RWTH Aachen

Department of Computer Science
Technical Report

Software & Systems Engineering Essentials 2013

Prof. Dr. Andreas Rausch and Dr. Marc Sihling

ISSN 0935–3232 · Aachener Informatik-Berichte · AIB-2013-05

RWTH Aachen · Department of Computer Science · March 2013

The publications of the Department of Computer Science of *RWTH Aachen University* are in general accessible through the World Wide Web.

http://aib.informatik.rwth-aachen.de/

Software & Systems Engineering Essentials 2013

Prof. Dr. Andreas Rausch, Dr. Marc Sihling (Hrsg.)

Keywords

Software Engineering, Software Development Process, Software Development Techniques

1. Einleitung

Software verändert unseren Alltag deutlich und nachhaltig. In der neuen IT-Welt installieren und nutzen auch technisch kaum versierte Anwender wie selbstverständlich Software auf mobilen Geräten - vom Smartphone über ihr Tablet bis hin zum Bordcomputer ihres Autos. Es ist einfacher denn je, Softwarelösungen auszuprobieren, up-to-date zu halten und bei Nicht-Gefallen wieder zu entfernen. Software wird damit zum Motor der Veränderung für Wirtschaft und Gesellschaft.

Technisch sind vor allem zwei Begriffe mit der neuen IT-Welt verbunden: Apps für unterschiedliche Plattformen bilden die Schnittstelle zum Benutzer. Die dafür nötigen Dienste kommen aus Servern in der Cloud. Für die Software-Entwicklung ergibt sich dadurch eine Fülle neuer Fragen. Die SEE2013 greift mit ihrem Schwerpunktthema die neuen Entwicklungsparadigma, Prozesse und Werkzeuge auf, die eine flexible, moderne Anwendungsentwicklung für die neue IT-Welt ermöglichen.

1.1. Überblick

Die Konferenz SEE2013 fand vom 27. Februar bis 1. März in Aachen im Rahmen der Multikonferenz SE2013 statt. Den Besuchern wurden parallel zu den Beiträgen der SE2013 16 Vorträge aus Industrie und Wirtschaft angeboten. Die Multikonferenz wurde von einer Ausstellung mit Vertretern von Werkzeugherstellern, Industriekunden und Beratungsfirmen begleitet.

1.2. Danksagung

Wie immer gilt ein besonderer Dank allen Autoren, die Ihre Vorträge für die SEE2013 eingereicht und für die Proceedings freigegeben haben. Auch den Mitgliedern des Programmkomitees sowie natürlich den Teilnehmern sei an dieser Stelle nochmals recht herzlich gedankt.

Inhaltsverzeichnis

1. Einleitung **i**
1.1. Überblick i
1.2. Danksagung i

2. Vorgehensmodelle **1**
2.1. OPAL - Ein Framework zur Entwicklung und projektspezifischen Anpassung von Vorgehensmodellen bei T-Systems (27.02.2013) 2
2.2. Allen Projekten ein gemeinsames Vorgehensmodell, jedem Projekt seine spezifische Arbeitsweise - und wie beides zusammen kommt (27.02.2013) 21
2.3. Application Lifecycle Management für die Entwicklung mobiler Anwendungen (27.02.2013) . . 35

3. Testen **55**
3.1. Toolwahl zu automatisierten Akzeptanztests - Was sollte ich vorher wissen? (27.02.2013) 56
3.2. Modultests in der Softwareentwicklung für mobile Geräte durch den Wissensträger mittels KBUnit (28.02.2013) 59

4. Agilität **75**
4.1. Agiler Einkauf (27.02.2013) 76
4.2. Einführung von Scrum (27.02.2013) 85
4.3. Agile Konzepte im Unternehmen verankern (28.02.2013) 99
4.4. Agile Software-Entwicklung in regulierten Umgebungen (28.02.2013) 104

5. Standards & Rechtliches **115**
5.1. Register Factory - Standards für die Cloud (27.02.2013) 116
5.2. SAGA im Werkvertrag (27.02.2013) 129
5.3. Ausschreibung von Software mit offener Lizenzierung für eine Institution der öffentlichen Finanzkontrolle in Österreich (27.02.2013) 141

6. Anforderungen **155**
6.1. „Make, Buy, Use oder Compose?“ als zentrales Element des Anforderungsmanagmentprozesses in einer großen Kommune (28.02.2013) 156
6.2. eBAföG - ein ehrlicher Erfahrungsbericht (01.03.2013) 172

7. Mobile Systeme **183**
7.1. Key considerations for high-quality mobile systems (28.02.2013) 184

8. Architektur **197**
8.1. Essential Architectural Views for Embedded Systems (01.03.2013) 199

A. Programmkomitee **217**

2. Vorgehensmodelle

Sessionüberblick

2.1. OPAL - Ein Framework zur Entwicklung und projektspezifischen Anpassung von Vorgehensmodellen bei T-Systems (27.02.2013) 2

2.2. Allen Projekten ein gemeinsames Vorgehensmodell, jedem Projekt seine spezifische Arbeitsweise - und wie beides zusammen kommt (27.02.2013) 21

2.3. Application Lifecycle Management für die Entwicklung mobiler Anwendungen (27.02.2013) 35

OPAL - Ein Framework zur Entwicklung und projektspezifischen Anpassung von Vorgehensmodellen bei T-Systems

Dr. Frank Marschall

Systems Integration, Process Improvement
T-Systems International GmbH
Dachauer Str. 651
80995 München
Frank.Marschall@t-systems.com

Abstract: Dieser Beitrag stellt die OPAL-Infrastruktur vor, die innerhalb der T-Systems zur Erstellung von Vorgehensmodellen und deren Einführung in Projekten genutzt wird. Es wird gezeigt, wie Vorgehensmodelle für unterschiedliche Projekttypen erstellt und in Projekten anhand definierter Regeln werkzeuggestützt angepasst werden können. Der Beitrag konzentriert sich hierbei zum einen auf die Möglichkeiten, die der Benutzer für das Tailoring hat. Zum anderen wird gezeigt, wie Prozessingenieure mit Hilfe des Frameworks die Konsistenz und fachliche Constraints für eine Vielzahl möglicher Varianten eines Vorgehensmodells automatisiert überprüfen können.

1. Motivation

Die Systems Integration der T-Systems ist weltweit in Software-Entwicklungsprojekten in zahlreichen Anwendungsdomänen tätig, wie z.B. Automotive, Public, etc. Hierbei wird eine Vielzahl von Technologien, wie z.B. Individualentwicklungen in verschiedenen Programmiersprachen oder aber Standardsoftware wie SAP verwendet. Zudem unterscheiden sich die einzelnen Projekte in Bezug auf Größe, Vorgehensmodell und nichtfunktionalen Anforderungen, wie beispielsweise dem Grad an Sicherheitskritikalität.

Dennoch müssen alle Projekte einheitlich festgelegten Qualitätsstandards genügen. Diese orientieren sich einerseits an der Größe und Art der Projekte. Andererseits müssen beispielsweise Projekte die sicherheitskritische Systeme erstellen, zusätzliche Maßnahmen zur Gewährleistung der Sicherheit dieser Systeme durchführen.

Um die Einhaltung dieser Qualitätsstandards zu gewährleisten, einheitliche und vergleichbare Ergebnisse in Projekten zu erhalten und die Konformität zu Standards wie beispielsweise CMMI [CM11] zu gewährleisten, gibt es innerhalb der T-Systems jeweils Standards für das Projektmanagement („PM Book“) und das Software-Engineering („SE Book“).

Um die verschiedenen Methodenbausteine weiterentwickeln, kombinieren und projektspezifisch zu einem durchgängigen Vorgehen kombinieren zu können, bedarf es einer einheitliche Prozessarchitektur und einer darauf aufbauenden technischen Infrastruktur. Diese OPAL („Organizational Process Asset Library“) genannte Infrastruktur ist Gegenstand dieses Beitrags.

2. Die OPAL-Infrastruktur

Die für Projekte relevanten Standards für das Projektmanagement und das Software-Engineering müssen zu durchgängigen Entwicklungsprozessen orchestriert werden. Hierfür wurden innerhalb der Prozessarchitektur „Standardprozesse“, wie in Abbildung 1 dargestellt, eingeführt.

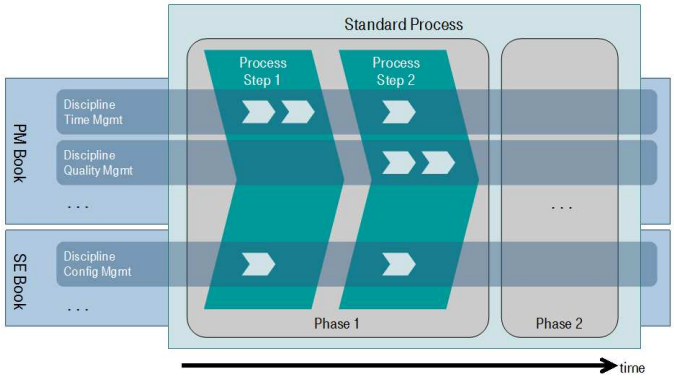


Abbildung 1: Schematische Darstellung des Prozessmodells

Die Prozessdomänen Projektmanagement und Software-Engineering gliedern sich in thematisch gegliederte „Books“ und Disziplinen, innerhalb derer einzelne Aktivitäten beschrieben sind. Standardprozesse orchestrieren diese Aktivitäten in Prozessschritten und Phasen. Die verschiedenen Standardprozesse bilden erschiedene Prozessmodelle wie z.B. agile oder inkrementelle Vorgehensweisen ab. Diese Systematik ist an das Konzept der Disziplinen und Phasen im „Rational Unified Process“ [Kru98] angelehnt.

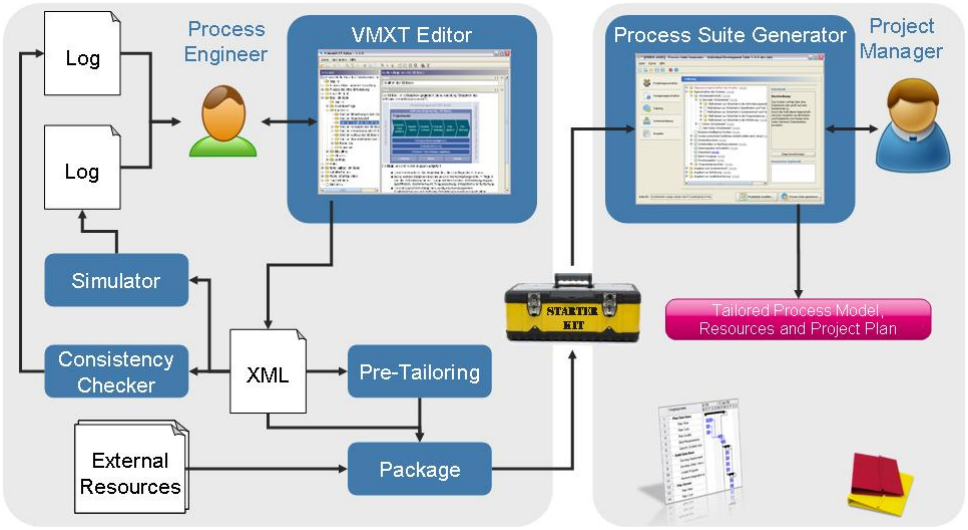


Abbildung 2: Überblick über das OPAL-Framework

Abbildung 2 zeigt einen schematischen Überblick über das OPAL-Framework.

2.1 Der Process Suite Generator

Auf der rechten Seite in Abbildung 2 findet sich der Process-Suite-Generator, mit dem ein Projekt seine Infrastruktur initial einrichten kann. Abbildung 3 skizziert die einzelnen Schritte dieses Vorgangs aus Sicht eines Projekts.

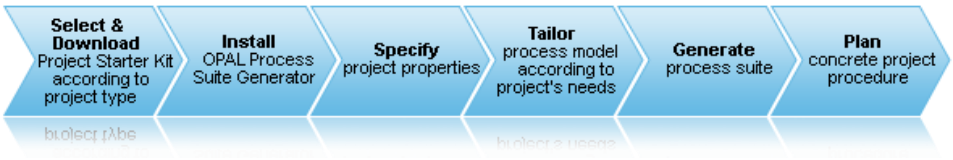


Abbildung 3: Vorgehen beim Einrichten einer Projektinfrastruktur

Starter-Kits sind vorkonfigurierte Pakete aus Process-Assets (also Prozesse, Methoden, sonstige Hilfsmittel). Diese können in das Werkzeug Process-Suite-Generator importiert werden. Der Process-Suite-Generator erlaubt es, ähnlich dem Projektassistenten des V-Modell XT [VM12], einen Standardprozess auszuwählen und das Vorgehensmodell anhand festgelegter Regeln projektspezifisch anzupassen („Tailoring“). Neben zahlreichen Kriterien, wie der Projektgröße, den verwendeten Technologien, Sicherheitskritikalität oder Festlegung der zu ermittelnden KPIs („Key Performance Indicators“) können hier auch Erweiterungen zur Unterstützung kundenspezifischer Vorgaben selektiert werden.

Dem Benutzer wird hierbei die Möglichkeit geboten, während des Tailorings jederzeit zu prüfen, welche konkreten Auswirkungen die Aus- oder Abwahl eines Features im

Vorgehensmodell hat. Darüber hinaus wird für ein erzeugtes Vorgehensmodell zum einen detailliert dokumentiert welche Auswirkung jede einzelne Entscheidung hatte. Zum anderen wird für jedes potentielle Element des Vorgehensmodells dokumentiert, aufgrund welcher Entscheidungen es Bestandteil der angepassten Modellvariante ist oder nicht.

Die Generierung einer projektspezifischen Infrastruktur erzeugt die folgenden Ergebnisse:

- Projektspezifische Prozessbeschreibung als HTML und PDF
- Verzeichnisstruktur mit projektspezifisch vorgefüllten Vorlagen und Guidelines
- Projektplanvorlage mit allen für das Projekt relevanten Aktivitäten
- Eine Tailoring-Übersicht mit der Dokumentation aller Entscheidungen und deren Auswirkungen

In Abhängigkeit vom gewählten Standardprozess kann interaktiv eine erste Version eines Projektplans erstellt werden, welcher bereits die geplanten Entwicklungsstufen und Releases, sowie deren Abhängigkeiten untereinander, berücksichtigt.

2.2 Von Process-Assets zu Starter-Kits

Auf der linken Seite von Abbildung 2 sind die Komponenten zur Pflege und Weiterentwicklung der Vorgehensmodelle abgebildet. Hier erfolgen die Erstellung der Books, die Orchestrierung zu Standardprozessen, die Festlegung der Tailoring-Möglichkeiten in Form eines Feature-Modells [KC90], sowie die Qualitätssicherung durch statische Prüfungen und Simulation. Die geprüften Ergebnisse werden im Wesentlichen als XML-Dokumente bereitgestellt und zusammen mit weiteren Hilfsmitteln in Form sogenannter Starterkits (z.B. für SAP-Entwicklung oder Individual-Entwicklung) verfügbar gemacht.

Der Beitrag konzentriert sich hierbei auf einen wesentlichen, interessanten Aspekt dieser Tätigkeiten: der automatisierten Überprüfung, ob alle, durch das Tailoring denkbaren, Varianten eines Vorgehensmodells fachliche oder syntaktische Eigenschaften (also Invarianten) aufweisen. Hierzu wird auf bereits in [KMO11] vorgestellten Ergebnissen aufgebaut.

Dort wurde vorgestellt, wie Feature-Modelle in logische Ausdrücke überführt werden. Ein SAT-Solver wertet diese Ausdrücke aus und sorgt dafür, dass Benutzereingaben immer wieder zu einem gültigen Feature Modell führen. Die gewählten Features repräsentieren die Auswahl der gewünschten Eigenschaften eines Vorgehensmodells. Jedes Feature kann durch seine Auswahl eine Reihe von Modelloperationen aktivieren. Ist eine Modelloperation aktiviert, entfernt diese Elemente aus dem Vorgehensmodell oder fügt neue hinzu. Abbildung vier skizziert diesen Ansatz informell.

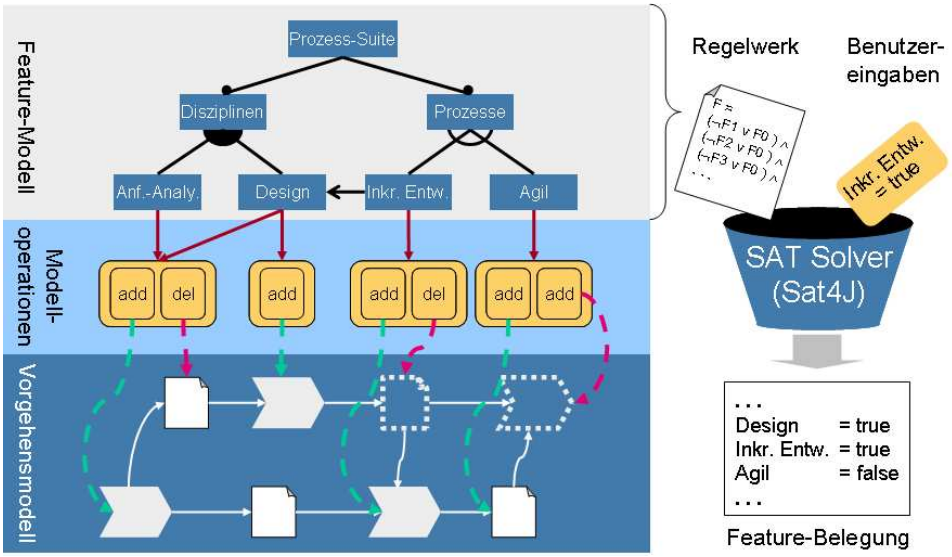


Abbildung 4: Ansatz zum Tailoring von Vorgehensmodellen aus [KMO11]

Mittlerweile wurde das OPAL-Framework dahingehend erweitert, dass das aussagelogische Regelwerk auch die Beziehungen zwischen Features und Modelloperationen und weiter zu den einzelnen Elementen des Vorgehensmodells abbildet. Der SAT-Solver kann anhand dieses erweiterten Regelwerks dazu verwendet werden, um den gesamten Vorgang des Tailorings zu berechnen. Hierbei werden boolesche Variablen für alle Modelloperationen eingeführt, welche Aussagen, ob eine Modelloperation für die aktuelle Feature-Belegung aktiv ist oder nicht. Für jedes Element des Vorgehensmodells gibt es eine boolesche Variable, die aussagt, ob das Modellelement für eine Feature-Belegung Teil des Vorgehensmodells nach dem Tailoring ist.

Anhand eines so erweiterten Regelwerks kann ein SAT-Solver auch genutzt werden, um zu überprüfen, ob bestimmte Konstellationen aus belegten Features, aktivierten Modelloperationen und im Vorgehensmodell verbleibenden Elementen möglich sind oder nicht. Anhand solcher Konstellationen lassen sich Invarianten formulieren, deren Überprüfung mit Hilfe des SAT-Solvers nun sehr einfach und automatisiert erfolgen kann. Dies können fachliche Constraints, wie z.B. „Keine Verantwortung für Abnahmetests bei Dienstverträgen“, oder auch Konsistenzbedingungen (z.B. „Jede Aktivität hat immer eine ausführende Rolle“) sein. Zur Überprüfung wird beispielsweise eine Verletzung der Invariante formuliert. Der SAT-Solver überprüft, ob des im bestehenden Regelwerk hierfür eine Lösung gibt – also eine Auswahl von Features, für welche die geforderte Eigenschaft verletzt wird. Ist dies der Fall, so genügt die Modellierung noch nicht den Anforderungen.

Im OPAL-Framework wurde das Unit-Test-Framework JUnit erweitert, um solche Überprüfungen automatisiert auf Basis bestehender JUnit-Werkzeuge durchführen zu können.

3. Bewertung

3.1 Erfahrungen aus Projekten

In den Projekten trifft der vorgestellte Ansatz auf sehr positive Resonanz, da er den Beteiligten zu Projektbeginn Arbeit abnimmt:

- Das Tailoring des Vorgehens wird wesentlich vereinfacht.
- „Zusatzverordnungen“, wie z.B. zum Thema Security, die neben dem Vorgehensmodell berücksichtigt werden müssen, sind in das Vorgehensmodell integriert und müssen nicht mehr zu Projektbeginn gesucht gesucht und interpretiert werden.
- Für das Projekt unwesentliche Teile des Vorgehensmodells entfallen und führen zu einer Verschlankung der Standards für das Projekt.
- Vorlagen sind bereits auf das Projekt zugeschnitten und mit allen Projektkenndaten befüllt.

Umgekehrt führt diese erhöhte Akzeptanz dazu, dass Standards bekannt sind und besser eingehalten werden. So ist es z.B. für einen Projektleiter oftmals einfacher einen generierten Projektplan mit allen relevanten Aktivitäten für seine Bedürfnisse anzupassen, als diesen neu zu erstellen. Im letzteren Fall ist die Gefahr groß, dass der Projektplan sich nicht an den Standards orientiert und wesentliche Maßnahmen schlicht vergessen werden.

Wie sich in diversen Befragungen und Usability-Tests erwiesen hat, ist die Möglichkeit, jederzeit die Auswirkungen ihrer Entscheidungen einschätzen zu können, ein wesentliches Akzeptanzkriterium für die Anwender. Auch die Möglichkeit im Nachhinein die Ursache für die Ausprägung eines Vorgehensmodells nachvollziehen zu können, ist entscheidend. Für Assessoren ist es zudem wichtig, dass Projekte gerade kritische Entscheidungen (z.B. „Warum wird kein Systemintegrationstest benötigt?“) nachvollziehbar dokumentieren.

3.2 Erfahrungen von Prozessingenieuren

Insbesondere die durch den Einsatz von Feature-Modellen erreichte Flexibilität beim Tailoring stößt bei Projekten wie bei Prozessingenieuren auf positive Resonanz. Dennoch ist es für Projekte wichtig Transparenz über die Auswirkungen des Tailorings zu haben und die detaillierten Auswirkungen ihrer Entscheidungen nachvollziehen zu können.

Die Verwendung eines aussagelogischen Regelwerks hat sich hierbei in der Praxis bewährt. Dies betrifft zum einen die Durchgängigkeit bei der Durchführung des Tailorings und der einfachen Vorhersage von Auswirkungen einzelner Entscheidungen.

Die Möglichkeit Invarianten zu überprüfen, vereinfacht darüber hinaus die Einarbeitung noch unerfahrener Prozessingenieure. Auch lassen sich – analog zum Refactoring von Quellcode – so schnell unerwünschte Nebeneffekte bei der Umstrukturierung von Teilen des Vorgehensmodells oder des Feature-Modells entdecken.

Der Einsatz des hier vorgestellten Ansatzes erfordert von Prozessingenieuren jedoch ein hohes Maß an Präzision und Grundverständnis der Formulierung logischer Aussagen. Geraten mehrere Modelloperationen in Konflikt, da sie die selben Modellelemente manipulieren möchten, so ist zwar durch die zugrundeliegende Semantik ein deterministisches Ergebnis sichergestellt. Es bedarf aber in solchen Fällen einiger handwerklicher Geschicklichkeit, um das gewünschte Verhalten korrekt zu spezifizieren, da beispielsweise eine Priorisierung von Modelloperationen nicht möglich ist.

3.3 Zusammenfassung und Ausblick

Der vorliegende Beitrag stellt dar, wie unterschiedliche „Process-Assets“ für unterschiedliche Vorhaben zu sogenannten Starter-Kits zusammengestellt werden. Diese können auf vielfältige, aber fest vorgegebene Weise an die Bedürfnisse ihrer Anwender angepasst werden. Dieser Ansatz erlaubt es spezielle Themen, die zwar sehr wichtig sein können, aber ansonsten nicht in einen allgemeinen Standard einfliesen würden, zu berücksichtigen. Beispielsweise wird so sichergestellt, dass sich spezielle Aktivitäten für die Erstellung hoch sicherheitskritischer Systeme direkt im Standard integriert finden und nicht als „Satellitenverordnungen“ im Intranet übersehen werden.

Um die hiermit einhergehende Komplexität, gerade bei Änderungen der Prozesse und Tailoring-Kriterien, noch beherrschen zu können, bietet die Verwendung eines SAT-Solvers hervorragende Möglichkeiten. Die an der Schaffung des Frameworks Beteiligten sind überzeugt, dass sich diese Techniken auch erfolgreich auf zahlreiche andere Anwendungen, bei denen Modelle erstellt und manipuliert werden, übertragen lassen.

Das vorgestellte Framework wird innerhalb der T-Systems mittlerweile auch von anderen Prozessfamilien, wie dem Service-Management, eingesetzt. Die aktuell bestehenden vier Starter-Kits werden stetig um Aspekte erweitert, die bisher als zu speziell galten. Zudem werden weitere Starter-Kits erschaffen. Die bestehende Werkzeug-Infrastruktur wird entsprechend derer Anforderungen erweitert, beispielsweise um diverse Plugins, um spezifische Hilfsmittel, wie domänenspezifische Checklisten, bei Projekt- (oder Sevice-)start zu generieren.

Literaturverzeichnis

[CM11] CMMI® - Capability Maturity Model Integration, Carnegie Mellon, Software Engineering Institute, Pittsburgh, USA, Webseite: http://www.sei.cmu.edu/CMMI, 2011

[KC90] Kang, K.C. and Cohen, S.G. and Hess, J.A. and Novak, W.E. and Peterson, A.S., Feature-oriented domain analysis (FODA) feasibility study, Technical Report CMU/SEI-90-TR-021, SEI, Carnegie Mellon University, November 1990

[KMO11] Georg Kalus, Frank Marschall, Philipp Offermann: Einsatz von Feature-Modellen beim Tailoring von Vorgehensmodellen, Proceedings zur SEE 2011, Juni 2011

[Kru98] Philippe Kruchten: The rational unified process. (An introduction). Addison-Wesley, Reading MA u. a. 1998, ISBN 0-201-60459-0.

[VM12] V-Modell XT, Webseite. http://www.v-modell-xt.de, 2012

OPAL

Ein Framework zur Entwicklung und projektspezifischen Anpassung von Vorgehensmodellen bei T-Systems

Dr. Frank Marschall

Inhalt

- Ziele und Überblick
- OPAL aus Projektsicht (→ Tailoring)
- OPAL für Prozessingenieure (→ Eigenschaften prüfen)
- Zusammenfassung

OPAL-Framework.
T-Systems

Rahmenbedingungen

Zahlreiche unterschiedliche Projekte

- Verschiedene Anwendungsdomänen (Public, Automotive, Telekommunikation, …)
- Verschiedene Technologien (Java, C#, SAP, …)
- Verschiedene Projektgrößen
- Unterschiedliche Sicherheitskritikalität
- …

Aufgaben:

- Vorgehensmodell(e) für viele Arten von Projekten
- Einhaltung von Qualitätsstandards
- Möglichst hoher Grad an Wiederverwendung

International:

Niederlassungen
Ländern, weltweit

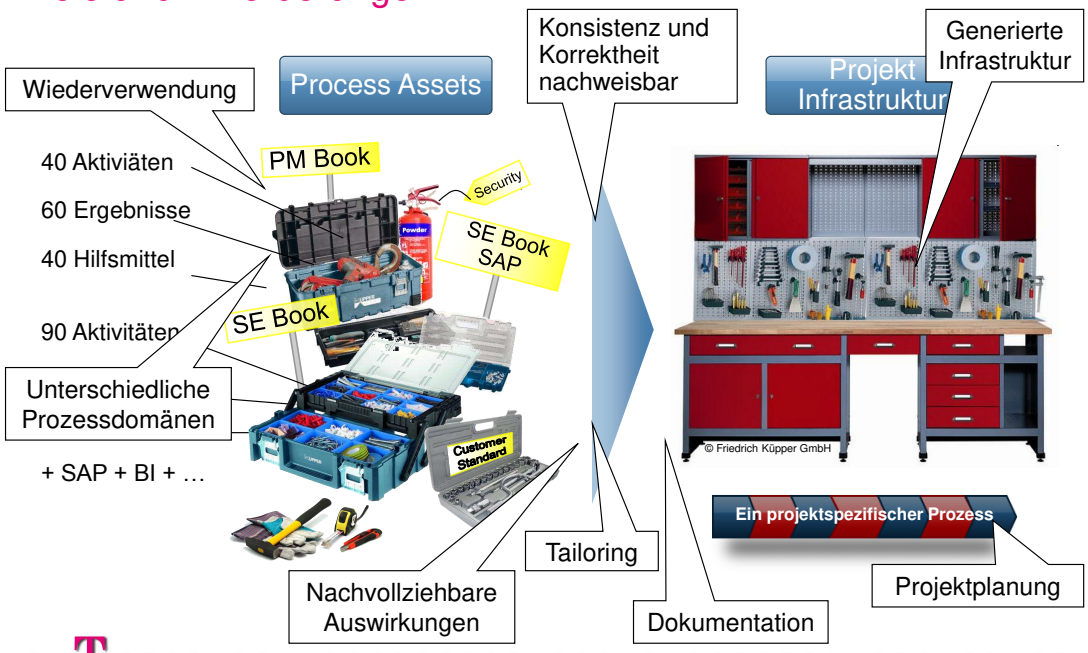
d: 2011)

Kundenfokus:

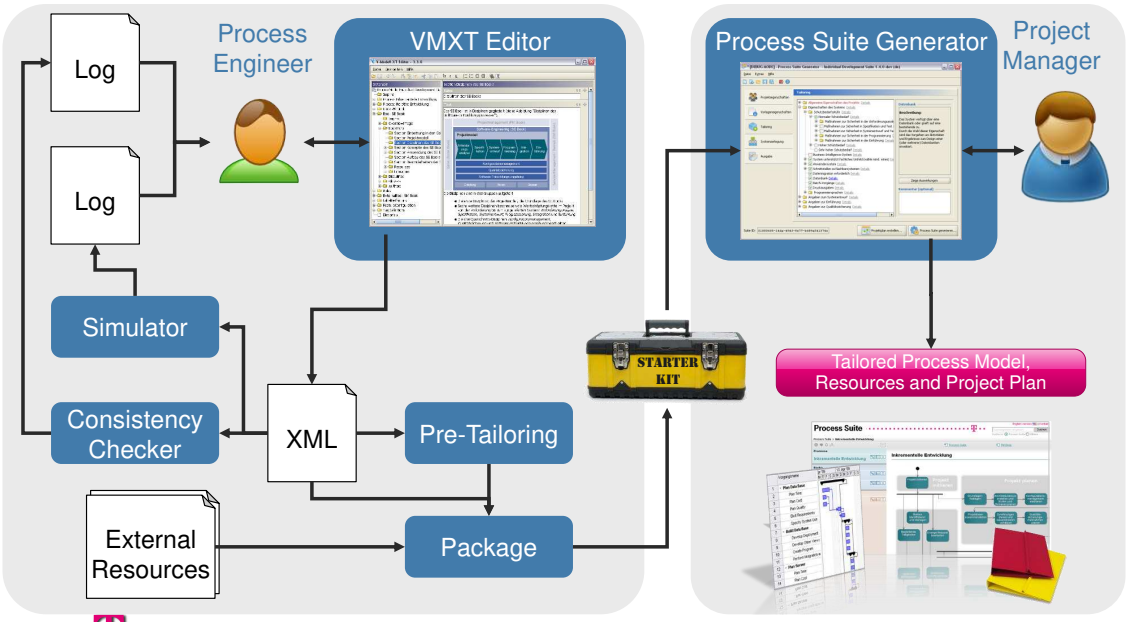
Großunternehmen und multi-nationale Konzerne sowie der Public & Health-Sektor

ca. 48.200 (25.500 in Deutschland, 22.700 international) (Stand: 2011)

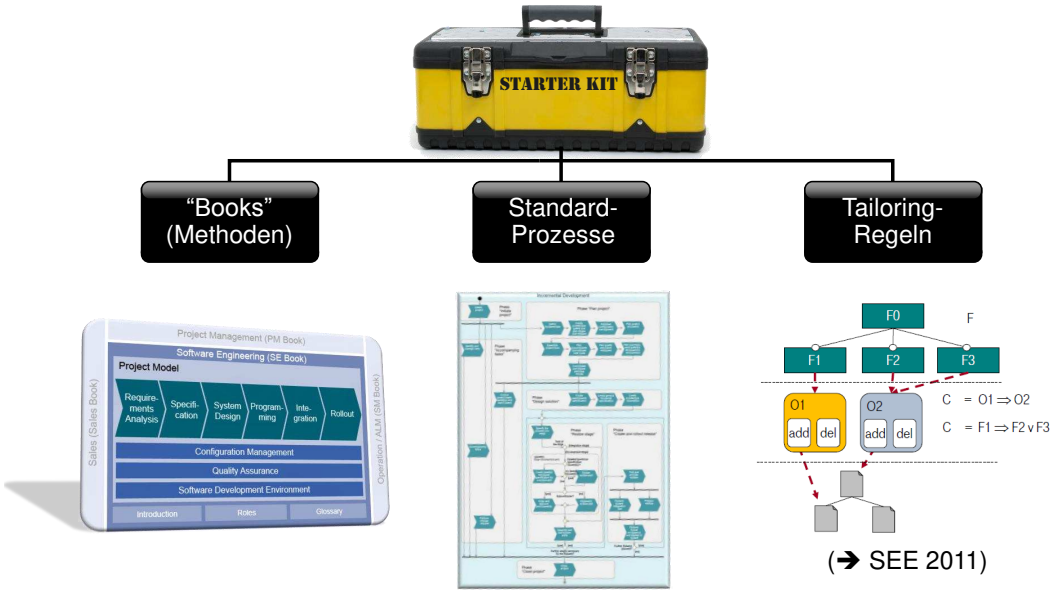
OPAL-Framework.
Ziele und Anforderungen



OPAL-Framework.
Überblick: Organizational Process Asset Library (OPAL)

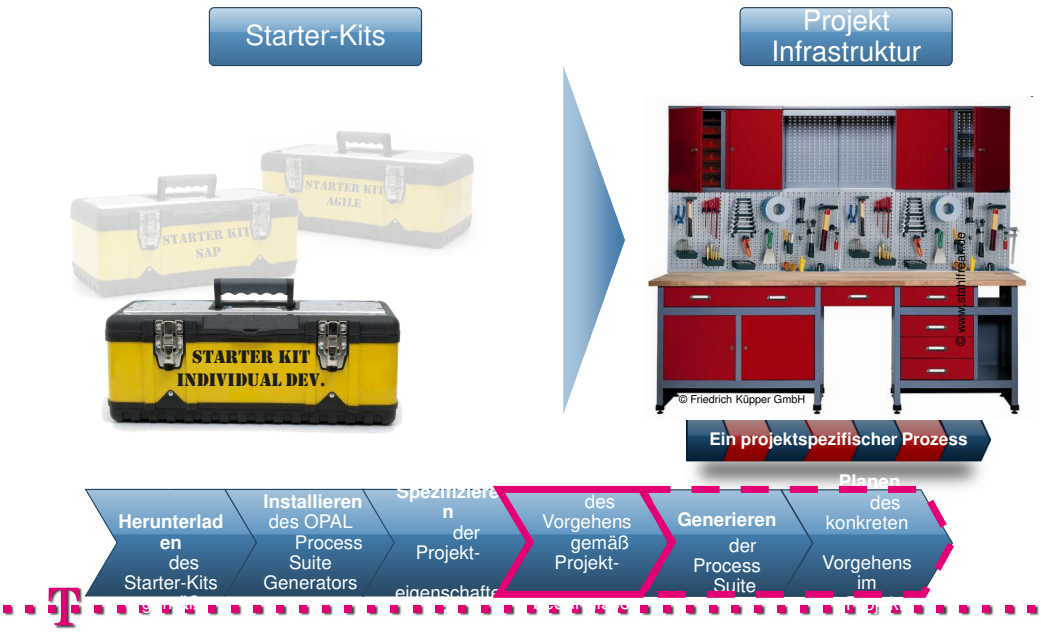


OPAL-Framework.
Starter-Kits: Inhalt

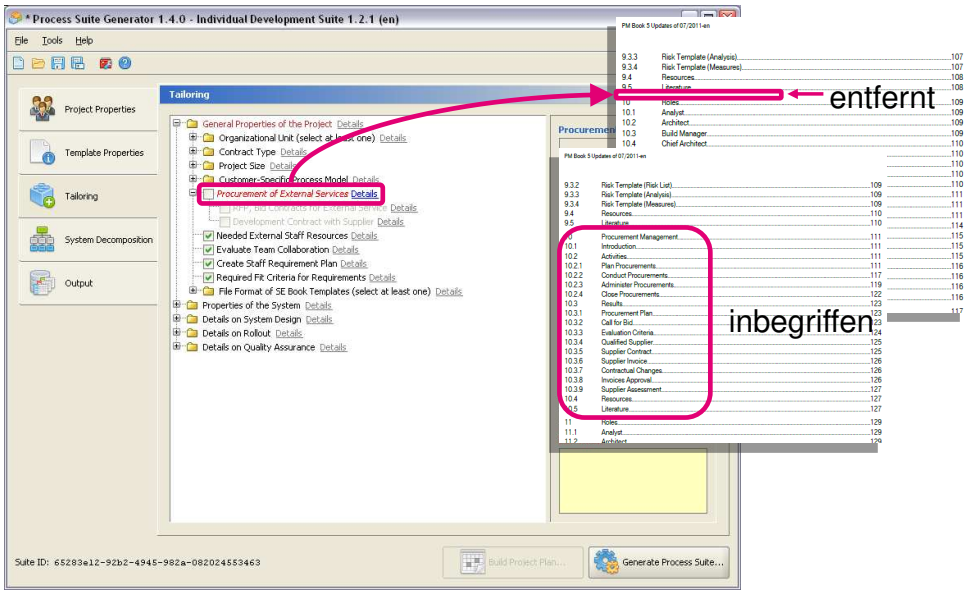


(➔ SEE 2011)

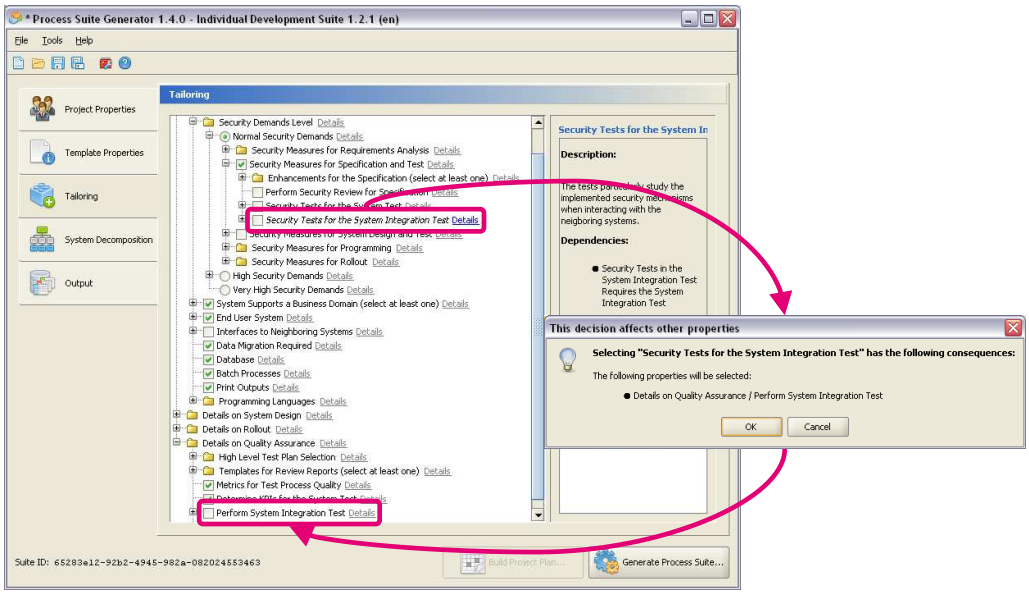
OPAL-Framework.
Process Suite Generator: vom Starter-Kit zur Projektinfrastruktur.



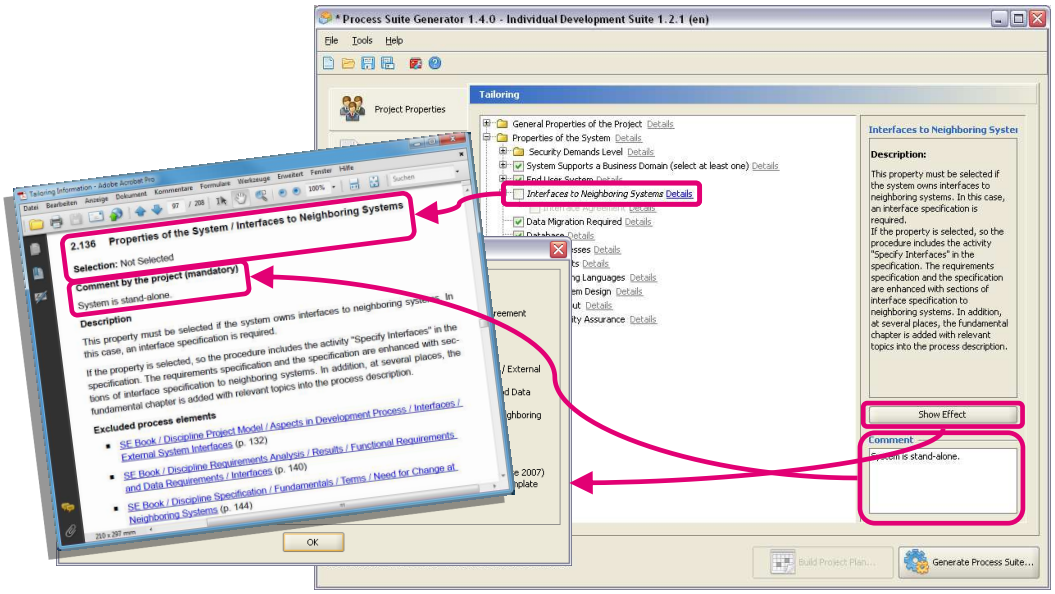
OPAL-Framework.
Auswirkungen des Tailorings.



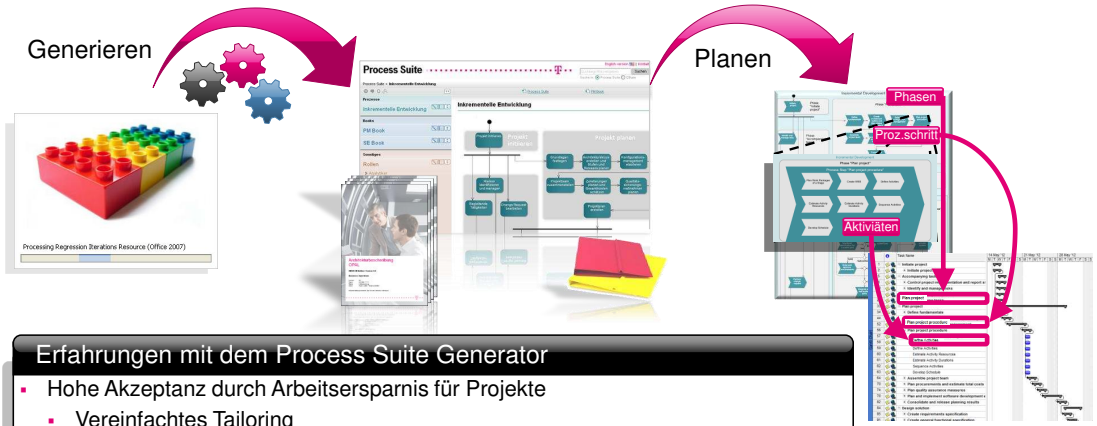
OPAL-Framework.
Cross Tree Constraints.



OPAL-Framework.
Anwenderhilfe und Dokumentation des Tailorings.



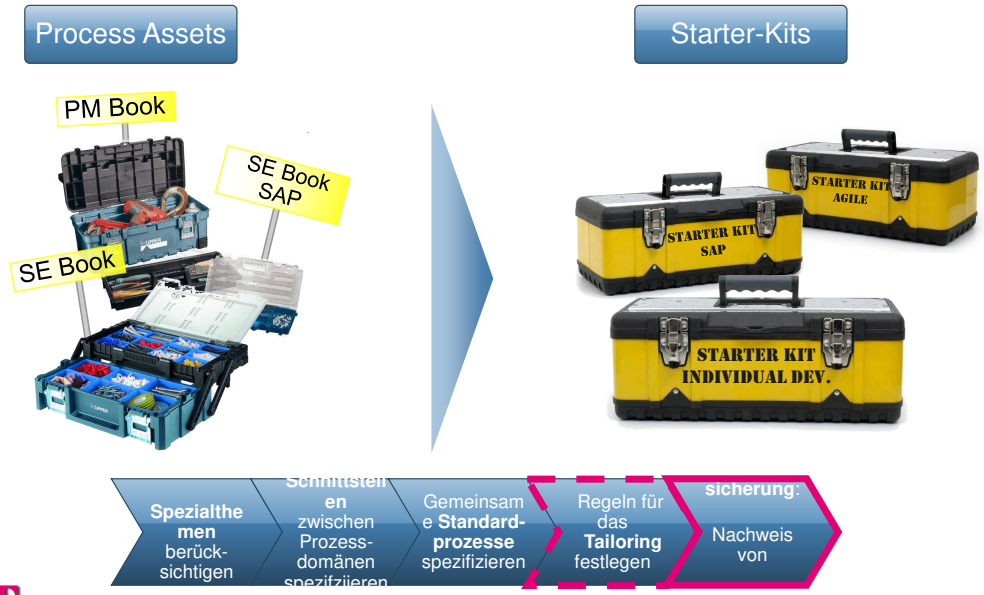
OPAL-Framework.
Generieren und Planen / Erfahrungen.



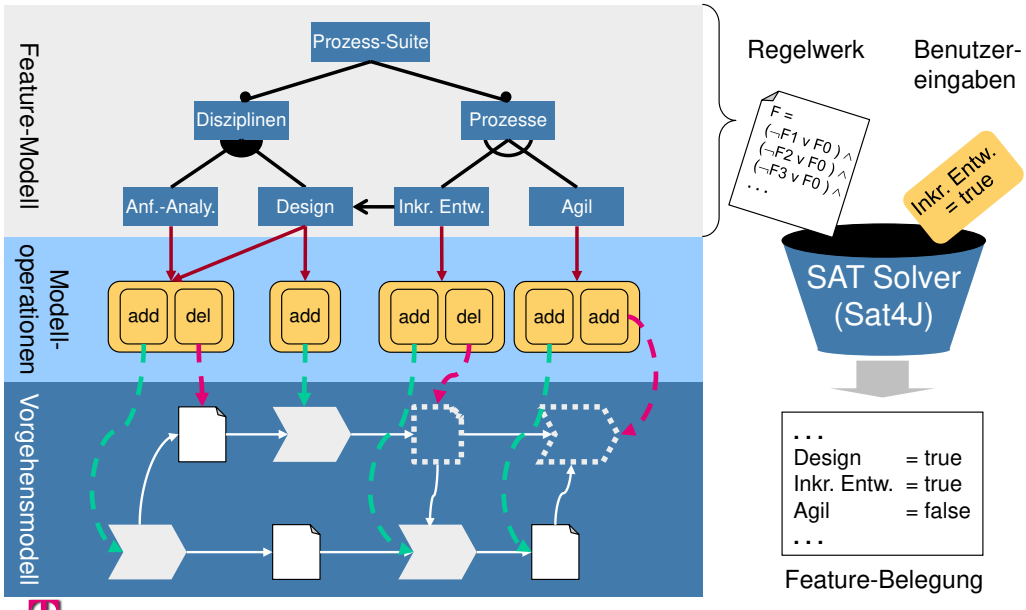
Erfahrungen mit dem Process Suite Generator

- Hohe Akzeptanz durch Arbeitsersparnis für Projekte
  - Vereinfachtes Tailoring
  - Automatisches Anlegen der Projektakte, Anpassen von Vorlagen, etc.
- Höhere Konformität zu Standards durch
  - Integration von Spezialthemen
  - Akzeptanz durch Arbeitsersparnis (Unterstützung für Planung, etc.)
- Aber: Auswirkungen von Entscheidungen müssen klar ersichtlich sein.

OPAL-Framework.
Von Process Assets zum Starter-Kit.



OPAL-Framework.
Feature-Modellierung und Tailoring (→SEE 2011).



OPAL-Framework.
Qualitätssicherung vs. kombinatorische Explosion.

- Kombinatorische Explosion am Beispiel Starter-Kit “Individuelle Entwicklung”:

158 wählbare Features → ca. $5 * 10^{32}$ getailorte Varianten des Prozessmodells

- Zahlreiche Fragestellungen:

Gibt es eine Aktivität, die nach einem Tailoring keine ausführende Rolle mehr hat?

Gibt es eine Konfiguration, bei der Literaturquellen unreferenziert bleiben?

Sind wir bei Dienstleistungsverträgen sicher niemals für die Abnahme zuständig?

…??

Problemstellung

Wie lassen sich solche Eigenschaften für alle $5 * 10^{32}$ möglichen Varianten prüfen? Durchprobieren??

➔ Bei 1ms/Variante: 16 Trilliarden Jahre = 1 Billion mal das Alter des Universums!

OPAL-Framework.
Qualitätssicherung vs. kombinatorische Explosion.

- Zahlreiche Fragestellungen:

Gibt es eine Aktivität, die nach einem Tailoring keine ausführende Rolle mehr hat?

Gibt es eine Konfiguration, bei der Literaturquellen unreferenziert bleiben?

Sind wir bei Dienstleistungsverträgen sicher niemals für die Abnahme zuständig?

…??

- Kombinatorische Explosion am Beispiel Starter-Kit “Individuelle Entwicklung”:

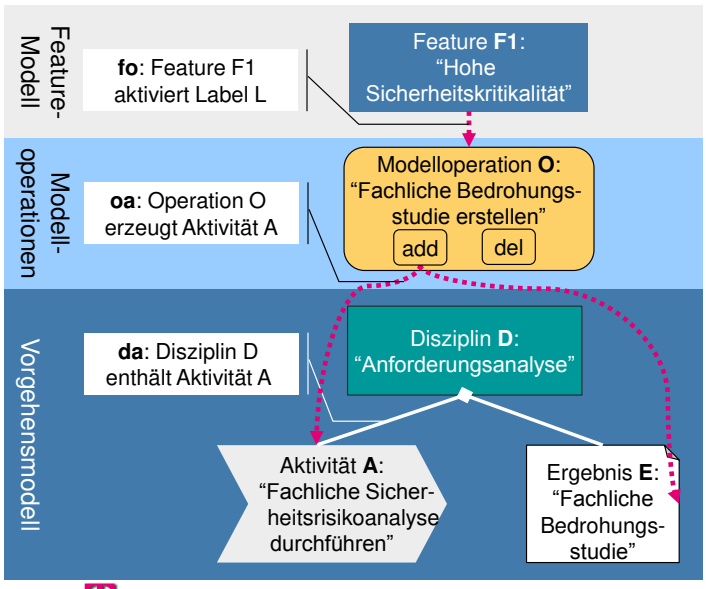
158 wählbare Features → ca. $5 * 10^{32}$ getailorte Varianten des Prozessmodells

Problemstellung

Wie lassen sich solche Eigenschaften für alle $5 * 10^{32}$ möglichen Varianten prüfen? Durchprobieren??

➔ Bei 1ms/Variante: 16 Trilliarden Jahre = 1 Billion mal das Alter des Universums!

OPAL-Framework.
Erweiterung des aussagelogischen Regelwerks am Beispiel.



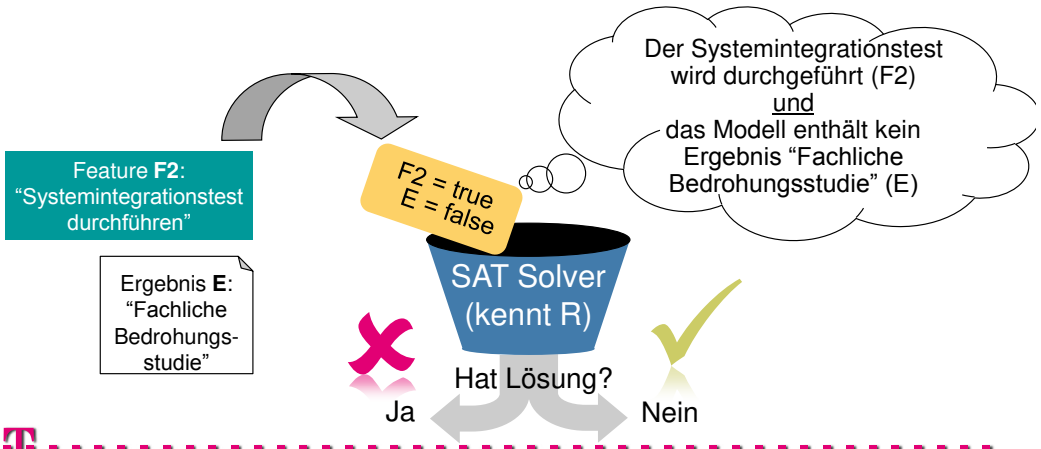
- SAT-Solver kann auch komplettes Tailoring berechnen.

➔ Zusätzliche Variablen:
- Modelloperation:true = aktiviert
- Modellelement: true = im Modell

➔ Erweitertes Regelwerk **R** (Auszug):
- fo: $F1 \Rightarrow L$
  "Wenn Feature F1 aktiv ist, dann ist auch Modelloperation O aktiv."
- da: $A \Rightarrow D$
  "Baumstruktur: Wenn Aktivität A existiert, dann ex. auch Disziplin D"
- da & oa: $(A \Rightarrow O) \wedge (O \Rightarrow A \vee \neg D)$
  "Wenn Aktivität A existiert, dann ist Operation O aktiviert
  und
  wenn Operation O aktiv ist, dann existiert Aktivität A oder Disziplin D existiert nicht"

OPAL-Framework.
Beispiel für die Prüfung einer Eigenschaft.

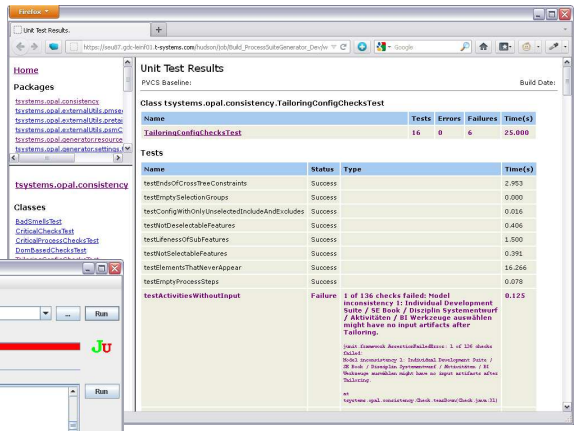
In *jedem* getailorten Prozessmodell muss gelten:
Falls ein Systemintegrationstest durchgeführt wird, dann muss im Projekt eine fachliche Bedrohungsstudie vorliegen.



OPAL-Framework.
Automatisierte Überprüfungen.

Modell-Checks als Erweiterung von JUnit:

```
public void testConstraint() {
      f2 = model.getElementByName("…");
      e  = model.getElementByName("…");
      solver.setVariable(f2, true);
      solver.setVariable(e, false);
      checkFalse( solver.hasSolution() );
      solver.clearAssumptions();
}
```



OPAL-Framework.
Erfahrungen aus Sicht eines Prozessingenieurs.

OPAL für Prozessingenieure

- Aussagenlogik zur Spezifikation des Tailorings hat sich in der Praxis bewährt
- SAT-Solver erwies sich als sehr mächtiges und nützliches Werkzeug
  - ➔ Durchführung des Tailorings
  - ➔ „Was-wäre-wenn“-Szenarien
  - ➔ Automatisierte Konsistenzchecks über verschiedene konzeptionelle Ebenen (Features, Operationen, Modellelemente)
  - ➔ …
- Aber:
  - Modellierung erfordert hohes Maß an Verständnis und Präzision
  - Umgang mit sich überschneidenden Modelloperationen oft sehr komplex

OPAL-Framework.
Zusammenfassung und Fazit.

- Zusammenfassung
  - Starter-Kits stellen Inhalte für verschieden Vorhaben zur Verfügung und orchestrieren Aktivitäten
  - Projekte können Prozessmodelle sehr flexibel anpassen
  - Ansatz ermöglicht Berücksichtigung von Spezialthemen
  - Konsistenz und Korrektheit kann durch automatisierte Verfahren effizient geprüft werden
- Ausblick
  - Erstellung weiterer Starter-Kits für unterschiedliche Projekttypen
  - Ausweitung auf zahlreiche andere Prozessdomänen
  - Integration weiterer Spezialthemen und Best-Practices
  - Plugins zur Generierung weiterer Hilfsmittel

OPAL-Framework.

Allen Projekten ein gemeinsames Vorgehensmodell, jedem Projekt seine spezifische Arbeitsweise – und wie beides zusammen kommt

Burkhard Perkens-Golomb
Munich Re
Zentralbereich Informatik
Königinstraße 107
80802 München
bperkens-golomb@munichre.com

Ian Spence
Ivar Jacobson International
18 Soho Square
London W1D 3QL
ispence@ivarjacobson.com

Abstract: Anwendungsentwicklung bedeutet mehr als kleine Teams, die neue oder verbesserte Softwareprodukte entwickeln. Anwendungsentwicklung beinhaltet Aspekte wie Wartung, Support, Hotfixes, Integration in Anwendungslandschaften, Systembetrieb, Projekte, Projekte in Programmen, Deployment und viele andere Facetten des Softwareengineerings. Wie ist bei solch einer Vielschichtigkeit ein einheitliches Projektvorgehen in einer großen und komplexen Organisation möglich? In dieser Fallstudie zeigen wir, wie Munich Re eine Methodenfamilie für die gesamte, global verteilte Anwendungsentwicklung geschaffen hat, basierend auf den beiden SEMAT-Konzepten eines „Kernels“ und der „Practices“. Die Methodik betont Aspekte der Zusammenarbeit und ermöglicht damit den Projekten den Einsatz von Praktiken aus dem Umfeld von „lean“ und „agile“.

1. Ausgangssituation

Jedes Softwareprojekt ist anders – eine Binsenweisheit. Unzählige Aspekte der verschiedenen Disziplinen des Softwareengineerings kommen in stets neuer Zusammensetzung und Gewichtung zum Tragen – dies ist auch bei den Softwareprojekten der Munich Re nicht anders. Diese Vielschichtigkeit allein stellt hohe Anforderungen an die Anpassungsfähigkeit eines jeden Vorgehensmodells.

Zusätzliche Komplexität entsteht bei Munich Re durch eine strikte weltweite Multisourcingstrategie in der Anwendungsentwicklung. Dies bedeutet, dass Arbeiten verschiedener Disziplinen wie Programmierung, Test usw. an unterschiedliche externe Provider vergeben werden; und aus ihr folgte auch eine Trennung der Mitarbeiter in der Linienorganisation gemäß den Disziplinen.

Damit Projektbeteiligte – interne Mitarbeiter, externe Dienstleister, weltweit verteilte Niederlassungen, Governancefunktionen – in einem solch komplexen Umfeld erfolgreich zusammenarbeiten, ist ein gemeinsames System von Prinzipien und Begriffen für alle unerlässlich. Und damit die Zusammenarbeit bei Projektbeginn schnell beginnt zu funktionieren, müssen darüber hinaus umfassende Blaupausen für Projektvorgehen zur Verfügung stehen. Gleichzeitig muss es Projekten möglich sein,

ihre Arbeitsweise spezifisch den Gegebenheiten anzupassen – und auch dabei soll die Methodik unterstützen. Ferner muss ein Vorgehensmodell für Munich Re Praktiken aus dem Umfeld von „lean“ und „agile“ beinhalten.

2. Vorgehen

Die Basis des Projektvorgehens von Munich Re ist in 9 Prinzipien formuliert, die grundlegende Eigenschaften der Arbeitsweise in der Anwendungsentwicklung festschreiben (u.a. iterativ, Risiko-getrieben, testorientiert). Als Systematik für die Methodendefinition dient das SEMAT-Konzept eines Kernels mit darauf aufbauenden Practices ([SEMAT], [Prac01]). Gängige Practices (Iteratives Vorgehen, Use Cases, komponentenbasierte Entwicklung und andere) wurden in die Methodik aufgenommen, einige davon spezifisch für Munich Re erweitert und angepasst.

Drei Kombinationen von Practices wurden als Blaupausen für Vorgehensweisen in Standardsituationen ausgewählt: Eine Vorgehensweise für Projekte mit hohen Risikoprofil (z.B. bei einem Projekt mit Vorstudiencharakter), für Projekte mit moderatem Risikoprofil (z.B. bei substantieller Neuentwicklung in einem fachlich und technisch gut verstandenen Umfeld) und für Projekte mit niedrigem Risikoprofil (z.B. in der Wartung). Die Betonung der Standardvorgehen liegt auf der Zusammenarbeit. Die Vorgaben gelten dabei vom Grundsatz her für Eigenentwicklung und für Einführung und Wartung von Drittprodukten gleichermaßen.

Um die Konzepte den Mitarbeitern der Organisation und den Projekten zu vermitteln, wurde ein Mix aus Trainings- und Coachingmaßnahmen aufgesetzt.

3. Ergebnisse und Bewertung

Die Methodenfamilie konnte die Anforderungen weitgehend erfüllen und befindet sich im weltweiten Rollout. Die definierten Practices greifen viele Techniken aus dem Umfeld „lean“ und „agile“ auf. Standards wie PMBoK und ISTQB wurden in die Methodik integriert, Techniken aus anderen Feldern wie z.B. User Experience werden derzeit als Practices aufgenommen, firmeneigene Standards werden als Practices eingearbeitet. Die SEMAT-Konzepte haben sich als ausreichend mächtig für diese Aufgabe gezeigt und erlauben gleichzeitig eine verständliche Darstellung der Inhalte.

Mit Blaupausen für Projektvorgehen in drei Standardsituationen kann die Arbeitsweise für einen Großteil der Projekte umfassend beschrieben werden. Anpassungen für ein Projekt – insbesondere der gewählte Grad an Formalismus – lassen sich mit dem Begriffen der Methodik beschreiben. Somit haben wir auch für die Beschreibung dieser Anpassungen eine Systematik, die von allen gleichermaßen verstanden wird.

Insgesamt haben sich für Munich Re die SEMAT-Konzepte eines Kernels und das Konzept der Practices als exzellente Basis für die Definition der Methodenfamilie erwiesen.

Literaturverzeichnis

[SEMAT] http://www.semat.org
[Prac01] Hauber, R.; Jacobson, I.; Mühlbauer, S.; Ng, P.; Spence, I.: Practices als Alternative zu vollständigen Prozessen. In: Objektspektrum, 3/2009.

2.2 Allen Projekten ein gemeinsames Vorgehensmodell, jedem Projekt seine spezifische Arbeitsweise - und wie beides zusammen kommt (27.02.2013)

A COMMON PROCEDURE MODEL, A PROJECT-INDIVIDUAL WAY-OF-WORKING – AND HOW BOTH FIT TOGETHER

IVAR JACOBSON INTERNATIONAL

Munich RE

Burkhard Perkens-Golomb, Ian Spence

Agenda

IVAR JACOBSON INTERNATIONAL

Munich RE

- A Brief Introduction to SEMAT
- Applying SEMAT @ MunichRe
- Lessons Learned / Conclusions

Introducing SEMAT

IVAR JACOBSON INTERNATIONAL

Munich RE

SEMAT

SOFTWARE ENGINEERING METHOD AND THEORY

www.semat.org

We support a process to refound software engineering based on a solid theory, proven principles and best practices.

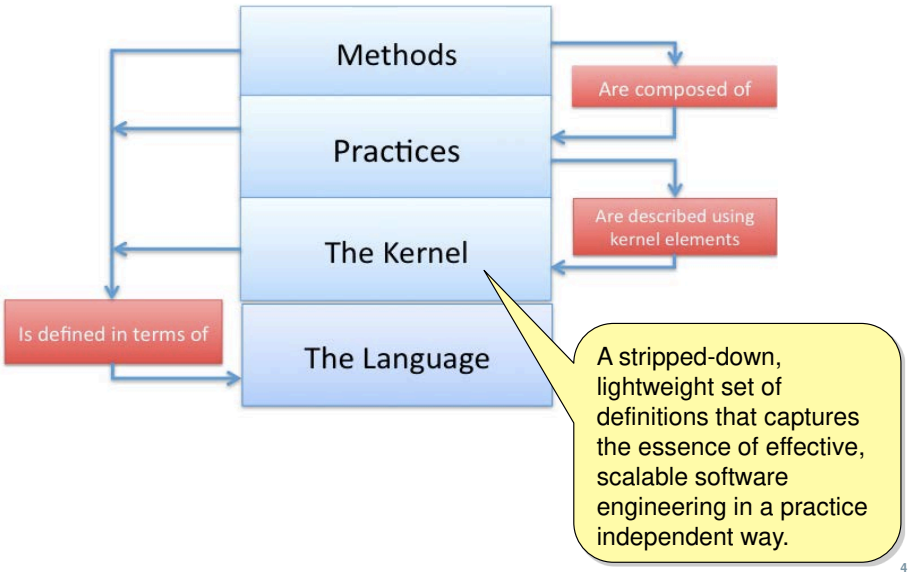
- 36 signatories
- 16 corporate signatories
- 11 academic partners
- 1000's of supporters
- Many chapters around the world

For details of the call for action, on-going work and a full list of signatories visit www.semat.org

The SEMAT Method Architecture

IVAR JACOBSON INTERNATIONAL

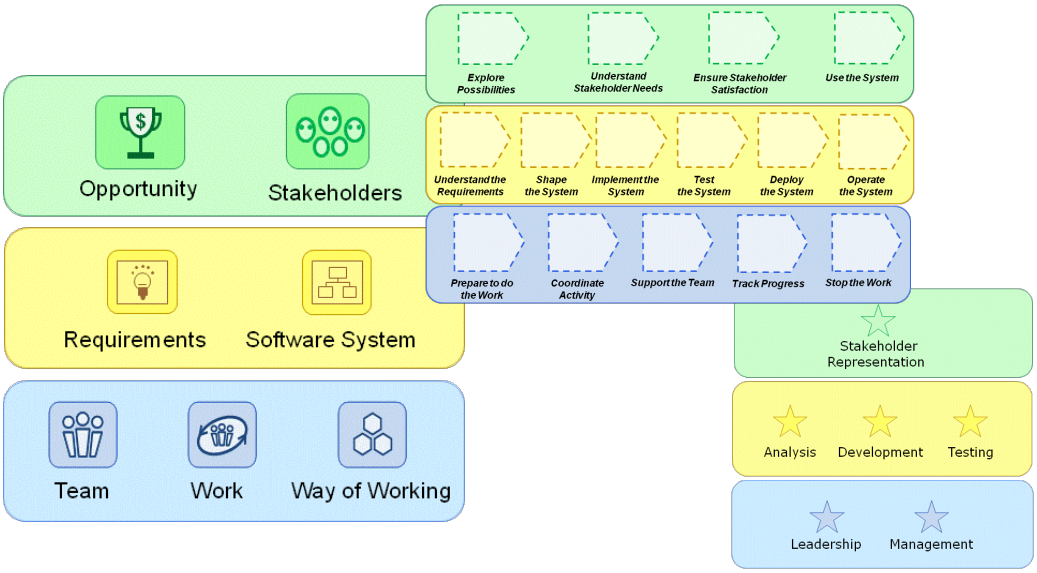
Munich RE



The kernel provides a list of things to progress …

IVAR JACOBSON INTERNATIONAL

Munich RE

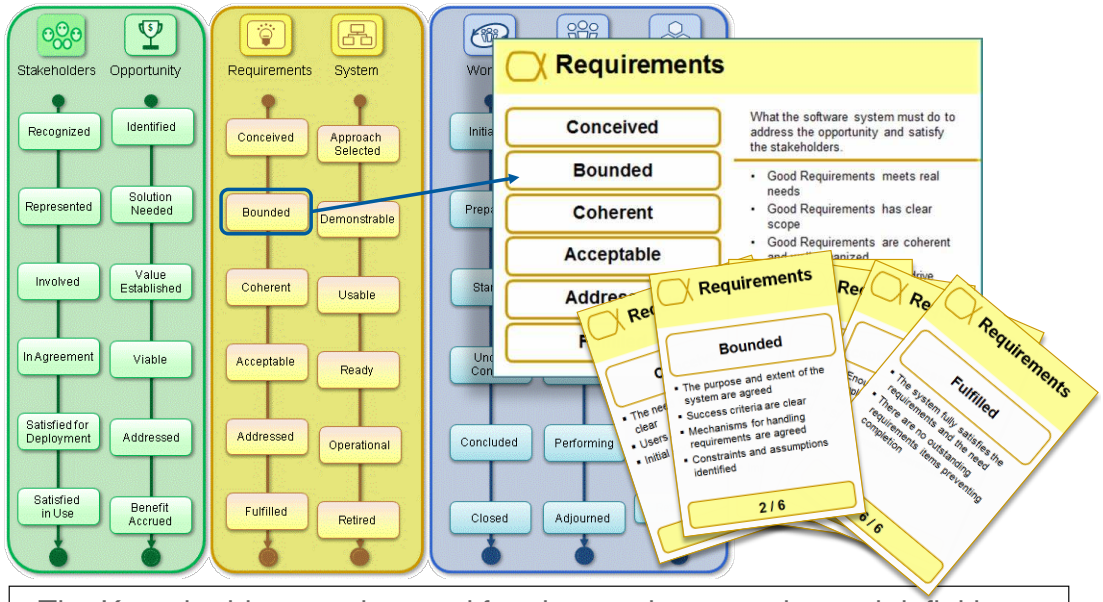


…a map of the territory, and an understanding of the competency needed

The Kernel has State to guide the team and help you understand progress, targets and project health

IVAR JACOBSON INTERNATIONAL

Munich RE

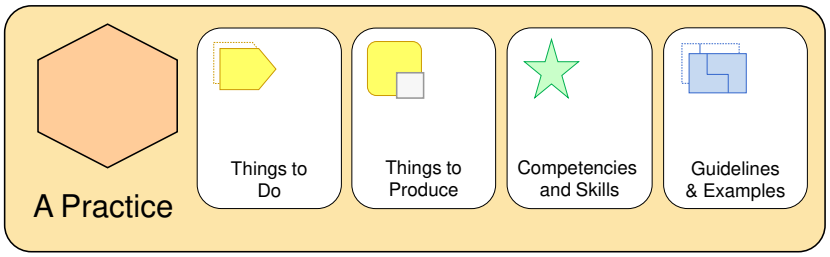


The Kernel addresses the need for clear and commonly used definitions, and enables the definition of quality gates along the project life cycle.

What is a Practice?

IVAR JACOBSON INTERNATIONAL
Munich RE

Practice: a repeatable approach to doing something with a specific purpose in mind. ..



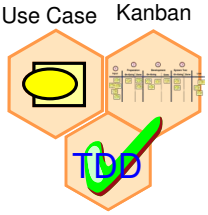
…that can be combined with other practices to create new and innovative ways-of-working.

Mix and Match Practices to Build Methods and Empower Your Teams

IVAR JACOBSON INTERNATIONAL
Munich RE

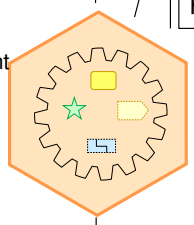
Team B

Use Case Kanban

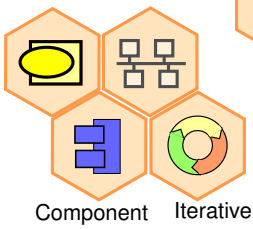


Test Driven Development

Team's build their way-of-working on top of pre-defined common ground and in-line with organizational principles and policies.

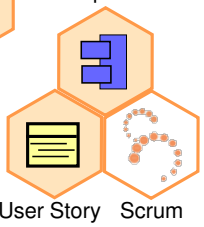


Use Case Architecture



Team A

Component



Team C

Practices enable teams to work the way that works for them

2.2 Allen Projekten ein gemeinsames Vorgehensmodell, jedem Projekt seine spezifische Arbeitsweise - und wie beides zusammen kommt (27.02.2013)

Agenda

IVAR JACOBSON INTERNATIONAL

Munich RE

- A Brief Introduction to SEMAT
- Applying SEMAT @ MunichRe
- Lessons Learned / Conclusions

The Munich Re Reinsurance Group – Present in all business markets

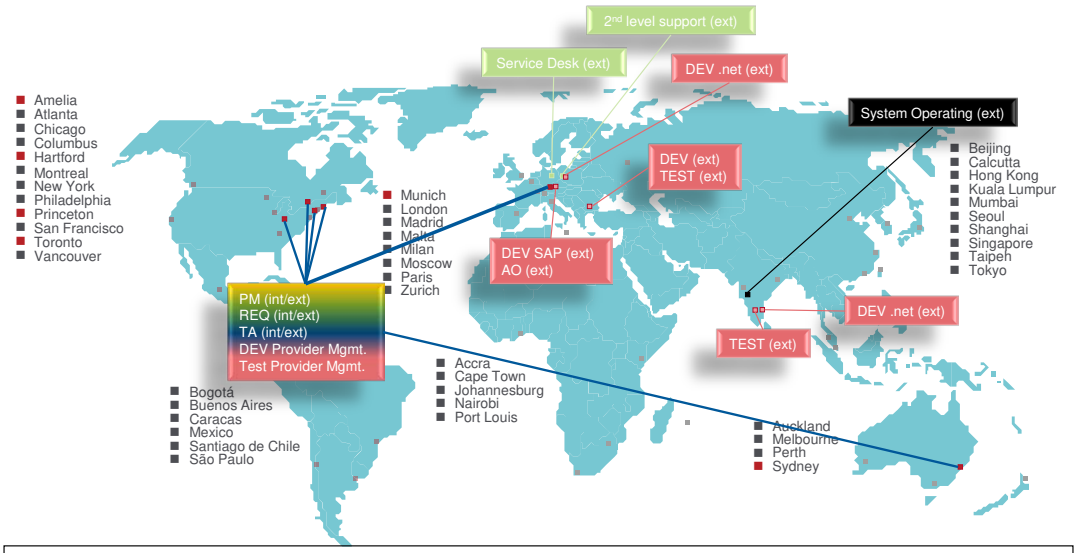
IVAR JACOBSON INTERNATIONAL

Munich RE

10

The services for Munich Re Application Development are distributed over the globe

IVAR JACOBSON INTERNATIONAL
Munich RE

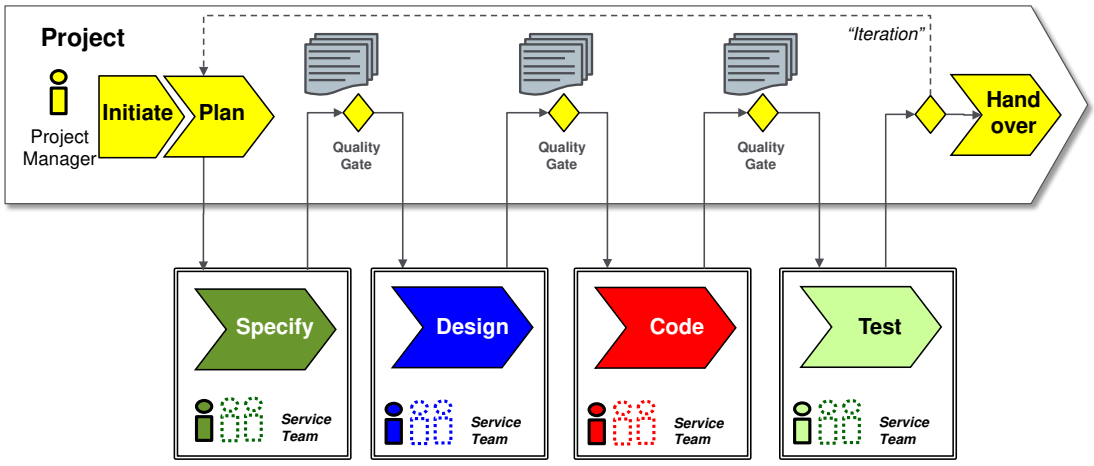


- The line organization of MR Application Development is divided into disciplines (PM, Requirements, Technical Architecture, Development, Test etc.)
- High outsourcing & offshoring rates for the services Development and Test

11

The old way of working: The discipline-oriented setup led to a strictly sequential &artefact-based approach

IVAR JACOBSON INTERNATIONAL
Munich RE



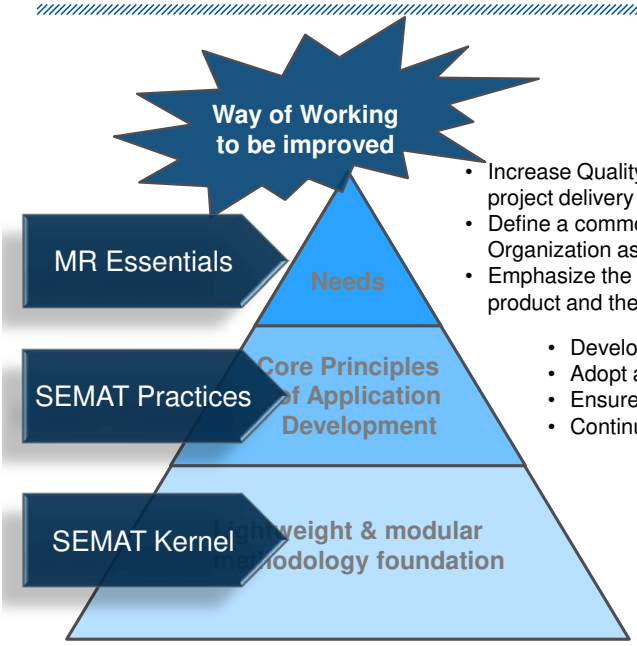
Sequential activities with formal artefact-based hand-over from one service to the next, 'orchestrated' by a Project Manager, each service with a specific way-of-working focused on their own activities.

12

The Common Way of Working: Why SEMAT ?
It supports our Principles and Requirements

IVAR JACOBSON INTERNATIONAL

Munich RE



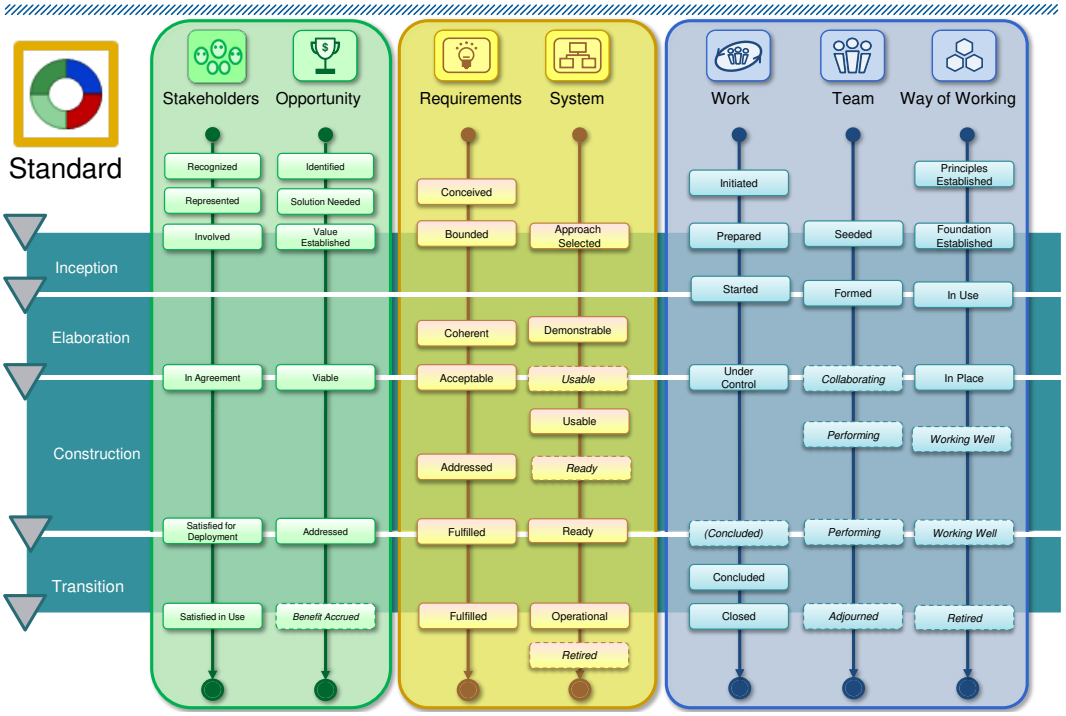
- Increase Quality, Transparency & Predictability in product and project delivery
- Define a common way of working all over the global IT Organization as base for global service provision.
- Emphasize the assumption of responsibility for the quality of a product and the success of a project by IT people
  - Develop risk-driven & reactive to change
  - Adopt agile practices
  - Ensure quality & transparency
  - Continuously improve
  - Focus description on collaboration
  - Must be adaptable & extensible for organization, projects, locations & over time
  - Supports a lean way of working
  - Based on industry standards

13

Coherent sets of states of „things to progress“ form a software development lifecycle with phases

IVAR JACOBSON INTERNATIONAL

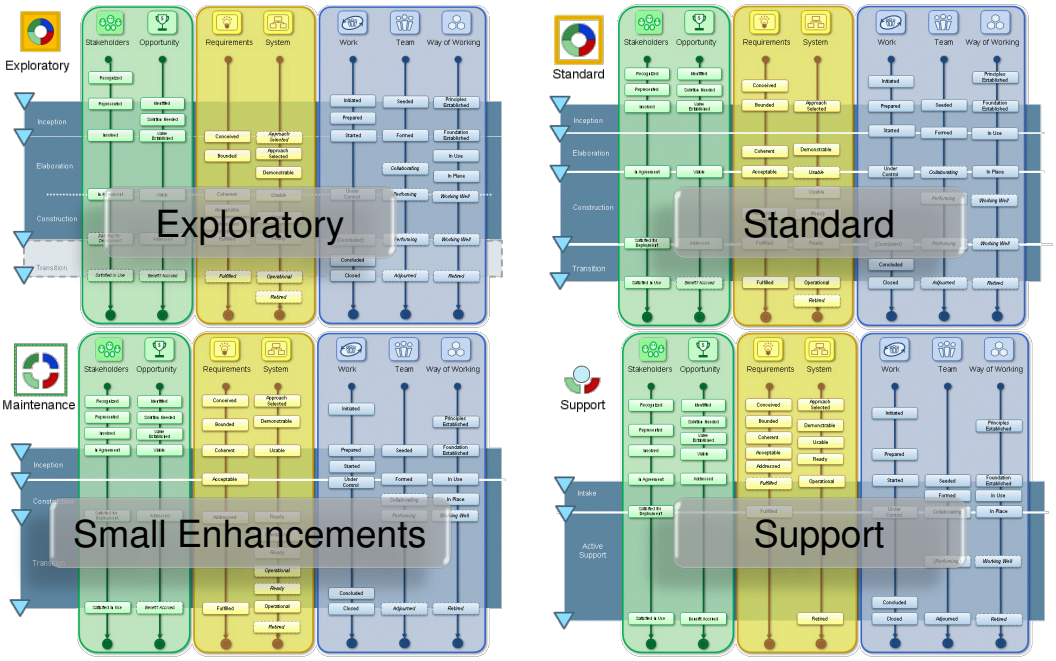
Munich RE



Depending on circumstances projects need different software development lifecycles

IVAR JACOBSON INTERNATIONAL

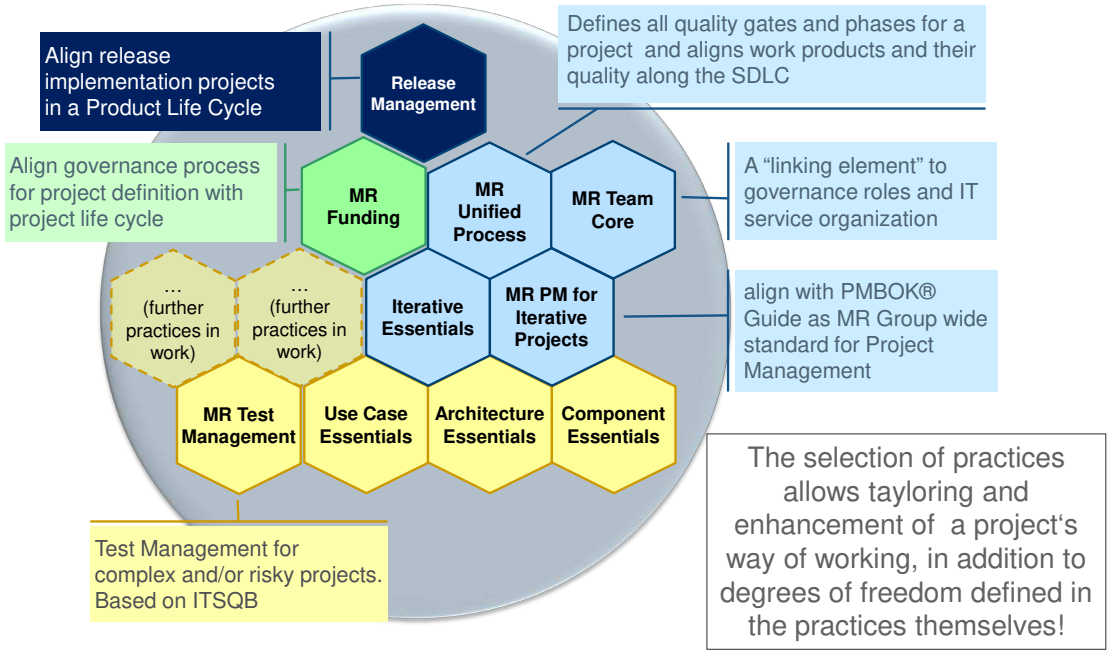
Munich RE



Practices describe ingredients to move “things to progress” forward

IVAR JACOBSON INTERNATIONAL

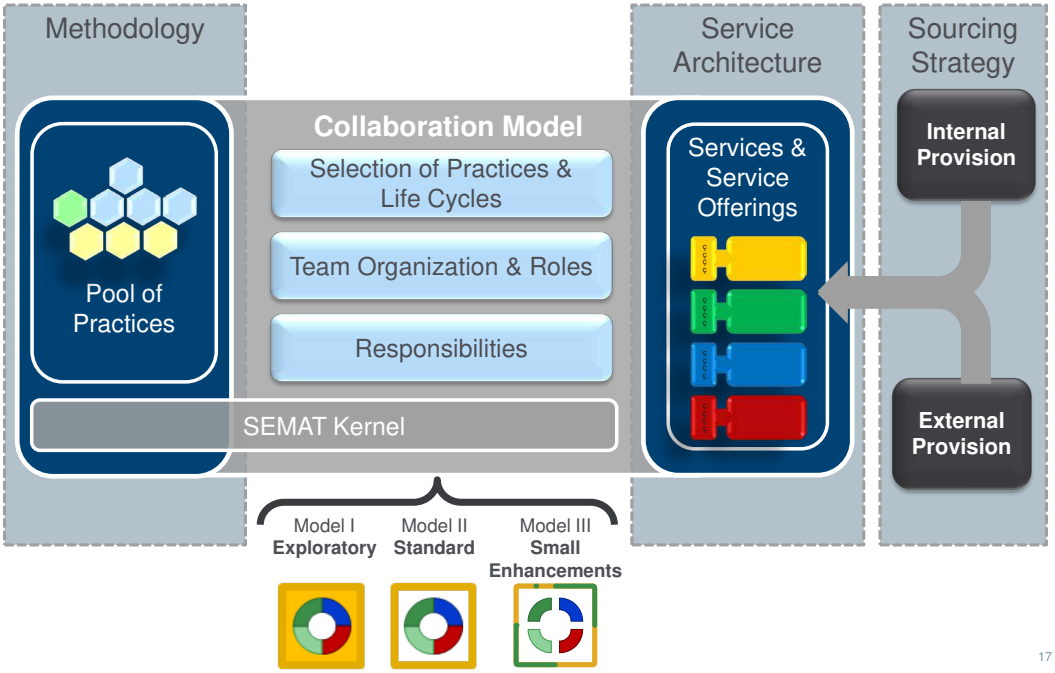
Munich RE



16

The SEMAT kernel and practices are the basis for the bridge between method and organizational architecture

IVAR JACOBSON INTERNATIONAL
Munich RE



17

Agenda

IVAR JACOBSON INTERNATIONAL
Munich RE

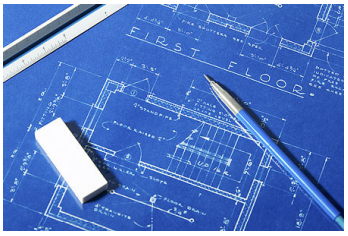
- A Brief Introduction to SEMAT
- Applying SEMAT @ MunichRe
- Lessons Learned / Conclusions

Summary: Helping teams to be more successful

IVAR JACOBSON INTERNATIONAL
Munich RE

To bring people together

To plan your moves



To build and analyze blueprints for an efficient way-of-working

To taylor your way-of-working

IVAR JACOBSON INTERNATIONAL
Munich RE

Thank You

For questions, feel free to contact us,
Ian Spence, at ispence@ivarjacobson.com, and
Burkhard Perkens-Golomb, at bperkens-golomb@munichre.com

2.2 Allen Projekten ein gemeinsames Vorgehensmodell, jedem Projekt seine spezifische Arbeitsweise - und wie beides zusammen kommt (27.02.2013)

Application Lifecycle Management für die Entwicklung mobiler Anwendungen

Andreas Entgelmeier, Hubert Biskup

Rational Software
IBM Deutschland GmbH
Karl-Arnold-Platz 1a
40474 Düsseldorf
und
Hollerithstr. 1
81829 München
entgelme@de.ibm.com
hbiskup@de.ibm.com

Abstract: In dem Beitrag werden zunächst Anwendungsbereiche mobiler Applikationen sowie deren spezifische Eigenschaften betrachtet. Ausgehend von der Frage, warum mobile Applikationen anders sind, wird deren typischer Entwicklungszyklus untersucht . Die zunehmende Bedeutung mobiler Anwendungen und deren Integration in die Unternehmensarchitektur macht auch ein Umdenken bei der Entwicklung von mobilen Applikationen notwendig. Ingenieurmäßiges Vorgehen bei Anforderungsmanagement, Entwicklung und Test ist gefordert. Damit werden viele Best Practices aus der IT-Entwicklung auch für mobile Entwicklung adaptiert.

1. Bedeutung der Entwicklung von mobilen Applikationen

Der Markt für Mobile Devices zeigt im Gegensatz zum restlichen Telekommunikationsmarkt weltweit ein starkes Wachstum. Mobile Geräte wie Smartphones und Tablets haben damit bereits heute eine wirtschaftliche Bedeutung, die weit über das hinaus geht, was unserer Wahrnehmung entspricht. Bereits heute werden fast doppelt soviel Mobile Devices verkauft wie Desktop PCs und Laptops. Es wird prognostiziert, dass es bereits im Jahr 2020 ca. 10 Mrd. Mobile Devices geben wird. [Economist 2011]

Die wirtschaftliche Bedeutung ergibt sich dabei nicht allein durch den reinen Verkauf von Hardware und mobilen Applikationen, sondern insbesondere durch ihren Einfluß auf bestehende und zukünftige Geschäftsmodelle in nahezu allen Wirtschaftsbereichen und Branchen. Es ist denkbar, dass z.B. ein Transportunternehmen oder eine Versicherung in einigen Jahren ohne einen „mobilen“ Vertriebskanal nicht mehr konkurrenzfähig ist. Dementsprechend hoch ist die Aufmerksamkeit in den Führungsetagen für die Nutzung mobiler Applikationen innerhalb der eigenen Geschäftsprozesse. In einer Umfrage gaben 74% der befragten CIOs das Thema „Mobile“ als Priorität an, was Platz 2 in der Rangfolge bedeutete (vgl. Abb. 1).

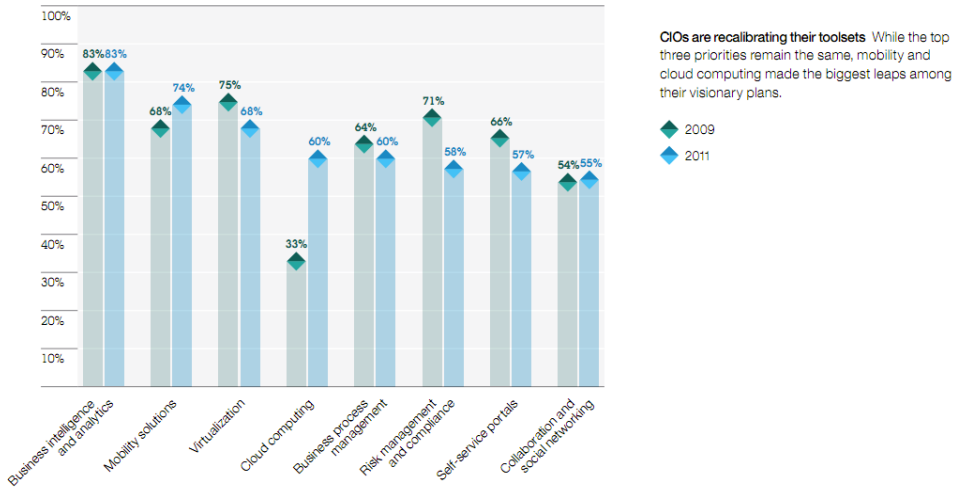


Abb. 1 aus [The Essential CIO, IBM]

Die Erwartung ist, dass die Investitionen für die Entwicklung von Mobilen Applikationen diejenige für klassische Plattformen bereits im Jahr 2015 überholt haben wird. [IBM Tech Trend Survey]

2. Anwendungsbereiche für Mobile Apps

Anforderungen an die einzusetzende Technologie und den Prozess der Softwareerstellung ergeben sich u.a. aus dem Anwendungsbereich der fertigen Applikation. Generell lassen sich hier zwei wesentliche Bereiche unterscheiden:

1. Business to Employee (B2E) / Business to Business (B2B)
   Das Ziel in diesen Bereichen ist es, Angestellte oder Geschäftspartner über mobile Geräte am Geschäftsprozeß teilnehmen zu lassen bzw. ihre Produktivität durch den ortsunabhängigen Zugriff auf die Unternehmenssysteme zu steigern. Auch eine Steigerung der Genauigkeit durch zeitnahe Datenaktualisierung z.B. innerhalb eines Materialfluss-Prozesses kann ein solches Ziel sein.

   Die mobilen Applikationen sind in diesen Szenarien typisch nicht isoliert, sondern stellen eine Komponente in einem größeren System von Applikationen dar. Auch im Entwicklungsprozess spiegelt sich dies wider, indem die Entwicklung der mobilen Applikation oft nur ein Pfad oder ein Teilprojekt eines größeren Entwicklungsprojektes ist. Sie ordnet sich damit auch den Randbedingungen und dem Zeitplan des Gesamtprojektes unter.

   Einige Beispiele für mobile Applikationen im B2E/B2B-Bereich sind
   - Chat / Collaboration
   - Kalender
   - Email
   - Profile
   - Analytics
   - Unterstützung von Geschäftsprozessen, z.B. ERP, Lagerhaltung, Materialflusssteuerung
   - Außendienstunterstützung (Abruf von Kundendaten, CRM)

2. Business to Customer (B2C)

   Bei dieser Art von Applikationen sollen Endkunden auch in mobilen Lebenssituationen erreicht werden können. Bankgeschäfte, Versicherungen, Einkäufe können jederzeit und überall getätigt werden. Über mobile Endgeräte kann der Kunde mittels mobiler Applikationen, eMail oder SMS erreicht werden. Getätigte Transaktionen werden direkt und unmittelbar in die Unternehmens-IT eingebunden.

   Beispiele für B2C-Anwendungen sind:
   - Webshops (mobile Bestellabwicklung)
   - Unterhaltung (Filme, Musik, Spiele, etc.)
   - Lokalisierungsdienste (Navigationssysteme, POIs in der Nähe)
   - Finanzdienstleistungen (Zahlungsabwicklung, Aktien- und Wertpapierhandel)
   - Kundenservice (Kundeninformationen)

3. Warum sind Mobile Applikationen anders?

Was ist nun spezifisch für mobile Applikationen und hat Einfluß auf den Entwicklungsprozeß?
Mobile Hardware und deren Benutzerinterfaces entwickeln sich sehr viel schneller weiter als in den traditionellen Software-Lebenszyklen. Die Benutzerinterfaces müssen leicht und intuitiv erfaßbar sein. Dafür ist sorgfältiges und durchdachtes Bildschirmlayout notwendig. Eingaben über eine Tastatur sollten auf das Nötigste beschränkt bleiben. Die Batteriekapazität mobiler Endgeräte ist natürlicherweise begrenzt und muß deshalb von der Software überwacht werden. Viele Geräte besitzen nur einen Touchscreen für die Bedienung.

Anwender mobiler Endgeräte haben typischerweise spezifische Anforderungen an die Geräte bzw. die Applikationen, die auf mobilen Geräten laufen. So muß z.B. der Zugriff auf Informationen effizient und zeitnah erfolgen (eine Stauwarnung bei einem Navigationsgerät macht keinen Sinn mehr, wenn der Fahrer bereits im Stau steht). Interaktionen sind kurz und gezielt, dabei sind Unterbrechungen häufig an der Tagesordnung. Interaktion mit anderen Nutzern muß zuverlässig erfolgen können (ein Telefonat mit einem Smartphone darf z.B. nicht unterbrochen werden, weil das Gerät gerade mit einem Software-Update

beschäftigt ist). Der Zeitbedarf für bestimmte Aktionen muß abschätzbar sein und deshalb dem Benutzer kommuniziert werden (time bar). Anwendungen müssen auch bei kurzfristigem Netzausfall weiterhin nutzbar sein. Datensicherheit bei Funkübertragung ist ausgesprochen wichtig, gerade wenn mobile Apps als Frontend für bestehende Backend-Prozesse fungieren.

Da der Mobile-Markt sehr dynamisch ist, hat die „Time-to-Market" einen hohen Stellenwert und macht kurze Entwicklungszyklen notwendig. Auch der Vertrieb mobiler Anwendungen unterscheidet sich stark von herkömmlichen Vertriebswegen: App Stores oder Downloads im Internet sind mittlerweile normal. Für eine weite Verbreitung ist es wichtig, mit einer Mobile-App Präsenz zu zeigen.
Einige Beispiele sind: Zeitungsverlage, Handel, Versicherungen, Reise-, Touristik- und Verkehrsbetriebe.

Die Entwicklung von mobilen Applikationen ist durch eine Diversität der Plattformen geprägt, eindeutige Trends zeichnen sich z.Zt. nicht ab. In den letzten 15 Jahren gab es z.B. nacheinander fünf unterschiedliche Dominanzen auf dem Markt: Palm, Nokia, RIM, Apple, Android. Weitere Plattformen wie z.B. Windows 8 sind in Sicht. Durch diese Diversifikation sind Applikationen nicht universell einsetzbar und für eine Portierung entsteht u.U. hoher Aufwand. Auch in Unternehmen setzt sich der Trend „Bring your own Device" durch, was für die Unternehmens-IT bedeutet, dass unterschiedlichste Plattformen unterstützt werden müssen.

Bestehende Funktionalität wird zunehmend von Web- oder Host-Applikationen auf mobile Geräte übertragen. Das bedeutet aber auch, dass sich ganz neue Anwendungsmöglichkeiten durch mobile Features auftun wie z.B.: Standortbestimmung: navigiere von meinem Standort nach ... , zeige Fahrplan von meinem Standort nach … , schließe Reiseversicherung für eine Strecke von hier nach da ab, leite mich zum nächsten freien Parkplatz; leite mich zum nächsten Carsharing-Fahrzeug; Kontaktiere Community-Member in meiner Nähe … oder z.B. Kamera: Preisvergleich durch ScanCode, Angereicherte Realität, Sachverständigen-Gutachten … oder durch Integration und Vernetzung: sende Foto in Community; ad-hoc Mitfahrgemeinschaft; Kaufe Parkticket.
Die Architekturansätze der Applikationen können dabei völlig unterschiedlich sein: native, webbasiert oder hybrid.

4. Mobile App Lebenszyklus

Ist Mobile App Entwicklung anders als „herkömmliche"? Sind die Software-Entwicklungsbereiche auf die spezielle Situation mobiler Apps vorbereitet? Welche Besonderheiten ergeben sich innerhalb des Software Lifecycles einer mobilen App?

Bisher dominiert codezentriertes Arbeiten in kleinen Teams die Entwicklung mobiler Apps. Bei der Projektsteuerung wird oft auf agile Prozesse mit kurzen Zyklen und „continuous integration" gesetzt.
Bei der Anforderungsdefinition finden leichtgewichtige Methoden wie Storyboarding und Bildschirmabläufe Anwendung. Die Umsetzung der Anforderungen in der Implementierung ist bestimmt durch herstellerabhängige SDKs und stark unterschiedliche Technologien, die oftmals spezialisierte Skills erfordern. Derartiges Know-How wird häufig von externen Partnern eingekauft. Ein Ziel ist deshalb die Lösung von der Plattformabhängigkeit.
Wegen der Diversität der Plattformen und Device-Eigenschaften erfolgt Testen häufig manuell. Auswege aus der Situation können Automatisierung und Virtualisierung bieten.

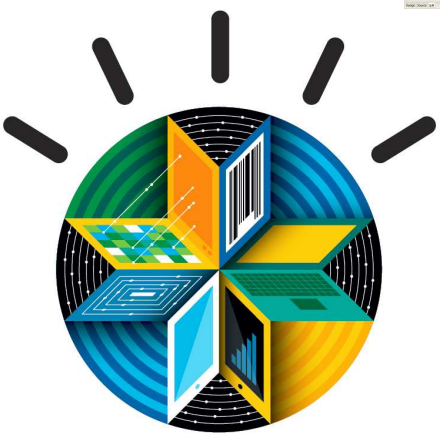
Die Erweiterung bewährter Entwicklungsplattformen wie z.B. die IBM-Plattform „Jazz" um die Besonderheiten der Entwicklung mobiler Apps bietet die Möglichkeit, bewährte Praktiken aus der traditionellen IT auf den dynamischen Bereich mobiler Apps auszuweiten.

5. Literatur

[Economist 2011]
Beyond the PC, Oct 7th 2011, by The Economist online.
http://www.economist.com/blogs/dailychart/2011/10/personal-technology (abgerufen 20.12.12.)
[The Essential CIO, IBM]
http://public.dhe.ibm.com/common/ssi/ecm/en/cie03083usen/CIE03083USEN.PDF (abgerufen 20.12.12.)
[IBM Tech Trend Survey]
http://www-03.ibm.com/press/us/en/pressrelease/32674.wss (abgerufen 20.12.12.)

Application Lifecycle Management für die Entwicklung mobiler Anwendungen

SEE 2013



Andreas Entgelmeier,
(entgelme@de.ibm.com)

Hubert Biskup
(hbiskup@de.ibm.com)

IBM Deutschland GmbH
Rational Software

© 2012 IBM Corporation

Mobile

Agenda

- Bedeutung der Entwicklung von mobilen Applikationen
- Anwendungsbereiche für Mobile Apps
- Warum sind Mobile Applikationen anders?
- Mobile App Lebenszyklus

© 2012 IBM Corporation

Mobile

IBM

Agenda

- **Bedeutung der Entwicklung von mobilen Applikationen**
- Anwendungsbereiche für Mobile Apps
- Warum sind Mobile Applikationen anders?
- Mobile App Lebenszyklus

3

© 2012 IBM Corporation

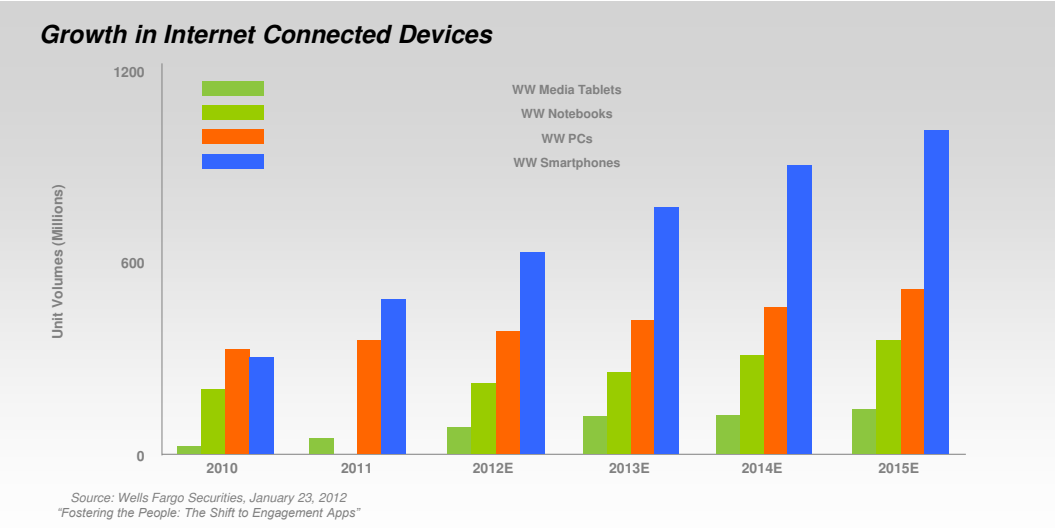
1

Mobile

IBM

Mobile is now bigger than PC

What's your mobile strategy?



10 Billion mobile connected devices by 2020

4

© 2012 IBM Corporation

Mobile

IBM

Mobile is both an enterprise mandate and challenge

45% increased productivity with mobile apps[1]

Mobile and Cloud are **top priorities** for CIOs[2]

Top Mobile Adoption Concerns:[3]

1. Security/privacy (53%)
2. Cost of developing for multiple mobile platforms (52%)
3. Integrating cloud services to mobile devices (51%)

[1] *Aberdeen Survey: ipadcto.com/2011/01/05/survey-mobile-apps-increase-enterprise-performance-and-productivity-advantages-top-three-mobile-app-strategies-gain-momentum/*
[2] *2011 IBM Global CIO Study*
[3] *2011 IBM Tech Trends Report* https://www.ibm.com/developerworks/mydeveloperworks/blogs/techtrends/entry/home?lang=en

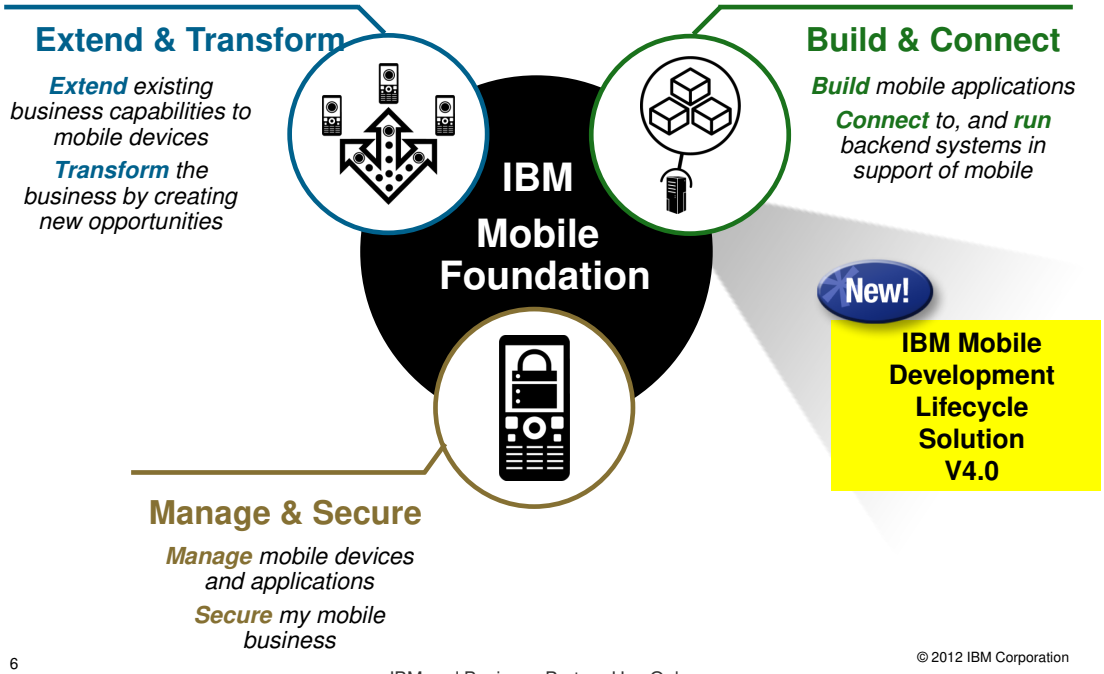
5

© 2012 IBM Corporation

Mobile

IBM

IBM's "Big Picture" on Mobile



6

© 2012 IBM Corporation
IBM and Business Partner Use Only

Mobile

Agenda

- Bedeutung der Entwicklung von mobilen Applikationen
- **Anwendungsbereiche für Mobile Apps**
- Warum sind Mobile Applikationen anders?
- Mobile App Lebenszyklus

© 2012 IBM Corporation

Mobile

Mobile is changing how people spend their time and interact

90% Of mobile users keep their device within arm's reach 100% of the time

50% By the end of this year, mobile transactions will have increased 50% over last year

20% Customers who shop on tablets tend to spend 20% more than those who shop on computers

© 2012 IBM Corporation

Mobile

IBM

Mobile is changing conventions for industries

Healthcare

From:
Traditional health institution

To:
Seamless interactions among physicians and providers, improving quality of care, patient safety and efficiency improving patient experience

Construction

From:
Construction general contractor with remote employees

To:
Empowered construction project managers armed with mobile solutions to pitch new projects in higher-growth industries

Government

From:
Sitting in traffic jams

To:
Real-time re-direction to optimal routes using mobile info

9

© 2012 IBM Corporation

Mobile

IBM

CIO's and LOB execs are recognizing the importance of mobile

CIOs[1]:

77% plan to use of personal mobile devices to access company data and applications

25+ Almost all expect to deploy more than 25 mobility applications in the next two years

[1]A report from McKinsey & Company: Based on a survey of 250 CIOs on their mobility strategies.

LOB:

With customer insight as the top CEO priority[3],

mobile changes everything.

#2 digital priority of CMOs is deploying tablet and mobile apps[2]

[2]IBM 2011 Global CMO study
[3]IBM 2012 CEO study

10

© 2012 IBM Corporation

Mobile

IBM

… linked to Business and IT Challenges

Enterprise Business Model Changes

- Neue Geschäftschancen beruhend auf Ortungsfunktionen
- Geschäftsvorgänge unabhängig von Zeit und Ort
- Bedeutung der sozialen Interaktionen im Geschäftsumfeld

App Development Lifecycle Complexity

- Komplexität durch verschiedene Plattformen mit fragmentierten Web-, native und hybrid Umgebungen
- Verbindung mit den unternehmensinternen Backend Services in einem sicheren und anpassbaren Maß
- Einheitliche mobile Anforderungen (Benutzeroberfläche, online/offline Verwendung, Upgrades, etc.)

Mobile Security and Management

- Schutz von privaten und sensiblen Informationen
- Nutzung von kundeneigenen Smartphones und Tablets
- Sichtbarkeit, Sicherheit & Management der Anforderungen mobiler Plattformen

Top Mobile Adoption Concerns:

1. Security/privacy (53%)
2. Kosten für die Entwicklung auf verschiedenen mobilen Plattformen (52%)
3. Integration von Cloud Services auf mobilen Endgeräten (51%)

Source: 2011 IBM Tech Trends Report
https://www.ibm.com/developerworks/mydeveloperworks/blogs/techtrends/entry/home?lang=en

© 2012 IBM Corporation

Mobile

IBM

Typical Use Cases for Mobile

- **Business To Consumer:**
  - Webshops (mobile Bestellabwicklung)
  - Unterhaltung (Filme, Musik, Spiele, etc.)
  - Lokalisierungsdienste (Navigationssysteme, POIs in der Nähe)
  - Finanzdienstleistungen (Zahlungsabwicklung, Aktien- und Wertpapierhandel)
  - Kundenservice (Kundeninformationen)

- **Business To Business:**
  - Sametime/Chat
  - Kalender
  - Email
  - Profile
  - Analytics
  - Unterstützung von Geschäftsprozessen
  - Außendienstunterstützung (Abruf von Kundendaten, CRM)
  - ...

© 2012 IBM Corporation

Mobile

IBM

Our clients know because *they already funded some* Mobile Initiatives …

Business to Consumer "B2C"	Business to Employee "B2E"
▪ Improve customer satisfaction ▪ Build deeper engagement and loyalty ▪ Increase sales with personalized offers ▪ Streamline customer service ▪ Differentiate from competition ▪ Improve brand perception ▪ Create deeper insight into customer buying behavior (up sell, cross sell)	▪ Increase worker productivity ▪ Improve processing times ▪ Extend existing applications to mobile workers and customers ▪ Increase employee and business partner responsiveness and decisions ▪ Resolve internal IT issues faster ▪ Reduce personnel cost utilizing personal devices

Rational Team Concert helps the Tennis GrandSlam team deliver software 10-15% ***faster*** by eliminating emails, documents, spreadsheets with collaborative work item tracking across *separate* iPhone, Android, Blackberry dev teams.

ING DIRECT
Save your money®

Rational Application Developer fueled ING Direct's claim as the first mobile banking application by leveraging *existing investment* in enterprise development with RAD/WAS.

13

© 2012 IBM Corporation
IBM and Business Partner Use Only

Mobile

IBM

Agenda

- Bedeutung der Entwicklung von mobilen Applikationen
- Anwendungsbereiche für Mobile Apps
- **Warum sind Mobile Applikationen anders?**
- Mobile App Lebenszyklus

14

© 2012 IBM Corporation

Mobile

Mobile applications are not miniature PC applications…

	Mobile Applications	PC Applications
Usage Context	▪ User may be in the middle of some other activity (e.g. shopping in a supermarket) ▪ Interactions are short and may be interrupted ▪ Often used for monitoring / data consumption ▪ Users are very impatient	▪ Using the application is the primary activity ▪ Interactions are longer and more focused ▪ Active consumption and data input ▪ Users are impatient
Mode of Interaction	▪ Non-keyboard: touch prevalent, also speech ▪ Typing should be minimized ▪ Social interactions are important ▪ Screen size/real-estate is small	▪ Keyboard and mouse ▪ Typing is okay ▪ Larger screen size for presenting information
Other considerations	▪ Integration with device capabilities (e.g. camera, GPS, accelerometer) ▪ Offline behavior ▪ Mobile hardware and user interfaces evolve much faster than the typical enterprise software cycle.	

© 2012 IBM Corporation

Mobile

Challenges in the mobile development lifecycle

Fragmentation of devices and platforms

Speed and frequent iteration of the mobile lifecycle

Online/Offline functionality

Security to protect corporate data

Connectivity to back-end systems and cloud

Mobile Context taking advantage of unique capabilities such as geo-location

© 2012 IBM Corporation

Mobile

IBM

Mobile handset dominance is difficult to predict

1995 Palm
500,000 units sold in 6mts.
70% market share

2001 Nokia
40% Market share of all mobile phones

2008 RIM
Fastest growing stock. 56% of smartphones, 77% 3yr CAGR (19.7% share in 10/11

2011 Apple
Most valuable company in the world, 27.3% of US smartphones

2013+ Android?
Has 43.7 of the US smartphone market 10/11

17

© 2012 IBM Corporation

Mobile

IBM

Biggest Challenges faced by Mobile Application Developers

Today's realities…

SD Times: What are the biggest challenges you and your organization face in terms of mobile development?	
Supporting multiple mobile platforms	46.0%
Cross-platform development	43.3%
Supporting multiple platforms from a common codebase	36.8%
Designing compelling user experiences	34.9%
Testing and quality assurance	33.3%
Developing applications as quickly as management wants	32.8%
Integrating mobile apps with enterprise data systems	27.7%
Choosing development tools	26.3%
Protecting customer data	26.3%
Supporting a wide range of hardware	25.5%
Working and syncing online/offline data	24.7%
Back-end integration with hosted services	22.3%

SD Times: "More than half of organizations are building mobile applications" http://www.sdtimes.com/link/36553

"Of all IT projects, only 47% are successful, 36% are challenged, and 17% are failures "

- Source: 2010 IT Project Success Rates, 2010 – Scott Ambler (report)

18

© 2012 IBM Corporation
IBM and Business Partner Use Only

Mobile

IBM

Agenda

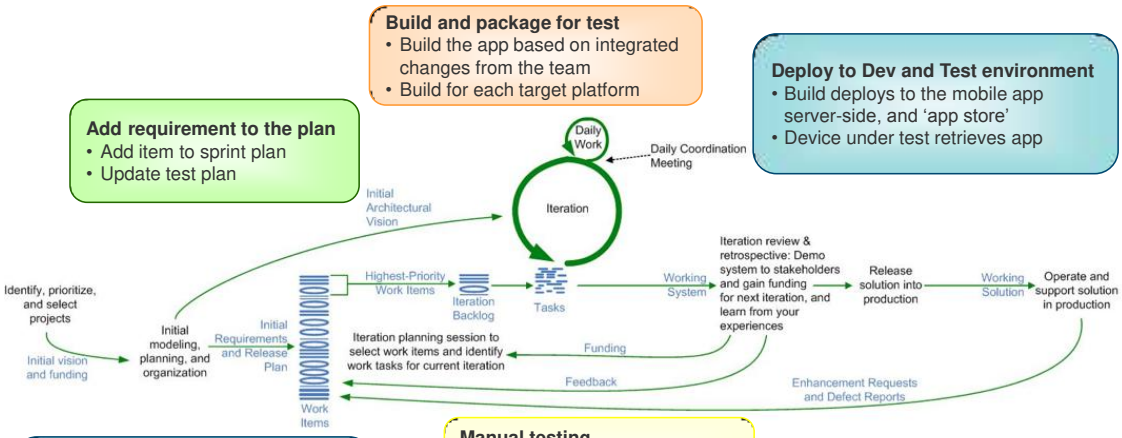
- Bedeutung der Entwicklung von mobilen Applikationen
- Anwendungsbereiche für Mobile Apps
- Warum sind Mobile Applikationen anders?
- **Mobile App Lebenszyklus**

© 2012 IBM Corporation

Mobile

IBM

Mobile application development includes more than just creating the source code

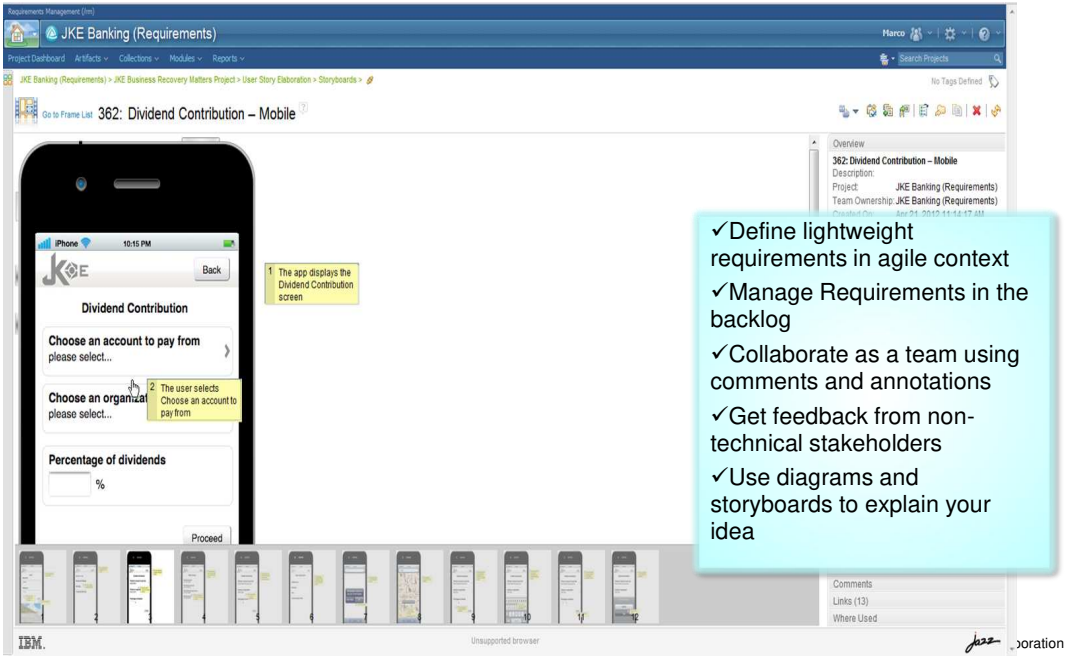


© 2012 IBM Corporation

Mobile

IBM

Requirements Definition and Management

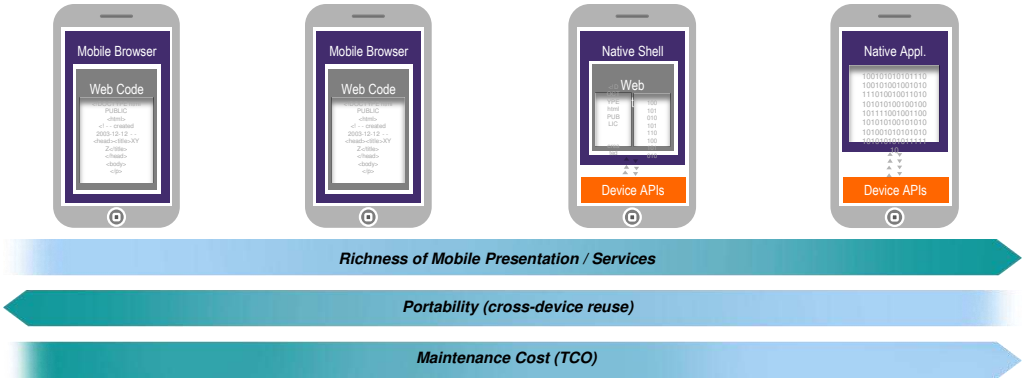


Mobile

IBM

Implementation: Mobile application development models

Web	Mobile Web	Hybrid Mobile	Native
▪ HTML, JavaScript, CSS ▪ Accessed from a mobile web browser ▪ No device-specific capabilities	▪ HTML, JavaScript, CSS ▪ Accessed from a mobile web browser; mobile-optimized UI ▪ Limited access to lower-level device capabilities	▪ HTML, JavaScript, CSS, with optional native code ▪ Installed and run like a native mobile app; mobile-optimized UI ▪ Access to lower-level device capabilities	▪ Native code ▪ Access to full set of lower-level device capabilities



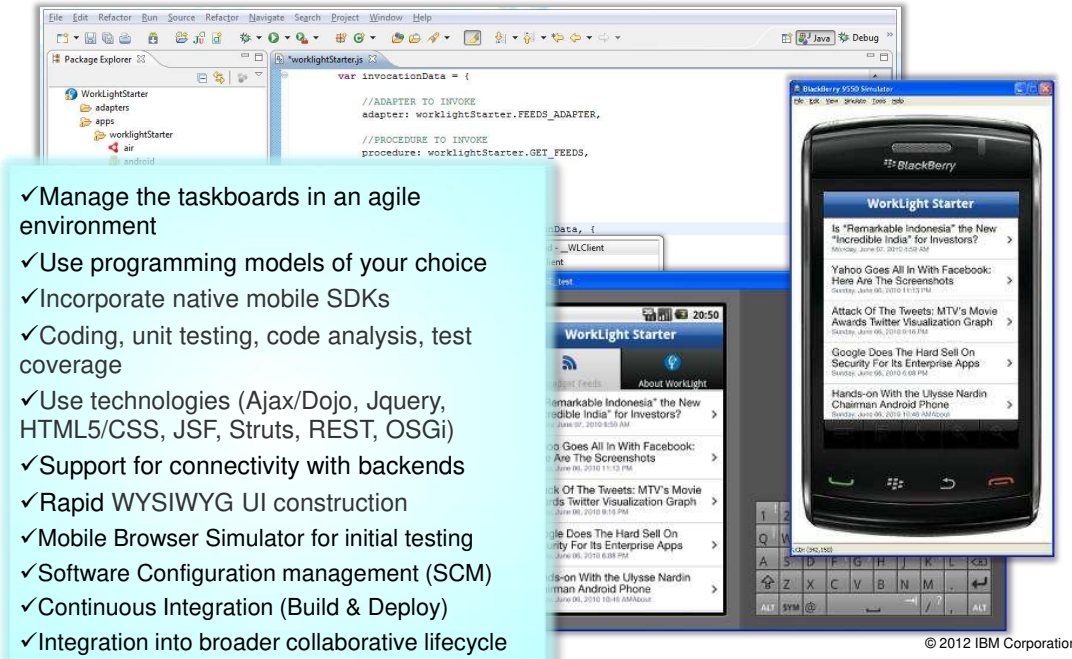
22

© 2012 IBM Corporation

Mobile

IBM

Mobile Development



- ✓Manage the taskboards in an agile environment
- ✓Use programming models of your choice
- ✓Incorporate native mobile SDKs
- ✓Coding, unit testing, code analysis, test coverage
- ✓Use technologies (Ajax/Dojo, Jquery, HTML5/CSS, JSF, Struts, REST, OSGi)
- ✓Support for connectivity with backends
- ✓Rapid WYSIWYG UI construction
- ✓Mobile Browser Simulator for initial testing
- ✓Software Configuration management (SCM)
- ✓Continuous Integration (Build & Deploy)
- ✓Integration into broader collaborative lifecycle

© 2012 IBM Corporation

Mobile

IBM

Challenge: Integrating with existing systems

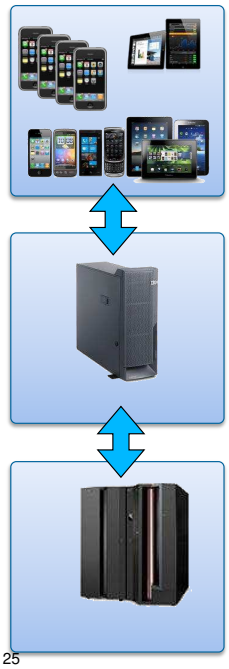
- Mobile applications need to connect to enterprise back-end data and services
  - Existing programs and services may need to be refactored to be made mobile-consumable
  - Mobile-optimized services have different characteristics than traditional web services (payload size, incremental data access, etc)
- Mobile application development lifecycle needs to bridge the multiple teams responsible for different parts of the mobile application
- Testing multi-tiered mobile applications can be slowed due to:
  - All integrations with back-end must be available to test entire app through the UI
  - Test environments are expensive, difficult and time consuming to configure
  - Difficult and time consuming to isolate defect root cause
  - Agile methodologies need fast iterations but testing delays are becoming a bottleneck

© 2012 IBM Corporation

Mobile

IBM

Test: What does a mobile tester need?



Client Tier Devices

Middle Tier Server

Back-end Data & Services

- ✓Test management
  - ✓Planning, tracking, etc.
  - ✓Manual testing capabilities
  - ✓Integration into broader collaborative lifecycle
- ✓Automation
  - ✓Device agnostic test cases
  - ✓Multi-target test cases
  - ✓Interaction with device-specific capabilities
  - ✓Virtualization of middle-tier and back-end systems
- ✓Access to a diverse set of devices
  - ✓Platforms
  - ✓Manufacturers
  - ✓Form factors
  - ✓Carriers

25

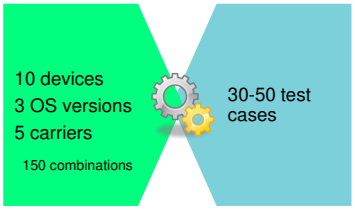
© 2012 IBM Corporation

Mobile

IBM

Quality management techniques and cloud-based solutions can help ensure adequate test coverage in this fragmented environment

- **Test coverage optimization**
  - Testing against all possible combinations of devices, OS versions, carriers, etc requires large combinatorial number of tests
  - Combinatorial optimization can dramatically cut down on number of test cases that need to be run while providing coverage for interaction between environment factors



- **"Mobile device cloud" services**
  - Sometimes it is necessary to test on specific devices/carriers
  - Maintaining a huge library of device across the world is impossible/extremely costly
  - "device-cloud" services provide virtual access to physical devices

26

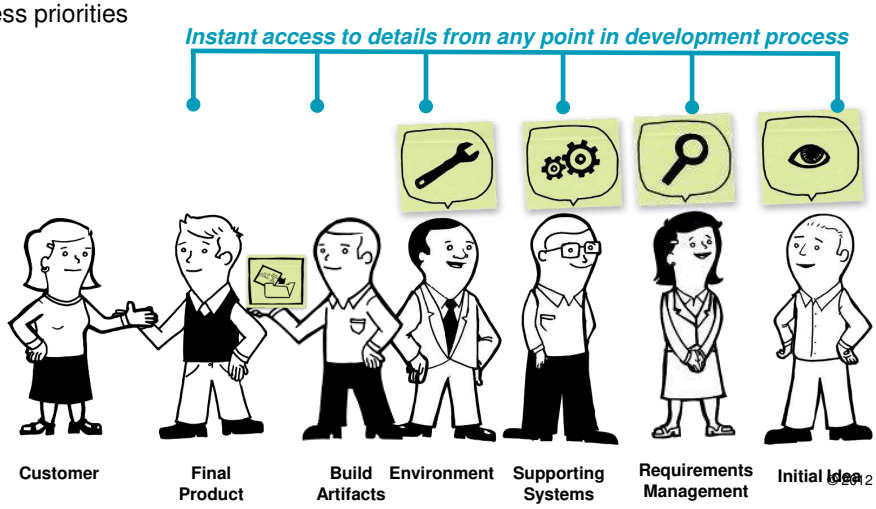
© 2012 IBM Corporation

Mobile

IBM

Lifecycle Traceability improves quality

- Establish **relationships between software artifacts**
- **Identify and close artifact gaps,** ensuring **coverage** across disciplines
- Provide visibility into the **completeness** of planned items by inspecting all related artifacts
- Provide easy **access to related artifacts** ensuring everyone shares the same view
- Deliver transparency which enables everyone to make **fully informed decisions** based business priorities



© 2012 IBM Corporation

Mobile

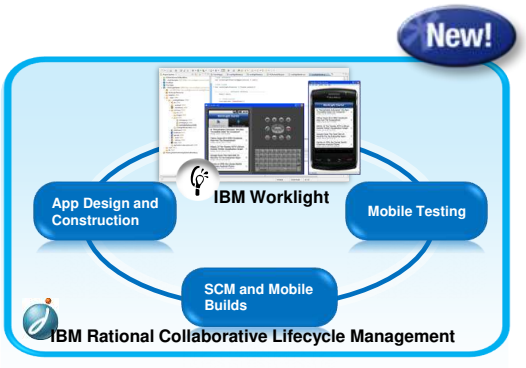
IBM

IBM Mobile Development Lifecycle Solution V4.0

Enact a collaborative, multi-platform mobile development lifecycle

A comprehensive mobile development solution combining:

- Enterprise-grade, standard-based mobile application platform, supporting native, hybrid and mobile web programming models
- Best-of-class collaborative mobile application development lifecycle capabilities
- Distributed team build and test integrations with the IBM Mobile Platform
- “Mobile device-cloud” service integration for on-device testing
- Support for Agile methodologies for dealing with fast-paced development for mobile
- Mobile development best practice



© 2012 IBM Corporation

Mobile

http://www.ibm.com/software/rational/mobile/

29

© 2012 IBM Corporation

Mobile

© Copyright IBM Corporation 2012. All rights reserved. The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. IBM, the IBM logo, Rational, the Rational logo, Telelogic, the Telelogic logo, and other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.

30

© 2012 IBM Corporation

© 2012 IBM Corporation

3. Testen

Sessionüberblick

3.1. Toolwahl zu automatisierten Akzeptanztests - Was sollte ich vorher wissen? (27.02.2013) 56
3.2. Modultests in der Softwareentwicklung für mobile Geräte durch den Wissensträger mittels KBUnit (28.02.2013) 59

Toolwahl zu automatisierten Akzeptanztests

Was sollte ich vorher wissen?

Meike Mertsch, Markus Gärtner

it-agile GmbH
Paul-Stritter-Weg 5
22297 Hamburg
meike.mertsch@it-agile.de
markus.gaertner@it-agile.de

Abstract: In dieser interaktiven Session geht es um die vielen Fallstricke, die sich Teams im Bezug auf automatisierte Akzeptanztests selbst auferlegen. Basierend auf den Erkenntnissen aus ATDD by Example [Gä12], stellen die Präsentatoren einen alternativen Ansatz zur Toolfindung für das richtige Tool zur funktionalen Testautomatisierung vor, und geben Tipps und Hinweise für langfristig wartbare Akzeptanztests.

1. Testautomatisierung, aber wie?

Die häufigste Frage, wenn ein Team mit Testautomatisierung beginnt, fängt mit der Frage nach dem richtigen Tool an. Dabei ist diese Frage zwar die erste, die intuitiv erscheint, aber nicht unbedingt die dringlichste Frage, die beantwortet werden sollte. Bei Testautomatisierung kommt es vor allem darauf an, dass die Tests nicht nur einfach zu erstellen sind, sondern langfristig auch einfach gewartet werden können, und dafür den Änderungen am Produktivcode gleichermaßen Stand halten können. Darüber hinaus passt nicht jedes Framework zu jeder Teamkonstellation. Hierbei spielen verschiedene Fragen eine Rolle, wie z.B.:

- Wer soll die Tests initial entwickeln? Tester, Programmierer, Kunde, eine Kombination aus allen dreien?
- Wer soll sie langfristig warten? Tester, Programmierer, das ganze Team?
- Wer soll zukünftig damit arbeiten? Tester, Programmierer, Product Owner, Kunde?
- Wie häufig sollen die Tests ausgeführt werden? Einmal täglich, bei jedem Check-In, alle zwei Wochen?
- Wer muss die Tests lesen können? Kunde, Entwickler, Tester, Projektleiter?

2. Alternative Herangehensweisen

Statt mit einer konkreten Toolfrage zu beginnen, sehen wir bei Kunden immer wieder, dass erfolgreiche Testautomatisierung vor allem damit zu tun hat, dass diese wartbar ist, und den Teams langfristig die sog. *Costs of Testownership* nicht im Weg stehen. Für erfolgreiche Testautomatisierung gelten dabei die gleichen Prinzipien wie bei erfolgreicher Softwareentwicklung: Sie muss modular anpassbar sein, ihre Intention dem Leser klar kommunizieren und redundanzfrei sein. Doch wie schafft man das?

2.1 Das richtige Format

Hierbei kommt es vor allem auf das richtige Format an. Derzeit auf dem Markt erhältliche Tools unterstützen dabei eine Vielzahl von Formaten, die sich in der Praxis bewährt haben. Darunter fallen sog. Tabellenbasierte Formate wie sie bei z.B. dem Framework for Integrated Tests [FIT], FitNesse [FitNesse] und Concordion [Concordion] zum Einsatz kommen. Darunter fallen aber auch Ansätze wie sie beim Behavior-Driven Development in Form von Given/When/Then eingesetzt werden. Dazu zählen Tools wie Cucumber [Cuke], JBehave [JBehave] und JNario [JNario].

3. Fazit

In dieser interaktiven Session stellen wir die unterschiedlichen Formate vor. Anschließend erhalten die Teilnehmer die Möglichkeit, mit verschiedenen Formaten an einem eigenen Beispiel zu arbeiten, und Vor- und Nachteile der jeweiligen Formate gegenseitig zu bewerten. Die Teilnehmer werden anschließend mit einem geschärften Verständnis von automatisierten Akzeptanztests und der wesentlichen Frage nach dem richtigen Format, ihre weitere Reise hin zur Testautomatisierung vorantreiben können. Dabei setzen die Präsentatoren auch auf die Vermittlung von Tipps und Tricks zur langfristigen Wartbarkeit von Testautomatisierung.

Literaturverzeichnis

[Gä12] Gärtner, M.: ATDD by Example – A Practical Guide to Acceptance Test-Driven Development. Addison-Wesley Longman, Amsterdam, 2012.

[FIT] Mugridge, R.; Cunningham, W.: FIT for Developing Software – Framework for Integrated Tests. Addison-Wesley Longman, Amsterdam, 2005.

[FitNesse] Martin, R.; Martin M.et.al.: FitNessen: http://www.fitnesse.org

[Concordion] Peterson, D.: Concordion: http://www.concordion.org

[Cuke] N.N.: Cucumber: http://cukes.info

[JBehave] N.N.: JBehave: http://jbehave.org

[JNario] Benz, S.: JNario: http://jnario.org

Modultests in der Softwareentwicklung für mobile Geräte durch den Wissensträger mittels KBUnit

Ursula Oesing, Roland Krüger

Fachbereich Elektrotechnik und Informatik
Hochschule Bochum
Lennershofstraße 140
44801 Bochum
ursula.oesing@hs-bochum.de

secunet Security Networks AG
Kronprinzenstraße 30
45128 Essen
roland.krueger@secunet.com

Abstract: In der Softwareentwicklung haben sich JUnit Tests als ein wichtiges Testframework zur Durchführung von Modultests etabliert. Für die Entwicklung von Software für mobile Geräte werden ebenfalls JUnit Tests verwendet und sind in der Entwicklungsumgebung integriert. Der Erfolg eines Projekts ist weiterhin abhängig von einer möglichst frühen Einbindung des Wissensträgers der geschäftsspezifischen Abläufe in den Qualitätssicherungsprozess [SP12]. Um die Situation entscheidend zu verbessern, soll dieser an JUnit Tests beteiligt werden.

Bei dem Wissensträger können aber Programmierkenntnisse oder gar Kenntnisse zur Entwicklungsumgebung von mobilen Geräten, die für die Durchführung von JUnit Tests erforderlich sind, nicht erwartet werden. Daher wurde die Java-Anwendung KBUnit *(Knowledge Based Unit Testing Application)* entwickelt. Mittels KBUnit können Wissensträger ohne die genannten Kenntnisse auf die von einem Entwickler definierten JUnit Testfälle der Anwendung für mobile Geräte zugreifen und diese geschäftsspezifisch variieren. KBUnit bietet darüber hinaus eine automatische Testfallgenerierung, durch die nicht nur getestet wird, was funktionieren soll, sondern auch das, was nicht funktionieren darf. Letzteres wird häufig vergessen, ist aber aus IT-Security-Sicht von besonderem Interesse, da Sicherheitslücken häufig durch undefinierte Zustände der Anwendung entstehen.

1. Ausgangssituation und Zielstellung

Die Funktionalität einer Anwendung für mobile Geräte soll in einer möglichst frühen Projektphase von Wissensträgern getestet werden können. Beispielsweise möchte ein Finanzdienstleister eine App entwickeln, welche Berechnungen zu seinen Finanzprodukten erlaubt. Eine Teilfunktion sei die Berechnung der Gesamtschuld, wobei der Anwender die Höhe des Darlehens, den Zinssatz und die Laufzeit vorgibt. Entwickler können JUnit Tests hierzu implementieren, aber sie kennen die Fachlichkeit nicht genügend, hier beispielsweise den maximal zugelassenen Zinssatz. Sie wissen weiterhin nicht, wie die Teilfunktion reagieren soll, wenn ein unzulässiger Zinssatz eingegeben wird. Ausrei-

chende Fachkenntnisse hat der Wissensträger, der aber wegen fehlender Programmierkenntnisse und fehlender Kenntnisse zur Entwicklungsumgebung nicht die JUnit Tests implementieren kann. Ziel ist, dem Wissensträger die Erstellung von JUnit Tests zu Anwendungen für mobile Geräte zu ermöglichen. Diese sollen unabhängig von der Oberfläche der App ohne Emulator ausgeführt werden können. Weiterhin sollen geeignete Testfälle automatisiert generiert und durchgeführt werden können, auch zu Fällen, die nicht funktionieren dürfen. Dann kann der Wissensträger festlegen, wie die App in so einem Fall reagieren soll, ein undefinierter Zustand wird vermieden.

2 Lösungsansatz und Funktionsweise von KBUnit

Die Anwendung jCUT, die auf der Software Engineering 2012 vorgestellt wurde [SP12], bietet die Basisfunktionalitäten, ist aber nicht für die Entwicklungsumgebung für mobile Geräte tauglich. Die Weiterentwicklung KBUnit, eine Java-Anwendung mit Swing-Oberfläche, kann auf die JUnit Tests von Android-Applikationen, die unter Verwendung der Eclipse IDE und dem Android SDK entwickelt werden, zugreifen. Tests können ohne das Starten des Emulators abgespielt werden. Die Testklassen und die zu testenden Klassen der Android-Applikation werden zusammen mit einer xml-Datei, die weitere Testinformationen enthält, zu einem jar-File gepackt. Dieses wird KBUnit zur Verfügung gestellt.

Mit KBUnit ist zudem die Testfallgenerierung substantiell um das Testen von Fällen, die nicht funktionieren dürfen, erweitert worden. Der Wissensträger gibt die Definitionsmengen der einzelnen Parameter einer zu testenden Methode in Form von Intervallen in die Anwendung ein. Um die Fälle zu testen, die funktionieren sollen, werden wie bisher Repräsentanten von Äquivalenzklassen zu einem Parameter mit Repräsentanten der Äquivalenzklassen der anderen Parameter kombiniert, die in der Definitionsmenge liegen. Um Grenzfälle und Fälle zu testen, die nicht funktionieren dürfen, werden Grenzwerte und Repräsentanten außerhalb der Definitionsmenge eines Parameters mit Werten innerhalb der Definitionsmengen der weiteren Parameter zu einem Testfall kombiniert. Grenzwerte und Werte außerhalb der Definitionsmenge sind als solche gekennzeichnet.

3 Bewertung und Ausblick

Die JUnit-Testfälle einer Android-Anwendung können mittels KBUnit ohne Emulator ausgeführt und variiert werden. Das führt zu Performancevorteilen, man kann aber die Oberfläche der App nicht testen, was bei Modultests auch nicht beabsichtigt ist. Geplant sind weitere Schritte zur Optimierung der Testfallgenerierung.

Literatur

[SP12] Sprengholz, P.; Oesing, U.: Durchführung von Modultests durch den Auftraggeber in Softwareentwicklungsprojekten mittels jCUT in (Jähnichen, S.; Küpper, A: Albayrak, S. Hrsg.): Software Engineering 2012 der Gesellschaft für Informatik e.V., Berlin 2012; S. 39-40.

Modultests in der Softwareentwicklung für mobile Geräte durch den Wissensträger mittels KBUnit

Ursula Oesing, Roland Krüger

Fachbereich Elektrotechnik und Informatik
Hochschule Bochum
Lennershofstraße 140
44801 Bochum
ursula.oesing@hs-bochum.de

secunet Security Networks AG
Kronprinzenstraße 30
45128 Essen
roland.krueger@secunet.com

AGENDA

Hochschule Bochum
Bochum University
of Applied Sciences

KBUnit :: ***Knowledge Based Unit Testing Application***

1 Motivation

2 Vorstellung von KBUnit

3 Live-Demo

4 Ausblick

KBUnit ist die Weiterentwicklung eines studentischen Forschungsprojekts zum Thema *Lean Software Development*:
Sprengholz, P.; Oesing, U.: Durchführung von Modultests durch den Auftraggeber in Softwareentwicklungsprojekten mittels jCUT in (Jähnichen, S.; Küpper, A: Albayrak, S. Hrsg.): Software Engineering 2012 der GI e.V., Berlin 2012.

1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Allgemein anerkannte Aussagen und ihre Folgen

1 Je früher Fehler in der Softwareentwicklung aufgedeckt werden, umso höher ist die Wahrscheinlichkeit, dass ein Softwareprojekt für alle Beteiligten erfolgreich verläuft.

2 Der Wissensträger kennt die Fachlichkeit zu seinen Anforderungen am besten und kann fachliche Fehler häufig schneller entdecken als ein Entwickler.

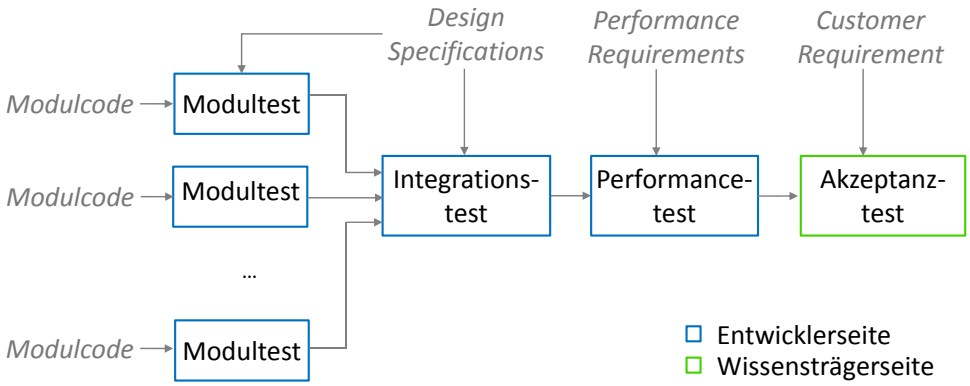
3 Beide Aussagen treffen auch für die Entwicklung von Software für mobile Geräte zu.

Wissensträger sollte so früh wie möglich in den Testprozess integriert werden. Dies gilt auch bei Software für mobile Geräte.

1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Testprozess :: Traditionell

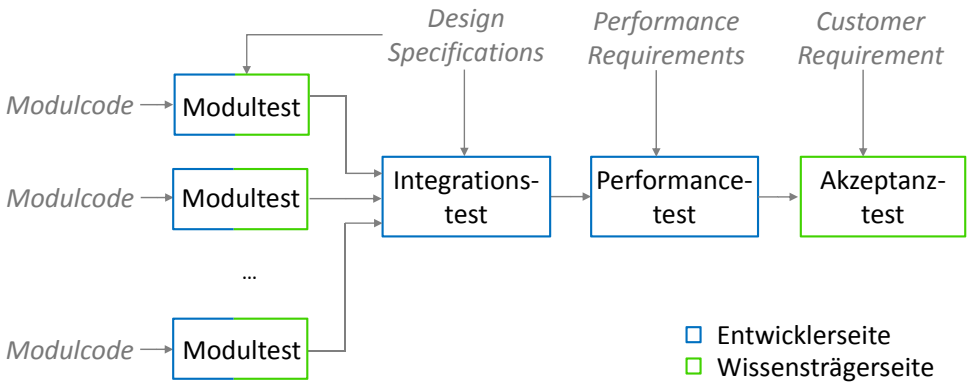


Wissensträger überprüft Funktionsfähigkeit eines (Teil-) Programms bei Durchführung von Akzeptanztests am Ende der (Teil-) Entwicklung

1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Testprozess :: Wunsch



Wissensträger soll Modultests formulieren und ausführen können, so dass Fehler früher erkannt werden.

1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel

1 Betrachtet werden Funktionalitäten von Modulen einer mobilen Anwendung, die mit dem Android SDK entwickelt werden.

2 Betrachtet werden Modultests, die mittels Android-JUnit-Tests durchgeführt werden.

Die Oberfläche der mobilen Anwendung wird hier nicht berücksichtigt!

1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: User Story

- Eine mobile Anwendung zur Unterstützung von Finanzierungs-entscheidungen soll entwickelt werden.
- User Story

> *Als potentieller Kunde eines Kreditinstituts möchte ich die Gesamt-belastung zu einem Tilgungsdarlehen berechnen, damit ich die verschiedenen Konditionen der Kreditinstitute vergleichen kann.*

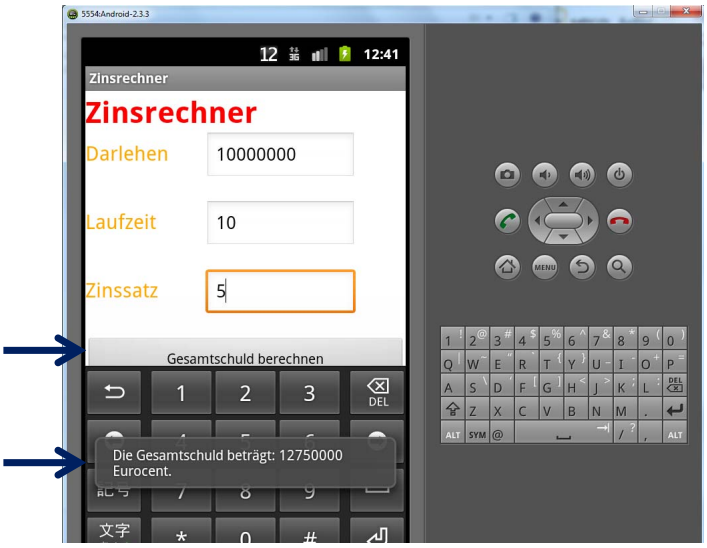
- Beispiel zur User Story

> *Bei einem Tilgungsdarlehen in Höhe von 100000,00 Euro, dass über 10 Perioden zu 5 % verzinst wird, ergibt sich eine Gesamtschuld von 127500,00 Euro.*

1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: Android App



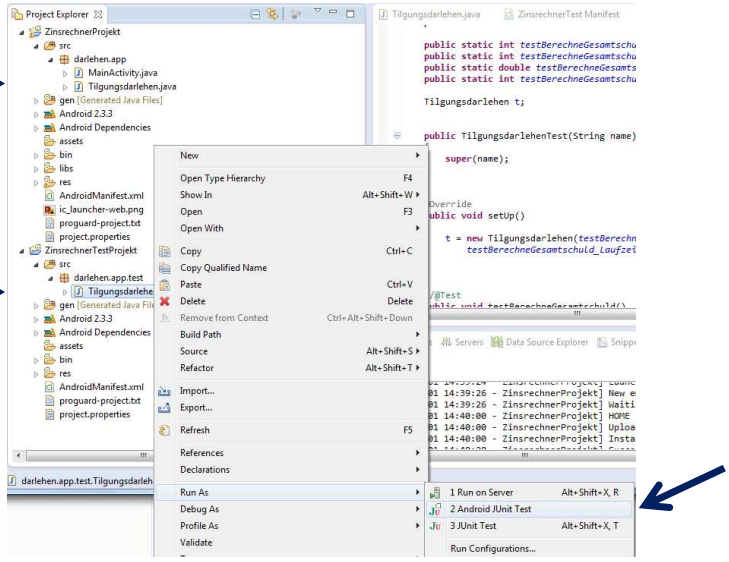
1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: Android JUnit Test zur Android App

Sourcecode

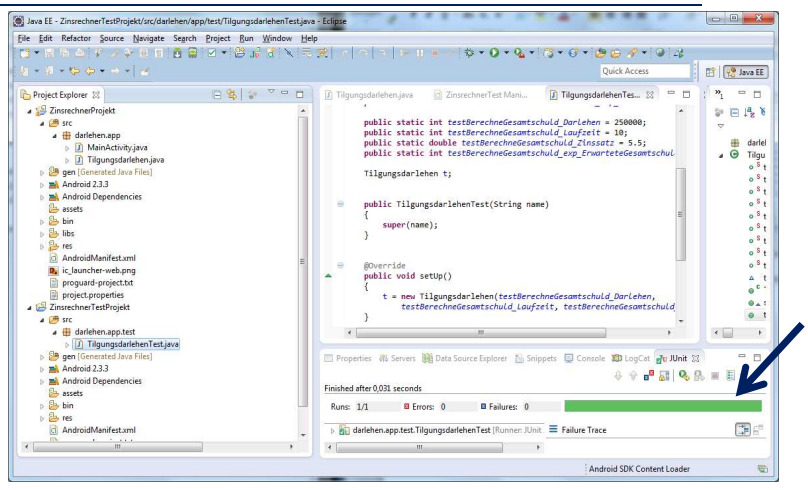
Testklasse



1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: Android JUnit Test zur Android App



Es öffnet sich der Android Emulator und dann auch das Android-JUnit-Testframework.

1 MOTIVATION

Hochschule Bochum
Bochum University
of Applied Sciences

Voraussetzungen / Wünsche für Modultests auf Seite des Wissensträgers

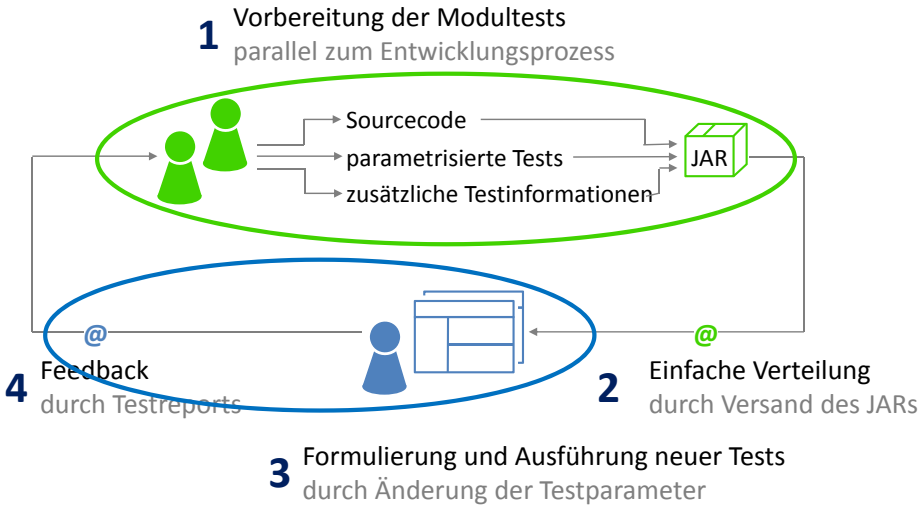
1. Modultests müssen ohne Kenntnis der Programmiersprache Java definiert und ausgeführt werden können.
2. Modultests müssen ohne Kenntnis der Entwicklungsumgebung und ohne Start des Emulators definiert und ausgeführt werden können.
3. Modultests müssen ohne theoretische Kenntnisse zur Testfall-erzeugung auch testen, was nicht funktionieren darf.

KBUnit erlaubt dem Wissensträger, (Android-) JUnit-Tests zu modifizieren und auszuführen, der Wissensträger kommt zu keinem Zeitpunkt mit Programmcode oder der Entwicklungs-umgebung oder dem Emulator in Berührung.
KBUnit generiert auf Wunsch Testfälle unter Berücksichtigung dessen, was nicht funktionieren darf.

2 VORSTELLUNG VON KBUnit

Hochschule Bochum
Bochum University
of Applied Sciences

Zusammenwirken von Entwicklern und Wissensträgern

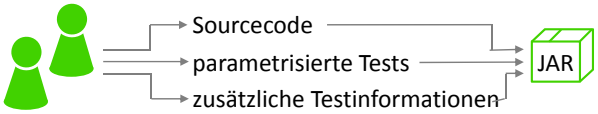


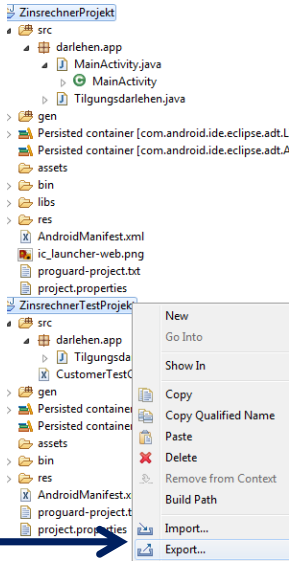
2 VORSTELLUNG VON KBUnit

Hochschule Bochum
Bochum University
of Applied Sciences

Zusammenwirken von Entwicklern und Wissensträgern

1 Vorbereitung der Modultests
parallel zum Entwicklungsprozess





Es ist ein jar-File aus dem Entwicklungsprojekt und dem Testprojekt inklusive zusätzlicher Testinformationen zu erstellen.

2 VORSTELLUNG VON KBUnit

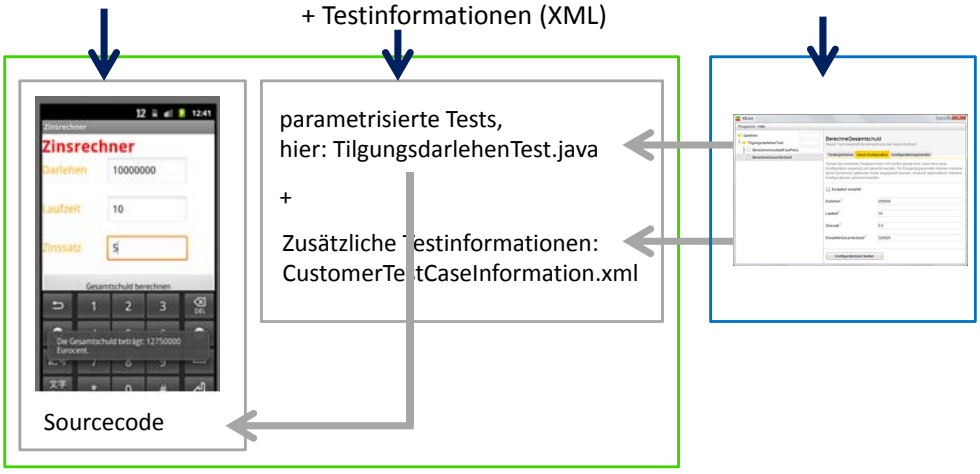
Hochschule Bochum
Bochum University
of Applied Sciences

Zusammenspiel der Komponenten

1 Die zu testende Anwendung

2 (Android-) JUnit-Tests zur Anwendung + Testinformationen (XML)

3 KBUnit



2 VORSTELLUNG VON KBUnit

Hochschule Bochum
Bochum University
of Applied Sciences

Voraussetzungen zum Testen einer Methode mit KBUnit

- Zu einer zu testenden Methode einer **beliebigen Anwendung** muss ein **lauffähiger JUnit-Test** existieren. Parameter, die der Wissensträger variieren soll, müssen als statische Variablen festgelegt werden.
- Informationen zur Methode und den Parametern müssen in einer xml-Datei aufgelistet werden.

Funktionsumfang von KBUnit

- Manuelle Konfiguration von Testfällen durch Modifikation der Test-parameter.
- Automatische Generierung von Testfällen inklusive solcher Testfälle, die Exceptions werfen sollen.
- Persistente Speicherung durchgeführter Tests mit der Möglichkeit zur wiederholten, stapelweisen Ausführung.
- Meldung aufgedeckter Probleme per E-Mail-Report an die Entwickler.

2 VORSTELLUNG VON KBUnit

Hochschule Bochum
Bochum University
of Applied Sciences

KBUnit testet auch

1. C-DLL's:
   Diese werden mittels JNI aus dem JUnit-Test heraus aufgerufen.
2. JavaScript-Funktionen:
   Diese werden mittels JavaScript API aus dem JUnit-Test heraus aufgerufen.
3. ???

KBUnit ist flexibel einsetzbar!
Die JUnit-Tests müssen angepasst werden.

3 LIVE-DEMO

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: KBUnit testet die Android-Anwendung

Es wird eine LIVE-DEMO vorgeführt.

3 LIVE-DEMO

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: KBUnit testet Methoden der Android-Anwendung

```
public class TilgungsdarlehenTest extends TestCase
{
    public static int testBerechneGesamtschuld_Darlehen = 10000000;
    public static int testBerechneGesamtschuld_Laufzeit = 10;
    public static double testBerechneGesamtschuld_Zinssatz = 5;
    public static int testBerechneGesamtschuld_exp_ErwarteteGesamtschuld = 12750000;

    private Tilgungsdarlehen tilg;

    public TilgungsdarlehenTest(String name){
        super(name);
    }

   @Override
    public void setUp(){
        this.tilg = new Tilgungsdarlehen(testBerechneGesamtschuld_Darlehen,
            testBerechneGesamtschuld_Laufzeit, testBerechneGesamtschuld_Zinssatz);
    }
    …
```

Parameterwerte als Attributwerte

3 LIVE-DEMO

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: KBUnit testet Methoden der Android-Anwendung

```
public void testBerechneGesamtschuld(){
    int berechneteGesamtschuld = tilg.berechneGesamtschuld();
    int erwarteteGesamtschuld
        = testBerechneGesamtschuld_exp_ErwarteteGesamtschuld;
    assertEquals(
        "Die berechnete Gesamtschuld entspricht nicht der erwarteten.",
        erwarteteGesamtschuld, berechneteGesamtschuld);
}
...
}
```

3 LIVE-DEMO

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: KBUnit testet Funktionen einer C-DLL

```
public void testBerechneGesamtschuld()
{
    int berechneteGesamtschuld
        = t.berechneGesamtschuld(testBerechneGesamtschuld_Darlehen,
        testBerechneGesamtschuld_Laufzeit,
        testBerechneGesamtschuld_Zinssatz / 100);
    assertEquals("Die berechnete Gesamtschuld entspricht nicht der erwarteten",
        testBerechneGesamtschuld_exp_ErwarteteGesamtschuld,
        berechneteGesamtschuld);
    ...
}
```

Parameterwerte als Parameter der zu testenden nativen Methode / C-Funktion der C-DLL

3 LIVE-DEMO

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: KBUnit testet JavaScript-Funktionen

```
public class TilgungsdarlehenTest extends TestCase
{
    public static int testBerechneGesamtschuld_Darlehen = 10000000;
    public static int testBerechneGesamtschuld_Laufzeit = 10;
    public static double testBerechneGesamtschuld_Zinssatz = 5;
    public static int testBerechneGesamtschuld_exp_ErwarteteGesamtschuld = 12750000;

    private ScriptEngine se;

    public TilgungsdarlehenTest(String name){
        super(name);
    }

    @Override
    public void setUp(){
        ScriptEngineManager manager = new ScriptEngineManager();
        this.se = manager.getEngineByExtension("js");
    }
    …
```

3 LIVE-DEMO

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: KBUnit testet JavaScript-Funktionen

Parameterwerte als Parameter der zu testenden JavaScript-Funktion

```
@Test
public void testBerechneGesamtschuld(){
        int berechneteGesamtschuld;
        this.se.put("aktDarlehen", testBerechneGesamtschuld_Darlehen);
        this.se.put("aktLaufzeit", testBerechneGesamtschuld_Laufzeit);
        this.se.put("aktZinssatz", testBerechneGesamtschuld_Zinssatz / 100);
        this.se.eval(new FileReader("….\\Tilgungsdarlehen.js"));
        berechneteGesamtschuld = ((Double)se.get("ergebnis")).intValue();
        int erwarteteGesamtschuld =
            testBerechneGesamtschuld_exp_ErwarteteGesamtschuld;
        assertEquals("Die berechnete Gesamtschuld entspricht nicht der erwarteten",
        erwarteteGesamtschuld, berechneteGesamtschuld);
}
…
```

3 LIVE-DEMO

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: KBUnit testet auch das, was nicht funktionieren darf

Es wird eine LIVE-DEMO vorgeführt.

3 LIVE-DEMO

Hochschule Bochum
Bochum University
of Applied Sciences

Beispiel :: KBUnit testet auch das, was nicht funktionieren darf

Tilgungsdarlehen.java

```
public int berechneGesamtschuld(String user)
    throws SQLException
{
    int gesamtschuld = 0;
    if("Oesing".equals(user)
        && (this.zinssatz > 0 && this.zinssatz <= 0.2)) {
        ....  ; gesamtschuld = ...;
    }
    else {
        if(!"Oesing".equals(user))
            throw new SQLException(
                "Benutzer ist unbekannt!");
        if(this.zinssatz <= 0 || this.zinssatz > 0.2)
            throw new IllegalArgumentException(
"Zinssatz muss > 0 % und kleiner gleich 20 % sein!");
    }
    return gesamtschuld;
}
```

TilgungsdarlehenTest.java

```
public void testBerechneGesamtschuld()
    throws SQLException
{
    int berechneteGesamtschuld
        = t.berechneGesamtschuld(
        testBerechneGesamtschuld_Benutzer);

    assertEquals("Die berechnete Gesamtschuld "
        + "entspricht nicht der erwarteten",
testBerechneGesamtschuld_exp_ErwarteteGesamtschuld,
        berechneteGesamtschuld);
}
```

Die Überprüfung der Grenzen für das Darlehen und die Laufzeit fehlen! KBUnit generiert dazu Testfälle.

4 AUSBLICK

Hochschule Bochum
Bochum University
of Applied Sciences

Was ist als nächstes geplant?

1. Einlesen der JavaDoc‘s (anstatt CustomerTestcaseInformation.xml, siehe zusätzliche Testinformationen) als Information für den Kunden.
2. Suche und Filterung nach bestimmten Testkonfigurationen.
3. Erweiterung der Eingabemöglichkeit von Parametern um komplexere Datentypen.
4. Modifikation der generierten Testfälle durch den Kunden.
5. Weitere Optimierung der Testfallgenerierung.
6. **Gibt es Ideen / Vorschläge Ihrerseits?**

4 AUSBLICK

Hochschule Bochum
Bochum University
of Applied Sciences

Kontakt

Prof. Dr. Ursula Oesing
Fachbereich Elektrotechnik und Informatik
Hochschule Bochum
Lennershofstraße 140
44801 Bochum
E-Mail: ursula.oesing@hs-bochum.de

Roland Krüger
secunet Security Networks AG
Kronprinzenstraße 30
45128 Essen
E-Mail: roland.krueger@secunet.com

Modultests in der Softwareentwicklung für mobile Geräte durch den Wissensträger mittels KBUnit

---

Vielen Dank für Ihr Interesse!

Ursula Oesing
Fachbereich Elektrotechnik und Informatik, HS Bochum

Roland Krüger
secunet Security Networks AG, Essen

4. Agilität

Sessionüberblick

4.1. Agiler Einkauf (27.02.2013) 76
4.2. Einführung von Scrum (27.02.2013) 85
4.3. Agile Konzepte im Unternehmen verankern (28.02.2013) 99
4.4. Agile Software-Entwicklung in regulierten Umgebungen (28.02.2013) 104

Agiler Einkauf

Norbert Hölsken

it-agile GmbH
Paul-Stritter-Weg 5
22297 Hamburg
norbert.hoelsken@it-agile.de

Abstract: Dieser Vortrag befasst sich mit der Fragestellung, warum klassische Werkverträge selten zu guter Software führen und was es für Alternativen gibt. Am Beispiel konkreter Projekte wird gezeigt, wo die Stärken und Schwächen von Werkverträgen im Softwareumfeld liegen. Darauf aufbauend werden alternative Ansätze vorgestellt, welche die hohe Komplexität und Unsicherheit der Softwareentwicklung stärker berücksichtigen und daher meist eine bessere Alternative darstellen.

1. Was ist das Problem am klassischen Werkvertrag?

Im klassischen Werkvertrag wird vom Auftraggeber ein zu lieferndes Gewerk detailliert beschrieben und anschließend vom Auftragnehmer zu einem vereinbarten Preis geliefert. Für Änderungen am Gewerk ist vertraglich geregelt, wie mit diesen umzugehen ist. Oft trifft man an dieser Stelle auf die berüchtigten Change Requests, die teilweise teurer sein können als das ursprüngliche Gewerk. Nicht selten versucht nämlich der Auftragnehmer, seinen günstigen Preis aus dem ursprünglichen Angebot über diese Änderungen wieder aufzubessern und dadurch überhaupt erst in die Gewinnzone zu kommen.

Aufgrund des teuren Wechsels zu einem neuen Anbieter bleibt dem Auftraggeber häufig keine andere Wahl, als auf diesen erhöhten Preis des Change Requests einzugehen. Neben dieser Problematik, verursacht der Werkvertrag weitere Probleme, die im Vortrag näher beleuchtet werden.

2. Welche Alternativen gibt es?

Die offensichtliche Alternative hierzu ist ein Dienstleistungsvertrag. Nur verliert man dadurch viele der Vorteile eines Werkvertrages. Was wird aus der Gewährleistung? Wie verhindere ich, dass langsamer als möglich gearbeitet wird? Muss ich als Auftraggeber jetzt das ganze Projekt leiten?

Der Vortrag zeigt, dass es durchaus Mischformen gibt, die zumindest einige der Vorteile von Werkverträgen enthalten, aber dennoch besser auf die Realität in der Softwareentwicklung zugeschnitten sind. Dies kann ein “agiler Festpreis” sein oder verschiedene Ansätze wie “Proviant und Prämie” oder “Money for Nothing / Change for Free”.

Neben den Vor- und Nachteilen der einzelnen Vertragsarten, geht der Vortrag darauf ein, wie die Scrum-Rollen zwischen Auftraggeber und Auftragnehmer verteilt werden sollten, um eine optimale Projektabwicklung zu unterstützen.

3. Fazit

Oft scheitern viele IT-Projekte schon beim Einkauf. Viel Kreativität und Potenzial kann durch eine falsche Vertragsart gleich zu Beginn verloren gehen. Dieser Vortrag richtet sich sowohl an Einkäufer und IT-Entscheider als auch an IT-Dienstleister und soll helfen , eine bessere Zusammenarbeit zwischen Auftraggebern und Auftragnehmern zu ermöglichen. Nur so können Projekte erfolgreicher und Produkte besser werden.

Agiler Einkauf

Warum auch der Einkauf agiler werden sollte

Norbert Hölsken, nhoelsken@it-agile.de, 27.2.13, SEE Aachen

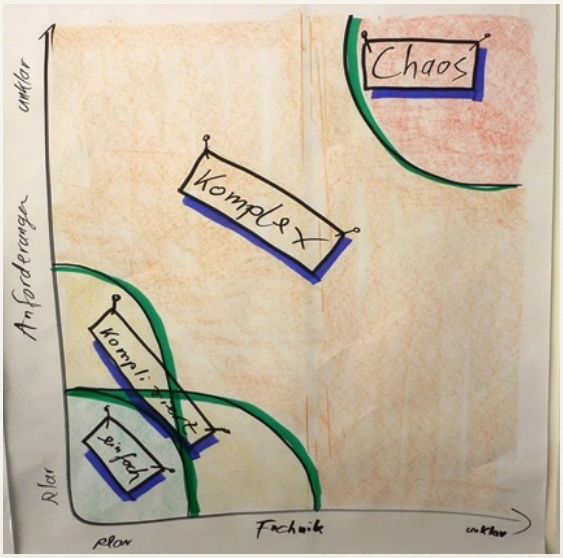
it-agile

Es gibt mehrere gute Gründe für Werkverträge.

- Projektverantwortung beim Auftragnehmer
- Gewährleistung
- Klarer Umfang
- Bekannter Preis
- Fester Liefertermin

it-agile

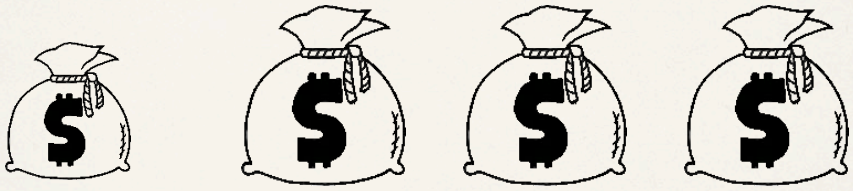
Denkfehler #1: Klare Anforderungen



Stacey-Diagramm

it-agile

Denkfehler #2: Budgetsicherheit



Festpreis | Change Request | Change Request | Change Request

it-agile

Denkfehler #3:
Zur Not zahlt der Auftragnehmer.

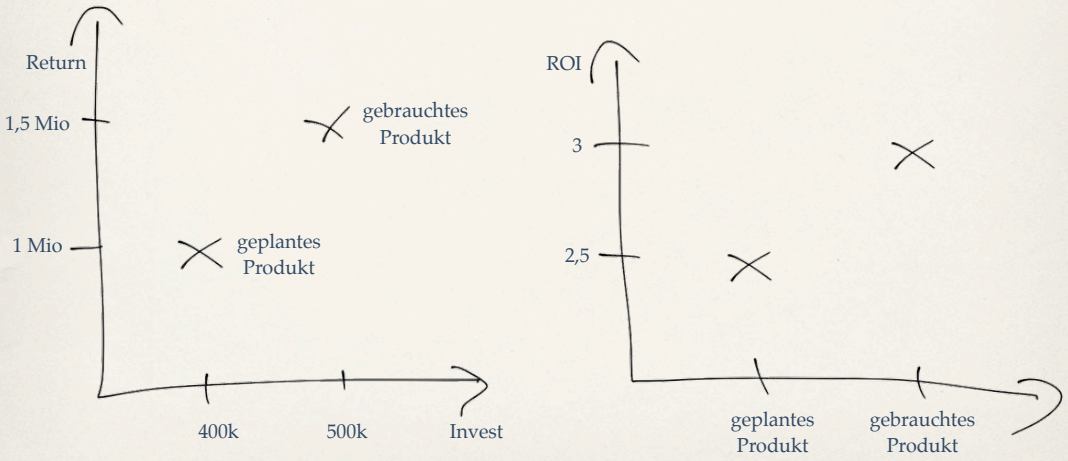
it-agile

Denkfehler #4:
Dann weiß ich, was ich bekomme.

... oft ist es nur leider nicht das, was ich am Ende des Projekts wirklich brauche.

it-agile

Der Markt / die Anwender entscheiden über den Erfolg.



Auf den Gesamt Return-on-Investment kommt es an.

„Beim Dienstleistungs-Vertrag habe ich keine Sicherheit.“

it-agile

Bedenken #1: Was wird aus der Gewährleistung?

- Meist schon mit 15-20% eingepreist
- Separater Wartungs-/Pflegevertrag
- Mit agilen Methoden schnell sichtbar, wenn die Leistung nicht stimmt

it-agile

Bedenken #2: Es wird zu langsam entwickelt.

- Mit agilen Methoden schnell sichtbar, wenn die Leistung nicht stimmt
- Vertrauen muss da sein
- Restrisiko bleibt sicherlich

it-agile

Bedenken #3: Risiko allein beim Auftraggeber

- Risiko deutlich kleiner bei agiler Lieferung
- Bessere Priorisierung durch Auftraggeber möglich
- Coaching des Product Owners beim Auftraggeber wichtig
- Auftragnehmer hat Eigeninteresse an Wieder-/Weiterbeauftragung

it-agile

Mögliche Alternativen zum Werkvertrag

- Dienstleistungsvertrag mit agilem Vorgehen
- Agiler Festpreis
- Proviant und Prämie
- „Money for Nothing, Change for Free“

it-agile

Vielen Dank!

Coaching
Management-Beratung
Lean-Startup
ScrumMaster
Scrum
Produktentwicklung
Product-Owner
Entwicklungsteam
Schulung
Kanban

Nehmen Sie gerne Kontakt auf:
E-Mail: norbert.hoelsken@it-agile.de
Twitter: @nhoelsken
Web: www.it-agile.de

it-agile

Einführung von Scrum

Stefan Roock

it-agile HmbH
Paul-Stritter-Weg 5
22297 Hamburg
stefan.roock@it-agile.de

Abstract: Scrum ist heute der vorherrschende agile Ansatz in der Softwareentwicklung. Grund genug, sich mit der Einführung von Scrum in Projekte und Unternehmen zu befassen. Dieser Artikel diskutiert verschiedene Einführungsstrategien von Scrum und zeigt die Vor- und Nachteile der einzelnen Verfahren auf.

1. Projekte, Teams und Unternehmen

Zunächst lohnt sich ein Blick auf den Umfang der Scrum-Einführung. Idealtypisch können wir Scrum in ein Team einführen oder großflächiger in das ganze Unternehmen. Und wir können zwei extreme Einführungsstrategien unterscheiden. Wir können Scrum schrittweise einführen oder komplett.

	Team	**Unternehmen**
schritt-weise		
komplett		

Daraus resultieren vier idealtypische Einführungsstrategien. Wir können Scrum in ein Team schrittweise oder komplett einführen und wir können Scrum ins Unternehmen schrittweise oder komplett einführen.

1.1. Scrum schrittweise ins Team einführen

Auf Team-Ebene ist eine inkrementelle Scrum-Einführung dann angesagt, wenn im Team bereits ohne Scrum alles sehr gut läuft und nach Verbesserungsmöglichkeiten gesucht wird. Durch das vergleichsweise geringe Risiko der schrittweisen Einführung kann das, was bereits gut läuft, leicht bewahrt werden. Ein anderer Anwendungsfall für die schrittweise Einführung ins Team liegt dann vor, wenn es keine Management-Unterstützung für Scrum gibt. Dann ist eine vollständige Scrum-Einführung ins Team i.d.R. nicht möglich. Man führt dann die Scrum-Praktiken ein, die nach außen hin noch keine großen Auswirkungen haben und versucht dann, das Management ins Boot zu holen. In beiden Fällen dauert eine schrittweise Scrum-Einführung verhältnismäßig lange und führt i.d.R. nicht zu einer vollständigen Scrum-Implementation.

1.2. Scrum komplett ins Team einführen

Scrum wird man dann komplett ins Team einführen, wenn es bereits große Probleme gibt. Scrum wird beispielsweise immer häufiger eingesetzt, wenn Projekte mit traditionellen sequenziellen Verfahren gescheitert sind. Ein zweiter Anwendungsfall liegt vor, wenn das Team vor einer neuen Herausforderung steht, die es mit dem existierenden Verfahren nicht meistern kann. Die Kompletteinführung führt schnell zu vollständigen Scrum, bringt aber natürlich ein größeres Risiko mit sich als die schrittweise Einführung. Andererseits ist Scrum auf diese Weise unzählige Male erfolgreich eingeführt worden, so dass das Risiko überschaubar bleibt - wenn man sich qualifizierte Hilfe holt.

1.3. Scrum schrittweise ins Unternehmen einführen

Wenn positive Erfahrungen mit Scrum in einzelnen Teams vorliegen, entscheiden sich Unternehmen häufig, Scrum breiter ins Unternehmen einzuführen. Das bedeutet zunächst, in mehr oder sogar alle Teams Scrum einzuführen. Um Risiken zur reduzieren, werden dann nicht alle Teams auf einmal umgestellt, sondern schrittweise ein Team nach dem anderen. Die Einführung in die Teams ist dann aber häufig komplett und nicht mehr schrittweise - durch die vorherige erfolgreiche Anwendung von Scrum in die initialen Teams kennt man die meisten Stolpersteine bereits.

1.4. Scrum komplett ins Unternehmen einführen

Scrum komplett ins ganze Unternehmen oder zumindest alle Teams auf einmal einzuführen, erscheint auf den ersten Blick als der reine Wahnsinn. Es gibt ein großes Risiko, dass durch die massiven Änderungen am Ende gar nichts mehr funktioniert. Wer würde so ein Risiko in Kauf nehmen? Unternehmen, die mit dem Rücken zur Wand stehen. Tatsächlich wird Scrum immer wieder komplett ins Unternehmen eingeführt, wenn diese akut vom Konkurs bedroht sind. Natürlich gelingt es auch mit Scrum nicht, jeden Konkurs abzuwenden. Da es für das Unternehmen aber auch nicht mehr schlimmer werden kann, wird das entsprechende Risiko in Kauf genommen. Aber nicht nur

Unternehmen, die vor dem Konkurs stehen, stehen mit dem Rücken zur Wand. Mitunter ändert ein wirtschaftlich sehr erfolgreiches Unternehmen seine Strategie und weiß, dass es diese Strategie mit dem bisherigen Entwicklungsverfahren nicht umsetzen kann. Auch dann kann die Situation durch die Kompletteinführung von Scrum nicht mehr schlimmer werden.

1.5. Überblick über die Einführungsstrategien

	Team	**Unternehmen**
schritt-weise	wenn alles gut läuft oder Management-Unterstützung fehlt; geringes Risiko; dauert lange und endet häufig nicht mit vollständigem Scrum	wenn positive Erfahrungen mit Scrum in einzelnen Teams vorliegen; geringes Risiko; dauert lange
komplett	wenn im Projekt eine besondere Herausforderung existiert oder es schwerwiegende Probleme gibt; moderates Risiko; schnell zu vollständigem Scrum	wenn das Unternehmen mit dem Rücken zur Wand steht; hohes Risiko; mitunter hohe Änderungsresistenz bei den Mitarbeitern; schnell

2. Scrum schrittweise ins Team einführen

Scrum wird auf Teamebene in der Regel komplett auf einmal eingeführt. Das ist aber nicht die einzige Option. Es ist sehr wohl möglich, Scrum auch schrittweise und sanft auf Teamebene einzuführen.

Dabei beginnt man häufig mit Daily Scrums und einem Taskboard als Visualisierung der anstehenden Arbeit. Die einzige Voraussetzung ist, dass sich eine Gruppe von Menschen identifizieren lässt, die als Team arbeiten können und so von einem Austauschmeeting wie dem Daily Scrum auch profitieren.

Danach kann man sich die Retrospektiven vornehmen, um Zusammenarbeit und Prozess zu verbessern. Die Retrospektiven finden dann z.B. einfach monatlich statt.

Der nächste sinnvolle Schritt ist dann überraschenderweise häufig das Sprint-Review. In regelmäßigen Abständen (z.B. auch monatlich), wird der erreichte Projektstand demonstriert und evaluiert. Ausgehend davon kann dann i.d.R. leicht das Sprint-Planning installiert werden. Dieses führt allerdings meist noch nicht zu richtigen Sprints. Sprint-

Planning und –Review basieren zunächst weiter auf dem klassischen Projektplan und der sieht keine monatlichen Produktinkremente vor.

Und dann kommen die anspruchsvolleren Bestandteile des Scrum-Frameworks. Sie greifen mehr oder weniger stark in die existierenden Unternehmensstrukturen ein.

Das Product Backlog ist mitunter ein sinnvoller nächster Schritt. Es unterscheidet sich allerdings so stark von klassischen Projektplänen und Lastenheften, dass es sich i.d.R. nur in Abstimmung mit weiteren Unternehmensteilen sinnvoll einführen lässt.

Wenn das Product Backlog installiert ist, können richtige Sprints durchgeführt werden, die tatsächlich potenziell auslieferbare Produktinkremente als Ergebnis haben.

Und schließlich bleiben die Rollen. Es ist i.d.R. leicht abgrenzbar, wer zum Entwicklungsteam gehört. Allerdings ist es damit nicht getan. Die Teammitglieder müssen lernen, wie sie wirklich als Team zusammenarbeiten und ein gemeinsam verantwortetes Ergebnis erstellen können.

Eine Art ScrumMaster kann man bereits sehr früh im Team benennen. Ein echter ScrumMaster kümmert sich allerdings auch um die Hindernisse, die außerhalb des Teams liegen und beginnt damit, das Unternehmen zu verändern. Daher benötigt diese Rolle i.d.R. die Erlaubnis des Managements.

Noch deutlicher wird dies beim Product Owner. Dieser ist nach Scrum bevollmächtigt, das Product Backlog zu priorisieren und zu ändern, wenn dadurch ein erfolgreicheres Produkt entsteht. Es erfordert vom Unternehmen Einiges, soviel Delegation zuzulassen.

3. Scrum komplett ins Team einführen

Um Scrum komplett ins Team einzuführen, benötigt man mind. die „Erlaubnis“ des Managements. Ansonsten wird man sehr viele Schwierigkeiten mit der Etablierung der Product Owner-Rolle sowie dem Product Backlog bekommen.

Wir haben gute Erfahrungen damit gemacht, das Management für ein zeitlich begrenztes Pilotprojekt mit ins Boot zu holen. So muss das Management nicht gleich einer dauerhaften Reorganisation zustimmen, sondern nur für die Dauer das Pilotprojektes (typischerweise zwischen 3 und 6 Monaten) dem Team den Rücken freihalten.

Weiterhin hat es sich als sehr sinnvoll erwiesen, das Pilotprojekt gleich von Anfang an mit umfangreicher Scrum-Erfahrung auszustatten. Das kann dadurch geschehen, dass man einen ScrumMaster mit Praxiserfahrung ins Team integriert oder das Team und den ScrumMaster durch einen externen Scrum-Coach begleiten lässt. Häufig ist hier eine Mischform sinnvoll, das sogenannte Tandem. Der externe Coach übernimmt die offizielle ScrumMaster-Rolle und der interne ScrumMaster begleitet ihn auf Schritt und Tritt, um von ihm zu lernen. In der Tandem-Metapher sitzt der Coach vorne und lenkt. Der Coach übergibt Aufgaben und Verantwortung schrittweise an den internen

ScrumMaster, bis dieser beim Tandem vorne sitzt und letztlich ganz alleine arbeiten kann.

4. Scrum schrittweise ins Unternehmen einführen

Der Scrum-Piloten sollte durch mehrere Management-Workshops begleitet werden, in denen die Einsichten aus dem Pilotprojekt ausgewertet werden. Auf dieser Basis soll gegen Ende des Piloten entschieden werden, ob man im Unternehmen Scrum weiter verbreiten möchte oder nicht.

Wichtig ist hier, dass tatsächlich eine Entscheidung gefällt wird. Allzu oft gibt weder ein klares Bekenntnis zu Scrum aus dem Management noch eine klare Absage. Das Ergebnis ist für die Beteiligten dann sehr irritierend. War der Pilot ein Erfolg, wissen die Teammitglieder nicht, ob sie mit Scrum weitermachen sollen oder dürfen.

Wenn die Entscheidung pro Scrum fällt, hat sich eine organische Ausbreitung von Scrum im Unternehmen bewährt. Das Pilotteam wird aufgeteilt in zwei oder drei Teams. Diese werden dann mit neuen Teammitgliedern aufgefüllt. Nachdem diese neuen Teams erfolgreich mit Scrum gearbeitet haben, werden auch sie wieder geteilt und aufgefüllt usw.

Auch wenn durch die ständigen Teamänderungen jeweils die Entwicklungsgeschwindigkeit einbricht, überwiegen i.d.R. die Vorteile. Das Scrum-Wissen und die –Erfahrung verbreiten sich sehr schnell. Die Art und Weise, wie Scrum im Unternehmen gelebt wird, breitet sich ebenfalls organisch aus.

Mit der Einführung von Scrum in alle Teams ist es in der Regel aber nicht getan. Die generellen Management-Strukturen müssen parallel an die höhere Eigenverantwortlichkeit der Product Owner und Teams eingepasst werden. Evtl. vorhandene Stage-Gate-Prozesse müssen adaptiert werden etc.

Daher lohnt sich in größeren Unternehmen ein eigenes Transitionsteam zu gründen, das sich mit Scrum organisiert. Ein hochrangiger Manager (z.B. Geschäftsführer) nimmt die Product Owner-Rolle des Transitionsteams ein. Das Team selbst ist durch weitere Manager, Fachexperten und Change-Agents besetzt. Der ScrumMaster ist häufig ein externer Coach. Die Aufgaben des Transitionsteams stehen in einem Transitions-Backlog und das Ergebnis (Produkt) des Transitionsteams ist ein agiles Unternehmen. Entsprechend macht jeder Sprint des Unternehmen ein Stück agiler.

5. Scrum komplett ins Unternehmen einführen

Bei der Kompletteinführung in ein größeres Unternehmen ist unbedingt ein Transitionsteam notwendig, wie es im vorigen Abschnitt beschrieben wurde. Außerdem ist nach unserer Erfahrung massive Unterstützung durch erfahrene Scrum-Coaches notwendig.

Dann allerdings kann man mit der Strategie sehr schnell sehr gute Erfolge erzielen.

6. Abschluss

Die vorgestellten Einführungsstrategien sind idealtypisch. In der Praxis gibt es natürlich beliebige Misch- und Zwischenformen.

Scrum einführen

SEE 2013, Aachen

stefan.roock@it-agile.de

Twitter: @StefanRoock

Montag, 7. Januar 13

Einführungssituationen

	Team	Unternehmen
schritt-weise		
komplett		

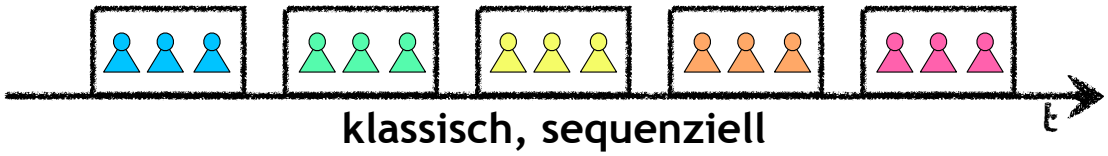
Montag, 7. Januar 13

Einführungssituationen

	Team	**Unternehmen**
schritt-weise	wenn alles gut läuft oder Management-Unterstützung fehlt; geringes Risiko; dauert lange und endet häufig nicht mit vollständigem Scrum	wenn positive Erfahrungen mit Scrum in einzelnen Teams vorliegen; geringes Risiko; dauert lange
komplett	wenn im Projekt eine besondere Herausforderung existiert oder es schwerwiegende Probleme gibt; moderates Risiko; schnell zu vollständigem Scrum	wenn das Unternehmen mit dem Rücken zur Wand steht; hohes Risiko; mitunter hohe Änderungsresistenz bei den Mitarbeitern; schnell

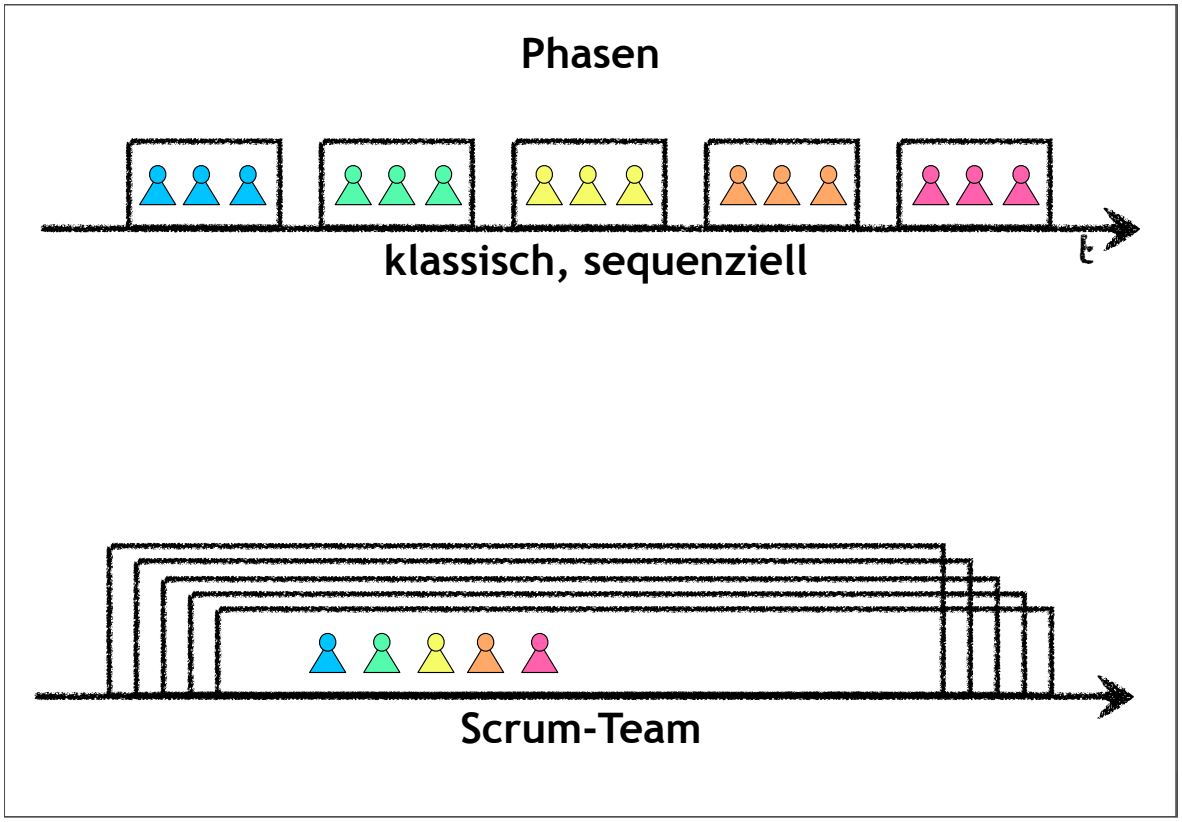
Montag, 7. Januar 13

Phasen

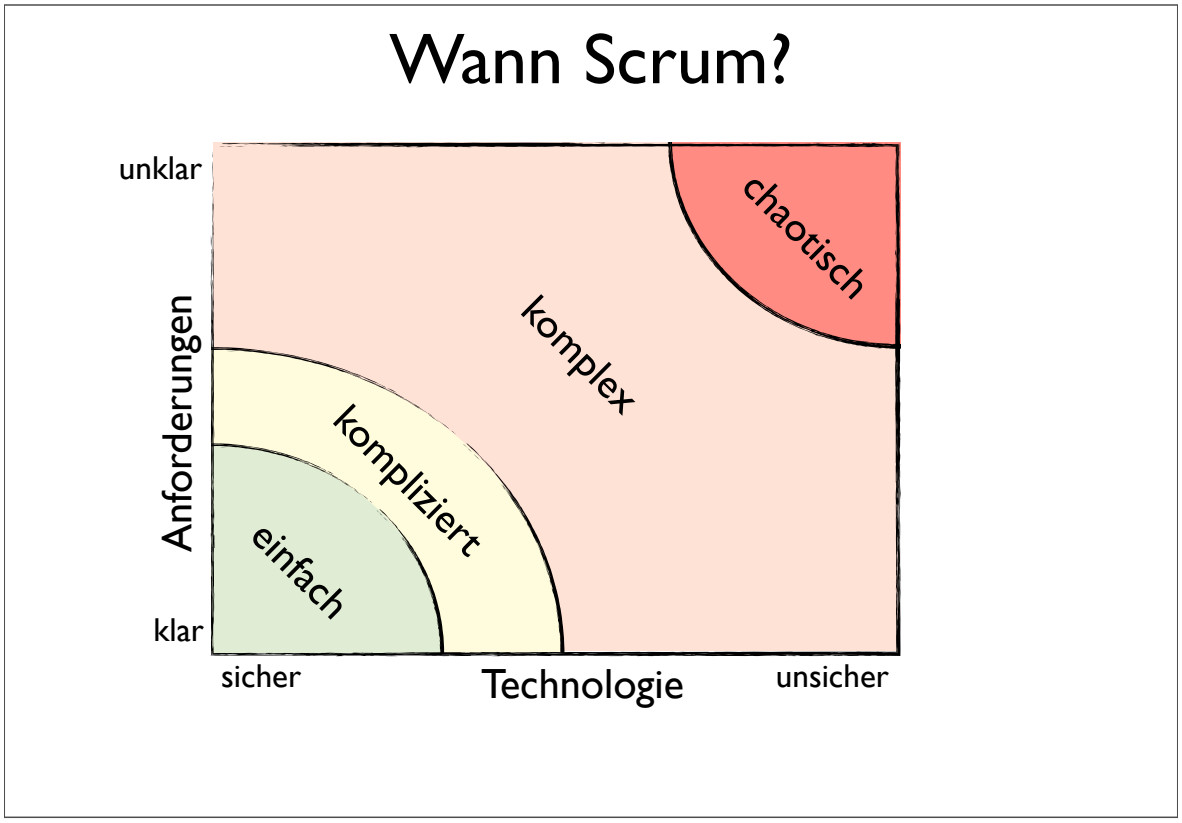


klassisch, sequenziell

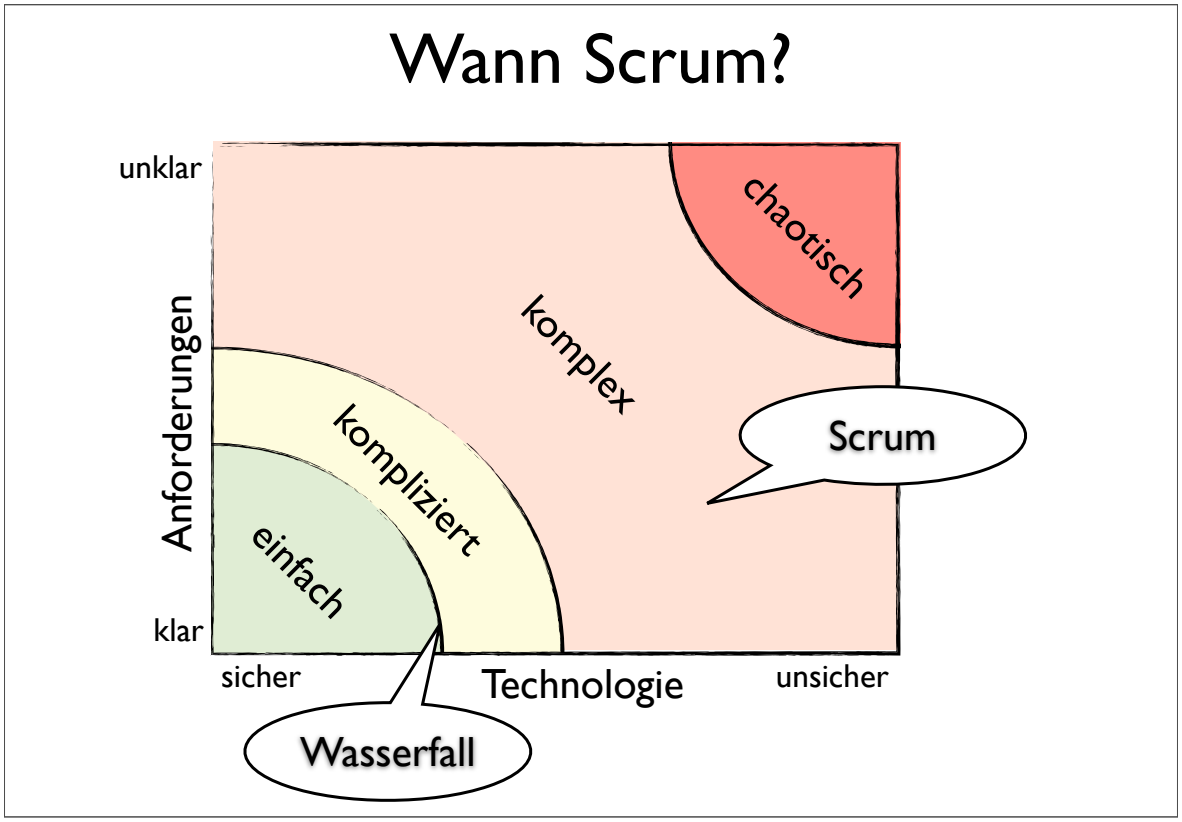
Montag, 7. Januar 13



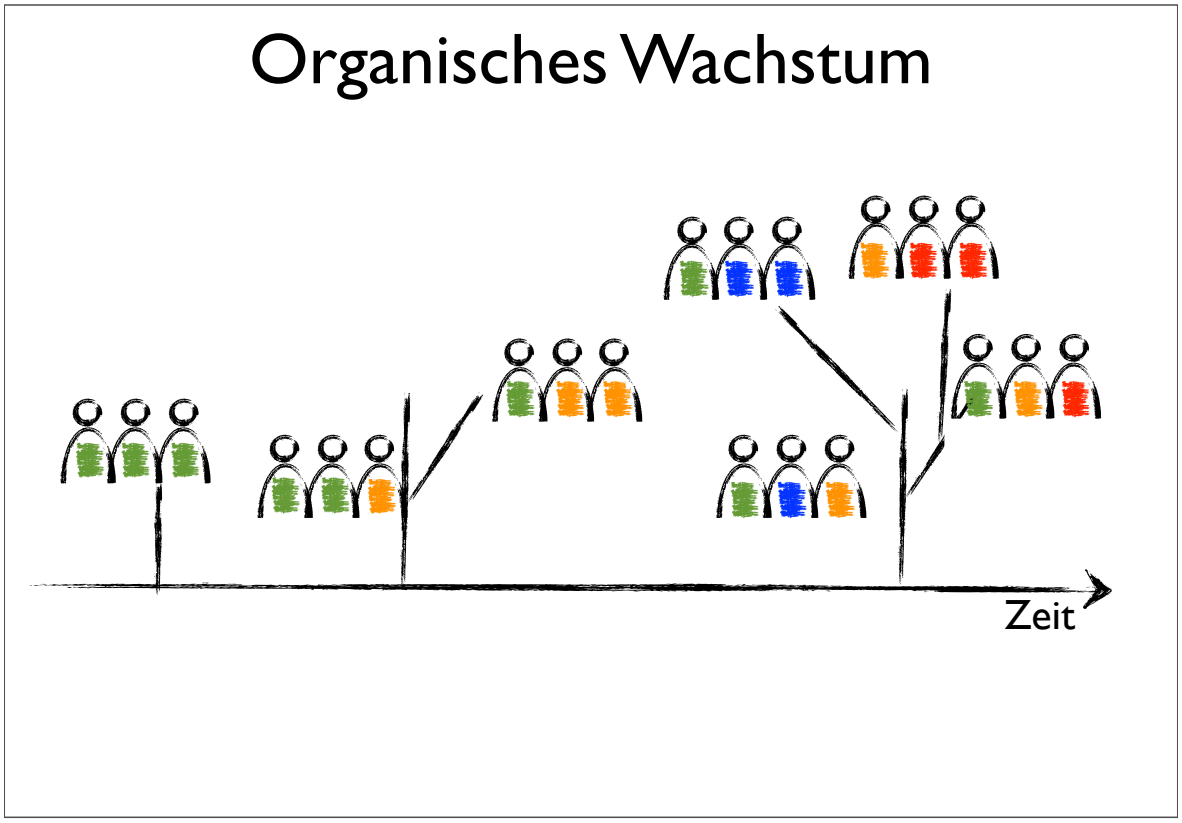
Montag, 7. Januar 13



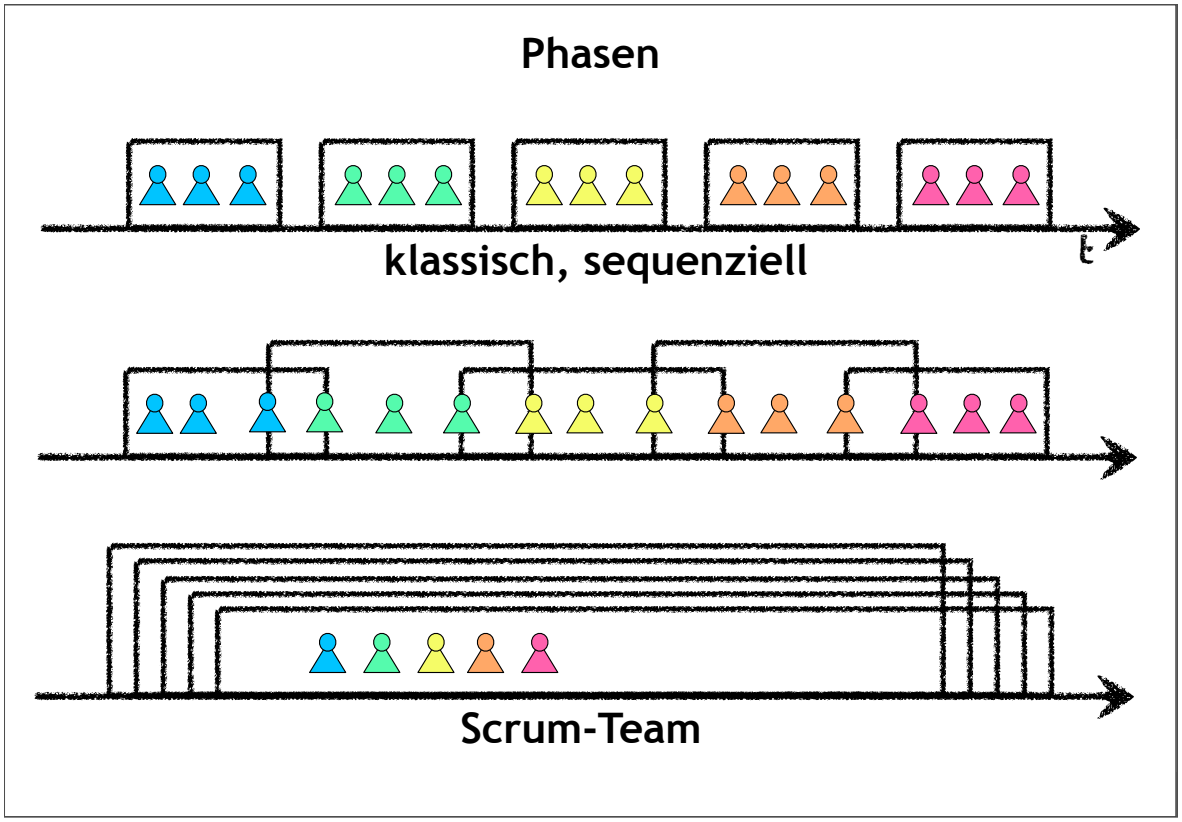
Montag, 7. Januar 13



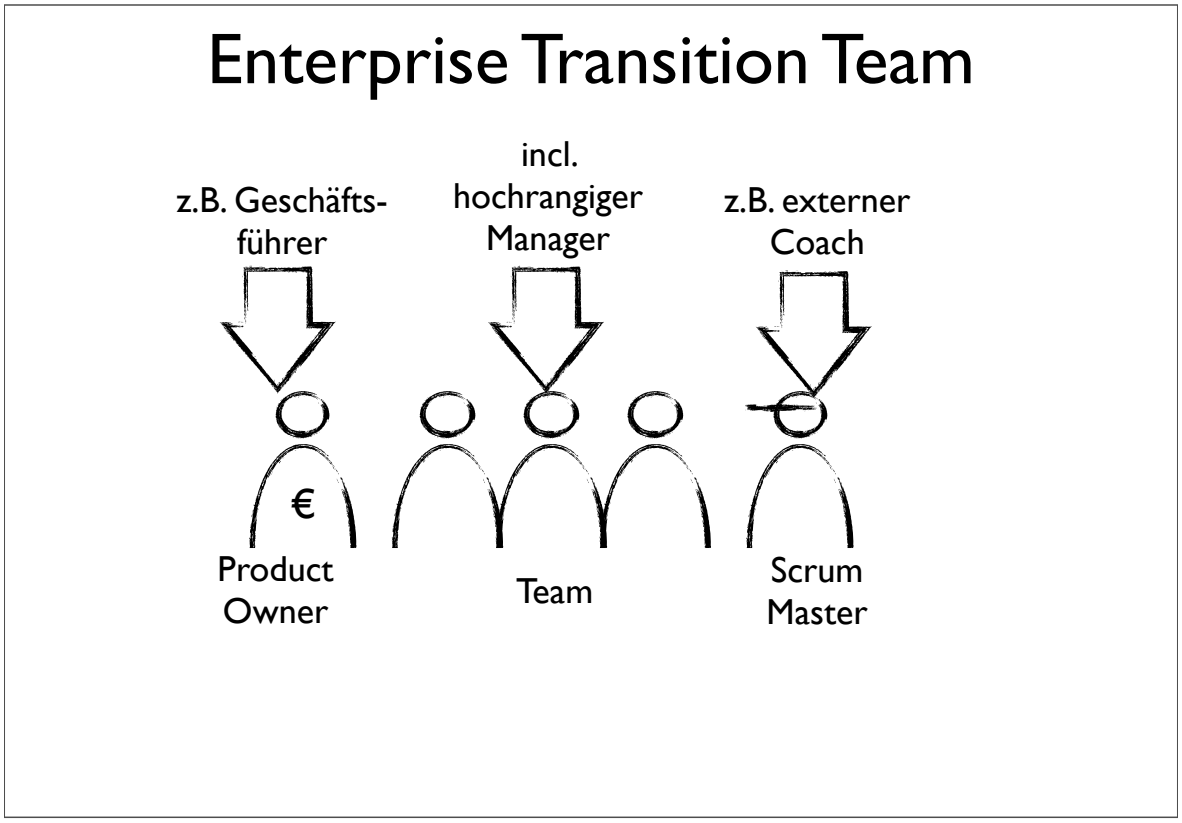
Montag, 7. Januar 13



Montag, 7. Januar 13

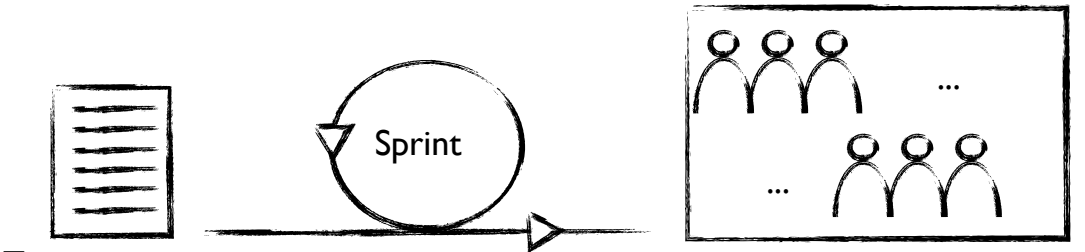


Montag, 7. Januar 13



Montag, 7. Januar 13

Enterprise Transition mit Scrum

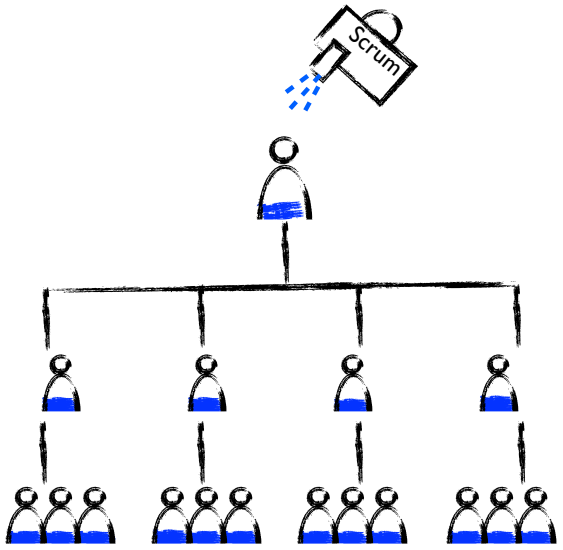


Transition Backlog

Produkt: Scrum-Teams in einer agilen Organisation

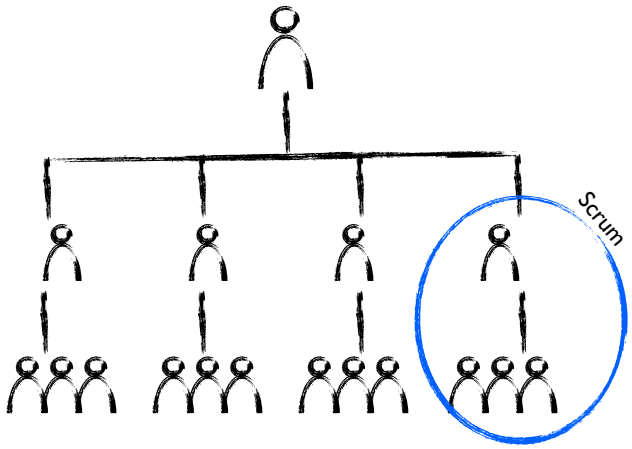
Montag, 7. Januar 13

Agile Transition mit der Gießkanne



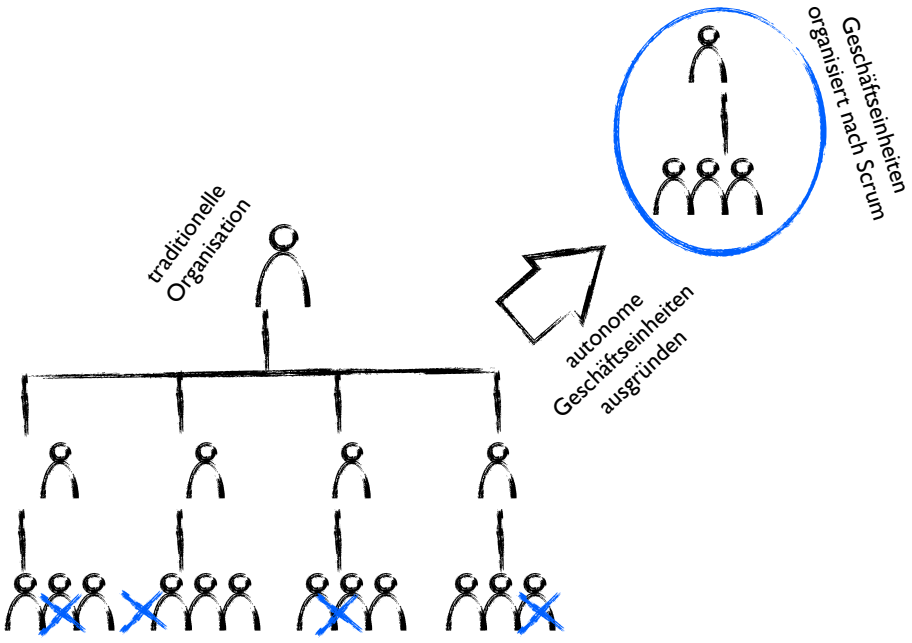
Montag, 7. Januar 13

Abteilungsweise Scrum-Einführung

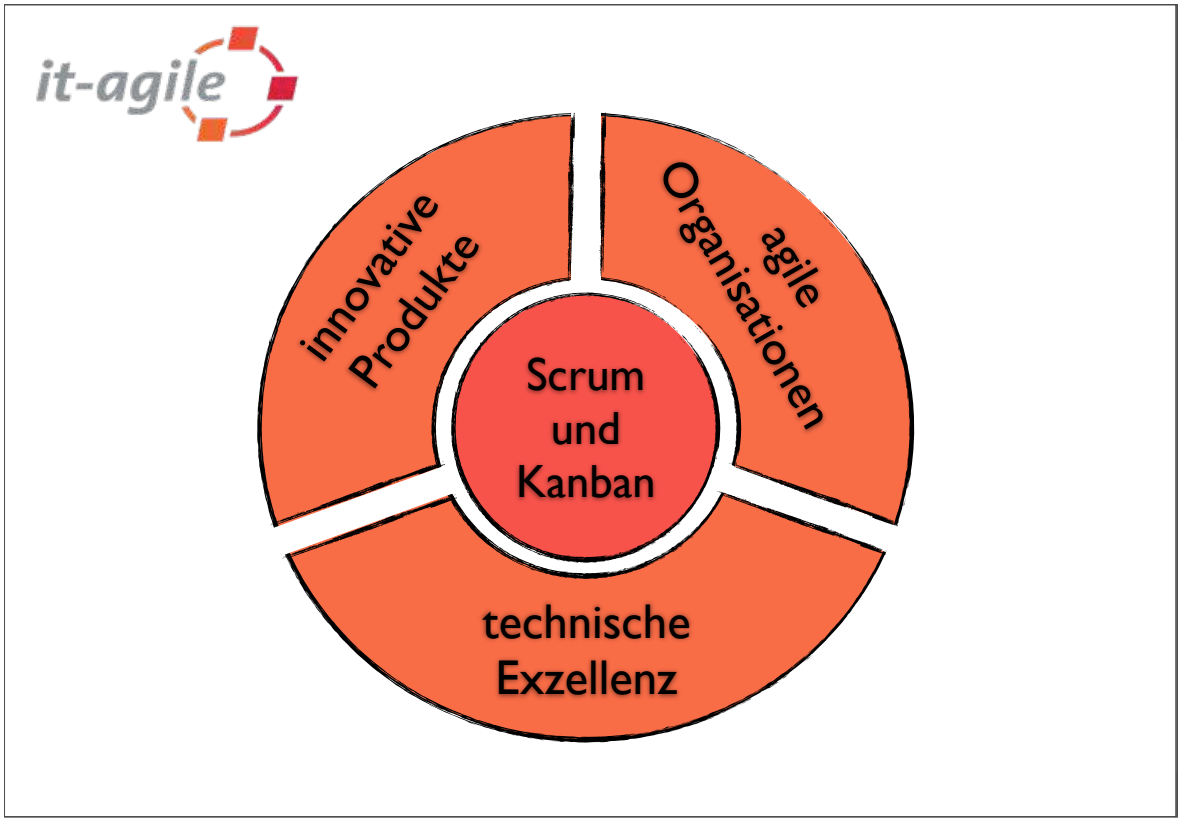


Montag, 7. Januar 13

Geschäftseinheiten ausgründen



Montag, 7. Januar 13



Montag, 7. Januar 13

Agile Konzepte im Unternehmen verankern

Dipl.-Inform. Jörn Koch, Prof. Dr. Heinz Züllighoven

C1 WPS GmbH
Vogt-Köll-Str. 30
22527 Hamburg
Joern.Koch@c1-wps.de
Prof.Dr.Heinz.Zuellighoven@c1-wps.de

Abstract: "Agil" ist inzwischen als risikoarme und zielorientierte Vorgehensweise akzeptiert. Wir begegnen kaum noch Unternehmen, die nicht zumindest ein agiles Pilotprojekt durchführen. Doch ein erfolgreiches Pilotprojekt bedeutet noch keine agile Unternehmenskultur! Wichtig ist eine unternehmensweite Denkweise, die ein inkrementelles Vorgehen auch außerhalb der Entwicklungsteams etabliert. Im Vortrag berichten wir auf Basis unserer umfangreichen Erfahrungen über die kritischen Punkte bei der Einführung agiler Konzepte und geben in unseren "Lessons Learned" einen Einblick in bewährte Strategien und Vorgehensweisen, sowie Beispiele für deren erfolgreiche Anwendung.

1. Agilität als risikoarme und zielorientierte Vorgehensweise für das ganze Unternehmen

"Agil" ist inzwischen als risikoarme und zielorientierte Vorgehensweise akzeptiert. Wir begegnen kaum noch Unternehmen, die nicht zumindest ein agiles Pilotprojekt erfolgreich durchgeführt haben. Jenseits der Grenzen einzelner agiler Entwicklungsteams stoßen wir aber immer wieder auf Zweifel, ob Agilität auch über das Entwicklungsteam hinaus sinnvoll ist und gelingen kann. Neben festsitzenden Missverständnissen zum Thema Agilität, die wir im Vortrag aufgreifen, erweisen sich auch typische Strukturen in großen Unternehmen als schwer zu überwindende Hindernisse auf dem Weg zu einer agilen Unternehmenskultur: Große Planungshorizonte, eine Vielzahl von Legacy-Systemen, Kopfmonopole, eine strikte Arbeitsteilung, lange Kommunikationswege, viele parallele Aufgaben und Projekte pro Mitarbeiter – um nur einige typische Hindernisse zu nennen.

Im Vortrag schaffen wir einen Überblick über diese problematischen Strukturen und geben eine detaillierte Einschätzung derartiger Hindernisse vor dem Hintergrund, ein Unternehmen agiler zu machen zu wollen.

2. Wasserfall-artige Strukturen außerhalb der Entwicklungsteams aufbrechen

Um Agilität unternehmensweit zu etablieren, müssen die oft Wasserfall-artigen und schwergewichtigen Prozesse, die den Entwicklungsprozess umhüllen, aufgebrochen werden. Desgleichen müssen Kommunikationswege und Rückkopplungszyklen so verkürzt werden, dass in kurzen Abständen Ergebnisse angefordert, geliefert und abgenommen werden können. Neben der erfolgreichen Projektdurchführung sollte auch der Projektstatus regelmäßig berichtet werden und projektübergreifend vergleichbar sein.

Dazu ist eine Vielzahl von Schritten und Techniken notwendig, die wir im Vortrag genauer betrachten. Neben bekannten agilen Techniken, wie z.B. Retrospektiven, die weiterhin eine wichtige Rolle spielen, sind das unter anderem die folgenden Ansätze:

- Wir beginnen oft mit einer verbindlichen Teamzuordnung, die von der Anforderungsdefinition über die Umsetzung und Qualitätskontrolle bis hin zur Abnahme alle beteiligten Rollen umfasst – also insbesondere auch die Rollen außerhalb des Entwicklungsteams! Dabei ist uns eine möglichst hundertprozentige Zuordnung der Teammitglieder wichtig, wenngleich sie selten erreicht werden kann.

- Um Wasserfall-artige Strukturen aufzubrechen, setzen wir den zentralen Fokus auf die Qualität der Ergebnisse der einzelnen Sprints. Dadurch sind die Anforderungsgeber des Teams gezwungen, ihre Anforderungen in abnehmbare Ausbaustufen aufzuteilen und dafür nachprüfbare Akzeptanzkriterien zu definieren – das ist ein entscheidender Schritt hin zu Agilität. Die Prozessqualität kann nun anhand der erreichten Ergebnisse kontrolliert werden.

- Dabei hilft uns ein besonderes, von uns entwickeltes Quality-Gate zur Prozessqualität. Das bekannte Konzept der Quality-Gates betrachtet üblicherweise die Produktqualität. In unserem Fall kontrollieren wir jedoch die Qualität des agilen Prozesses über ein Quality-Gate, indem wir bestimmte, gut ermittelbare Kennzahlen am Ende jedes Sprints abgreifen. Beispielsweise: Wie viele Stories wurden begonnen? Wie viele Stories wurden abgeschlossen? Wie viele davon getestet? Wie viele davon erfolgreich abgenommen? Auf der Basis dieser Kennzahlen lassen sich Probleme im Prozess schnell identifizieren. So lässt eine hohe Anzahl offener Stories am Ende eines Sprints z.B. nicht zu, das Sprintergebnis verbindlich zu bewerten – ein gefährlicher Befund, da trotz agiler Techniken ähnliche Risiken wie in einem Wasserfall-Prozess drohen.

- Auf der Basis des Quality-Gates ist ein verbindliches und projektübergreifend vergleichbares Reporting möglich, ohne Themen dadurch zu vereinfacht oder zu projektspezifisch darzustellen.

3. Agilität ist Projektsache

Den von uns entwickelte Ansatz, um Agilität im Unternehmen zu verankern, konnten wir inzwischen in mehreren Unternehmen erfolgreich anwenden und weiter ausarbeiten. Ein Unternehmen insgesamt so umzustellen, funktioniert nur schrittweise und bleibt damit immer vor allem Projektsache.

Agilität sollte aber nicht nur für das Entwicklungsteam eines Projekts gefordert und etabliert werden! Betrachten wir ein gesamtes Projekt von der Projektidee bis zum Einsatz des Produktes, sind die Entwicklungs-Prozesse und die Rückkopplungs-Zyklen innerhalb der Entwicklung vergleichsweise kurz. Problematisch sind vor allem umhüllende Wasserfall-artige Prozesse von der Anforderungsanalyse und -definition zu Beginn bis hin zur oft aufwändigen Abnahme am Ende des Projekts.

Hier setzen wir an und versuchen kontinuierlich, bewährte agile Techniken und Kriterien aus dem Entwicklungsprozess auf diese umhüllenden Prozesse zu übertragen und bei Bedarf funktionierende Analogien zu finden. Der Vortrag beschreibt unseren aktuellen Stand.

Literaturverzeichnis

[KochMiddeke] Jörn Koch, Sebastian Middeke: "Transparenz und Qualitätskontrolle in agilen Projekten: Agile Selbstheilungsmechanismen". In: Javamagazin, 7/2011. Seiten 108-113. http://it-republik.de/jaxenter/java-magazin-ausgaben/Java-7-000453.html

[KochSauer] Jörn Koch, Joachim Sauer: "A Task-Driven Approach on Agile Knowledge Transfer". In: Agility Across Time and Space, ISBN 978-3-642-12441-9. Springer-Verlag, Berlin Heidelberg, 2010. Seiten 311-319. http://www.springerlink.com/content/q4q22061184266x1/

Agile Software- Entwicklung in regulierten Umgebungen

Dr. Claudia Hirschmann

Corporate Technology
Siemens AG
Wladimirstr. 3
91058 Erlangen
Claudia.Hirschmann@siemens.com

Abstract: Das Manifest für Agile Softwareentwicklung schätzt u. a. Individuen und Interaktionen höher ein als Prozesse und Werkzeuge oder stellt das Reagieren auf Veränderungen über das strikte Befolgen eines Plans. Dies scheint konträr zu der Welt regulierter Branchen, wie z.B. der Medizintechnik zu stehen, die stark durch Normen und Gesetze reguliert wird. Solche Vorschriften richten sich beispielsweise an das Qualitätsmanagementsystem, die Produkt-Sicherheit oder bis hinein an einzelne Prozess-Schritte in der Software Entwicklung. Der Vortrag wird den Balanceakt beleuchten, wie agile Methoden nahtlos in regulierten Umgebungen eingesetzt werden können. Ferner werden Lösungsmethoden dazu aus der Praxis gezeigt, wie der Umgang mit Meilensteinen, pragmatische Möglichkeiten von Review- Protokollen oder geeignete Änderungsverfahren.

1. Einführung

Agile Methoden zeichnen sich durch das Manifest für Agile Softwareentwicklung [BB01a] und den Prinzipien hinter dem Agilen Manifest [BB01b] aus. Darin werden unter anderem Individuen und Interaktionen über Prozesse und Werkzeuge gestellt und funktionierende Software über umfassende Dokumentation; eine frühe und kontinuierliche Auslieferung von Software an den Kunden wird als höchste Priorität angesehen. Dies scheint in Konflikt zu einer Anwendung agiler Methoden in regulierten Umgebungen zu stehen, in welchen die Einhaltung von Entwicklungsprozessen sowie eine angemessene Dokumentation zum Nachweis der Befolgung der Vorgaben durchweg gefordert werden. Obendrein dürfen i.d.R. nicht vorbehaltlos Software-Inkremente an den Kunden ausgeliefert werden.

2. Charakteristik regulierter Umgebungen

Branchen wie die Medizintechnik oder die Bahntechnik sind durch Gesetze, Normen, Standards und weitere Vorgaben reguliert, um in erster Linie im Sinne der Produktsicherheit Personen vor möglichen Schäden zu bewahren. Die Regularien beziehen sich auf das Qualitätsmanagementsystem, das die Vorgehensweisen in der jeweiligen Produktentwicklung in dokumentierten Prozessbeschreibungen festlegen

muss. Darüber hinaus ist vorgeschrieben, wie das entwickelte Produkt in den jeweiligen Markt gebracht darf. Normen und Standards bestimmen recht genau, wie der Entwicklungsprozess ablaufen soll und welche Dokumentationen, Überprüfungen in Form von Reviews und Tests von wem und wann erstellt werden müssen und welche Rückverfolgbarkeiten nachzuweisen sind (siehe IEC/ISO 62304:2006 Medical device software - Software life cycle processes [IE06], ISO 14971:2007 Medical devices - Application of risk management to medical devices [IS07] oder EN 50126-1:1999 Bahnanwendungen – Spezifikation und Nachweis der Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit, Sicherheit, RAMS [EN99]).

3. Balanceakt: Agile Methoden in regulierten Umgebungen

Auf den ersten Blick mag es nun so aussehen, als ob agile Methoden in regulierten Umgebungen nicht einsetzbar wären: Es geht allerdings doch, wenn man sich an die Kernelemente agiler Methoden, nämlich an das Manifest für Agile Softwareentwicklung und die Prinzipien dahinter als die wesentliche Basis hält und die agilen Methoden, z.B. SCRUM oder XP, derart modelliert, dass die Vorgaben aus den Regularien angemessen umgesetzt werden. Ein Beispiel dafür wäre ein Softwareentwicklungs- Prozess, in dem iterativ- inkrementelle Zyklen in verschiedenen Prozessphasen eingesetzt werden, die mit ausgewählten Meilensteinen abgeschlossen und an die nächste Prozessphase übergeben werden. Des Weiteren können Review- Nachweise durch einfach-handhabbare Formulare und elektronisch unterstützte Unterschriften in typische agile Methoden wie Pair Programming oder Sprint Reviews integriert werden. Änderungsverfahren können durch gut- durchdachte Markierungen vom Einreicher des Änderungsantrags zu Thema, Priorität, Entscheiderkreis, Sicherheitsrelevanz, etc. in entsprechend zusammengesetzten Daily Scrum Meetings schneller bearbeitet werden.

4. Zusammenfassung

Wenn man die Rahmenbedingungen der regulierten Umgebung beherrscht und die agilen Kernbotschaften verstanden hat, zeigt die Erfahrung, dass gute Wege der Agilen Software- Entwicklung in regulierten Umgebungen gefunden werden können.

Literaturverzeichnis

[BB01a] Beck, K.; Beedle, M.; et.al.: Manifesto for Agile Software Development, 2001
[BB01b] Beck, K.; Beedle, M.; et.al.: Principles behind the Agile Manifesto, 2001
[DB10] Deemer, P.; Benefiled, G.; Larman, C.; Vodde, B.: The Scrum Primer, version 1.2, 2010
[SS11] Schwaber, K.; Sutherland, J: The Scrum Guide, The Definitive Guide to Scrum: The Rules of the Game, 2011
[IE06] IEC/ISO 62304:2006 Medical device software - Software life cycle processes
[IS07] ISO 14971:2007 Medical devices - Application of risk management to medical devices
[EN99] EN 50126-1:1999 Bahnanwendungen – Spezifikation und Nachweis der Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit, Sicherheit, RAMS

SIEMENS

Agile Software- Entwicklung in regulierten Umgebungen

Agile Software Development in Regulated Environments

27.Feb.- 1.March 2013

Copyright © Siemens AG 2013. All rights reserved.

SIEMENS

Überblick

1 Introduction

2 Software Development in Regulated Environments

3 Agile Development Methods

4 Agile Methods in Regulated Environments

5 Conclusion

© Siemens AG 2013. All rights reserved.
Siemens AG, Corporate Technology

Introduction
Siemens AG, Corporate Technology

SIEMENS

Development Processes,
Assessments & Models

SEI Partner | CarnegieMellon

LAI MIT ENABLING ENTERPRISE EXCELLENCE

Partner von
Method Park Software AG

© Siemens AG 2013. All rights reserved.
Siemens AG, Corporate Technology

Introduction
Software Development in Regulated Environments

SIEMENS

FDA, MDD, ...

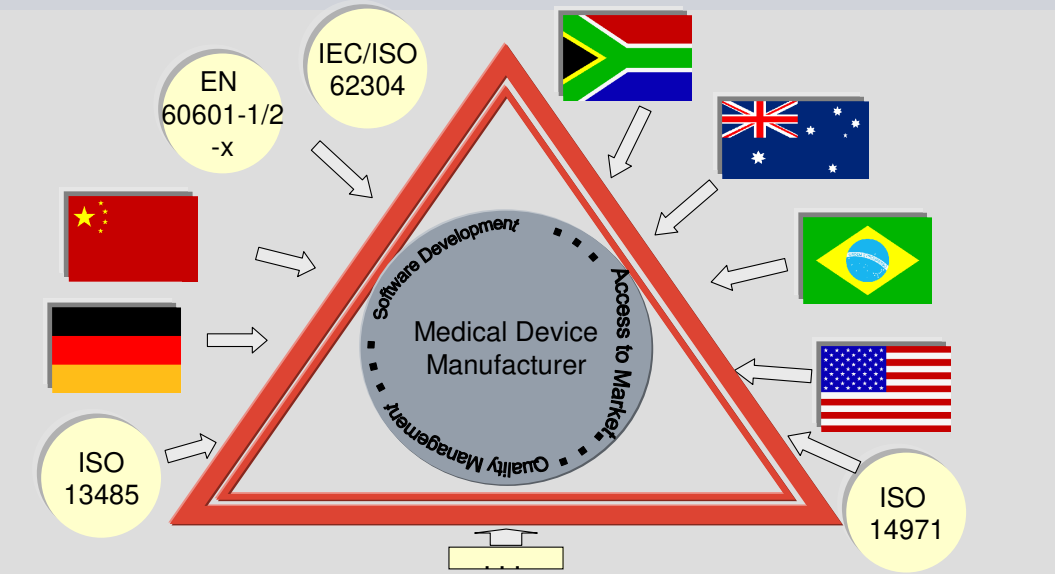
CENELEC, §§, ...

Texas, USA, April 1986: a patient with skin cancer is treated with radiation therapy using a linear accelerator. The patient's treatment is terminated with the cryptic error message '**malfunction 54**'. The patient dies 20 days later due to **radiation overdose**. To keep citizens from experiencing such conditions, the US passed a series of laws.

© Siemens AG 2013. All rights reserved.

Software Development in Regulated Environments
Example: Medical Device Development

SIEMENS

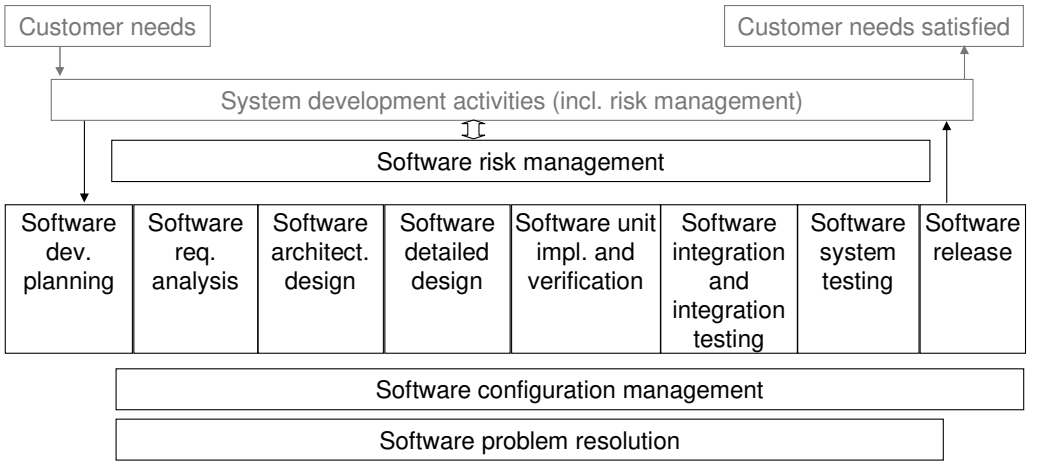


© Siemens AG 2013. All rights reserved.

Software Development in Regulated Environments
Example: Medical Device Development

SIEMENS

Software development processes and activities



according to IEC 62304:2006

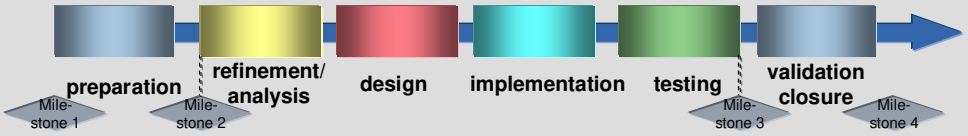
© Siemens AG 2013. All rights reserved.

SIEMENS

Software Development in Regulated Environments

Traditional Software Development in regulated environments, like, e.g., in medical device development:

- **Process of step-wise refinement** of the requirements over coding, … to tests.
- Usually modeled in a "Waterfall-" or "V Model-" Process



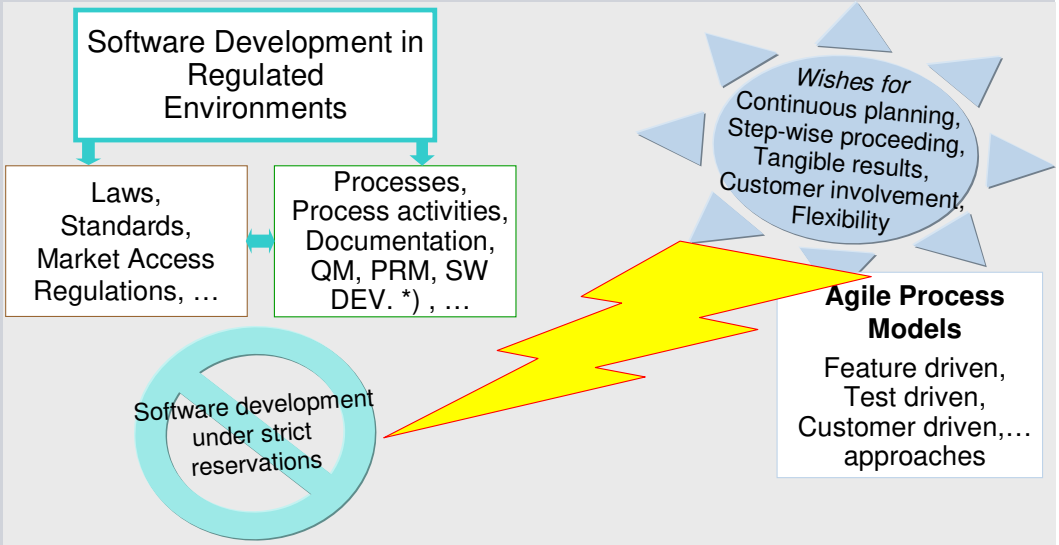
Control of fulfillment of regulations, quality, time, and budget:
via milestones, reviews, split of responsibilities, documentation, etc..

- Adequate for complex projects with stable product definition and development with known technology
- Not adequate for projects with instable requirements, innovative technologies and high risks
- Long period until the product is integrated and tested for the first time
- Project progress can only be measured by documents

© Siemens AG 2013. All rights reserved.

SIEMENS

Software Development in Regulated Environments



*) QM Quality Management, PRM Product Risk Management, SW Dev. Software Development Activities

© Siemens AG 2013. All rights reserved.

It began by 2001 through the Agile Manifesto and the Principles behind the Agile Manifesto

SIEMENS

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Principles behind the Agile Manifesto

We follow these principles:

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome …

…

Continuous attention to technical excellence and good design enhances agility.

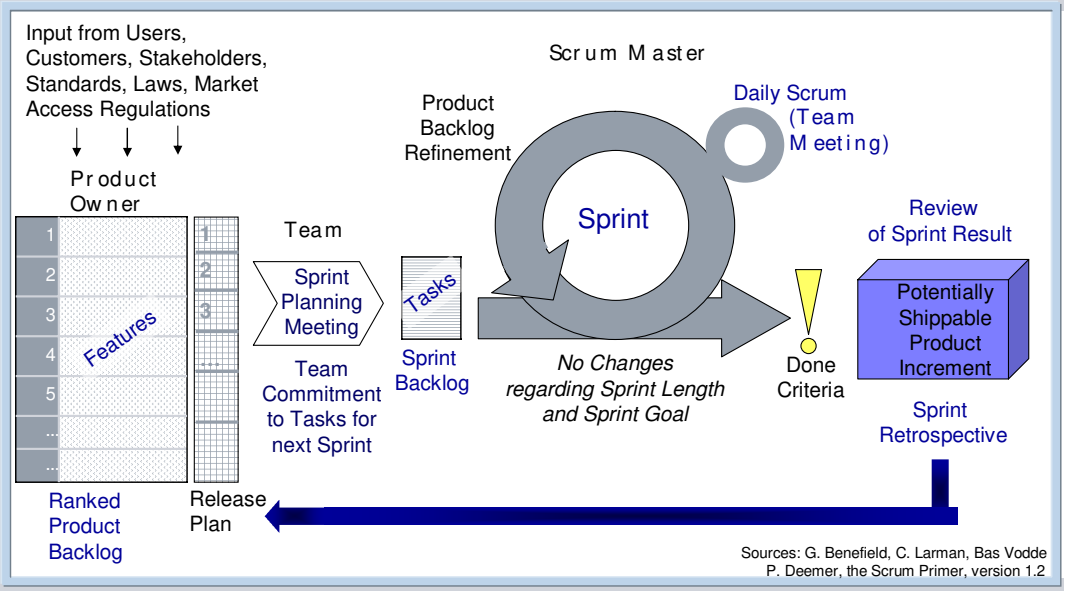
Simplicity...

http://www.agilemanifesto.org/

© Siemens AG 2013. All rights reserved.

Agile Development Methods One of the most important agile approaches is SCRUM

SIEMENS



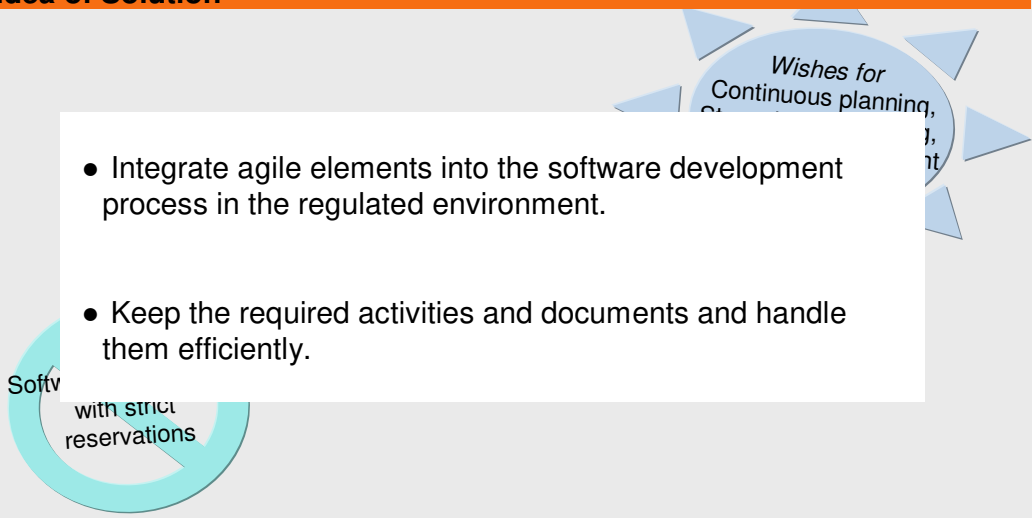
Sources: G. Benefield, C. Larman, Bas Vodde P. Deemer, the Scrum Primer, version 1.2

© Siemens AG 2013. All rights reserved.

Agile Methods in Regulated Environments: Idea of Solution

SIEMENS

Idea of Solution



- Integrate agile elements into the software development process in the regulated environment.
- Keep the required activities and documents and handle them efficiently.

© Siemens AG 2013. All rights reserved.

Agile Methods in Regulated Environments

SIEMENS

Requirements to the ➢ Quality Management System	▪ Documentation of the development process ▪ Requirements regarding roles, assignment, split of roles ▪ Change management requirements, . . .
➢ Software Development ➢ Product Risk Management	▪ Documentation during development: planning, design input / specifications, design, test, tracing ▪ Documentation of Reviews ▪ . . .
➢ Market Access ➢ . . .	▪ Specific work results: specifications, test protocols, review minutes, ▪ Approvals, approval periods, …

© Siemens AG 2013. All rights reserved.

SIEMENS

Agile Methods in Regulated Environments

Requirements to the
- Quality Management System

- Process:
  - Integration of iterations and synchronization points in the existing development process
- Roles:
  - Add : Scrum Master,
  - Product Owner is taken typically by Product Management
  - Keep Risk Manager, Safety Manager, Assessor
- Change management
  - Keep the change management process,
  - Add regular decision point in synchronization with the agile iterations

© Siemens AG 2013. All rights reserved.

SIEMENS

Agile Methods in Regulated Environments

- Software Development
- Product Risk Management

- Product Backlog:
  - Have a valid product backlog at implementation start
  - Have a valid product backlog at implementation end
- Specifications:
  - Keep your design input specifications in an adequate level of detail.
  - Care for tracing of requirements in each iteration.
- Reviews:
  - Define the Definition of Done together with Quality Management, Risk Management, Safety Manager
  - Document the Check of the Definition of Done during the Sprint Review

© Siemens AG 2013. All rights reserved.

SIEMENS

Agile Methods in Regulated Environments

- Product Backlog & Release Plan:
  - Include required work results for Market Access in the Product Backlog
  - Plan Market Access, approval periods, … in the Release Plan in close cooperation with Quality department / Regulatory Affairs department

- Market Access
- . . .

© Siemens AG 2013. All rights reserved.

SIEMENS

Conclusion

Experience shows:

- Design input is an fundamental topic!
- Updates of plans… are essential!
- Discipline is indispensable!

Agile Development in Regulated Environments is possible,
but it must be handled with care !

© Siemens AG 2013. All rights reserved.

SIEMENS

Agile Software Development in Regulated Environments

Dr. Claudia Hirschmann,
Siemens AG, Corporate Technology
Claudia.Hirschmann@siemens.com

Copyright © Siemens AG 2013. All rights reserved.
Siemens AG, Corporate Technology

5. Standards & Rechtliches

Sessionüberblick

5.1. Register Factory - Standards für die Cloud (27.02.2013) 116
5.2. SAGA im Werkvertrag (27.02.2013) 129
5.3. Ausschreibung von Software mit offener Lizenzierung für eine Institution der öffentlichen Finanzkontrolle in Österreich (27.02.2013) 141

Register Factory – Standards für die Cloud

(Extended Abstract)

Ralf Leonhard, Gerhard Pews, Simon Spielmann

Bundesverwaltungsamt
Barbarastraße 1,50735 Köln
Capgemini
Berliner Str 76, 63065 Offenbach
Ralf.Leonhard@bva.bund.de
Gerhard.Pews@capgemini.com
Simon.Spielmann@capgemini.com

Abstract: Die Register Factory wird seit 2007 als Standard für große IT-Projekte beim Bundesverwaltungsamt genutzt. Dieser Artikel stellt vor, welche Ausprägungen einer Cloud für den öffentlichen Bereich besonders interessant sind und welche Standards der Register Factory geeignet sind, eine solche Cloud zu etablieren.

1 Einsatzmöglichkeiten im öffentlichen Bereich

Typischerweise werden Clouds nach den *Service Models* und *Deployment Models* des NIST [NIST11] klassifiziert. Für die Beurteilung über einen Einsatz von Cloud-Techonolgien ist es zusätzlich sinnvoll, nach Anwendungsbereichen zu unterscheiden. Diese werden im Folgenden vorgestellt und dabei die Nutzbarkeit für den öffentlichen Bereich bewertet.

- Gemeinsame, geräteunabhängige Nutzung von Daten: Beispiele hierfür sind Apples *iCloud* oder Amazons *Cloud Drive*. Ein Anwender hält seine Daten nicht mehr auf einem lokalen Gerät, sondern in einem Rechenzentrum und kann diese von verschiedenen Geräten aus zugreifen.

  Dieser Bereich ist vornehmlich für Privatanwender interessant, da im professionellen Bereich Rechenzentrumsbetrieb bereits etabliert ist.

- Virtualisiertes Betriebssystem. Beispiele dafür sind Microsofts *azure* oder Amazons *EC2*. Hier wird im Wesentlichen ein virtueller Rechner in einem Rechenzentrum bereitgestellt. Dazu ist anzumerken, dass diese Angebote in die nächste Kategorie (virtualiserte Betriebsplattform) hineinragen.

Virtualisierte Betriebssysteme werden bereits vielerorts eingesetzt, allerdings sind sie eher ein Thema für die interne Organisation des Systembetriebs.

- Virtualisierte Betriebsplattform: Ein Beispiel dafür ist vmwares *vmfabric*. Hier werden nicht nur virtualisierte Rechner, sondern auch eine Betriebsplattform mit Application Server und Datenbanken und sonstiger Infrastruktur bereit gestellt.

  Dies ist sehr interessant für den Bereich der individuell entwickelten Verfahren. Hier bietet sich die Möglichkeit, Anwendungen einheitlich zu betreiben, sofern diese gewisse Standards bei der Nutzung der Betriebsplattform einhalten.

- Services und Entwicklungsumgebungen: Ein Beispiel dafür ist *force.com* bzw. *salesforce.com*. Hier kommen zu der Betriebsplattform auch eine Web-Entwicklungsumgebung und fachliche Lösungsbausteine und Frameworks dazu.

  Diese Möglichkeit erscheint sehr attraktiv, beinhaltet allerdings eine starke Bindung an einen Produkthersteller und läuft damit entgegen den Bestrebungen zur Verwendung offener Standards, so wie sie auch in SAGA festgeschrieben wurden.

- Virutalisierte Standardanwendungen: Beispiele dafür sind Google *Apps for Business*. Hier werden typische Office-Anwendungen wie Textverarbeitung, Email oder Teamkalender betriebsfertig angeboten.

  Dies kann ein nützliches Szenario sein, allerdings ist es fraglich, ob diese Anwendungen im Funktionsumfang mit etablierten Office-Paketen mithalten können. Der Platzhirsch Microsoft Office konnte in den letzten Jahren im öffentlichen Bereich nur in vergleichsweise geringem Umfang durch Open Office ersetzt werden. Ob eine solche Ablösung durch eine virtualisierte Standardanwendung möglich ist, ist daher offen.

Ein Bereich, der damit für den öffentlichen Sektor interessant ist, ist der Bereich der Virtualisierten Betriebsplattformen. Dies wird in folgenden Abschnitt detailliert.

2 Virtualisierte Betriebsplattformen für den öffentlichen Bereich

Im öffentlichen Bereich ist es möglich, dass in einer Behörde eine virtualisierte Betriebsplattform bereit gestellt wird. Diese Plattform ethielte dann Application Server und Datenbank Server in einer BSI-konformen Zonenarchitektur. Weiterhin könnten man dort die folgenden querschnittlichen Dienste bereitstellen:

- Authentifizierung und Autorisierung
- Datenkataloge, z. B. Staatenlisten

- Protokollierung und Protokollrecherche
- Dokumentenerstellung und Druck
- Email-Versand
- Verschlüsselung von Datenverbindungen
- Virenprüfung
- ggf. domänenspezifische Dienste, wie z. B. eine alphanumerische Suche oder eine biometrische Suche.

Das Bereitstellen einer solchen Plattform wäre eine praktikable Schnittstelle zwischen dem Systembetrieb und der Systementwicklung und würde es ermöglichen, solche Systeme günstiger zu betreiben und Ressourcen effizient zu nutzen. Damit dies allerdings möglich ist, müssen die Anwendungen sich an bestimmte Standards halten, die einen solchen Betrieb ermöglichen.

3 Standards der Register Factory

Generell betreffen die Standards vier Bereiche:

- Softwaretechnische Stuktur der Anwendung: Aufteilung in technische Schichten und Fachkomponenten, Zustandshaltung in der Anwendung, genutzter Technologiestack.
- Schnittstelle zum Betrieb: Konfiguration, Überwachung und Steuerung der Anwendung
- Schnittstelle zum Anwender/Nachbarsystemen: Technologien für GUI, Batches und Services
- Einbettung in Infrastruktur: Schnittstellen zur Authentifizierung und Autorisierung

Die Register Factory umfasst Standards für diese Bereiche. Mit der Register Factory ist es schon jetzt beim Bundesversaltungsamt schon jetzt möglich, eine Vielzahl von Anwendungen aus der Domäne der inneren Sicherheit auf einer solchen gemeinsamen Plattform zu betreiben.

Literaturverzeichnis

[BK09] BITKOM-Leitfaden: Cloud Computing – Evolution in der Technik, Revolution im Business, http://www.bitkom.org/de/publikationen/38337_61111.aspx

[PLB10] Gerhard Pews, Ralf Leonhard, Stefan Brombach: *Ein Standard für Register-anwendungen: die Register Factory,* Proceedings SEE 2010

[NIST11] Peter Mell, Timothy Grance: The NIST Definition of Cloud Computing, NIST Special Publication 800-145, 2011, http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

Bundesverwaltungsamt

Register Factory – Standards für die Cloud

SEE-Conference 2013

Ralf Leonhard, Gerhard Pews, Simon Spielmann

Ralf.Leonhard@bva.bund.de
Gerhard.Pews@capgemini.com
Simon.Spielmann@capgemini.com

Bundesverwaltungsamt
Der zentrale Dienstleister des Bundes

Bundesverwaltungsamt

Inhalt

- **Motivation**
- Cloud-Modelle
- Register Factory und Cloud
- Standards der Register Factory
- Zusammenfassung und Ausblick

Bundesverwaltungsamt
Der zentrale Dienstleister des Bundes

Bundesverwaltungsamt

Motivation

- Potential für eine Enterprise Cloud für den öffentlichen Bereich
  - Viele Spezialanwendungen
  - Immer mehr Verwaltungsvorgängen werden durch IT unterstützt.
  - Immer schneller und kostengünstiger mit geringeren Ressourcen zu entwickeln
- Aber: „Cloud“ ist unscharf „cloudy“
  - ➔ Welche Typen einer Cloud sind im öffentlichen Bereich interessant?
  - ➔ Wie kann man eine solche Cloud umsetzen?

Bundesverwaltungsamt

Inhalt

- **Motivation**
- Cloud-Modelle
- Register Factory und Cloud
- Standards der Register Factory
- Zusammenfassung und Ausblick

Bundesverwaltungsamt

Verfeinerung der NIST-Kategorisierung mit dem Fokus auf Enterprise Cloud für den öffentlichen Bereich

Modell	Beispiele
Geräteunabhängige Nutzung von Daten	▪ Daten nicht auf lokaler Platte, sondern im Rechenzentrum von überall zugreifbar ▪ Apple iCloud, Amazon cloud drive
Virtualisiertes Betriebssystem	▪ Der Rechner im Netz ▪ Microsoft azure, Amazon elastic compute cloud EC2
Virtualisierte Betriebsplattform	▪ Infrastruktur und Betriebsplattform (DB, AppServer, etc.) im Netz ▪ Vmware vmfabric
Services + Entwicklungs-umgebungen	▪ Entwicklungsumgebung und Ablaufumgebung, fachliche Bausteine ▪ force.com, salesforce.com
Virtualisierte Anwendungen	▪ Komplette Anwendungen ▪ Google Apps for Business

Bundesverwaltungsamt
Der zentrale Dienstleister des Bundes

Seite: 5

Bundesverwaltungsamt

Bewertung im Blick auf Einsatz im öffentlichen Bereich

Modell	Bewertung
Geräteunabhängige Nutzung von Daten	▪ Eher für den Privatanwender interessant, im professionellen Bereich ist RZ-Betrieb in der Regel etabliert
Virtualisiertes Betriebssystem	▪ Virtualisierung im Systembetrieb vielerorts eingesetzt und etabliert. ▪ Eher ein Diskussionspunkt in Betriebsabteilungen
Virtualisierte Betriebsplattform	▪ Der Systembetrieb stellt höherwertige Infrastruktur (AppServer, etc.) bereit, durch die der Betrieb von Anwendungen vereinheitlicht wird. ▪ Sehr interessant für den Bereich individuell entwickelter Verfahren.
Services + Entwicklungs-umgebungen	▪ Attraktiv, aber starke Bindung an Produkthersteller ▪ Konträr zu Bestrebungen zur Verwendung offener Standards wie in SAGA beschrieben
Virtualisierte Anwendungen	▪ Potential vor allem für Standardanwendungen ▪ Offen, ob es sich etablieren kann, vgl. Ablösung MS Office durch Open Office

Bundesverwaltungsamt
Der zentrale Dienstleister des Bundes

Seite: 6

Bundesverwaltungsamt

Inhalt

- **Motivation**
- Cloud-Modelle
- Register Factory und Cloud
- Standards der Register Factory
- Zusammenfassung und Ausblick

Bundesverwaltungsamt

Erfahrungsbericht

- Mit der Register Factory umfangreiche Erfahrungen für standardisierten Bau von Individuallösungen gesammelt
- Erfahrungen gesammelt, wie man eine Betriebsplattform, Bausteine und Services gemeinsam nutzen kann
- Potential für den Aufbau einer Cloud auf Plattformebene im öffentlichen Bereich.

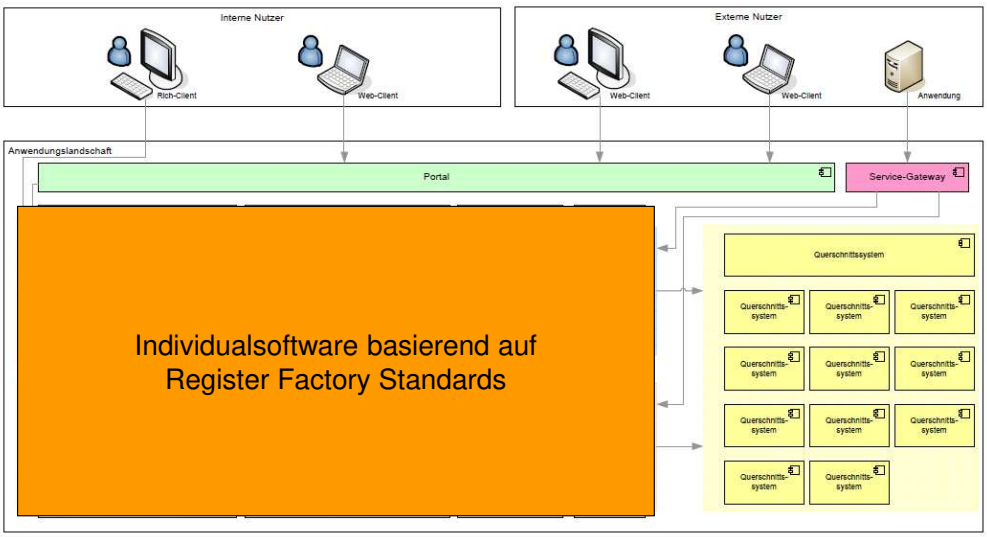
Bundesverwaltungsamt

Erfahrungen beim standardisierten Bau von Systemen

- Primäre Domäne: Öffentliche Sicherheit beim Bundesverwaltungsamt.
- Seit 2007, ca. 15 Systeme verschiedener Größe, auch sehr große Systeme (100Mio Datensätze, 24/7, 80.000 Nutzer)
- Register, Fachverfahren, behördenübergreifender Datenaustausch
- Web- und native GUI

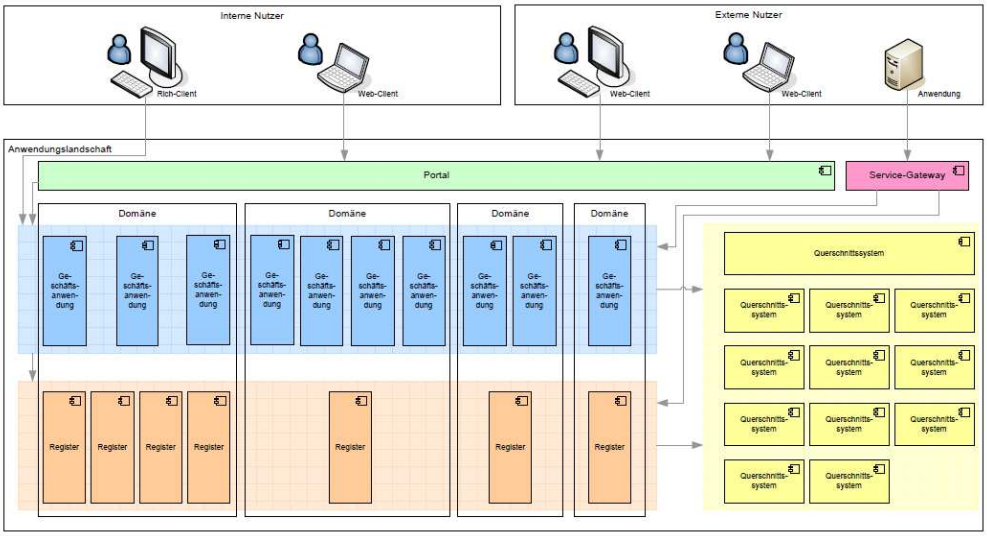
Bundesverwaltungsamt

Betriebsplattform auf Basis der Register Factory



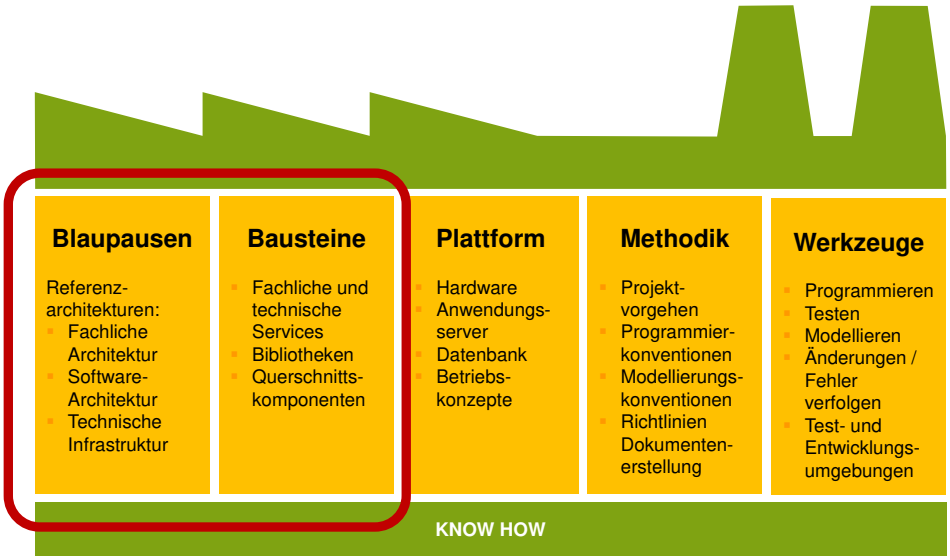
Bundesverwaltungsamt

Konkret im BVA in Betrieb



Bundesverwaltungsamt

Standardisierung durch die Vorgaben der Register Factory



Bundesverwaltungsamt

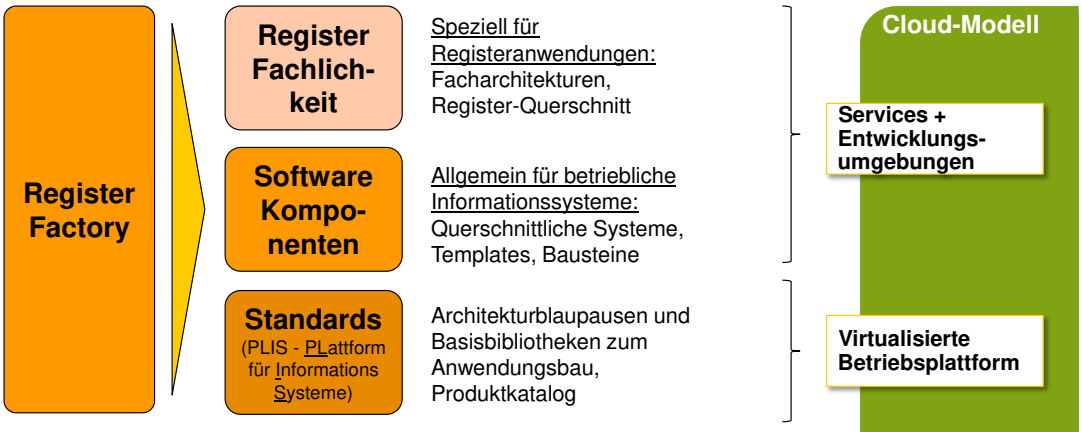
Inhalt

- **Motivation**
- Cloud-Modelle
- Register Factory und Cloud
- Standards der Register Factory
- Zusammenfassung und Ausblick

Bundesverwaltungsamt

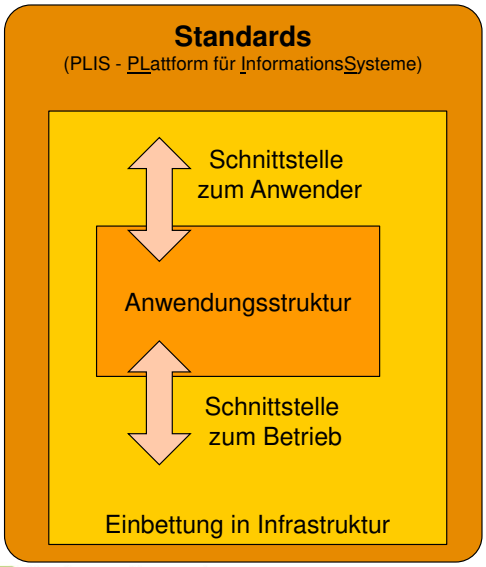
Für eine Cloud-Plattform einsetzbare Standards

Die Blaupausen und Bausteine der Register Factory können in unterschiedliche Inhaltsbereiche zerlegt werden.



Bundesverwaltungsamt

Standardisierung



Die Standardisierung umfasst drei große Bereiche:

- Schnittstelle zum Anwender/Nachbarsystemen
  - Technologie für GUI
  - Technologie für Batches
  - Technologie für Services
- Struktur der Anwendung
  - Aufteilung in technische Schichten und Fachkomponenten
  - Genutzter Technologiestack
- Schnittstelle zum Betrieb
  - Konfiguration, Überwachung und Steuerung der Anwendung
- Einbettung in Infrastruktur
  - Authentifizierung/Autorisierung

Bundesverwaltungsamt
Der zentrale Dienstleister des Bundes

Seite: 15

Bundesverwaltungsamt

Standards im Detail

Standard	Konzepte	Datenstrukturen, Schnittstellen (Code-Packages)
Struktur der Anwendung	• Referenzarchitektur • Detailkonzept Anwendungskern, Detailkonzept Datenzugriff • Konzept Umgang mit Sonderzeichen • Konzept Fehlerbehandlung • Produktkatalog • Namenskonventionen, Programmierkonventionen	• plis-aufrufkontext • plis-exception • plis-nummernkreis • plis-persistence • plis-sonderzeichen • plis-util
Schnittstelle zum Betrieb	• Konzept Überwachung/Konfiguration • Konzept Logging	• plis-konfiguration • plis-logging • plis-ueberwachung,
Schnittstelle zum Anwender	• Detailkonzept Batch • Detailkonzept GUI • Detailkonzept Service • Konzept Servicekommunikation	• plis-batchrahmen • plis-nativegui, plis-nativegui-adapter • plis-web • plis-serviceapi, plis-sgw-utilities
Einbettung in Infrastruktur	• Nutzerdokumentation Sicherheit • Rollen-Rechte-Schema	• plis-anwenderdaten • plis-sicherheit

PLIS - PLattform für InformationsSysteme

Seite: 16

Bundesverwaltungsamt

Inhalt

- **Motivation**
- Cloud-Modelle
- Register Factory und Cloud
- Standards der Register Factory
- Zusammenfassung und Ausblick

Bundesverwaltungsamt

Fazit und Ausblick

- Standards der Register Factory
  - Ursprünglich zur Homogenisierung der Anwendungslandschaft und Effizienzsteigerung entwickelt
  - Nutzbar auch zur Bereitstellung einer Anwendungsplattform in einer Cloud
- Standards haben sich bewährt, Grundlagen sind geschaffen
- Ausblick
  - Planung einer stärker virtualisierten Plattform für Fachverfahren

SAGA im Werkvertrag

Bernd Weber

Bereich Öffentliche Verwaltung
4Soft GmbH
Mittererstr. 3
80336 München
bernd.weber@4soft.de

Abstract: Seit November 2011 muss die Bundesverwaltung das Rahmenwerk für IT-Standards „SAGA“ anwenden. Der Vortrag beleuchtet die Einbeziehung von SAGA-Kriterien in Werkverträge, vorvertragliche und Folgedokumente auf Auftraggeber- und Auftragnehmerseite. Dabei wird der Zusammenhang zwischen SAGA-Zielen, Konformitätskriterien und technischen Spezifikationen verdeutlicht.

1. Motivation

Seit November 2011 muss die Bundesverwaltung das Rahmenwerk für IT-Standards „SAGA“ anwenden. SAGA ist seither eine verbindliche Randbedingung von Ausschreibungen von IT-Leistungen, entsprechenden Verträgen und Abnahmen. Allerdings ist für viele Beteiligte auf beiden Seiten noch nicht transparent, welche konkreten Auswirkungen SAGA auf Vertragsanbahnung, Vertragsschluss und Lieferung hat.

2. Ansatz

Die SAGA-Module und begleitenden Dokumente gehen an verschiedenen Stellen auf vertraglich relevante Aspekte ein. Diese Stellen sollen zusammengeführt und hinsichtlich ihrer Bedeutung für die Auftraggeber- und die Auftragnehmerseite erläutert werden. Dabei soll der Zusammenhang zwischen SAGA-Zielen, Konformitätskriterien und technischen Spezifikationen verdeutlicht werden.

3. Bewertung

Der Vortrag beleuchtet auf der Auftraggeberseite die Einbeziehung von SAGA-Kriterien in Verdingungsunterlagen, in die Angebotsprüfung und in Abnahmeprotokolle und auf der Auftragnehmerseite deren Einbeziehung in Angebote und Konformitätserklärungen. Anhand von Beispielen wird der Einsatz von SAGA-Kriterien über die verschiedenen Projektphasen hinweg demonstriert.

4SOFT
Solid Innovation

SAGA im Werkvertrag

SEE 2013
27.02. – 01.03.2013
RWTH Aachen

Was ist SAGA?

4SOFT

„**SAGA** ist eine Zusammenstellung von Referenzen auf **Spezifikationen** und Methoden für **Software-Systeme** der öffentlichen Verwaltung.“ SAGA-Modul Grundlagen Version de.bund 5.1.0, Kap. 1 Satz 1

Rahmeninformationen

- Beschluss des IT-Rats vom 03.11.2011 zu SAGA Version de.bund 5.x
- Verbindliche Anwendung für die Bundesverwaltung
- Derzeit 3 unabhängige Module
  - Grundlagen
  - Konformität
  - Technische Spezifikationen
- SAGA wird zum Eigennamen erklärt

SAGA

~~Standards und Architekturen für eGovernment-Anwendungen~~

Geltungsbereich und Zielgruppe

Geltungsbereich

- Beschaffung, Erstellung, Weiterentwicklung von Software-Systemen
- Keine Geltung für Konfiguration, Bugfixes oder Hardware/Embedded

Zielgruppe

- Entscheider, Projektleiter, Architekten, Entwickler **mit Verantwortung für Software-Systeme** der öffentlichen Verwaltung
- Auftraggeber (AG) und Auftragnehmer (AN)

Ziele

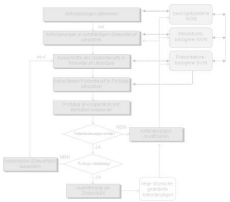
- Wirtschaftlichkeit
  - U.a. Kosten bei späterer Ablösung des SW-Systems
- Agilität
- Offenheit
  - Unabhängigkeit von Interessen einzelner Marktteilnehmer
  - Nachhaltige, transparente Basis für Verwaltung und Lieferanten
- Sicherheit
- Interoperabilität
  - (EU-weiter) Datenaustausch über ursprünglichen Bereich hinaus
  - Vermeiden von Einschränkungen
- Wiederverwendbarkeit
- Skalierbarkeit

Mindestanforderungen an die Offenheit

- Die Spezifikation wurde **vollständig publiziert** und die Publikation ist entweder **kostenfrei oder gegen ein angemessenes Entgelt erhältlich**.
- Die **Verwendung** der Spezifikation ist für Hersteller und Nutzer der Software-Systeme **uneingeschränkt und kostenfrei** möglich.
- Zum Zeitpunkt der Bewertung ist nicht erkennbar, dass die Spezifikation in der Zukunft die ersten zwei Anforderungen nicht mehr erfüllen wird.

Themengebiete technischer Spezifikationen

- IT-Sicherheitskonzeption
- Prozessmodelle
- Datenmodelle
- Applikationsarchitektur
- Client
- Präsentation
- Kommunikation
- Backend
- Verschlüsselung
- Elektronische Signatur
- Smartcards
- Langzeitspeicherung



Auszug aus Modul Technische Spezifikationen

4.2 Austauschformate für Datenmodelle

Verbindlich: XML Schema Definition Language (XSD) 1.0	PI

XSD[39] ist eine Spezifikation, die es erlaubt, Strukturen für XML-Dokumente zu beschreiben und damit Datenmodelle auszutauschen. Durch XML-Validatoren können XML-Dateien darauf geprüft werden, ob sie die durch eine XSD-Datei beschriebene Struktur einhalten.

XSD 1.0 wurde vom World Wide Web Consortium (W3C)[40] 2004 in einer Second Edition als Recommendation herausgegeben.

XSD *MUSS* für den Austausch von XML-Datenmodellen verwendet werden. Nur wenn die Funktionalitäten von XSD für die fachlichen Anforderungen des jeweiligen Anwendungsfalls nicht ausreichen, *SOLLTE* RelaxNG eingesetzt werden.

Empfohlen: Regular Language Description for XML New Generation (Relax NG)	PI

RELAX NG[41] ist eine Schema-Sprache für XML. Ursprünglich wurde RelaxNG in den Jahren 2001/2002 von der Organization for the Advancement of Structured Information Standards (OA-

Definition der SAGA-Konformität

4SOFT

- Die fachlichen Anforderungen stehen über den Klassifikationen von SAGA.
- Diese Klassifikationen dienen dazu, die fachlichen Anforderungen hinsichtlich der Ziele von SAGA optimal umzusetzen.
- **SAGA-Konformität wird deshalb durch den Einsatz der Teilmenge aller SAGA-Spezifikationen erreicht, die für das jeweilige Software-System fachlich sinnvoll sind.**

Quelle: SAGA-Modul Konformität Version de.bund 5.1.0, Unterkap. 2.2 Abs. 3

Klassifikation und Konformität

4SOFT

- Konformitätsprüfung durch Untergliederung
  - Externe Einheit (Produkt) „P“: Schnittstellen, Formate, Sicherheit
  - Software-Einheit (individuell) „I“: „P“ + Modellierung, Implementierung
- Konformität auch bei niedriger Klassifikation möglich
- Verantwortung für Konformität: Fachlich zuständiger AG

Klasse	Auswirkung
Vorgeschlagen	Einsatz nicht konform oder nicht relevant
Beobachtet	Einsatz konform ohne höherrangige Spezifikation
Empfohlen	Einsatz immer konform; Abweichungen begründen!
Verbindlich	Einsatz immer konform. Abweichung nicht möglich!
Bestandsgeschützt	Konform bei Änderung/Erweiterung. Alternativen prüfen!
Verworfen	Allein nie konform; Migrationspflicht bei Änderung!

SAGA-Konformität in Ausschreibung und Vertrag

- Keine pauschale Forderung nach SAGA-Konformität erlaubt
- => Aufnahme konkreter SAGA-Kriterien in Vergabeunterlagen

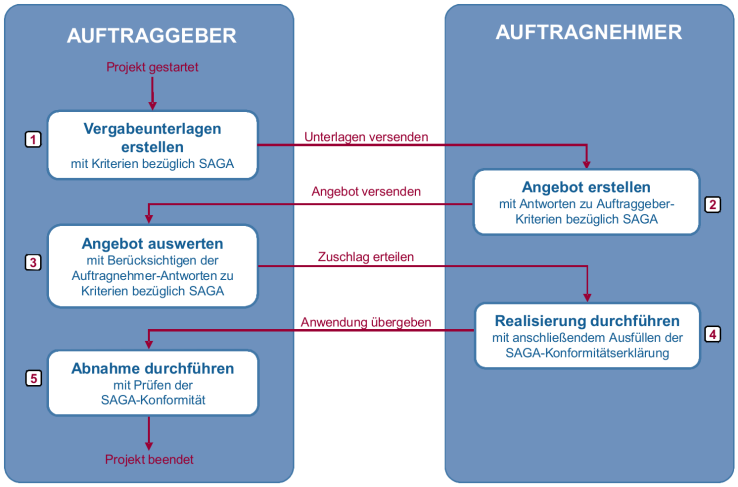


Abbildung 3-1: Prozess zur SAGA-Konformitätserklärung

Vorlagen für SAGA-Konformität

- cio.bund.de/saga -> SAGA-Konformität -> Downloads

Beispiel 1: JavaScript, Ausschreibung und Angebot

4SOFT

- Ausschreibung, Auszug aus Bewertungskriterien
  A 9: Die Nutzung von Javascript ist zugelassen, sofern die Webseiten auch dann verwendbar sind, wenn Javascript deaktiviert wurde (siehe SAGA-Modul „Technische Spezifikationen“ de.bund 5.0.0, Abschnitt 6.1.2 „Web-Browser“).
  Sichern Sie diese Anforderung zu (ja oder nein)?

- Angebot, Auszug Antwort auf A 9:
  Ja. Die Webseiten können auch bei abgeschaltetem JavaScript inhaltlich vollumfänglich verwendet werden.

- Auswertung der Antwort zu diesem Ausschlusskriterium in Bewertungsmatrix

Beispiel 1: JavaScript, Lieferung

4SOFT

SAGA-Konformitätserklärung

Für das Software-System

Name des Software-Systems, Leistungsbeschreibung im Anhang

sind folgende Einheiten identifiziert worden, für die der Auftragnehmer die SAGA-Konformität nach SAGA de.bund 5-0 anhand der beigefügten Checklisten erklärt.

Eigenentwickelte Einheiten

Eigenentwickelte Einheiten eines Software-Systems werden vom Auftragnehmer

Beispiel 1: JavaScript, Inhalt Konformitätserklärung

- Konformitätserklärung bei Lieferung, Auszug zu Kap. 6

Client (siehe SAGA-Modul „Technische Spezifikationen“ de.bund 5.0.0, Kapitel 6)

Relevanter Konformitätsaspekt	Anforderungen	Zusicherung	
		Ja	Nein
Einsatz aktiver Inhalte im Client	Es werden keine Active-X-Controls eingesetzt.	X	
Web-Browser	Cookies werden nur verwendet, solange sie nicht persistent sind und Webseiten einer Domain keine Inhalte anderer Domains inkludieren, welche Cookies setzen.	X	
	Bei der Verwendung von Javascript wird sichergestellt, dass die Webseiten auch dann verwendbar sind, wenn Javascript deaktiviert wurde.	X	

Beispiel 1: JavaScript, Abnahmeprüfung

- Prüfprotokoll Lieferung, Auszug zu SAGA-Anforderungen

Lieferung vom 01.03.2013 **Prüfprotokoll Anwendung ABC**

Nr.	Anforderung	Kriterium	Erwartetes Ergebnis	Tatsächliches Ergebnis	Mangelklasse	Bemer
1	A 9	Webseiten ohne JavaScript verwendbar	Alle Webseiten der Anwendung können fehlerfrei aufgerufen werden.			
2			Die zu Auswahlfeldern hinterlegten Listen sind korrekt befüllt.			

Beispiel 2: Textdokumente Ausschreibung und Angebot

- B 30: Einfache Textdokumente ohne Anforderungen an das Layout, die zum Informationsaustausch und Download angeboten werden, sollten im einfachen Textformat angeboten werden (siehe SAGA-Modul „Technische Spezifikationen“ de.bund 5.0.0, Abschnitt 7.7.1 „Formate für Textdokumente zum Informationsaustausch“).
  Sichern Sie diese Anforderung zu (ja oder nein)? Wenn nein, begründen Sie die Abweichung. Stellen Sie zusätzliche Formate zur Verfügung?

- Angebot, Auszug Antwort auf B 30:
  Das Software-System enthält keine Beschränkungen bezüglich der Dateitypen, die zum Download angeboten werden. Deshalb wird es möglich sein, einfache Dokumente im Textformat zur Verfügung zu stellen, aber auch beispielsweise PDF, RTF, ODF und OOXML sind möglich. Vom System generierte Dokumente, die nicht zur Weiterbearbeitung vorgesehen sind, werden stets im PDF-Format erstellt. So kann auch für einfache Texte die Einhaltung des Corporate Design Ihrer Behörde sichergestellt werden.

4Soft GmbH 17

Beispiel 2: Textdokumente, Bewertungsmatrix

Kriterium	Erfüllung	Bewertung
B 30	Textdokumente werden nicht im einfachen Textformat angeboten.	0
	Textdokumente werden im einfachen Textformat angeboten.	5
	Textdokumente werden im einfachen Textformat und weiteren sinnvollen Formaten angeboten.	10

4Soft GmbH 18

Beispiel 2: Textdokumente Konformitätserklärung und Abnahmeprüfung

- Konformitätserklärung, Auszug zur SAGA-Anforderung

Präsentation (siehe SAGA-Modul „Technische Spezifikationen“ de.bund 5.0.0, Kapitel 7)

Relevanter Konformitätsaspekt	Standard	SAGA-Konformität gegeben? Ja / Nein
Formate für Textdokumente zum Informationsaustausch	PDF 1.4	Ja
Formate für Textdokumente zum Informationsaustausch	Text	Ja

- Prüfprotokoll Lieferung analog zu Beispiel 1

4Soft GmbH 19

4SOFT

Solid Innovation

4Soft GmbH • Mittererstraße 3 • 80336 München
Tel.: (089) 53 07 44 44 • Fax: (089) 53 07 44 45
info@4soft.de • www.4soft.de

Ausschreibung von Software mit offener Lizenzierung für eine Institution der öffentlichen Finanzkontrolle in Österreich

Stefan Schiffer, Norbert Sterrer

Johannes Kepler Universität Linz
Institut für Wirtschaftsinformatik – Software Engineering

Oberösterreichischer Landesrechnungshof

stefan@schiffer.at
norbert.sterrer@lrh-ooe.at

Abstract: Der Vortrag erläutert eine praxisorientierte Herangehensweise für die Ausschreibung von Software mit offener Lizenzierung, mit der eine faire, nachvollziehbare und effiziente Auswahl des besten Angebots möglich ist. Der vorgestellte Ansatz wurde im Jahr 2012 in einem Vergabeverfahren einer Institution der öffentlichen Finanzkontrolle in Österreich angewendet, das u.a. die Beschaffung eines Audit Information Systems, eines Customer Relationship Management Systems und eines Document Management Systems zum Ziel hatte.

Die Ausschreibung von Open-Source-Software (OSS) ist für öffentliche Auftraggeber, die dem Vergaberecht unterliegen, mit der Ungewissheit verbunden, ob wegen besonderer Merkmale von OSS, die in die Ausschreibungskriterien einfließen, z.B. der Quelloffenheit oder spezieller OSS-Lizenzen, mit Einsprüchen von Anbietern proprietärer Software zu rechnen ist und durch ein diesbezügliches Verfahren bei der Vergabekontrollbehörde die Beschaffung verzögert oder gar die Zuschlagserteilung aufgehoben wird. Der Grundsatz des freien, lauteren und fairen Wettbewerbs und das Prinzip der Gleichbehandlung aller Bewerber und Bieter erlauben nämlich keine Bevorzugung von OSS gegenüber proprietärer Software aufgrund allgemeiner Erwägungen, beispielsweise aus förderungspolitischen Motiven oder anderer vergaberechtsfremder Kriterien, sondern fordern sachliche Gründe für alle Eignungs-, Auswahl- und Zuschlagskriterien sowie deren Bewertung. In [Be12] wird darauf hingewiesen, dass Ausschreibungen neutral erfolgen müssen, ohne Anbieter bestimmter Software-Typen von vorneherein auszuschließen. Zwar sollen sich Behörden nicht vom pauschalen Hinweis auf angebliche rechtliche Gefahren von einer Migration zum OSS abschrecken lassen, allerdings wäre eine abschließende Beurteilung vom Einzelfall abhängig. Erenli [Er07] vertritt die Ansicht, dass Forderungen nach einem bestimmten Lizenztyp vergaberechtlich ebenso unrechtmäßig wären, wie solche nach Nutzungs- bzw. Verwertungsrechten, die nur einzelne Lizenzen einräumen und auch das Verlangen nach Offenlegung des Quellcodes nur in begründeten Fällen zulässig sei.

Der Vortrag zeigt die vergaberechtskonforme Gestaltung und Abwicklung einer Ausschreibung zur Beschaffung einer Software-Lösung mit offener Lizenzierung für die Unterstützung der Prüfungsprozesse und administrativen Tätigkeiten einer Institution der öffentlichen Finanzkontrolle in Österreich.[1]

Um die verfassungsrechtlich gewährleistete Unabhängigkeit dieser Institution auch in Bezug auf den Hersteller der zu beschaffende Softwarelösung sicherzustellen, entschied man sich für die Ausschreibung einer Softwarelösung mit offener Lizenzierung, die dem Auftraggeber das Recht einräumt, die (quelloffene) Software zu analysieren, zu nutzen und zu verändern und sich dazu Dritter zu bedienen, ohne eine bestimmte, bestehende OSS-Lizenz vorzugeben. Vergabetechnische Herausforderungen dabei waren vor allem die Fokussierung auf einen geeigneten Kreis von Bewerbern in der ersten Stufe des Vergabeverfahrens, die grundsätzlich bereit und der Lage sind, eine solche Lizenz zu erteilen (obwohl dies in dieser Stufe nicht gefordert werden kann), die Festlegung von sachlich begründeten und einfach bewertbaren Zuschlagskriterien sowie eine teilautomatisierte Bewertung, um den dafür notwendigen Aufwand zu reduzieren und Fehler zu vermeiden, wofür entsprechende Formulare entwickelt wurden.

Bei der Vorbereitung der Ausschreibung wurde das V-Modell XT als Leitlinie für die Beschaffung festgelegt, was insofern eine Besonderheit darstellt, als dieses Vorgehensmodell in Österreich kaum verbreitet ist, obwohl es wegen seiner klaren Struktur und Begrifflichkeit gerade bei Ausschreibungen wertvolle Hilfestellung leisten kann und sich auch diesmal bewährt hat. Nach einer Geschäftsprozessanalyse und der darauf beruhenden Erstellung des Lastenheftes wurden mehrere am Markt befindliche Produkte evaluiert, um sicherzustellen, dass der aus budgetären und strategischen Gründen angestrebte Einsatz von Standardsoftware mit offener Lizenzierung und individuellen Anpassungen prinzipiell machbar ist. Besonderes Augenmerk wurde dabei auf die Teilsysteme *Audit Information System*, *Customer Relationship Management System* und *Document Management System* gelegt.

Da sowohl der Auftraggeber als auch die zentrale Beschaffungsstelle mit der speziell für diese Ausschreibung entwickelten Vorgehensweise streckenweise Neuland betraten, bestand anfänglich Unsicherheit darüber, ob die Ausschreibung in der geplanten Weise durchführbar sei. Es hat sich jedoch gezeigt, dass die festgelegten Kriterien und Verfahren für die Prüfung und Bewertung der Angebote eine faire, nachvollziehbare und effiziente Auswahl des besten Angebots ermöglichen.

Der unangefochten gebliebene Zuschlag wurde an den Bestbieter im Dezember 2012 erteilt.

[1] Das dafür gewählte Verhandlungsverfahren mit vorheriger Bekanntmachung beruht auf dem Bundesvergabegesetz 2006, mit dem Österreich die einschlägigen EU-Richtlinien umgesetzt hat, die auch für Deutschland verbindlich sind und sich dort im Gesetz gegen Wettbewerbsbeschränkungen (GWB), in der Vergabeverordnung (VgV) und weiteren Bestimmungen wiederfinden, z.B. der Vergabe- und Vertragsordnung für Leistungen (VOL), wo auch das Verhandlungsverfahren geregelt ist.

Literaturverzeichnis

[Be12] Die Beauftragte der Bundesregierung für Informationstechnik (Hrsg.): Rechtliche Aspekte der Nutzung, Verbreitung und Weiterentwicklung von Open-Source-Software, Bundesministerium des Innern, Berlin, 2012.

[Er07] Erenli, K.: Die rechtliche Relevanz von Open Source Lizenzen unter besonderer Berücksichtigung praktischer Problemstellungen, Dissertation, Karl-Franzens-Universität Graz, 2007

Ausschreibung von Software mit offener Lizenzierung

für eine Institution der öffentlichen Finanzkontrolle in Österreich

SEE2013

Aachen, 27.2.2013

Stefan Schiffer, Johannes Kepler Universität Linz
Norbert Sterrer, Oberösterreichischer Landesrechnungshof

Der Oö Landes-rechnungshof

- Unabhängige Institution der öffentlichen Finanzkontrolle
- Kontrollorgan des Oö Landtags
- Eingerichtet mit Landesgesetz seit 1.1.2000
- 27 Mitglieder: Direktor + Prüferinnen und Prüfer + Service Office
- Initiativprüfungen, Sonder-prüfungen, Folgeprüfungen, Gutachten

Ausgangssituation

- Proprietäre Applikation für Unterstützung der Prüfungsprozesse
- Probleme mit Zuverlässigkeit und Leistung dieser Applikation
- 2009/2010 provisorische Sanierung der Applikation
- Entscheidung die Applikation abzulösen
- Erhebung der Geschäftsprozesse und Anforderungen
- **Entscheidung für Software mit offener Lizenzierung**
- Evaluierung von Open-Source-Software-Produkten

Beschaffungsvarianten gem. BVerG 2006

RIS

- Variante 1
  - **Unentgeltlicher** Download von OS-Software
  - **Im Haus:** Weiterentwicklung und Wartung
- Variante 2
  - **Unentgeltlicher** Download von OS-Software
  - **Vergabe**: Weiterentwicklung und Wartung
- Variante 3
  - **Vergabe**: Gesamtlösung mit offener Lizenzierung

Dienstleistungsaufträge

§ 6. Dienstleistungsaufträge sind entgeltliche Aufträge, die keine Bau- oder Lieferaufträge sind und deren Vertragsgegenstand Dienstleistungen im Sinne der Anhänge III (prioritäre Dienstleistungsaufträge) oder IV (nicht prioritäre Dienstleistungsaufträge) sind.

Allgemeine Bestimmungen betreffend die Vergabe von Leistungen und Teilleistungen

§ 22. (1) Leistungen können gemeinsam oder getrennt vergeben werden. […] Für die Gesamt- oder getrennte Vergabe von Leistungen sind wirtschaftliche oder technische Gesichtspunkte, wie zB die Notwendigkeit einer einheitlichen Ausführung und einer eindeutigen Gewährleistung, maßgebend. […]

(3) Die Wahl zwischen der Vergabe eines einzigen Auftrages oder die Vergabe mehrerer getrennter Aufträge darf nicht mit der Zielsetzung erfolgen, die Anwendung dieses Bundesgesetzes zu umgehen.

Quelle: Rechtsinformationssystem des Bundes (RIS), Bundesvergabegesetz 2006, http://goo.gl/oRWi2

Vergabegrundsätze

- Beachtung unionsrechtlicher Grundfreiheiten
- Freier und lauterer Wettbewerb
- Diskriminierungsverbot
- Gleichbehandlungsgebot
- Transparenzgebot

Grundsätze des Vergabeverfahrens

§ **19.** (1) Vergabeverfahren sind nach einem in diesem Bundesgesetz vorgesehenen Verfahren, unter Beachtung der unionsrechtlichen Grundfreiheiten sowie des Diskriminierungsverbotes entsprechend den Grundsätzen des freien und lauteren Wettbewerbes und der Gleichbehandlung aller Bewerber und Bieter durchzuführen. Die Vergabe hat an befugte, leistungsfähige und zuverlässige Unternehmer zu angemessenen Preisen zu erfolgen.

- Keine Bevorzugung von OSS gegenüber proprietärer Software aufgrund allgemeiner Erwägungen, beispielsweise aus förderungspolitischen Motiven oder anderer vergaberechtsfremder Kriterien
- Sachliche Gründe für alle Eignungs-, Auswahl- und Zuschlagskriterien sowie deren Bewertung

27.2.13 Schiffer, Sterrer: Ausschreibung von Software mit offener Lizenzierung 5

Neutrale Leistungsbeschreibung [1]

5. Eine ausdrückliche Anforderung von Open Source Produkten (z.B. Linux), eine Leistungsbeschreibung, die als Bedarf ausdrücklich Open Source vorschreibt oder einengende Kriterien wie die Offenlegung des Quellcodes oder die Vereinbarung von Open Source Lizenzen nennt, ist vergaberechtlich rechtfertigungsbedürftig. Sie widerspricht im Regelfall dem Gebot einer neutralen Leistungsbeschreibung (§ 8 Nr. 3 VOL/A).

6. Eine Rechtfertigung solcher Ausschreibung durch die „Art der zu vergebenden Leistung" scheidet im Prinzip aus, weil es im Rahmen einer funktionalen Leistungsbeschreibung ohne Weiteres möglich wäre, die hinter Open Source Produkten vermuteten Vorteile abstrahierend so zu beschreiben, dass Hersteller proprietärer Software in ihrer konkurrierenden Bieterrolle nicht diskriminiert werden.

Quelle: Dirk Heckmann: *Open Source - Closed Shop? Vergaberechtliche Anforderungen an behördliche IT-Ausschreibungen zugunsten von Open Source Software* - Wissenschaftliches Rechtsgutachten im Auftrag der Microsoft GmbH, Passau, 2006, http://goo.gl/rhldb

27.2.13 Schiffer, Sterrer: Ausschreibung von Software mit offener Lizenzierung 6

Neutrale Leistungsbeschreibung 2

- Kein Verweis auf bestimmte Produkte
- Einzige Ausnahme:

  Auftragsgegenstand kann nicht hinreichend genau und allgemein verständlich beschrieben werden
- Dann aber Zusatz

  "oder gleichwertig"

Technische Spezifikationen

§ 98. (7) Soweit es nicht durch den Auftragsgegenstand gerechtfertigt ist, darf in technischen Spezifikationen nicht auf eine bestimmte Produktion oder Herkunft oder ein besonderes Verfahren oder auf Marken, Patente, Typen, einen bestimmten Ursprung oder eine bestimmte Produktion verwiesen werden, wenn dadurch bestimmte Unternehmer oder bestimmte Produkte begünstigt oder ausgeschlossen werden.

Solche Verweise sind jedoch ausnahmsweise zulässig, wenn der Auftragsgegenstand nicht hinreichend genau und allgemein verständlich beschrieben werden kann. Solche Verweise sind ausnahmslos mit dem Zusatz „oder gleichwertig“ zu versehen.

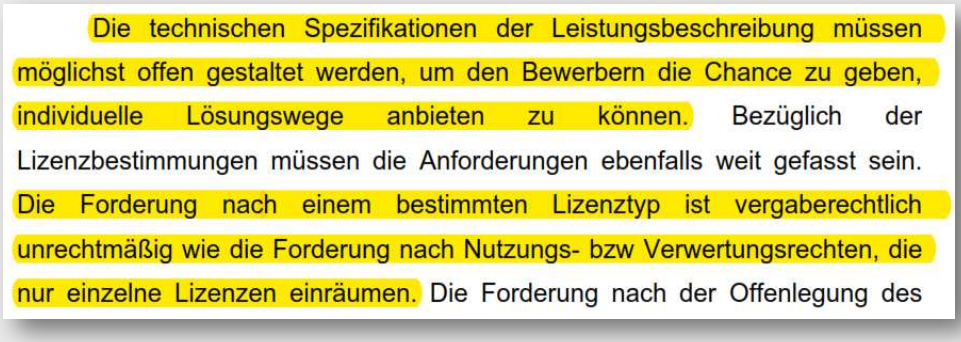
Keine vergabefremden Kriterien aber auch nicht bloß Preisvergleiche

Die Beauftragte der Bundesregierung für Informationstechnik

Bei Anlegung dieser Grundsätze ergibt sich das folgende Bild: Pauschale Hinweise auf die Förderung von Open-Source-Software oder des Wettbewerbs auf den IT-Märkten sind als vergabefremde Kriterien unzulässig. Die Beschaffung durch Behörden ist nicht der richtige Platz, um Wettbewerbspolitik zu betreiben. Das Gleiche gilt für sozialpolitische oder sonstige allgemeine Erwägungen. Behörden dürfen entsprechende Argumente bei der Begründung einer Vergabeentscheidung nicht zugrunde legen. Gleichwohl müssen sich Behörden nicht mit einem einfachen Preisvergleich der Gesamtangebote begnügen. Die Erfahrung zeigt, dass kurzfristige monetäre Vorteile oftmals später teuer bezahlt werden müssen. Dies kann insbesondere dann der Fall sein, wenn Behörden Produkte beschaffen, die nur mit Produkten desselben Anbieters kombiniert werden können oder für die ausschließlich dieser Anbieter Supportleistungen anbietet. Kurzfristige Preisnachteile können oft mittelfristig durch die Unabhängigkeit von einzelnen Anbietern auf den Folgemärkten ausgeglichen werden. Freie Software bietet hier einen strategischen Vorteil. Offene Quelltexte und die

Quelle: Die Beauftragte der Bundesregierung für Informationstechnik (Hrsg.): *Rechtliche Aspekte der Nutzung, Verbreitung und Weiterentwicklung von Open-Source-Software*, Bundesministerium des Innern, Berlin, 2012, http://goo.gl/NEUC0

Keine Forderung bestimmter Lizenzen



Die technischen Spezifikationen der Leistungsbeschreibung müssen möglichst offen gestaltet werden, um den Bewerbern die Chance zu geben, individuelle Lösungswege anbieten zu können. Bezüglich der Lizenzbestimmungen müssen die Anforderungen ebenfalls weit gefasst sein. Die Forderung nach einem bestimmten Lizenztyp ist vergaberechtlich unrechtmäßig wie die Forderung nach Nutzungs- bzw Verwertungsrechten, die nur einzelne Lizenzen einräumen. Die Forderung nach der Offenlegung des

Quelle: Kai Erenli: *Die rechtliche Relevanz von Open Source Lizenzen unter besonderer Berücksichtigung praktischer Problemstellungen*, Dissertation, Karl-Franzens-Universität Graz, 2007, http://goo.gl/oEi8G

27.2.13 Schiffer, Sterrer: Ausschreibung von Software mit offener Lizenzierung 9

Begründung der offenen Lizenzierung in den Teilnahmebedingungen

- Der LRH ist ein unabhängiges Organ des Landtages. Gem. § 1 Oö. LRHG ist der LRH bei der Besorgung seiner Aufgaben unabhängig; gem. § 7 leg. cit. ist er überdies zur Wahrung strengster Verschwiegenheit verpflichtet. Diese gesetzlichen Bestimmungen sind umfassend zu verstehen und betreffen - soweit dies möglich und zweckmäßig ist - auch die vom LRH eingesetzte Software.
- Um Unabhängigkeit und Vertraulichkeit zu wahren, erachtet es der LRH als notwendig, die zu beschaffende Software jederzeit hinsichtlich allfälliger Mängel überprüfen zu können und diese beheben zu lassen, sowie die Software nach eigenem Ermessen warten und weiterentwickeln zu lassen und zwar auch dann, wenn die Mitwirkung des Softwareherstellers nicht möglich oder nicht angebracht ist.
- Dem LRH ist das unwiderrufliche, räumlich, zeitlich und inhaltlich uneingeschränkte Recht einzuräumen, den Source Code der Software für eigene Zwecke zu analysieren, zu nutzen und zu verändern und sich dazu Dritter zu bedienen.

27.2.13 Schiffer, Sterrer: Ausschreibung von Software mit offener Lizenzierung 10

V-Modell XT 1.4 als Leitlinie für den Beschaffungsprozess

- Keine strikte Anwendung des V-Modells, weil damit keine oder zu wenig Erfahrung bei Auftraggeber und potentiellen Bietern (geringer Verbreitungsgrad in Österreich)
- Rollen, Produkte und Aktivitäten des V-Modells als wertvolle Leitlinie während der Vorbereitung und Durchführung der Ausschreibung, insbesondere wegen klarer Begrifflichkeit
- Für Projektdurchführung entsprechende Festlegungen im Rahmenvertrag
- Beispiele für genutzte und geforderte **Rollen**
  - Anforderungsanalytiker, Lenkungsausschuss, Projektleiter, Prüfer, QS-Verantwortlicher, SW-Architekt, SW-Entwickler, Trainer
- Beispiele für genutzte und geforderte **Produkte**
  - Lastenheft, Pflichtenheft, Änderungsentscheidungen, Ausbildungsunterlagen, Migrationskonzept, Nutzungsdokumentation, Projektplan, Projektstatusbericht, Prüfprotokolle und Prüfspezifikationen, QS-Bericht, Risikoliste, Systemarchitektur

Verhandlungsverfahren mit vorheriger Bekanntmachung (Nov. 2011 – Dez. 2012)

I. Einladung zur Teilnahme
1. Abgabe von Teilnahmeanträgen
2. Auswahl von Bewerbern

II. Einladung zur Angebotsabgabe
1. Abgabe von Angeboten
2. Verhandlungen mit Bietern
3. Abgabe von Angeboten
4. Verhandlungen mit Bietern
5. […]
6. Aufforderung zur Abgabe des Last and Final Offers

III. Zuschlagserteilung

Arten der Verfahren zur Vergabe von Aufträgen

§ 25. (5) Beim Verhandlungsverfahren mit vorheriger Bekanntmachung werden, nachdem eine unbeschränkte Anzahl von Unternehmern öffentlich zur Abgabe von Teilnahmeanträgen aufgefordert wurde, ausgewählte Bewerber zur Abgabe von Angeboten aufgefordert. Danach kann über den gesamten Auftragsinhalt verhandelt werden.

Begriffsbestimmungen

§ 2. Im Geltungsbereich dieses Bundesgesetzes sind folgende Begriffsbestimmungen maßgebend:

20. **Kriterien:**

a) **Auswahlkriterien** sind die vom Auftraggeber in der Reihenfolge ihrer Bedeutung festgelegten, nicht diskriminierenden, auf den Leistungsinhalt abgestimmten, unternehmerbezogenen Kriterien, nach welchen die Qualität der Bewerber beurteilt wird […]

c) **Eignungskriterien** sind die vom Auftraggeber festgelegten, nicht diskriminierenden, auf den Leistungsinhalt abgestimmten Mindestanforderungen an den Bewerber oder Bieter, die gemäß den Bestimmungen dieses Bundesgesetzes nachzuweisen sind.

d) **Zuschlagskriterien** […]

aa) sind bei der Wahl des technisch und wirtschaftlich günstigsten Angebotes die vom Auftraggeber im Verhältnis oder ausnahmsweise in der Reihenfolge ihrer Bedeutung festgelegten, nicht diskriminierenden und mit dem Auftragsgegenstand zusammenhängenden Kriterien, nach welchen das technisch und wirtschaftlich günstigste Angebot ermittelt wird […]

Bewertungsverfahren für offene Lizenzierung

- Ziel: Erreichung eines möglichst hohen Anteils offener Lizenzen
- Offene Lizenz: individuelle offene Lizenz oder von Open Source Initiative (OSI) anerkannte Lizenz
- **Keine Forderung nach 100% Open-Source-Software, weil fachlich und technisch einschränkend sowie vergaberechtlich riskant**
- 15% Anteil der Lizenzierung an den Zuschlagskriterien (50% Preis, 30% Teststellung, 5% Sollanforderungen)
- Für Individualsoftware-Anteile jedenfalls volle Werknutzungsbewilligung mit Ausfolgung des Source Codes
- **Festlegung der Lizenzierung mittels Dateiliste einer Standardinstallation**
- Automatische Auswertung, um Aufwand und Fehler zu minimieren

Vertragliche Vereinbarung zu offener Lizenzierung

Für die in der Beilage "Dateiliste einer Standardinstallation" angeführten Dateien, die mit "ja" in Spalte "Bieter gewährt Lizenz gem. Punkt 9.1 des Rahmenvertrages" gekennzeichnet wurden, räumt der Auftragnehmer

- das unwiderrufliche, räumlich, zeitlich und inhaltlich uneingeschränkte Recht ein,
- diese Dateien und deren Source Code
- für eigene Zwecke zu analysieren, zu nutzen und zu verändern und
- sich dazu Dritter zu bedienen.

Neben dem Source Code muss der Auftragnehmer auch Build-Skripts, alle erforderlichen Libraries und die Dokumentation des Build-Vorgangs übergeben, damit der Auftraggeber in die Lage versetzt wird, in seiner Umgebung selbst die lauffähige Anwendung zu erzeugen.

Dateiliste einer Standardinstallation

SUMMEWENN | =WENN(K6="ja";WENN(L6="ja";"";I6);"")

	A	B	D	E	G	H	I	J	K	L
1	**Pfad**	**Dateigröße aufgerundet auf volle Kilobyte**	**Duplikat**	Extension	**Konfigurationsdatei, Quelldatei, daraus erzeugt oder Teil der Build-Umgebung**	**Datei unterliegt einer von OSI anerkannten Lizenz**	Anteil %	Punkteabzug	LRH OÖ fordert Lizenz gem. Punkt 9.1 des Rahmenvertrages	**Bieter gewährt Lizenz gem. Punkt 9.1 des Rahmenvertrages**
2	:86)\PDFCreator\AFPL License.txt	13		txt	nein				nein	
3	)05\C#\Sample1\AssemblyInfo.cs	1		cs	ja		0,0076		ja	ja
4	et\VS2005\C#\Sample1\Form1.cs	12		cs	ja		0,0908		ja	ja
5	\VS2005\C#\Sample1\Form1.resx	7		resx	nein				nein	
6	:005\C#\Sample1\Sample1.csproj	5		csproj	ja		0,0378	=WENN(K6="	ja	nein
7	)05\C#\Sample2\AssemblyInfo.cs	1	ja	cs	ja				nein	
8	et\VS2005\C#\Sample2\Form1.cs	7		cs	ja		0,0530	0,0530	ja	
9	\VS2005\C#\Sample2\Form1.resx	9		resx	nein				nein	
10	:005\C#\Sample2\Sample2.csproj	5		csproj	ja		0,0378	0,0378	ja	nein
11	Basic\Sample1\AssemblyInfo.vb	1		vb	ja		0,0076	0,0076	ja	nein
12	Visual Basic\Sample1\Form1.resx	7	ja	resx	nein				nein	
13	:\Visual Basic\Sample1\Form1.vb	11		vb	ja		0,0833		ja	ja
14	al Basic\Sample1\Sample1.vbproj	5		vbproj	ja		0,0378		ja	ja
15	Basic\Sample2\AssemblyInfo.vb	1	ja	vb	ja				nein	
16	Visual Basic\Sample2\Form1.resx	9	ja	resx	nein				nein	
17	:\Visual Basic\Sample2\Form1.vb	8		vb	ja		0,0606		ja	ja
18	al Basic\Sample2\Sample2.vbproj	5		vbproj	ja		0,0378		ja	ja
19	OTNET Scripting Host\readme.txt	1		txt	nein				nein	
73	tor\GS9.04\gs9.04\Bin\gsdll32.dll	14.296		dll	ja	ja			nein	
74	tor\GS9.04\gs9.04\Bin\gsdll32.lib	8		lib	ja	ja			nein	

Bewertungsschema

- Bewertungsrelevant sind alle Dateien, für die eine offene Lizenz gefordert wird
- Die Aufteilung der Punkte auf die einzelnen Positionen erfolgt anhand der Dateigröße
- Es wird die Größe aller bewertungsrelevanten Dateien summiert. Diese Summe ergibt 100% der Lizenzierungspunkte
- Für jede Datei, für die eine offene Lizenz gefordert aber nicht gewährt wird, werden Punkte entsprechend dem Größenanteil der jeweiligen Datei abgezogen
- Der Bieter hat für solche Dateien zu begründen, warum er keine offene Lizenz gewährt
- Die Gesamtpunkte ergeben sich im konkreten Fall durch 0,15 * (100-Punkteabzug), weil 15%-Anteil der Lizenzierung an Zuschlagskriterien
- Festlegung einer Mindestpunkteanzahl möglich, andernfalls das Angebot auszuscheiden ist (z.B. 80% der Lizenzierungspunkte)

Tabellenblatt für Standardfestlegung der Lizenzforderung

	A	C
1	Extension	**Konfigurationsdatei, Quelldatei, daraus erzeugt oder Teil der Build-Umgebung**
17	au	nein
18	awk	ja
19	bas	ja
20	bat	nein
21	bau	nein
22	bfc	nein
23	bin	ja
24	bki	nein
25	bmp	nein
26	browser	ja
27	bsh	ja
28	buildinfo	ja
29	c	ja
30	cer	nein
31	cfg	nein
32	cfs	nein
33	chm	nein
34	class	ja
35	cluster	nein

- Der Bieter trägt alle Dateitypen des Produkts ein
- Zuordnung von Dateityp zu
  - Konfigurations- oder Quelldatei (z.B. INI-Dateien, Skript-Dateien)
  - Dateien, die aus Konfigurations- oder Quelldateien erzeugt werden (z.B. DLL-Dateien, JAR-Dateien)
  - Dateien der Build-Umgebung (z.B. Projektdateien, Build-Skripts)
- Forderung nach offener Lizenz für alle Dateien mit Angabe "ja" in Spalte C

Überprüfung der Bieterangaben

- Teststellung der Standardinstallation auf virtuellem Computer mit Zugriff auf Dateisystem
- Überprüfung der Bieterangaben hinsichtlich der Kategorisierung der Dateitypen durch Inspektion der Dateien
- Aufklärungspflicht des Bieters bei Zweifel an der Kategorisierung

Beispiel: Sehr geehrter Bieter, die Analyse der von Ihnen in der Dateiliste einer Standardinstallation genannten Dateien vom Typ CSS und XSD, die keine Duplikate sind und von Ihnen in Spalte G der Dateiliste einer Standardinstallation "nein" gekennzeichnet wurden, hat ergeben, dass diese Dateien aus Sicht des Auftraggebers als Quelldateien und/oder Teil der Build-Umgebung zu klassifizieren sind.

Wir fordern Sie daher auf, für jede der im Anhang angeführten Dateien bekanntzugeben, warum diese Datei

- ein Teil der Standardinstallation der von Ihnen angebotenen Software ist und
- vom Auftraggeber keinesfalls benötigt wird, um die von Ihnen angebotene Software hinsichtlich allfälliger Mängel überprüfen zu können und diese beheben zu lassen, sowie die Software nach eigenem Ermessen warten und weiterentwickeln zu lassen.

Wir weisen Sie ausdrücklich darauf hin, dass das Angebot eines Bieters, der unrichtige oder ungenügende Angaben gemacht hat, ausgeschieden werden kann.

Zusammenfassung

- Das beschriebene Verfahren ermöglicht die Beschaffung von Software mit offener Lizenzierung unter Berücksichtigung der vergaberechtlichen Vorgaben
- Keine Forderung eines bestimmten Produktes oder bestimmter Lizenzen
- Keine Forderung von 100%-Open-Source-Anteil, sondern sachlich begründbare und praxisgerechte Festlegung einer offenen Lizenzierung
- Das V-Modell XT ist eine wertvolle Hilfestellung im Beschaffungsprozess, insbesondere wegen seiner klaren Begrifflichkeit

5.3 Ausschreibung von Software mit offener Lizenzierung für eine Institution der öffentlichen Finanzkontrolle in Österreich (27.02.2013)

6. Anforderungen

Sessionüberblick

6.1. „Make, Buy, Use oder Compose?“ als zentrales Element des Anforderungsmanagmentprozesses in einer großen Kommune (28.02.2013) 156
6.2. eBAföG - ein ehrlicher Erfahrungsbericht (01.03.2013) 172

„Make, Buy, Use oder Compose?“ als zentrales Element des Anforderungsmanagmentprozesses in einer großen Kommune

Friedrich Strauß[1], Helmut Siemon[2]
Landeshauptstadt München, Sozialreferat[1]/ IT@M[2]
Orleanstr. 50, 81667 München
friedrich.strauss@muenchen.de
helmut.siemon@muenchen.de

Abstract: Die Landeshauptstadt München hat ein erweitertes Vorgehensmodell für die Entwicklung von IT-Services und Systemen entwickelt, das als ein zentrales Element die frühe Orientierungsphase des Anforderungsmanagements strukturiert. Hierdurch wird auch der Prozess bis zur Entscheidung für eine Individual- (**M**ake) oder Standardsoftware (**B**uy), die Verwendung vorhandener Software (**U**se) oder eine Kombination dieser Elemente (**C**ompose) standardisiert. Dieser MBUC-Prozess stellt neben einer transparenten Entscheidungsfindung zusätzlich sicher, dass übergeordnete strategische Anforderungen beispielsweise aus dem Enterprise Application Management (EAM) systematisch und rechtzeitig Berücksichtigung finden. Das besondere des Münchner Ansatzes ist die Einführung von zwei sich ergänzenden neuen Rollen, Facharchitekt und IT-Architekt, die diesen MBUC-Prozess verantworten.

1. Einleitung & Problemstellung

Die Landeshauptstadt München (LHM) hat ihre gesamte IT nach dem Modell der Kernkompetenzfokussierung reorganisiert [Na10] und dabei eine dezentrale IT in den Referaten etabliert, die das Anforderungsmanagement verantwortet und die Einführungsprozesse neuer Software unterstützt. Daneben wurde ein zentraler IT-Dienstleister IT@M gegründet, der als städtischer Eigenbetrieb für alle technischen IT-Themen verantwortlich ist (Planung, technische Konzeption, Infrastrukturmanagement und Betrieb der IT)[1].

Unterstützend zur Kernkompetenzfokussierung hat die LHM auch einen generischen Prozess für die Konzeption und Erstellung einer IT-Lösung eingeführt, der durch eine neu etablierte MBUC-Empfehlung (**M**ake, **B**uy, **U**se or **C**ompose a Software) als Abschluss der Anforderungsqualifizierung auch die frühe Orientierungsphase für IT-Vorhaben definiert (siehe Abbildung 1) [HS12].

Für öffentliche Auftraggeber ist auch diese *frühe Phase erfolgskritisch*, da die EU-Vergaberegelungen öffentliche Auftraggeber zu produktneutralen Ausschreibungen verpflichtet. Andererseits weist eine Anforderungsermittlung für Individualsoftware deutliche Unterschiede zu der für eine Standardsoftware auf [KMS05], so dass

[1] Die IT der Stadt München umfasst dezentral und zentral mehr als 1.000 Mitarbeiter und betreut ca. 15.000 EDV-Arbeitsplätze.

rechtzeitig klar sein muss, ob eine Standardsoftware oder eine Individualsoftware beschafft werden soll. In der Vergangenheit gab es Ausschreibungen, die zu keiner Vergabe führten (z.B. zu hohe strategische Anforderungen) oder unterschiedliche Lösungen für verwandte Aufgaben (zu kleinteilige „unabhängige“ Vergaben) ergaben. Das neue Vorgehensmodell mit der dedizierten Make or Buy Entscheidung berücksichtigt auch Erfahrungen, die München mit ihrer Linux-Strategie gesammelt hat: Es ist ein komplexer Entscheidungsprozess, wann eine nur unter Windows existierende Standardsoftware sinnvoll durch eine Neuprogrammierung unter Linux ersetzt werden kann oder zum Beispiel aus Wirtschaftlichkeitsgründen doch mittels Terminal-Server weiter betrieben werden sollte.

Bei der MBUC-Empfehlung sollen gleichberechtigt fachliche, technische und strategische Anforderungen berücksichtigt werden. Um diesen Prozess auch in der gewünschten Qualität leben zu können, wurden zwei neue Rollen Facharchitekt (dezentral in jedem Referat) und IT-Architekt (beim zentralen IT-Dienstleister) eingeführt, die dauerhaft die IT-Vorhaben einer Domäne (z.B. das Stadtjugendamt oder das Kreisverwaltungsreferat) betreuen.

2. Der neue Anforderungsmanagementprozess

Die folgende Abbildung gibt eine Übersicht über den Anforderungsmanagementprozess der LHM:

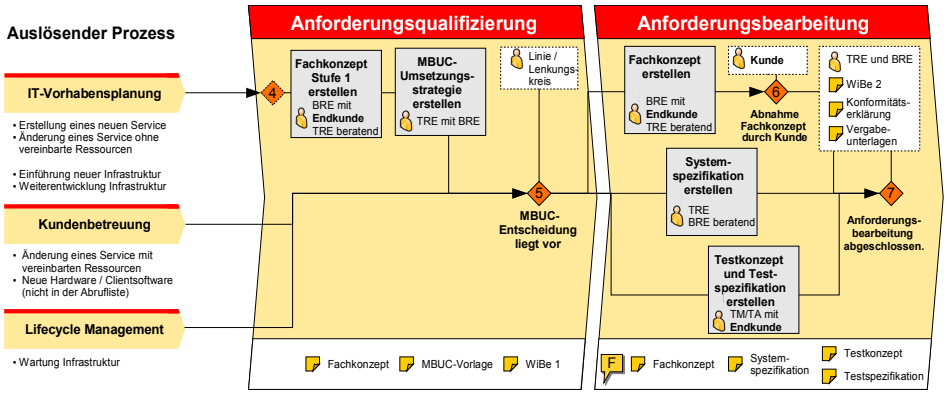


Abbildung 1: Prozessübersicht Anforderungsmanagement.[2]

2.1 Die MBUC-Empfehlung

Für die MBUC-Empfehlung wesentlich ist das Wechselspiel von Markterkundung käuflicher Standard-Lösungen, der Priorisierung von fachlichen und technischen Anforderungen sowie die Berücksichtigung von strategischen Aspekten (EAM, weitere

[2]BRE/TRE = Business bzw. Technical Requirements Engineer (Fach bzw. IT-Architekt)

Vorhaben mit ggf. überlappenden Anforderungsbereichen, ggf. auch Lifecycle Management). Daneben gehen natürlich die grundlegenden Kriterien ein (Kosten, Zeit bis zur Bereitstellung, benötigte ggf. knappe interne / externe Ressourcen).

2.2 Kompentenzen Fach- und IT-Architekt

Die organisatorische Einführung von Facharchitekten (im Referat) und IT-Architekten (beim zentralen Dienstleister) ist eine Voraussetzung, um eine ausgewogene Bewertung von ggf. gegensätzlichen fachlichen und technischen Anforderungen zu ermöglichen. Zu wichtigen Aufgabenbereichen gehört das Stake-Holder Management, wobei es entscheidend ist, die MBUC-Empfehlung für die betroffenen Teilorganisationen nachvollziebar darzustellen und ggf. zu erläutern, warum bestimmte Anforderungen nicht oder nicht vollständig erfüllt werden können.

Zusätzlich umfasst der Aufgabenbereich der Architekten auch die Mitarbeit bei der fachlichen und technischen Soll-Bebauung für ihren Zuständigkeitsbereich. Dadurch werden sie in der Lage versetzt, kontinuierlich bei jeder Projektentscheidung (insbesondere bei der MBUC-Entscheidung und beim Projekt-Scoping) zu prüfen, inwieweit man sich der SOLL-Bebauung annähern kann:

2.3 Relevanz des Enterprise Application Managements

Ein zumindest bei öffentlichen Einrichtungen auftretendes Problem ist die häufig isolierte Umsetzung von IT-Anforderungen für einzelne Abteilungen oder Sachgebiete. Eine lokale Bearbeitung dieser IT-Anforderungen führt ggf. zu kleinen und isolierten IT-Systemen. Zudem sind strategische Rahmenbedingungen geeignet zu berücksichtigen (wie z.b. die Linux-Strategie der LHM, das Lifecycle Management). In einer großen Kommune wie der LHM ist es daher wichtig, diese Herausforderungen durch ein systematisches Enterprise Application Management zu bearbeiten. Zusätzlich muss sichergestellt werden, dass die im Rahmen des EAMs erarbeite fachliche und technische Bebauung auch ausreichend in die MBUC-Empfehlungen einfließt.

Damit die Architekten auch die beim EAM erarbeiteten strategischen Ziele sinnvoll berücksichtigen, ist bei der LHM der Aufgabenbereich der Architekten passend geschnitten worden: Kernpunkt ist die gemeinsame Verantwortung für die Erstellung einer fachlichen und technischen IST und SOLL-Bebauung als Teil des EAMs der LHM München. Für die vom Facharchitekten zu entwickelnde fachliche Bebauung muss dabei mit den Fachbereichen bzw. der Referatsleitung abgestimmt sein , welche Kompetenzen die Facharchitekten beim Geschäftsprozessmanagement erhalten und wann/wie die Facharchitekten bei Geschäftsprozessveränderungen beteiligt werden.

3. Ergebnis der IT-Neuausrichtung

Die kontinuierliche Zusammenarbeit von Fach- und IT-Architekt für einen festen Wirkungsbereich (ein Referat oder ein Eigenbetrieb der LHM) hat es zum Beispiel im Sozialreferat ermöglicht, projektübergreifende Optimierungsbedarfe und -möglichkeiten zu entdecken. In einer Ausschreibung für eine Software für das Stadtjugendamt konnte der Funktionsumfang im Rahmen der Verhandlung so erweitert werden, dass mehrere isolierte kleine IT-Systeme mit abgelöst werden und so die IT-Anwendungslandschaft vereinfacht wird – hier war die Übersicht über die fachlichen Prozesse und zur bisherigen IT-Unterstützung wesentlich. In einem anderen Projekt konnten wir im Rahmen der Markterkundung feststellen, dass es mittlerweile ausreichend Anbieter gibt, die zur LHM-Strategie kompatible Anwendungen (mit Web-Frontend) entwickelt haben und so die Client-Infrastrukturanforderungen für die Vergabe passend gestalten.

4. Ausblick

Die Pärchenbildung von Fach- und IT-Architekt sowie die personelle Verbindung von IT-Projektbetreuung und EAM-Aufgaben wirkt schon in den einzelnen Einheiten der LHM, auch wenn derzeit noch eine durchgängige Werkzeugunterstützung fehlt. Weitere organisatorische Regelungen sind aber nötig bei bei der Konzeption stadtweiter Vorhaben mit mehreren beteiligten Architekten. Hierzu sind übergreifend besetzte Professional Groups angedacht, die die Anforderungen verschiedener Referate konsolidieren sollen.

Literaturverzeichnis

[HS12] Hauber, R.; Schmailzl, H.: Anforderungsmanagement bei der Landeshauptstadt München, ReConf 2012.

[Na10] Naefe, T.: IT im öffentlichen Bereich, Tagung Softwareengineering-Live, Wildbad Kreuth, 2011.

[KMS05] Kowallik, P.; Müller-Prove, M.; Strauß, F.: Requirements-Engineering im Spannungsfeld von Individual- und Produktsoftware. In: i-com Zeitschrift für interaktive und kooperative Medien, Volume: 4, Issue: 3/2005, S. 41 – 46.

„Make, Buy, Use oder Compose?“

als zentrales Element des Anforderungsmanagmentprozesses in einer großen Kommune

Friedrich Strauß, Facharchitekt, Landeshautpstadt München Sozialreferat

Helmut Siemon, IT-Architekt, Landeshauptstadt München it@M

Aachen, Software & Systems Engineering Essentials (SEE) 2013

„Make, Buy, Use oder Compose?“
Dr. Friedrich Strauß, Helmut Siemon
Landeshauptstadt München

Agenda

1. Einleitung
2. Kernelemente der MBUC-Methodik bei der Landeshauptstadt
   - Anforderungsqualifizierung und Anforderungsbearbeitung
   - Zweiteilung der zentralen Aufgaben FAR und ITA in einem Projekt
   - Fachliche Bebauung und MBUC als Verantwortungsbereich in Personalunion
3. MBUC-Beispiele
4. Ausblick

„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 3

Projekt zur strategischen Neuausrichtung der IT der Landeshauptstadt München (LHM): MIT-KonkreT

(Konzeption ab 2007, Umsetzung von 2010 bis 2015)

Ziele von MIT-KonkreT

- Erhöhung der Kundenzufriedenheit
- Ausbau Finanz- und Kostentransparenz
- Klare Regelung von Zuständigkeiten, Verantwortlichkeiten u. Entscheidungswegen
- Standards für zukunftsorientierte IT
- Effizienzsteigerung
- Optimierte und transparente Prozesse
- Sicherstellung Innovationsfähigkeit

Neustrukturierung:

- **der IT-Organisation**
  mit ca. 1.000 Mitarbeiter
- **der Prozesse**
- **der Technik**
  für ca. 15.000 DV-Arbeitsplätze

„Make, Buy, Use oder Compose?“

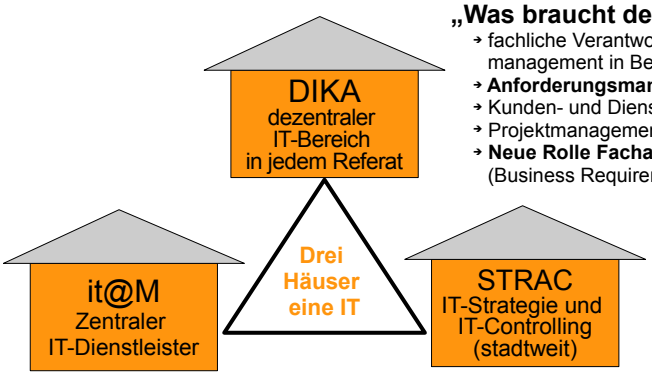
Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 4

Organisatorische Veränderung durch MIT-KonkreT

Die IT der Stadt München wird nach dem Modell der Kernkompetenzfokussierung neu gestaltet



„Was braucht der Kunde?“
- fachliche Verantwortung für das Geschäftsprozess-management in Bezug auf IT-Unterstützung
- **Anforderungsmanagement**
- Kunden- und Dienststellenbetreuung
- Projektmanagement
- **Neue Rolle Facharchitekt**
  (Business Requirements Engineer)

„Wie stelle ich die IT zur Verfügung?“
- **Verantwortung für Umsetzung**, IT-Standards und Betrieb (Server, Netze, Rechenzentrum)
- Vergabe und Beschaffung
- **Neue Rolle IT-Architekt**
  (Technical Requirements Engineer)

Wie gehen wir stadtweit vor?“
- Verantwortung für IT-Governance
- **IT-Strategie**
- IT Security
- Steuerungsunterstützung und IT-Controlling
- Stadtweite Vorhabensplanung

„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 5

Projekte fallen nicht vom Himmel: Ihre Inhalte müssen sorgsam definiert werden

- **IT-Projekte Auswählen**
  Ein komplexer Priorisierungsprozess im Rahmen einer Portfolio-Planung legt fest, welche IT-Projekte durchgeführt werden sollen/können
  - Abhängig vom Portfolio-Management sind die Planungsthemen schon grob konzipiert, ggf. existiert auch nur eine High-Level Problembeschreibung
- **IT-Projekte Definieren**
  - In einer frühen Phase sind für ein IT-Projekt die grundlegenden Rahmenbedingungen festzulegen
- **IT-Projekte Durchführen**
  Unternehmen und Behörden standardisieren die Erstellung von Anwendungs-Software in der Regel durch Vorgehensmodelle,
  - Hier wird bereits ein (grobes) Scoping der zu erstellenden Anwendung und eine getroffene Make-or-Buy Entscheidung vorausgesetzt

Was ist die richtige Reihenfolge für die frühen Phasen eines Projekts:

Scoping

Kaufen oder Bauen

EAM-Strategien

Umsetzungs-entscheidung

„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 6

Wesentliche Aspekte zur Projektdefinition – und ihre Stolpersteine in öffentlichen Verwaltungen

- **Scoping**
  - Festzulegen ist, welche Funktionalitäten eine Anwendung besitzen soll bzw. welche (Geschäfts-) Prozesse unterstützt werden sollen
    - *Scoping anhand des Zuständigkeitsbereichs der treibenden Abteilung?*
    - *(Benachbarte) Sachgebiete fokussieren jeweils auf eine optimale Unterstützung ihrer Abläufe*
- **Make, Buy, Use** (eine existierende Lösung) **or Compose** (eine Kombination)
  - Standardsoftware hat in vielen Fällen Vorteile bei Kosten und Zeitbedarf, dafür ist aber eine Anpassung der Software nötig …
    - *dominieren fachlichen Aspekte (unsere Abläufe können nicht abgeändert werden)*
    - *werden technische Strategien priorisiert (nur Linux-kompatible Software)*
    - *Umständlicher Vergabeprozess bei Buy vs. Interne, teure Umsetzung bei Make ohne öffentliche Vergabe …*
- **Enterprise Architecture Management-Strategien berücksichtigen**
  - Eine im Rahmen einer EAM-Strategie erstellte Ziel-Bebauung muss fort-laufend berücksichtigt werden – ansonsten nähert man sich dem Ziel nie an.
    - *Integrieren wir die separate Statistik-Meldung in die Anwendung?*
    - *Migrieren wir bei dem neuen Release auf einen neuen Datenbankstandard?*

„Make, Buy, Use oder Compose?“

Dr. Friedrich Strauß, Helmut Siemon

Landeshauptstadt München

Seite 7

Agenda

1 Einleitung

2 **Kernelemente der MBUC-Methodik bei der Landeshauptstadt**
- Anforderungsqualifizierung und Anforderungsbearbeitung
- Zweiteilung der zentralen Aufgaben FAR und ITA in einem Projekt
- Fachliche Bebauung und MBUC als Verantwortungsbereich in Personalunion

3 MBUC-Beispiele

4 Ausblick

„Make, Buy, Use oder Compose?“

Dr. Friedrich Strauß, Helmut Siemon

Landeshauptstadt München

Seite 8

Für qualifizierte Projektdefinitionen müssen mehrere Mechanismen ineinander greifen

- **Standardisiertes Vorgehen für die frühen Phasen eines Projekts**
  - Aufteilung in Anforderungsqualifizierung und Anforderungsbearbeitung
- **Feste, getrennt besetzte Rollen für fachliches und technisches Design** (schon in frühen Phasen besetzt)
  - Pro Referat feste Stellen für fachliches und technisches Design
- **Fachliche und technische Soll-Bebauung fortschreiben und verantworten**
  - Fach- und IT-Architekt definieren die Soll-Bebauung für ihre Anwendungsdomäne

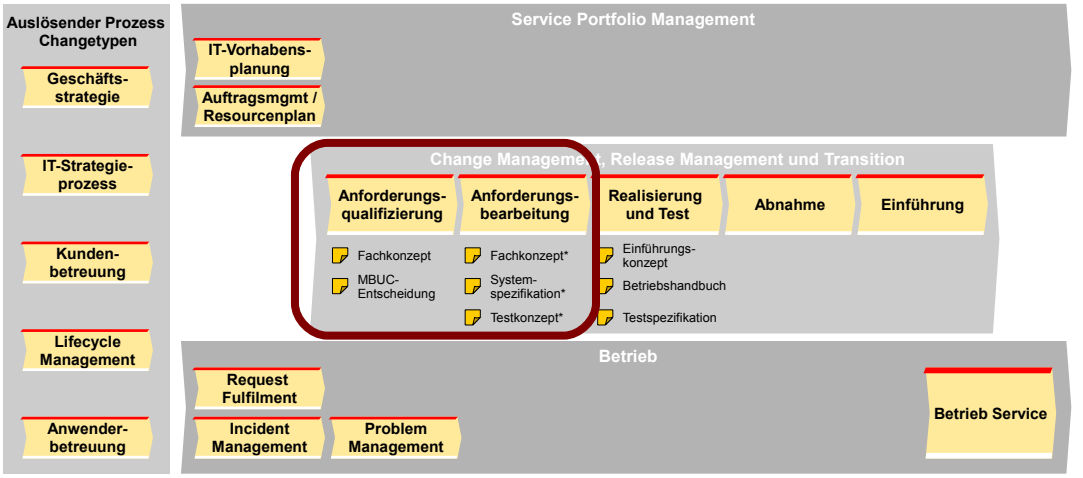
„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 9

Gesamtübersicht Prozessmodell*



* Detailtiefe abhängig von der MBUC-Entscheidung

* Dargestellt ist das in München eingeführten Prozessmodell für die die Erstellung von IT-Services

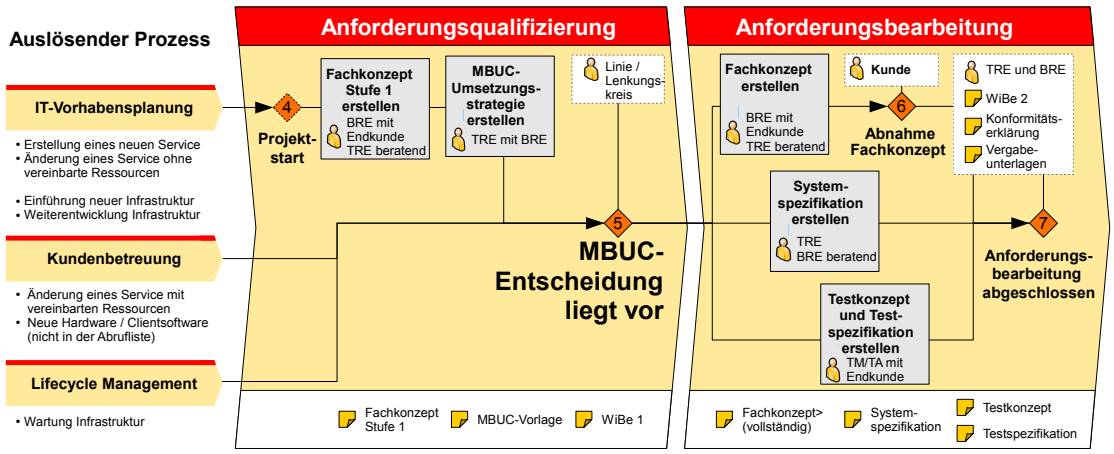
„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 10

Die MBUC-Empfehlung unterteilt die Anforderungsermittlung in eine Qualifizierungs- und eine Bearbeitungsphase



Legende zum Ausschnitt des Prozessmodells

BRE / TRE	= Business / Technicel Requirements Engineer in der Regel mit Fach-/IT-Architekten besetzt	4	Meilenstein
WiBEx	= Wirtschaftlichkeitsberechnung Stufe x		Verantwortliche Person
Designvorgaben Konformitätserklärung)	= IT-Sicherheit, IT-Strategie, Datenschutz, Personalrat		Ergebnisdokument der Aktivität / Phase

„Make, Buy, Use oder Compose?“

Dr. Friedrich Strauß, Helmut Siemon

Landeshauptstadt München

Seite 11

Wesentliche Aspekte des Anforderungsmanagements

Anforderungsqualifizierung

- Die MBUC-Umsetzungsempfehlung wird so frühzeitig wie möglich erstellt
- Die Fachkonzepterstellung durch den BRE erfolgt nur soweit, bis eine MBUC-Empfehlung erstellt werden kann
- BRE und TRE sind dafür verantwortlich,
  - alle relevanten Stakeholder einzubinden
  - die wesentlichen Designvorgaben zu berücksichtigen
- Die MBUC-Entscheidung wird von den Projektverantwortlichen (Lenkungskreis) gefällt
- Markterkundung erfolgt von BRE und TRE gemeinsam !

Anforderungsbearbeitung

- Erst wenn MBUC-Entscheidung vorliegt und Entscheidung über eine Vergabe bzw. über die Plattform für ein Make getroffen ist, wird das Fachkonzept vervollständigt
- Die Systemspezifikation erfolgt überlappend zum FK und dokumentiert alle technischen Designentscheidungen (bzw. Vorgaben für eine Vergabe)
- Zum Abschluss liegen alle erforderlichen Unterlagen für eine Vergabe vor

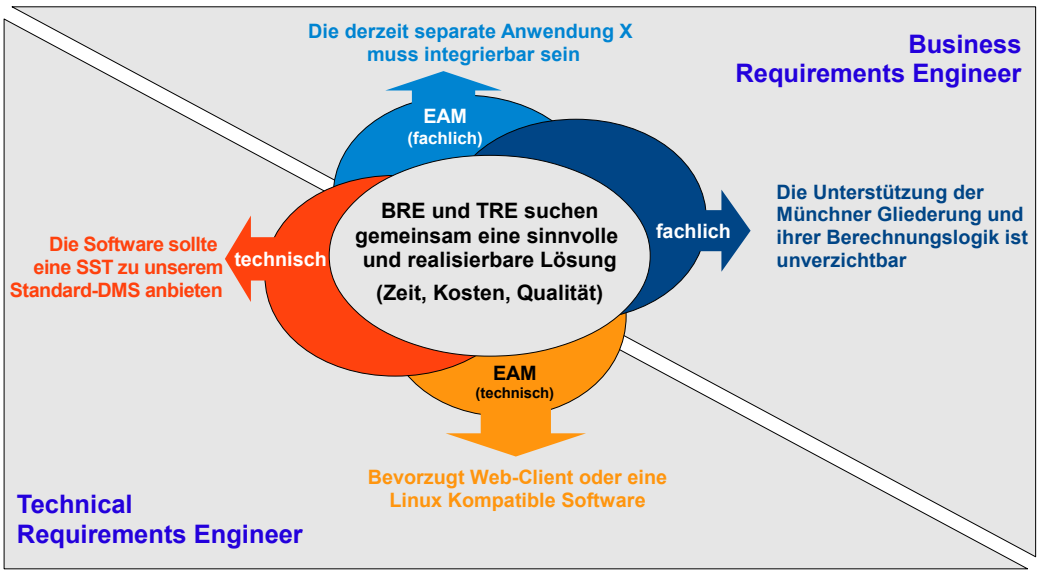
„Make, Buy, Use oder Compose?“

Dr. Friedrich Strauß, Helmut Siemon

Landeshauptstadt München

Seite 12

Die feste Rollenzuordnung für BRE und TRE vereinfacht es, die widerstreitenden Anforderungen geeignet abzuwägen



„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 13

Vorteile der festen Besetzung von Fach- und IT-Architekten

Facharchitekten

- Ein stabiler Verantwortungsbereich ermöglicht es, übergreifendes Wissen zur fachlichen Domäne und den existierenden Geschäftsprozessen anzusammeln
- Dauerhaftes Stakeholder-Management des Fachbereichs ermöglicht Aufbau von vertrauensvoller Zusammenarbeit
  - Verständnis für Abwägungen in MBUC-Empfehlungen nötig
  - Übergreifende fachliche Strategien können aus IT-Sicht begleitet werden – und ermöglicht Rückkopplung bzgl. (un-) möglicher IT-Unterstützungsanforderungen

IT-Architekten

- Können IT-strategische und LifeCycle relevante Anforderungen an die Anwendungslandschaft einbringen
- Wg. Spezialisierung der IT-Architekten sind themenabhängig ggf. weitere IT-Architekten einzubeziehen
  - Zum Beispiel bei EAI-Anforderungen, Spezieller Software wie Groupware, SAP, bei besonderen Infrastruktur oder Security-Anforderungen, ...

„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 14

Erst die Kombination von Vorgehensmodell, Rollen und Verantwortlichkeiten sichert eine belastbare MBUC-Entscheidung

- **Standardisiertes Vorgehen für die frühen Phasen eines Projekts**
  - Aufteilung in Anforderungsqualifizierung und Anforderungsbearbeitung
    → **MBUC möglichst frühzeitig und ein definiertes Quality Gate für MBUC**
- **Feste, getrennt besetzte Rollen für fachliches und technisches Design**
  (schon in frühen Phasen besetzt)
  - Pro Referat feste Stellen für fachliches und technisches Design
    → **Zum Ausgleich fachlicher und technischer Aspekte**
    → **kontinuierliche Zuständigkeit**
- **Fachliche und technische Soll-Bebauung fortschreiben und verantworten**
  - Fach- und IT-Architekt definieren die Soll-Bebauung für ihren Anwendungsdomäne
    → **die EAM-Strategie der öffentlichen Einrichtung ist für die fachliche Domäne konkretisiert bzw. definiert**
    → **Entscheidungen zu Projektinhalte und Projektvorgehen können in gleicher Weise auch EAM Aspekte berücksichtigen.**

„Make, Buy, Use oder Compose?“

Dr. Friedrich Strauß, Helmut Siemon

Landeshauptstadt München

Seite 15

Agenda

1 Einleitung

2 Kernelemente der MBUC-Methodik bei der Landeshauptstadt
- Anforderungsqualifizierung und Anforderungsbearbeitung
- Zweiteilung der zentralen Aufgaben FAR und ITA in einem Projekt
- Fachliche Bebauung und MBUC als Verantwortungsbereich in Personalunion

3 MBUC-Beispiele

4 Ausblick

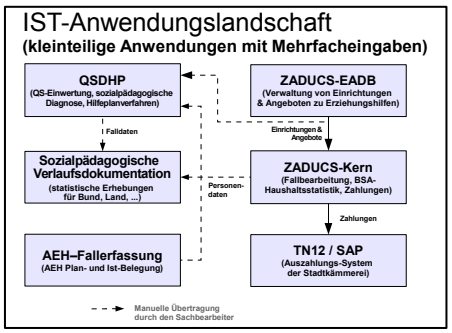
„Make, Buy, Use oder Compose?“

Dr. Friedrich Strauß, Helmut Siemon

Landeshauptstadt München

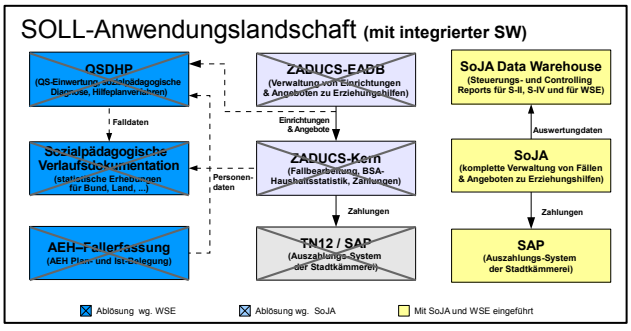
Seite 16

Fusion der IT-Projekte SoJA und WSE nach MBUC Entscheidung zu WSE



- WSE,1 Jahr nach SoJA gestartet, hat erhöhte Anforderungen an die Auswertung von Erziehungshilfen in SoJA

  → MBUC: eine Integration von QSDHP in SoJA ist die optimale Lösung
- Die laufende Vergabe für SoJA konnte im Rahmen der Verhandlung angepasst werden
- Dabei konnten auch einige Kleinsysteme ganz abgeschafft werden.



Projekt- und IT-Systemnamen

SoJA: Software für wirtschaftliche Jugendhilfe und soziale Arbeit

WSE: Wirkungsorientierte Steuerung in den Hilfen zur Erziehung

QSDHP: System u.a. zur Verwaltung von Hilfen zur Erziehung

„Make, Buy, Use oder Compose?“

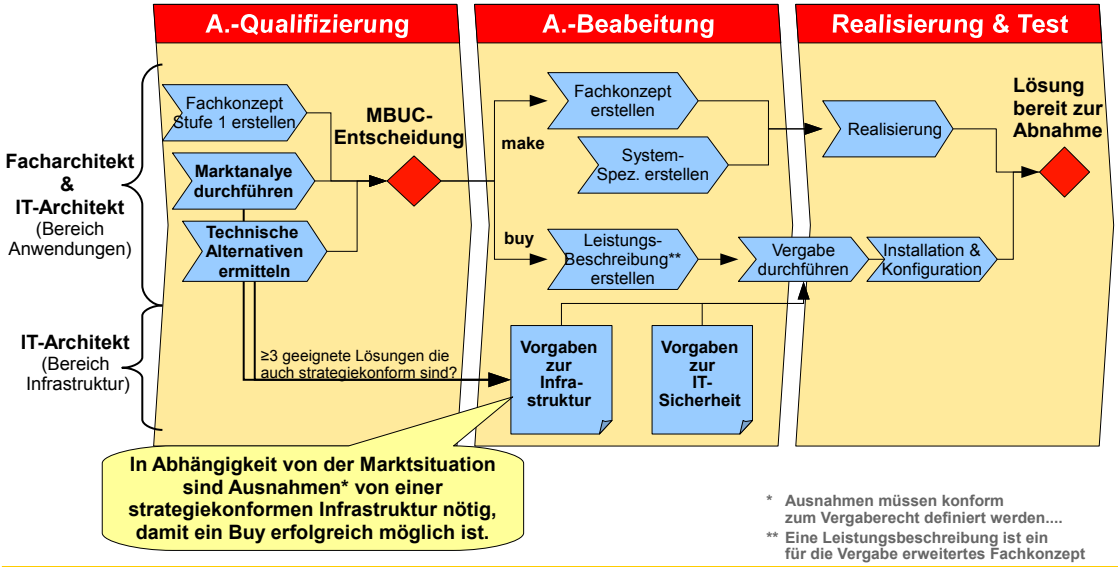
Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 17

MBUC-Beispiele

In den Vergabekriterien für eine Jugendamtsoftware konnte die IT-Strategie der Stadt München adäquat berücksichtigt werden



„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 18

Agenda

1 Einleitung

2 Kernelemente der MBUC-Methodik bei der Landeshauptstadt
- Anforderungsqualifizierung und Anforderungsbearbeitung
- Zweiteilung der zentralen Aufgaben FAR und ITA in einem Projekt
- Fachliche Bebauung und MBUC als Verantwortungsbereich in Personalunion

3 MBUC-Beispiele

4 **Ausblick**

„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 19

Ausblick

Stadtweite Projekte stellen weiterhin eine Herausforderung dar

- Die Vorteile der festen Rollen von Fach- und IT-Achitekt gehen bei stadtweiten Themen verloren, da hier mehrere Architekten verschiedener Referate einzubinden sind.
  - Ein erster Ansatz ist, referatsübergreifende *Professional Groups* aufzubauen, die Anforderungen der verschiedenen Referate konsolidieren

Der Umfang der zu erstellenden Fachkonzepte ist pragmatisch festzulegen

- Der Umfang des Fachkonzepts Stufe 1 ist nicht fest definierbar, es ist darauf zu achten, dass nur die „kritischen“ Anforderungen und diese nur soweit wie nötig detailliert werden.
- Für kleine Themen ist eine zweistufige Fachkonzepterstellung nicht zweckmäßig
  - Bei Änderungen an bestehender SW wird jetzt schon standardmäßig keine neue Anforderungsqualifizierung durchlaufen (siehe Prozessmodell).
  - Vereinfachte Prozesse und „reduzierte“ Vorlagen sind für kleine Themen nötig.

Enterprise Application Management

- Die fachliche Bebauung ist weitgehend erfasst, es mangelt jedoch noch an einer durchgängigen Werkzeugunterstützung (siehe auch Folgefolie).

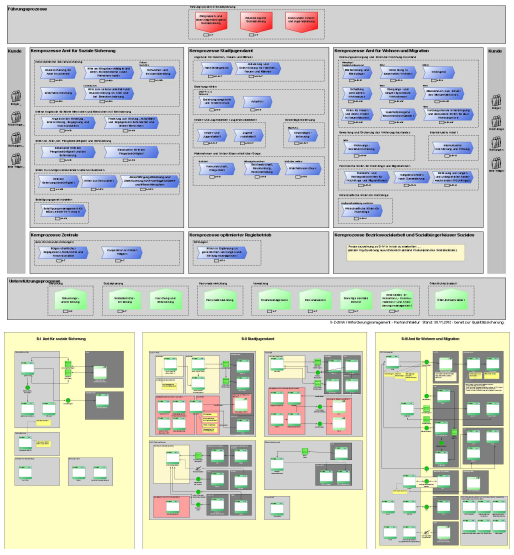
„Make, Buy, Use oder Compose?“

Landeshauptstadt München

Dr. Friedrich Strauß, Helmut Siemon

Seite 20

Prozesslandkarte* & IT-Anwendungsbebauung* im Sozialreferat der LHM unterstützen bei Projektscoping und MBUC – decken aber noch nicht alle wichtigen Bereiche ab



- Auch die Arbeitsversionen der Bebauungen helfen beim Strukturieren von IT-Projekten
  - Die Soll-Bebauung reflektiert derzeit nur die schon laufenden IT-Projekte
- Die derzeitigen Tools der Facharchitekten sind noch nicht mit den entsprechenden Tools der IT-Architekten gekoppelt
- Die EAM-Fragestellungen ändern sich mit der Zeit, so dass mittelfristig ein flexibleres Repository benötigt wird
- Die IT-Anwendungen sind den unterstützten Geschäftsprozessen zugeordnet
  - Offen ist noch die Berücksichtigung von *externen* Anwendungen, die im Sozialreferat genutzt werden

* Beide Karten stellen unfertige Arbeitsstände dar

„Make, Buy, Use oder Compose?“

Dr. Friedrich Strauß, Helmut Siemon

Landeshauptstadt München

Seite 21

Autoren

Dr. Friedrich Strauß

- Facharchitekt im Sozialreferat (seit Sommer 2011)
- Mitarbeiter im Programm MiT-KonkreT (u.a. Detaillierung Prozessmodell)
- Über 15 Jahre Berufserfahrung als fachlicher Chef-Designer, Projektmanager, Projektleiter bei der software design & management AG (sd&m) jetzt Capgemini
- 1995 Dissertation an der Albert-Ludwigs-Universität Freiburg

Helmut Siemon

- IT-Architekt bei it@M GBA, zuständig für das Sozialreferat (seit Herbst 2011)
- Mitarbeiter im Programm MiT-KonkreT (u.a. Detaillierung Prozessmodell)
- 13 Jahre Berufserfahrung als Senior Manager und technischer Architekt bei Accenture

eBAföG – ein ehrlicher Erfahrungsbericht

Kai Bürger

Referat I B 4
Bundesverwaltungsamt
50728 Köln
kai.buerger@bva.bund.de

Abstract: Über sechs Jahre wurde an der Einführung einer neuen Fachanwendung inklusive einer Datenmigration gearbeitet. Der Weg zum Projekterfolg führte über so manche Irr- und viele Königswege. Die gemachten Erfahrungen sind größtenteils übertragbar auf vergleichbare Entwicklungsprojekte. Dieser Vortrag soll dazu dienen, andere insbesondere vor den Irrwegen zu bewahren.

1. Das Ziel des Projektes eBAföG

Im Jahr 2005 stand das Bundesverwaltungsamt vor der Aufgabe, alle auf dem Großrechner betriebenen Anwendungen durch moderne, webbasierte Anwendungen abzulösen. Eine dieser Anwendungen war die Fachanwendung BAföG mit deren Hilfe das Bundesverwaltungsamt die Rückforderung und Verwaltung von Ausbildungsdarlehen abwickelt (ca. 200 Benutzer, ca. 600.000 Postausgänge, ca. 600.000 Posteingänge).

Die Zielvorgabe war, am 31.12.2011 die großrechnerbasierten Verfahrensteile abgelöst zu haben.

2. Die Lösung

Nach einer europaweit durchgeführten Ausschreibung wurden in der Phase der Anforderungsanalyse die funktionalen und nicht-funktionalen Anforderungen definiert. Im zweiten Schritt wurden in der Migrationsstudie die Architektur und die Migrationsstrategie erarbeitet. Mit einer Verzögerung von zwei[1] Monaten konnte das vorgegebene Ziel erreicht werden und seit Ende Juni 2012 sind die großrechnerbasierten Verfahrensteile abgelöst. Die Anwendung läuft seitdem stabil und fehlerfrei.

[1] Die eigentliche Inbetriebnahme musste durch projektexterne Einflussfaktoren auf Juni verschoben werden.

3. Irr- und Königswege

3.1 Methodik

In der Phase der Migrationsstudie war die Entscheidung über das Vorgehen zu fällen. Es wurden folgende Vorgehensweisen betrachtet: Big Bang, Big Bang mit einer datenabhängigen Einführung und eine schrittweise Einführung. Es wurden die jeweiligen Vor- und Nachteile gegenübergestellt. Die wichtigsten Entscheidungsmerkmale waren die Steuerbarkeit, die Verbindlichkeit in der Terminplanung und die Komplexität. Der entscheidende Nachteil der schrittweisen Einführung, die zeitweise Nutzung von zwei Systemen, wurde durch Verbesserungen für die Anwender kompensiert.

Während der Spezifikationsphase des zweiten Releases drohten die Aufwände zu explodieren. Von einem der Irrwege – auf die Aufnahme des ISTs zu verzichten, weil das System gut dokumentiert war – musste wieder auf die Erfolgsspur zurückgekehrt werden. Es musste eine Vereinbarung auf Management-Ebene geschlossen werden, dass grundsätzlich die bestehenden Funktionen 1:1 umzusetzen sind.

Einer der Königswege war die frühzeitige Einbeziehung der Nutzer. In der schrittweisen Einführung lag die Ursache, dass über einen längeren Zeitraum der Anwender mit zwei Systemen arbeiten musste. Aus diesem Grund wurden bewusst für jedes Release Funktionalitäten vorgesehen, die für den Anwender eine Verbesserung gegenüber dem Altsystem darstellten. Dies hat zu einer positiven Akzeptanz geführt.

Insbesondere während der ersten beiden Phasen (Anforderungsanalyse, Migrationsstudie) des Projektes existierte eine starke Management-Attention. Diese war mit eine Ursache für die schrittweise Einführung und dadurch mit verantwortlich für den Projekterfolg. Auch während schwieriger Projektphasen (siehe drohende Explosion der Aufwände) war die Unterstützung des Managements gegeben. Im weiteren Projektverlauf wurde die Management-Attention auf das übliche Niveau reduziert und schaffte Raum für eine gute Projektarbeit.

Das Projekt wurde grundsätzlich nach dem V-Modell XT abgewickelt. Besonders positiv bemerkbar auf das Projekt hat sich die Anforderung der Nachverfolgbarkeit gemacht. Dies hat in diesem Projekt zu einer sehr wertvollen und langfristig nutzbaren Systemdokumentation geführt. Allerdings hätte gemäß V-Modell XT nach der Migrationsstudie eine erneute Ausschreibung stattfinden müssen. Zwei Gründe sprachen dagegen: Erstens war für eine Ausschreibung die Anforderungsanalyse nicht präzise genug. Zweitens hätte dies dazu geführt, dass der Endtermin nicht haltbar gewesen wäre.

Aus den Projektmanagement-Methoden ist das Thema Risikomanagement hervorzuheben. Hier ergänzten die Methoden des Großprojektmanagements des BVAs perfekt die Vorgaben aus dem V-Modell XT. Das Risikomanagement wurde als Arbeitsmittel verstanden und sorgte für entsprechende Transparenz.

An dem Projekt arbeiteten zwei Organisationseinheiten beim Auftraggeber mit einem externen Auftragnehmer, dieser temporär mit Subunternehmern, zusammen. Darüber

hinaus waren weitere Organisationseinheiten beteiligt. Ohne eine klare Trennung von Verantwortung und Rollen wäre das Chaos vorprogrammiert gewesen.

3.2 Architektur und Technologie

Während der Projektlaufzeit mussten unvermeidlich Architekturentscheidungen überdacht werden (z.B. Schnittstellen zu Nachbarsystemen per Web Service). Auch hier erwies sich die schrittweise Einführung von Vorteil. So konnten während des zweiten Releases Batches anstelle von Web Services mit Konvertern eingesetzt werden.

Auf der anderen Seite hat es sich gelohnt, Entscheidungen durchzusetzen, auch wenn der Weg am Anfang sehr beschwerlich war. So hat sich die Umsetzung der Single-Sign-On-Lösung als sehr positiv herausgestellt. Dies gilt ebenfalls für den Einsatz einer Business-Intelligence-Komponente.

4 Fazit

Ob die Entscheidungen (methodisch und architektonisch) wirklich richtig waren, kann sich erst im Produktivbetrieb von Softwaresystemen zeigen. Die Anwendung eBAföG läuft seit Einführung stabil und fehlerfrei. In dem Projekt wurden wichtige Erfahrungen gesammelt. Diese sind insbesondere für vergleichbare, entsprechend große und komplexe Projekte sehr wertvoll. Hervorzuheben ist die schrittweise Einführung als eine der wichtigsten Entscheidungen. Resultat dieses Vorgehens ist auch, dass verhältnismäßig wenige Änderungsanträge umgesetzt werden mussten.

Bundesverwaltungsamt

Bundesverwaltungsamt

Der zentrale Dienstleister des Bundes

Software & Systems Engineering Essentials 2013:
eBAföG – ein ehrlicher Erfahrungsbericht

27./28.02./01.03.2013

Bundesverwaltungsamt

Agenda

1. Aufgabenstellung
2. Herangehensweise und Lösung
3. Irr- und Königswege
   - Methodisch
   - Technisch
4. Fazit

Bundesverwaltungsamt

1. Aufgabenstellung: Ausgangslage

- **Massenverfahren**
  - Einziehung des BAföG-Staatsdarlehens
- ca. 200 Anwender an 3 Standorten
- 6 Schnittstellenpartner
- ca. 350.000 Lines of Code
- seit 1983 aufgebautes System
- ca. 90 Mio. Datensätze
- BS2000 (**Großrechner**), ADABAS/Natural-Basis

Bundesverwaltungsamt

1. Aufgabenstellung: Ziele

- Ablösung bis zum 31.12.2011
- plattformunabhängige, langfristig einsetzbare Technologie
- höhere Flexibilität bei fachlichen Änderungen
- Ausweitung des eGovernment-Angebotes
- Verbesserung der Verwaltungs- und Abwicklungsprozesse

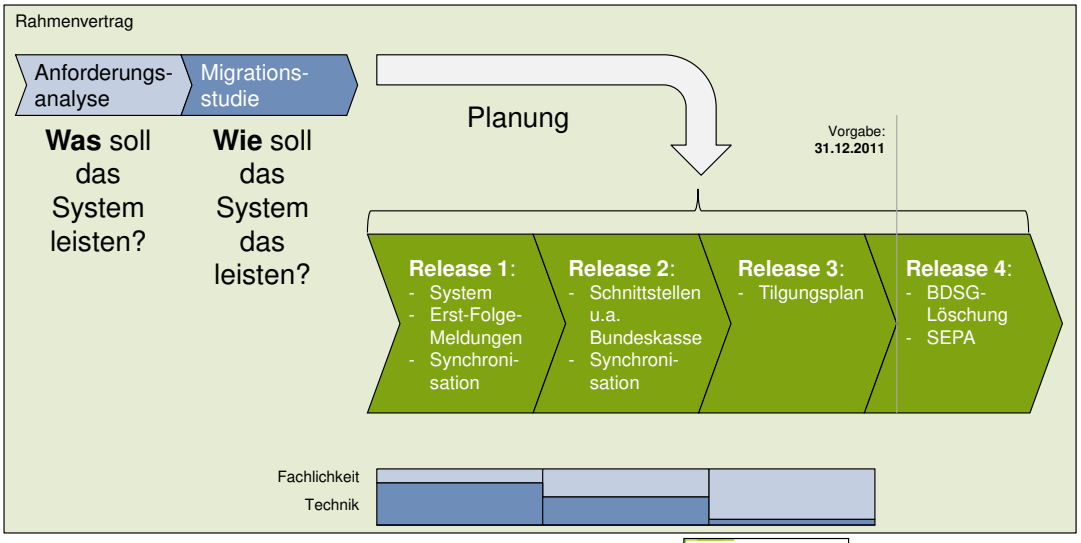
Bundesverwaltungsamt

Agenda

1. Aufgabenstellung
2. Herangehensweise und Lösung
3. Irr- und Königswege
   - Methodisch
   - Technisch
4. Fazit

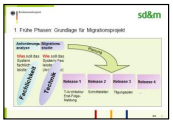
Bundesverwaltungsamt

2. Herangehensweise und Lösung



Bundesverwaltungsamt
Der zentrale Dienstleister des Bundes

Siehe SEE 2007…



Bundesverwaltungsamt

Agenda

1. Aufgabenstellung
2. Herangehensweise und Lösung
3. Irr- und Königswege
   - Methodisch
   - Technisch
4. Fazit

Bundesverwaltungsamt

3.1 Irrweg: Big Bang

- 1. Idee: Big Bang
- 2. Idee: Big Bang + datenabhängige Einführung
- **3. Idee: Schrittweise Einführung nach Funktionen (Releases)**

Die Inbetriebnahmen ab Release 2 verliefen problemlos!

Vorteile	Nachteile
Risikominimierung und Steuerbarkeit	Kosten (Synchronisation)
Verbindlichkeit in Terminplanung und Test	Komplexität (Synchronisation, Fachbereich)
Flexibilität bzgl. Änderungen	„Kunst des optimalen Schnittes“

Bundesverwaltungsamt

3.2 Irrweg: Verzicht auf IST-Aufnahme

- Verzicht auf IST-Aufnahme
- Drohende Aufwandsexplosion während Spezifikationsphase Release 2
- Maßnahme:
  - Basis: bestehende Anwendung + Verbesserungen

Vorteile	Nachteile
Aufwandsexplosion vermieden	Einschränkung der Kreativität (Spezifikation)
Effiziente Spezifikation	
Termintreue erreicht	

Bundesverwaltungsamt

3.2 Königsweg: Einbeziehung der Nutzer

- Ziel: Bereits mit den ersten beiden Releases Verbesserungen umsetzen
  - Release 1: Automatische Adressänderungen
  - Release 2: Integrierte Kontoanzeige
- Änderungswünsche gesammelt / priorisiert

Vorteile	Nachteile
Akzeptanz erhöht	
Hemmschwelle abgebaut	
Schulungsaufwand reduziert	

Bundesverwaltungsamt

3.3 Königsweg: Management-Attention

- Zu Beginn: Intensiv
- Monatliche Projektsteuerungsgremium-Sitzung
- Monatlicher Statusbericht
- Eskalation bei Bedarf

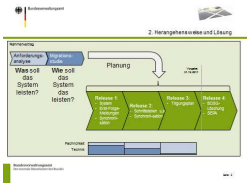
Selbständige Projektarbeit mit Rückendeckung!

Vorteile	Nachteile
Kontinuierliche Unterstützung	
Kontinuierliche Informationsstände	
Selbständige Projektarbeit	

Bundesverwaltungsamt

3.4 Königsweg: V-Modell XT

- Abwicklung gemäß V-Modell XT
- Nachverfolgbarkeit:
  - Systemdokum. auf Basis der Spezifikation
- Abweichung:
  - Vorgehen: Keine Aufteilung des Projektes



Vorteile	Nachteile
Wertvolle Systemdokumentation	Wettbewerb eingeschränkt
Zeitersparnis	

Bundesverwaltungsamt

3.5 Königsweg: Risikomanagement

- Mind. monatliche Aktualisierung
- Integriert in Statusbericht
- Agenda-Punkt in jeder PStG-Sitzung

Vorteile	Nachteile
Transparenz	
Effiziente Projektsteuerung	

Bundesverwaltungsamt

3.6 Königsweg: Trennung von Verantwortlichkeiten

- AG:
  - IT: Gesamtverantwortung (inkl. Budget), Technik
  - FB: Feinspezifikation, Abnahmetest (inkl. Planung)
- AN:
  - Umsetzung der Anforderungen

Vorteile	Nachteile
Klare Verantwortlichkeiten/Ansprechpartner	Kommunikationsaufwand (z.B. PL-JF)
Unterschiedliche Ziele aneinander angeglichen	

Bundesverwaltungsamt

Agenda

1. Aufgabenstellung
2. Herangehensweise und Lösung
3. Irr- und Königswege
   - Methodisch
   - Technisch
4. Fazit

Bundesverwaltungsamt

4 Irr- und Königswege: Technisch

- Unvermeidlich: Überdenken von Architekturentscheidungen
- Positiv: schrittweise Einführung hat die Möglichkeit eröffnet

Irrweg	Best Practices
Batches mittels Web-Service	Batches ohne Web-Services
	Single-Sign-On
	Business-Intelligence-Komponente

Bundesverwaltungsamt

Agenda

1. Aufgabenstellung
2. Herangehensweise und Lösung
3. Irr- und Königswege
   - Methodisch
   - Technisch
4. Fazit

Bundesverwaltungsamt

5 Fazit

- Gut strukturiertes Vorgehen (schrittweise Einführung)
- Maßnahmen bei Problemen / Flexibilität
- Engagiertes Team

➔Zwei reibungslose Inbetriebnahmen

➔Änderungsanträge im Verhältnis des Gesamtvolumens: ca. 3%

7. Mobile Systeme

Sessionüberblick

7.1. Key considerations for high-quality mobile systems (28.02.2013) 184

Key considerations for high-quality mobile systems

Sven Euteneuer

SQS Research
SQS Software Quality Systems AG
Stollwerckstrasse 11
51149 Köln
sven.euteneuer@sqs.com

Abstract: Mobile computing is currently seen as one of the key trends. Many adopters question how to adapt established quality assurance techniques to the mobile ecosystem. This paper identifies the key factors that require adaptation when conducting testing and quality assurance in a mobile context and gives guidance on how to tackle them.

1 Introduction

With recent software glitches such as international blackouts and new handsets not functioning as expected, software failures in the mobile industry have been hitting the headlines regularly for the past 18 months. These software glitches can ruin mobile device manufacturers', operators' and developers' reputations, cause inconvenience for consumers and rack up hefty costs for fixes.

Additionally, businesses now use mobile systems on a daily basis, from checking emails and organizing diaries to internet browsing and virtual file sharing. Therefore, companies must consider what testing criteria are required to ensure glitch-free usage of their mobile systems. The following sections discuss key considerations that need to be considered and aim to give high-level guidance on how to tackle them in real-life projects.

2 The Mobile Quality Fingerprint

In principle, the quality criteria of mobile and traditional software are no different to those of conventional software. The focus, however, shifts to functionality, security, performance, ease of use, reliability, portability and maintainability. These criteria need to be weighted differently for mobile solutions creating a different "Quality Fingerprint". Mobile software, for example, has to be as sparing as possible on hardware resources, to extend the battery life.

The subject of security also needs to be taken even more seriously with mobile systems than with traditional IT, as mobile devices come with substantially more interfaces. In terms of hardware, this may be WiFi, in terms of software for instance social media apps. Mobile systems permanently exchange data over the internet via these interfaces and each additional interface increases the security risk and is a potential gateway into the device, and thus into the company's IT.

On a high-level, the definition of appropriate quality objectives needs to be put in place so that subsequent activities of the software development lifecycle can be prioritized, controlled and managed according to these objectives. Widely accepted standards such as the ISO/IEC 25010 can serve as a basis for such a selection and prioritization. In a subsequent step, each quality objective is then used to specify relevant testing methods and techniques.

3 The Mobile Test Items

Compared to stationary computers, mobile systems bring with them a range of new requirements that need to be taken into account when preparing for testing. Starting with the prioritized quality objectives, requirements are clustered and test items identified. This usually gives rise to a number of test items that are unique for mobile devices. Smartphones, for example, are generally not connected with the rest of the company via a secure network, but use GSM, UMTS or a public WLAN, depending on the location. In addition, there is often a range of different, some privately purchased, devices.

Other examples for mobile-specific test items include items focusing mobile operation, so that the quality objective of “frugal use of energy” is supported by concrete test items such as CPU load or screen use times. Additionally, test items associated with the user interface are bound to differ for mobile devices as they are most often used without mouse or keyboard.

On the other hand, the basic principles of software quality assurance and software testing remain the same with mobile systems (cf. [SpL05]). Existing testing methods are often suitable for use in the mobile domain with little or no adaptation. However, the concrete implementation of such techniques may differ, requiring unique test tools, test environments or test data.

New testing tools need to be considered to be able to cater to the mobile technology but also to test quality criteria such as security or efficiency, which are often not in scope for conventional software projects. One example of this is "fuzzing", which bombards the interfaces of mobile systems in a kind of stress test, with the aim of breaching them. In this way, "fuzzing" can detect any problems in the areas of security or robustness.

4 The Mechanics of the Mobile Ecosystem

When developing and introducing mobile systems, flexible process models are essential, as mobile products change more quickly and frequently than traditional information technology. Iterative and incremental developmental models are recommended, as they test the system requirements and their implementation much more frequently than sequential procedures.

In the selected process model, the user should also be closely involved in the definition of requirements. Combined with early definition of requirements and a systematic approach to quality assurance overall, this substantially reduces the time to market. The testing function needs to collaborate with the remainder of the team's roles to reach this goal. In some cases this requires adapting or changing existing processes and procedures.

Additionally, mobile devices rely on a technically different approach to marketing, rolling out and deploying software releases, impacting the whole process of software creation. Testing needs to support this changed deployment by including the relevant scenarios in installability testing.

5 Multi-Channel Development and Testing

The final consideration arises from the relative immaturity of the mobile device market. As of today there still are many different platforms with an even larger number of concrete devices available, differing in aspects such as OS/technology stack, hardware architecture, form factor, etc.

Even though solutions exist that attempt to cover multiple of these platforms at once, it is still quite difficult to write a single app targeting multiple platforms at the same time. On the other hand, however, the differences between platforms are often purely technical (such as the difference between Apple's iOS API and Google's Android API) while the core functionality remains largely the same, only catering for different user interaction patterns or form factors.

In these cases, systematic approaches to test planning and design can contribute to increasing the efficiency and agility of testing. This is achieved by systematically identifying core functionality and platform-specific functionality at an early stage. This information can then be used to guide development and testing as common functionality may be provided only once, eliminating the need for multiple high-level test cases.

6 Literature

[SpL05] Spillner, A.; Linz, Tilo: Basiswissen Softwaretest. dpunkt-Verlag, 2005.

Agenda

- Intro
- Mobile Computing: Driving Change in the Testing Business
- Key Differentiators to Business As Usual
- Resulting Challenges
- Summary
- Q&A

Intro

At a glance: SQS is the world's leading specialist in software quality.

» *The global leader in independent software testing and quality management services – majority of its business in Europe* «

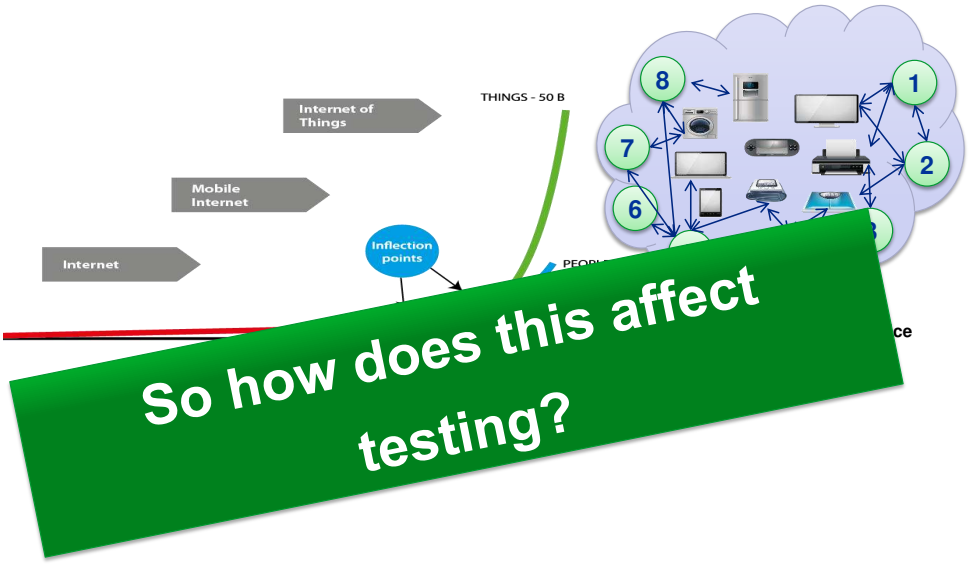
Financial Times, 21 August 2007

- More than 30 years of prosperous operations
- Over 7,000 completed projects
- The customer base includes 20 FTSE-100 companies, half of the DAX 30 companies and nearly a third of the STOXX-50 companies
- The SQS philosophy is to help our customers improve the quality of their business solutions.

SQS is listed on the AIM London

Mobile Computing: Driving Change in the Testing Business

Cloud and Mobile/Ubiquitous Computing drive some of the biggest changes we have seen in the last decade



Key Differentiators to Business As Usual

Mobile Testing differs from traditional testing in four key areas

The Mobile Quality Fingerprint

- Characterized by: Security, User Experience, Efficiency, Reliability
- Requiring different priorities and testing techniques

The Mobile Test Items

- Characterized by: Software delivery, changing networks, etc.
- Requiring different test cases, test data, test environments and test tools

The Mechanics of the Mobile Ecosystem

- Characterized by: Short innovation cycles, delivery to tight schedules
- Requiring highly systematic yet nimble project and test management approaches

Multi-Channel S/W Development and Testing

- Characterized by: Value-add across multiple, technically variant channels, generating high variability in target platforms
- Requires SPL-like approach to software engineering and testing in particular

Resulting Challenges

Mobile Testing differs from traditional testing in four key areas

The Mobile Quality Fingerprint

- Characterized by: Security, User Experience, Efficiency, Reliability
- Requiring different priorities and testing techniques

The Mobile Test Items

- Characterized by: Software delivery, changing networks, etc.
- Requiring different test cases, test data, test environments and test tools

The Mechanics of the Mobile Ecosystem

- Characterized by: Short innovation cycles, delivery to tight schedules
- Requiring highly systematic yet nimble project and test management approaches

Multi-Channel S/W Development and Testing

- Characterized by: Value-add across multiple, technically variant channels, generating high variability in target platforms
- Requires SPL-like approach to software engineering and testing in particular

Resulting Challenges

The Mobile Quality Fingerprint

Resulting Challenges

Mobile Testing differs from traditional testing in four key areas

The Mobile Quality Fingerprint

- Characterized by: Security, User Experience, Efficiency, Reliability
- Requiring different priorities and testing techniques

The Mobile Test Items

- Characterized by: Software delivery, changing networks, etc.
- Requiring different test cases, test data, test environments and test tools

The Mechanics of the Mobile Ecosystem

- Characterized by: Short innovation cycles, delivery to tight schedules
- Requiring highly systematic yet nimble project and test management approaches

Multi-Channel S/W Development and Testing

- Characterized by: Value-add across multiple, technically variant channels, generating high variability in target platforms
- Requires SPL-like approach to software engineering and testing in particular

Resulting Challenges

Mobile Test Items – Different Test Items and Test Cases

Network Constraints

- Network throughput
- Fidelity
- Network hopping, hand-over

Mobile Operation

- Finite amount of charge
- Need for frugal use of resources such as CPU, display

User Interface

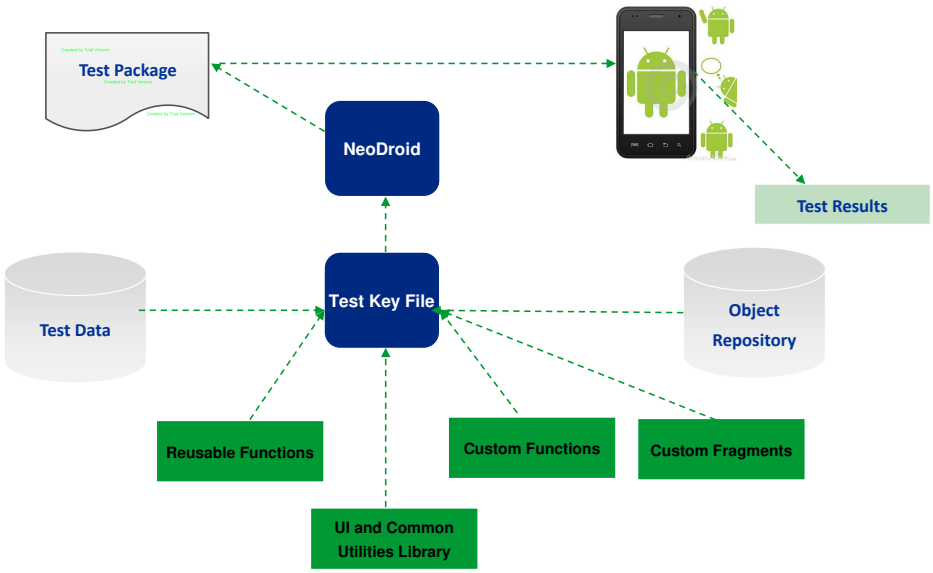
- (Relatively) small screen
- Touch-based
- No mouse
- Often only soft-keys

Differing S/W Delivery

- (Closed) App store concept
- Installation via centralized download
- Differing S/W product lifecycle

Resulting Challenges

Mobile Test Items – Different Mobile Test Toolchain



Resulting Challenges

Mobile Test Items – Different Mobile Test Environments

Test Types		Manual Test		Mobile Test Automation
		Device	Emulator	Device
Functional	Unit testing	No	Yes	Yes
	Integration testing	No	Yes	Yes
	System testing	Yes	No	Yes
	Regression testing	Yes	No	Yes
Non-functional	Compatibilty testing	Yes	No	Yes
	GUI-test	Yes	No	Yes
	Performance testing	Yes	No	No
	Security testing	Yes	No	Yes
	Synchronisation testing	Yes	No	No

Resulting Challenges

Mobile Testing differs from traditional testing in four key areas

The Mobile Quality Fingerprint

- Characterized by: Security, User Experience, Efficiency, Reliability
- Requiring different priorities and testing techniques

The Mobile Test Items

- Characterized by: Software delivery, changing networks, etc.
- Requiring different test cases, test data, test environments and test tools

The Mechanics of the Mobile Ecosystem

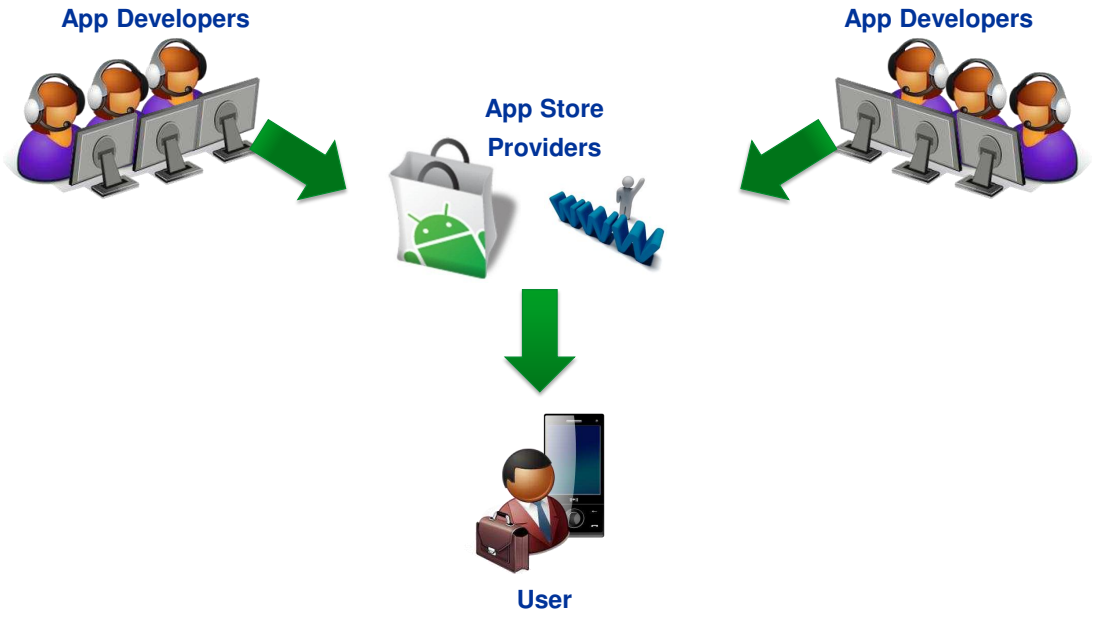
- Characterized by: Short innovation cycles, delivery to tight schedules
- Requiring highly systematic yet nimble project and test management approaches

Multi-Channel S/W Development and Testing

- Characterized by: Value-add across multiple, technically variant channels, generating high variability in target platforms
- Requires SPL-like approach to software engineering and testing in particular

Resulting Challenges

The Mobile Ecosystem



Resulting Challenges

Mobile Testing differs from traditional testing in four key areas

The Mobile Quality Fingerprint

- Characterized by: Security, User Experience, Efficiency, Reliability
- Requiring different priorities and testing techniques

The Mobile Test Items

- Characterized by: Software delivery, changing networks, etc.
- Requiring different test cases, test data, test environments and test tools

The Mechanics of the Mobile Ecosystem

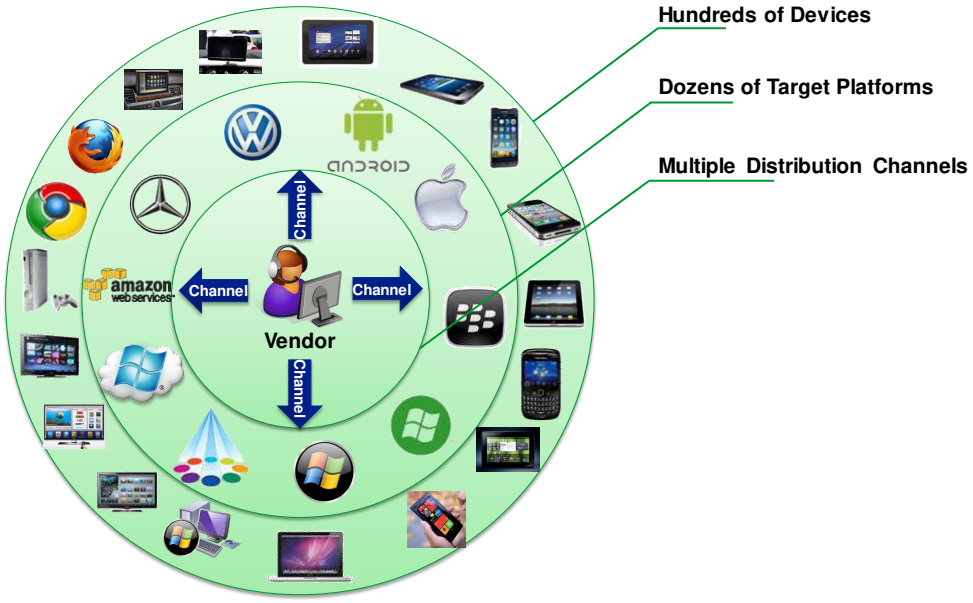
- Characterized by: Short innovation cycles, delivery to tight schedules
- Requiring highly systematic yet nimble project and test management approaches

Multi-Channel S/W Development and Testing

- Characterized by: Value-add across multiple, technically variant channels, generating high variability in target platforms
- Requires SPL-like approach to software engineering and testing in particular

Resulting Challenges

What is Multi-Channel about?



Resulting Challenges

Multi-Channel S/W Development: A Definition

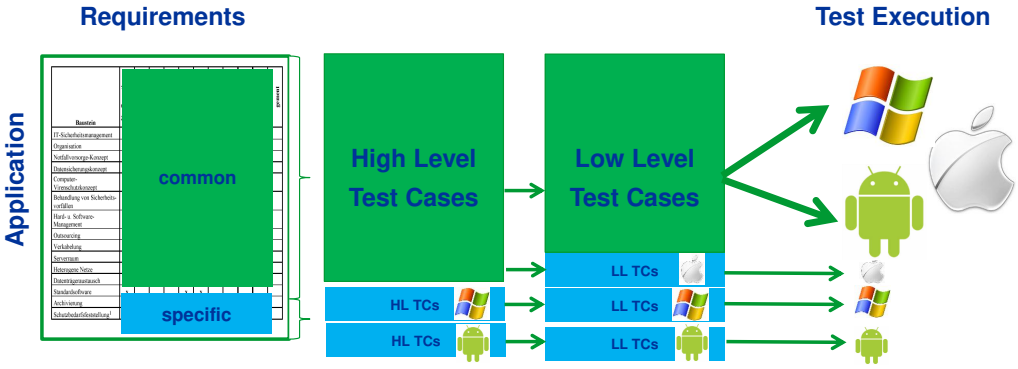
The term multi-channel development (MCD) characterizes the act of **developing software products** for distributing their value-add through **multiple different channels** with **identical subsets of functionality**, resulting in a **potentially large number of target platforms**.

A set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.

(Source: CMU Software Engineering Institute)

Resulting Challenges

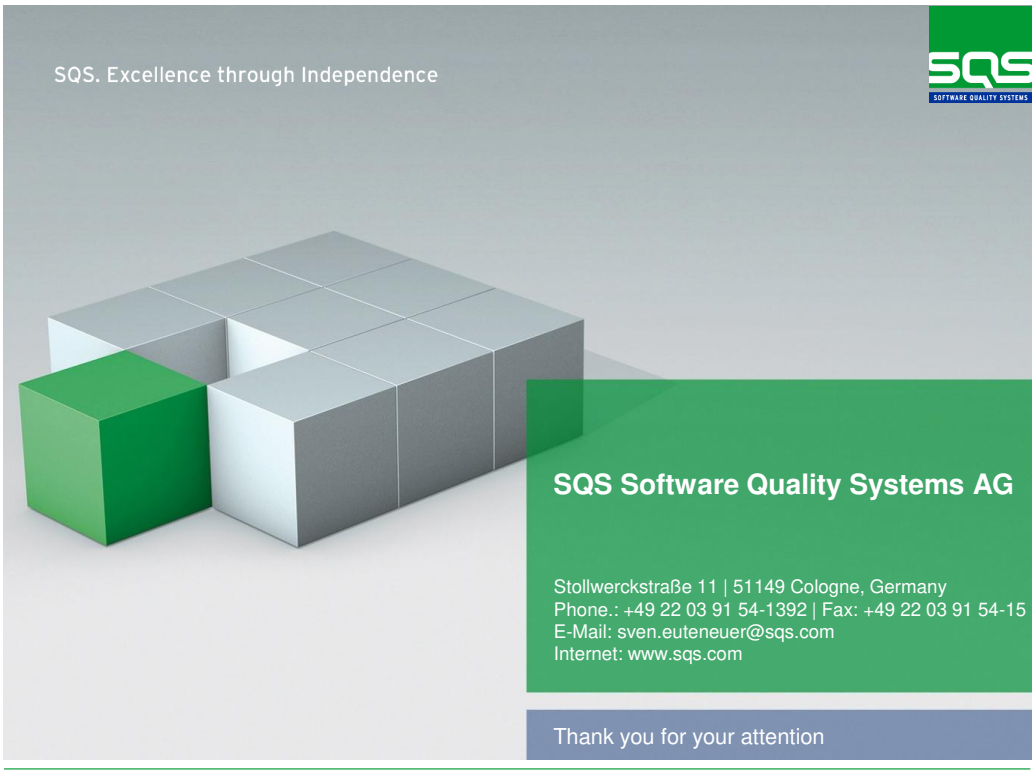
Multi-Channel Testing: From Requirements to Testware



Summary

Testing for High-Quality Mobile Systems

- Testing constitutes a key component of successful quality assurance and, thus, ensures high-quality mobile systems
- Testing for mobile device software does not differ dramatically from established concepts, having been around for 30+ years
- Some key extensions and adaptations are necessary, however, to derive maximum value (i.e. effectiveness and efficiency!)
- Open issues are mainly around maturity of SME and developing organizations, the former not knowing (yet) what they need and the latter not having much process maturity


SQS. Excellence through Independence
SQS
SOFTWARE QUALITY SYSTEMS
SQS Software Quality Systems AG
Stollwerckstraße 11 | 51149 Cologne, Germany
Phone.: +49 22 03 91 54-1392 | Fax: +49 22 03 91 54-15
E-Mail: sven.euteneuer@sqs.com
Internet: www.sqs.com
Thank you for your attention

8. Architektur

Sessionüberblick

8.1. Essential Architectural Views for Embedded Systems (01.03.2013) 199

Essential Architectural Views for Embedded Systems

Dr. Jens Liebehenschel
metio GmbH
Sulzbacher Straße 105
65835 Liederbach
jens.liebehenschel@metio.de

Abstract: The necessity of architecture documentation for embedded systems is given by standards, norms, and process frameworks. Necessary architectural views are determined by the concerns of the stakeholders in general. Certainly it is not defined how the views look like and what language and language elements are to be used. Often the effort is disproportionally high.
Especially for embedded systems there is a set of essential views. This paper describes those architectural views based on a small subset of the graphical modeling language SysML and gives examples. It also points out what additional views can be usefully combined with the others for reaching different and specific goals, also illustrated by examples.
This work is driven especially from the practical point of view. It enables the creation of useful documentation at reasonable effort. Thus it delivers a good starting point for the methodological approach when modeling embedded systems. The paper is based on the experience in developing safety critical systems.

1. Extended Abstract

The necessity of architecture development and documentation is indisputable for embedded systems. Efficient documentation is one of the key success factors. The importance is growing with the number of releases and variants. This is relevant for example for the automotive domain. Further, architecture documentation is required by the established standards and norms. If they are misinterpreted, then too much documentation is required by the development process. Often, this is not questioned, and then time is wasted for the creation of documentation with small benefit.

Anyhow, the methodological work of finding an appropriate way of documenting system and software architectures is often difficult due to several contradicting requirements for the documentation. On the one hand the architecture documentation shall be comprehensive, easy to handle, and compliant to applicable standards and norms. But on the other hand the effort for the documentation shall be not too high – for the initial creation and even more important for keeping it up-to-date.

The necessary architectural views are always driven by the stakeholders' concerns, which also include the project context, applicable standards and norms, and used process frameworks.

For embedded systems there is a crucial set of views that is necessary for many reasons like getting a system overview, becoming acquainted with the system, possessing a base for architectural discussions, and enable partitioning of work based on elements with defined interfaces. Even more important reasons will be discussed later.

The established modeling language SysML is used for the visualization. Only a very limited set of language elements is sufficient. The approach can also be applied with other languages supporting similar elements.

In the first part of the paper, the essential architectural views for embedded systems are described in detail. The structural views consist of

- a view that describes the system boundary and the connection to elements in the system environment and
- views for the involved development domains like software and hardware.

All views have in common that possible interaction between the elements is visible by signal flows. Certainly the views are closely connected. This complete system view of signal flows up to the system boundary is necessary for performing several safety analyses. They are absolutely mandatory for safety critical systems. Further the signal flow oriented software view discloses details about the possibilities for parallel execution on multicore processors. This will become more and more important in the future.

In the second part, several extensions to the essential structural views are described:

- Behavior (activation of elements and interaction between elements)
- Behavior (internal behavior of elements)
- Requirements
- Design decisions
- Traceability

With the structural views and an adequate behavioral view describing the schedule it is possible to gain insight in reaction times of the system. Incorporating the traceability between requirements, design decisions, and the architecture is efficient, because all aspects can be documented in one tool.

Based on a simple example, graphical views are shown for illustrating the simplicity of the methodological approach.

The paper closes with modeling tips that can also be used as a checklist for efficient modeling.

Essential Architectural Views for Embedded Systems

Dr. Jens Liebehenschel

metio
methodik · information · organisation

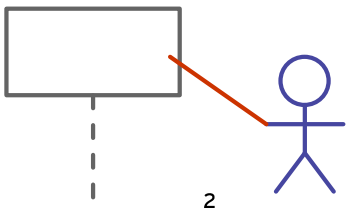
About metio

methodik · information · organisation

- System- and Software-Engineering
  - Architecture development, analysis and improvement

square-m

- Consulting
  - Method development
    - Architecture and product line approaches
- Training



2

metio
methodik · information · organisation

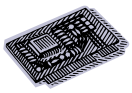
What views shall be used?

- Overview
- Essential Architectural Views
  - Structure Views
  - Behavior Views
  - Views Beyond
- Instead of a Summary

3

metio
methodik · information · organisation

Embedded Systems in the Scope

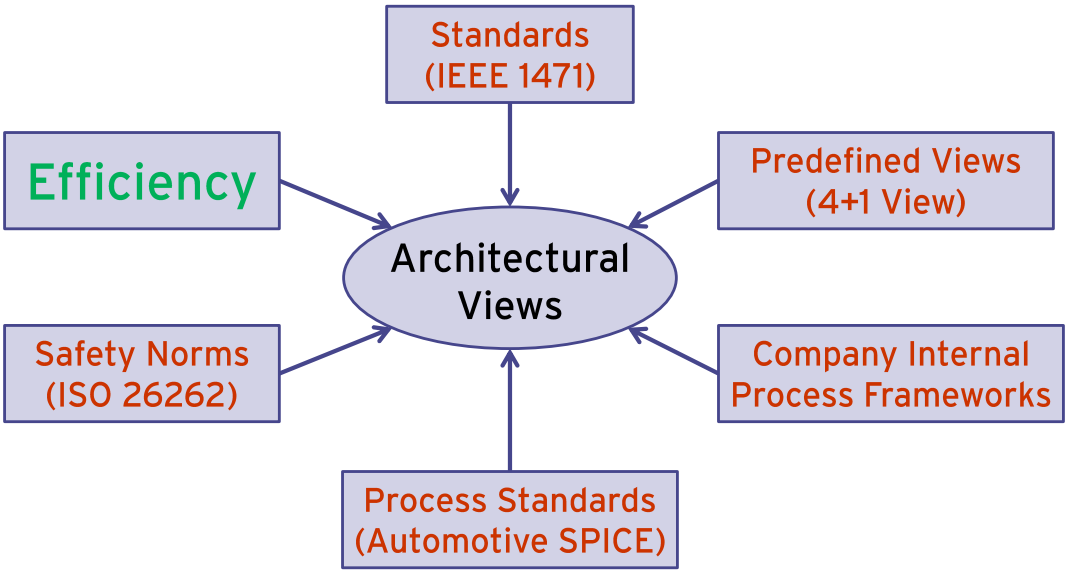


- Hard or soft real-time requirements
- Close interaction with specialized hardware
- No operator available
- Limited resources
- Important qualities
  - Availability
  - Safety
  - Security
  - Cost

4

metio
methodik · information · organisation

Challenge



5

metio
methodik · information · organisation

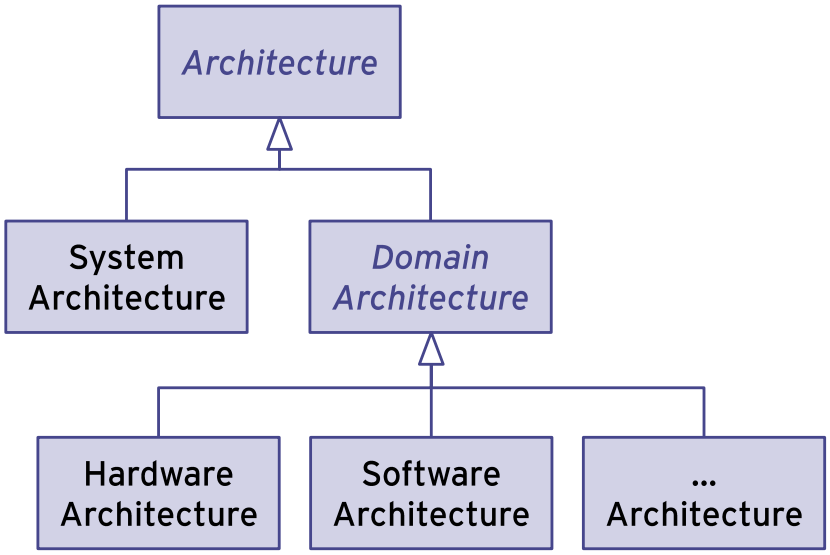
Used Modeling Language

- SysML
  - Standardized language
    - Several tools available
  - UML-based, but less complex than the UML
    - Model-view paradigm
  - Language elements
    - FlowPorts and Flows
    - Requirements and Traceability

6

metio
methodik · information · organisation

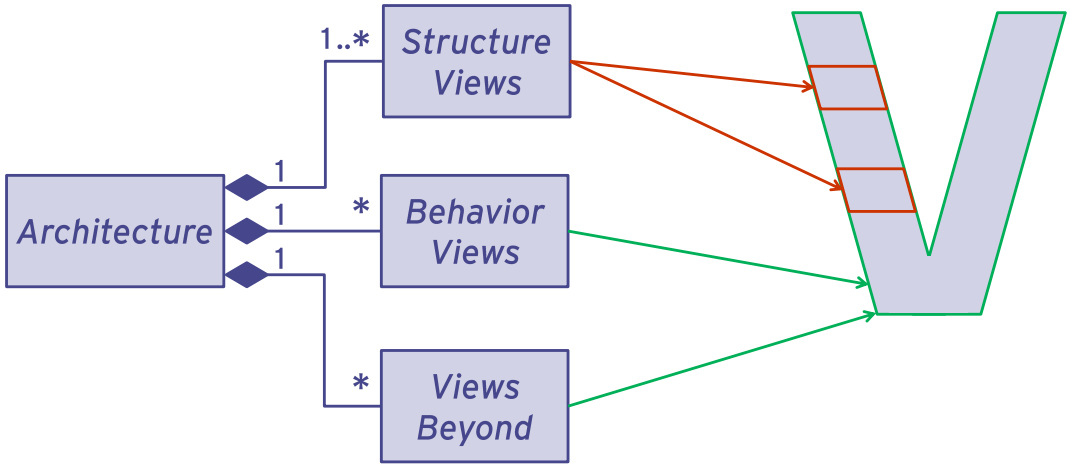
Embedded System Architectures



7

metio
methodik · information · organisation

Architecture Views in the V-Model

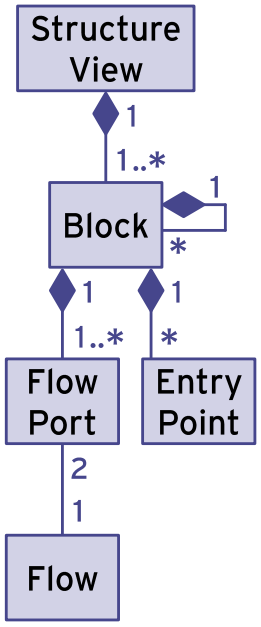


8

metio
methodik · information · organisation

Structure Views

- Signal flow oriented
- Internal Block Diagrams
- View content
  - Boundary of design level
  - Components
  - Interfaces
  - Connections
- Further refinement in domain specific engineering tools

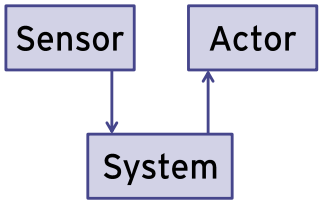


9

metio
methodik · information · organisation

Example

- System Environment
  - Sensor
  - Actor
- System: Electronic Control Unit
  - Housing
  - Printed circuit board
  - Micro controller
  - Software

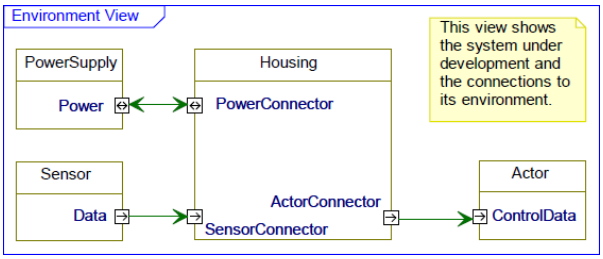


Structural refinement during design

10

metio
methodik · information · organisation

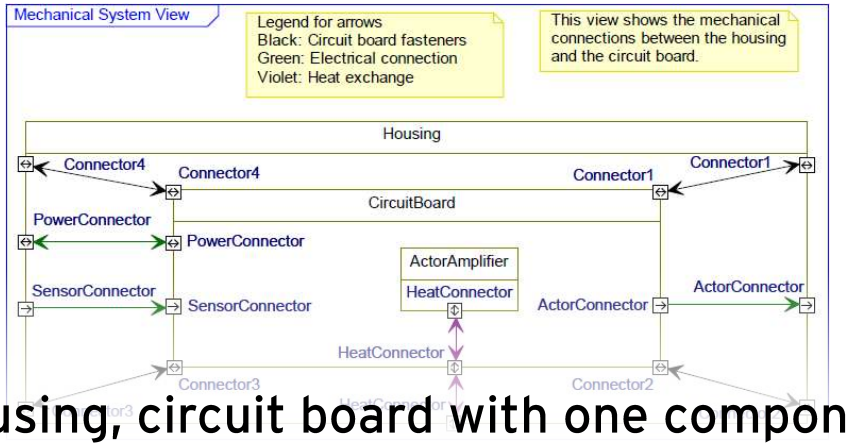
Environment View



- Electrical system boundary
  - Mechanical system boundary not shown here
- Components in the system environment

metio
methodik · information · organisation

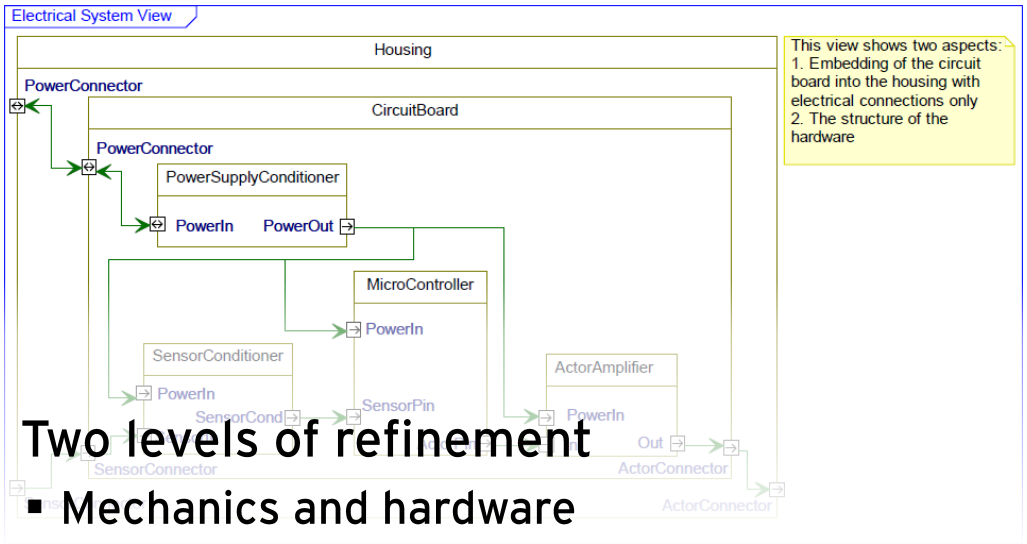
Mechanical System View



- Housing, circuit board with one component
- Different types of physical connections
  - Flow of currents, heat, forces

metio
methodik · information · organisation

Electrical System View

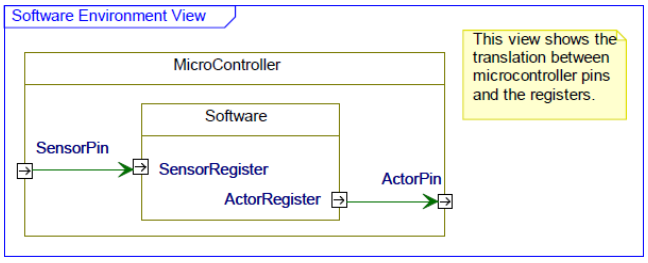


- Two levels of refinement
  - Mechanics and hardware
  - Hardware refinement

13

metio
methodik · information · organisation

Software Environment View

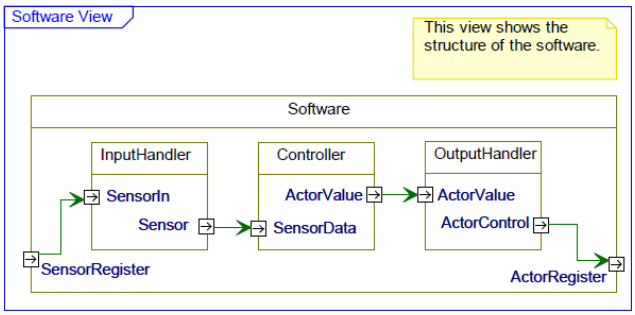


- Software inside the micro controller
- Assignment of micro controller pins to the software

14

metio
methodik · information · organisation

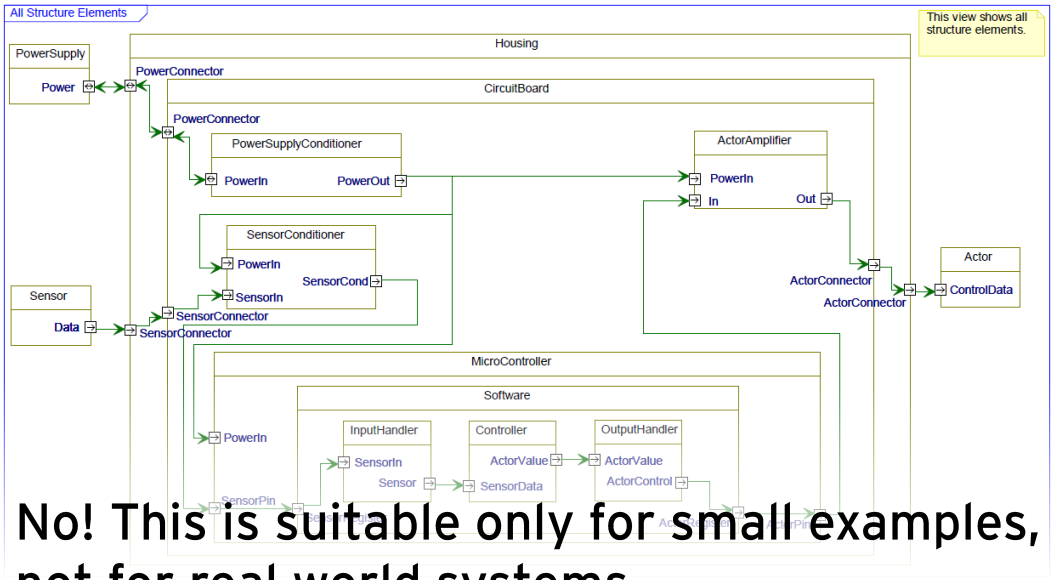
Software View



- Software refinement on top level
- Further refinement might be designed in this or other views

metio
methodik · information · organisation

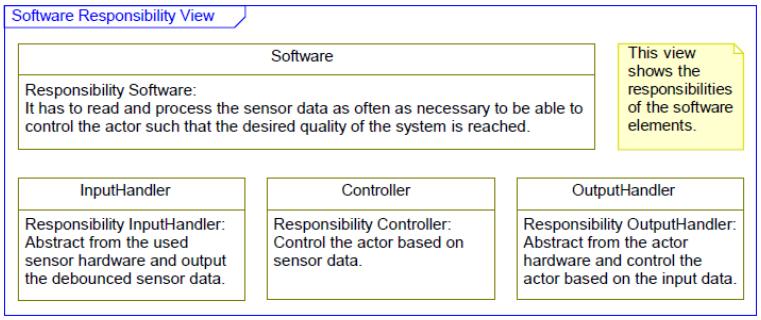
Putting all Structure Views Together?



- No! This is suitable only for small examples, not for real world systems

metio
methodik · information · organisation

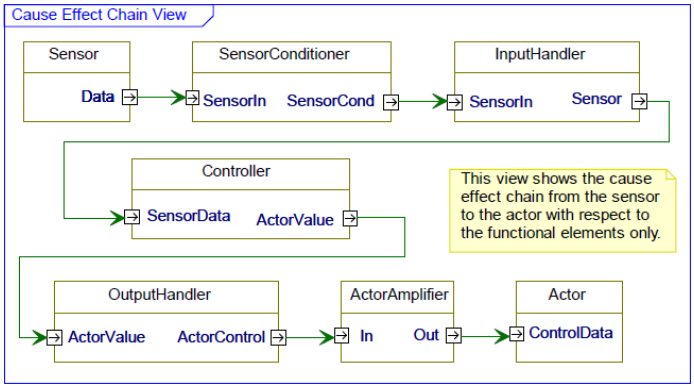
Software Responsibility View



- The responsibilities might also be shown in the structure views

metio
methodik · information · organisation

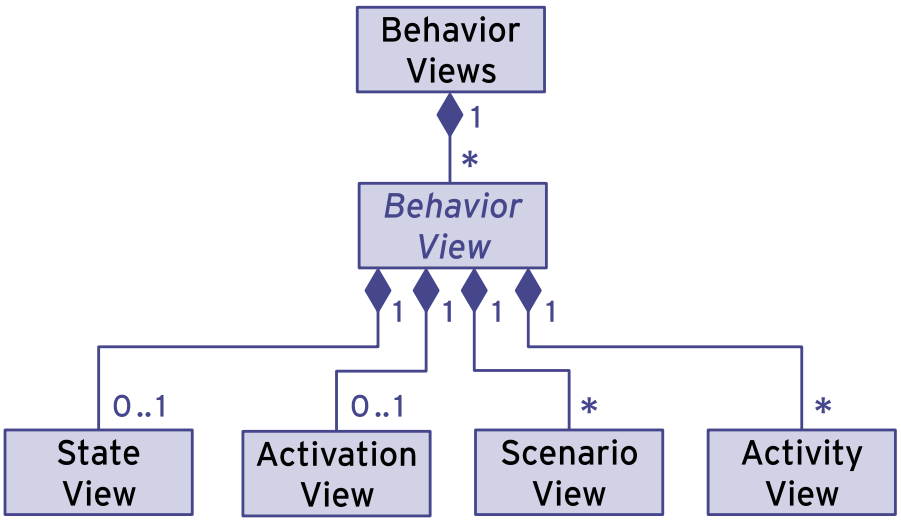
Cause Effect Chain View



- Logical view
  - View with elements of different domains

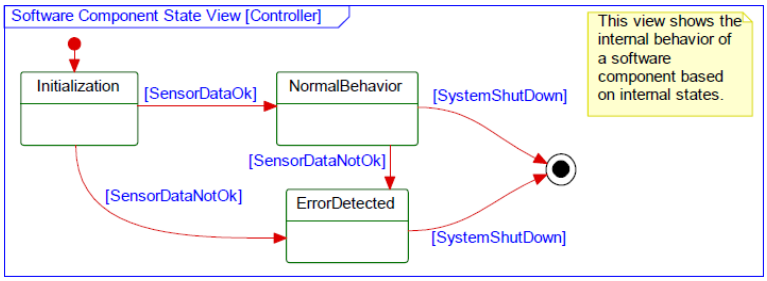
metio
methodik · information · organisation

Behavior Views



metio
methodik · information · organisation

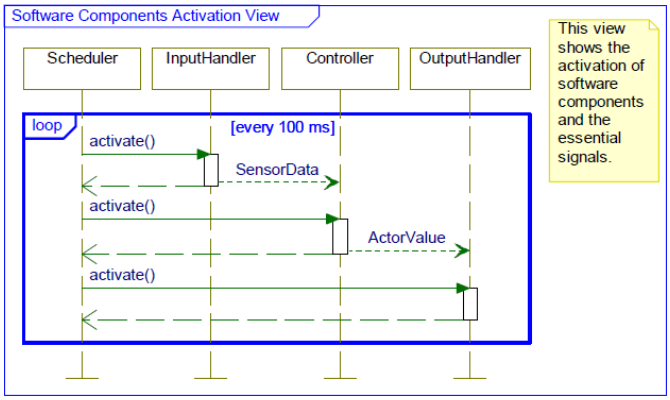
Software Component State View



- States of one component and the transition between the states

metio
methodik · information · organisation

Software Components Activation View

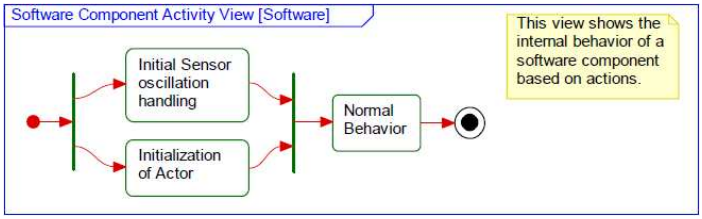


- Order and timing of component activations
- Software Components Scenario View
  - Focus on relevant components and signals

21

metio
methodik · information · organisation

Software Component Activity View

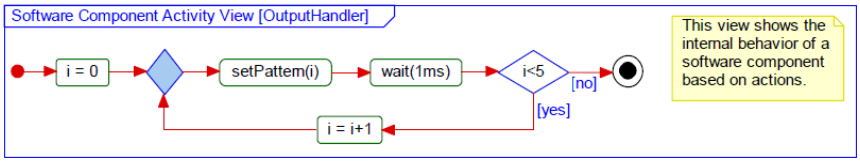


- Abstract view
  - Concurrency of activities
  - Synchronization points
  - "High level" algorithms

22

metio
methodik · information · organisation

Software Component Activity View

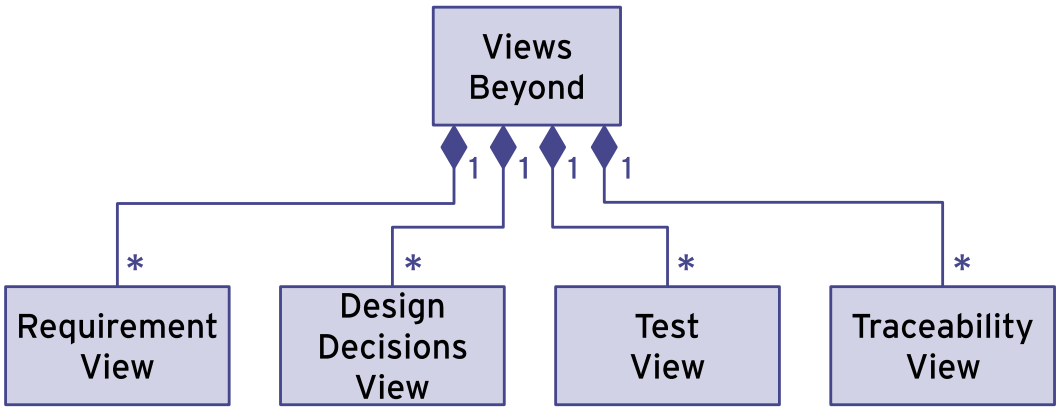


- Detailed view
  - Algorithms: Complete or partial description of the component behavior

23

metio
methodik · information · organisation

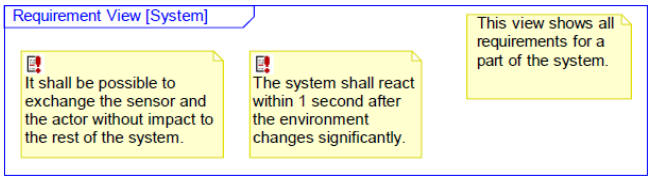
Views Beyond

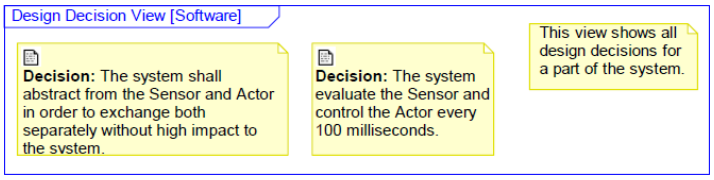


24

metio
methodik · information · organisation

Requirement and Design Decisions View





25

metio
methodik · information · organisation

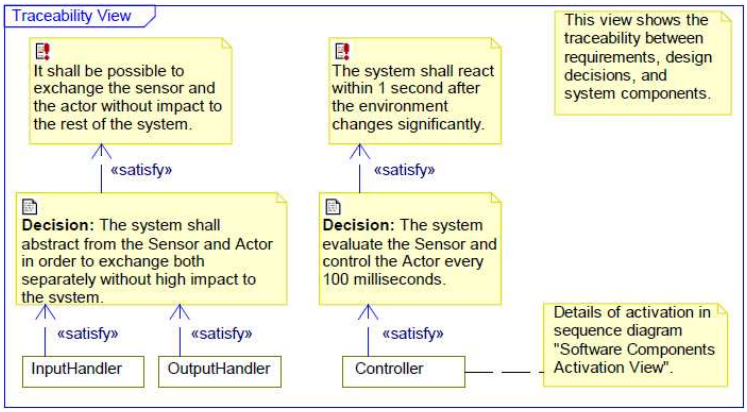
Test View

- Description of tests
  - Based on
    - requirements
    - design
  - Sequence diagrams
- Test traceability in Traceability View

26

metio
methodik · information · organisation

Traceability View



- Vertical traceability shown here
- Horizontal traceability can be shown as well

27

metio
methodik · information · organisation

What views shall be used?

- Essential: Structure views of the system
  - Use as few refinement views as necessary
- Additional views: Behavior and views beyond
  - Add behavior and views beyond where useful
- Reduction of the number of views?
  - What is missing if this view was not available?

28

metio
methodik · information · organisation

Instead of a Summary

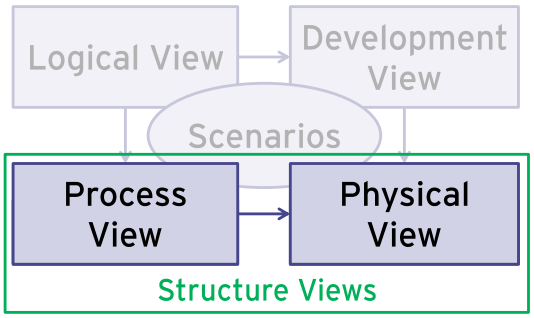
- Tailoring of the approach is necessary
  - depending on what the model shall deliver
- Not included here, but high impact (and very specific to the development context)
  - Classification of safety criticality
    - Block, FlowPort, Flow?
  - Handling of releases
  - Handling of variants

29

metio
methodik · information · organisation

Approach related to others

- P. Kruchten
  - Architectural Blueprints - "4+1" View Model of Software Architecture



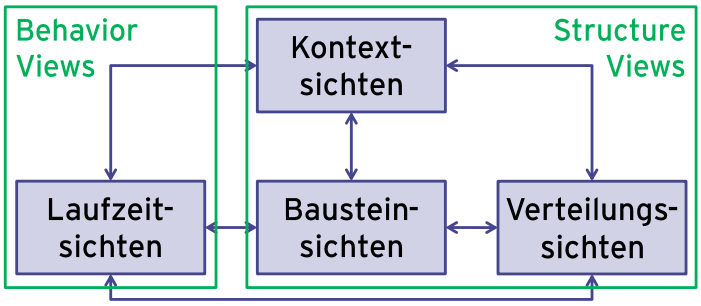
- Our approach: Focus on complete signal flow
  - Process View more detailed
  - Physical View included in Process View

30

metio
methodik · information · organisation

Approach related to others

- G. Starke
  - Effektive Software-Architekturen



- Our approach: Focus on complete signal flow
  - "Bausteinsichten" refine "Kontextsichten"
  - "Bausteinsichten" extended by "Datensicht"
  - "Verteilungssichten" in "Bausteinsichten"

31

metio
methodik · information · organisation

Modeling Tips

- Do not look for the one and only view
- Work with one model instead of several
- Do not try to model every small detail
- Often a note is more intuitive than a diagram
- Try to keep the number of views as small as possible
- Use a small number of language elements

32

metio
methodik · information · organisation

Modeling Tips

- Add a short description to every diagram
- Eliminate redundancy
- Connect the views and assure their consistency
- If diagrams get to complex split them under consideration of the model-view paradigm
- Add a legend to your diagrams or even create a central legend for the model

33

metio
methodik · information · organisation

metio
methodik · information · organisation

A. Programmkomitee

Prof. Dr. Urs Andelfinger (Software Engineering Institute Europe / HS Darmstadt)

Prof. Dr. Gerd Beneken (Fachhochschule Rosenheim)

Prof. Dr. Stefan Biffl (OCG/Technische Universität Wien)

Hubert Biskup (IBM Deutschland GmbH)

Prof. Dr. Ruth Breu (Universität Innsbruck)

Dr. Peter Haumer (IBM Software Group)

Dr. Karl Kollischan (kobaXX Consultants)

Dr. Marco Kuhrmann (Technische Universität München)

Peter Lang (Peter Lang Management- und IT-Consulting)

Dr. Christian Lange (Bundesstelle für Informationstechnik, BIT)

Dr. Oliver Linssen (Liantis GmbH & Co. KG)

Rudolf Neurath (IABG Industrieanlagen-Betriebsgesellschaft mbH)

Prof. Dr. Andreas Rausch (Technische Universität Clausthal

Martin Rother (PRINCE2 Deutschland e.V.)

Dr. Ina Schäfer (TU Braunschweig)

Dr. André Schnackenburg (Bundesstelle für Informationstechnik, BIT)

Dr. Marc Sihling (4Soft GmbH)

Hans-Jürgen Thönnißen-Fries (ESG Elektroniksystem- und Logistik GmbH)

Aachener Informatik-Berichte

This is the list of all technical reports since 1987. To obtain copies of reports please consult

http://aib.informatik.rwth-aachen.de/

or send your request to:

Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen,
Email: biblio@informatik.rwth-aachen.de

1987-01 * Fachgruppe Informatik: Jahresbericht 1986

1987-02 * David de Frutos Escrig, Klaus Indermark: Equivalence Relations of Non-Deterministic Ianov-Schemes

1987-03 * Manfred Nagl: A Software Development Environment based on Graph Technology

1987-04 * Claus Lewerentz, Manfred Nagl, Bernhard Westfechtel: On Integration Mechanisms within a Graph-Based Software Development Environment

1987-05 * Reinhard Rinn: Über Eingabeanomalien bei verschiedenen Inferenzmodellen

1987-06 * Werner Damm, Gert Döhmen: Specifying Distributed Computer Architectures in AADL*

1987-07 * Gregor Engels, Claus Lewerentz, Wilhelm Schäfer: Graph Grammar Engineering: A Software Specification Method

1987-08 * Manfred Nagl: Set Theoretic Approaches to Graph Grammars

1987-09 * Claus Lewerentz, Andreas Schürr: Experiences with a Database System for Software Documents

1987-10 * Herbert Klaeren, Klaus Indermark: A New Implementation Technique for Recursive Function Definitions

1987-11 * Rita Loogen: Design of a Parallel Programmable Graph Reduction Machine with Distributed Memory

1987-12 J. Börstler, U. Möncke, R. Wilhelm: Table compression for tree automata

1988-01 * Gabriele Esser, Johannes Rückert, Frank Wagner Gesellschaftliche Aspekte der Informatik

1988-02 * Peter Martini, Otto Spaniol: Token-Passing in High-Speed Backbone Networks for Campus-Wide Environments

1988-03 * Thomas Welzel: Simulation of a Multiple Token Ring Backbone

1988-04 * Peter Martini: Performance Comparison for HSLAN Media Access Protocols

1988-05 * Peter Martini: Performance Analysis of Multiple Token Rings

1988-06 * Andreas Mann, Johannes Rückert, Otto Spaniol: Datenfunknetze

1988-07 * Andreas Mann, Johannes Rückert: Packet Radio Networks for Data Exchange

1988-08 * Andreas Mann, Johannes Rückert: Concurrent Slot Assignment Protocol for Packet Radio Networks

1988-09 * W. Kremer, F. Reichert, J. Rückert, A. Mann: Entwurf einer Netzwerktopologie für ein Mobilfunknetz zur Unterstützung des öffentlichen Straßenverkehrs

1988-10 * Kai Jakobs: Towards User-Friendly Networking

1988-11 * Kai Jakobs: The Directory - Evolution of a Standard

1988-12 * Kai Jakobs: Directory Services in Distributed Systems - A Survey

1988-13 * Martine Schümmer: RS-511, a Protocol for the Plant Floor

1988-14 * U. Quernheim: Satellite Communication Protocols - A Performance Comparison Considering On-Board Processing

1988-15 * Peter Martini, Otto Spaniol, Thomas Welzel: File Transfer in High Speed Token Ring Networks: Performance Evaluation by Approximate Analysis and Simulation

1988-16 * Fachgruppe Informatik: Jahresbericht 1987

1988-17 * Wolfgang Thomas: Automata on Infinite Objects

1988-18 * Michael Sonnenschein: On Petri Nets and Data Flow Graphs

1988-19 * Heiko Vogler: Functional Distribution of the Contextual Analysis in Block-Structured Programming Languages: A Case Study of Tree Transducers

1988-20 * Thomas Welzel: Einsatz des Simulationswerkzeuges QNAP2 zur Leistungsbewertung von Kommunikationsprotokollen

1988-21 * Th. Janning, C. Lewerentz: Integrated Project Team Management in a Software Development Environment

1988-22 * Joost Engelfriet, Heiko Vogler: Modular Tree Transducers

1988-23 * Wolfgang Thomas: Automata and Quantifier Hierarchies

1988-24 * Uschi Heuter: Generalized Definite Tree Languages

1989-01 * Fachgruppe Informatik: Jahresbericht 1988

1989-02 * G. Esser, J. Rückert, F. Wagner (Hrsg.): Gesellschaftliche Aspekte der Informatik

1989-03 * Heiko Vogler: Bottom-Up Computation of Primitive Recursive Tree Functions

1989-04 * Andy Schürr: Introduction to PROGRESS, an Attribute Graph Grammar Based Specification Language

1989-05 J. Börstler: Reuse and Software Development - Problems, Solutions, and Bibliography (in German)

1989-06 * Kai Jakobs: OSI - An Appropriate Basis for Group Communication?

1989-07 * Kai Jakobs: ISO's Directory Proposal - Evolution, Current Status and Future Problems

1989-08 * Bernhard Westfechtel: Extension of a Graph Storage for Software Documents with Primitives for Undo/Redo and Revision Control

1989-09 * Peter Martini: High Speed Local Area Networks - A Tutorial

1989-10 * P. Davids, Th. Welzel: Performance Analysis of DQDB Based on Simulation

1989-11 * Manfred Nagl (Ed.): Abstracts of Talks presented at the WG '89 15th International Workshop on Graphtheoretic Concepts in Computer Science

1989-12 * Peter Martini: The DQDB Protocol - Is it Playing the Game?

1989-13 * Martine Schümmer: CNC/DNC Communication with MAP

1989-14 * Martine Schümmer: Local Area Networks for Manufactoring Environments with hard Real-Time Requirements

1989-15 * M. Schümmer, Th. Welzel, P. Martini: Integration of Field Bus and MAP Networks - Hierarchical Communication Systems in Production Environments

1989-16 * G. Vossen, K.-U. Witt: SUXESS: Towards a Sound Unification of Extensions of the Relational Data Model

1989-17 * J. Derissen, P. Hruschka, M.v.d. Beeck, Th. Janning, M. Nagl: Integrating Structured Analysis and Information Modelling

1989-18 A. Maassen: Programming with Higher Order Functions

1989-19 * Mario Rodriguez-Artalejo, Heiko Vogler: A Narrowing Machine for Syntax Directed BABEL

1989-20 H. Kuchen, R. Loogen, J.J. Moreno Navarro, M. Rodriguez Artalejo: Graph-based Implementation of a Functional Logic Language

1990-01 * Fachgruppe Informatik: Jahresbericht 1989

1990-02 * Vera Jansen, Andreas Potthoff, Wolfgang Thomas, Udo Wermuth: A Short Guide to the AMORE System (Computing Automata, MOnoids and Regular Expressions)

1990-03 * Jerzy Skurczynski: On Three Hierarchies of Weak SkS Formulas

1990-04 R. Loogen: Stack-based Implementation of Narrowing

1990-05 H. Kuchen, A. Wagener: Comparison of Dynamic Load Balancing Strategies

1990-06 * Kai Jakobs, Frank Reichert: Directory Services for Mobile Communication

1990-07 * Kai Jakobs: What's Beyond the Interface - OSI Networks to Support Cooperative Work

1990-08 * Kai Jakobs: Directory Names and Schema - An Evaluation

1990-09 * Ulrich Quernheim, Dieter Kreuer: Das CCITT - Signalisierungssystem Nr. 7 auf Satellitenstrecken; Simulation der Zeichengabestrecke

1990-11 H. Kuchen, R. Loogen, J.J. Moreno Navarro, M. Rodriguez Artalejo: Lazy Narrowing in a Graph Machine

1990-12 * Kai Jakobs, Josef Kaltwasser, Frank Reichert, Otto Spaniol: Der Computer fährt mit

1990-13 * Rudolf Mathar, Andreas Mann: Analyzing a Distributed Slot Assignment Protocol by Markov Chains

1990-14 A. Maassen: Compilerentwicklung in Miranda - ein Praktikum in funktionaler Programmierung (written in german)

1990-15 * Manfred Nagl, Andreas Schürr: A Specification Environment for Graph Grammars

1990-16 A. Schürr: PROGRESS: A VHL-Language Based on Graph Grammars

1990-17 * Marita Möller: Ein Ebenenmodell wissensbasierter Konsultationen - Unterstützung für Wissensakquisition und Erklärungsfähigkeit

1990-18 * Eric Kowalewski: Entwurf und Interpretation einer Sprache zur Beschreibung von Konsultationsphasen in Expertensystemen

1990-20 Y. Ortega Mallen, D. de Frutos Escrig: A Complete Proof System for Timed Observations

1990-21 * Manfred Nagl: Modelling of Software Architectures: Importance, Notions, Experiences

1990-22 H. Fassbender, H. Vogler: A Call-by-need Implementation of Syntax Directed Functional Programming

1991-01 Guenther Geiler (ed.), Fachgruppe Informatik: Jahresbericht 1990

1991-03 B. Steffen, A. Ingolfsdottir: Characteristic Formulae for Processes with Divergence

1991-04 M. Portz: A new class of cryptosystems based on interconnection networks

1991-05 H. Kuchen, G. Geiler: Distributed Applicative Arrays

1991-06 * Ludwig Staiger: Kolmogorov Complexity and Hausdorff Dimension

1991-07 * Ludwig Staiger: Syntactic Congruences for w-languages

1991-09 * Eila Kuikka: A Proposal for a Syntax-Directed Text Processing System

1991-10 K. Gladitz, H. Fassbender, H. Vogler: Compiler-based Implementation of Syntax-Directed Functional Programming

1991-11 R. Loogen, St. Winkler: Dynamic Detection of Determinism in Functional Logic Languages

1991-12 * K. Indermark, M. Rodriguez Artalejo (Eds.): Granada Workshop on the Integration of Functional and Logic Programming

1991-13 * Rolf Hager, Wolfgang Kremer: The Adaptive Priority Scheduler: A More Fair Priority Service Discipline

1991-14 * Andreas Fasbender, Wolfgang Kremer: A New Approximation Algorithm for Tandem Networks with Priority Nodes

1991-15 J. Börstler, A. Zündorf: Revisiting extensions to Modula-2 to support reusability

1991-16 J. Börstler, Th. Janning: Bridging the gap between Requirements Analysis and Design

1991-17 A. Zündorf, A. Schürr: Nondeterministic Control Structures for Graph Rewriting Systems

1991-18 * Matthias Jarke, John Mylopoulos, Joachim W. Schmidt, Yannis Vassiliou: DAIDA: An Environment for Evolving Information Systems

1991-19 M. Jeusfeld, M. Jarke: From Relational to Object-Oriented Integrity Simplification

1991-20 G. Hogen, A. Kindler, R. Loogen: Automatic Parallelization of Lazy Functional Programs

1991-21 * Prof. Dr. rer. nat. Otto Spaniol: ODP (Open Distributed Processing): Yet another Viewpoint

1991-22 H. Kuchen, F. Lücking, H. Stoltze: The Topology Description Language TDL

1991-23 S. Graf, B. Steffen: Compositional Minimization of Finite State Systems

1991-24 R. Cleaveland, J. Parrow, B. Steffen: The Concurrency Workbench: A Semantics Based Tool for the Verification of Concurrent Systems

1991-25 * Rudolf Mathar, Jürgen Mattfeldt: Optimal Transmission Ranges for Mobile Communication in Linear Multihop Packet Radio Networks

1991-26 M. Jeusfeld, M. Staudt: Query Optimization in Deductive Object Bases

1991-27 J. Knoop, B. Steffen: The Interprocedural Coincidence Theorem

1991-28 J. Knoop, B. Steffen: Unifying Strength Reduction and Semantic Code Motion

1991-30 T. Margaria: First-Order theories for the verification of complex FSMs

1991-31 B. Steffen: Generating Data Flow Analysis Algorithms from Modal Specifications

1992-01 Stefan Eherer (ed.), Fachgruppe Informatik: Jahresbericht 1991

1992-02 * Bernhard Westfechtel: Basismechanismen zur Datenverwaltung in strukturbezogenen Hypertextsystemen

1992-04 S. A. Smolka, B. Steffen: Priority as Extremal Probability

1992-05 * Matthias Jarke, Carlos Maltzahn, Thomas Rose: Sharing Processes: Team Coordination in Design Repositories

1992-06 O. Burkart, B. Steffen: Model Checking for Context-Free Processes

1992-07 * Matthias Jarke, Klaus Pohl: Information Systems Quality and Quality Information Systems

1992-08 * Rudolf Mathar, Jürgen Mattfeldt: Analyzing Routing Strategy NFP in Multihop Packet Radio Networks on a Line

1992-09 * Alfons Kemper, Guido Moerkotte: Grundlagen objektorientierter Datenbanksysteme

1992-10 Matthias Jarke, Manfred Jeusfeld, Andreas Miethsam, Michael Gocek: Towards a logic-based reconstruction of software configuration management

1992-11 Werner Hans: A Complete Indexing Scheme for WAM-based Abstract Machines

1992-12 W. Hans, R. Loogen, St. Winkler: On the Interaction of Lazy Evaluation and Backtracking

1992-13 * Matthias Jarke, Thomas Rose: Specification Management with CAD

1992-14 Th. Noll, H. Vogler: Top-down Parsing with Simultaneous Evaluation on Noncircular Attribute Grammars

1992-15 A. Schuerr, B. Westfechtel: Graphgrammatiken und Graphersetzungssysteme(written in german)

1992-16 * Graduiertenkolleg Informatik und Technik (Hrsg.): Forschungsprojekte des Graduiertenkollegs Informatik und Technik

1992-17 M. Jarke (ed.): ConceptBase V3.1 User Manual

1992-18 * Clarence A. Ellis, Matthias Jarke (Eds.): Distributed Cooperation in Integrated Information Systems - Proceedings of the Third International Workshop on Intelligent and Cooperative Information Systems

1992-19-00 H. Kuchen, R. Loogen (eds.): Proceedings of the 4th Int. Workshop on the Parallel Implementation of Functional Languages

1992-19-01 G. Hogen, R. Loogen: PASTEL - A Parallel Stack-Based Implementation of Eager Functional Programs with Lazy Data Structures (Extended Abstract)

1992-19-02 H. Kuchen, K. Gladitz: Implementing Bags on a Shared Memory MIMD-Machine

1992-19-03 C. Rathsack, S.B. Scholz: LISA - A Lazy Interpreter for a Full-Fledged Lambda-Calculus

1992-19-04 T.A. Bratvold: Determining Useful Parallelism in Higher Order Functions

1992-19-05 S. Kahrs: Polymorphic Type Checking by Interpretation of Code

1992-19-06 M. Chakravarty, M. Köhler: Equational Constraints, Residuation, and the Parallel JUMP-Machine

1992-19-07 J. Seward: Polymorphic Strictness Analysis using Frontiers (Draft Version)

1992-19-08 D. Gärtner, A. Kimms, W. Kluge: pi-Redˆ+ - A Compiling Graph-Reduction System for a Full Fledged Lambda-Calculus

1992-19-09 D. Howe, G. Burn: Experiments with strict STG code

1992-19-10 J. Glauert: Parallel Implementation of Functional Languages Using Small Processes

1992-19-11 M. Joy, T. Axford: A Parallel Graph Reduction Machine

1992-19-12 A. Bennett, P. Kelly: Simulation of Multicache Parallel Reduction

1992-19-13 K. Langendoen, D.J. Agterkamp: Cache Behaviour of Lazy Functional Programs (Working Paper)

1992-19-14 K. Hammond, S. Peyton Jones: Profiling scheduling strategies on the GRIP parallel reducer

1992-19-15 S. Mintchev: Using Strictness Information in the STG-machine

1992-19-16 D. Rushall: An Attribute Grammar Evaluator in Haskell

1992-19-17 J. Wild, H. Glaser, P. Hartel: Statistics on storage management in a lazy functional language implementation

1992-19-18 W.S. Martins: Parallel Implementations of Functional Languages

1992-19-19 D. Lester: Distributed Garbage Collection of Cyclic Structures (Draft version)

1992-19-20 J.C. Glas, R.F.H. Hofman, W.G. Vree: Parallelization of Branch-and-Bound Algorithms in a Functional Programming Environment

1992-19-21 S. Hwang, D. Rushall: The nu-STG machine: a parallelized Spineless Tagless Graph Reduction Machine in a distributed memory architecture (Draft version)

1992-19-22 G. Burn, D. Le Metayer: Cps-Translation and the Correctness of Optimising Compilers

1992-19-23 S.L. Peyton Jones, P. Wadler: Imperative functional programming (Brief summary)

1992-19-24 W. Damm, F. Liu, Th. Peikenkamp: Evaluation and Parallelization of Functions in Functional + Logic Languages (abstract)

1992-19-25 M. Kesseler: Communication Issues Regarding Parallel Functional Graph Rewriting

1992-19-26 Th. Peikenkamp: Charakterizing and representing neededness in functional loginc languages (abstract)

1992-19-27 H. Doerr: Monitoring with Graph-Grammars as formal operational Models

1992-19-28 J. van Groningen: Some implementation aspects of Concurrent Clean on distributed memory architectures

1992-19-29 G. Ostheimer: Load Bounding for Implicit Parallelism (abstract)

1992-20 H. Kuchen, F.J. Lopez Fraguas, J.J. Moreno Navarro, M. Rodriguez Artalejo: Implementing Disequality in a Lazy Functional Logic Language

1992-21 H. Kuchen, F.J. Lopez Fraguas: Result Directed Computing in a Functional Logic Language

1992-22 H. Kuchen, J.J. Moreno Navarro, M.V. Hermenegildo: Independent AND-Parallel Narrowing

1992-23 T. Margaria, B. Steffen: Distinguishing Formulas for Free

1992-24 K. Pohl: The Three Dimensions of Requirements Engineering

1992-25 * R. Stainov: A Dynamic Configuration Facility for Multimedia Communications

1992-26 * Michael von der Beeck: Integration of Structured Analysis and Timed Statecharts for Real-Time and Concurrency Specification

1992-27 W. Hans, St. Winkler: Aliasing and Groundness Analysis of Logic Programs through Abstract Interpretation and its Safety

1992-28 * Gerhard Steinke, Matthias Jarke: Support for Security Modeling in Information Systems Design

1992-29 B. Schinzel: Warum Frauenforschung in Naturwissenschaft und Technik

1992-30 A. Kemper, G. Moerkotte, K. Peithner: Object-Orientation Axiomatised by Dynamic Logic

1992-32 * Bernd Heinrichs, Kai Jakobs: Timer Handling in High-Performance Transport Systems

1992-33 * B. Heinrichs, K. Jakobs, K. Lenßen, W. Reinhardt, A. Spinner: EuroBridge: Communication Services for Multimedia Applications

1992-34 C. Gerlhof, A. Kemper, Ch. Kilger, G. Moerkotte: Partition-Based Clustering in Object Bases: From Theory to Practice

1992-35 J. Börstler: Feature-Oriented Classification and Reuse in IPSEN

1992-36 M. Jarke, J. Bubenko, C. Rolland, A. Sutcliffe, Y. Vassiliou: Theories Underlying Requirements Engineering: An Overview of NATURE at Genesis

1992-37 * K. Pohl, M. Jarke: Quality Information Systems: Repository Support for Evolving Process Models

1992-38 A. Zuendorf: Implementation of the imperative / rule based language PROGRES

1992-39 P. Koch: Intelligentes Backtracking bei der Auswertung funktional-logischer Programme

1992-40 * Rudolf Mathar, Jürgen Mattfeldt: Channel Assignment in Cellular Radio Networks

1992-41 * Gerhard Friedrich, Wolfgang Neidl: Constructive Utility in Model-Based Diagnosis Repair Systems

1992-42 * P. S. Chen, R. Hennicker, M. Jarke: On the Retrieval of Reusable Software Components

1992-43 W. Hans, St.Winkler: Abstract Interpretation of Functional Logic Languages

1992-44 N. Kiesel, A. Schuerr, B. Westfechtel: Design and Evaluation of GRAS, a Graph-Oriented Database System for Engineering Applications

1993-01 * Fachgruppe Informatik: Jahresbericht 1992

1993-02 * Patrick Shicheng Chen: On Inference Rules of Logic-Based Information Retrieval Systems

1993-03 G. Hogen, R. Loogen: A New Stack Technique for the Management of Runtime Structures in Distributed Environments

1993-05 A. Zündorf: A Heuristic for the Subgraph Isomorphism Problem in Executing PROGRES

1993-06 A. Kemper, D. Kossmann: Adaptable Pointer Swizzling Strategies in Object Bases: Design, Realization, and Quantitative Analysis

1993-07 * Graduiertenkolleg Informatik und Technik (Hrsg.): Graduiertenkolleg Informatik und Technik

1993-08 * Matthias Berger: k-Coloring Vertices using a Neural Network with Convergence to Valid Solutions

1993-09 M. Buchheit, M. Jeusfeld, W. Nutt, M. Staudt: Subsumption between Queries to Object-Oriented Databases

1993-10 O. Burkart, B. Steffen: Pushdown Processes: Parallel Composition and Model Checking

1993-11 * R. Große-Wienker, O. Hermanns, D. Menzenbach, A. Pollacks, S. Repetzki, J. Schwartz, K. Sonnenschein, B. Westfechtel: Das SUKITS-Projekt: A-posteriori-Integration heterogener CIM-Anwendungssysteme

1993-12 * Rudolf Mathar, Jürgen Mattfeldt: On the Distribution of Cumulated Interference Power in Rayleigh Fading Channels

1993-13 O. Maler, L. Staiger: On Syntactic Congruences for omega-languages

1993-14 M. Jarke, St. Eherer, R. Gallersdoerfer, M. Jeusfeld, M. Staudt: ConceptBase - A Deductive Object Base Manager

1993-15 M. Staudt, H.W. Nissen, M.A. Jeusfeld: Query by Class, Rule and Concept

1993-16 * M. Jarke, K. Pohl, St. Jacobs et al.: Requirements Engineering: An Integrated View of Representation Process and Domain

1993-17 * M. Jarke, K. Pohl: Establishing Vision in Context: Towards a Model of Requirements Processes

1993-18 W. Hans, H. Kuchen, St. Winkler: Full Indexing for Lazy Narrowing

1993-19 W. Hans, J.J. Ruz, F. Saenz, St. Winkler: A VHDL Specification of a Shared Memory Parallel Machine for Babel

1993-20 * K. Finke, M. Jarke, P. Szczurko, R. Soltysiak: Quality Management for Expert Systems in Process Control

1993-21 M. Jarke, M.A. Jeusfeld, P. Szczurko: Three Aspects of Intelligent Cooperation in the Quality Cycle

1994-01 Margit Generet, Sven Martin (eds.), Fachgruppe Informatik: Jahresbericht 1993

1994-02 M. Lefering: Development of Incremental Integration Tools Using Formal Specifications

1994-03 * P. Constantopoulos, M. Jarke, J. Mylopoulos, Y. Vassiliou: The Software Information Base: A Server for Reuse

1994-04 * Rolf Hager, Rudolf Mathar, Jürgen Mattfeldt: Intelligent Cruise Control and Reliable Communication of Mobile Stations

1994-05 * Rolf Hager, Peter Hermesmann, Michael Portz: Feasibility of Authentication Procedures within Advanced Transport Telematics

1994-06 * Claudia Popien, Bernd Meyer, Axel Kuepper: A Formal Approach to Service Import in ODP Trader Federations

1994-07 P. Peters, P. Szczurko: Integrating Models of Quality Management Methods by an Object-Oriented Repository

1994-08 * Manfred Nagl, Bernhard Westfechtel: A Universal Component for the Administration in Distributed and Integrated Development Environments

1994-09 * Patrick Horster, Holger Petersen: Signatur- und Authentifikationsverfahren auf der Basis des diskreten Logarithmusproblems

1994-11 A. Schürr: PROGRES, A Visual Language and Environment for PROgramming with Graph REwrite Systems

1994-12 A. Schürr: Specification of Graph Translators with Triple Graph Grammars

1994-13 A. Schürr: Logic Based Programmed Structure Rewriting Systems

1994-14 L. Staiger: Codes, Simplifying Words, and Open Set Condition

1994-15 * Bernhard Westfechtel: A Graph-Based System for Managing Configurations of Engineering Design Documents

1994-16 P. Klein: Designing Software with Modula-3

1994-17 I. Litovsky, L. Staiger: Finite acceptance of infinite words

1994-18 G. Hogen, R. Loogen: Parallel Functional Implementations: Graphbased vs. Stackbased Reduction

1994-19 M. Jeusfeld, U. Johnen: An Executable Meta Model for Re-Engineering of Database Schemas

1994-20 * R. Gallersdörfer, M. Jarke, K. Klabunde: Intelligent Networks as a Data Intensive Application (INDIA)

1994-21 M. Mohnen: Proving the Correctness of the Static Link Technique Using Evolving Algebras

1994-22 H. Fernau, L. Staiger: Valuations and Unambiguity of Languages, with Applications to Fractal Geometry

1994-24 * M. Jarke, K. Pohl, R. Dömges, St. Jacobs, H. W. Nissen: Requirements Information Management: The NATURE Approach

1994-25 * M. Jarke, K. Pohl, C. Rolland, J.-R. Schmitt: Experience-Based Method Evaluation and Improvement: A Process Modeling Approach

1994-26 * St. Jacobs, St. Kethers: Improving Communication and Decision Making within Quality Function Deployment

1994-27 * M. Jarke, H. W. Nissen, K. Pohl: Tool Integration in Evolving Information Systems Environments

1994-28 O. Burkart, D. Caucal, B. Steffen: An Elementary Bisimulation Decision Procedure for Arbitrary Context-Free Processes

1995-01 * Fachgruppe Informatik: Jahresbericht 1994

1995-02 Andy Schürr, Andreas J. Winter, Albert Zündorf: Graph Grammar Engineering with PROGRES

1995-03 Ludwig Staiger: A Tight Upper Bound on Kolmogorov Complexity by Hausdorff Dimension and Uniformly Optimal Prediction

1995-04 Birgitta König-Ries, Sven Helmer, Guido Moerkotte: An experimental study on the complexity of left-deep join ordering problems for cyclic queries

1995-05 Sophie Cluet, Guido Moerkotte: Efficient Evaluation of Aggregates on Bulk Types

1995-06 Sophie Cluet, Guido Moerkotte: Nested Queries in Object Bases

1995-07 Sophie Cluet, Guido Moerkotte: Query Optimization Techniques Exploiting Class Hierarchies

1995-08 Markus Mohnen: Efficient Compile-Time Garbage Collection for Arbitrary Data Structures

1995-09 Markus Mohnen: Functional Specification of Imperative Programs: An Alternative Point of View of Functional Languages

1995-10 Rainer Gallersdörfer, Matthias Nicola: Improving Performance in Replicated Databases through Relaxed Coherency

1995-11 * M.Staudt, K.von Thadden: Subsumption Checking in Knowledge Bases

1995-12 * G.V.Zemanek, H.W.Nissen, H.Hubert, M.Jarke: Requirements Analysis from Multiple Perspectives: Experiences with Conceptual Modeling Technology

1995-13 * M.Staudt, M.Jarke: Incremental Maintenance of Externally Materialized Views

1995-14 * P.Peters, P.Szczurko, M.Jeusfeld: Oriented Information Management: Conceptual Models at Work

1995-15 * Matthias Jarke, Sudha Ram (Hrsg.): WITS 95 Proceedings of the 5th Annual Workshop on Information Technologies and Systems

1995-16 * W.Hans, St.Winkler, F.Saenz: Distributed Execution in Functional Logic Programming

1996-01 * Jahresbericht 1995

1996-02 Michael Hanus, Christian Prehofer: Higher-Order Narrowing with Definitional Trees

1996-03 * W.Scheufele, G.Moerkotte: Optimal Ordering of Selections and Joins in Acyclic Queries with Expensive Predicates

1996-04 Klaus Pohl: PRO-ART: Enabling Requirements Pre-Traceability

1996-05 Klaus Pohl: Requirements Engineering: An Overview

1996-06 * M.Jarke, W.Marquardt: Design and Evaluation of Computer–Aided Process Modelling Tools

1996-07 Olaf Chitil: The Sigma-Semantics: A Comprehensive Semantics for Functional Programs

1996-08 * S.Sripada: On Entropy and the Limitations of the Second Law of Thermodynamics

1996-09 Michael Hanus (Ed.): Proceedings of the Poster Session of ALP96 - Fifth International Conference on Algebraic and Logic Programming

1996-09-0 Michael Hanus (Ed.): Proceedings of the Poster Session of ALP 96 - Fifth International Conference on Algebraic and Logic Programming: Introduction and table of contents

1996-09-1 Ilies Alouini: An Implementation of Conditional Concurrent Rewriting on Distributed Memory Machines

1996-09-2 Olivier Danvy, Karoline Malmkjær: On the Idempotence of the CPS Transformation

1996-09-3 Víctor M. Gulías, José L. Freire: Concurrent Programming in Haskell

1996-09-4 Sébastien Limet, Pierre Réty: On Decidability of Unifiability Modulo Rewrite Systems

1996-09-5 Alexandre Tessier: Declarative Debugging in Constraint Logic Programming

1996-10 Reidar Conradi, Bernhard Westfechtel: Version Models for Software Configuration Management

1996-11 * C.Weise, D.Lenzkes: A Fast Decision Algorithm for Timed Refinement

1996-12 * R.Dömges, K.Pohl, M.Jarke, B.Lohmann, W.Marquardt: PRO-ART/CE* — An Environment for Managing the Evolution of Chemical Process Simulation Models

1996-13 * K.Pohl, R.Klamma, K.Weidenhaupt, R.Dömges, P.Haumer, M.Jarke: A Framework for Process-Integrated Tools

1996-14 * R.Gallersdörfer, K.Klabunde, A.Stolz, M.Eßmajor: INDIA — Intelligent Networks as a Data Intensive Application, Final Project Report, June 1996

1996-15 * H.Schimpe, M.Staudt: VAREX: An Environment for Validating and Refining Rule Bases

1996-16 * M.Jarke, M.Gebhardt, S.Jacobs, H.Nissen: Conflict Analysis Across Heterogeneous Viewpoints: Formalization and Visualization

1996-17 Manfred A. Jeusfeld, Tung X. Bui: Decision Support Components on the Internet

1996-18 Manfred A. Jeusfeld, Mike Papazoglou: Information Brokering: Design, Search and Transformation

1996-19 * P.Peters, M.Jarke: Simulating the impact of information flows in networked organizations

1996-20 Matthias Jarke, Peter Peters, Manfred A. Jeusfeld: Model-driven planning and design of cooperative information systems

1996-21 * G.de Michelis, E.Dubois, M.Jarke, F.Matthes, J.Mylopoulos, K.Pohl, J.Schmidt, C.Woo, E.Yu: Cooperative information systems: a manifesto

1996-22 * S.Jacobs, M.Gebhardt, S.Kethers, W.Rzasa: Filling HTML forms simultaneously: CoWeb architecture and functionality

1996-23 * M.Gebhardt, S.Jacobs: Conflict Management in Design

1997-01 Michael Hanus, Frank Zartmann (eds.): Jahresbericht 1996

1997-02 Johannes Faassen: Using full parallel Boltzmann Machines for Optimization

1997-03 Andreas Winter, Andy Schürr: Modules and Updatable Graph Views for PROgrammed Graph REwriting Systems

1997-04 Markus Mohnen, Stefan Tobies: Implementing Context Patterns in the Glasgow Haskell Compiler

1997-05 * S.Gruner: Schemakorrespondenzaxiome unterstützen die paargrammatische Spezifikation inkrementeller Integrationswerkzeuge

1997-06 Matthias Nicola, Matthias Jarke: Design and Evaluation of Wireless Health Care Information Systems in Developing Countries

1997-07 Petra Hofstedt: Taskparallele Skelette für irregulär strukturierte Probleme in deklarativen Sprachen

1997-08 Dorothea Blostein, Andy Schürr: Computing with Graphs and Graph Rewriting

1997-09 Carl-Arndt Krapp, Bernhard Westfechtel: Feedback Handling in Dynamic Task Nets

1997-10 Matthias Nicola, Matthias Jarke: Integrating Replication and Communication in Performance Models of Distributed Databases

1997-11 * R. Klamma, P. Peters, M. Jarke: Workflow Support for Failure Management in Federated Organizations

1997-13 Markus Mohnen: Optimising the Memory Management of Higher-Order Functional Programs

1997-14 Roland Baumann: Client/Server Distribution in a Structure-Oriented Database Management System

1997-15 George Botorog: High-Level Parallel Programming and the Efficient Implementation of Numerical Algorithms

1998-01 * Fachgruppe Informatik: Jahresbericht 1997

1998-02 Stefan Gruner, Manfred Nagel, Andy Schürr: Fine-grained and Structure-Oriented Document Integration Tools are Needed for Development Processes

1998-03 Stefan Gruner: Einige Anmerkungen zur graphgrammatischen Spezifikation von Integrationswerkzeugen nach Westfechtel, Janning, Lefering und Schürr

1998-04 * O. Kubitz: Mobile Robots in Dynamic Environments

1998-05 Martin Leucker, Stephan Tobies: Truth - A Verification Platform for Distributed Systems

1998-06 * Matthias Oliver Berger: DECT in the Factory of the Future

1998-07 M. Arnold, M. Erdmann, M. Glinz, P. Haumer, R. Knoll, B. Paech, K. Pohl, J. Ryser, R. Studer, K. Weidenhaupt: Survey on the Scenario Use in Twelve Selected Industrial Projects

1998-09 * Th. Lehmann: Geometrische Ausrichtung medizinischer Bilder am Beispiel intraoraler Radiographien

1998-10 * M. Nicola, M. Jarke: Performance Modeling of Distributed and Replicated Databases

1998-11 * Ansgar Schleicher, Bernhard Westfechtel, Dirk Jäger: Modeling Dynamic Software Processes in UML

1998-12 * W. Appelt, M. Jarke: Interoperable Tools for Cooperation Support using the World Wide Web

1998-13 Klaus Indermark: Semantik rekursiver Funktionsdefinitionen mit Striktheitsinformation

1999-01 * Jahresbericht 1998

1999-02 * F. Huch: Verifcation of Erlang Programs using Abstract Interpretation and Model Checking — Extended Version

1999-03 * R. Gallersdörfer, M. Jarke, M. Nicola: The ADR Replication Manager

1999-04 María Alpuente, Michael Hanus, Salvador Lucas, Germán Vidal: Specialization of Functional Logic Programs Based on Needed Narrowing

1999-05 * W. Thomas (Ed.): DLT 99 - Developments in Language Theory Fourth International Conference

1999-06 * Kai Jakobs, Klaus-Dieter Kleefeld: Informationssysteme für die angewandte historische Geographie

1999-07 Thomas Wilke: CTL+ is exponentially more succinct than CTL

1999-08 Oliver Matz: Dot-Depth and Monadic Quantifier Alternation over Pictures

2000-01 * Jahresbericht 1999

2000-02 Jens Vöge, Marcin Jurdzinski A Discrete Strategy Improvement Algorithm for Solving Parity Games

2000-03 D. Jäger, A. Schleicher, B. Westfechtel: UPGRADE: A Framework for Building Graph-Based Software Engineering Tools

2000-04 Andreas Becks, Stefan Sklorz, Matthias Jarke: Exploring the Semantic Structure of Technical Document Collections: A Cooperative Systems Approach

2000-05 Mareike Schoop: Cooperative Document Management

2000-06 Mareike Schoop, Christoph Quix (eds.): Proceedings of the Fifth International Workshop on the Language-Action Perspective on Communication Modelling

2000-07 * Markus Mohnen, Pieter Koopman (Eds.): Proceedings of the 12th International Workshop of Functional Languages

2000-08 Thomas Arts, Thomas Noll: Verifying Generic Erlang Client-Server Implementations

2001-01 * Jahresbericht 2000

2001-02 Benedikt Bollig, Martin Leucker: Deciding LTL over Mazurkiewicz Traces

2001-03 Thierry Cachat: The power of one-letter rational languages

2001-04 Benedikt Bollig, Martin Leucker, Michael Weber: Local Parallel Model Checking for the Alternation Free mu-Calculus

2001-05 Benedikt Bollig, Martin Leucker, Thomas Noll: Regular MSC Languages

2001-06 Achim Blumensath: Prefix-Recognisable Graphs and Monadic Second-Order Logic

2001-07 Martin Grohe, Stefan Wöhrle: An Existential Locality Theorem

2001-08 Mareike Schoop, James Taylor (eds.): Proceedings of the Sixth International Workshop on the Language-Action Perspective on Communication Modelling

2001-09 Thomas Arts, Jürgen Giesl: A collection of examples for termination of term rewriting using dependency pairs

2001-10 Achim Blumensath: Axiomatising Tree-interpretable Structures

2001-11 Klaus Indermark, Thomas Noll (eds.): Kolloquium Programmiersprachen und Grundlagen der Programmierung

2002-01 * Jahresbericht 2001

2002-02 Jürgen Giesl, Aart Middeldorp: Transformation Techniques for Context-Sensitive Rewrite Systems

2002-03 Benedikt Bollig, Martin Leucker, Thomas Noll: Generalised Regular MSC Languages

2002-04 Jürgen Giesl, Aart Middeldorp: Innermost Termination of Context-Sensitive Rewriting

2002-05 Horst Lichter, Thomas von der Maßen, Thomas Weiler: Modelling Requirements and Architectures for Software Product Lines

2002-06 Henry N. Adorna: 3-Party Message Complexity is Better than 2-Party Ones for Proving Lower Bounds on the Size of Minimal Nondeterministic Finite Automata

2002-07 Jörg Dahmen: Invariant Image Object Recognition using Gaussian Mixture Densities

2002-08 Markus Mohnen: An Open Framework for Data-Flow Analysis in Java

2002-09 Markus Mohnen: Interfaces with Default Implementations in Java

2002-10 Martin Leucker: Logics for Mazurkiewicz traces

2002-11 Jürgen Giesl, Hans Zantema: Liveness in Rewriting

2003-01 * Jahresbericht 2002

2003-02 Jürgen Giesl, René Thiemann: Size-Change Termination for Term Rewriting

2003-03 Jürgen Giesl, Deepak Kapur: Deciding Inductive Validity of Equations

2003-04 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, Stephan Falke: Improving Dependency Pairs

2003-05 Christof Löding, Philipp Rohde: Solving the Sabotage Game is PSPACE-hard

2003-06 Franz Josef Och: Statistical Machine Translation: From Single-Word Models to Alignment Templates

2003-07 Horst Lichter, Thomas von der Maßen, Alexander Nyßen, Thomas Weiler: Vergleich von Ansätzen zur Feature Modellierung bei der Softwareproduktlinienentwicklung

2003-08 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, Stephan Falke: Mechanizing Dependency Pairs

2004-01 * Fachgruppe Informatik: Jahresbericht 2003

2004-02 Benedikt Bollig, Martin Leucker: Message-Passing Automata are expressively equivalent to EMSO logic

2004-03 Delia Kesner, Femke van Raamsdonk, Joe Wells (eds.): HOR 2004 – 2nd International Workshop on Higher-Order Rewriting

2004-04 Slim Abdennadher, Christophe Ringeissen (eds.): RULE 04 – Fifth International Workshop on Rule-Based Programming

2004-05 Herbert Kuchen (ed.): WFLP 04 – 13th International Workshop on Functional and (Constraint) Logic Programming

2004-06 Sergio Antoy, Yoshihito Toyama (eds.): WRS 04 – 4th International Workshop on Reduction Strategies in Rewriting and Programming

2004-07 Michael Codish, Aart Middeldorp (eds.): WST 04 – 7th International Workshop on Termination

2004-08 Klaus Indermark, Thomas Noll: Algebraic Correctness Proofs for Compiling Recursive Function Definitions with Strictness Information

2004-09 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Parameterized Power Domination Complexity

2004-10 Zinaida Benenson, Felix C. Gärtner, Dogan Kesdogan: Secure Multi-Party Computation with Security Modules

2005-01 * Fachgruppe Informatik: Jahresbericht 2004

2005-02 Maximillian Dornseif, Felix C. Gärtner, Thorsten Holz, Martin Mink: An Offensive Approach to Teaching Information Security: "Aachen Summer School Applied IT Security"

2005-03 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp: Proving and Disproving Termination of Higher-Order Functions

2005-04 Daniel Mölle, Stefan Richter, Peter Rossmanith: A Faster Algorithm for the Steiner Tree Problem

2005-05 Fabien Pouget, Thorsten Holz: A Pointillist Approach for Comparing Honeypots

2005-06 Simon Fischer, Berthold Vöcking: Adaptive Routing with Stale Information

2005-07 Felix C. Freiling, Thorsten Holz, Georg Wicherski: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks

2005-08 Joachim Kneis, Peter Rossmanith: A New Satisfiability Algorithm With Applications To Max-Cut

2005-09 Klaus Kursawe, Felix C. Freiling: Byzantine Fault Tolerance on General Hybrid Adversary Structures

2005-10 Benedikt Bollig: Automata and Logics for Message Sequence Charts

2005-11 Simon Fischer, Berthold Vöcking: A Counterexample to the Fully Mixed Nash Equilibrium Conjecture

2005-12 Neeraj Mittal, Felix Freiling, S. Venkatesan, Lucia Draque Penso: Efficient Reductions for Wait-Free Termination Detection in Faulty Distributed Systems

2005-13 Carole Delporte-Gallet, Hugues Fauconnier, Felix C. Freiling: Revisiting Failure Detection and Consensus in Omission Failure Environments

2005-14 Felix C. Freiling, Sukumar Ghosh: Code Stabilization

2005-15 Uwe Naumann: The Complexity of Derivative Computation

2005-16 Uwe Naumann: Syntax-Directed Derivative Code (Part I: Tangent-Linear Code)

2005-17 Uwe Naumann: Syntax-directed Derivative Code (Part II: Intraprocedural Adjoint Code)

2005-18 Thomas von der Maßen, Klaus Müller, John MacGregor, Eva Geisberger, Jörg Dörr, Frank Houdek, Harbhajan Singh, Holger Wußmann, Hans-Veit Bacher, Barbara Paech: Einsatz von Features im Software-Entwicklungsprozess - Abschlußbericht des GI-Arbeitskreises "Features"

2005-19 Uwe Naumann, Andre Vehreschild: Tangent-Linear Code by Augmented LL-Parsers

2005-20 Felix C. Freiling, Martin Mink: Bericht über den Workshop zur Ausbildung im Bereich IT-Sicherheit Hochschulausbildung, berufliche Weiterbildung, Zertifizierung von Ausbildungsangeboten am 11. und 12. August 2005 in Köln organisiert von RWTH Aachen in Kooperation mit BITKOM, BSI, DLR und Gesellschaft fuer Informatik (GI) e.V.

2005-21 Thomas Noll, Stefan Rieger: Optimization of Straight-Line Code Revisited

2005-22 Felix Freiling, Maurice Herlihy, Lucia Draque Penso: Optimal Randomized Fair Exchange with Secret Shared Coins

2005-23 Heiner Ackermann, Alantha Newman, Heiko Röglin, Berthold Vöcking: Decision Making Based on Approximate and Smoothed Pareto Curves

2005-24 Alexander Becher, Zinaida Benenson, Maximillian Dornseif: Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks

2006-01 * Fachgruppe Informatik: Jahresbericht 2005

2006-02 Michael Weber: Parallel Algorithms for Verification of Large Systems

2006-03 Michael Maier, Uwe Naumann: Intraprocedural Adjoint Code Generated by the Differentiation-Enabled NAGWare Fortran Compiler

2006-04 Ebadollah Varnik, Uwe Naumann, Andrew Lyons: Toward Low Static Memory Jacobian Accumulation

2006-05 Uwe Naumann, Jean Utke, Patrick Heimbach, Chris Hill, Derya Ozyurt, Carl Wunsch, Mike Fagan, Nathan Tallent, Michelle Strout: Adjoint Code by Source Transformation with OpenAD/F

2006-06 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Divide-and-Color

2006-07 Thomas Colcombet, Christof Löding: Transforming structures by set interpretations

2006-08 Uwe Naumann, Yuxiao Hu: Optimal Vertex Elimination in Single-Expression-Use Graphs

2006-09 Tingting Han, Joost-Pieter Katoen: Counterexamples in Probabilistic Model Checking

2006-10 Mesut Günes, Alexander Zimmermann, Martin Wenig, Jan Ritzerfeld, Ulrich Meis: From Simulations to Testbeds - Architecture of the Hybrid MCG-Mesh Testbed

2006-11 Bastian Schlich, Michael Rohrbach, Michael Weber, Stefan Kowalewski: Model Checking Software for Microcontrollers

2006-12 Benedikt Bollig, Joost-Pieter Katoen, Carsten Kern, Martin Leucker: Replaying Play in and Play out: Synthesis of Design Models from Scenarios by Learning

2006-13 Wong Karianto, Christof Löding: Unranked Tree Automata with Sibling Equalities and Disequalities

2006-14 Danilo Beuche, Andreas Birk, Heinrich Dreier, Andreas Fleischmann, Heidi Galle, Gerald Heller, Dirk Janzen, Isabel John, Ramin Tavakoli Kolagari, Thomas von der Maßen, Andreas Wolfram: Report of the GI Work Group "Requirements Management Tools for Product Line Engineering"

2006-15 Sebastian Ullrich, Jakob T. Valvoda, Torsten Kuhlen: Utilizing optical sensors from mice for new input devices

2006-16 Rafael Ballagas, Jan Borchers: Selexels: a Conceptual Framework for Pointing Devices with Low Expressiveness

2006-17 Eric Lee, Henning Kiel, Jan Borchers: Scrolling Through Time: Improving Interfaces for Searching and Navigating Continuous Audio Timelines

2007-01 * Fachgruppe Informatik: Jahresbericht 2006

2007-02 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, and Harald Zankl: SAT Solving for Termination Analysis with Polynomial Interpretations

2007-03 Jürgen Giesl, René Thiemann, Stephan Swiderski, and Peter Schneider-Kamp: Proving Termination by Bounded Increase

2007-04 Jan Buchholz, Eric Lee, Jonathan Klein, and Jan Borchers: coJIVE: A System to Support Collaborative Jazz Improvisation

2007-05 Uwe Naumann: On Optimal DAG Reversal

2007-06 Joost-Pieter Katoen, Thomas Noll, and Stefan Rieger: Verifying Concurrent List-Manipulating Programs by LTL Model Checking

2007-07 Alexander Nyßen, Horst Lichter: MeDUSA - MethoD for UML2-based Design of Embedded Software Applications

2007-08 Falk Salewski and Stefan Kowalewski: Achieving Highly Reliable Embedded Software: An empirical evaluation of different approaches

2007-09 Tina Kraußer, Heiko Mantel, and Henning Sudbrock: A Probabilistic Justification of the Combining Calculus under the Uniform Scheduler Assumption

2007-10 Martin Neuhäußer, Joost-Pieter Katoen: Bisimulation and Logical Preservation for Continuous-Time Markov Decision Processes

2007-11 Klaus Wehrle (editor): 6. Fachgespräch Sensornetzwerke

2007-12 Uwe Naumann: An L-Attributed Grammar for Adjoint Code

2007-13 Uwe Naumann, Michael Maier, Jan Riehme, and Bruce Christianson: Second-Order Adjoints by Source Code Manipulation of Numerical Programs

2007-14 Jean Utke, Uwe Naumann, Mike Fagan, Nathan Tallent, Michelle Strout, Patrick Heimbach, Chris Hill, and Carl Wunsch: OpenAD/F: A Modular, Open-Source Tool for Automatic Differentiation of Fortran Codes

2007-15 Volker Stolz: Temporal assertions for sequential and concurrent programs

2007-16 Sadeq Ali Makram, Mesut Güneç, Martin Wenig, Alexander Zimmermann: Adaptive Channel Assignment to Support QoS and Load Balancing for Wireless Mesh Networks

2007-17 René Thiemann: The DP Framework for Proving Termination of Term Rewriting

2007-18 Uwe Naumann: Call Tree Reversal is NP-Complete

2007-19 Jan Riehme, Andrea Walther, Jörg Stiller, Uwe Naumann: Adjoints for Time-Dependent Optimal Control

2007-20 Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf: Three-Valued Abstraction for Probabilistic Systems

2007-21 Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre: Compositional Modeling and Minimization of Time-Inhomogeneous Markov Chains

2007-22 Heiner Ackermann, Paul W. Goldberg, Vahab S. Mirrokni, Heiko Röglin, and Berthold Vöcking: Uncoordinated Two-Sided Markets

2008-01 * Fachgruppe Informatik: Jahresbericht 2007/2008

2008-02 Henrik Bohnenkamp, Marielle Stoelinga: Quantitative Testing

2008-03 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, Harald Zankl: Maximal Termination

2008-04 Uwe Naumann, Jan Riehme: Sensitivity Analysis in Sisyphe with the AD-Enabled NAGWare Fortran Compiler

2008-05 Frank G. Radmacher: An Automata Theoretic Approach to the Theory of Rational Tree Relations

2008-06 Uwe Naumann, Laurent Hascoet, Chris Hill, Paul Hovland, Jan Riehme, Jean Utke: A Framework for Proving Correctness of Adjoint Message Passing Programs

2008-07 Alexander Nyßen, Horst Lichter: The MeDUSA Reference Manual, Second Edition

2008-08 George B. Mertzios, Stavros D. Nikolopoulos: The λ-cluster Problem on Parameterized Interval Graphs

2008-09 George B. Mertzios, Walter Unger: An optimal algorithm for the k-fixed-endpoint path cover on proper interval graphs

2008-10 George B. Mertzios, Walter Unger: Preemptive Scheduling of Equal-Length Jobs in Polynomial Time

2008-11 George B. Mertzios: Fast Convergence of Routing Games with Splittable Flows

2008-12 Joost-Pieter Katoen, Daniel Klink, Martin Leucker, Verena Wolf: Abstraction for stochastic systems by Erlang's method of stages

2008-13 Beatriz Alarcón, Fabian Emmes, Carsten Fuhs, Jürgen Giesl, Raúl Gutiérrez, Salvador Lucas, Peter Schneider-Kamp, René Thiemann: Improving Context-Sensitive Dependency Pairs

2008-14 Bastian Schlich: Model Checking of Software for Microcontrollers

2008-15 Joachim Kneis, Alexander Langer, Peter Rossmanith: A New Algorithm for Finding Trees with Many Leaves

2008-16 Hendrik vom Lehn, Elias Weingärtner and Klaus Wehrle: Comparing recent network simulators: A performance evaluation study

2008-17 Peter Schneider-Kamp: Static Termination Analysis for Prolog using Term Rewriting and SAT Solving

2008-18 Falk Salewski: Empirical Evaluations of Safety-Critical Embedded Systems

2008-19 Dirk Wilking: Empirical Studies for the Application of Agile Methods to Embedded Systems

2009-01 * Fachgruppe Informatik: Jahresbericht 2009

2009-02 Taolue Chen, Tingting Han, Joost-Pieter Katoen, Alexandru Mereacre: Quantitative Model Checking of Continuous-Time Markov Chains Against Timed Automata Specifications

2009-03 Alexander Nyßen: Model-Based Construction of Embedded Real-Time Software - A Methodology for Small Devices

2009-05 George B. Mertzios, Ignasi Sau, Shmuel Zaks: A New Intersection Model and Improved Algorithms for Tolerance Graphs

2009-06 George B. Mertzios, Ignasi Sau, Shmuel Zaks: The Recognition of Tolerance and Bounded Tolerance Graphs is NP-complete

2009-07 Joachim Kneis, Alexander Langer, Peter Rossmanith: Derandomizing Non-uniform Color-Coding I

2009-08 Joachim Kneis, Alexander Langer: Satellites and Mirrors for Solving Independent Set on Sparse Graphs

2009-09 Michael Nett: Implementation of an Automated Proof for an Algorithm Solving the Maximum Independent Set Problem

2009-10 Felix Reidl, Fernando Sánchez Villaamil: Automatic Verification of the Correctness of the Upper Bound of a Maximum Independent Set Algorithm

2009-11 Kyriaki Ioannidou, George B. Mertzios, Stavros D. Nikolopoulos: The Longest Path Problem is Polynomial on Interval Graphs

2009-12 Martin Neuhäußer, Lijun Zhang: Time-Bounded Reachability in Continuous-Time Markov Decision Processes

2009-13 Martin Zimmermann: Time-optimal Winning Strategies for Poset Games

2009-14 Ralf Huuck, Gerwin Klein, Bastian Schlich (eds.): Doctoral Symposium on Systems Software Verification (DS SSV'09)

2009-15 Joost-Pieter Katoen, Daniel Klink, Martin Neuhäußer: Compositional Abstraction for Stochastic Systems

2009-16 George B. Mertzios, Derek G. Corneil: Vertex Splitting and the Recognition of Trapezoid Graphs

2009-17 Carsten Kern: Learning Communicating and Nondeterministic Automata

2009-18 Paul Hänsch, Michaela Slaats, Wolfgang Thomas: Parametrized Regular Infinite Games and Higher-Order Pushdown Strategies

2010-01 * Fachgruppe Informatik: Jahresbericht 2010

2010-02 Daniel Neider, Christof Löding: Learning Visibly One-Counter Automata in Polynomial Time

2010-03 Holger Krahn: MontiCore: Agile Entwicklung von domänenspezifischen Sprachen im Software-Engineering

2010-04 René Wörzberger: Management dynamischer Geschäftsprozesse auf Basis statischer Prozessmanagementsysteme

2010-05 Daniel Retkowitz: Softwareunterstützung für adaptive eHome-Systeme

2010-06 Taolue Chen, Tingting Han, Joost-Pieter Katoen, Alexandru Mereacre: Computing maximum reachability probabilities in Markovian timed automata

2010-07 George B. Mertzios: A New Intersection Model for Multitolerance Graphs, Hierarchy, and Efficient Algorithms

2010-08 Carsten Otto, Marc Brockschmidt, Christian von Essen, Jürgen Giesl: Automated Termination Analysis of Java Bytecode by Term Rewriting

2010-09 George B. Mertzios, Shmuel Zaks: The Structure of the Intersection of Tolerance and Cocomparability Graphs

2010-10 Peter Schneider-Kamp, Jürgen Giesl, Thomas Ströder, Alexander Serebrenik, René Thiemann: Automated Termination Analysis for Logic Programs with Cut

2010-11 Martin Zimmermann: Parametric LTL Games

2010-12 Thomas Ströder, Peter Schneider-Kamp, Jürgen Giesl: Dependency Triples for Improving Termination Analysis of Logic Programs with Cut

2010-13 Ashraf Armoush: Design Patterns for Safety-Critical Embedded Systems

2010-14 Michael Codish, Carsten Fuhs, Jürgen Giesl, Peter Schneider-Kamp: Lazy Abstraction for Size-Change Termination

2010-15 Marc Brockschmidt, Carsten Otto, Christian von Essen, Jürgen Giesl: Termination Graphs for Java Bytecode

2010-16 Christian Berger: Automating Acceptance Tests for Sensor- and Actuator-based Systems on the Example of Autonomous Vehicles

2010-17 Hans Grönniger: Systemmodell-basierte Definition objektbasierter Modellierungssprachen mit semantischen Variationspunkten

2010-18 Ibrahim Armaç: Personalisierte eHomes: Mobilität, Privatsphäre und Sicherheit

2010-19 Felix Reidl: Experimental Evaluation of an Independent Set Algorithm

2010-20 Wladimir Fridman, Christof Löding, Martin Zimmermann: Degrees of Lookahead in Context-free Infinite Games

2011-01 * Fachgruppe Informatik: Jahresbericht 2011

2011-02 Marc Brockschmidt, Carsten Otto, Jürgen Giesl: Modular Termination Proofs of Recursive Java Bytecode Programs by Term Rewriting

2011-03 Lars Noschinski, Fabian Emmes, Jürgen Giesl: A Dependency Pair Framework for Innermost Complexity Analysis of Term Rewrite Systems

2011-04 Christina Jansen, Jonathan Heinen, Joost-Pieter Katoen, Thomas Noll: A Local Greibach Normal Form for Hyperedge Replacement Grammars

2011-06 Johannes Lotz, Klaus Leppkes, and Uwe Naumann: dco/c++ - Derivative Code by Overloading in C++

2011-07 Shahar Maoz, Jan Oliver Ringert, Bernhard Rumpe: An Operational Semantics for Activity Diagrams using SMV

2011-08 Thomas Ströder, Fabian Emmes, Peter Schneider-Kamp, Jürgen Giesl, Carsten Fuhs: A Linear Operational Semantics for Termination and Complexity Analysis of ISO Prolog

2011-09 Markus Beckers, Johannes Lotz, Viktor Mosenkis, Uwe Naumann (Editors): Fifth SIAM Workshop on Combinatorial Scientific Computing

2011-10 Markus Beckers, Viktor Mosenkis, Michael Maier, Uwe Naumann: Adjoint Subgradient Calculation for McCormick Relaxations

2011-11 Nils Jansen, Erika Ábrahám, Jens Katelaan, Ralf Wimmer, Joost-Pieter Katoen, Bernd Becker: Hierarchical Counterexamples for Discrete-Time Markov Chains

2011-12 Ingo Felscher, Wolfgang Thomas: On Compositional Failure Detection in Structured Transition Systems

2011-13 Michael Förster, Uwe Naumann, Jean Utke: Toward Adjoint OpenMP

2011-14 Daniel Neider, Roman Rabinovich, Martin Zimmermann: Solving Muller Games via Safety Games

2011-16 Niloofar Safiran, Uwe Naumann: Toward Adjoint OpenFOAM

2011-17 Carsten Fuhs: SAT Encodings: From Constraint-Based Termination Analysis to Circuit Synthesis

2011-18 Kamal Barakat: Introducing Timers to pi-Calculus

2011-19 Marc Brockschmidt, Thomas Ströder, Carsten Otto, Jürgen Giesl: Automated Detection of Non-Termination and NullPointerExceptions for Java Bytecode

2011-24 Callum Corbett, Uwe Naumann, Alexander Mitsos: Demonstration of a Branch-and-Bound Algorithm for Global Optimization using McCormick Relaxations

2011-25 Callum Corbett, Michael Maier, Markus Beckers, Uwe Naumann, Amin Ghobeity, Alexander Mitsos: Compiler-Generated Subgradient Code for McCormick Relaxations

2011-26 Hongfei Fu: The Complexity of Deciding a Behavioural Pseudometric on Probabilistic Automata

2012-01 Fachgruppe Informatik: Annual Report 2012

2012-02 Thomas Heer: Controlling Development Processes

2012-03 Arne Haber, Jan Oliver Ringert, Bernhard Rumpe: MontiArc - Architectural Modeling of Interactive Distributed and Cyber-Physical Systems

2012-04 Marcus Gelderie: Strategy Machines and their Complexity

2012-05 Thomas Ströder, Fabian Emmes, Jürgen Giesl, Peter Schneider-Kamp, and Carsten Fuhs: Automated Complexity Analysis for Prolog by Term Rewriting

2012-06 Marc Brockschmidt, Richard Musiol, Carsten Otto, Jürgen Giesl: Automated Termination Proofs for Java Programs with Cyclic Data

2012-07 André Egners, Björn Marschollek, and Ulrike Meyer: Hackers in Your Pocket: A Survey of Smartphone Security Across Platforms

2012-08 Hongfei Fu: Computing Game Metrics on Markov Decision Processes

2012-09 Dennis Guck, Tingting Han, Joost-Pieter Katoen, and Martin R. Neuhäußer: Quantitative Timed Analysis of Interactive Markov Chains

2012-10 Uwe Naumann and Johannes Lotz: Algorithmic Differentiation of Numerical Methods: Tangent-Linear and Adjoint Direct Solvers for Systems of Linear Equations

2012-12 Jürgen Giesl, Thomas Ströder, Peter Schneider-Kamp, Fabian Emmes, and Carsten Fuhs: Symbolic Evaluation Graphs and Term Rewriting — A General Methodology for Analyzing Logic Programs

2012-15	Uwe Naumann, Johannes Lotz, Klaus Leppkes, and Markus Towara: Algorithmic Differentiation of Numerical Methods: Tangent-Linear and Adjoint Solvers for Systems of Nonlinear Equations
2012-16	Georg Neugebauer and Ulrike Meyer: SMC-MuSe: A Framework for Secure Multi-Party Computation on MultiSets
2013-01 *	Fachgruppe Informatik: Annual Report 2013
2013-03	Markus Towara and Uwe Naumann: A Discrete Adjoint Model for OpenFOAM
2013-04	Max Sagebaum, Nicolas R. Gauger, Uwe Naumann, Johannes Lotz, and Klaus Leppkes: Algorithmic Differentiation of a Complex C++ Code with Underlying Libraries

* These reports are only available as a printed version.

Please contact `biblio@informatik.rwth-aachen.de` to obtain copies.