem Auftraggeber auf 60%. Für das Produkt Grün können schon Enhancements mit externem Auftraggeber beobachtet werden.

Ein möglicher Grund für den hohen Anteil der Cases des Produkts Rot kann die Entwicklung des Nachfolgeprodukts Grün sein. Mögliche Erweiterungen mit internem Auftraggeber des Produkts Rot werden unter Umständen vordringlich im Produkt Grün realisiert. Anhand der Cases lässt sich jedoch nicht feststellen, in welchem Umfang die Enhancements mit externem Auftraggeber in das Standardprodukt eingehen.

Werden die Anteile auf Basis der Cases ermittelt, so zeigen sich sehr ähnliche Entwicklungstendenzen. Dies ist, aufgrund des im Qualitätsindikator (14) hohen Anteils an erfasstem Zeitaufwand plausibel. Die Unterschiede hinsichtlich der Granularität der Cases (15) spielen eine untergeordnete Rolle, da nur Cases vom Typ Enhancement verglichen werden.

7.1.3 Wie zuverlässig ist eine erste Klassifikation der Cases hinsichtlich ihres Typs?

Eine zielgerichtete Analyse und Bearbeitung der Cases wird gefördert, indem eingehenden Cases der richtige Typ (Bug, Enhancement oder SupportCall) zugewiesen wird. Hierdurch wird der Case als Änderungsanforderung, Fehlermeldung oder Kundenanfrage klassifiziert. Betrachtet man den Qualitätsindikator (03), kann man den Trend erkennen, dass sich zunehmend der Typ des Cases bis zum ersten Zustandswechsel nach RESOLVED+FIXED ändert, was unter Umständen Verzögerungen in der Analyse und Bearbeitung zur Folge hat.

Betrachtet man insbesondere den Qualitätsindikator (04), wird deutlich, dass eine Klassifikation des Typs SupportCall fragil ist. Dies ist plausibel, wenn man die über das Kundenportal eingestellten Cases betrachtet (siehe 7.10.2 Ändert sich die Kommunikationsstruktur zum Kunden?). Werden mögliche Wechsel des Typs nicht ausreichend in den Prozessabläufen berücksichtigt, können Verzögerungen während des Bearbeitungsprozesses entstehen. Der ab der fünften Periode angestiegene Anteil der Typ-Wechsel der Cases des Produkts Rot von Bugs zu SupportCalls kann ein möglicher Hinweis auf ein Problem bei der Klassifikation sein. Dies kann beispielweise durch eine unzureichende Abgrenzung der einzelnen Kategorien Bug und SupportCall entstehen.

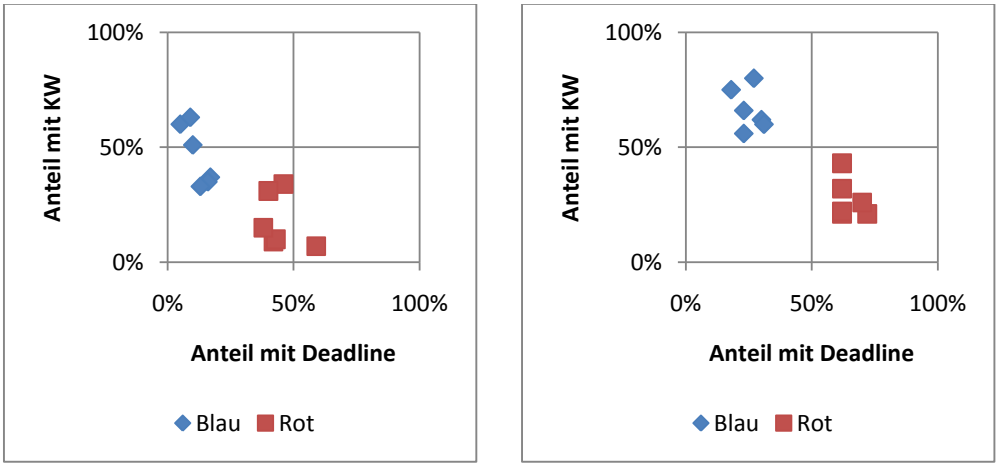
7.1.4 Kann parallele oder doppelte Arbeit vermieden werden?

Um Parallelentwicklung und doppelte Arbeit zu vermeiden sollten diese frühzeitig erkannt werden. Im Qualitätsindikator (05) wird daher der Anteil der Cases bestimmt, welche vor ihrer Kennzeichnung als DUPLICATE einen Zustand aus der Menge {ASSIGNED, PROCESSING} besucht haben. Der Anteil hinsichtlich des Produkts Blau steigt innerhalb des Betrachtungszeitraums auf über 30%, was ein Anhaltspunkt dafür sein kann, dass es zu parallelen Arbeiten oder mehrfacher Arbeit kommen kann.

7.2 Planung

7.2.1 Werden Termine geplant?

In den Bugzilla Cases werden zwei verschiedene Arten von Terminen erfasst. Dies ist zum einen der geplante Termin des Bearbeitungsbeginns als Kalenderwoche (KW) und gegebenenfalls eine Deadline, welche den Termin festlegt, an dem der Case gelöst sein muss. Hierzu wurden die Qualitätsindikatoren (06) und (07) ausgewertet. Eine mögliche Beziehung zwischen der Planung mit KW und der Planung mit Deadline lassen sich in einem Streudiagramm darstellen (siehe Abb. 7.1). In den Streudiagrammen bilden die in mehreren Zeitintervallen betrachteten Produkte jeweils voneinander getrennte Punktwolken. Beim Produkt Blau wird schwerpunktmäßig der Bearbeitungsbeginn (KW) als Planungsinstrument genutzt. Hingegen ist beim Produkt Rot das dominierende Planungsinstrument die Deadline, welche in den Cases dokumentiert wird. Vergleicht man die beiden Planungsaspekte für die Bugs und Enhancements, sind die Planungsanteile für die Enhancements im Allgemeinen größer. Ein möglicher Grund ist, dass der benötigte geplante Arbeitsaufwand den Planungsaufwand für Enhancements rechtfertigt. Ebenso besitzen die Enhancements häufig einen längeren Planungshorizont, im Gegensatz zu den unvorhersehbar auftretenden Bugs.



(a) Bugs (b) Enhancements

Abbildung 7.1: Streudiagramm: Anteil der Cases mit dokumentiertem Deadline und Anteil der Cases mit dokumentiertem Bearbeitungsbeginn (Kalenderwoche) des jeweiligen Betrachtungszeitraums

Eine mögliche Begründung für den hohen Planungsanteil mit Deadlines bezogen auf die Enhancements des Produkts Rot (siehe Abb. 6.7) ist möglicherweise der hohe Anteil von Enhancements mit externen Auftraggebern (siehe Abb. 6.2). Betrachtet man die Verwendung der Planungsinstrumente des Produkts Grün, so sind diese eher dem Produkt Rot als dem Produkt Blau ähnlich.

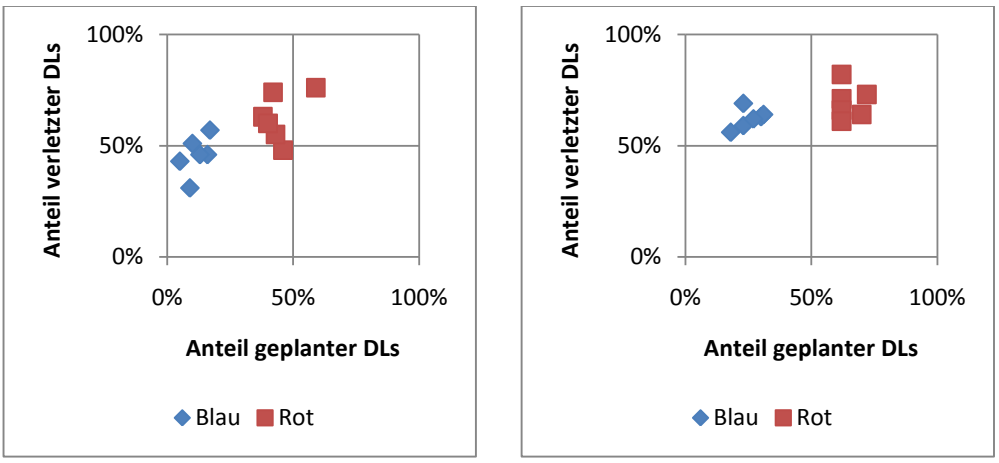
7.2.2 Werden die geplanten Termine eingehalten?

Die Auswertung der Qualitätsindikatoren (08) Anteil der Cases mit verschobenem Bearbeitungsbeginn und Qualitätsindikator (09) Anteil der verletzten Deadlines im Zustand RESOLVED+FIXED ist ein Hinweise darauf, ob die oben diskutierten geplanten Termine eingehalten werden. Zur Betrachtung, ob die geplanten Bearbeitungstermine (geplant als Bearbeitungsbeginn) eingehalten werden, ergibt sich ein schwaches Indiz anhand des verschobenen Bearbeitungsbeginns. Denn die Metrik lässt keine Aussage zu, ob ein Termin vor- oder zurückverlegt wurde. Im Allgemeinen wird für Blau ein höherer Anteil an dokumentierten Bearbeitungsterminen verschoben, was an der intensiveren Nutzung hinsichtlich des Planungsinstruments Kalenderwoche liegen kann. Vergleicht man die Anteile von Bugs und Enhancements, so ist der Anteil der verschobenen Bearbeitungstermine der Enhancements im Allgemeinen größer als der der Bugs. Ein möglicher Grund kann im unvorhergesehenen Auftreten der Bugs liegen.

Der Qualitätsindikator (09) Anteil der verletzten Deadlines bei Zustandsübergang nach RESOLVED+FIXED zeigt, dass die Mehrheit der Deadlines schon vor dem Ende der Bearbeitung verletzt wird. Das in Abbildung 7.2 dargestell-

te Streudiagramm verdeutlicht, dass sich unabhängig vom Anteil der geplanten Deadlines, der Anteil der verletzten Deadlines auf einem ähnlichen Niveau befindet. Somit ist der Anteil der verletzten Deadlines möglicherweise unabhängig von den Erfahrungen mit dem Planungsinstrument Deadline.

Ein Grund hierfür könnte der mögliche gefundene Konsens mit dem Kunden hinsichtlich der Festlegung der Deadline sein. Des Weiteren sind die Bearbeitungszeiten der Enhancements (16) aufgrund der volatilen Schwankungen nur schwer zu schätzen.



(a) Bugs (b) Enhancements

Abbildung 7.2: Streudiagramm: Anteil der geplanten Deadlines und Anteil der verletzten Deadlines beim Zustandsübergang nach RESOLVED +FIXED des jeweiligen Betrachtungszeitraums

7.2.3 Wird der benötigte Zeitaufwand zur Umsetzung der Cases geplant?

Diese Fragestellung wird im Qualitätsindikator (10) betrachtet. Der Anteil des geplanten Zeitaufwands für das Produkt Rot befindet sich hinsichtlich der Enhancements auf dem höchsten Niveau und bezüglich der Bugs auf einem sehr hohen Niveau. Folglich wird der Aufwand für fast alle Cases im Betrachtungszeitraum von Produkt Rot geplant. Beim Produkt Blau liegt das Niveau geringfügig niedriger. Die Anteile des Produkts Grün sind deutlich niedriger, was mit dem Neuentwicklungscharakter begründet werden kann. Mit steigendem Anteil an erfassten Cases im Produkt Grün (siehe Tab. 6.1) erhöht sich der Anteil der Cases mit geplantem Zeitaufwand.

7.2.4 Ist die Schätzung des Zeitaufwands hinreichend genau?

Das Gewicht `originalEffortEstimationAccuracy` vergleicht den geschätzten mit dem tatsächlichen Arbeitsaufwand. Der Qualitätsindikator (11) bestimmt den Median der Cases hinsichtlich des Gewichts `originalEffortEstimationAccuracy`. Die Auswertung zeigt, dass der überwiegende Anteil der Cases hinsichtlich des tatsächlich dokumentierten und des geplanten dokumentierten Arbeitsaufwands abweicht.

7.2.5 Wie groß ist die Planungsreichweite?

Zur Bestimmung der Größe der Planungsreichweite (siehe Qualitätsindikator (12)) wird das Verhältnis des verbleibenden Zeitaufwands am Ende der Betrachtungsperiode und des geleisteten Zeitaufwands der Periode betrachtet.

Für die Cases vom Typ Bug ist die Größe der Planungsreichweite im Allgemeinen wesentlich kleiner dimensioniert als jene der Enhancements. Ein möglicher Grund hierfür ist die zeitnahe Behebung von Bugs.

Die Größe des Planungspuffers hinsichtlich der Enhancements schwankt sehr stark und ist im Allgemeinen deutlich größer als bei den Bugs. Möglicherweise entstehen die Schwankungen des Puffers dadurch, dass größere Blöcke funktionaler Erweiterungen in Bugzilla eingestellt und in den folgenden Perioden abgearbeitet werden. Eine qualitative Wertung der Puffergröße ist nicht sinnvoll, da der Planungspuffer an die individuellen Bedürfnisse der einzelnen Produkte und deren Releasezyklen angepasst sein muss.

7.3 Überwachung und Verfolgung des Fortschritts

7.3.1 Wird der Entwicklungsfortschritt der Bearbeitung dokumentiert?

Der Entwicklungsfortschritt eines Cases lässt sich zum einen anhand der Verwendung der Zustände und zum anderen anhand des Arbeitsaufwands bestimmen.

Im Qualitätsindikator (13) wird der Anteil der Cases mit Besuch der Zustände ASSIGNED und PROCESSING bestimmt. Dieser Anteil ist für die Enhancements üblicherweise deutlich höher als bei den Bugs. Die Enhancements werden mehrheitlich für die Produkte Blau und Rot dokumentiert. Dies scheint für das Produkt Blau plausibel, da für die Cases mit einem geringen Zeitaufwand (vgl. Granularität (15)) der entstehende Dokumentationsaufwand unter Umständen aus Sicht des Entwicklers nicht gerechtfertigt ist. Zudem dauert die Bearbeitung im Allgemeinen (vgl. 7.4.2 Lösungszeit) länger.

Die im Qualitätsindikator (14) untersuchte Dokumentation der Arbeitszeiten liegt für alle Produkte auf einem hohen Niveau. Eine bevorzugte Dokumentation von Enhancements gegenüber Bugs ist nicht zu beobachten. Der Qualitätsindikator (14) betrachtet lediglich, ob der Zeitaufwand erfasst wird, nicht jedoch in welcher Granularität dieser erfasst wird.

7.3.2 Wie ist die Granularität der Cases?

Wird bei der Analyse ein möglicher Vergleich auf Basis der Anzahl der Cases angestellt, werden unter Umständen der Zeitaufwand und die Komplexität zur Lösung des Cases nicht ausreichend berücksichtigt. Betrachtet man die anonymisierte Darstellung des Qualitätsindikators (15) (siehe Abb. 6.15), zeigt sich, dass insbesondere für die Enhancements des Produkts Blau mehrheitlich ein größerer Zeitaufwand im Vergleich zu den Bugs geschätzt wird. Folglich kann ein direkter Vergleich der Cases des Typs Bug mit den Cases des Typs Enhancement für dieses Produkt irreführend sein. Für die Cases des Produkts Rot zeigt sich in den ersten Betrachtungsperioden ein ähnlicher Sachverhalt.

In der folgenden Abbildung 7.3 werden die einzelnen Cases aufsteigend nach ihrer Größe des geschätzten Zeitaufwands auf der Abszisse (x-Achse) aufgetragen und ihr normalisierter geschätzter Zeitaufwand in Prozent (%) auf der Ordinate (y-Achse) abgetragen. Der geschätzte Zeitaufwand wurde anhand der größten Beobachtung innerhalb des Betrachtungszeitraums für das Produkt normalisiert. Betracht wurden die Enhancements mit geschätztem Zeitaufwand des Produkts Rot für das fünfte Halbjahr.

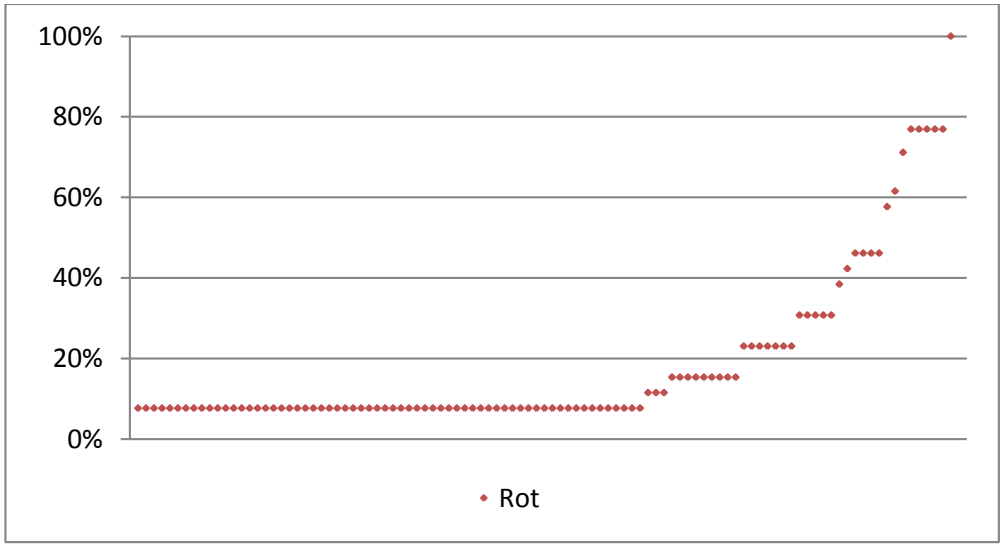


Abbildung 7.3: Geschätzter Zeitaufwand in Prozent (%) der einzelnen Cases normalisiert mit der größten Beobachtung des Betrachtungszeitraums für die Enhancements des Produkts Rot

Eine Betrachtung ausgehend vom Median führt unter Umständen ebenfalls zu falschen Annahmen über die Granularität der Cases vom Typ Enhancements, da insbesondere Cases mit einem großen geschätztem Zeitaufwand (siehe Abb. 7.3) nicht ausreichend berücksichtigt werden. Somit ist die anhand des Medians ermittelte Granularität nur ein Hinweis auf unterschiedliche Casegrößen zwischen den Bugs und Enhancements.

7.4 Lösungsgeschwindigkeit

7.4.1 Bearbeitungszeit? (time to solution)

Die Bearbeitungszeit wird bestimmt anhand des Zeitintervalls von der Einstellung des Cases bis zu seinem ersten Zustandsübergang nach RESOLVED +FIXED (vgl. Qualitätsindikator (16)). Der Median des Zeitintervalls gibt einen Hinweis auf die Bearbeitungszeit der Cases. Hierzu wurden im Kapitel 6 repräsentativ die Cases mit der Priorität P2 betrachtet und in **anonymisierter** Form dargestellt.

Vergleicht man die Cases vom Typ Bug und Enhancement hinsichtlich ihrer Bearbeitungszeit, zeigt sich, dass die Zeitspanne hinsichtlich der Bugs wesentlich geringer ist als bei den Enhancements. Dies ist plausibel, da Bugs im Allgemeinen zeitnah gelöst werden müssen, um die Beeinträchtigung für den Kunden möglichst gering zu halten. Der Median der Zeitintervalle hinsichtlich der Enhancements weist eine enorme Volatilität auf. Somit sind nur sehr unzulängliche Prognosen hinsichtlich der Fertigstellung der Bearbeitung von Enhancements möglich.

7.4.2 Lösungszeit? (time to customer accept)

Im Gegensatz zur Bearbeitungszeit wird die Lösungszeit als das Zeitintervall zwischen dem Einstellen des Cases und dem Zustandsübergang nach CLOSED +FIXED bestimmt. In der Auswertung des **anonymisierten** Qualitätsindikators (17) für die Cases mit Priorität P2 wird deutlich, dass der Median der Zeitintervalle der Bugs üblicherweise kleiner ist als der Median hinsichtlich der Enhancements. Auch hier steht eine zeitnahe Behebung der Bugs im Vordergrund um die Beeinträchtigungen für den Kunden möglichst klein zu halten. In der Betrachtung fällt zudem die Volatilität des Medians der Lösungszeiten bezüglich der Cases des Produkts Blau auf. Deren Lösungszeiten scheinen sich für Bugs und Enhancements aufzuschwingen (vgl. Abb. 6.17) und sollten weiter beobachtet werden.

Zur weiteren Analyse werden beispielhaft die Cases in aufsteigender Reihenfolge hinsichtlich ihrer Lösungszeit geordnet und ihre Lösungszeit im Verhältnis zur maximalen beobachteten Lösungszeit des Betrachtungszeitraums auf der Ordinate (y-Achse) eingetragen (siehe Abb. 7.4).

Im Folgenden werden die Bugs des Produkts Blau mit der Priorität P2 aus dem fünften Halbjahr in der Abbildung dargestellt. Betrachtet man die Länge der Zeitintervalle der Lösungszeit, so lassen sich diese in zwei Abschnitte unterteilen. Während die Cases des ersten Abschnitts einen hohen Zeitbezug zwischen dem Einstellen der Cases und ihrer Lösung bezüglich der Betrachtungsperiode aufweisen, ist dieser Zeitbezug für die Cases des zweiten Abschnitts nicht gegeben. Eine plausible Erklärung für die Cases des zweiten Abschnitts sind „Aufräumarbeiten“, dass heißt Cases, die schon im Produkt integriert sind, werden erst in den folgenden Perioden ausgetragen. Folglich ist der Median des Zeitintervalls zur Prognose hinsichtlich der Lösungszeiten als naiver Schätzer nur bedingt geeignet.

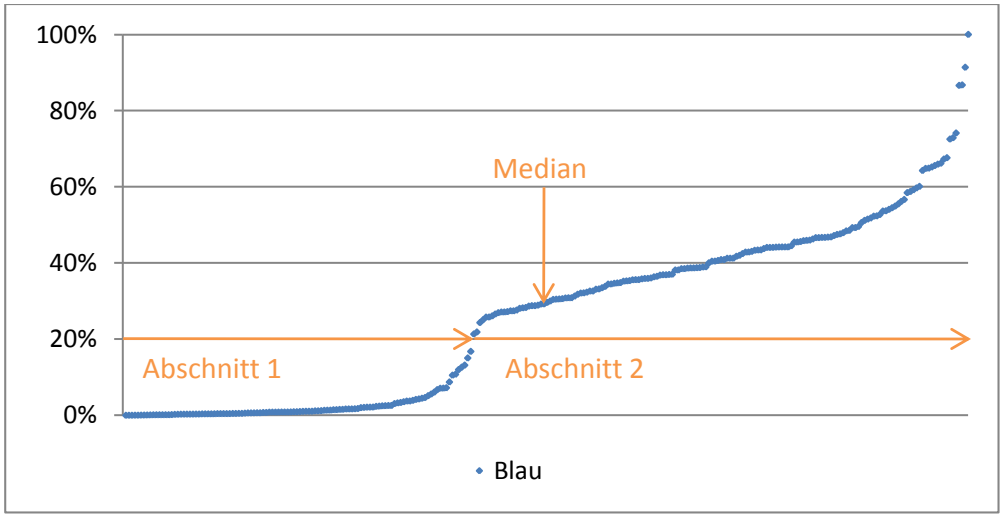


Abbildung 7.4: Bugs des Produkts Blau mit Priorität P2 des 5. Betrachtungshalbjahres hinsichtlich ihrer normalisierten Lösungszeit

Im Vergleich zwischen Lösungszeiten und Bearbeitungszeiten (siehe oben) fällt auf, dass die Lösungszeiten deutlich von denen der Bearbeitungszeit nach oben abweichen. Eine mögliche Begründung sind die Releasezyklen und dass der Kunde unter Umständen erst später die Lösung bestätigt.

7.5 Software-Qualitätssicherung

7.5.1 In welchem Umfang fallen ungeplante Nacharbeiten an?

Die ungeplanten Nacharbeiten werden anhand des Zustandswechsels nach REOPENED dokumentiert. Der Qualitätsindikator (18) betrachtet hierzu alle Zustandswechsel, deren Verweildauer länger als eine Stunde beträgt. Betrachtet man die Analyse der Cases mit der Priorität P2 anhand des Qualitätsindikators (18), fällt für die Enhancements ein signifikanter Anteil innerhalb der Betrachtungsperioden auf. Das Niveau der Bugs befindet sich auf einem geringeren Niveau.

Als Zeitintervall wurde eine Stunde gewählt, um die Nacharbeiten zu identifizieren. Aus technischer Sicht ist es in Bugzilla nur möglich Felder, wie die Resolution, beim Zustandsübergang nach RESOLVED zu setzen. Daher sind Änderungen nur über den Zustand REOPENED möglich. Der Wert von einer Stunde bietet einen geeigneten Kompromiss zwischen der Präzision der Messung und den Klassifikationsfehlern aufgrund der Bugzilla Eigenschaften.

7.5.2 Wie lange ist die Bearbeitungsdauer der ungeplanten Nacharbeiten?

Die Fragestellung beschäftigt sich mit den **ungeplanten** Nacharbeiten. Die Nacharbeiten gelten so lange als ungeplant, bis sie den Zustand REOPENED verlassen. Werden die Nacharbeiten mithilfe der Zustände NEW, ASSIGNED, WAITCUSTINFO oder PROCESSING dokumentiert, gelten sie im weiteren Verlauf als **geplant**. Eine Einschätzung bezüglich der Bearbeitungsdauer der ungeplanten Nacharbeiten liefert die Analyse des Qualitätsindikators (16). Hierzu wurde exemplarisch die Menge der Cases mit der Priorität P2 betrachtet. Ebenso wie in der zuvor diskutierten Fragestellung werden nur Zeitintervalle von mindestens einer Stunde berücksichtigt.

Vergleicht man den Median der Bearbeitungsdauer der ungeplanten Nacharbeiten mit dem Median der Zeitspannen bis zum Zustandsübergang nach RESOLVED+FIXED (siehe Tab.7.1, Tab. 7.2), so zeigt sich, dass die ungeplanten Nacharbeiten wesentlich unter denen der Bearbeitungszeit liegen.

7.5.3 Wie sensitiv ist die Fehlersuche?

Die Fehler, die nicht im Rahmen von qualitätssichernden Maßnahmen entdeckt werden, beeinträchtigen den Kunden bei seiner Arbeit mit dem Produkt. Um dies zu verhindern, gilt es möglichst wenig Fehler zum Kunden vordringen zu lassen. Hierzu wird der Qualitätsindikator (20) Defect Escape Rate betrachtet,

Bugs	1. Halbjahr	2. Halbjahr	3. Halbjahr	4. Halbjahr	5. Halbjahr	6. Halbjahr
Blau						
TimeToSolution	42,72%	32,97%	49,04%	74,67%	55,81%	100,00%
Median Dauer Nacharbeiten	21,42%	7,96%	22,32%	12,81%	6,08%	7,51%
Rot						
TimeToSolution	75,96%	99,15%	93,04%	75,91%	87,96%	100,00%
Median Dauer Nacharbeiten	6,77%	17,90%	33,61%	23,32%	20,99%	6,81%

Tabelle 7.1: Median der Zeitintervalle der Cases mit der Priorität P2 normalisiert, jeweils mit der größten Beobachtung für das Produkt

Enhancements	1. Halbjahr	2. Halbjahr	3. Halbjahr	4. Halbjahr	5. Halbjahr	6. Halbjahr
Blau						
TimeToSolution	63,67%	42,63%	34,47%	69,05%	52,04%	100,00%
Median Dauer Nacharbeiten	4,07%	9,51%	2,32%	1,38%	1,37%	1,48%
Rot						
TimeToSolution	46,18%	100,00%	30,15%	66,97%	63,64%	70,35%
Median Dauer Nacharbeiten	1,69%	4,38%	2,17%	5,09%	7,07%	6,35%

Tabelle 7.2: Median der Zeitintervalle der Cases mit der Priorität P2 normalisiert, jeweils mit der größten Beobachtung für das Produkt

welcher den Anteil aller extern gemeldeten Fehler bezogen auf die Fehlermeldungen einer Periode ermittelt. Anhand der Abbildung 6.20 wird deutlich, dass der überwiegende Anteil der Fehlermeldungen ohne Hinweis auf einen externen Unternehmensbezug eingestellt wird. Betrachtet man beispielsweise das 6. Halbjahr erreichen die externen Fehlermeldungen einen Anteil von 26% für das Produkt Rot, 24% für das Produkt Grün und 10% für das Produkt Blau.

7.6 Software-Konfigurationsmanagement

7.6.1 Wird der `targetMilestone` für den Case dokumentiert?

Der Qualitätsindikator (21) betrachtet den Anteil der Cases, welche mit einem `targetMilestone` beim Zustandsübergang nach CLOSED+FIXED dokumentiert wurden. Anhand der Anteile sieht man, dass für die Bugs der Produkte Blau und Rot viel dokumentiert werden und hinsichtlich der Enhancements sehr viel dokumentiert wird. Der Anteil der mit `targetMilestone` dokumentierten Cases bei Zustandsübergang nach RESOLVED+FIXED lässt sich mithilfe von BugzillaMetrics nicht bestimmen (siehe Kapitel 5.3). Zudem werden zur Dokumentation individuelle Flags verwendet, die Auskunft geben, ob ein Cases in einem Release enthalten ist.

7.6.2 Werden die umgesetzten Anforderungen in den `release Notes` dokumentiert?

Die `releaseNotes` dokumentieren für den Kunden, dass die Änderungen eines Bugs oder eines Enhancement im Release enthalten sind. Hierzu wird der Anteil der Cases mit `releaseNotes` beim Zustandsübergang nach CLOSED+FIXED bestimmt (vgl. Qualitätsindikator (22)). Eine einheitliche Regelung zum Verwenden von `releaseNotes` existiert während des Betrachtungszeitraums für und innerhalb der verschiedenen Produkte nicht.

Beim Vergleich von Bugs und Enhancements besitzen die Enhancements den höheren Anteil an Cases, die mit `releaseNotes` dokumentiert wurden. Insbesondere in der letzten Periode des Betrachtungszeitraums steigt der Anteil bei allen Bugs der Produkte an, wobei ein allgemeiner Aufwärtstrend hinsichtlich der Nutzung zu beobachten ist. Folglich werden die `releaseNotes` unabhängig von bestehenden Regelungen stärker genutzt. Ebenso wie der `targetMilestone` kann der Anteil an gesetzten `releaseNotes` nicht beim Zustandsübergang nach RESOLVED+FIXED nachvollzogen werden. Somit kann nicht bestimmt werden, ob die `releaseNotes` durch die Entwickler oder die Qualitätssicherung dokumentiert wurde.

7.7 Koordination der beteiligten Anspruchsgruppen

7.7.1 Wie hoch ist der Koordinationsaufwand bei der Bearbeitung der Cases?

Im Bugzilla Case werden verschiedene Aspekte der Koordination während der Bearbeitung abgebildet. Die Wechsel der Assignees wurden im Qualitätsindikator (26) untersucht. Das Mittel der Assignee-Wechsel bis zum ersten Zustandsübergang nach RESOLVED+FIXED der Cases steigt für alle Produkte im Betrachtungszeitraum an. Auffällig ist der Anstieg der Assignee-Wechsel der Enhancements des Produkts Rot ab der vierten Betrachtungsperiode.

Ein ähnlicher Kurvenverlauf ergibt sich beim Qualitätsindikator (25), welcher die mittlere Anzahl der Kommentare ermittelt. Auch in diesem Fall steigt die mittlere Anzahl der Kommentare bei den Enhancements des Produkts Rot ab der vierten Betrachtungsperiode an. Ein automatischer Kommentar beim Wechsel des Assignees wird nicht erstellt. Eine exakte Bestimmung, ob ein Kommentar beim Assignee-Wechsel in den Cases eingepflegt wurde, ist nicht möglich. Zudem ist eine Auswertung des natürlich sprachlichen Kommentars bezüglich des Assignee-Wechsels nur unzureichend möglich. In Folge eines Assignee-Wechsels sollte vor der weiteren Bearbeitung der bisherige dokumentierte Bearbeitungsverlauf nachvollzogen werden. Somit steigt der Koordinationsaufwand durch die häufiger werdenden Assignee-Wechsel und die dabei zu betrachtenden Kommentare. Dies sollte insbesondere für das Produkt Rot weiter beobachtet werden.

Ebenso lässt sich der Koordinationsaspekt anhand des CC's pro Case (`ccCount`) betrachten. Hierzu wurden zuerst der Anteil der Cases mit Einträgen in der CC-Liste (siehe Qualitätsindikator (23)), sowie die mittlere Anzahl der Einträge in der CC-Liste bei Zustandsübergang nach RESOLVED+FIXED (siehe Qualitätsindikator (24)) ermittelt. Die Nutzung der CC-Listen Funktionalität besitzt einen aufsteigenden Trend und erreicht unabhängig vom betrachteten Produkt ein hohes Niveau. Das arithmetische Mittel der Anzahl der CC's pro Case beschreibt in welcher Intensität die CC-Listen Funktionalität genutzt wird. Auch hier zeigt sich eine deutliche Zunahme insbesondere bei den Enhancements im Betrachtungszeitraum. Wird ein Kommentar zu einem Case bei Bugzilla eingestellt, so erhalten neben dem reporter, assignee und qaContact ebenso die Personen auf der CC-Liste über E-Mail den Kommentar zugesandt. Betrachtet man die steigende Anzahl an Kommentaren pro Case (siehe Qualitätsindikator (25)) und die intensivere Nutzung der CC-Listen Funktionalität, so besteht die Gefahr, dass wichtige Informationen, wie die Verletzung der Deadline, nicht ausreichend hinsichtlich ihrer Bedeutung eskaliert werden.

Analog zu den Qualitätsindikatoren (25) und (26) fällt bei den Enhancements des Produkts Rot eine intensivere Nutzung der Koordinationsmechanismen (23) und (24) ab der vierten Periode im Betrachtungszeitraum auf. Ein möglicher Grund hierfür kann die Koordination mit dem Produkt Grün sein, um doppelte oder parallele Entwicklungen zu vermeiden.

7.7.2 Wieviel Reibung entsteht im Koordinationsprozess mit dem Kunden?

Einen Hinweis über die Reibung in den Koordinationsprozessen mit dem Kunden gibt der Qualitätsindikator (27). Hierzu wird der Anteil der Cases, welche den Zustand WAITCUSTINFO besucht haben, betrachtet. Ein Besuch des Zustands WAITCUSTINFO ist ein Hinweis auf Reibung im Koordinationsprozess mit dem Kunden, da die Koordination im Prozess explizit in diesem Zustand dokumentiert wird. Normalerweise ist ein Kommentar ausreichend um den Kommunikationsvorgang in Bugzills abzubilden. Im Betrachtungszeitraum liegt der Anteil für alle Produkte auf einem niedrigen Niveau.

Zudem ist zu berücksichtigen, dass hier nicht bezogen auf die Gesamtanzahl der Kundenkontakte (E-Mail oder Telefonate) innerhalb eines Cases gemessen wurde, sondern der Anteil bezogen auf die Gesamtzahl der Cases ermittelt wurde. Die Kundenkontakte werden im Allgemeinen in den Cases nur in unstrukturierter Form als Kommentare erfasst, so dass deren Auswertung mithilfe von BugzillaMetrics nicht möglich ist.

7.8 Supportprozess

7.8.1 Wie viele der SupportCalls können direkt gelöst werden?

Diese Fragestellung betrachtet den Anteil der Cases, die nach ihrem einstellen in Bugzilla ohne weitere Zustandsübergänge in den Zustand CLOSED+FIXED gewechselt sind. Der Anteil der Sofortlösungen wurde mittels des Qualitätsindikators (28) bestimmt. Sowohl für die Produkte Blau als auch Rot bleibt der Anteil an Sofortlösungen über den Betrachtungszeitraum hinweg stabil. Der Anteil des Produkts Grün liegt auf dem Niveau von Rot.

Betrachtet wurde nur die Anzahl der Schritte bis zum Zustand CLOSED +FIXED. Aufgrund der Nachdokumentation von Cases können ebenfalls scheinbare Sofortlösungen entstehen. Des Weiteren können die Cases, welche ohne weitere Zustandsübergänge nach RESOLVED+FIXED wechseln, ebenfalls schon gelöst sein. Jedoch wurde die Lösung durch den Kunden unter Umständen noch nicht bestätigt. Der Fokus liegt bei CLOSED+FIXED jedoch darauf, dass keine weiteren Ressourcen zur Überwachung der Cases, wie bei RESOLVED+FIXED, benötigt werden.

7.9 Fire-Fighting Aktivitäten („P1“)

7.9.1 Wie hoch ist der Anteil der Fire-Fighting Aktivitäten im Prozess?

Die Störungen aufgrund der Fire-Fighting Aktivitäten sollte so gering wie möglich sein. Daher wurde der Anteil der Fire-Fighting Aktivitäten im Verhältnis zu der Entwicklungsaktivität der Betrachtungsperiode ermittelt (siehe Qualitätsindikator (29)). Im Allgemeinen kann das Fire-Fighting bei den Enhancements der Produkte Blau und Rot vermieden werden. Im Bezug auf die Bugs liegen die Anteile im Allgemeinen auf leicht höherem Niveau.

Da die Fire-Fighting Aktivitäten eine Störung des normalen Prozessablaufs darstellen, wurden auch mehrfache Bearbeitungen eines Cases als Fire-Fighting Aktivitäten gezählt. Dies führt beispielsweise im letzten Betrachtungszeitraum für die Bugs des Produkts Blau dazu, dass insgesamt circa 27% der Fire-Fighting Aktivitäten schon einmal als P1 in der Periode bearbeitet wurden.

7.9.2 Wie lange dauert die Lösung der Cases hinsichtlich der Priorität P1?

Die Fire-Fighting Aktivitäten sollen in einem eng begrenzten zeitlichen Rahmen stattfinden. Hierzu wurde die Länge der Zeitintervalle der Fire-Fighting Aktivitäten entsprechend des Qualitätsindikators (30) gemessen. Die Bugs der Produkte Blau und Rot konnten im Betrachtungszeitraum mehrheitlich deutlich in unter 24 Stunden gelöst werden. Auffällig ist insbesondere der erste Betrachtungszeitraum des Produkts Rot; dort besaßen circa 50% der P1-Cases den gleichen Wert. Zudem fallen 10% der Cases, deren gemessenes Zeitintervall mehrere Jahre umfasst, auf. Eine mögliche Erklärung ist, dass Anpassungen am Prozess noch nicht in der Datenbank abgebildet wurden (vgl. Kapitel 5.1 und Kapitel 3.2). Die Enhancements der Produkte Blau und Rot werden ebenso wie die Bugs mehrheitlich in den einzelnen Betrachtungsperioden in unter 12 Stunden gelöst.

7.10 Kundenzufriedenheit

7.10.1 Wie zuverlässig ist der Software-Anbieter hinsichtlich der Termin-Absprachen?

Die Deadlines können Zusagen eines verbindlichen Termins gegenüber dem Kunden sein. Des Weiteren können Deadlines aufgrund der Umsetzungen von rechtlichen Belangen in der Software gesetzt werden. Somit können die verletzten Deadlines einen Hinweis auf die Zuverlässigkeit hinsichtlich der Terminabsprachen des Software-Anbieters mit dem Kunden geben. Der Qualitätsindikator (31) betrachtet den Anteil der verletzten Deadlines aus Kundensicht beim Zustandsübergang nach CLOSED+FIXED. Der Anteil der nicht eingehaltener Deadlines ist unabhängig vom betrachteten Produkt auf einem sehr hohen Niveau. Der Anteil hinsichtlich der Bugs des Produkts Blau sinkt jedoch in der letzten Periode deutlich ab.

7.10.2 Ändert sich die Kommunikationsstruktur zum Kunden?

Diese Fragestellung beschäftigt sich damit, inwiefern das Kundenportal zum KISTERS-Bugzilla genutzt wird. Hierzu wird der Anteil der SupportCalls, die von unternehmensexternen Personen eingestellt werden, bezogen auf alle SupportCalls betrachtet (siehe Qualitätsindikator (32)). Hinsichtlich der Produkte Blau und Rot wird ein steigender Anteil der SupportCalls durch Unternehmensexterne eingestellt. Dies zeigt, dass das Internetportal zunehmend an Bedeutung in der Kommunikationsstrategie von KISTERS gewinnt.

Die über das Kundenportal eingestellten Cases erhalten automatisch den Typ SupportCall zugewiesen. Betrachtet man die geänderte Kommunikationsstruk-

turen und den im Qualitätsindikator (03) beschriebenen hohen Anteil an Typ-Wechseln von SupportCall nach Bug, so besteht bei häufigen Typ-Wechseln die Gefahr, dass sich unter Umständen die Interpretation des Typs SupportCall weg von der Kundenanfrage hin zu einem „Unclassified“-Typ entwickelt.

7.11 Flusskarten

Die Flusskarten visualisieren die gelebten Prozesse anhand der dokumentieren Zustandsübergänge. Hierzu werden die Zustände des KISTERS-Bugzilla als Knoten und die möglichen Zustandsübergänge als gerichtete Kanten in einem Graphen visualisiert. Die beobachteten Zustandsübergänge zwischen den Zuständen, werden zusammengefasst und als Kanten in entsprechender Stärke im Graphen dargestellt (siehe Abb. 7.6).

Die Flusskarten werden werkzeugunterstützt erstellt. Hierzu wurde der Prototyp eines Editors entwickelt (siehe Abb. 7.5), in dem das Layout der Flusskarte festgelegt wird und die Flusskarte generiert wird.

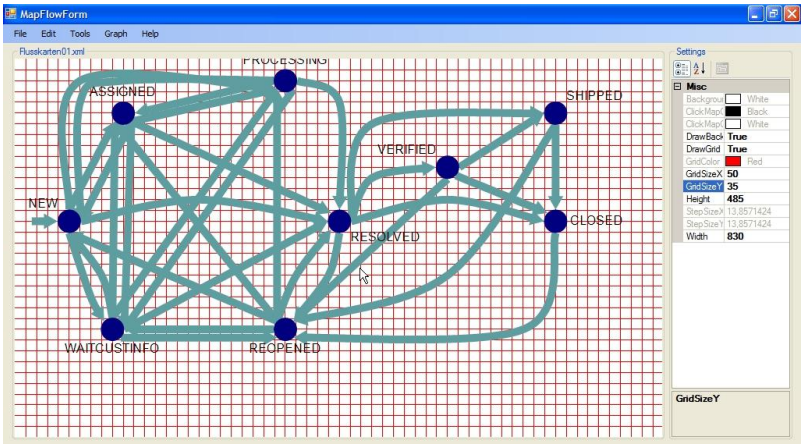


Abbildung 7.5: Prototyp des Editor zum Entwurf von Flusskarten

Anhand der Stärke der verschiedenen Kanten im Graphen lassen sich rein visuell die hauptsächlich genutzten Zustandsübergänge ermitteln, so dass die benutzten Prozessattribute hervortreten. Dies erlaubt es, den gelebten Prozess zu identifizieren. Insbesondere lassen sich ungenutzte beziehungsweise kaum genutzte Zustände erkennen (siehe Abb. 7.6). Anhand der Differenz zwischen eingehenden und ausgehenden Transitionen lassen sich mögliche Ressourcenengpässe erkennen. Jedoch lassen sich längere Verweildauern in einem Zustand nicht darstellen. Eine mögliche Erweiterung wäre die Visualisierung der Anzahl der Cases zu Beginn beziehungsweise zum Ende des Betrachtungszeitraums anhand der Knotengröße.

Die Flusskarten sind zur Visualisierung der Cases vom Typ SupportCall aufgrund des hohen Verknüpfungsgrades zwischen den Zuständen ungeeignet. Ebenso wird das Feld Resolution nicht ausreichend berücksichtigt. Zudem müssen die im Qualitätsindikator (03) beschrieben Typ-Wechsel (siehe Abb. 6.3) bei der Betrachtung berücksichtigt werden. Beispielsweise kann ein Case vom Typ SupportCall unter Umständen in den Zustand WAITCUSTINFO zum Bug wechseln.

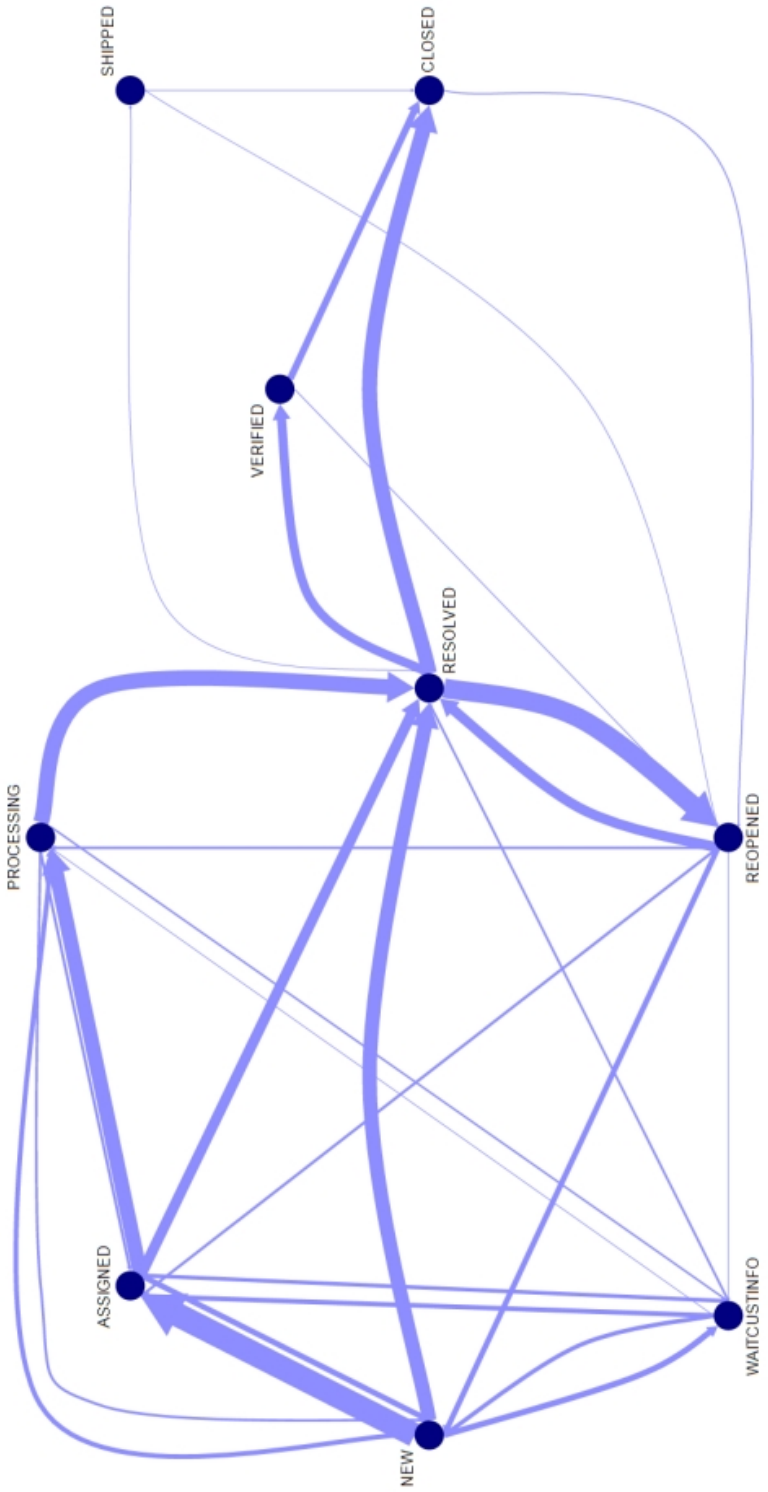


Abbildung 7.6: Flusskarte für die Enhancements des Produkt Blau

8 Zusammenfassung und Ausblick

Inhalt

In dieser Arbeit wurden die Softwareentwicklungs- und Wartungsprozesse der Firma KISTERS anhand von gesammelten Change-Request Daten analysiert. Der Schwerpunkt dieser Ausarbeitung ist es, einzelne Prozessmerkmale anhand von Change-Request Daten sichtbar zu machen, damit die laufenden Entwicklungsprozesse transparent werden.

In einem ersten Schritt wurde ein bidirektionales Qualitätsmodell für das KISTERS-Umfeld zur Analyse der Softwareentwicklungs- und Wartungsprozesse entwickelt.

Dazu wurden zuerst die möglichen Informationsbedürfnisse bezüglich der Prozesse gesammelt. Die möglichen Informationsbedürfnisse wurden unter anderem anhand von Prozessbewertung (wie CMMI), dem IT-Service Management (wie ITIL) und dem betriebswirtschaftlichen Kontext (betriebswirtschaftliche Kennzahlen) abgeleitet, um die unterschiedlichen Betrachtungsaspekte der Softwareentwicklungs- und Wartungsprozesse hinreichend zu berücksichtigen. Zusätzlich wurden die konkreten Informationsbedürfnisse anhand der Qualitätsziele der Firma KISTERS ergänzt.

Anschließend wurden anhand der verschiedenen Informationsbedürfnisse die Qualitätseigenschaften und ihre Beziehung untereinander modelliert. Die konkreten Informationsbedürfnisse wurden dabei in Form von Fragestellungen formuliert. Hierdurch wurde eine ausreichende Flexibilität des Modells gewährleistet, um das Modell auch bei künftig auftretenden Fragestellungen flexibel anpassen zu können.

Die zweite Stoßrichtung des bidirektionalen Modells umfasst die Qualitätsmerkmale und Qualitätsindikatoren. Die Qualitätsindikatoren verbinden die technische Sichtweise der erfassten Qualitätsmerkmale mit der Sichtweise der Qualitätseigenschaften, indem die Qualitätsindikatoren die erfassten Qualitätsmerkmale und deren Beziehung zu den Qualitätseigenschaften beschreiben. Hierzu wurden geeignete Qualitätsmerkmale zur Analyse der von KISTERS gesammelten Change-Request Daten bestimmt und deren Eignung im Rahmen der Diskussion erörtert.

Bei der Entwicklung des konkreten bidirektionalen Qualitätsmodells für die Fir-

ma KISTERS konnte ein schlankes Beziehungsgeflecht zwischen den einzelnen Modellelementen verwirklicht werden. Dies erleichtert die Interpretation der Beziehung zwischen den Qualitätsindikatoren und den Qualitätseigenschaften, so dass sich potentielle Ansätze zur Prozessverbesserung leichter identifizieren lassen. Des Weiteren erlaubt es das schlanke Beziehungsgeflecht, das Qualitätsmodell flexibel zu erweitern, da die Anzahl der Interdependenzen gering ist.

In einem explorativen Vorgehen wurden die Qualitätsmerkmale anhand der entwickelten Metriken mithilfe des Werkzeugs BugzillaMetrics ausgewertet und anschließend validiert. Aufgrund der flexiblen Definition konnten die Metriken in zahlreichen Iterationsschritten weiter verfeinert werden. Zudem wurden neue Anforderungen (Bsp. `ccCount` und `Hours Worked`) beziehungsweise Verbesserungen (Bsp. `IntervalLengthCalculator`) der deklarativen Sprache zur Metrikdefinition entwickelt.

In der abschließenden Diskussion wurden die Ergebnisse der Analyse mithilfe des für KISTERS entwickelten Qualitätsmodells interpretiert. Die auf der Grundlage der Metriken ermittelten Ausprägungen der Qualitätsmerkmale wurden anhand der in den Qualitätsindikatoren beschriebenen Beziehung zu den Qualitätseigenschaften erörtert. Hierbei wurde zugleich die Aussagekraft der einzelnen Qualitätsindikatoren hinsichtlich der Qualitätseigenschaften untersucht. Die entwickelten Qualitätsindikatoren unterstützen die Firma KISTERS hinsichtlich ihrer Informationsbedürfnisse bezüglich der Softwareentwicklungs- und Wartungsprozesse.

8.1 Ausblick

Die Analyse der Softwareentwicklungs- und Wartungsprozesse sollte kontinuierlich im Rahmen eines Messprozesses erfolgen (vgl. [ISO 15939]). Hierzu bieten sich die gesammelten Change-Request Daten an, da diese aufgrund der Integration des Werkzeugs Bugzilla in die Entwicklungsprozesse sowohl kostengünstig als auch zeitnah zu ermitteln sind.

Hierzu sollte die Nutzerakzeptanz des Werkzeugs Bugzilla durch die Einführung eines aufgabenorientierten Interfaces, welches den Anwender der Software entlastet, erhöht werden. Hingegen ist es problematisch, den Nutzer von Bugzilla durch die Verwendung von Default-Werten zu entlasten, da dies eine Auswertung der in Bugzilla gesammelten Change-Requests unter Umständen erschweren kann.

Durch die einfach zu erlernende Sprache bei der Definition von Metriken des Werkzeugs BugzillaMetrics ist die Einstiegshürde relativ gering. Das in dieser Arbeit entwickelte bidirektionale Qualitätsmodell und die dokumentierten Erfahrungen können als Ausgangspunkt zur Implementierung eines Messprozesses für die Firma KISTERS dienen.

Aufgrund der Flexibilität des bidirektionalen Qualitätsmodells lassen sich zusätzliche Datenquellen, wie zum Beispiel Konfigurationsverwaltungssysteme, flexibel einbinden. Ein kontinuierlicher Messprozess kann sowohl mögliche Verbesserungspotentiale der Entwicklungsprozesse aufdecken als auch den Erfolg der Verbesserungsmaßnahmen dokumentieren.

Jedoch ist es eine Herausforderung, die Ergebnisse der Analyse für die verschiedenen Anspruchsgruppen (engl. stakeholder) adressatengerecht zu kommunizieren. Ein viel versprechender Ansatz ist die Erweiterung durch die QualityModel Tools [Bugb]. Dieser Ansatz erlaubt es, das Beziehungsgeflecht zwischen Qualitätseigenschaften und Qualitätsindikatoren werkzeugunterstützt zu modellieren und auszuwerten. Die Zielgruppe der QualityModel Tools sind die Prozessverantwortlichen. Geeignete Visualisierungen für andere Anspruchsgruppen sind „Tree Maps“ zur Visualisierung der Indexzahlen oder die in dieser Arbeit entwickelten Flusskarten. Hierdurch können unter Umständen bestehende Vorbehalte gegenüber den Messprozessen gemindert werden.

Literaturverzeichnis

[Buga] *Bugzilla.* http://www.bugzilla.org/.

[Bugb] BugzillaMetrics: *BugzillaMetrics.* http://www.bugzillametrics.org/.

[Drö98] Dröschel, Wolfgang: *Inkrementelle und objektorientierte Vorgehensweisen mit dem V-Modell 97.* Oldenbourg, 1998.

[ED07] Ebert, Christof und Reiner Dumke: *Software Measurement: Establish, Extract, Evaluate, Execute.* Springer, 2007.

[ER03] Endres, Albert und Dieter Rombach: *A Handbook of Software and Systems Engineering : Empirical Observations, Laws, and Theories.* Addison Wesley, 2003.

[FP98] Fenton, Norman E. und Shari Lawrence Pfleeger: *Software Metrics: A Rigorous and Practical Approach.* PWS Publishing Co, 1998.

[GVG05] Gadatsch, Andreas, Frank Victor und Holger Günther: *IT-Controlling realisieren: Praxiswissen für IT-Controller, CIOs und IT-Verantwortliche.* Vieweg+Teubner Verlag, 2005.

[IEE90] IEEE: *IEEE Standard Glossary of Software Engineering Terminology (IEEE Std. 610.12-1990)*, 1990.

[IEE96] IEEE: *ISO/IEC 12207 - Standard for Information Technology - Software life cycle processes (IEEE/EIA 12207.0-1996)*, 1996.

[IEE03] IEEE: *1490 - IEEE Guide Adoption of PMI Standard – A Guide to the Project Management Body of Knowledge (IEEE Std. 1490-2003)*, 2003.

[IEE04a] IEEE: *IEEE Standard for a Software Quality Metrics Methology (IEEE Std. 1061-1998 (R2004))*, 2004.

[IEE04b] IEEE: *Standard 1061 – IEEE Standard for a Software Quality Metrics Methology (IEEE Std. 1061-1998 (R2004))*, 2004.

[ISO01] ISO/IEC: *Standard 9126-1 – Software engineering - Product quality - Part 1: Quality model (ISO/IEC Std. 9126-1:2001(E))*, 2001.

[ISO03a] ISO/IEC: *Technical Report 9126-2 – Software engineering - Product quality - Part 2: External metrics (ISO/IEC TR 9126-2:2003(E))*, 2003.

[ISO03b] ISO/IEC: *Technical Report 9126-3 – Software engineering - Product quality - Part 3: Internal metrics (ISO/IEC TR 9126-3:2003(E))*, 2003.

[ISO04] ISO/IEC: *Technical Report 9126-4 – Software engineering - Product quality - Part 4: Quality in use metrics (ISO/IEC TR 9126-4:2004(E))*, 2004.

[ISO06] ISO/IEC: *Standard 14764 – Software Engineering - Software Life Cycle Processes - Maintenance (ISO/IEC Std. 14764:2006(E))*, 2006.

[ISO07] ISO/IEC: *Standard 15939 – Systems and software engineering - Measurement process (ISO/IEC Std. 15939:2007(E))*, 2007.

[ISO08] ISO/IEC: *Standard 12207 – Systems and software engineering - Software life cycle processes (ISO/IEC Std. 12207:2008(E))*, 2008.

[KIS08] KISTERS: *Software development in the RMS division of KISTERS*, 2008.

[KIS09] KISTERS: *Qualitätsziele*, 2009.

[KJ08] Kulpa, Margaret K. und Kent A. Johnson: *Interpreting the CMMI: A Process Improvement Approach*. CRC Press, 2008.

[KMS07] Kesten, Ralf, Arno Müller und Hinrich Schröder: *IT-Controlling: Messung und Steuerung des Wertbeitrages der IT*. Vahlen Franz GmbH, 2007.

[Küt05] Kütz, Martin: *IT-Controlling für die Praxis: Konzeption und Methoden*. dpunkt.Verlag GmbH, 2005.

[Küt06] Kütz, Martin: *IT-Steuerung mit Kennzahlensystemen*. dpunkt.Verlag GmbH, 2006.

[Küt07] Kütz, Martin: *Kennzahlen in der IT Werkzeuge für Controlling und Management*. dpunkt.Verlag GmbH, 2007.

[LL07] Ludewig, Jochen und Horst Lichter: *Software Engineering*. dpunkt.Verlag GmbH, 2007.

[QMe] QMetrics: *BugzillaMetrics Manual*. http://www.qmetric.org/BugzillaMetrics/documentation.html.

[Sch06] Schlebusch, Heinz-Josef: *work instruction: bugzilla*, 2006.

[Sch08] Schlebusch, Heinz-Josef: *KISTERS Bugzilla Internal Training*, 2008.

[SL08] Schackmann, Holger und Horst Lichter: *Comparison of Process Quality Characterisitics Based on Change Request Data.* Software Process and Product Measurement, International Conferences: IWSM 2008, Metrikon 2008, and Mensura 2008, Munich, Germany, November 18-19, 2008. Proceedings, 2008.

[SP01] Sommerville, Ian und H E Philippsborn: *Software Engineering.* Addison-Wesley, 2001.

[SSM06] Simon, Frank, Olaf Seng und Thomas Mohaupt: *Code-Quality-Management: Technische Qualität industrieller Softwaresysteme transparent und vergleichbar gemacht.* dpunkt-Verlag, 2006.

[SWE04] *Software Engineering Body of Knowledge.* http://www.swebok.org/, 2004.

[vB04] Bon, J. van: *IT Service management: Eine Einführung.* Van Haren Publishing, 2004.

[Wal01] Wallmüller, Ernest: *Software-Qualitätsmanamgent in der Praxis.* Hanser Fachbuchverlag, 2001.

[Wal07] Wallmüller, Ernest: *SPI - Software Process Improvement mit CMMI und ISO 15504.* Hanser Fachbuchverlag, 2007.