Softwarekartographie

Methoden und Modelle zur Beschreibung, Bewertung und Gestaltung von Anwendungslandschaften

Sabine Buckl
Alexander M. Ernst

Software Engineering betrieblicher Informationssysteme (sebis)
Ernst Denert-Stiftungslehrstuhl

wwwmatthes.in.tum.de

Agenda und Lernziele der Einheit

Agenda

- Motivation und thematische Einordnung
- Softwarekartographie
- UML für Softwarekarten?
- Anwendung von Softwarekarten
- EAM Pattern Catalog
- Fazit

Lernziele der Einheit

- Grundlagen der Softwarekartographie und von Softwarekarten als Architekturdokumentation kennen
- Modellierungssprache für Softwarekarten grundlegend kennen
- Verbindung zwischen Softwarekartographie und Enterprise Architecture Management verstehen
- Strukturierungsprinzipien für Anwendungslandschaften kennen
- EA Management Pattern Catalog kennen und auf Problemstellungen des Enterprise Architecture Managements anwenden können

Literatur der Einheit

[Al77] Alexander, C.; Ishikawa, S.; Silverstein, M.: *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press, USA, 1977.

[Bu08a] Buckl, S.; Ernst, A.; Lankes, J.; Matthes, F.: *Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008)* . Technical Report TB 0801, Chair for Informatics 19, Technische Universität München, 2008.

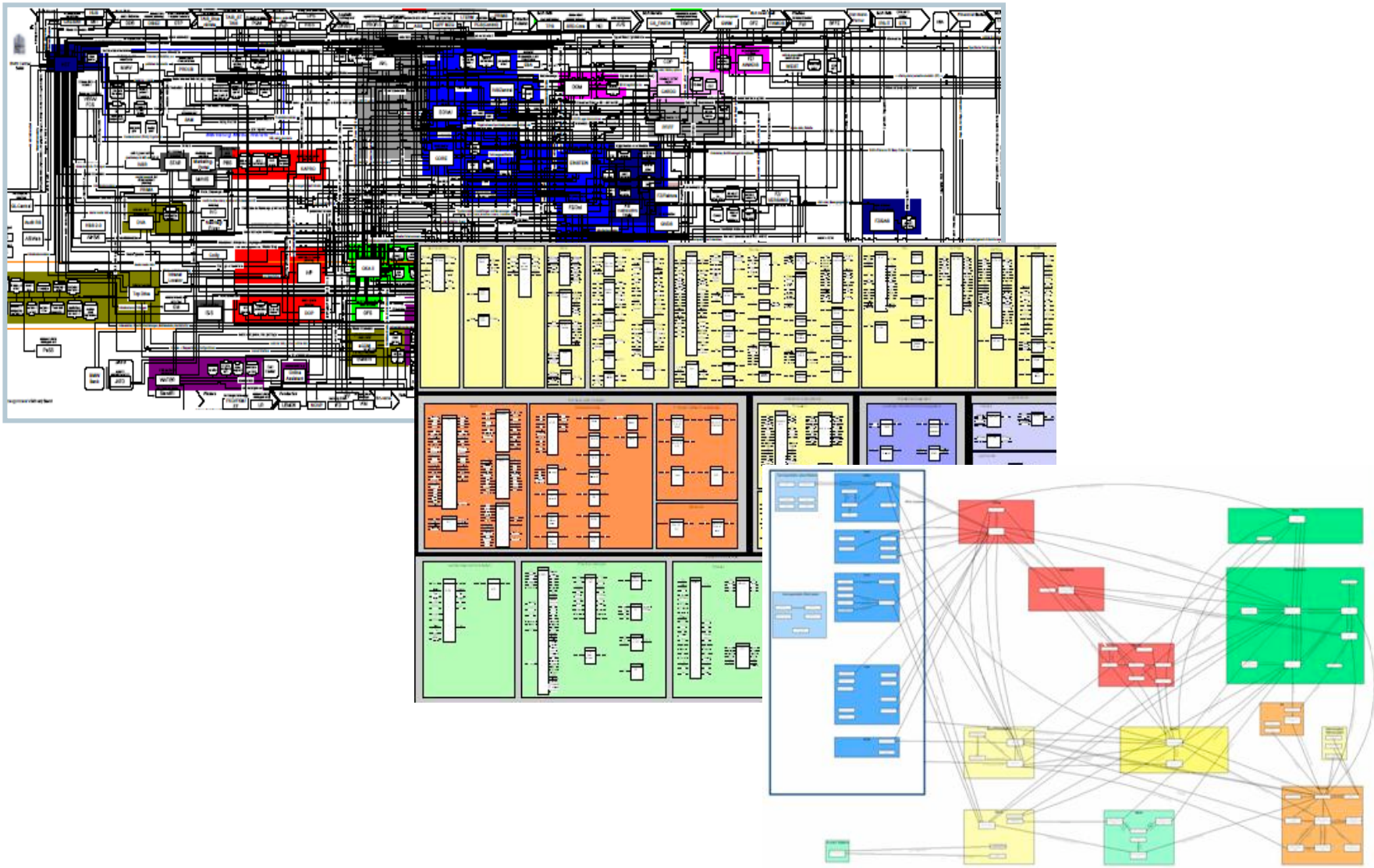
[Bu08b] Buckl, S.; Leitel, J.; Matthes, F.; Schweda, C.: *Enterprise Architecture Management Tool Survey 2005*. Technische Universität München, Chair for Informatics 19 (sebis), 2005.

[Er06b] Ernst, A.; Lankes, J.; Schweda, C.; Wittenburg, *A.: Using Model Transformation for Generating Visualizations from Repository Contents - An Application to Software Cartography.* Technische Universität München, Institut für Informatik, Lehrstuhl für Informatik 19, Technischer Bericht TB0601, 2006.

[Ga94] Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. M.: *Design Patterns: Elements of Reusable Object-Oriented Software (Addison-Wesley Professional Computing Series)*. Addison-Wesley Professional, 1994.

[Wi07] Wittenburg, A.: *Softwarekartographie: Modelle und Methoden zur systematischen Visualisierung von Anwendungslandschaften.* Dissertation, Fakultät für Informatik, Technische Universität München, 2007.

Eine Anwendungslandschaft besteht heute aus 10^2 bis 10^3 vernetzten Anwendungen



Anwendungslandschaft

sebis

Charakteristika

- vernetztes System von teilautonomen Systemen
- lebendig, wachsend und zeitlich unbeschränkt
- Menschen sind integraler Bestandteil des Systems

- für Interessengruppen in einer Organisation geschaffen
- von diesen zu finanzieren
- langfristiger Interessenausgleich notwendig

- ganzheitliche und langfristige Perspektive erforderlich (Ist, Plan, Soll)

Herausforderungen

- Unternehmensnutzen vs. Gruppennutzen (➔ Monetarisierung)
- Gemeinsames Vokabular zur Kommunikation (➔ Modellbildung)
- Nutzergerechte Abstraktionen zur Beherrschung der inhärenten Komplexität (➔ Sichten und Karten)

Probleme beim Management von Anwendungslandschaften

sebis

Nebel liegt über der IT

- Geschäft und Management beklagen geringe Kosten- und Nutzentransparenz der IT
- IT-Projekte beginnen mit einer Analyse der Nachbarsysteme und Schnittstellen
- Immer wiederkehrende Befragungen, um Informationen zu sammeln

Desinteresse bei Geschäft und Management

- „IT ist nicht zu verstehen und unnötig komplex“
- Keine Konkretisierung strategischer Geschäftsziele (z.B. „capability maps“)

Verantwortlichkeiten oft unklar

- Keine nachhaltige Dokumentation der **Eigentümer** für Prozesse, Anwendungen, Schnittstellen, Dienste und Domänen
- Keine verbindlichen daraus abgeleiteten **Rechte & Pflichten** für IT und Geschäft

Die hilfreiche Analogie der Stadtentwicklung

Karten sind ein vertrautes und effektives Kommunikationsmittel

Kommunikation zwischen Menschen mit unterschiedlichen Interessen und Erfahrungen

Sponsoren und Projektpartner seit 2002

sebis

Ernst Denert-Stiftung für Software-Engineering

Anwender

B/S/H/

Allianz · AXA · BMW · MINI · Credit Suisse · Deutsche Bank · Deutsche Börse Systems · Deutsche Post · T-Com · Deutsche Telekom Laboratories · Die Bahn DB · EWE · FIDUCIA Ihr IT-Partner · Fraport · HSH NORDBANK · HVB Information Services · UniCredit Group · KRONES · KUEHNE+NAGEL · LVM Versicherungen · Münchener Rück Munich Re Group · Nokia Siemens Networks · O2 · P&G · Postbank · RWE · SEB · SIEMENS · schufa Wir schaffen Vertrauen · World of TUI · WACKER · Antriebs- und Fahrwerktechnik ZF · Zollner

Berater

act! · BTC · Consulting DETECON · FJH · sd&m A Company of Capgemini · cirquent credible consulting · softlab group · SYRACOM The Business and IT Architects

Agenda

- Motivation und thematische Einordnung
- Softwarekartographie
- UML für Softwarekarten?
- Anwendung von Softwarekarten
- Wege zum Enterprise Architecture Management
- Fazit

Softwarekartographie

sebis

- Die Softwarekartographie stellt Modelle und Methoden zur Beschreibung, Bewertung und Gestaltung von Anwendungslandschaften.

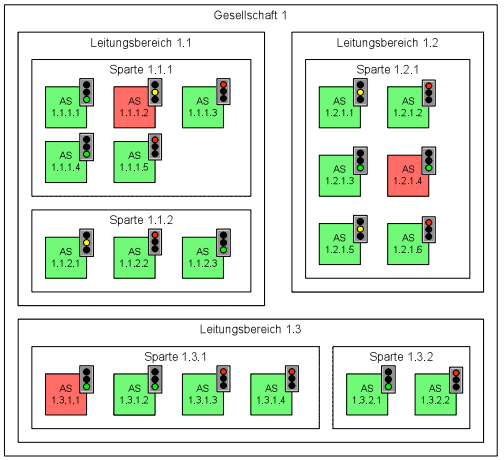
- Modelle der Softwarekartographie
  - Softwarekarten als graphische Modelle von Anwendungslandschaften
  - Automatisches Generieren der Softwarekarten und deren Pflege ermöglichen
  - Fokussierung liegt auf dem *Modellieren*, nicht dem Malen!
  - Visualisierungen im Einklang mit den Interessen der Stakeholder
- Methoden der Softwarekartographie
  - Dokumentation von Anwendungslandschaften mit Softwarekarten
  - Evaluierung von Anwendungslandschaften mit Metriken und die Visualisierung dieser Metriken auf Softwarekarten
  - Gestaltung von Anwendungslandschaft mittels Softwarekarten

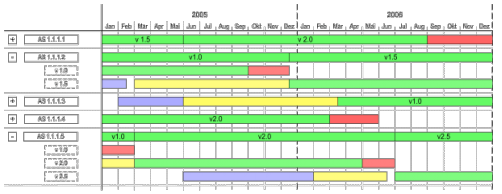
sebis

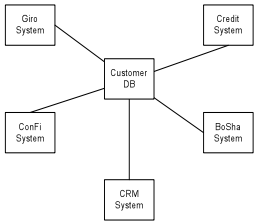
Ein paar echte Beispiele…

Konsolidierung bestehender Softwarekarten aus der Unternehmenspraxis

- Zahlreiche Interviews mit verschiedensten Interessenvertretern
- Aufwändig manuell gestaltete Kartendarstellungen







Kartengrund – Ansatz 1: Clusterkarte

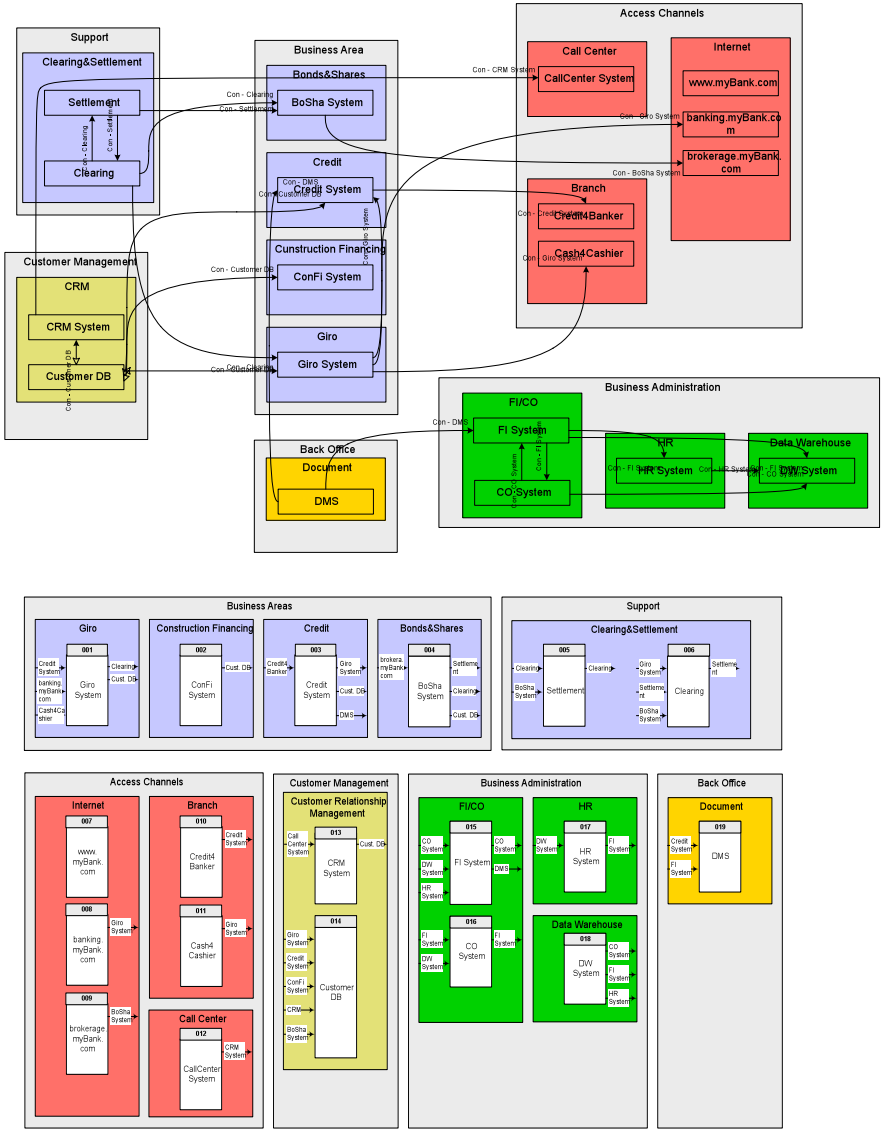
Partitionierung der Karte in logische **Domänen** nach

- Funktionsbereichen bzw. Anwendungsbereichen
- Organisationseinheiten, ...

Anordnung der Einheiten

- platzoptimiert
- nach Zusammengehörigkeit
- nach Unternehmensstandards (Kunde am rechten Rand ➔ Gruppe Zugriffskanäle rechts)

Domänen können **geschachtelt** werden und definieren so auch grobgranulare **Distanzmaße**.

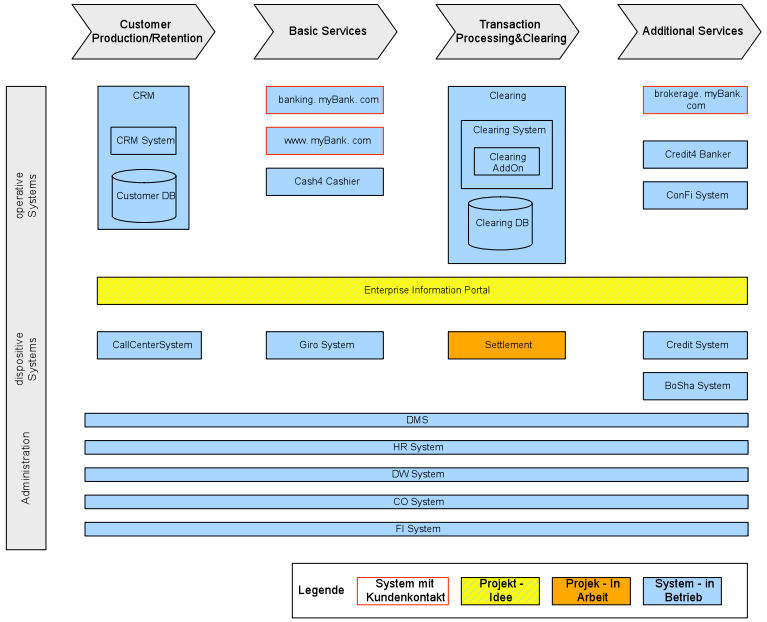


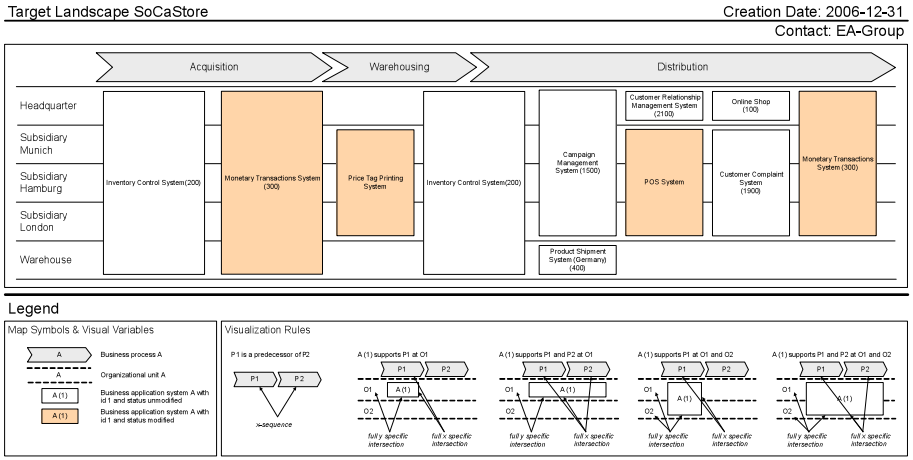
Kartengrund – Ansatz 2: Prozessunterstützungskarte

Verortung der Elemente

- x-Achse für Geschäftsprozesse
  - Ebene 0 bis 3
  - Lineare Prozesse
  - Notation als Wertschöpfungskette
- y-Achse für
  - Organisationseinheiten, Standorte, Zielmärkte, Produkte, …

Nützlich für Benchmarks und Konsolidierungsprojekte.



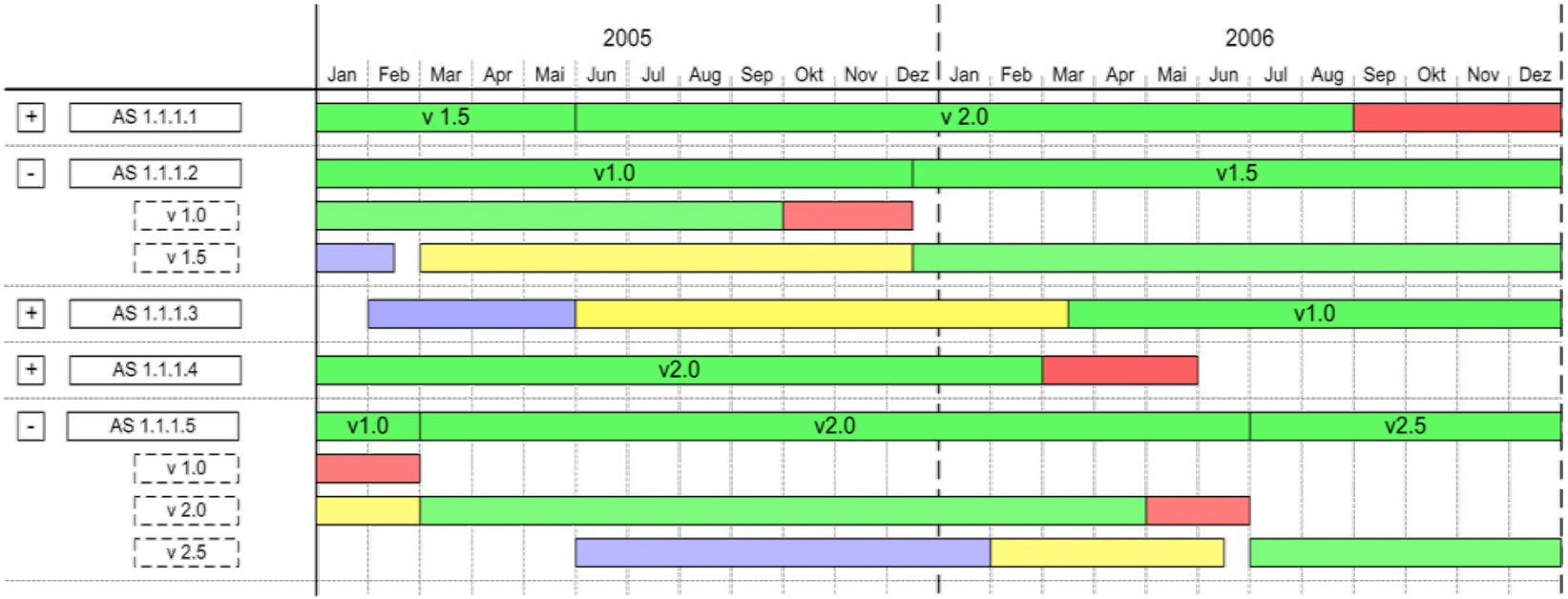


Kartengrund – Ansatz 2: Intervallkarte

Verortung der Elemente entlang der x- und y-Achse

- **X-Achse:** Zeit
- **Y-Achse:** Gruppierung
  - Versionen von Informationssystemen, Projekte, Programme, Organisationseinheiten, …
- **Farbe:** Zustand des Elements

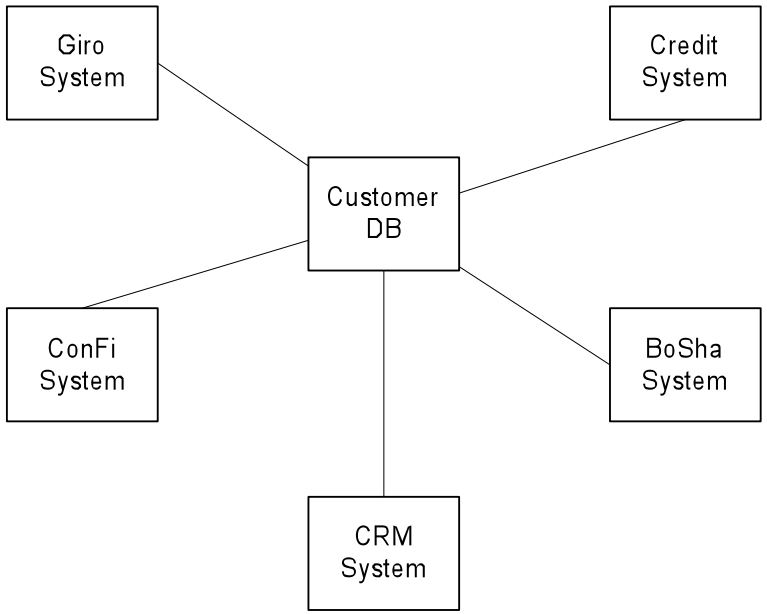
An Gantt-Diagramme angelehnt



Softwarekarten ohne Kartengrund – Ansatz 3

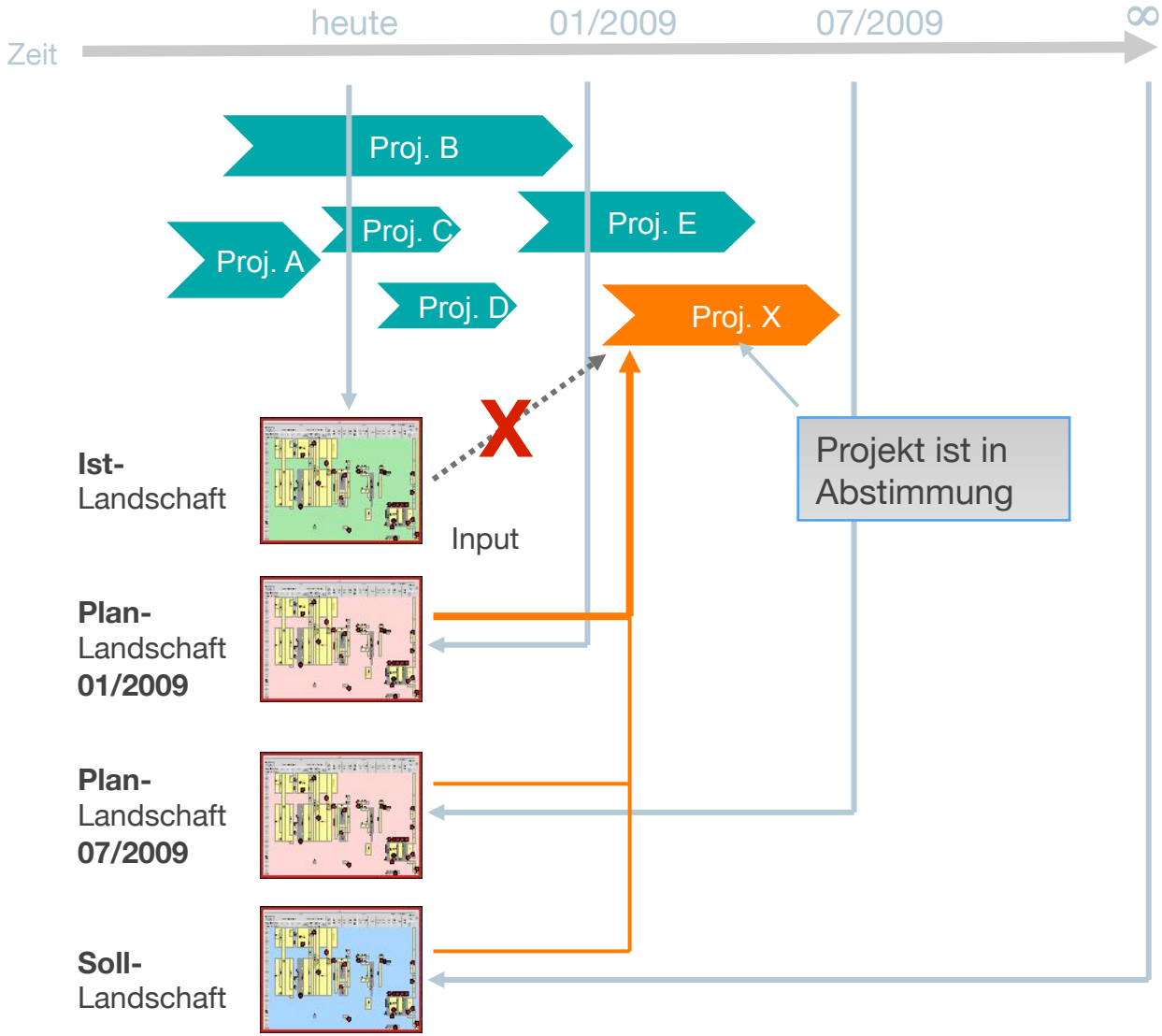
- Zentrales Betrachtungsobjekt in der Mitte positioniert
- Angrenzende Systeme kreisförmig angeordnet (ggf. mehrere Kreise)
- Ad hoc Generierung aus Repository

- Nutzung des gleichen „kartographischen Vokabulars“

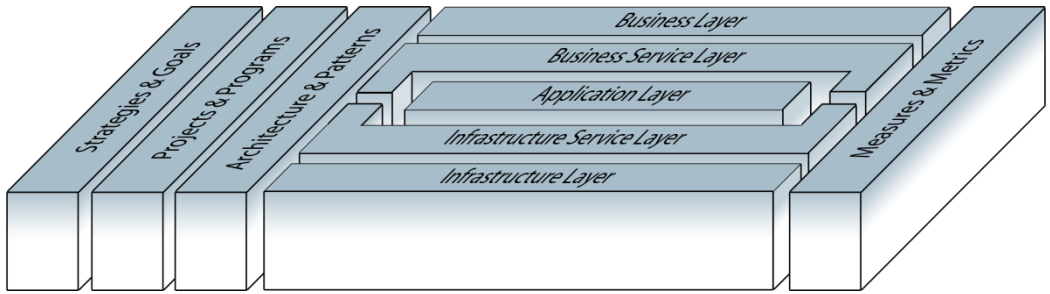


Jede Softwarekarte hat einen Zeitbezug

- Ist-, Plan- oder Soll-Zustand
- Varianten werden erzeugt
- Varianten sind zusammenzuführen
- Nachvollziehbare Historien sind wichtig



Was bedeutet „ganzheitliche Sicht“ der Anwendungslandschaft?

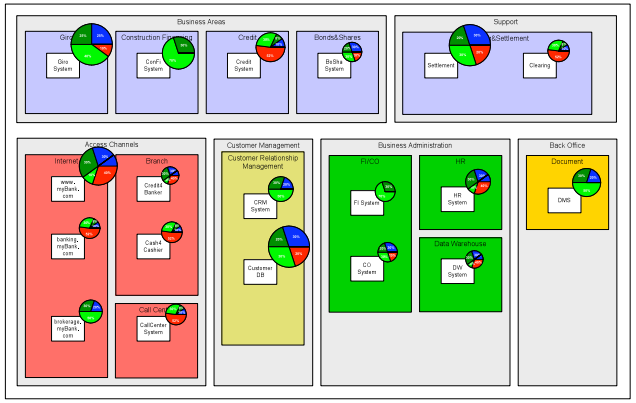


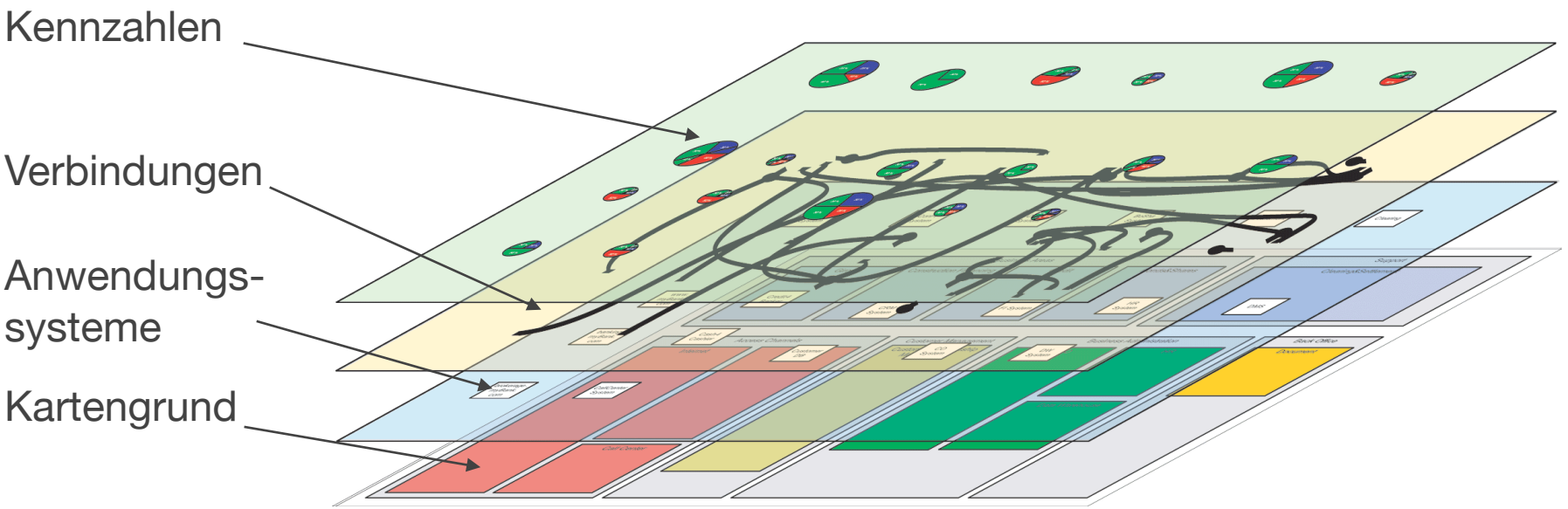
Prinzipien von Softwarekarten an einem Beispiel

sebis

Das Schichtenprinzip

- Problemangemessener Kartengrund
- Regelbasiertes Layout der visuellen Elemente
- Ein-/Ausblenden von Details durch Schichtenbildung



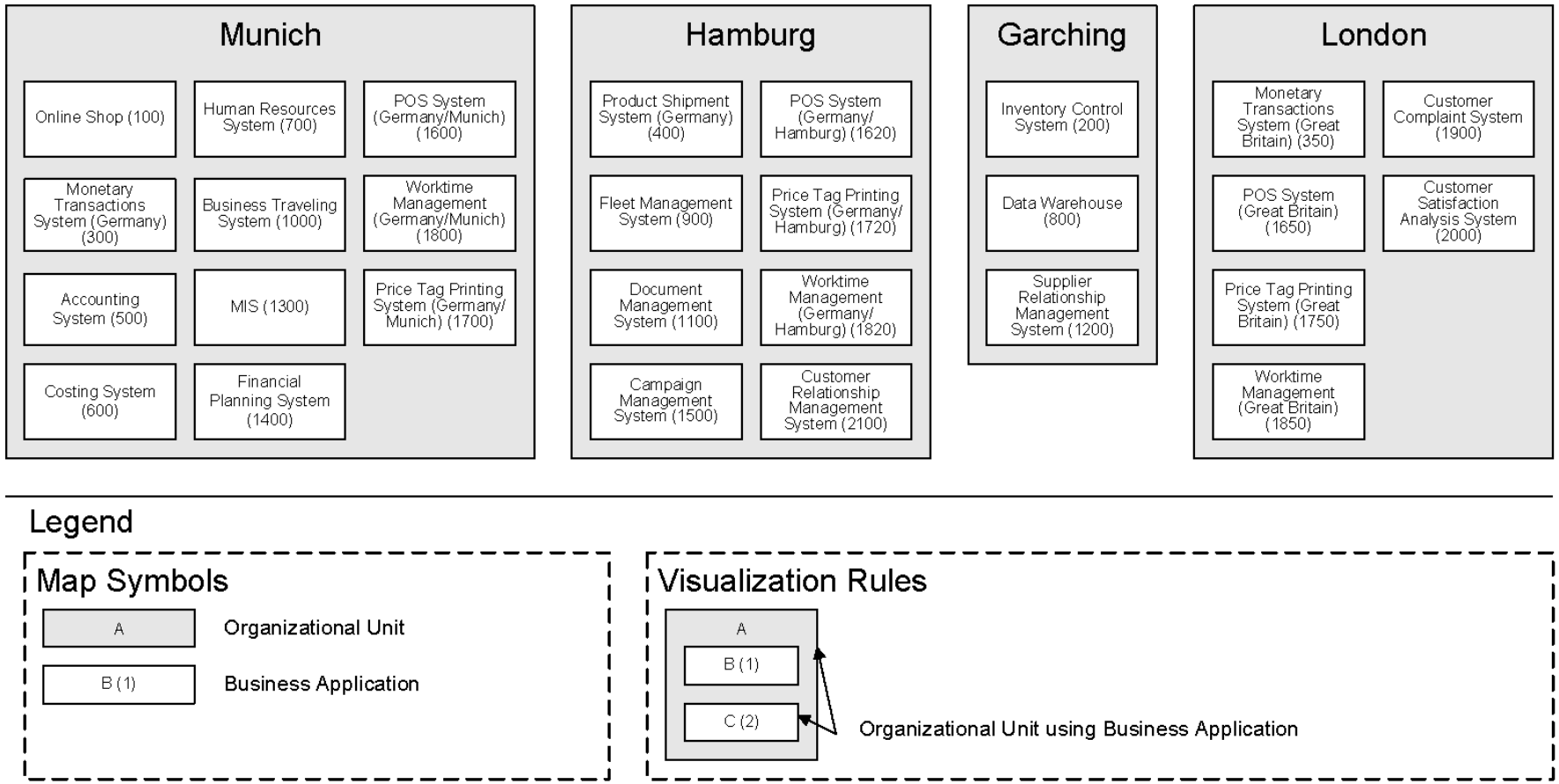


Relevante Informationen für die Softwarekartographie

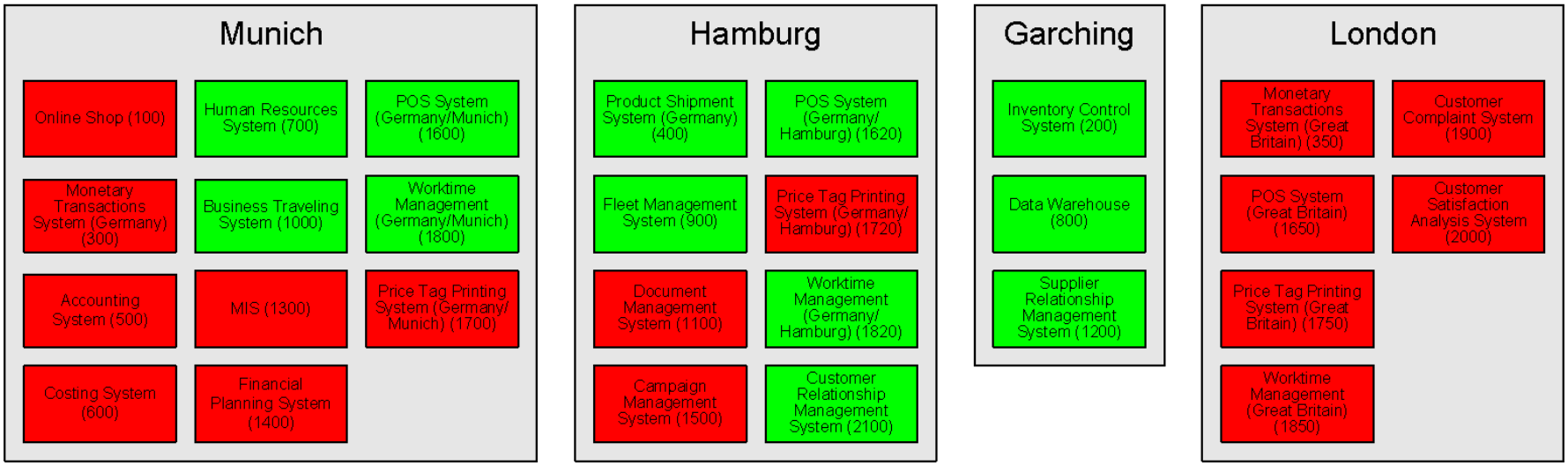
sebis

- Zentrales Untersuchungsobjekt: (betriebliche) Anwendungssysteme und ihre Umgebung
- Klassifikation von relevanten Informationen
  - Funktionale
    - Organisationseinheiten, Geschäftsprozesse, Funktionsbereiche, Geschäftsservices, …
  - Planerische/Strategische
    - Strategien, Ziele, Projekte, Programme, …
    - Lebenszyklen, Versionen, …
  - Wirtschaftliche
    - Betriebskosten, Wartungskosten, Investitionen, …
  - Technische
    - Schnittstellen, Programmiersprachen, Middleware-Systeme, Softwarearchitekturen, …
  - Operative
    - Uptime/Downtime von Systemen, Abhängigkeiten, (Geographische) Betriebsorte, …

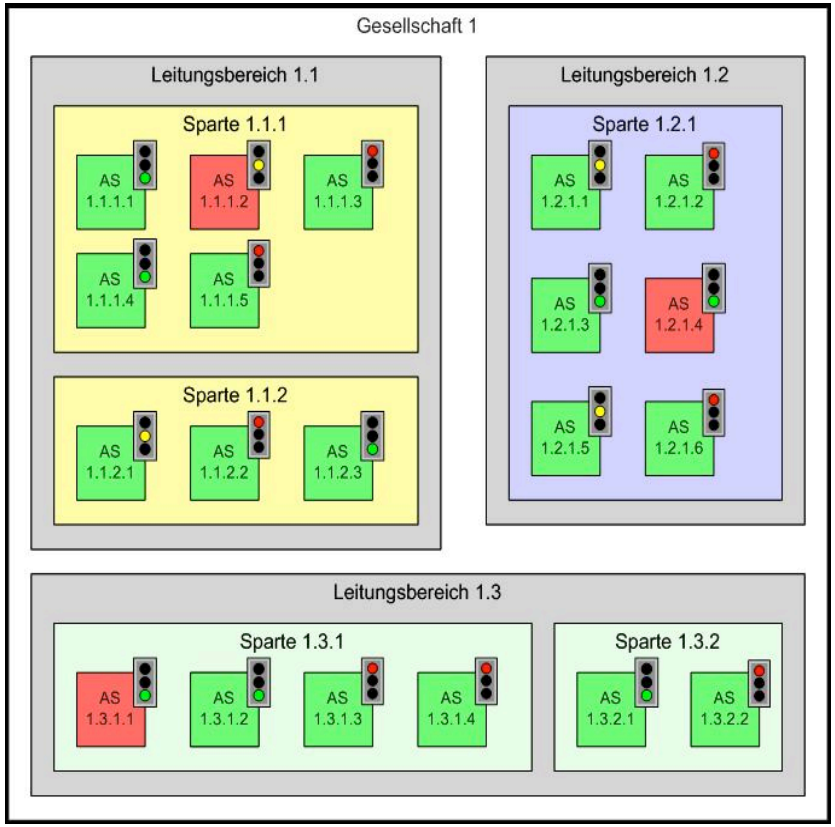
Visualisierung von Informationen auf Softwarekarten (1)

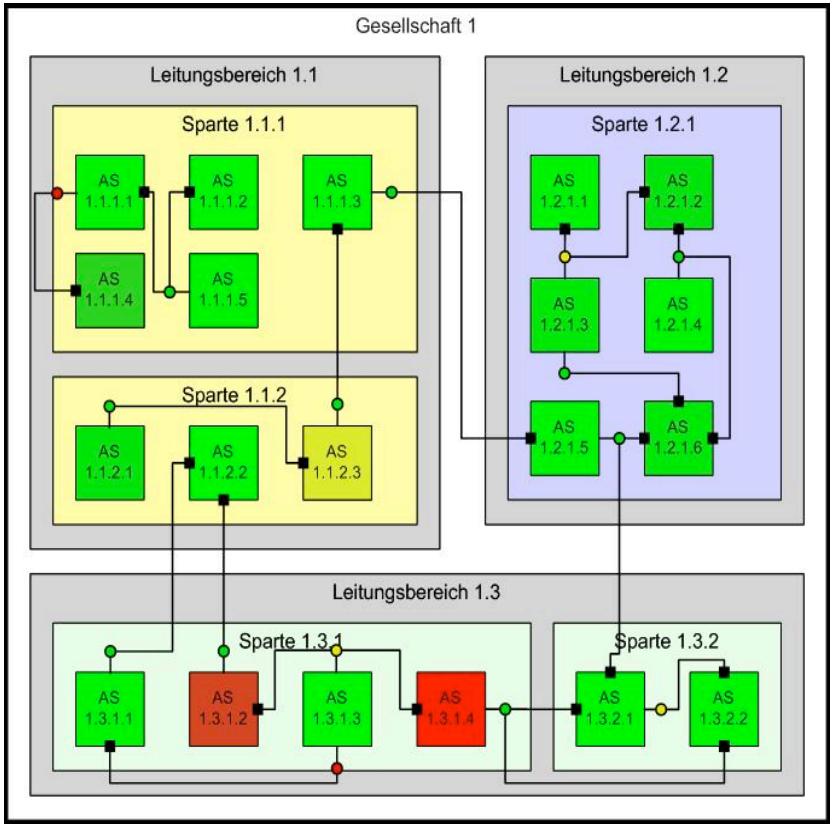


Visualisierung von Informationen auf Softwarekarten (2)



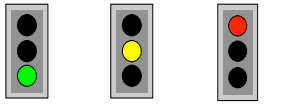
Visualisierung von Informationen auf Softwarekarten (3)





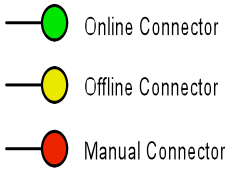
Application does not conform to architectural standards

Application conforms to architectural standards



> 99,5 % // 99,5 – 99,0 % // <99,0 %

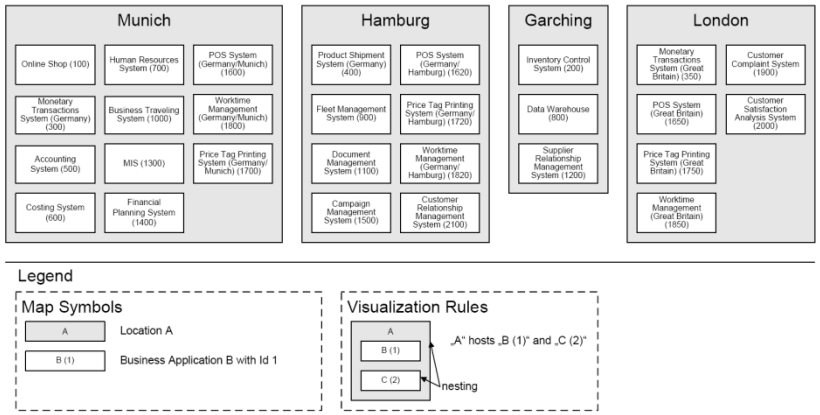
Availability per day



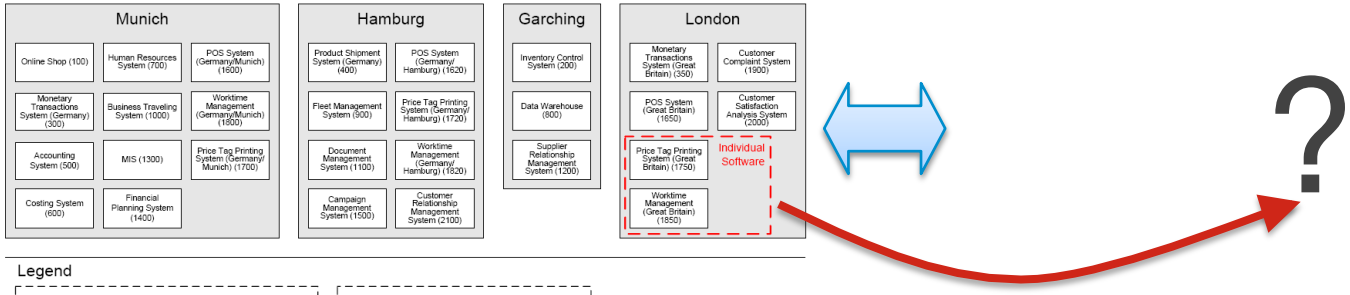
0 50

Response time in seconds

Malen vs. Modellieren: Semantik von Visualisierungen

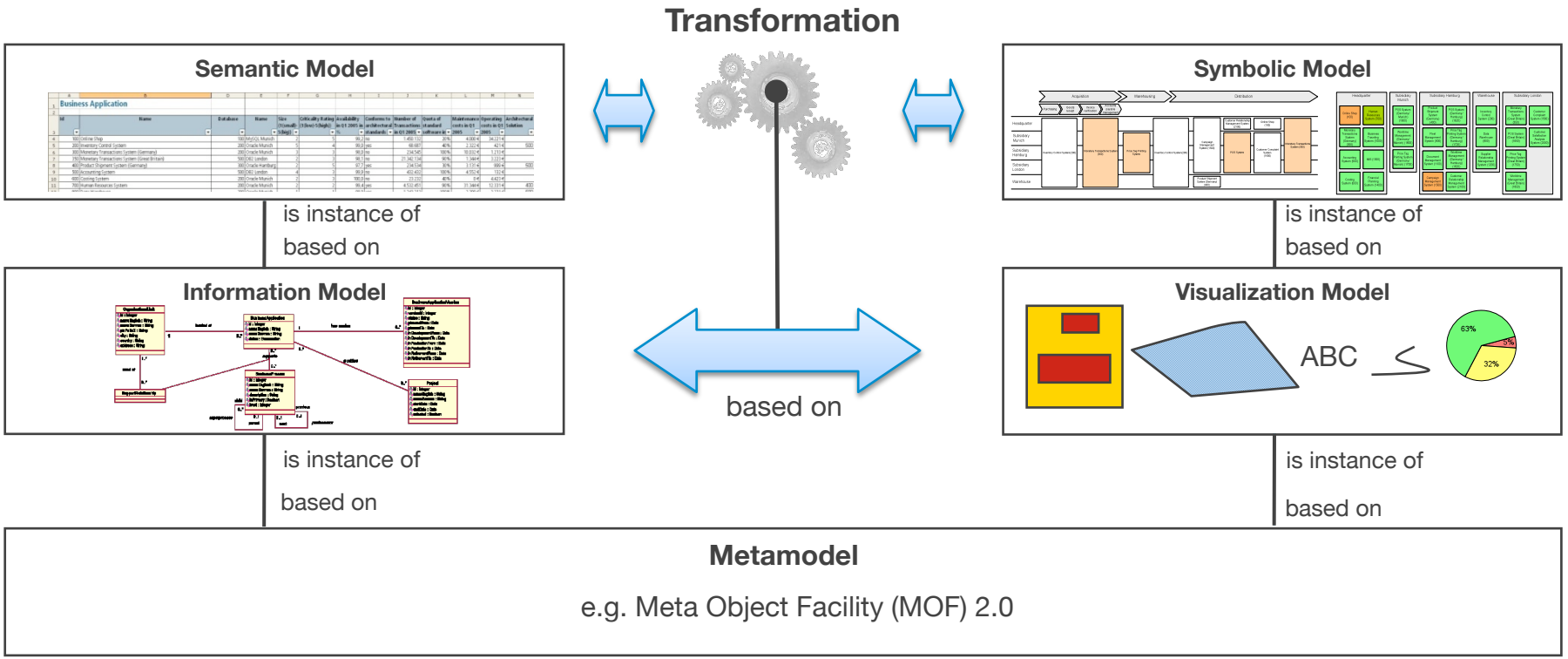


Nutzung der *Malfunktionalität* eines Werkzeugs um ein Rechteck hinzuzufügen



➔ „Malen ist kein Management"

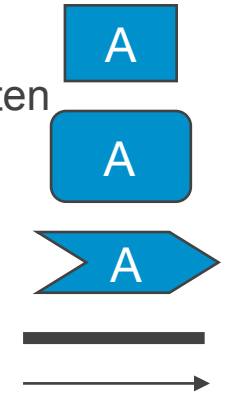
Zusammenhang zwischen semantischem und symbolischem Modell



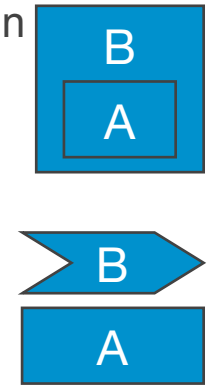
[Er06b]

Visualisierungselemente von Softwarekarten

- Gestaltungsmittel (engl. *Map Symbols*)
  - In der Kartographie sind Gestaltungsmittel die Basiselemente von Karten
  - Softwarekarten unterscheiden
    - Flächartige Gestaltungsmittel (engl. *Planar Map Symbols*)
      - » Rechtecke, Ellipsen, Polygone, Chevrons
    - Lineare Gestaltungsmittel (engl. *Linear Map Symbols*)
      - » Linien, Pfeile, Polylinien, …
  - Jedes Gestaltungsmittel besitzt eine Menge von Gestaltungsvariablen
    - Beispiel Rechteck: Mittelpunkt, Höhe, Breite, Füllfarbe, Rahmenfarbe, …



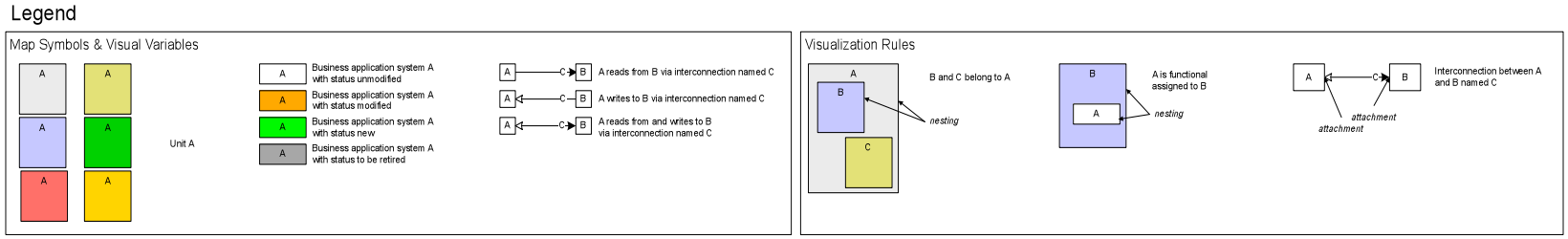
- Gestaltungsregeln (engl. *Visualization Rules*)
  - Positionierung von Elementen auf Softwarekarten wird durch Gestaltungsregeln ausgedrückt
  - Muss-Gestaltungsregeln, müssen für eine korrekte Semantik erfüllt sein
    - Bspw. Verschachteln eines Rechtecks für ein Anwendungssystem innerhalb eines Rechtecks für einen Betriebsstandort
  - Ziel-Gestaltungsregeln, sollen so gut wie möglich erfüllt sein, um eine *gute* und *ästhetische* Softwarekarten zu erhalten
    - Bspw. Minimieren der Fläche eines Rechtecks für einen Cluster, …



Aufbau einer Softwarekarte

Eine *gute* Softwarekarte besitzt neben dem Kartenfeld

- Ein Titelfeld
  - Enthält Titel, Autor, Erstellungsdatum, Kontakt (Person, Rolle oder Gruppe)
  - Softwarekarten visualisieren unterschiedliche Status zu verschiedenen Zeitpunkten
- Eine Legende
  - Erklärt die Gestaltungsmittel, ihre Gestaltungsvariablen und die Gestaltungsregeln
  - Gestaltungsmittel (bspw. Rechteck) werden unterschiedlich benutzt
  - Legende bringt Klarheit über die verwendete Semantik

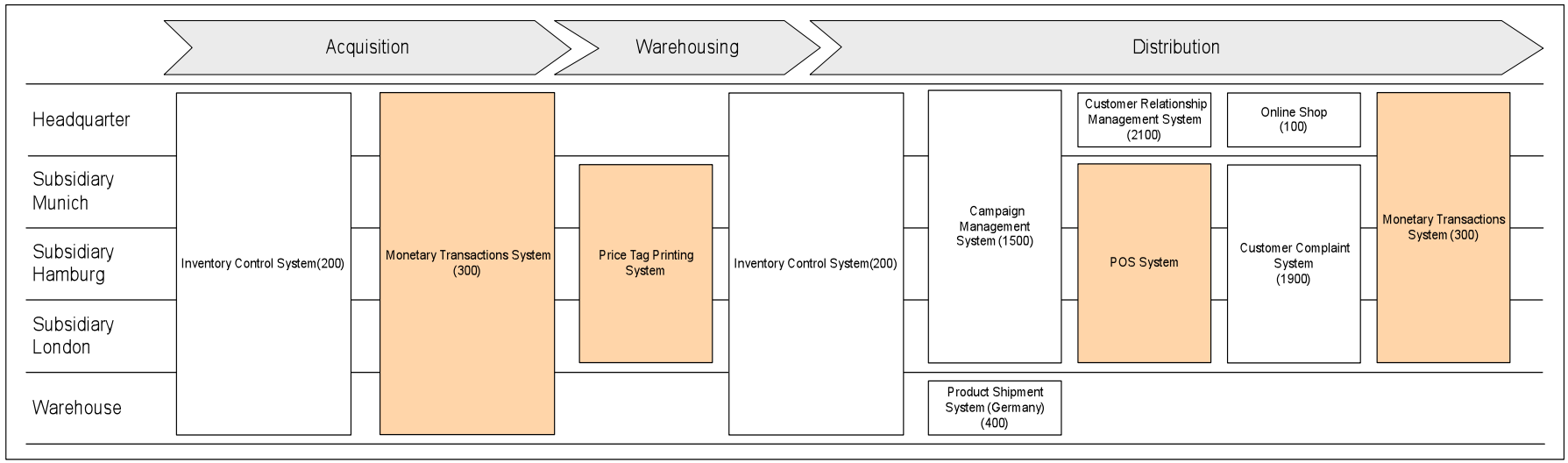


Eine *gute* Softwarekarte

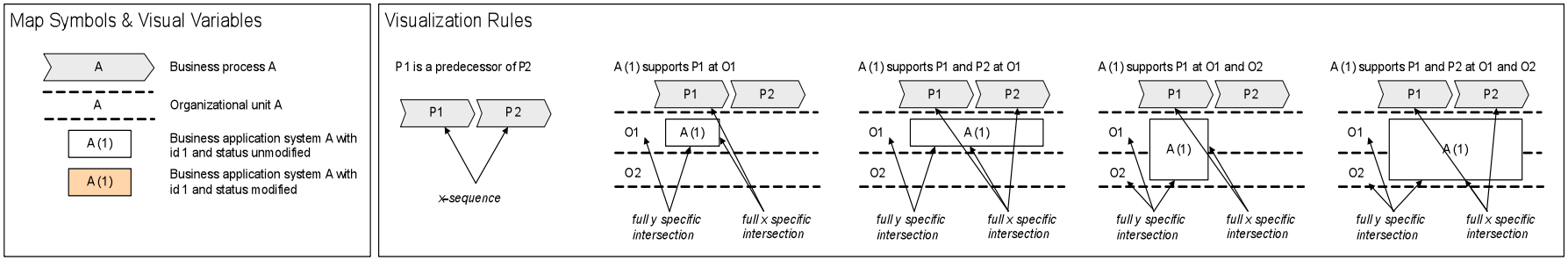
Target Landscape SoCaStore

Creation Date: 2006-12-31

Contact: EA-Group



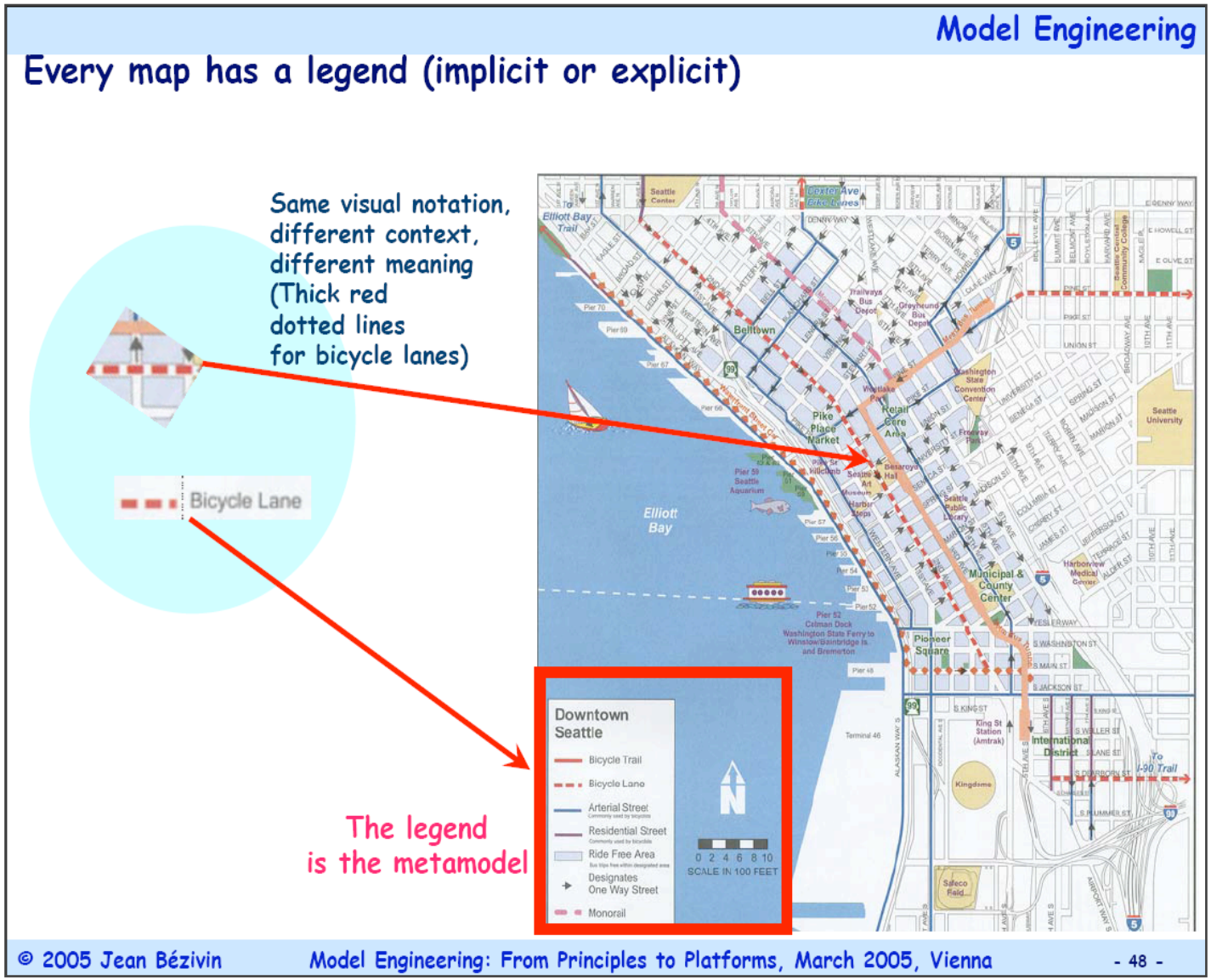
Legend



Agenda

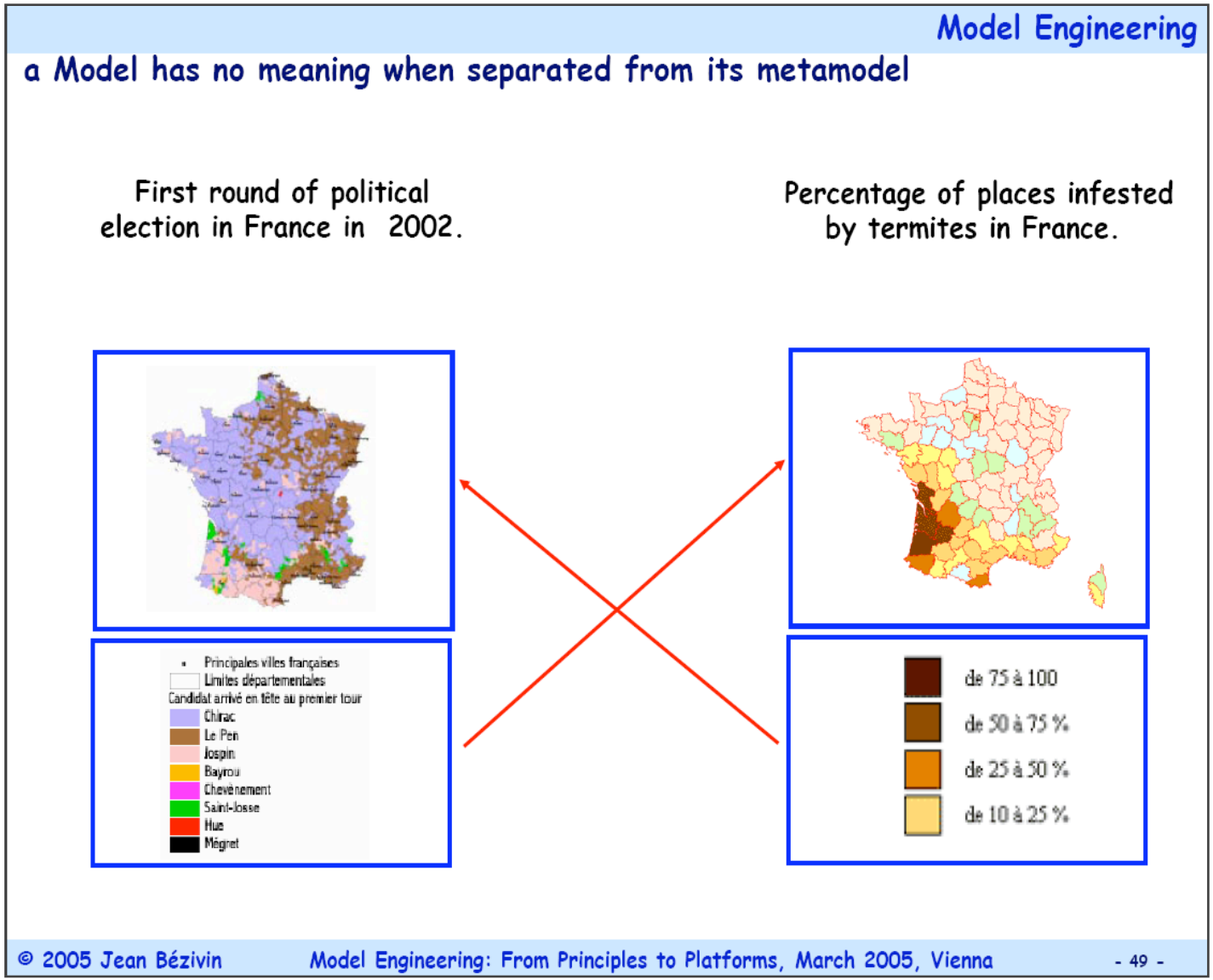
- Motivation und thematische Einordnung
- Softwarekartographie
- UML für Softwarekarten?
- Anwendung von Softwarekarten
- Wege zum Enterprise Architecture Management
- Fazit

Relevanz des Metamodells (1)



Quelle: Bézivin, J.: Model Engineering: From Principles to Platforms, TU Wien, Vortrag, März, 2005

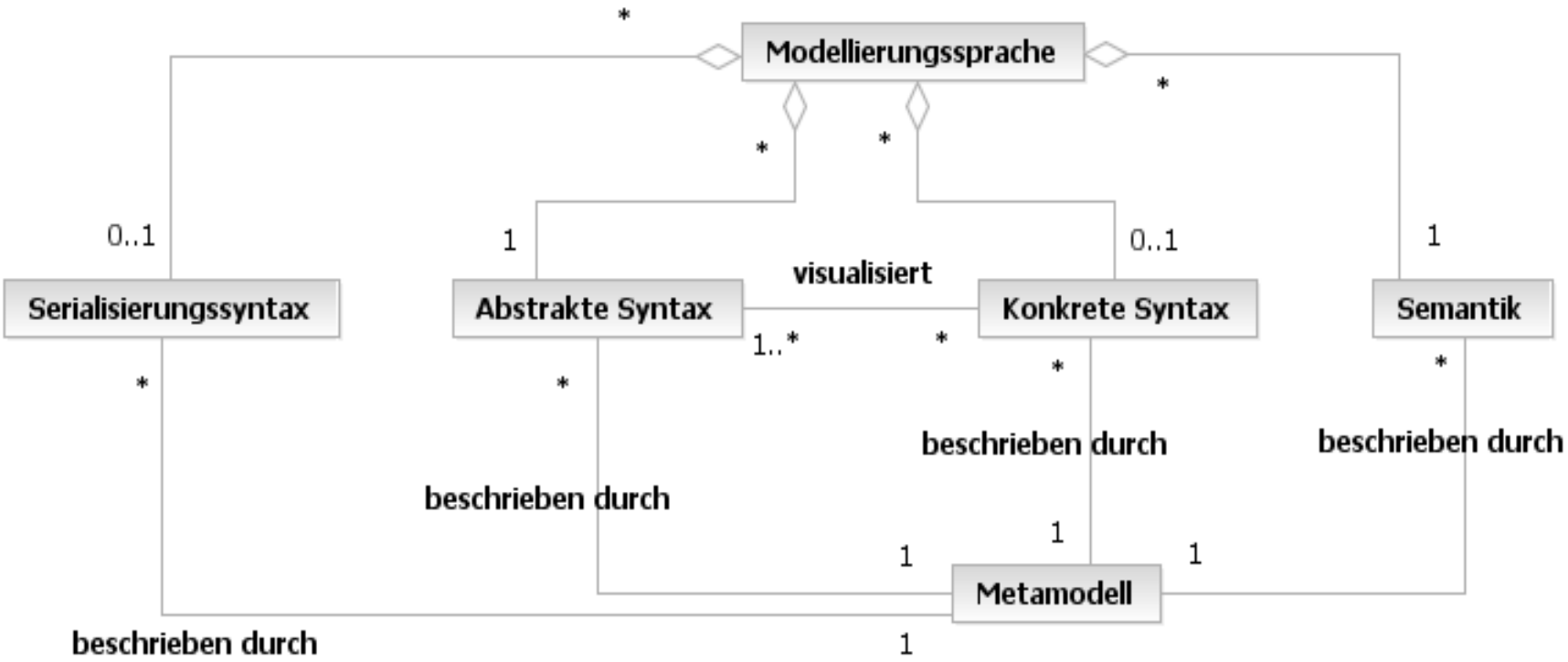
Relevanz des Metamodells (2)



Quelle: Bézivin, J.: Model Engineering: From Principles to Platforms, TU Wien, Vortrag, März, 2005

Metamodell und Modellierungssprache

- Ein Metamodell ist ein Modell *über* ein Modell
- Ein Metamodell beschreibt die Syntax und Semantik einer Modellierungssprache
  - (Hinweis: Die Semantik wird nicht immer zum Metamodell dazugezählt)



Softwarekarten und UML (1)

- Unified Modeling Language (UML) 2.0 beinhaltet
  - Erweiterungsmechanismen wie Stereotypen und UML-Profiles
    - Können zur Modellierung von Anwendungslandschaften genutzt werden
    - Bspw.: Heberling, M.; Maier, C.; Tensi, T.: *Visual Modelling and Managing the Software Architecture Landscape in a large Enterprise by an Extension of the UML*. In: OOPSLA, Workshop DSVL, Seattle, 2002.
- Konzepte mit *geringer* oder *unzureichender* Semantik in UML
  - Farbe für Symbole bzw. Gestaltungsmittel
    - Es existieren ein paar Ansätze zur Modellierung mit Farben außerhalb der OMG
  - Größe von Symbolen bzw. Gestaltungsmitteln
    - Eine Klasse *wächst* mit der Anzahl der Attribute und Methoden (vertikal) und der Länge der Namen dieser (horizontal)
  - Positionierung von Symbolen ist nicht *wohldefiniert*
    - Strukturdiagramme nutzen Verschachtelung für Klassen in Paketen,
    - Komponentendiagramme nutzen Verschachtelung für Klassen in Komponenten,
    - Verteilungsdiagramme nutzen Verschachtelung für Artefakte in Knoten, …
  - *Softwarekarten nutzen Farben, Größen und Positionierungen wesentlich extensiver und müssen diese ebenso definieren!*

Softwarekarten und UML (2)

sebis

- Anzahl von Klassen in UML
  - UML 1.1: ca. 120
  - UML 1.5: ca. 200
  - UML 2.0: ca. 260

- Anwendung von Klassendiagrammen zur Modellierung von Anwendungslandschaften kann mehrdeutig und somit *gefährlich* sein
  - Was ist eine Assoziationsklasse, die `Support` heißt und an eine Assoziation zwischen einem Anwendungssystem und einem Prozess gehängt wird?
  - Was ist die Sichtbarkeit von Attributen bei eine Klasse?
  - Was ist eine Komponente, die Klassen mit dem Stereotyp `Anwendungssystem` enthält?
  - …

Spracharchitektur von UML 2.0 – Abstrakte und Konkrete Syntax

- OMG beschreibt die abstrakte Syntax informell
  - Nur Teile der abstrakten Syntax werden mittels OCL oder BNF definiert
- OMG besitzt *keine* formale oder semi-formale Definition der konkreten Syntax
  - Konkrete Syntax wird mittels Notationstabellen beschrieben
- Beispiel aus der UML Superstructure 2.0

Node Type	Notation	Reference
Artifact	«artifact» ArtifactName	See "Artifact."
Node	NodeName	See "Node." Has keyword options «device» and «execution environment».
Artifact deployed on Node	«artifact» ArtifactName	See "Deployment."
Node with deployed Artifacts	Node «artifact» ArtifactName	See "Deployment."

Agenda

- Motivation und thematische Einordnung
- Softwarekartographie
- UML für Softwarekarten?
- Anwendung von Softwarekarten
  - Sichten und Betrachtungswinkel bei Softwarekarten
  - Operationen auf Softwarekarten
  - Strukturierungsprinzipien für Anwendungslandschaften
- Wege zum Enterprise Architecture Management
- Fazit

Prozessunterstützungskarte (1) – Verschiedene Sichten auf eine Landschaft

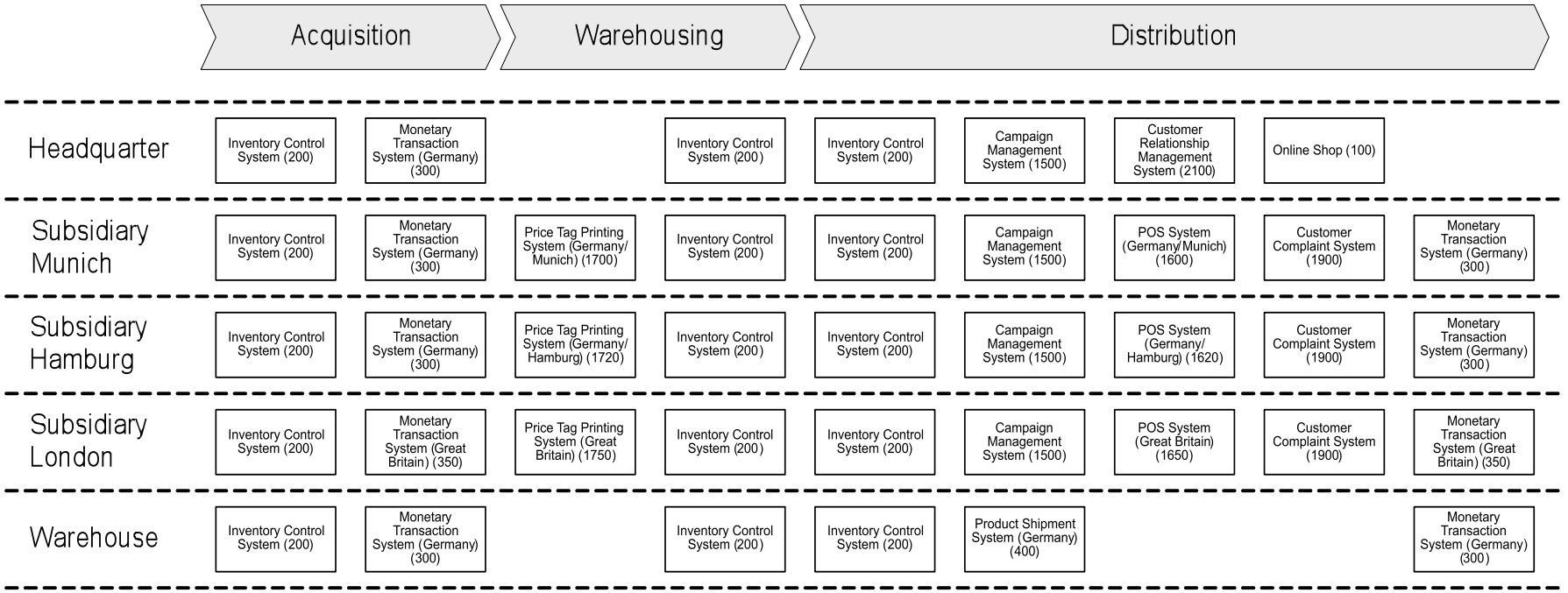
sebis

- Für eine Softwarekarte können vier Betrachtungswinkel (Viewpoints) unterschieden werden
  - Ohne Integration: Kein Symbol wird in x- oder y-Richtung gedehnt
  - Vertikale Integration: Symbole werden (nur) in y-Richtung gedehnt
  - Horizontale Integration: Symbole werden (nur) in x-Richtung gedehnt
  - Horizontale + Vertikale Integration: Symbole werden in x- und y-Richtung gedehnt

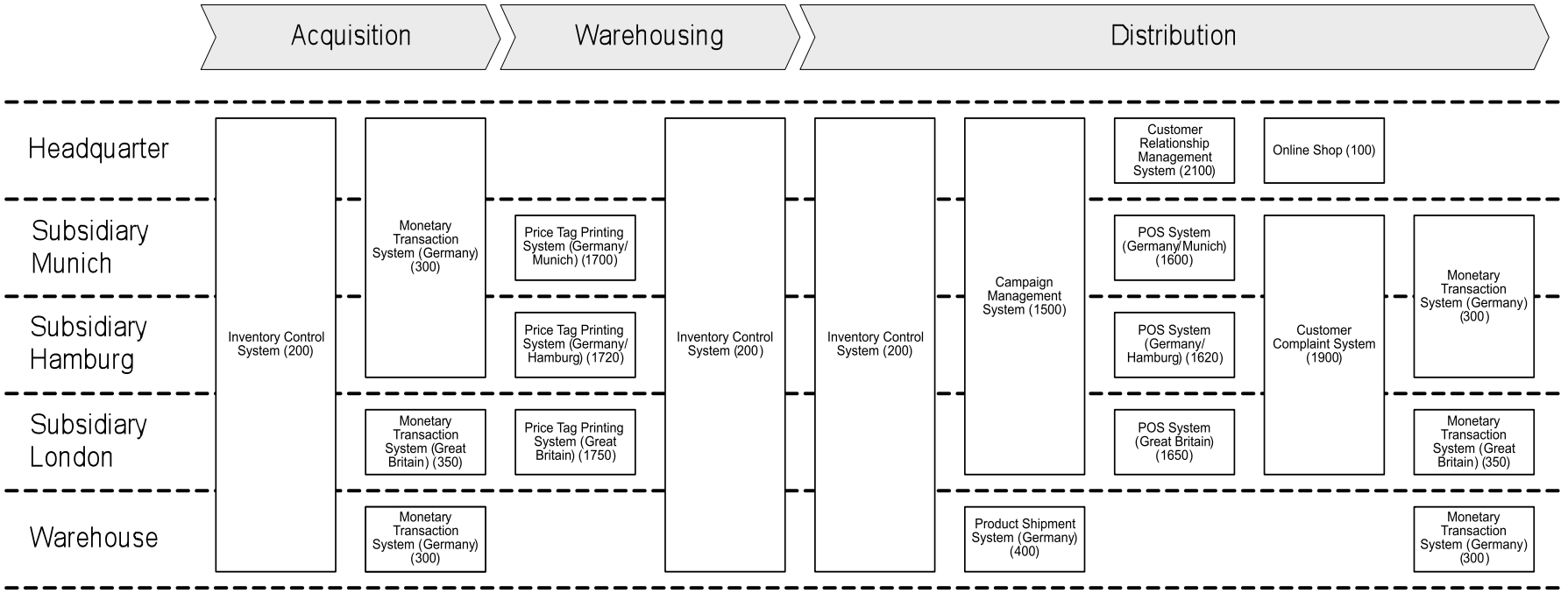
  … die obige Aufzählung unterscheidet nicht hinsichtlich der Visualisierung von zusätzlichen Attributen, wie Standardkonformität, SLA-Erfüllungsgrad etc.

- Probleme die durch die Darstellung der Integration entstehen
  - Wenn ein Anwendungssystem zwei Geschäftsprozesse unterstützt, die nicht direkte Vorgänger oder Nachfolger sind, funktioniert das Dehnen der Symbole nicht!
    ➡ Zwei oder mehr Symbole repräsentieren das gleiche Anwendungssystem
  - Wenn ein Anwendungssystem von mehr als einer Organisationseinheit genutzt, welche nicht als Nachbarn visualisiert werden, funktioniert das Dehnen der Symbole nicht!
    ➡ Zwei oder mehr Symbole repräsentieren das gleiche Anwendungssystem
  - …

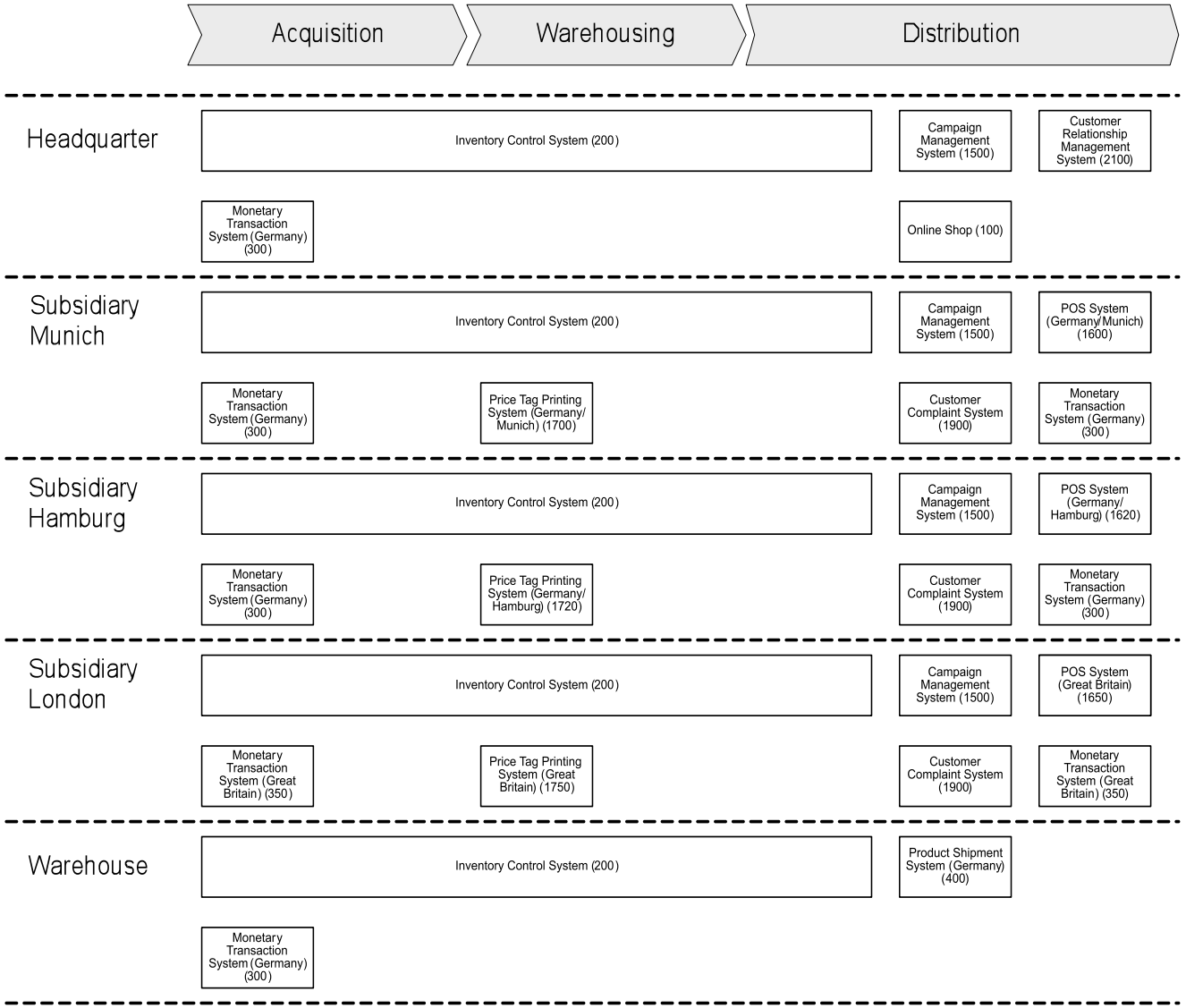
Prozessunterstützungskarte (2) – Ohne Integration



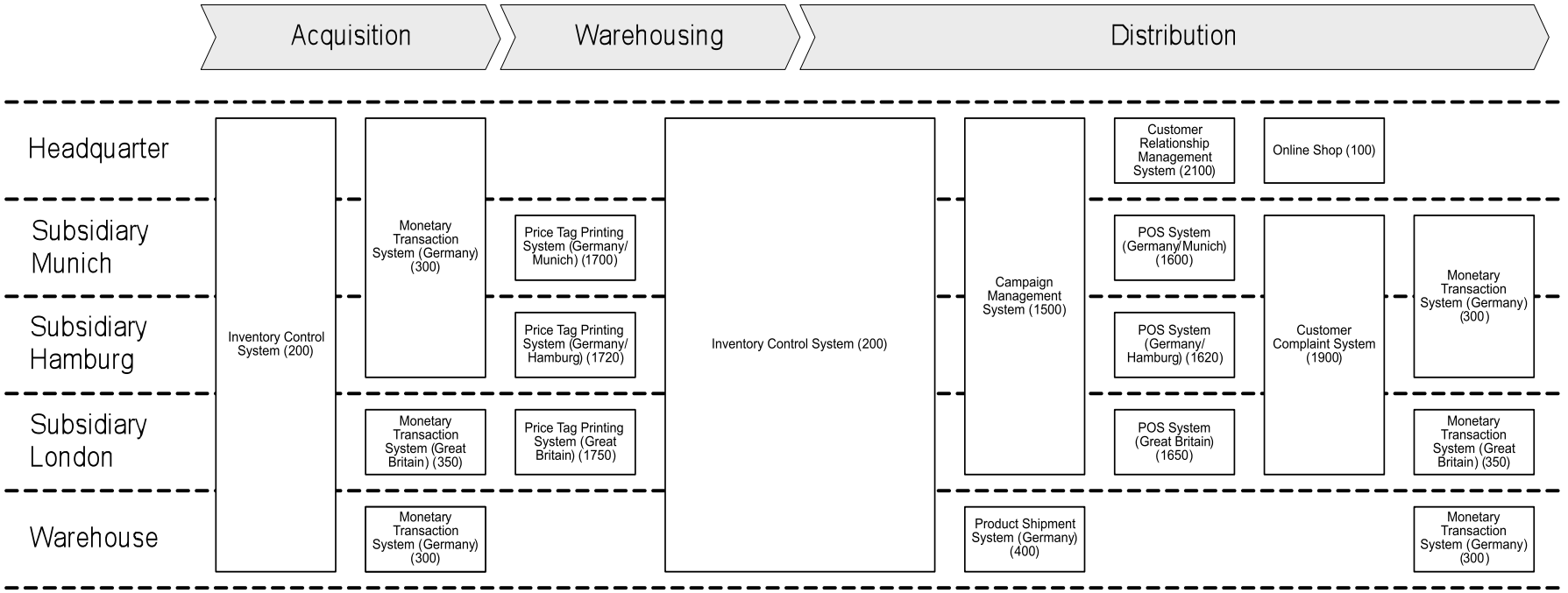
Prozessunterstützungskarte (3) – Vertikale Integration



Prozessunterstützungskarte (4) – Horizontale Integration



Prozessunterstützungskarte (5) – Horizontale + Vertikale Integration



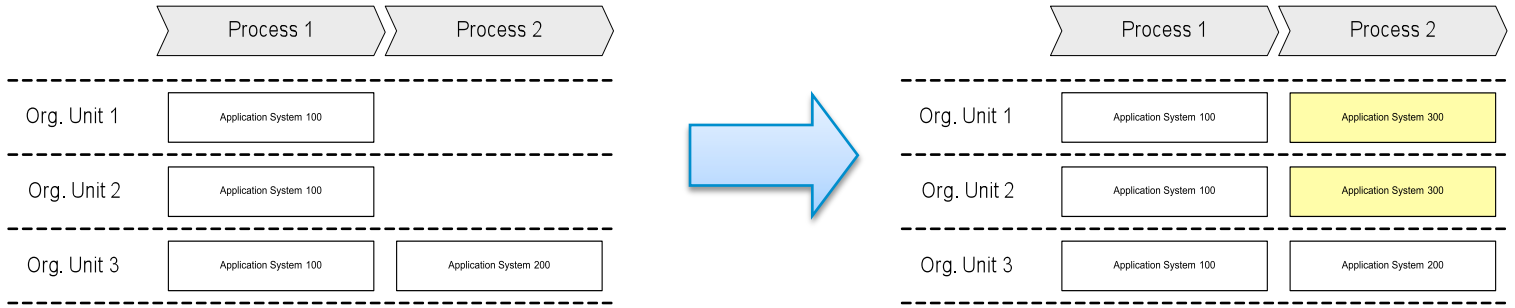
Agenda

- Motivation und thematische Einordnung
- Softwarekartographie
- UML für Softwarekarten?
- Anwendung von Softwarekarten
  - Sichten und Betrachtungswinkel bei Softwarekarten
  - Operationen auf Softwarekarten
  - Strukturierungsprinzipien für Anwendungslandschaften
- Wege zum Enterprise Architecture Management
- Fazit

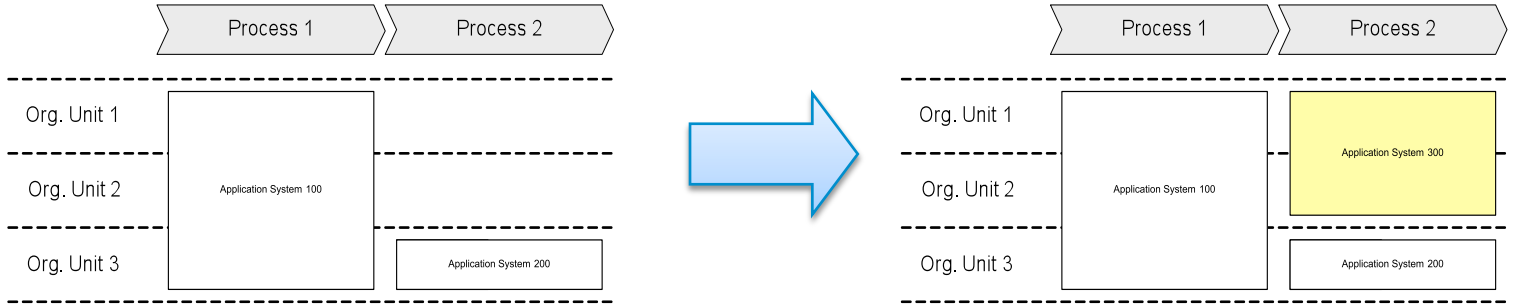
Veränderungen in der Anwendungslandschaft und ihre Visualisierung – Beispiel 1

- Neue Geschäftsprozessunterstützung durch ein ***neues*** Anwendungssystem
  - Ergebnis: (create new MapSymbol)

Ohne Integration



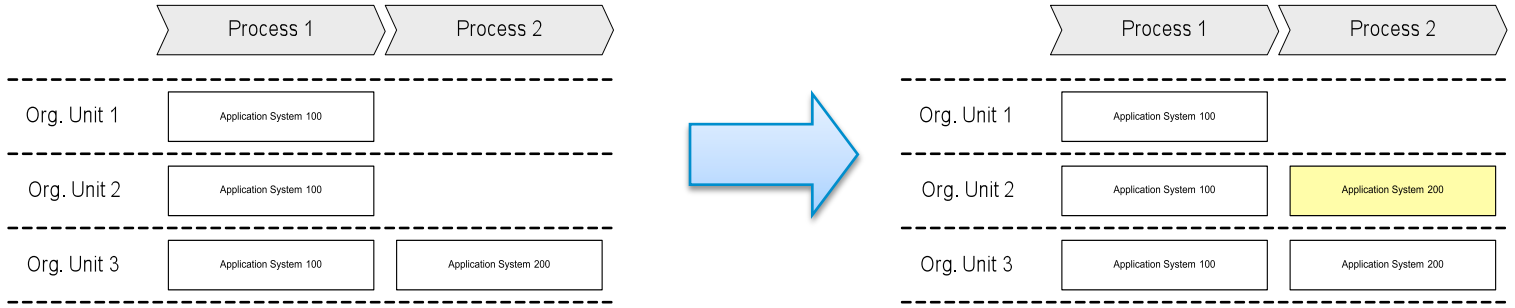
Vertikale Integration



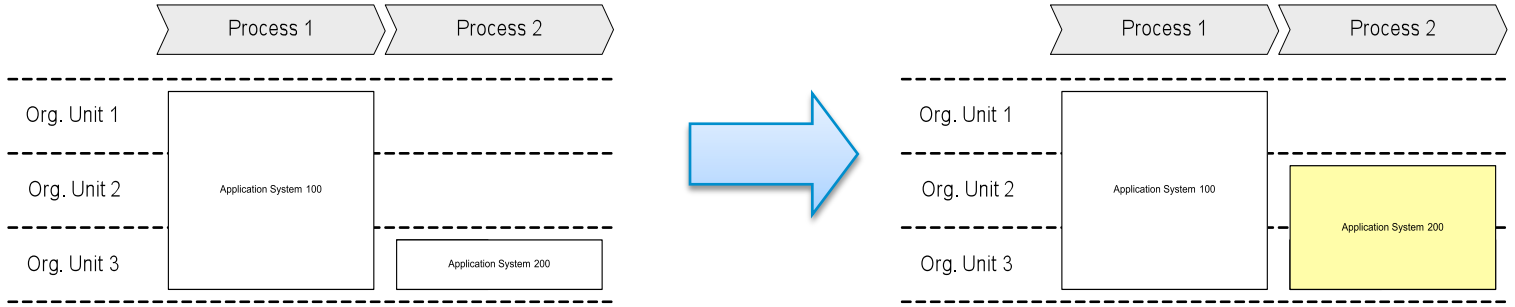
Veränderungen in der Anwendungslandschaft und ihre Visualisierung – Beispiel 2

- Neue Geschäftsprozessunterstützung durch ein ***existierendes*** Anwendungssystem
  - Ergebnis: (create new MapSymbol) OR (stretch existing MapSymbol)

Ohne Integration



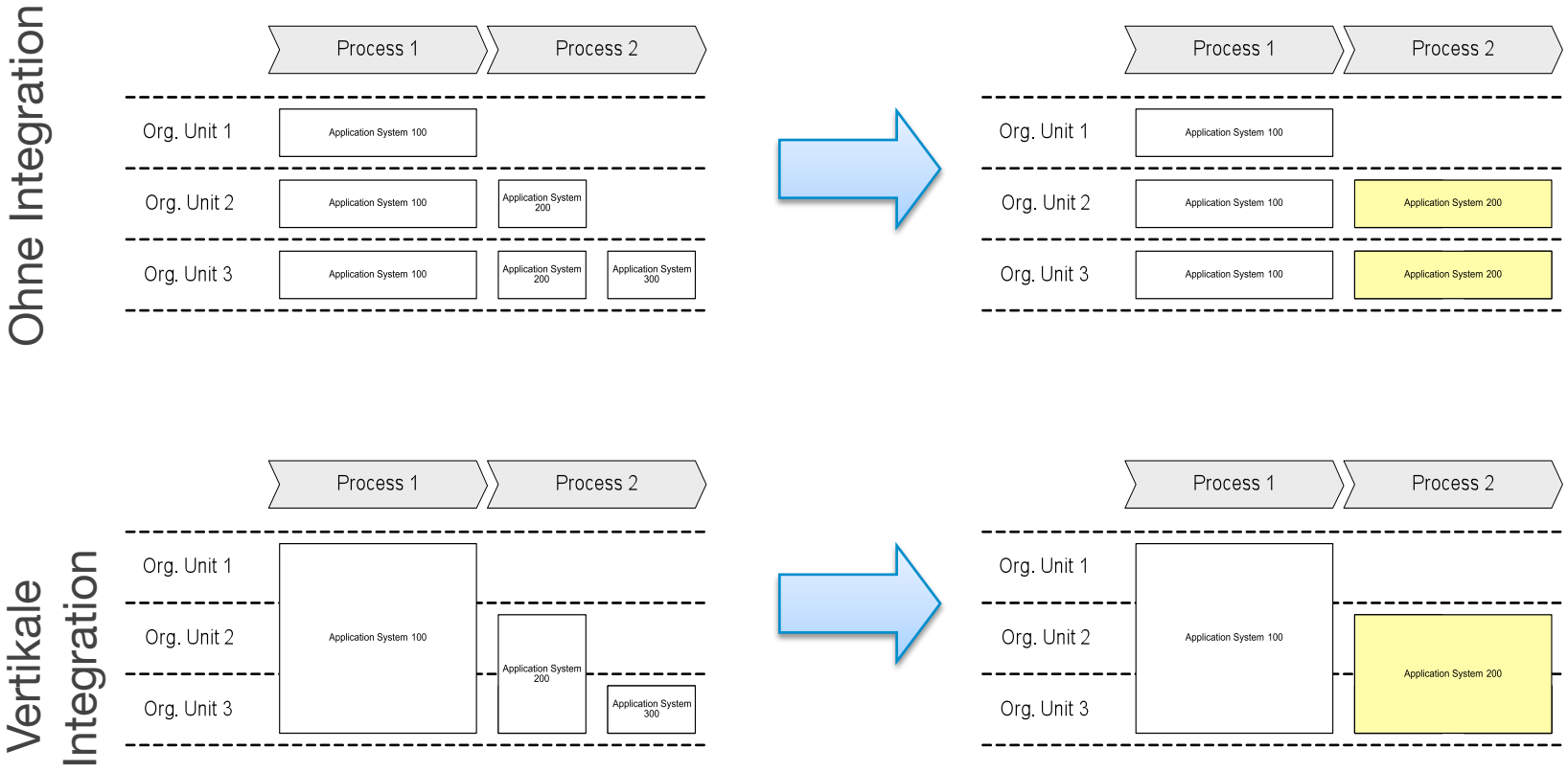
Vertikale Integration



Wenn die Geschäftsprozessunterstützung von Anwendungssystemversionen visualisiert wird, können die Darstellungen anders aussehen.

Veränderungen in der Anwendungslandschaft und ihre Visualisierung – Beispiel 3

- Konsolidiere die Geschäftsprozessunterstützung von zwei Anwendungssystemen
  - Ergebnis: (delete obsolete MapSymbol) AND ((create new MapSymbol) OR (stretch existing MapSymbol))



Veränderungen in der Anwendungslandschaft und ihre Visualisierung – Fortsetzung

sebis

… weitere Beispiele sind möglich

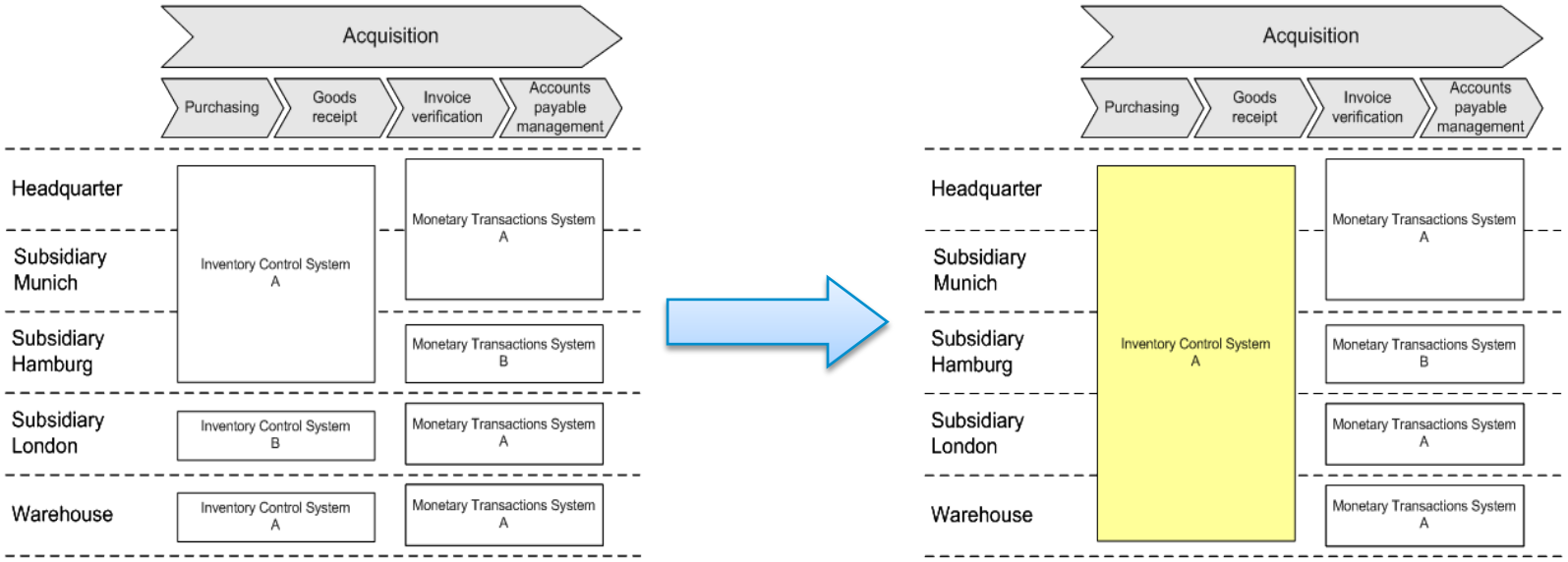
- Ablösen eines existierenden Anwendungssystems durch ein neues Anwendungssystem
- Dekomposition eines Anwendungssystems in mehrere Systeme
- Ersatzloses Herauslösen eines Anwendungssystem

Agenda

- Motivation und thematische Einordnung
- Softwarekartographie
- UML für Softwarekarten?
- Anwendung von Softwarekarten
  - Sichten und Betrachtungswinkel bei Softwarekarten
  - Operationen auf Softwarekarten
  - Strukturierungsprinzipien für Anwendungslandschaften
- Wege zum Enterprise Architecture Management
- Fazit

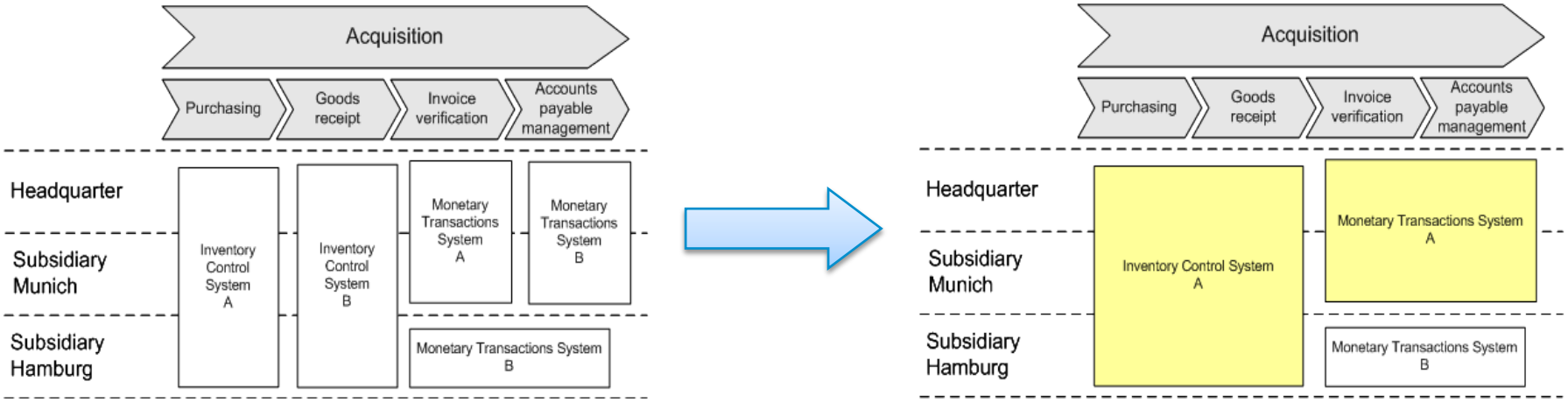
Strukturierungsprinzipien für Anwendungs-landschaften – Vertikale Integration

- Vertikale Integration
  - Mehrere Organisationseinheiten nutzen das gleiche Anwendungssystem für denselben Geschäftsprozess
    - oder verschiedene Produkte werden von dem gleichen Anwendungssystem unterstützt
  - Erzielen von Skaleneffekten (*Economies of Scale*)
  - Reduzieren von Betriebs- und Wartungskosten
  - Aber: Flexibilität der einzelnen Organisationseinheit wird reduziert



Strukturierungsprinzipien für Anwendungs-landschaften – Horizontale Integration

- Horizontale Integration
  - Zahlreiche aufeinanderfolgende Geschäftsprozesse werden durch das gleiche Anwendungssystem unterstützt
  - Reduzieren von Betriebs- und Wartungskosten
  - Aber: Horizontale Integration ist nicht in jedem Fall zielgerichtet
    - Teilen von Zuständigkeiten erhöht die Flexibilität, Wartbarkeit, …

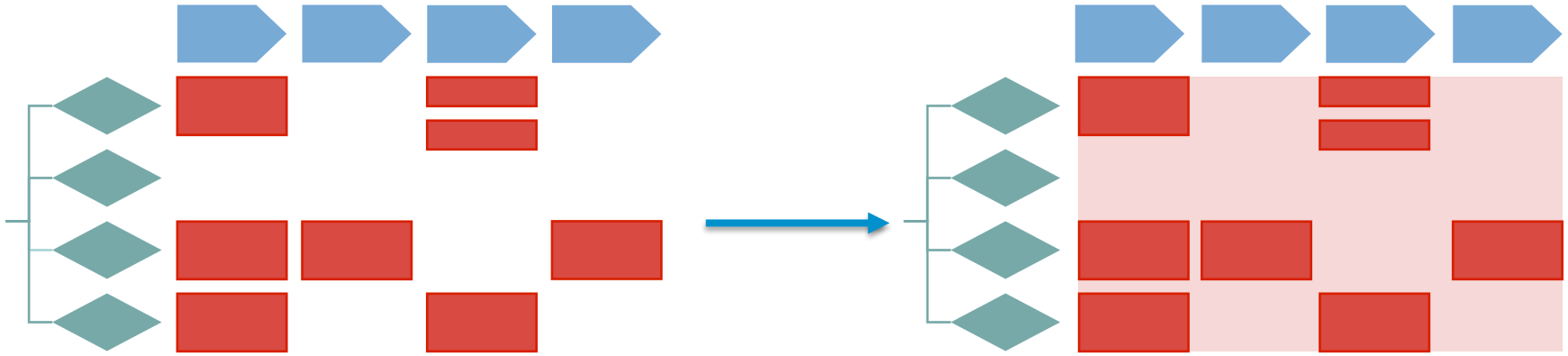


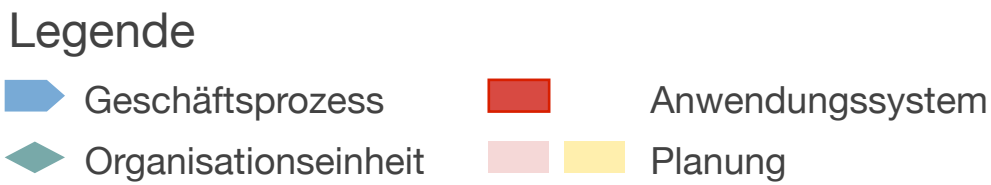
Strukturierungsprinzipien für AWL – Horizontale vs. Vertikale Integration (1)

sebis

- *Vollständige* Integration der Anwendungslandschaft durch Kombination von horizontaler und vertikaler Integration
  - Alle Geschäftsprozesse bei allen Organisationseinheiten wird durch ein *einziges* Anwendungssystem unterstützt

➡ Unrealistisch und viel zu komplex!

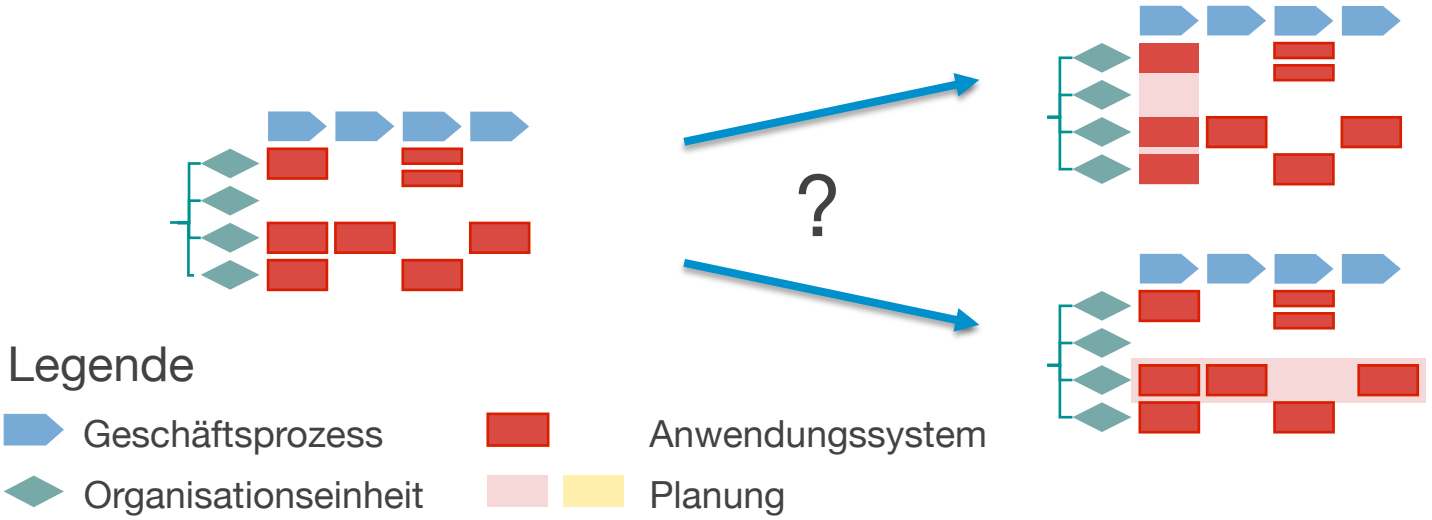




Quelle: BMW Group – Strategie, Planung und Steuerung Group IT

Strukturierungsprinzipien für AWL – Horizontale vs. Vertikale Integration (2)

- Was ist die bevorzugte Lösung, vertikale oder horizontale Integration?
  - Jeder Fall benötigt eine individuelle Entscheidung
    - Vorteile und Nachteile abwägen, um eine passende Lösung zu finden
  - Unterstützungsprozesse (auch Sekundärprozesse), wie Accounting, HR etc., sind Kandidaten für vertikale Integration
  - Primärprozesse (Produktion, Vertrieb etc.), welche einen Diversifikationscharakter besitzen, benötigen eine niedrige vertikale Integration, um Flexibilität zu erhalten bzw. zu erreichen

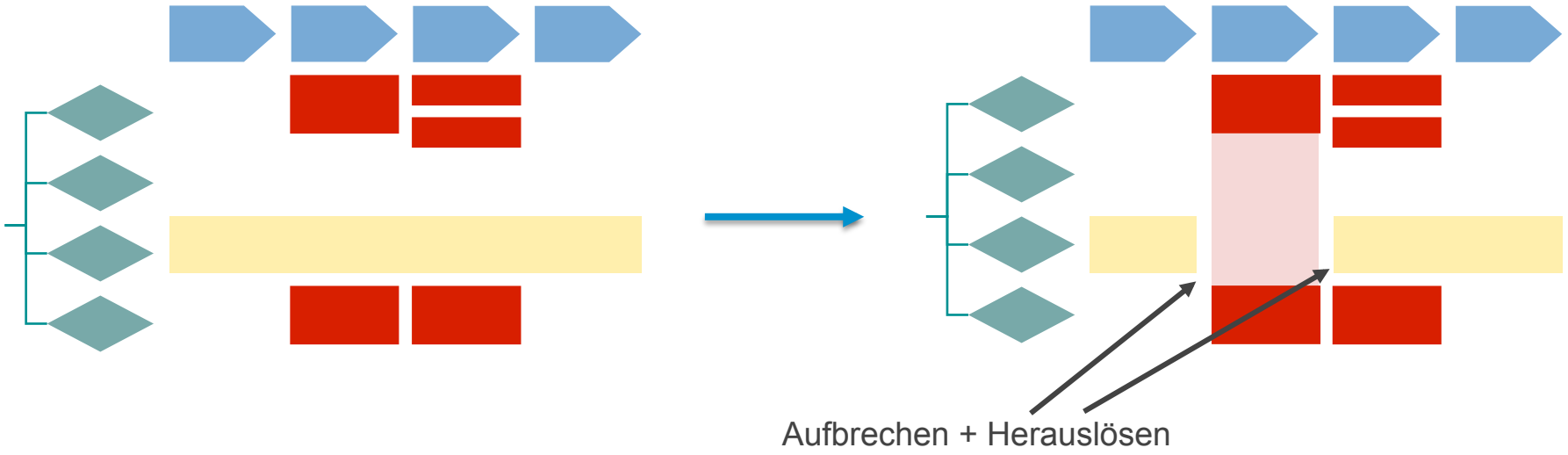


Quelle: BMW Group – Strategie, Planung und Steuerung Group IT

Strukturierungsprinzipien für AWL – Horizontale vs. Vertikale Integration (3)

sebis

- Einführung von vertikaler Integration kann horizontale Integration aufbrechen



Legende

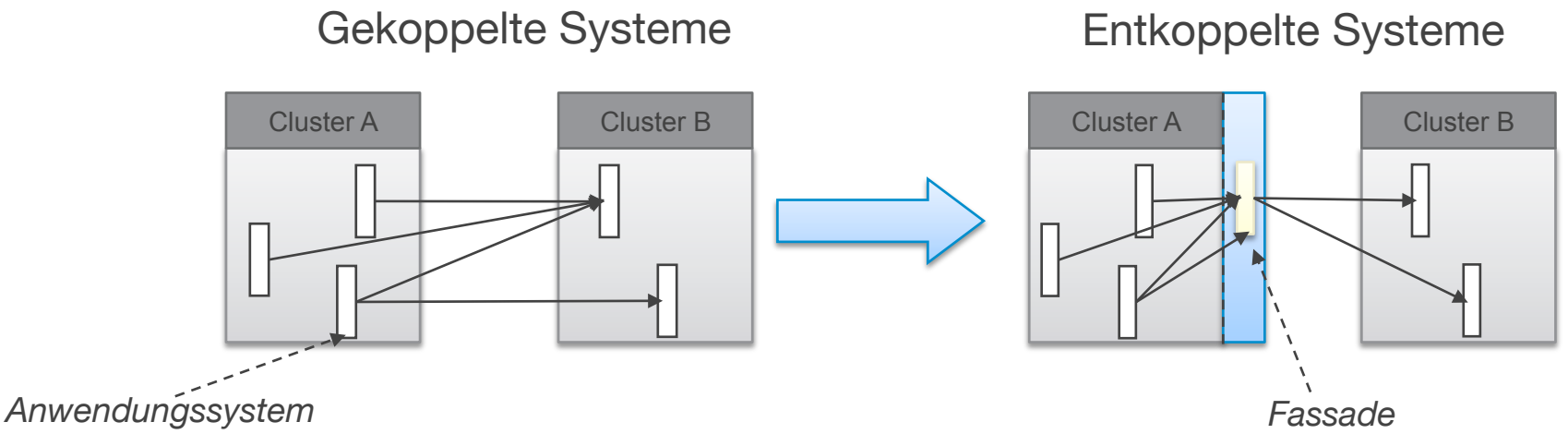
Geschäftsprozess
Organisationseinheit
Anwendungssystem
Planung

Quelle: BMW Group – Strategie, Planung und Steuerung Group IT

Strukturierungsprinzipien für Anwendungslandschaften – Entkopplung

sebis

- Cluster-Bildung (auch Building Blocks)
  - Organisationseinheiten
  - Domänen oder Sparten oder „Lines of Business“
- Einführung von Fassaden zwischen Clustern
- Ziele
  - „Separation of Concerns“
  - Komplexität beherrschen
    - Schnittstellenreduktion zwischen den Clustern mittels definierter Kommunikationswege



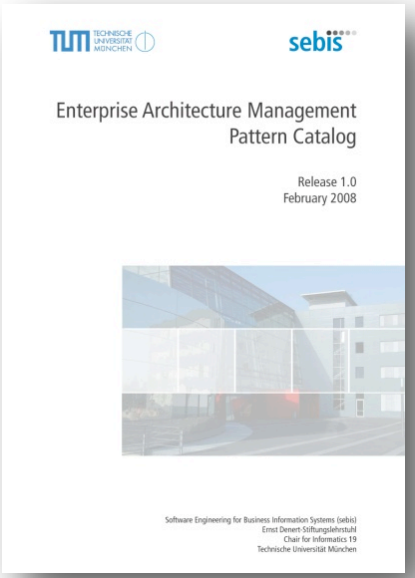
Weitere Strukturierungsprinzipien für Anwendungslandschaften

sebis

- Musterarchitekturen und Musterlösungen
  - Standardisierte Vorgaben für Softwarearchitekturen inkl. Lösungsbausteinen
    - Musterarchitektur: 4-tier-architecture mit Thin-Client, Webserver, Applikationsserver und Datenbank
    - Musterlösung: Internet Exporer, Apache HTTP Server, BEA WebLogic und Oracle 9i
- Service-orientierung
  - Definition von Infrastruktur-Services, z.B.
    - E-Mail-Service, LAN-Service, Datenbank-Service, …
  - Definition von Business-Services
    - Fokussierung auf fachliche Definition von Schnittstellen
    - Ermöglicht Wiederverwendung standardisierte Kommunikation
  - Service-orientierung und SLAs vereinfachen IT-Controlling
    - Einführen von Service-Level-Agreements (SLA)
    - SLAs können verrechnet werden

Agenda

- Motivation und thematische Einordnung
- Softwarekartographie
- UML für Softwarekarten?
- Anwendung von Softwarekarten
  - Sichten und Betrachtungswinkel bei Softwarekarten
  - Operationen auf Softwarekarten
  - Strukturierungsprinzipien für Anwendungslandschaften
- EAM Pattern Catalog
- Fazit



Download unter http://www.softwarekartographie.de/eampc

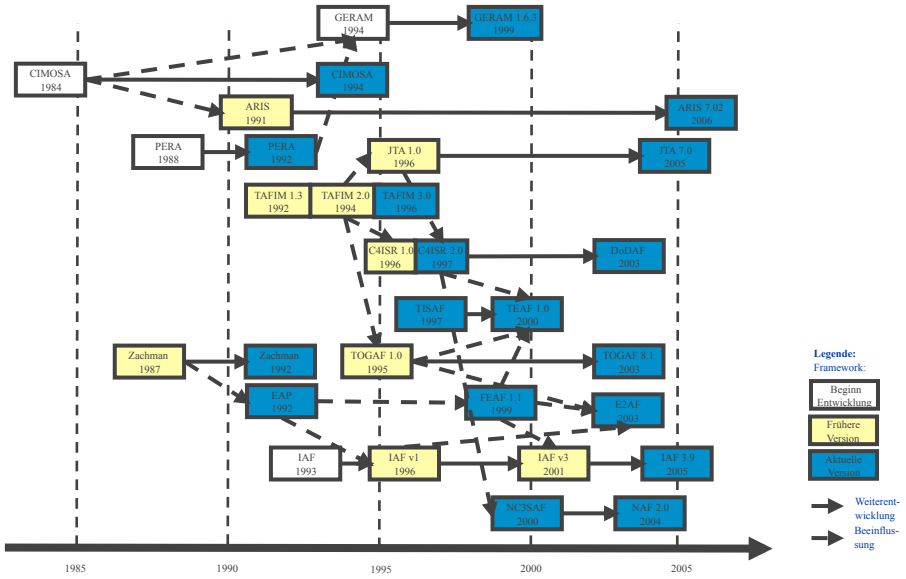
Motivation (1) – Methodologies for EAM

Various EAM frameworks exist, but distribution in practice is limited because of either reasons

- too abstract for practical utilization (e.g. Zachmann)
- too extensive for practical utilization (e.g. TOGAF)
- usually a “complete or nothing” approach
- limited adaptability of frameworks to company needs

Greenfield approaches

- are usually not based on best practices
  ➔ typical errors are repeated
- are usually not adequately documented
- tend to grow unlimited



Motivation (2) – Viewpoints in EAM

sebis

Defined viewpoints for reoccurring and known problems in Software Engineering are

- described in the literature
- universal and can be used organization independent
- constitute reusable knowledge and experience
- are continuously developed

Modeling languages and methods for Enterprise Architectures and Application Landscapes are currently emerging

- ArchiMate (http://www.archimate.com)
- Softwarekartographie (http://www.softwarekartographie.de)

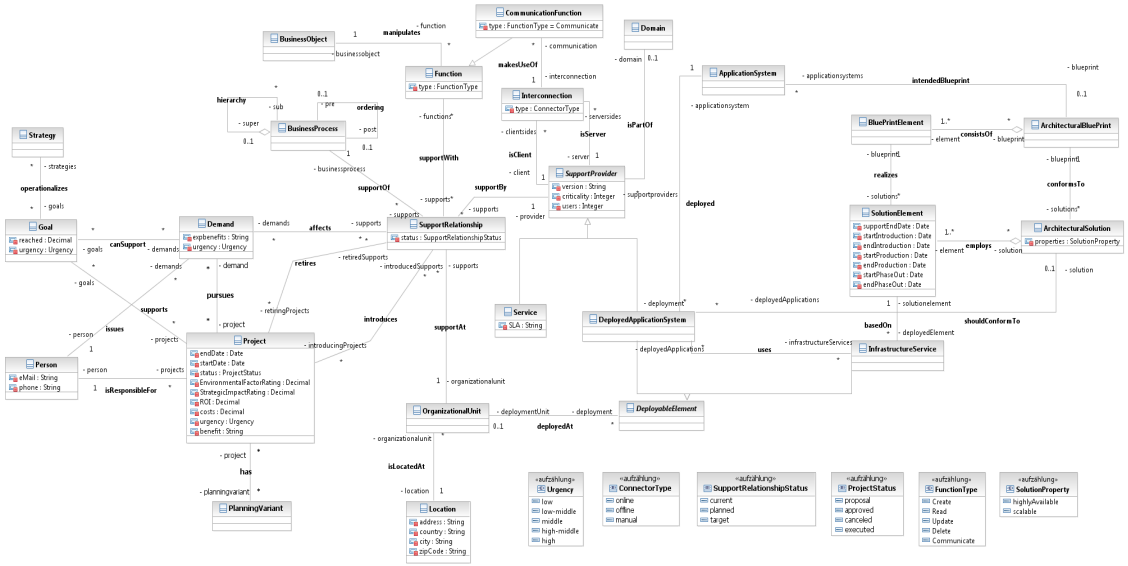
But: There are still many organization specific viewpoints

- Knowledge about addressed concerns and utilization are often not documented
- Utilization only within one organization
- Exchange of experiences only within one organization
- Future development is limited

Motivation (3) – Information Models: Which Concerns are addressed where in what Way?

sebis

EA management information models tend to be rather complex:



Exemplary information model from the EAMTS2008 [Bu08]

Sizes of actual information models in practice:

- ~ 50 entity types (Tool A)
- ~ 54 entity types (Tool B)
- ~ 210 entity types (Tool C)
- ~ 220 entity types (Tool D)
- ~ 470 entity types (Tool E)

Important questions in model utilization become difficult to answer:

- What structures in an information model are suitable for addressing which concerns?
- How can these concerns be addressed using the information model?
- How can adaptations to organization-specific needs be made?

A Patterns is a general, reusable solution to a common problem in a given context

sebis

Analogy to other disciplines, like architecture, software engineering, etc. Address recurring problems with patterns.

Alexander et al. [Al77]

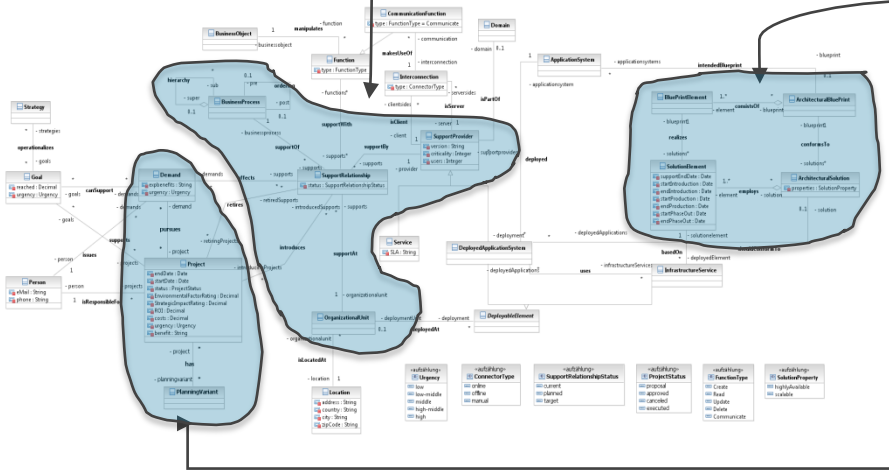
- Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.
- Each pattern is a three-part rule, which expresses a relation between a certain context, a problem and a solution

Gamma et al. [Ga94]

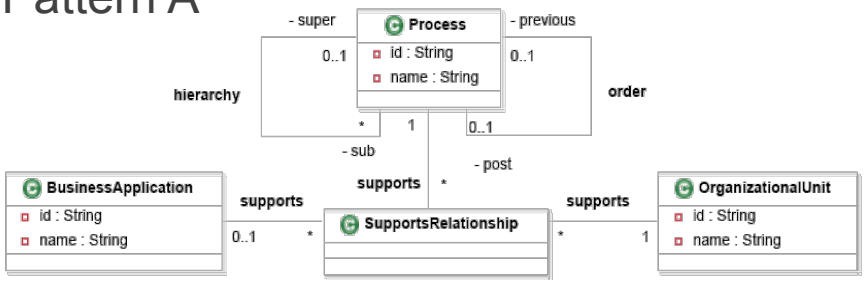
- Descriptions of communicating objects and classes that are customized to solve a general design problem in a particular context.

Using Patterns: Constructing EA Management Information Models Based on I-Patterns

I-Pattern as manageable, concern-specific units, of which an information model can be composed



Pattern A



Pattern B

...

Pattern X

EA Management Pattern Approach

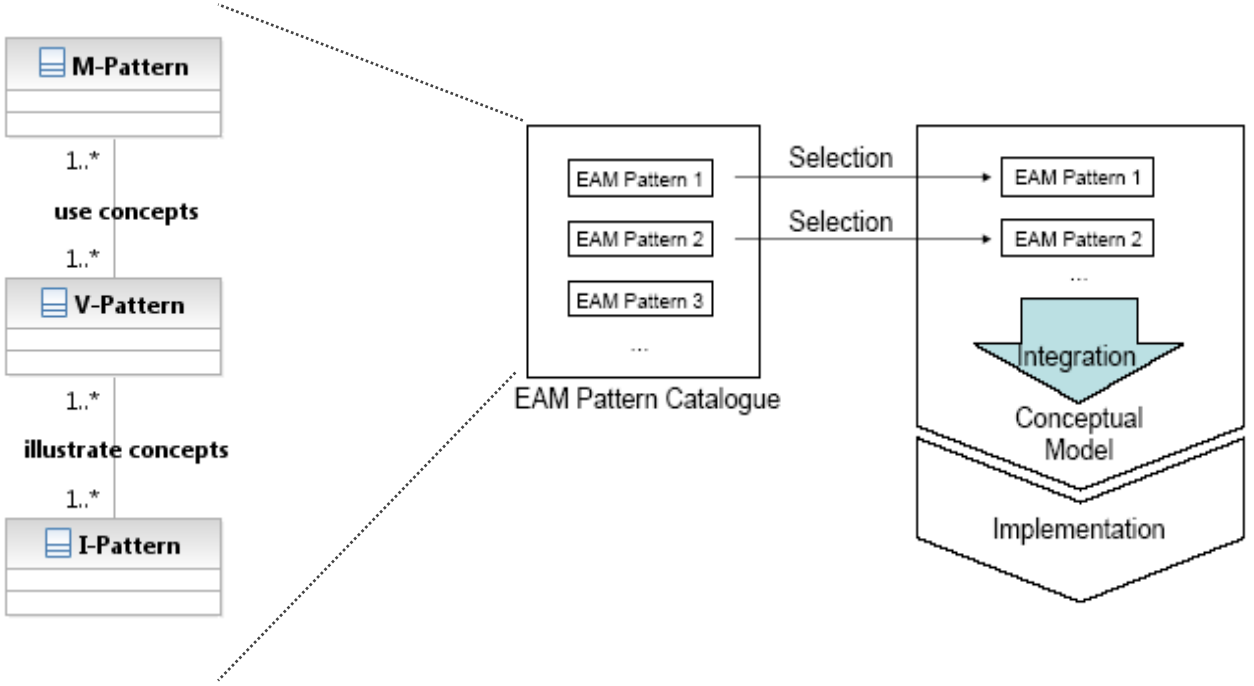
Important questions in model utilization EA management patterns are meant to answer:

- What structures of an information model are suitable for addressing which concerns?
- How can these concerns be addressed using the information model?
- What visualizations are thereby employed?
- How can adaptations resulting from organization-specific needs be made?

Concerns and steps for addressing concerns

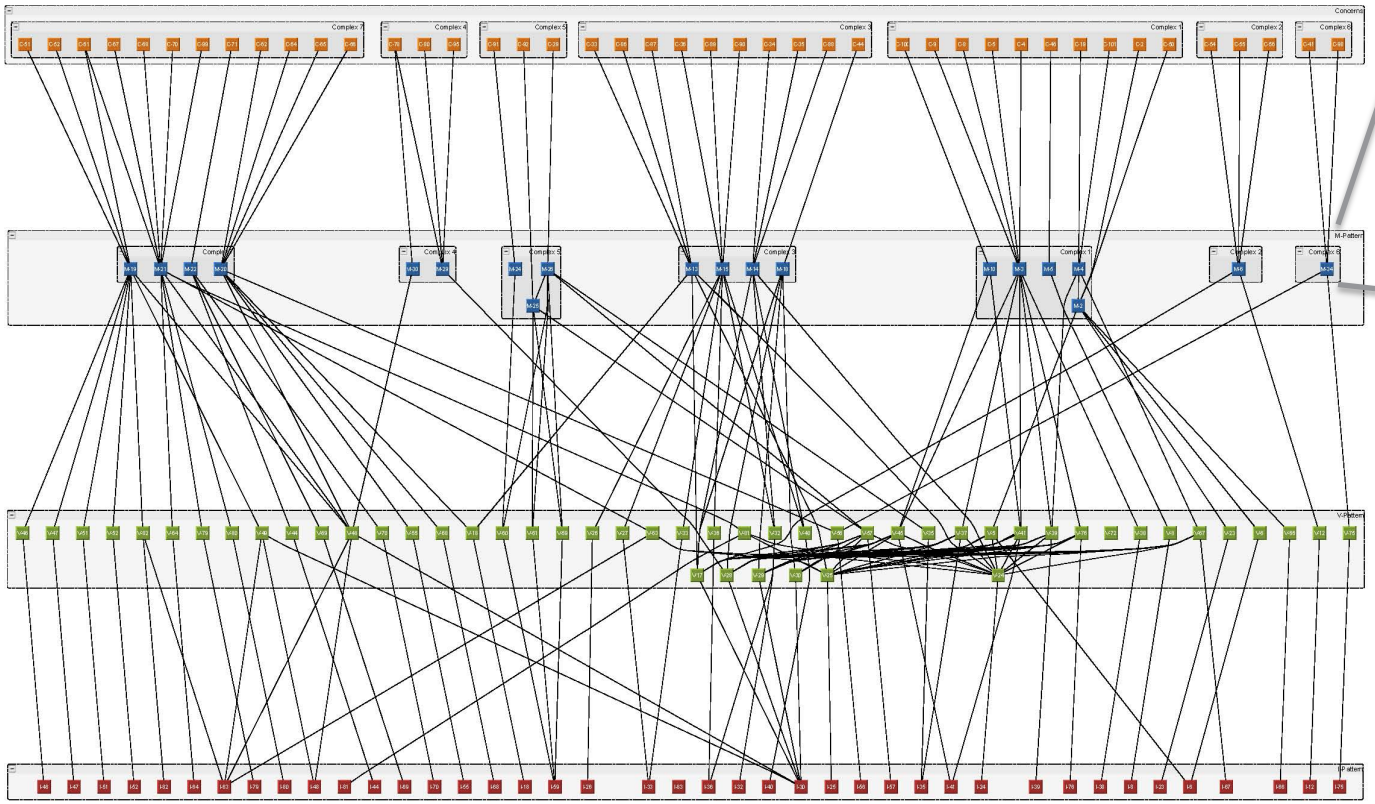
Visualization of needed information

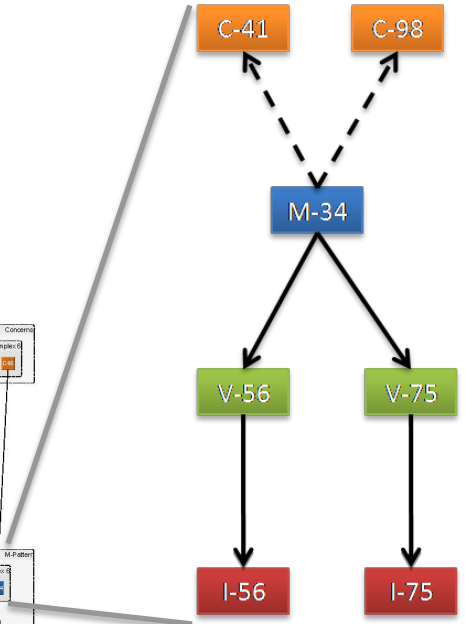
Information needed for addressing concerns and creating visualizations



EA Management Pattern Catalog

- 43 Concerns (48 have been excluded)
- 20 M-Patterns (10 have been excluded)
- 53 V-Patterns (21 have been excluded)
- 47 I-Patterns (19 have been excluded)





Uniform Structure of the EAM Patterns

sebis

Overview Section	
Id	An unique alphanumerical identifier
Name	A short and expressive name for the pattern
Alias	Names this pattern is also known as (optional)
Summary	A short summary of the EAM pattern
Version	Version number of the EAM pattern and date of the last changes
Problem Section	
	Concerns to be address by an M-Pattern (only included in M-Patterns)
Solution Section	
	Detail description of the EAM pattern
Consequence Section	
	Consequences resulting from the usage of the EAM pattern (optional)

EA Management Pattern Catalog – Usage Scenarios

sebis

- Establishing an organization-specific EA Management through EAM Pattern Integration
  - Concern based, light-weight, organization-specific approach to EAM based on best practices
- Inspiring and assessing an already implemented EA Management approach
  - Suggestions for extending an existing EAM approach
  - Comparison of current EAM approach to best practices
- EAM Pattern Catalog as a basis for academic research
  - Punctiform approaches for specific EAM topics exist, but no integration to a holistic EAM approach possible
  - Common ground for research on EAM, which can be iteratively enhanced and extended

Scenarios for EAM Pattern Catalog utilization

sebis

- Planning the future development of the application landscape
  - Current, planned, and target landscapes
  - Visualizing changes in the application landscape
  - Horizontal and vertical integration
- Visualizing and evaluating the conformance to architectural blueprints
- Evaluating fullfilment of service level agreements (SLA)
- Analyzing dependencies between infrastructure elements and business applications
- … and a lot more

Exemplary Scenario – Redundancy Reduction

Concern

- C-44: How can the operating expenses and maintenance costs be reduced, e.g. by identification of business applications providing the same functionality (redundancy)?

M-Pattern

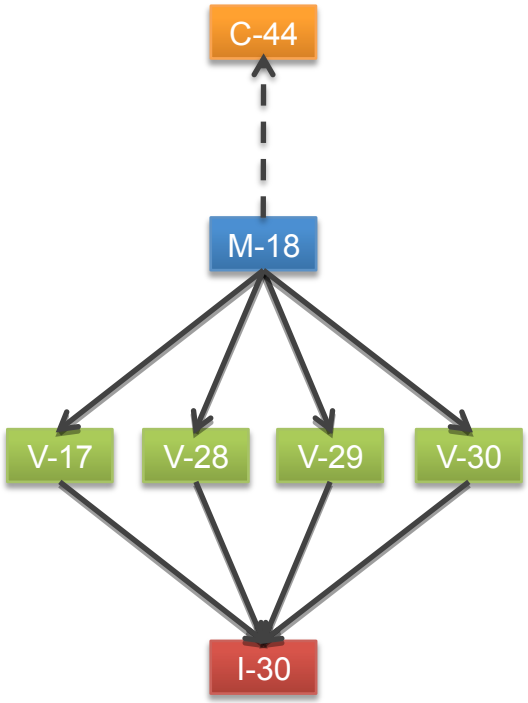
- M-18: Horizontal and vertical integration (see previous slides for explanation)

V-Pattern

- V-17: Process Support Map
- V-28: Process Support Map visualizing horizontal Integration
- V-29: Process Support Map visualizing vertical Integration
- V-30: Process Support Map visualizing vertical and horizontal Integration

I-Pattern

- I-30: Process Support



Exemplary Scenario – Introduction of utilized M-Pattern M-18

4.3.4 Horizontal and vertical integration (M-18)

M-Pattern Overview	
Id	M-18
Name	Horizontal and vertical integration
Alias	
Summary	The M-Pattern helps to analyze the level of process support provided by the business applications. Thereby, vertical integration deals with the level of uniformity of process support for different e.g. organizational units, products or locations. Horizontal integration refers to the level continuity of process support provided, thus a business applications provides support for more than one process.
Version	1.0

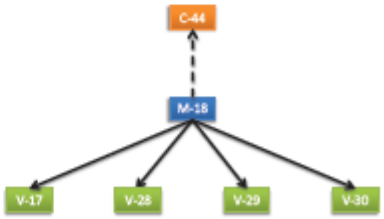
Problem Section

The methodology addresses the following concerns:

- C-44: How can the operating expenses and maintenance costs be reduced, e.g. by identification of business applications providing the same functionality (redundancy)?

Solution Section

The methodology relies on the usage of a process support map, which visualizes a process chain, as a linearly ordered sequence of process steps on the x-axis. The y-axis of the process support map can differ according to the enterprise's kind or the addressed concerns. Elements that can be visualized on the y-axis are for example locations or organizational units.



The methodology analyzes the process support provided by different business applications. Thereby, horizontal integration means that several successive business processes are continually supported by one business application. Vertical integration describes the uniform process support provided by one business application for a dedicated number of organizational units or locations. Supporting the analysis of process support according to vertical an horizontal integration, viewpoints as e.g. V-28, V-29, and V-30 can be used. These visualize the integration by expanding the border of the business application in the y- or x-direction.

The methodology uses the following viewpoints:

- V-17: Process Support Map
- V-28: Process Support Map visualizing horizontal Integration

- V-29: Process Support Map visualizing vertical Integration
- V-30: Process Support Map visualizing vertical and horizontal Integration

The objective of this methodology is to identify potential for cost reductions as well as potentials for optimization. Cost reduction may be achieved by e.g. changing the process support of organizational units, that use different business applications providing the same functionality to one standardized business application. Optimization possibilities may be achieved by e.g. using identified economies of scale.

Consequence Section

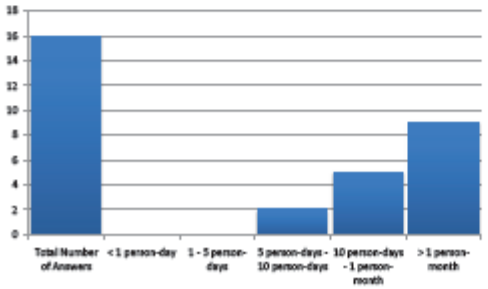
The methodology only deals with business processes on a high level of abstraction. Typically the view on a process chain as a linearly ordered sequence of processes is only possible for the levels 0 to 3. Thus, the methodology cannot be downscaled to a more detailed level.

However, the analysis as introduced above provides support to identify potential candidates of redundancy on a high abstraction level. Nevertheless, the identified potential redundancies need to be analyzed in more detail, as they can turn out to be no redundancies at all. They might, for example support other subprocesses of the process step as visualized on the aggregated view of the software map.

Moreover, if there are redundancies, they might have been deliberately introduced, e.g. in order to achieve a higher flexibility. In such cases, it may be reasonable to retain the redundancy. In case that no such reasons can be found, the results from the analysis regarding redundancies can be used as input for activities defining visions or plans for the evolution of the application landscape. This can include definitions of project proposals that serve the elimination of the redundancies.

Nevertheless, it must be noted, that an integration in both directions, horizontally and vertically is not always possible as the symbols might intersect or overlap.

The data collection effort per year for information about *detailed information about business processes, including the responsible organizational units*, and *supporting business applications* etc. has been stated by practitioners using such methodologies as:



For further information concerning the evaluation of methodology M-18 see Section A.2.15.

Exemplary Scenario – Introduction of utilized V-Pattern V-17

5.5 Viewpoint V-17

V-Pattern Overview	
Id	V-17
Name	Process Support Map
Alias	
Summary	This V-Pattern visualizes, which business applications support which business processes at which organizational units.
Version	1.0

5.5.1 Solution Section

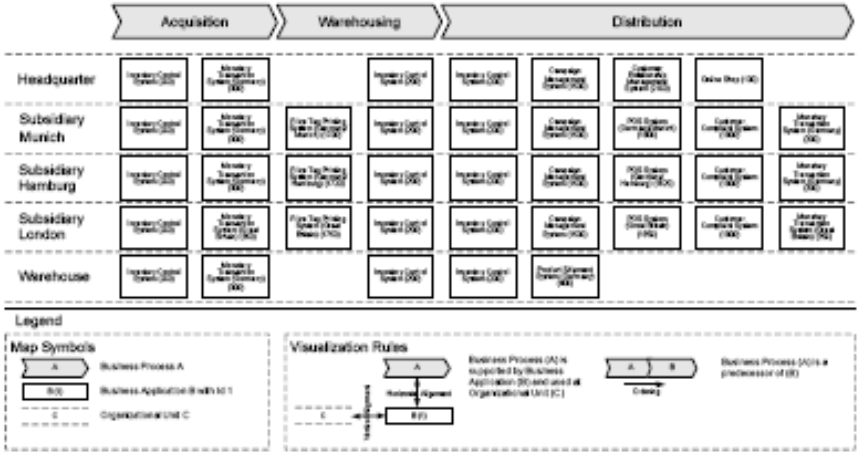
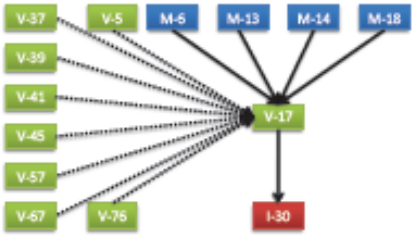


Figure 5.5: Viewpoint V-17

5. Viewpoint Patterns (V-Patterns)

This V-Pattern, a process support map, visualizes, which business applications support which business processes at which organizational units. This V-Pattern is based on I-Pattern I-30.



5.5.2 Consequence Section

This V-Pattern can also be used to show the relationship between business applications and locations, instead of business applications and organizational units. In this case the corresponding I-Pattern has to be adapted to include an entity for location.

Exemplary Scenario – Introduction of utilized I-Pattern I-30

6.9 I-Pattern I-30

I-Pattern Overview	
Id	I-30
Name	Process Support
Alias	
Summary	
Version	1.0

6.9.1 Solution Section

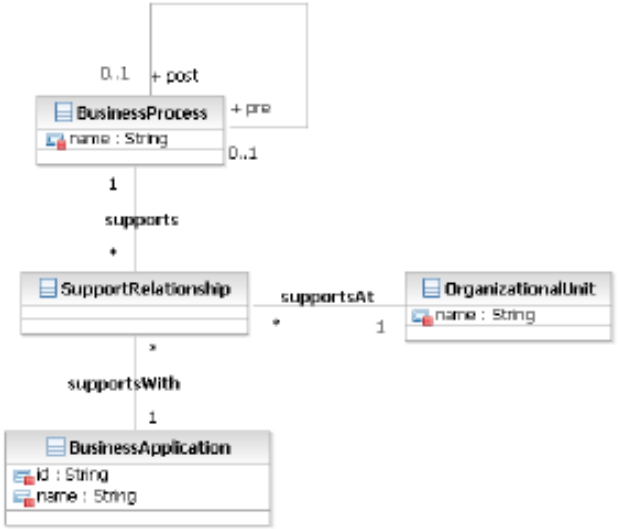


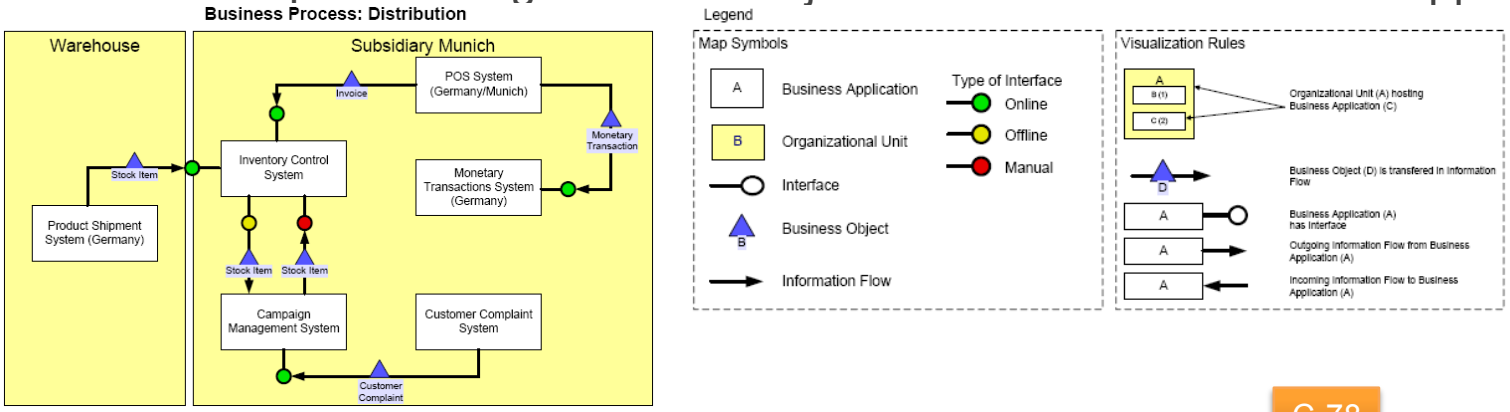
Figure 6.9: Information Model I-30

6. Information Model Patterns (I-Patterns)

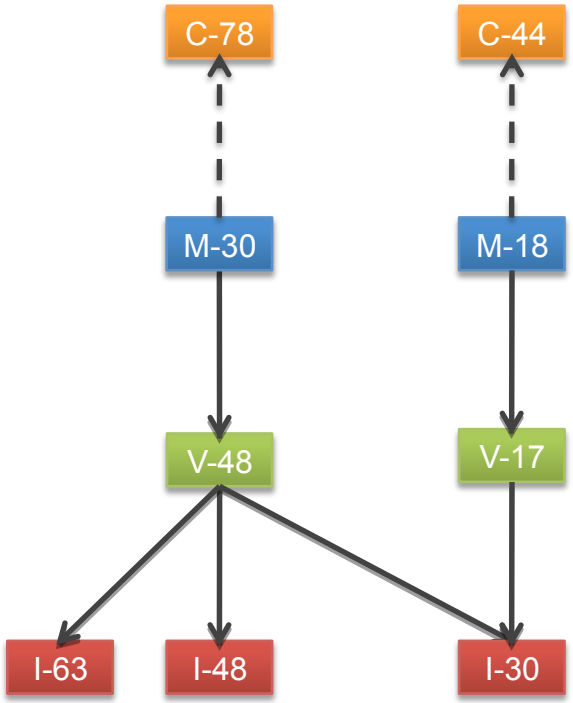
- BusinessApplication: A business application is a software system, which is part of an information system of an organization. An information system is according to [Kre05] understood as a sociotechnical system, which is, besides the software system, made up of the infrastructure the software system is based on, and a social component, namely the employees or stakeholders concerned with it. Thereby, infrastructure and social component are not considered as belonging to the business application, while the characterization "business" restricts the term to applications that support at least one process of the respective organization. Thus, business application denotes here an acutal deployement of a software.
- SupportRelationship: Represents the support of a process by a business application at a specific organizational unit. Basically, it constitutes, together with its three associations, a ternary relationship between BusinessProcess, OrganizationalUnit, and BusinessApplication. This is necessary in order to be able to tell exactly which organizational unit uses which business application to support a given process.
- BusinessProcess: According to [Kre05], defined as a sequence of logical individual functions with connections between them. [DFH03] states input and output factors and a defined process objective as important characteristics of a business process. The business process should not be identified with single process steps or individual functions, but with high-level processes at a level similar to the one used in value chains.
- OrganizationalUnit: An organizational unit represents a subdivision of the organization according to its internal structure. A possible example are the entities showing up in an organigram.

Outlook: Utilizing collected Information to address additional Concerns

- V-48: Cluster Map visualizing Business Object Flows between Business Applications



- M-30: Business Process Data Flow Analysis
  - The M-Pattern analyzes the support provided by the application landscape for an individual business process. The focus of the methodology lies on a single business process and its support instead of the business process landscape in a holistic view. Thereby, the data owns between different business applications are analyzed.
- C-78: To which extent are the business processes supported by business applications? Which business processes are supported manually? Can the automated support be extended?



Agenda

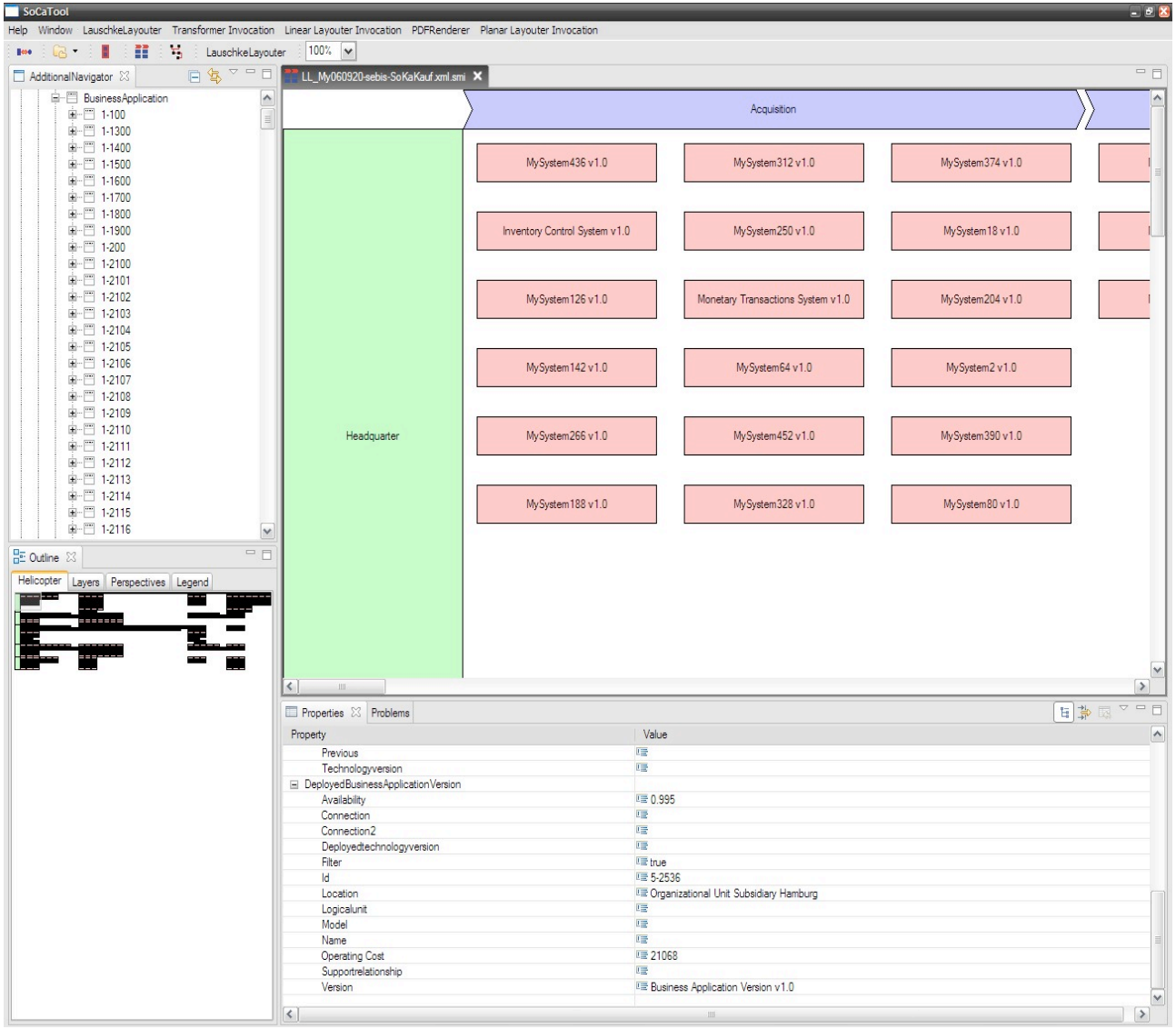
- Motivation und thematische Einordnung
- Softwarekartographie
- UML für Softwarekarten?
- Anwendung von Softwarekarten
- Wege zum Enterprise Architecture Management
- Fazit

Fazit: Malen vs. Modellieren und EAM Patterns

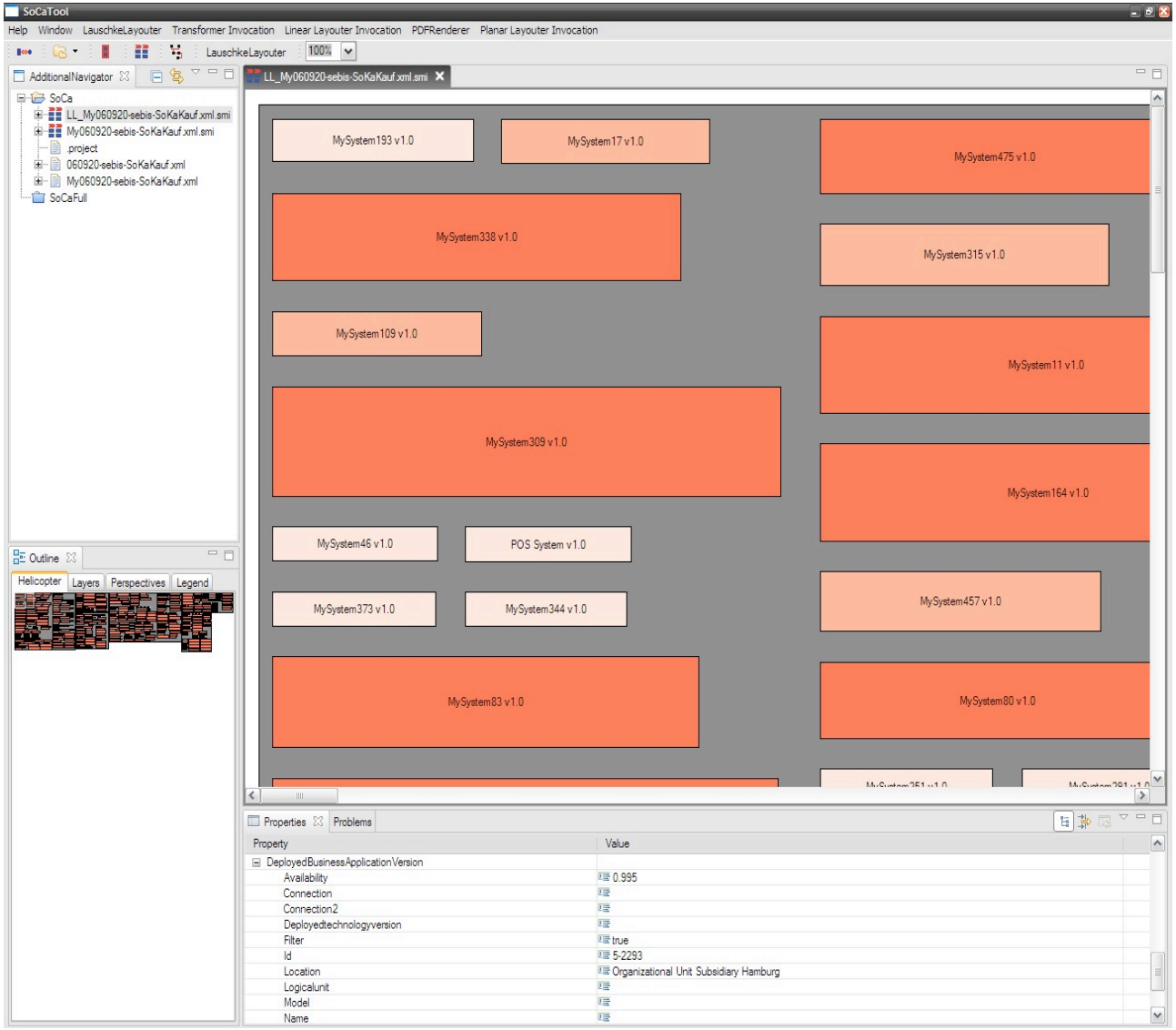
sebis

- Für das Management sollten die Informationen im Bild stimmig sein.
  - Wie das Bild entsteht, ist „fast“ egal.
  - Modellierungswerkzeuge helfen,
    - Bilder zu bauen, die konsistent mit den Informationen sind,
    - ermöglichen es die Bilder zu modifizieren und
    - gleichzeitig die Daten zu aktualisieren.
    - ➡ Aus den Bildern werden Modelle!
  - Es entwickeln sich zusehend Methoden im EA Management, die Modelle verwenden, um die EA derart zu gestalten, dass IT und Geschäft effizienter und effektiver zusammenarbeiten.
  - In Präsentationen sind die gemalten Bilder vorherrschend,
    - aber wenn jemand nachfragt, wie die Bedeutung einzelner Elemente ist,
    - so sollte ein Modell, welches eine Semantik und Syntax hat und
    - welches einen konsistenten Informationsstand widerspiegelt,
    - existieren… hier helfen Werkzeuge!
- EAM Patterns stellen einen strukturierten, leichtgewichtigen und auf best practices basierenden Ansatz für die Einführung von neuen und Erweiterung bestehender EAM Initiativen dar.
  - Anwendung der Pattern Idee auf das EA Management

Ein Beispiel aktueller Forschungsarbeit: *Software Cartography Tool* (1)



Ein Beispiel aktueller Forschungsarbeit: *Software Cartography Tool* (2)



Vielen Dank für Ihre Aufmerksamkeit!

sebis

Diskussion...

Informationen zum Forschungsprojekt unter
http://www.softwarekartographie.de oder http://www.systemcartography.info