



## SEACON2009

Erfolgsfaktoren der Softwareentwicklung



## Agenda

- KN Login
- Ausgangssituation
- Erfolgsfaktoren



## KN Login

### Migration und Weiterentwicklung

- weltweites Portal für K+N Kunden
  - > 13.000 Kunden
  - > 18.000 registrierte Nutzer + anonymer Bereich
- Tracking & Tracing für Kunden- und Transportaufträge über die gesamte Supply Chain
- viele operative Funktionen, z. B.
  - elektronische Buchungen
  - Packlisten
  - Data Extracts
  - Interactive Delivery Planner
- 2007 wurde die Ablösung des Altsystems initiiert (erste Teilmigration komplett Q1/2008)
- 2008 begann die Neuentwicklung weiterer Teile der operativen Funktionen
- Entwicklungsteam: 50-60 MA



## Ausgangssituation

### Herausforderungen

- Feinspezifikation wurde von Business Analysten erstellt → lange Reviewphasen
- Entwicklungen wurden ungenügend projektiert
- kein Standardvorgehen für Softwareentwicklung (Projektmanagement, technisches Design, ...)
  - keine Definition / Kontrolle der Standardartefakte
  - keine (einheitlichen) Projektpläne
  - unklares Projektsetup (1/2/3/... Projektleiter → kein Projektleiter)
- unklare Zuständigkeiten
  - jedes Deployment war ein individueller Fall
  - tw. Spaghetti-Code
  - ungenügende technische Dokumentation
- technische Basis (Framework) existierte, aber keine dedizierte FW-Entwicklung und -betreuung
- Großprojekte standen an
- Personalaufbau notwendig



## Grundsätze

- Entscheidung gegen
  - kostenintensive externe Beratung
  - kostenintensive Tools
- Entscheidung für
  - am Markt bekannte und bewährte Vorgehen/Tools und Adaption
  - frei verfügbare Software (Open Source, Freeware)
  - pragmatische Lösungen
  - kontinuierliche Verbesserung
- kein agiler Ansatz
- aber: Verwendung agiler Maßnahmen/Tools
- Reduktion der Entwicklungskosten durch near-shoring
- Fokussierung auf die tragende Säule: die Mitarbeiter
  - regelmäßige Kommunikation notwendiger Änderungen
  - intensive Weiterbildung
  - Mitarbeiterbindung / geringe Fluktuation



## Erfolgsfaktoren

### Projektmanagement

- Ausarbeitung eines Standardprozesses (SDP)
  - einheitliches Projektsetup
  - Definition von Standardartefakten
  - einheitliches Controlling über ein zentrales Tool
  - Standardapprovals für Phasenübergänge
  - Project Quality Assurance
- Changes werden flächendeckend projektiert und unterliegen damit dem Standardprozess
- Reorganisation der Linienführung
  - Zentralisierung der Verantwortlichkeiten im Bereich Spezifikation
  - Spezifikation ist nun Teil des Implementierungsprojekts
- Optimierung des Spezifikationsprozesses
  - führt zu höherer Qualität im Gesamtprozess
  - weniger Reviewaufwand, schnellere Entwicklung, weniger Testaufwand
- Einführung eines zentralen Releasemanagements



## Erfolgsfaktoren

### Arbeitsumgebung

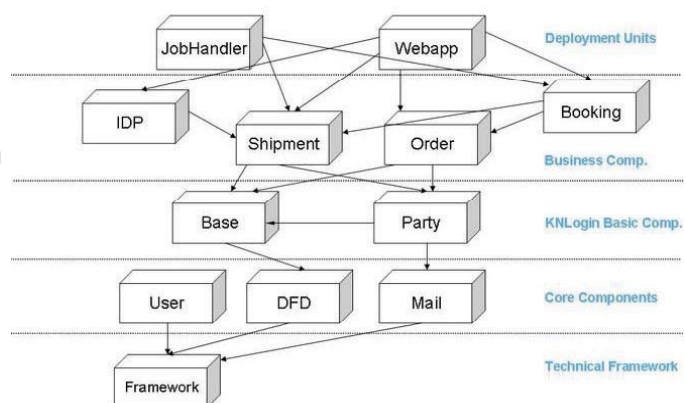
- Einarbeitung für neue Mitarbeiter ist kurz
  - Framework setzt Konventionen durch / Software hat hohen Wiedererkennungswert
  - Eclipse-Plugins für Codegenerierung
  - leichtgewichtige Architektur (POJO-based)
- Anwendung kann auf einem lokalen PC betreut werden
  - lokaler Applikations- und DB-Server
  - generischer XML-Testdaten-Generator
- klare Definition der Zuständigkeiten
  - Architekt in jedem Projekt involviert
  - dediziertes Framework-Team
- Standardtools für Incident Handling, technisches Design/Dokumentation, ...
- geringe Fluktuation
- kontinuierliche Verbesserung aus dem Team



## Erfolgsfaktoren

### Langlebige Architektur

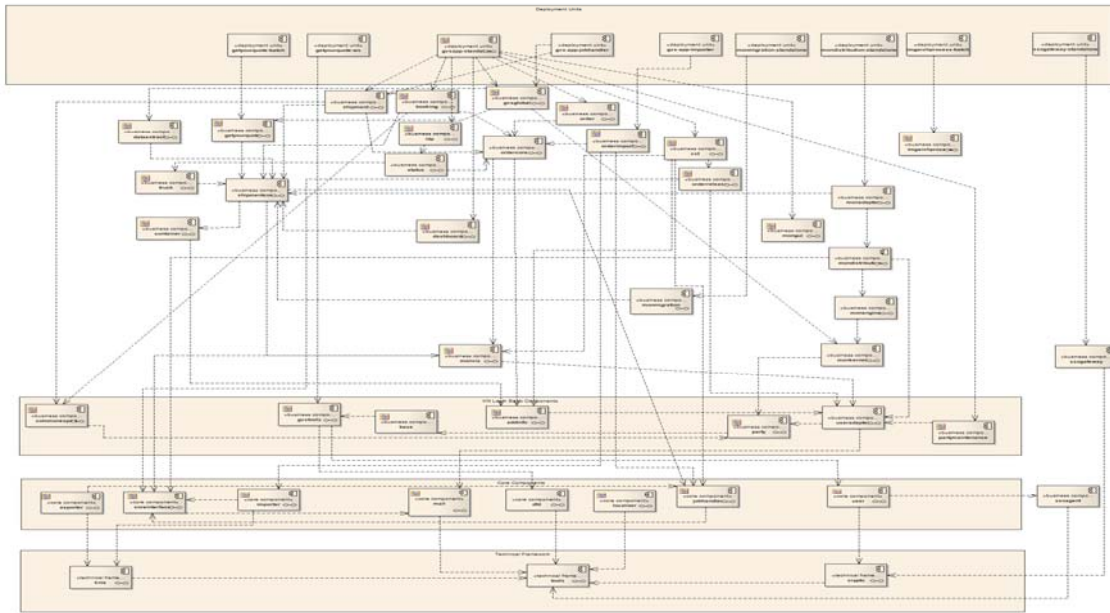
- Aufteilung der Applikation in Komponenten
  - divide and conquer
  - Komponentenverantwortliche
- klare Strukturierung der technischen Komponenten
  - Technical Framework / Core/Basic/Business Components
  - Code Reviews im Projekt durch KV
- rechtzeitige Refaktorisierung





# Erfolgsfaktoren

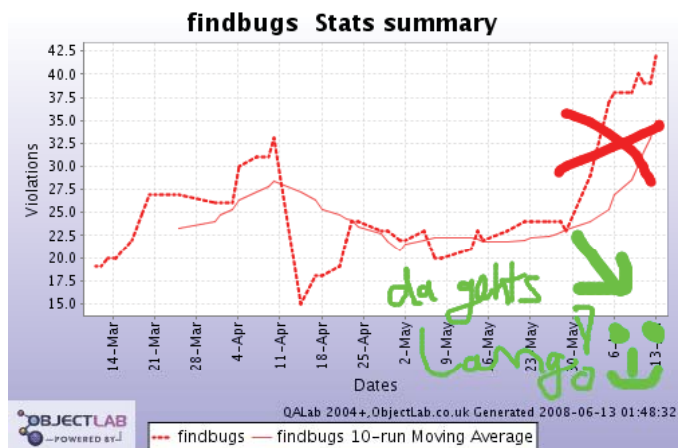
Technical Framework / Core/Basic/Business Components



# Erfolgsfaktoren

(Voll-)Automatische Codeüberprüfung I

- Etablierung von nightly builds
- Integration von automatischen Regressionstests
- Integration von findbugs
  - findbugs-Auswertung durch den Architekten
  - Fixing durch KV
- automatisches Deployment auf Integrationstestsystem





# Erfolgsfaktoren

## (Voll-)Automatische Codeüberprüfung II

- automatische Überprüfung der Dokumentation vs. Code
- Auswertung durch den Architekten
- Fixing durch KV

### WebDev Code/Documentation Consistency Check Report

Last run Wed Feb 18 13:13:02 CET 2009

- Java workspace path: C:\Data\workspace
- EA model XML path: C:\Data\workspace\ea-model\Product\_Models\Technical\XML\WebDevelopment

#Modules run	Total #checks performed	Total #checks failed	Overall correctness ratio	Trend
13	153603	3069	85%	

#### ICDL: Insufficient Code Documentation Length

This module parses all BC, I\*Service and \*Dialog types from EA XMLs for the length of their Notes. Additionally the public methods' comments of I\*Service and \*Dialog types are checked. The notes are filled with the Javadoc of the corresponding Java class with every code sync so errors reported by this module must be fixed in the Javadoc and not in EA. This module can of course only check the existence and quantity of documentation but not its quality.

Module total	#checks performed	#Checks failed	Correctness ratio	Trend
Double-click row to expand	4537	223	95%	

#### IDMD: Insufficient Data Model Documentation

This module parses all table stereotypes from EA XMLs for the length of their Notes. Additionally it is checked if the notes contain any special character like ä,ö,ü,Ä,Ö,Ü,ä,ö,ü etc. which would cause problems during DDL script execution.

Module total	#checks performed	#Checks failed	Correctness ratio	Trend
Double-click row to expand	5030	2173	56%	

#### MCD: Missing Component Dependencies

For every business component this module parses all import statements from all its classes in order to find dependencies to a) other business components and b) external libraries that are not reflected in the EA Model XMLs. Missing dependencies are reported as errors. Likewise all redundant dependencies are reported as errors.

Module total	#checks performed	#Checks failed	Correctness ratio	Trend
Double-click row to expand	41116	1	99%	

#### ADF: Architecture Design Flaws

This module checks the dependency direction for each component. If a component depends on another component which has a higher instability (I) value than itself it is considered a design flaw.

Module total	#checks performed	#Checks failed	Correctness ratio	Trend
Double-click row to expand	668	10	98%	

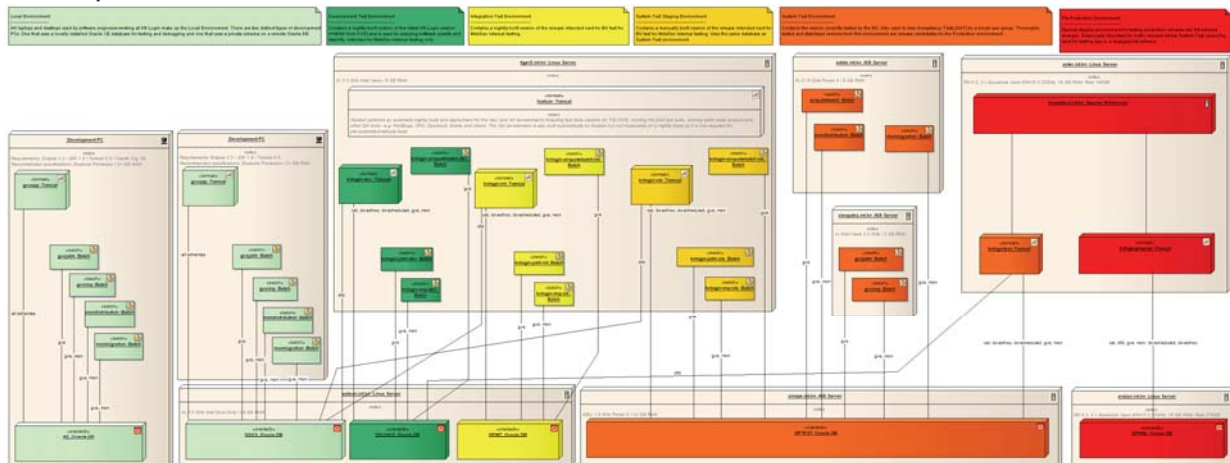
CDI: Component Dependency Details



# Erfolgsfaktoren

## Betrieb

- Erstellung und Pflege der Systemdokumentation
- neue Rollen System Owner/System Responsible





## Erfolgsfaktoren

### Betrieb

- aktives Monitoring der Systeme und Applikationen
- automatische Performancestatistiken

ReportFrom	ReportTo	Hits
05.01.2009 00:00	15.02.2009 23:59	4357030

#### ResponseTime Ranges

Calendarweek	Avg (ms)	Min (ms)	Max (ms)	<5s (%)	< 10s (%)	< 30s (%)	< 60s (%)	> 60s (%)	Hits
2009-07	1743	0	24519206	95,92	1,15	0,61	0,17	0,12	641546
2009-06	1219	0	7669081	95,80	1,19	0,63	0,16	0,11	625648
2009-05	1740	0	9245658	95,55	1,06	0,60	0,21	0,14	573660
2009-04	1278	0	10542553	96,31	0,89	0,51	0,15	0,11	728441
2009-03	1857	0	29840736	97,30	1,06	0,51	0,19	0,17	828722
2009-02	1417	0	6432996	97,64	0,78	0,43	0,18	0,13	959013

#### Times per Layer

Calendarweek	Avg (ms)	DAO (%)	Dialog (%)	Dtobo (%)	JSP (%)	Request (%)	Service (%)	Hits
2009-07	1743	33,747173	7,39692881	0,54115894	0,6823201	52,2638141	5,3686055	641546
2009-06	1219	48,862695	10,4717001	0,83123231	0,9202706	30,5204307	8,3936714	625648
2009-05	1740	41,881565	7,51008402	0,53933911	0,6154048	43,3004008	6,1532067	573660
2009-04	1278	50,060028	8,44966457	0,68074472	1,1035469	32,4561555	7,2498605	728441
2009-03	1857	25,826077	8,87567115	0,47665737	1,3094652	56,7051757	6,8069532	828722
2009-02	1417	29,673804	20,8100641	1,65820786	4,0850026	34,6421101	9,1308114	959013



## Erfolge

### die nackten Zahlen

- Performance:
  - durchschnittlich 700K Requests pro Woche
  - Abarbeitung: 96 % < 5 sec
  - das bei 1,5 TB Daten in der DB
- Verfügbarkeit des Produktionssystems: > 99 %
- Anzahl von Systemen, die mit KN Login in Verbindung stehen: 12 (ohne Kundensysteme)
  - über DB-Link (Replikation)
  - über dateibasierten Transfer
  - über SOAP
- Einarbeitungszeit eines neuen Softwareentwicklers: < 2 Wochen
- Einrichtung Entwicklungsumgebung: max. 1/2 Tag (inkl. Test-DB, lokaler Server etc.)
- Komplettes Build inkl. Aufsetzen der lokalen Testdatenbank (Dropen und Neu-Erzeugen aller Tabellen und Views) und Befüllung mit Testdaten < 10 Min.
- Lokaler Start der Anwendung: < 1 Min
- Workspace Subversion Update (von > 220 Eclipse Projekten) < 10 Min
- Refresh der Anwendung nach Änderungen am Code: < 20 Sek.
- das bei > 1,5 Mio. lines of code



Letzte Folie

Vielen Dank!