qinet® „Software Factory“

Software Process Engine 3.1
Design & Generate

SW ProEngine GmbH

Agenda

- Vorstellung der Teilnehmer
- Kundensituation
- Überblick, Zielsetzung
- Vorteile und USP‘s
- Übersicht der Software Process Engine 3.1 Architektur
- Vorgehensweise für die Verwendung des Code-Generators
- Was liefert der Framework 2.0 + J2EE Template-Satz?
- Referenzen
- Weitere Vorgehensweise

Überblick / Zielsetzung

Der von **SW-ProEngine und ISST Fraunhofer Institut**

verwendete Einsatz

der **Code-Generierung** löst diese Aufgabenstellungen,

durch eine Zusammenführungen von

Design & Generate

in einem **ganzheitlichen Verfahrensweg**

mit einer **iterativen Vorgehensweise**

bei der Projektrealisierung:

Qualität, Geschwindigkeit, Flexibilität und Kosten

Vorteile und USPs der SW Process Engine 3.1 (I)

- Erhöhte Produktivität (90-98% des Codes ist maschinell erzeugt), dadurch schnelles „Time-to-market“ und Kostenreduzierung
- Durchgängige Einhaltung der Programmiervorgaben führt zu hohe Qualität, Wartbarkeit und Investitionsschutz
- Produzierter Code ist wie „von Hand geschrieben“ und einfach lesbar
- Transformation der Objekt-Modelle in lauffähige Programme ohne Medienbrüche (MDA)
- Interaktive und iterative Abstimmungszyklen mit den Fachabteilungen basierend auf einer lauffähige Basisanwendung.
- Entwickler fängt seine Tätigkeit mit einer laufenden Anwendung an (kein „leeres Blatt“), die gemäß Fachvorgaben maschinell erzeugt wurde.

Vorteile und USPs der *SW Process Engine 3.1* (II)

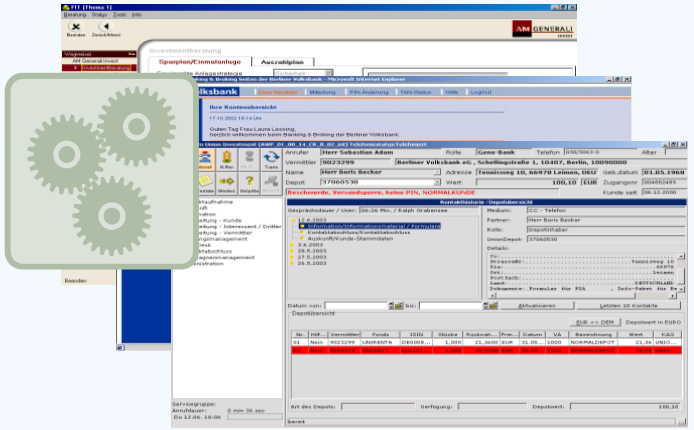
- Unabhängigkeit von Programmier-Sprachen
  - Templates (Schablonen) können für jede beliebige Sprache gepflegt und für die Code-Generierung verwendet werden
  - Templates sind auch für Online-Hilfe, Programmdokumentation, Testfälle und Fachkonzepte vorhanden
- Klare Trennung von Generator und Templates
- Templates werden ohne zusätzliche Skriptsprachen durch einfache Verwendung von Eingabemasken gepflegt
- für Code-Generierung in beliebigen Programmiersprachen ist keine Anpassung der Generator-Engine notwendig
- Templates-Werk und Objekt-Model werden in einem Repository gespeichert, das für die Datenkonsistenz sorgt
- Code-Wiederverwendung, modularer Aufbau,
  Service **O**rientierte **A**rchitektur

Software Process Engine: Übersicht der Architektur (I)

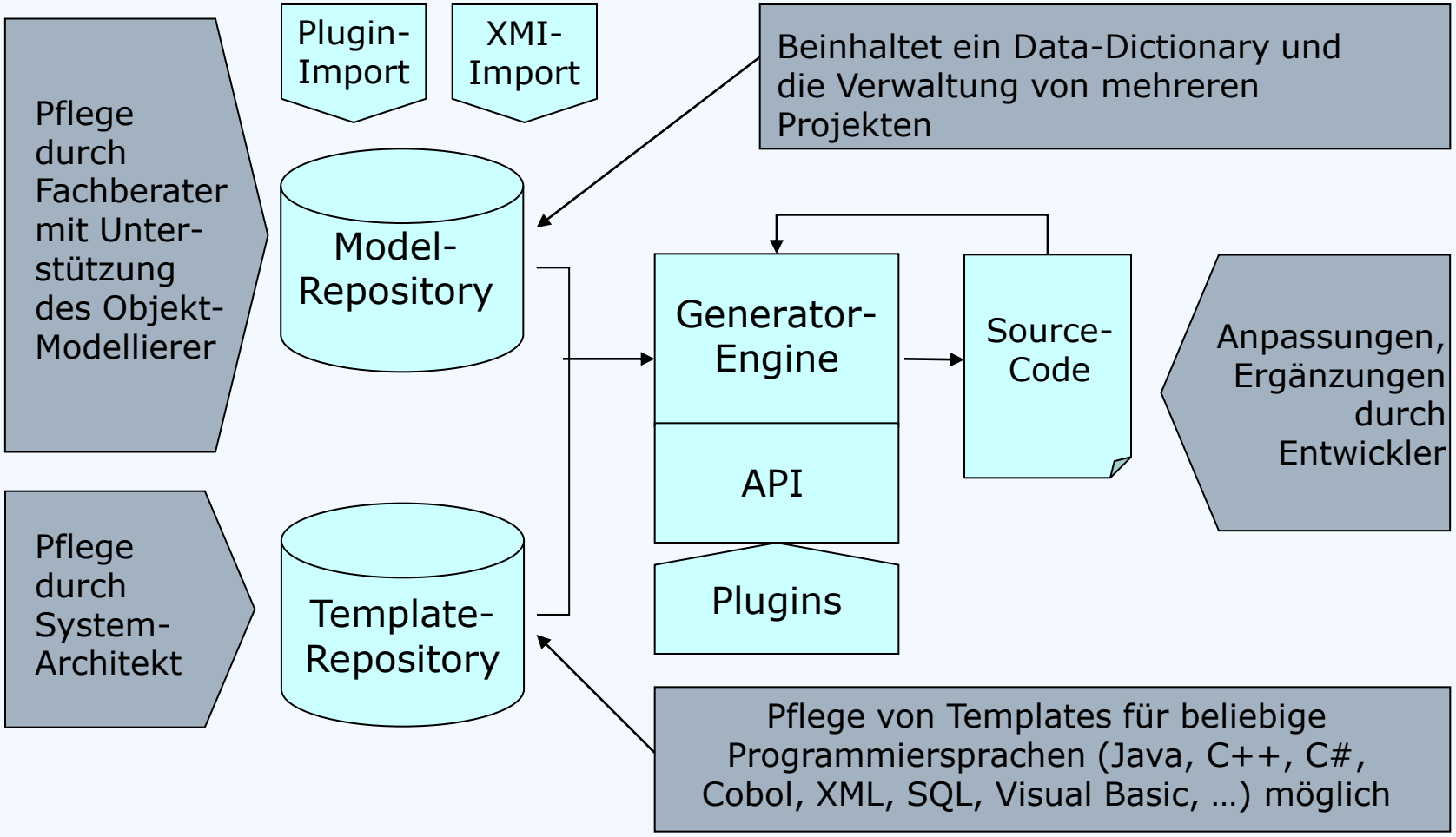
Software Process Engine 3.1 ist ein Tool-gestützter Verfahrensweg, der auf Knopfdruck

- anhand eines Bauplans (Geschäftsprozess-Definition) und
- ihm zur Verfügung stehenden Bauteile (Templates und Framework)

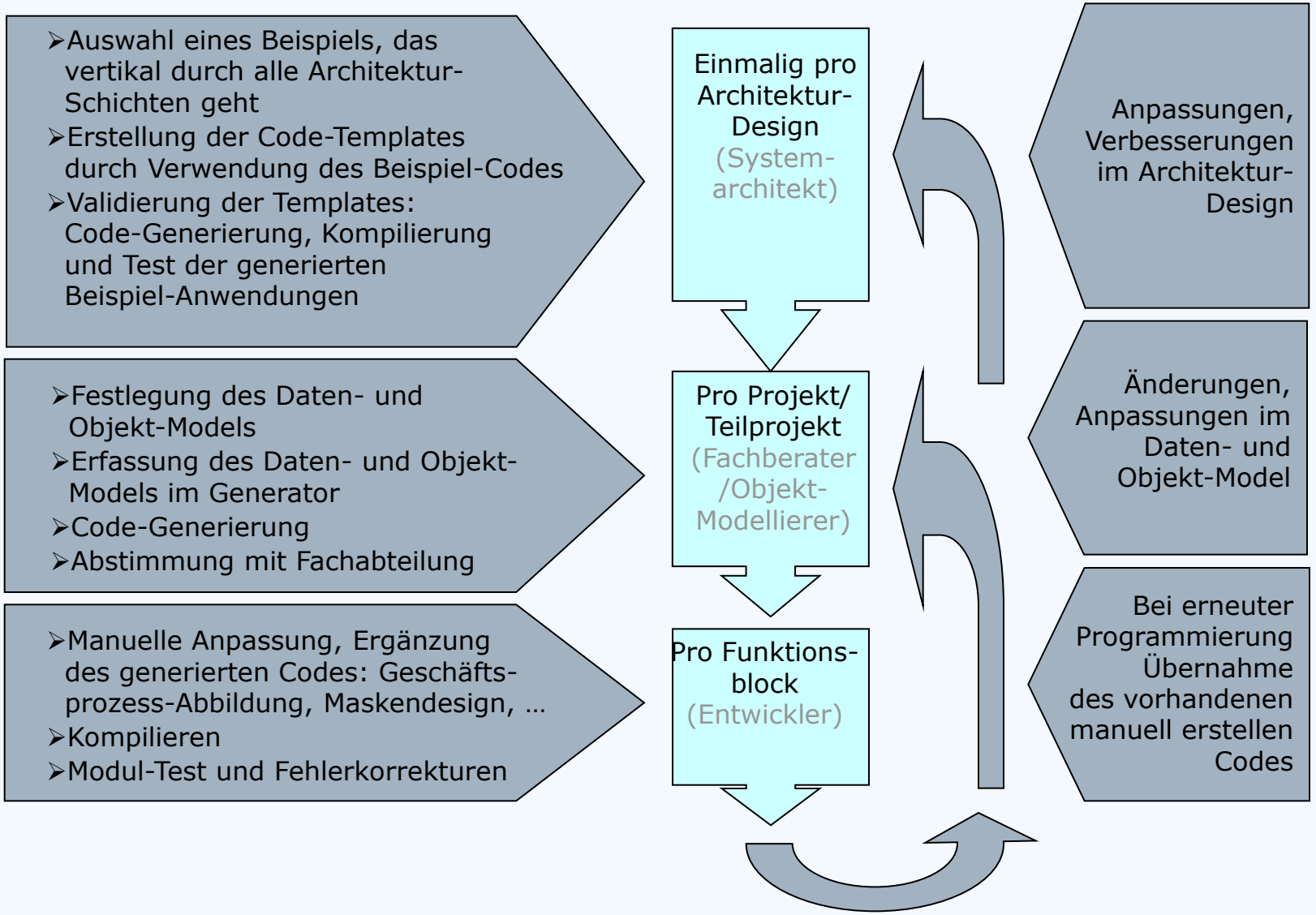
die Anwendung fertigt!



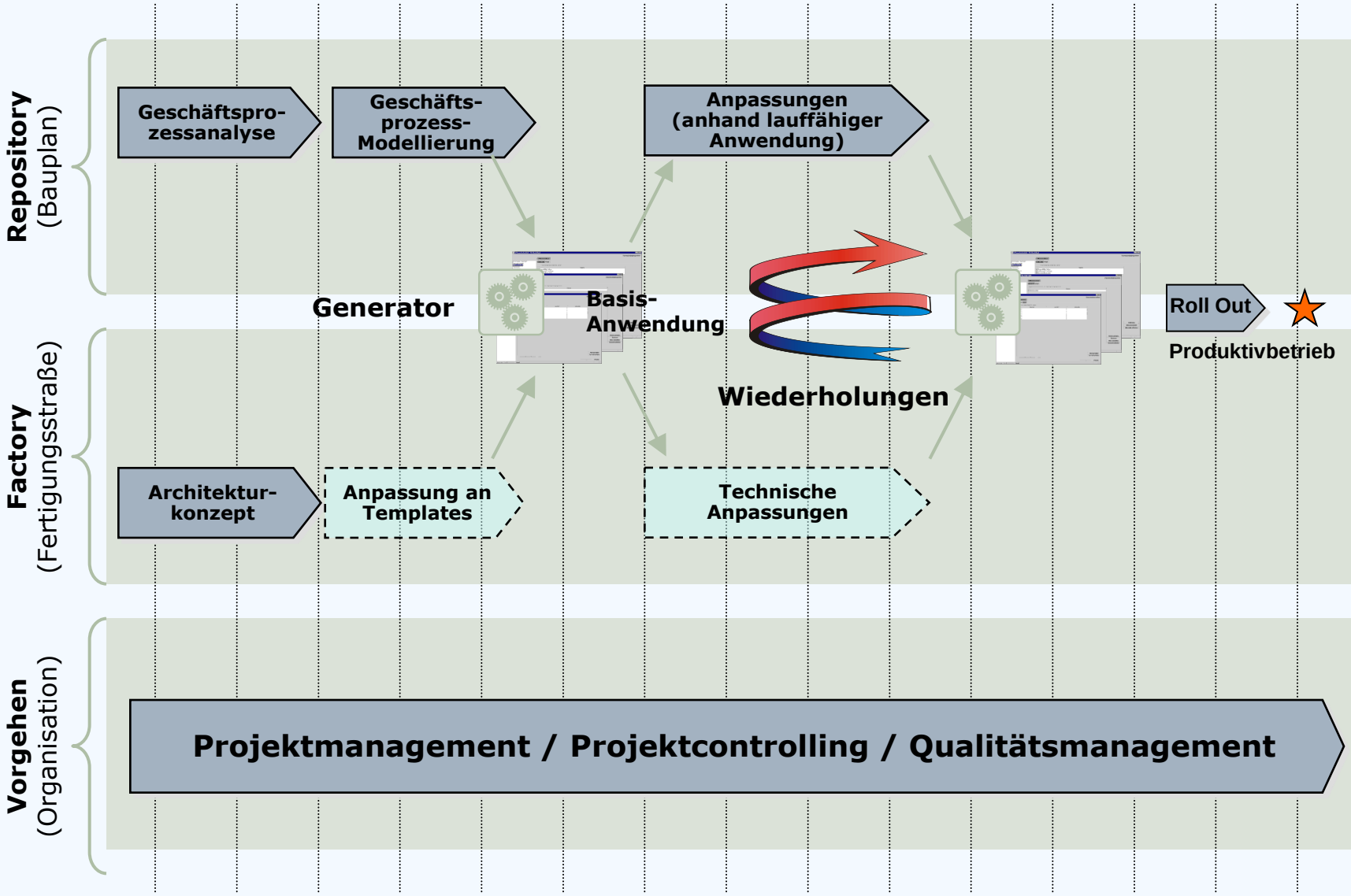
Software ProEngine: Übersicht der Architektur (II)



Iterative Vorgehensweise



Design & Generate im Zusammenspiel



Funktionsumfang des Business-Frameworks (I)

Allgemein

- Von hochskalierbaren Enterprise Anwendungen bis kleine Installationen mit gleichem architektonischen Aufbau
- Hohe Wiederverwendung von Code
- Bereitstellung allgemeiner Funktionen:
  - Mandantenfähigkeit
  - Multisprache
  - Multiuser Fähigkeit, Locking-Mechanismen
  - Verwaltung von User, Rollen und Rechte und Stellen
  - Berichtswesen / Reports
  - Prozessorientierte Bedienung, Workflow-Unterstützung, Wiedervorlage-Funktion
  - Online Help
  - Verwaltung und Anzeige von Meldungen (Fehler, Warnung, Info)

Funktionsumfang des Business-Frameworks (II)

Architekturen

- Swing-Client ➔ Datenbank
- Swing-Client ➔ JEE App. Server ➔ Datenbank
- HTML-Server / Servlet Container ➔ Datenbank
- HTML-Server / Srvl. Ct. ➔ JEE App. Server ➔ DB

Generierung

alle Fachkonzepte, Programmdokumentationen (JavaDoc), Applikationen, Deskriptoren, .Ini-Dateien, Masken, Datenbank-Strukturen etc.

Performance

Optimierungen für:
- Datenbankzugriffe
- Maskenaufbau
- Client-Server-Kommunikation

Funktionsumfang des Business-Frameworks (III)

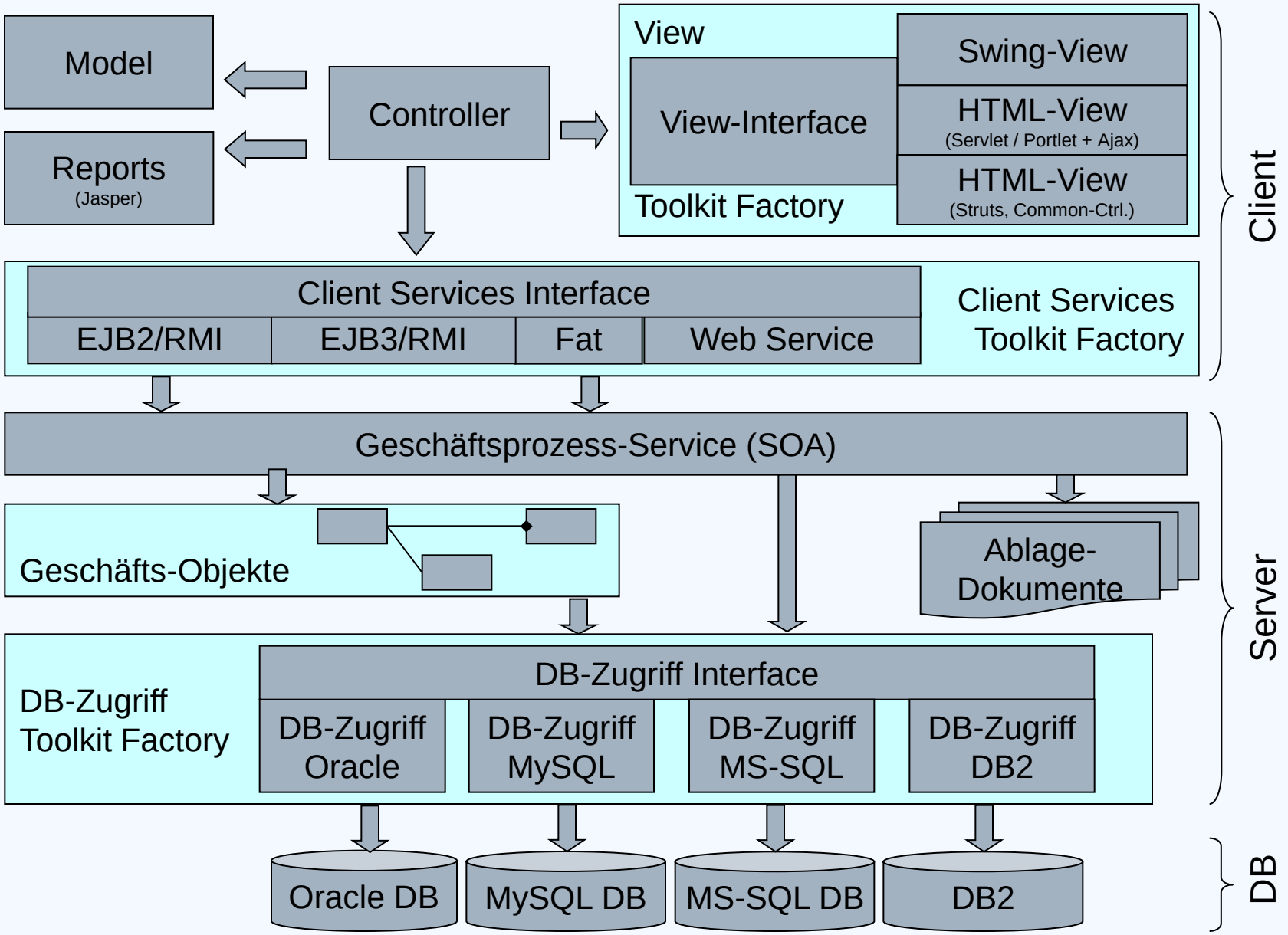
Client

- Frontend für Swing und HTML
- Weitere Frontend-Typen modular abbildbar
- Hohe Flexibilität in der Masken-Gestaltung und Benutzer-Interaktion in einem standardisierten Aufbau
- Abbildung des MVC Design Patterns
- Nur View unterschiedlich zwischen Swing und HTML: Controller und Model werden wieder verwendet
- Menüführung
- Multi-Workflow / Multi-Task mit ein Frontend-Rahmen
- Hierarchische Präsentation und Pflege der Daten
- Eingabe-Workflow / Assistenten (Wizzard)
- Kommunikation mit dem Server gekapselt, ersetzbar und modular erweiterbar
- Klare Trennung von der Business-Implementierung
- Zentrale Customizing der Maskengestaltung mit Styles

Funktionsumfang des Business-Frameworks (IV)

Server

- Implementierung von Business-Logik in einer serviceorientierte und wieder verwendbare Form
- Trennung zwischen Business-Services und Business-Modell
- Prüfung der Datenkonsistenz
- Transaktionsmanagement wahlweise:
  - In Server, pro Business-Service oder Serviceklammer
  - Alternativ auch Steuerung der Transaktionsklammer über Client möglich
- Datenhaltung:
  - Multidatenbank und Multidatenbank-Typ Zugriffe
  - Zugriff auf weitere Datenbanktypen modular abbildbar
  - Multiuser-Steuerung
  - Connection-Pooling (= Performance und Ressourcen sparen)
  - Automatische Erstellung und Anpassung der Datenbank-Strukturen
- Protokollierung / Logging


Model
Reports
(Jasper)
Controller
View
View-Interface
Toolkit Factory
Swing-View
HTML-View
(Servlet / Portlet + Ajax)
HTML-View
(Struts, Common-Ctrl.)
Client
Client Services Interface
EJB2/RMI
EJB3/RMI
Fat
Web Service
Client Services
Toolkit Factory
Geschäftsprozess-Service (SOA)
Geschäfts-Objekte
Ablage-
Dokumente
Server
DB-Zugriff
Toolkit Factory
DB-Zugriff Interface
DB-Zugriff
Oracle
DB-Zugriff
MySQL
DB-Zugriff
MS-SQL
DB-Zugriff
DB2
Oracle DB
MySQL DB
MS-SQL DB
DB2
DB

Zusammenfassung

- Diese einzigartige Lösung ermöglicht:
  - Eine dramatische Reduktion der Kosten und Projektlaufzeiten
  - Die standardisierte hohe Qualität der Software.
  - Durch den standardisierten Code ist ein einfacher Austausch zwischen Entwicklern möglich (der Code ist einfach zu verstehen und zu verfolgen).
  - Die Softwarewartung und -weiterentwicklung ist vereinfacht.
  - Das Ergebnis sind stabile Enterprise Applikationen mit der Qualität von Standardsoftware.
  - Die implementierten Systeme sind maßgeschneidert, um die speziellen Anforderungen des Kunden zu erfüllen.
  - Die fachlichen Anforderungen können in kurzen Zyklen basierend auf lauffähigen Applikationen mit dem Kunden verifiziert werden. Dadurch vervollständigt sich die Applikation zum Endprodukt.

Auszug Referenzen

- **Fraunhoferinstitut:** diverse interne und externe Projekte
- **Soltrx (Tochter der Commerzbank AG):**
  Ticket Management für den Wertpapier-Handel
  CWP (companyworld payment)
- **Kaiser's Tengelmann AG:**
  Warenwirtschaft Stammdaten Migrationsystem
  Konfigurationsmanagement
  Projektmanagement
  Personaleinsatzplanung für über 700 Filialen
- **Wüstenrot & Württembergische AG**: Beraterarbeitsplatz
- **FORMAXX AG**: CRM System für Versicherungsmakler
- **QS Qualitätssicherung von Lebensmittel GmbH**: Software Plattform für die gesamte Prozesssteuerung
- **Großer Einzelhändler**: Gesamte Warenwirtschaft Plattform
- **INVERTO AG:** Ausschreibungsplattform, Vertragsmanagement, Rohstoffrechner, Auktionsplattform
- **dbde Deutsche Bildung AG:** Softwareplattform für die gesamte Prozesssteuerung (Fondmanagement)
- **GEVA GmbH:** Internet Frontend für internationaler Zahlungsverkehr (SEPA)
- **ESCADA AG:** Vertragsmanagement
- **agentes AG:** Zeiterfassungssystem, Versicherungsmakler Plattform
- **SinnLeffers GmbH:** Personalbedarfsplanung
- **bäurer GmbH:** Entwicklung der ERP-Standardlösung
- **Nöll+ Partner Architekten:** Facilitymanagementsystem
- **Romservice Telecommunication:** ERP-System

Software Process Engine 3.1

Herzlichen Dank für Ihre Aufmerksamkeit!

Software Process Engine: Motivation (I)

- Es ist gängige Praxis, für neue Programme ein vorhandenes, ähnliches Programm zu kopieren und manuell anzupassen
- Wird nachträglich ein Fehler im „Vorlage-Programm" gefunden, muss die Korrektur manuell auch in allen daraus entstandenen Programmen durchgeführt werden. Was oft nicht mehr oder nur schwer nachvollziehbar ist.
- Für moderne 3-Schicht-Anwendungen muss viel Code geschrieben werden. Davon ist zu 70-90% Standard-Code ohne Geschäftsprozess-Intelligenz. Dieser Code muss aber geschrieben werden, sonst funktioniert nichts.

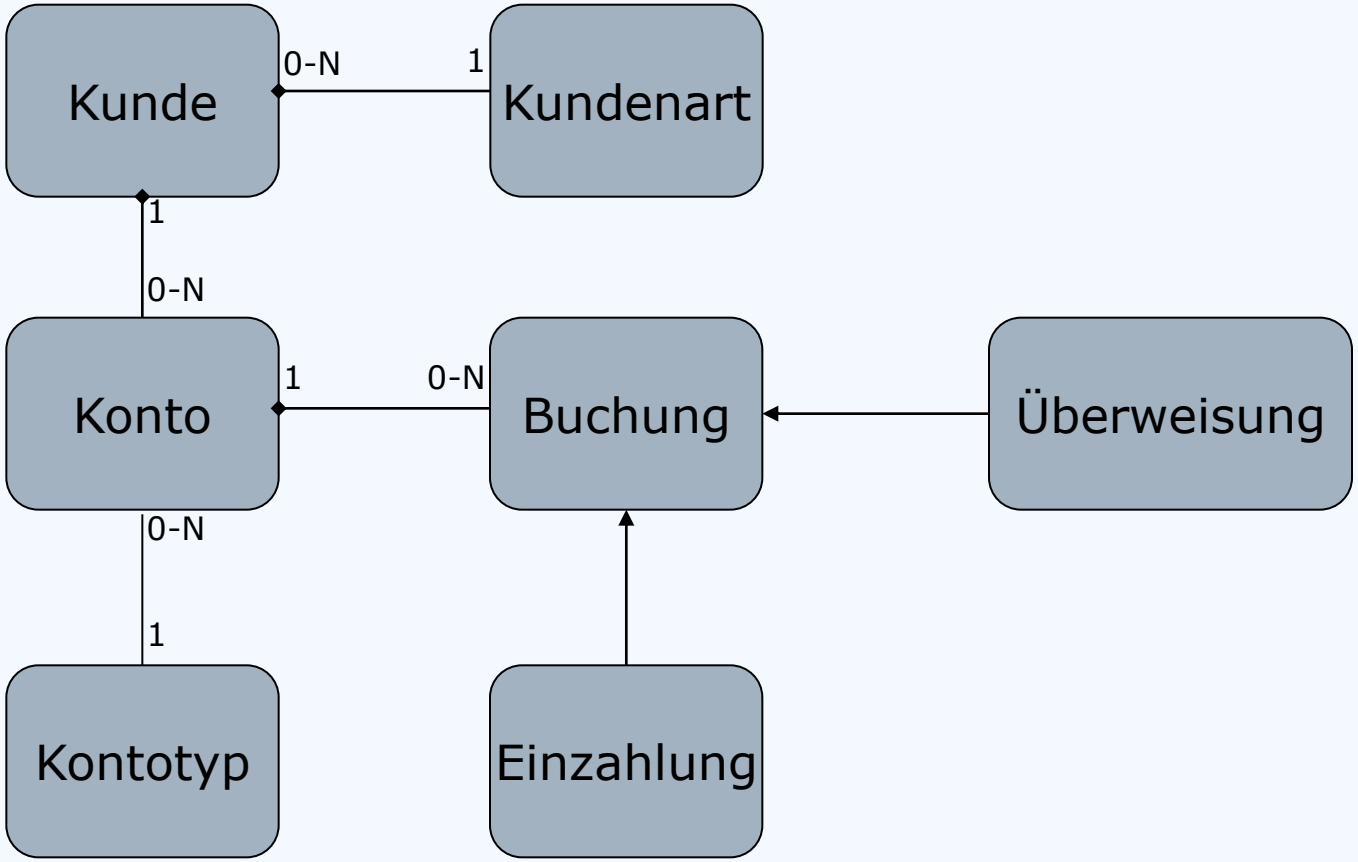
...

Software Process Engine: Motivation (II)

- Muss die fertige Anwendung durch ein neues Feld ergänzt werden, muss dies manuell und konsistent zueinander geschehen
  - In der Detail-Maske und der tabellarischen Anzeige
  - In den HTML / JSP-Formularen bei Web-Anwendungen
  - In der Kommunikations-Schicht Frontend<>Application-Server
  - Im Objekt-Model auf den Application-Server
  - In der Datenbank-Zugriff-Schicht
  - In der Datenbank selbst (SQL)
  - etc.

  Viel Aufwand und sehr fehleranfällig!

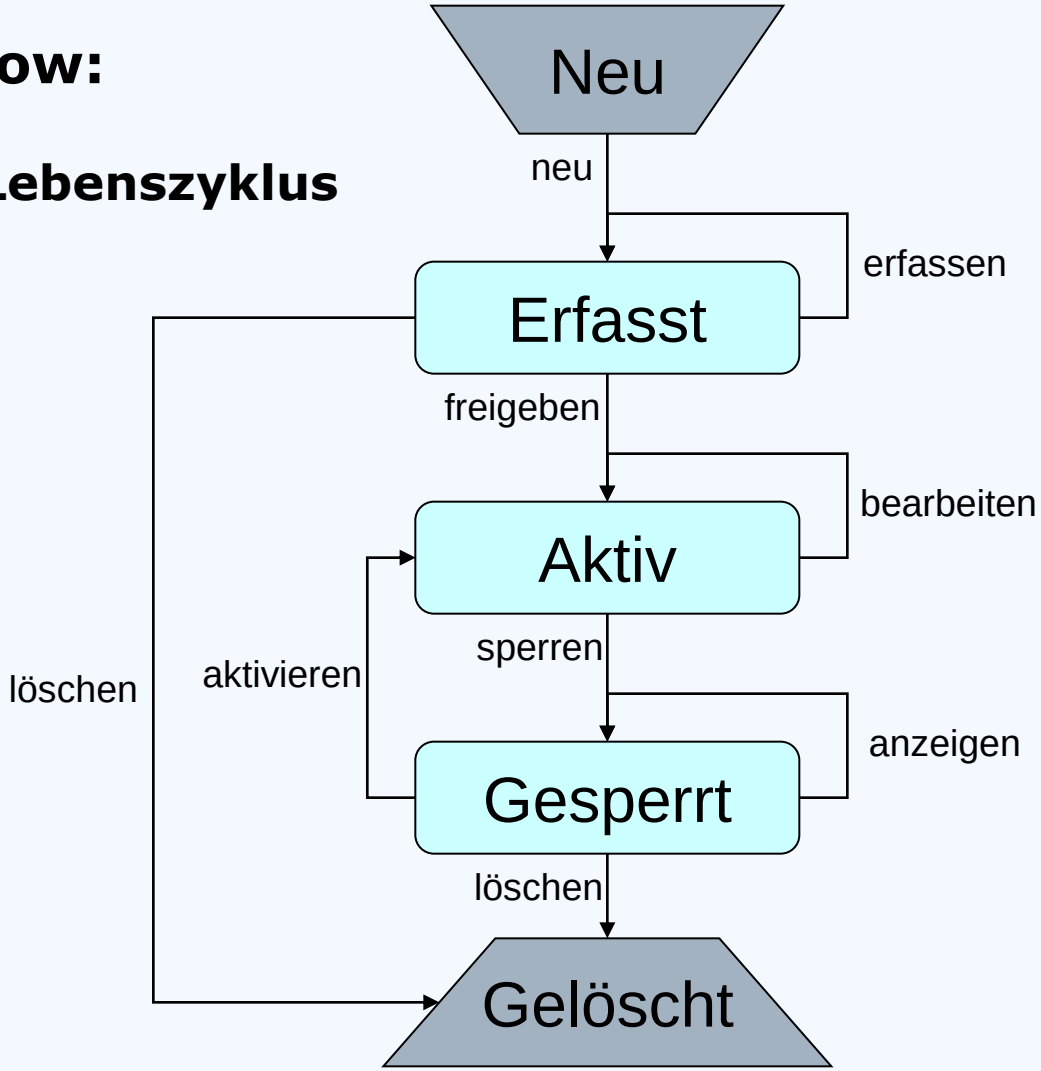
Beispiel-Anwendungen (I)



Beispiel-Anwendungen (II)

Workflow:

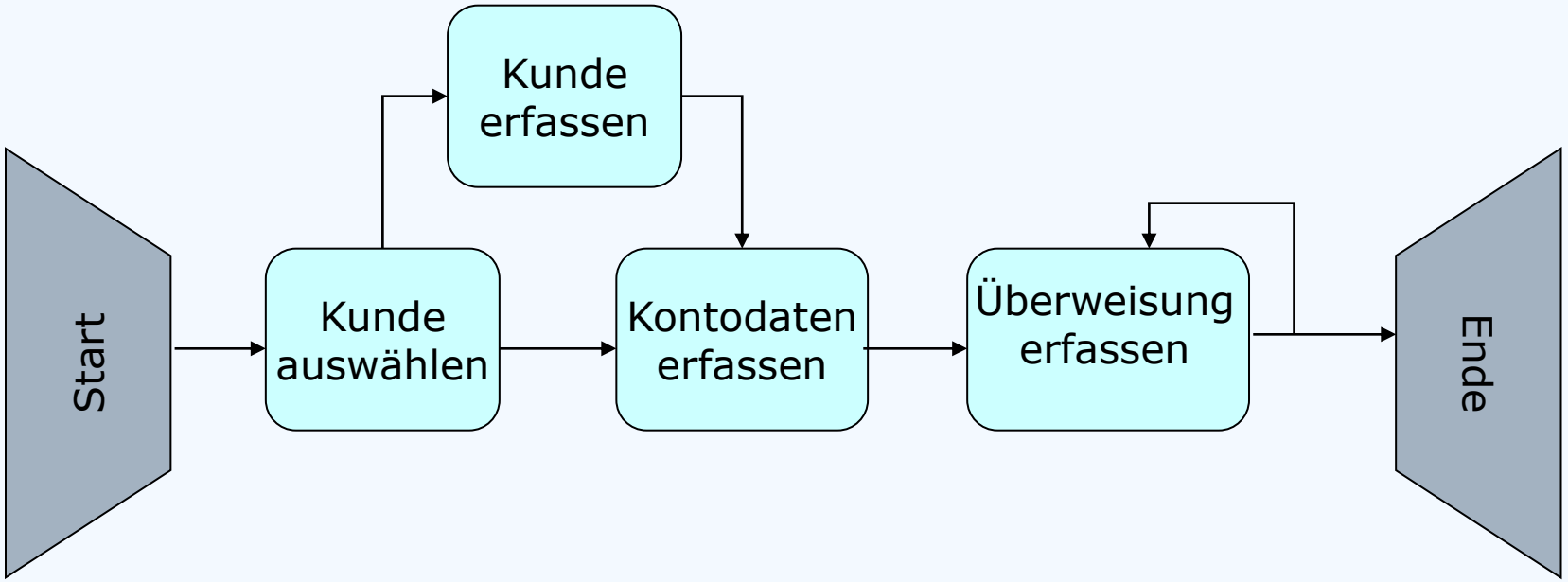
Konto-Lebenszyklus



Beispiel-Anwendungen (III)

Workflow-Dialog:

Konto-Erfassung



Beispiel-Anwendungen (IV)

Workflow-Kollaboration:

Kontotyp bearbeiten