System Cartography & EAM Best Practices

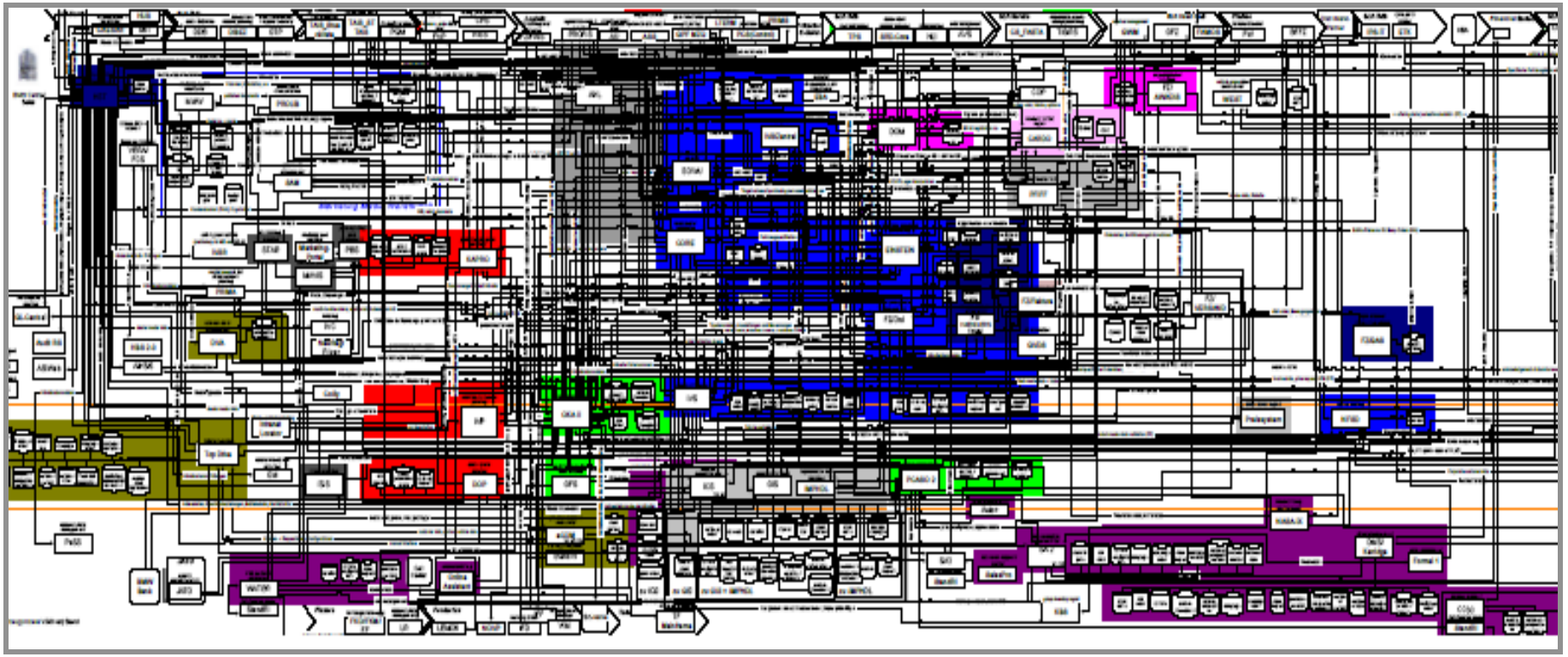
From application landscape maps towards situational EA management

Agenda and learning objectives of this unit

- Agenda
  - Software and system cartography
  - Architectural descriptions – the ISO Std. 42010
  - Challenges for EAM
  - BEAMS – Situational EA management

- At the end of this lecture you
  - know the basics of software cartography and software maps
  - understand how software maps can be used to document architectures
  - can apply a standardized terminology for architectural descriptions
  - understand the challenges arising in the context of managing complex application landscapes and enterprise architectures (EA)
  - know how to apply EAM best practices to typical EA-related problems
  - explain the meaning of the terms *current, planned,* and *target* state of an EA

Today's application landscapes consist of 10^2 -10^3 networked information systems



- Complexity ~ number of relationships
- IT agility does not keep pace with the increasing dynamicity of the business
- Number of services >> number of applications
  (smaller granularity + versioning)
- Extended enterprise: Coalitions, mergers, carve-outs, …

Application landscape ⇔ city

- **Shared characteristics**
  - networked system of semi-autonomous systems
  - alive, mostly growing, unbounded lifetime
  - people are key elements of the system
  - created and managed by people
  - to be financed by people
  - a long-term balance of interests has to be achieved
  - a holistic and long-term perspective is required (as-is, plan, to-be)
  - heterogeneity: managed core & evolutionary periphery

- **Challenges specific to application landscapes**
  - documentation of **ownerships** and derived **rights and obligations**
  - system benefit vs. individual benefits ➔ value & utility functions
  - shared vocabulary for communication ➔ holistic view
  - problem-specific abstractions to master the inherent complexity ➔ diagrams and views

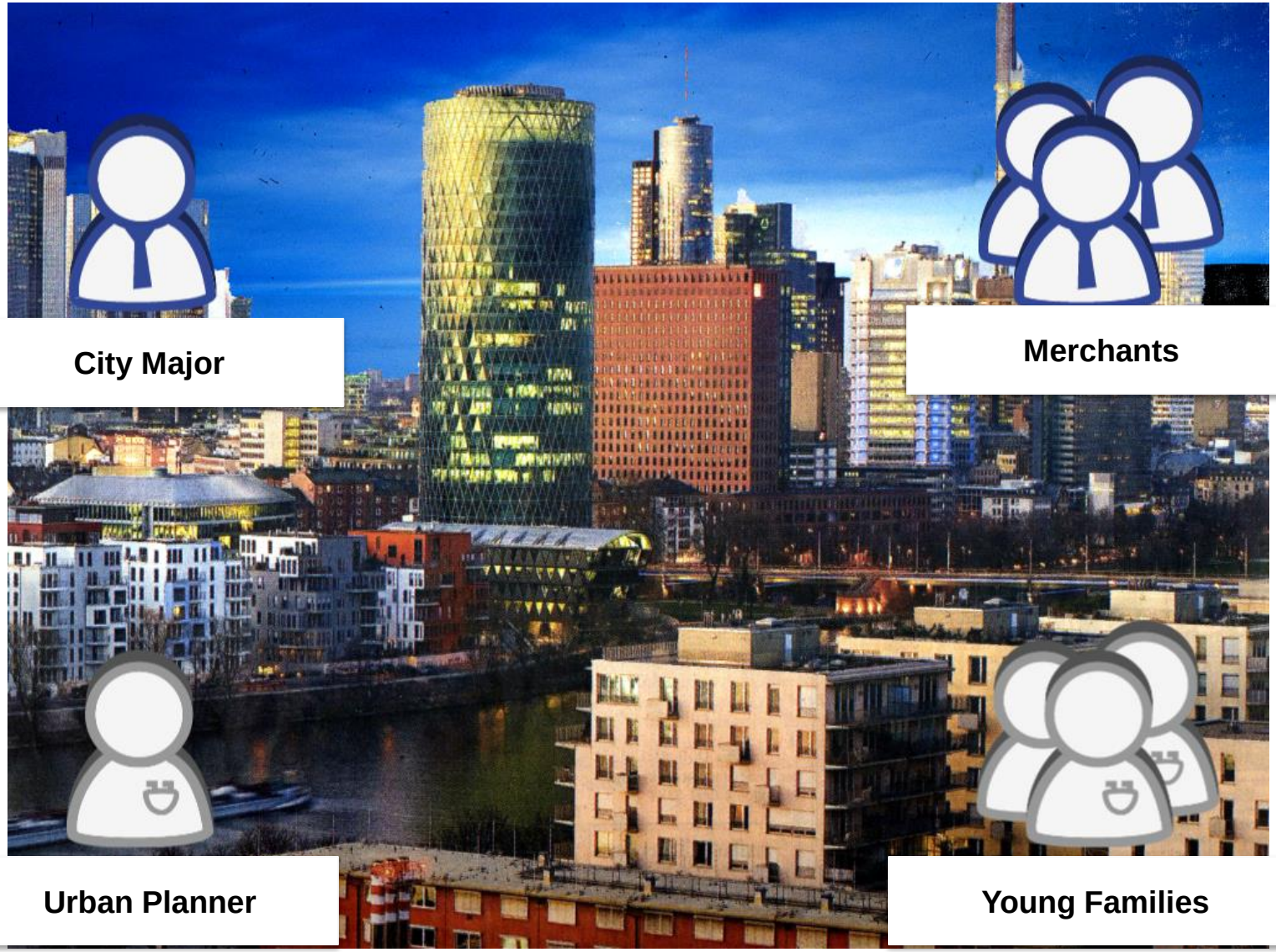
Problems in managing application landscapes

- **IT is covered by fog**
  - Business and management complain low transparency regarding costs and benefit of IT
  - IT projects start with analysis of related systems and their interfaces
  - Recurring surveys to collect information

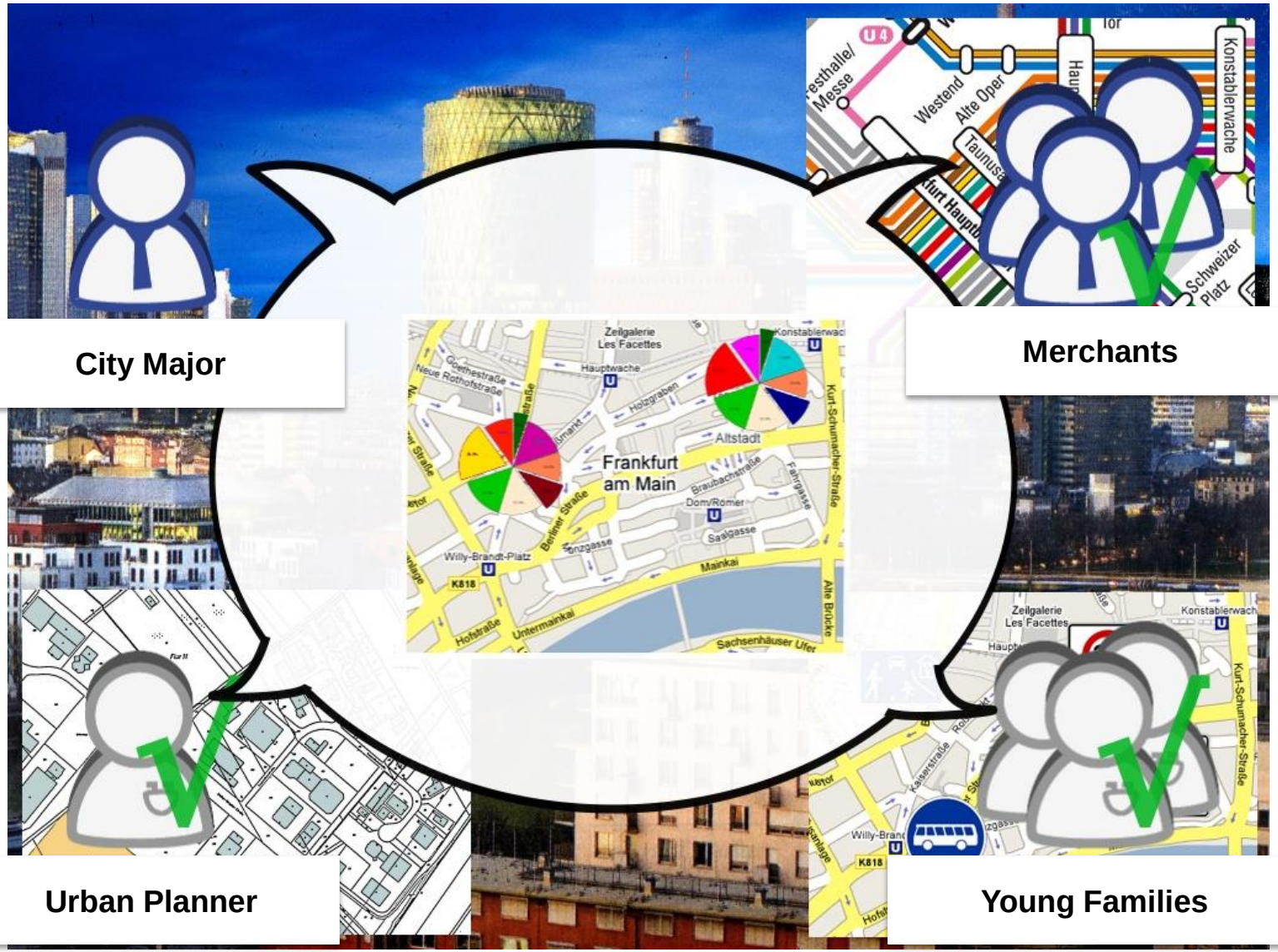
- **Lack of interest of business and management**
  - “IT is not understandable and unnecessary complex”
  - Strategic business goals are not concretized (e.g. „capability maps“)

- **Responsibilities are unclear**
  - No lasting documentation of the **owners** of processes, applications, interfaces, services and domains
  - **Rights** and **responsibilities** for IT and business are not obligatory derived.

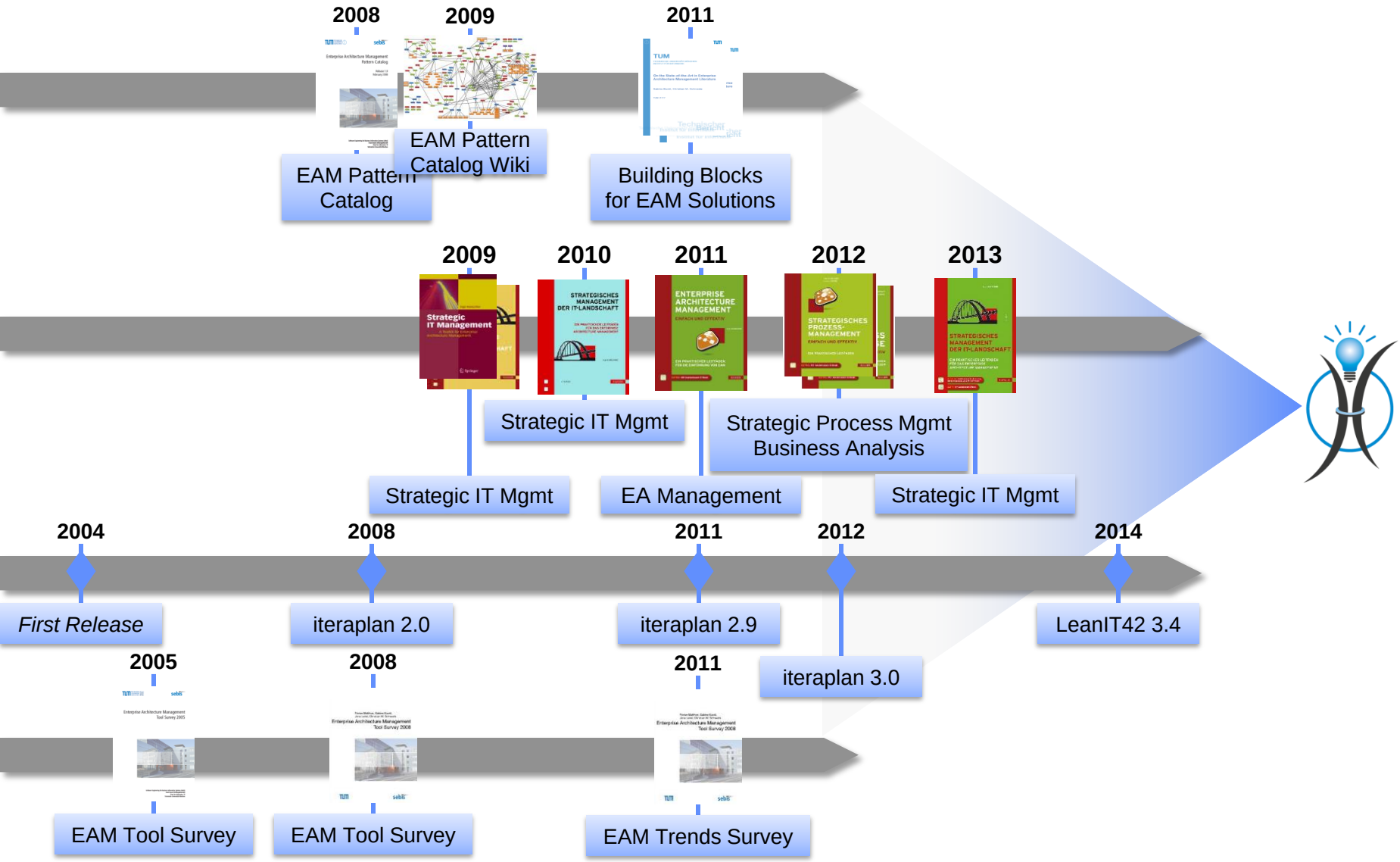
A helpful analogy: Urban planning and management



Maps are established means for communication



Drawing from different sources



Agenda

- Software and system cartography
- Architectural descriptions – the ISO Std. 42010
- Challenges for EAM
- BEAMS – Situational EA management

Software cartography

Ernst Denert-Stiftung für Software-Engineering

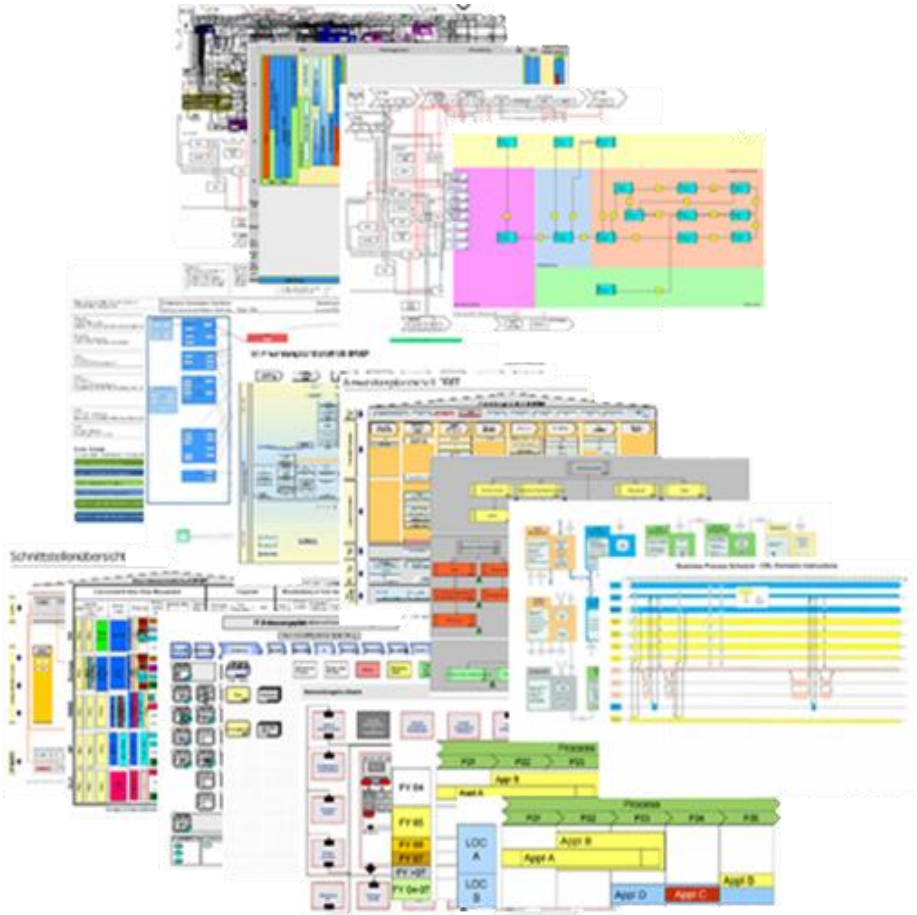
HPI Hasso Plattner Institut

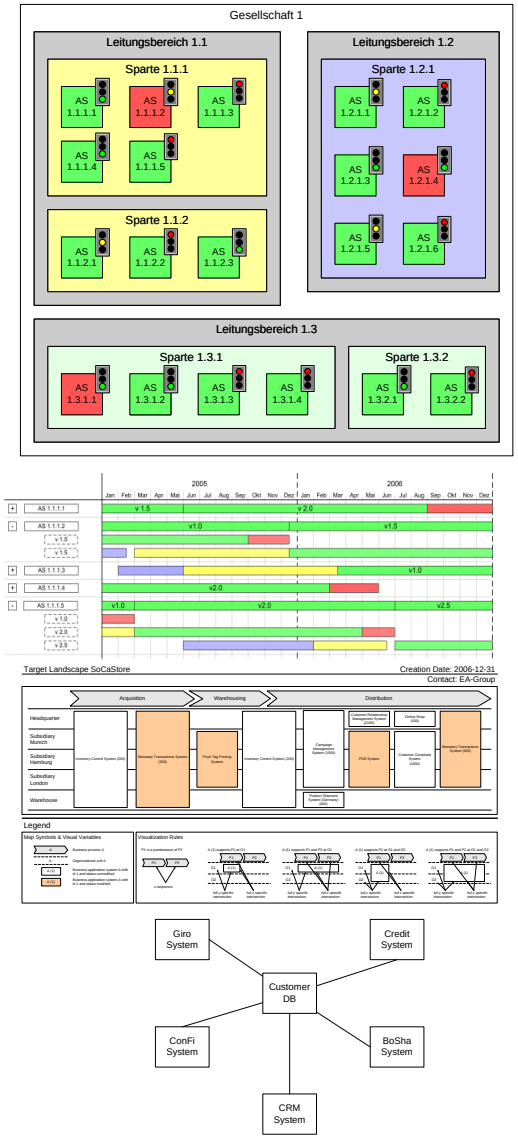
- Software cartography provides models and methods for describing, evaluating and designing application landscapes.

- Models in software cartography
  - Software maps as graphical models of application landscapes
  - Software maps enable automatic generation and maintenance of software maps
  - Focus is on modeling, not painting!
  - Visualizations in accordance with interests of the stakeholders
  - Methods in software cartography
- Documentation of application landscapes with software maps
  - Evaluation of application landscapes with metrics and visualization of these metrics and software maps
  - Modeling of application landscapes with the help of software maps

Consolidating existing software maps found in practice

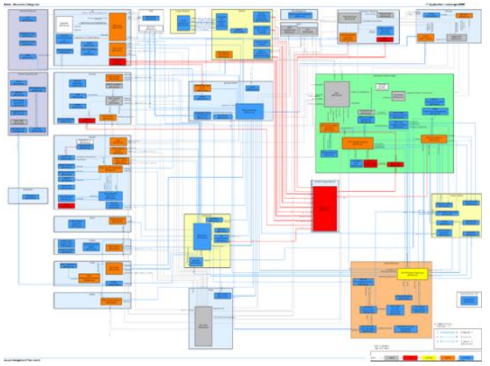
- Numerous interviews with various stakeholders
- Mostly time-consuming, manually created maps

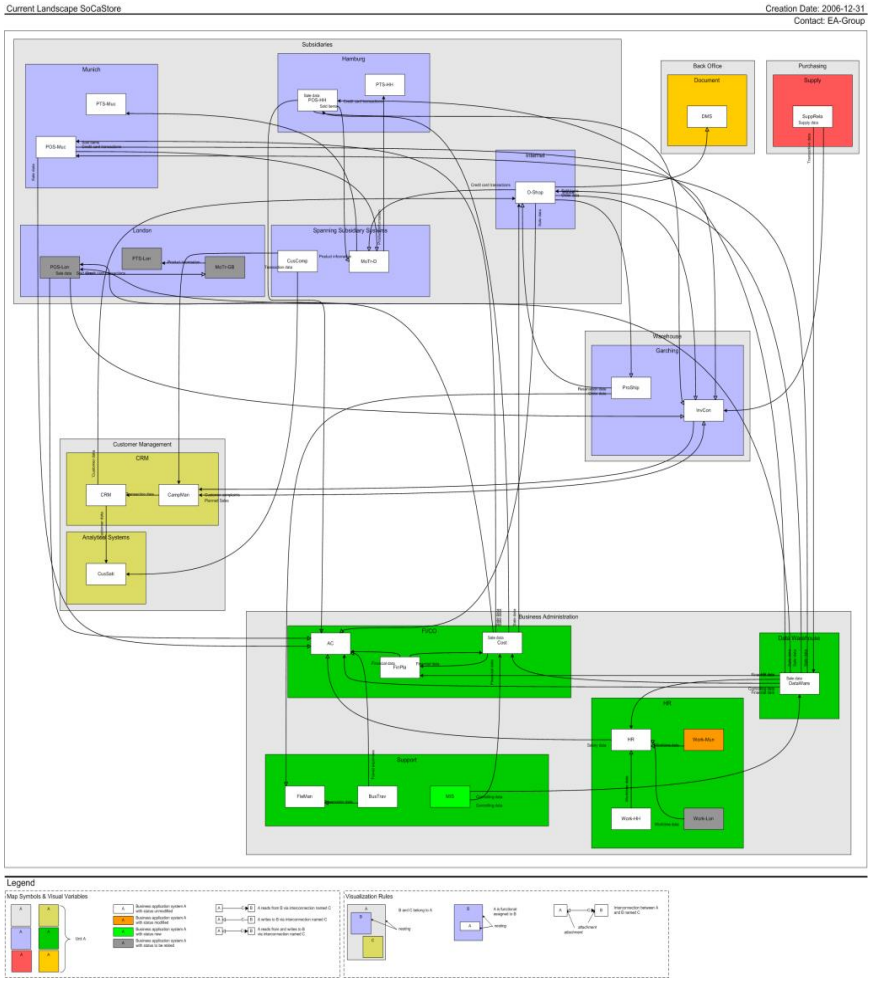




Examples for application landscapes (1)

- Multinational insurance company
- ~160 applications (location Munich, worldwide usage)



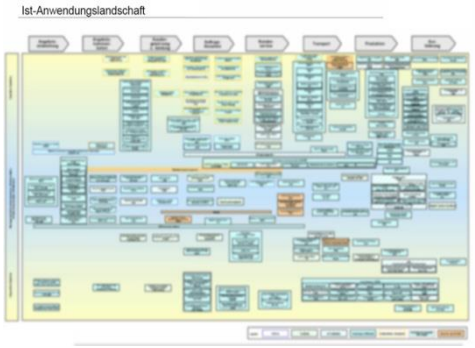


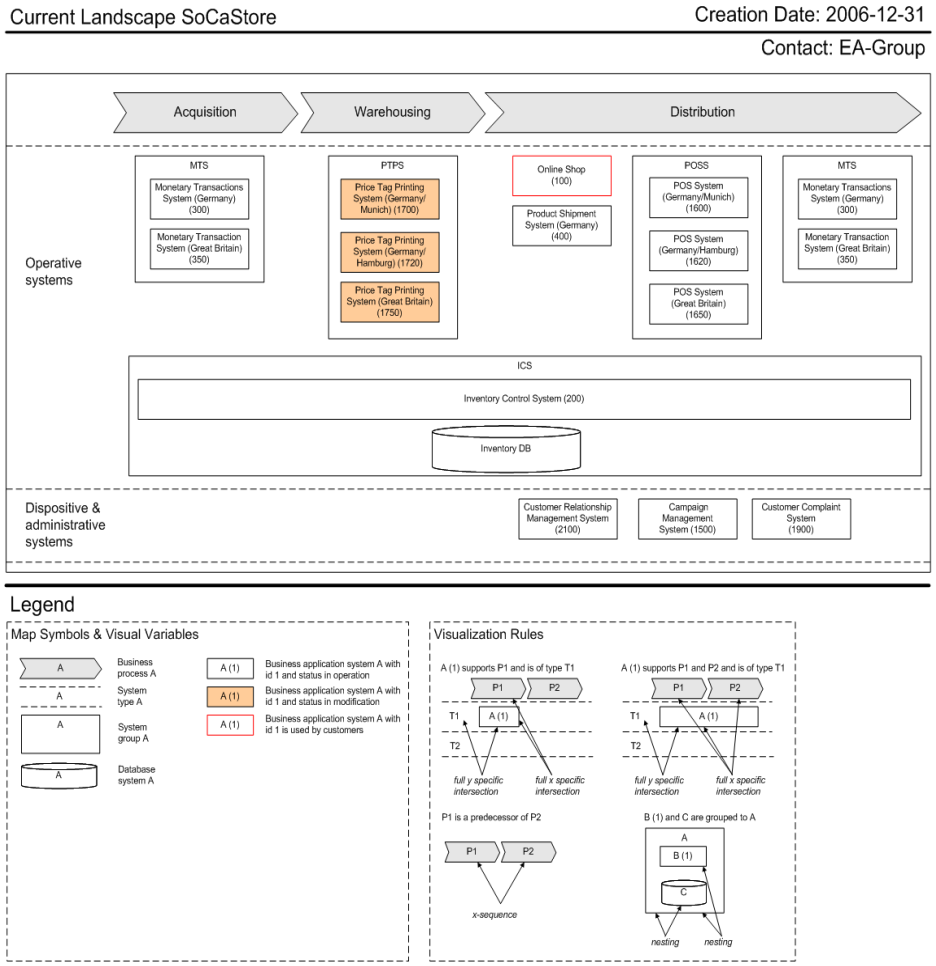
Examples for application landscapes (2)

- Insurance company
- ~150 applications (location Germany, functionally used)

Examples for application landscapes (3)

- Logistics service provider
- ~150 applications (one company division)

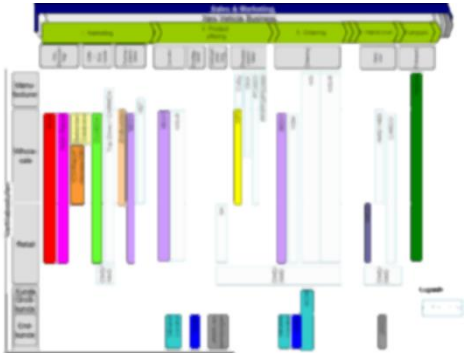




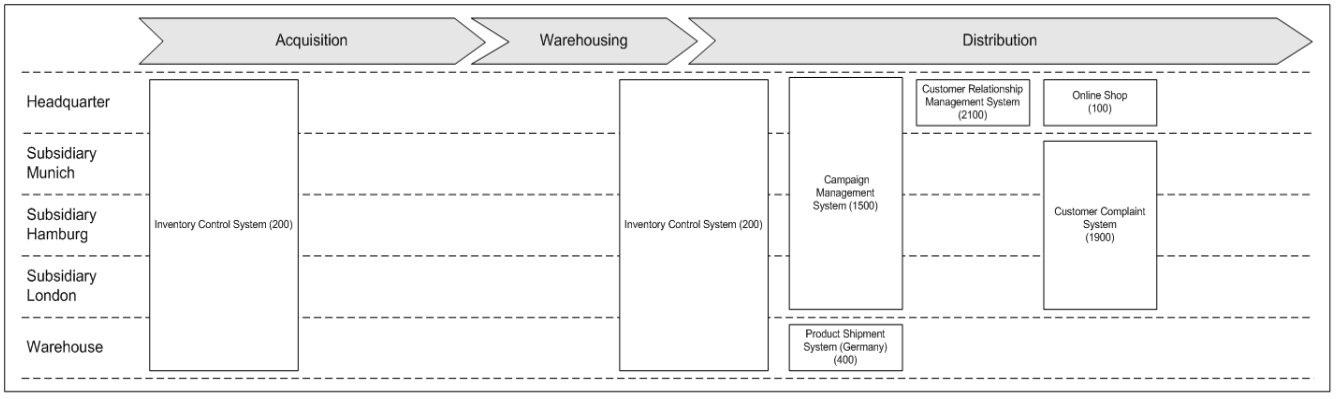
[Wi07]

Examples for application landscapes (4)

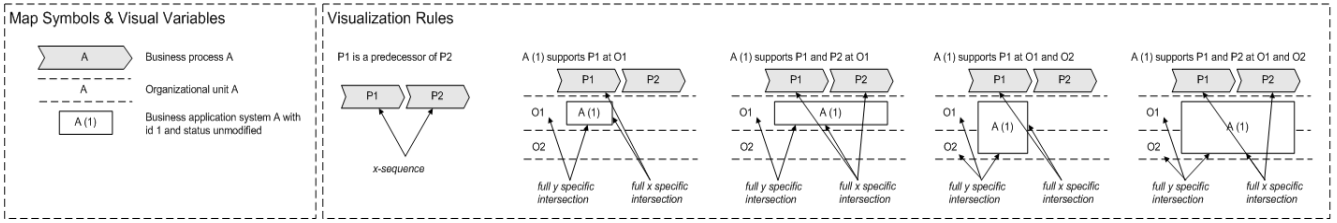
- Automobile manufacturer
- ~2500-3000 applications (worldwide)



Creation Date: 2006-12-31
Contact: EA-Group



Legend

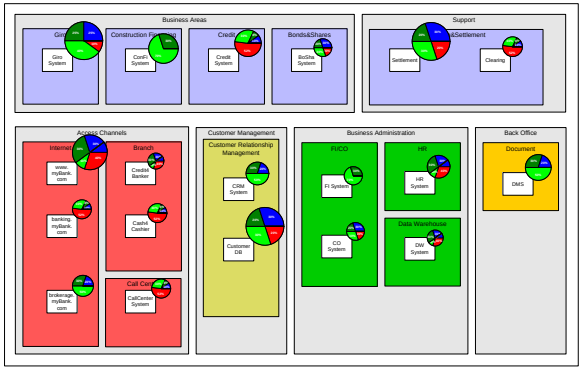


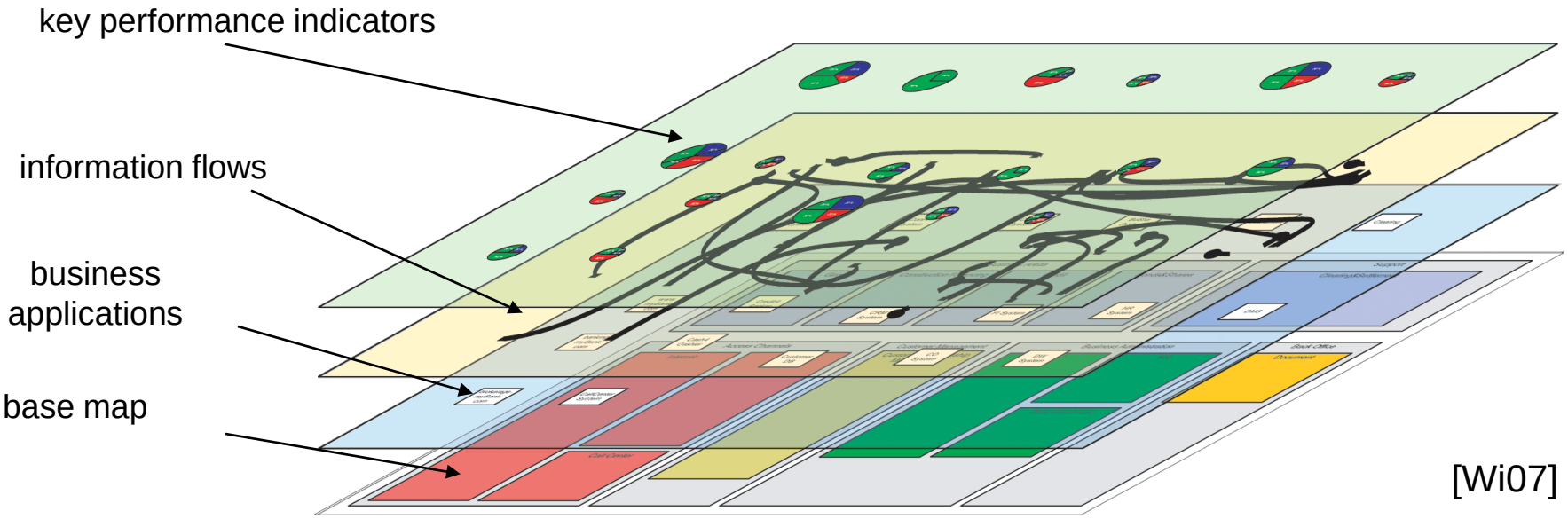
[Wi07]

Principles of software maps illustrated by example

The layering principle

- Problem-specific map type (base map)
- Rule-based layout of visual elements
- Hide / show details based on layers



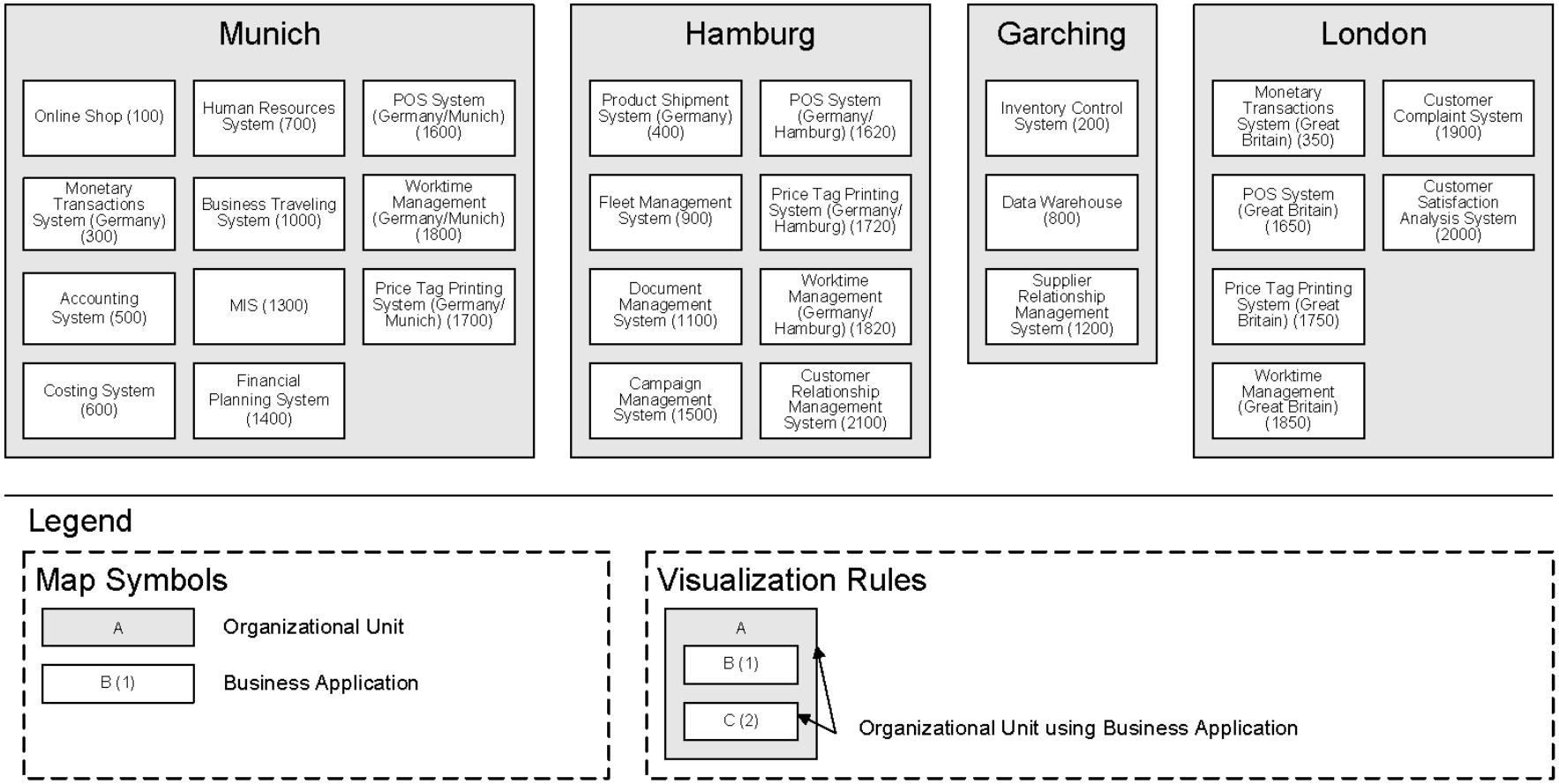


[Wi07]

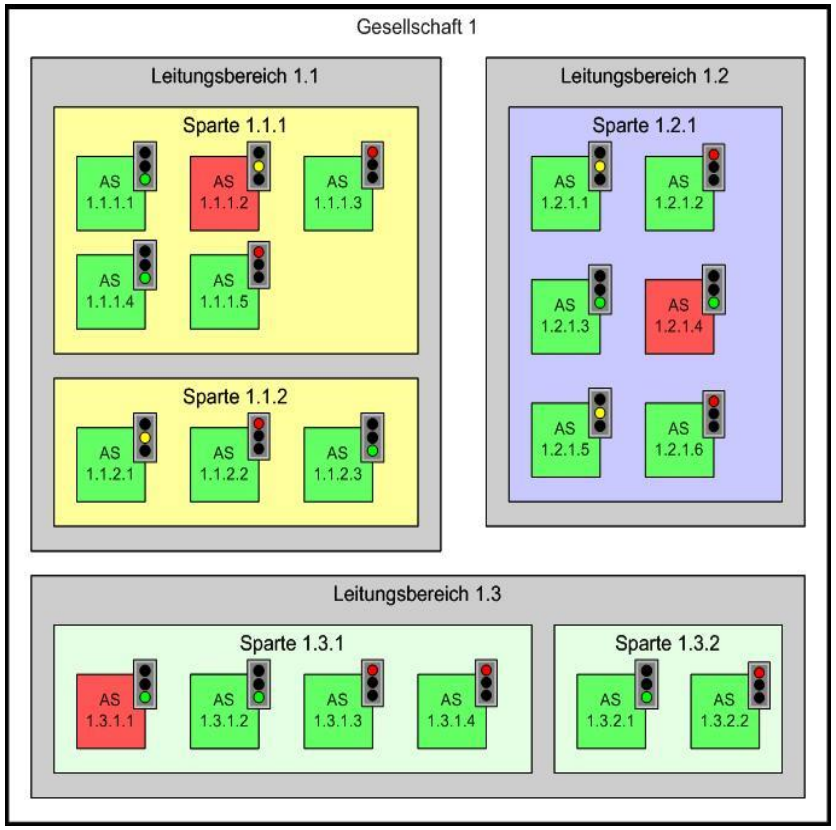
Relevant information for software cartography

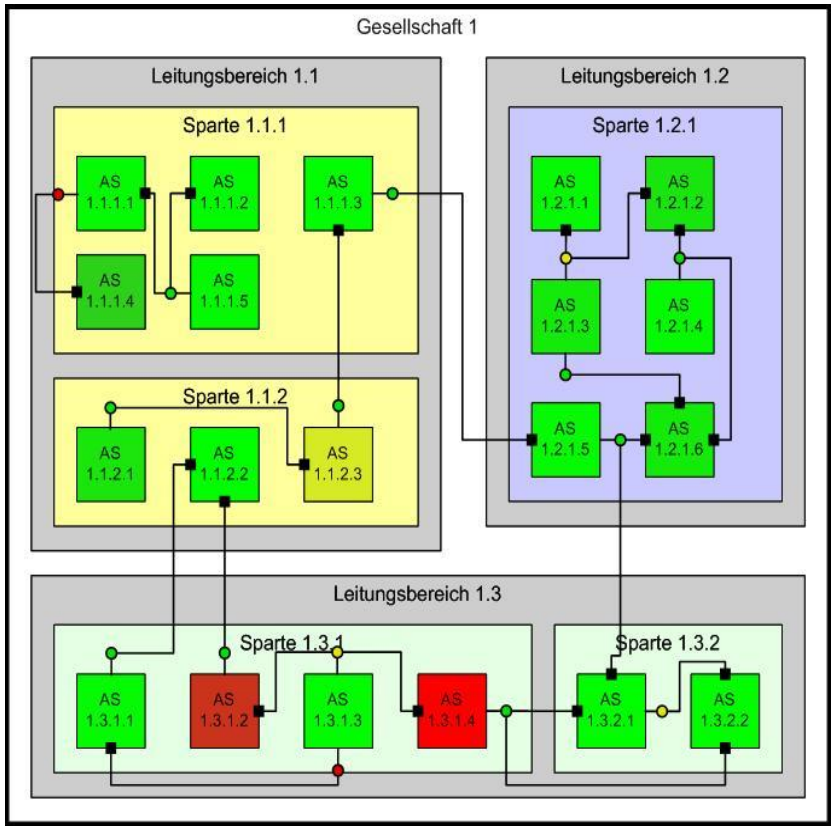
- Central subject of research: (business) applications systems and their environment
- Classification of relevant information
  - Functional:
    Organizational units, business processes, functions, business services, …
  - Plan and strategic aspects:
    Strategies, targets, projects, applications, …
    Lifecycles, versions, …
  - Economical:
    Running costs, maintenance costs, investments, …
  - Technical:
    Interfaces, programming languages, middleware systems, software architectures, …
  - Operative:
    uptime/downtime of systems, dependencies, (geographic) locations,…

Information visualization on software maps (1)



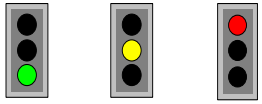
Information visualization on software maps (2)





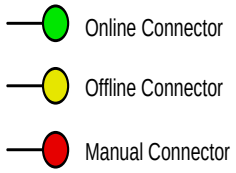
Application does not conform to architectural standards

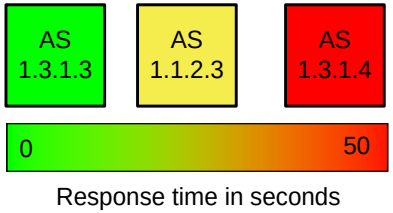
Application conforms to architectural standards



> 99,5 % // 99,5 – 99,0 % // <99,0 %

Availability per day

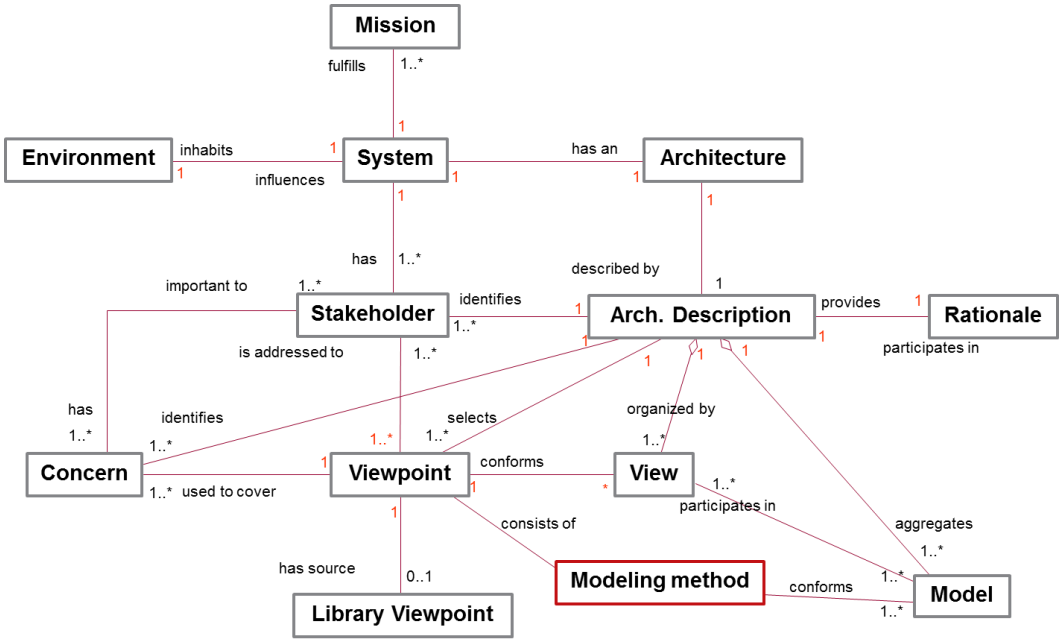




Response time in seconds

Agenda

- Software and system cartography
- Architectural descriptions – the ISO Std. 42010
- Challenges for EAM
- BEAMS – Situational EA management



ISO Std 42010: Recommended practice for architectural description of software-intensive systems

Scope

- Software-intensive systems
- Individual systems
- „Systems of systems“ (also application landscapes, enterprise architectures)

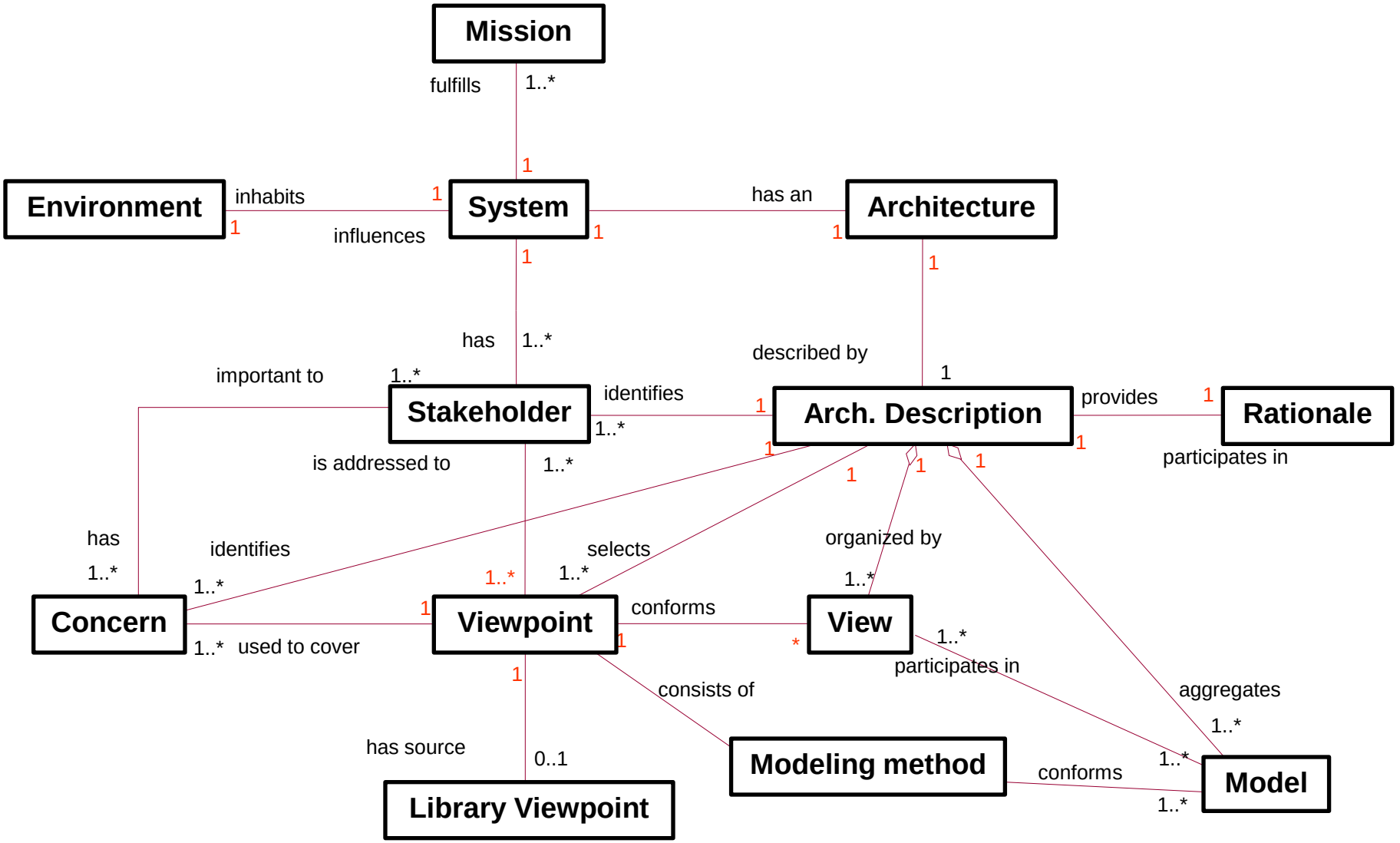
Goals

- Supports documentation, explanation, and communication of architectures.
- Does not provide a graphical notation nor defines any conformance of systems, projects, organizations, processes, methods, or tools
- Defines notions in the context of architectural description – how to describe an architecture

Architecture framework

Predefined set of concerns, stakeholders, viewpoints, and viewpoint correspondence rules; established to capture common practices for architecture descriptions within specific domains or user communities

Conceptual model of architectural descriptions according to the ISO Std 42010



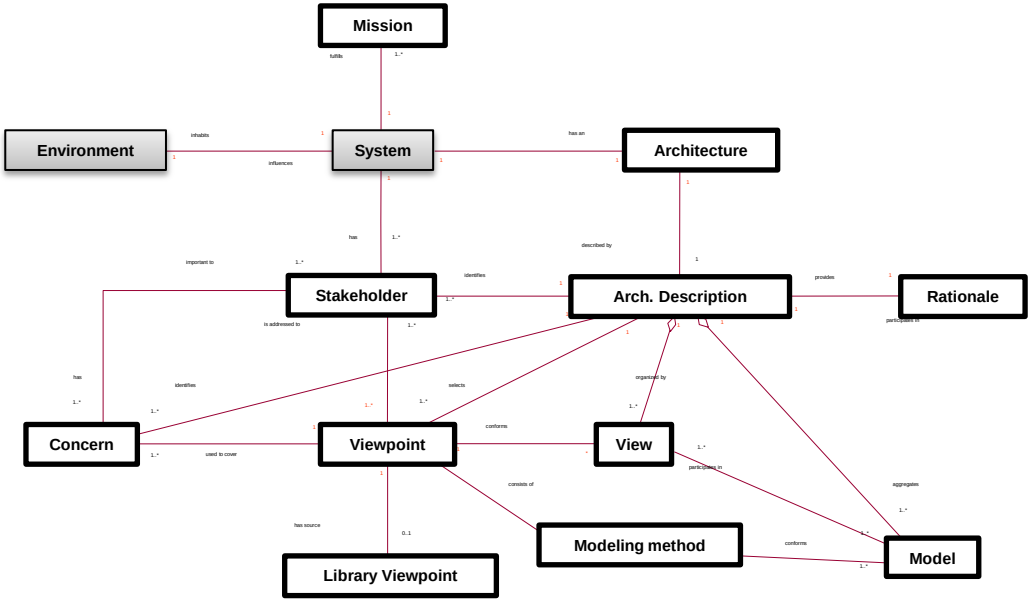
Red ➔ Adaptations to the ISO Standards, e.g. multiplicities are inserted according to the description in the standard

[IS07]

Notions: System and environment

System
A collection of components organized to accomplish a specific function or set of functions.

Software-intensive
Software contributes essential influences to the design, construction, deployment, and evolution of the system as a whole.



Environment
Environment or context, which exerts influence on a system's design. This comprises also other systems interacting with the latter one. The environment determines settings and circumstances of developmental, operational, political, and other influences upon that system.

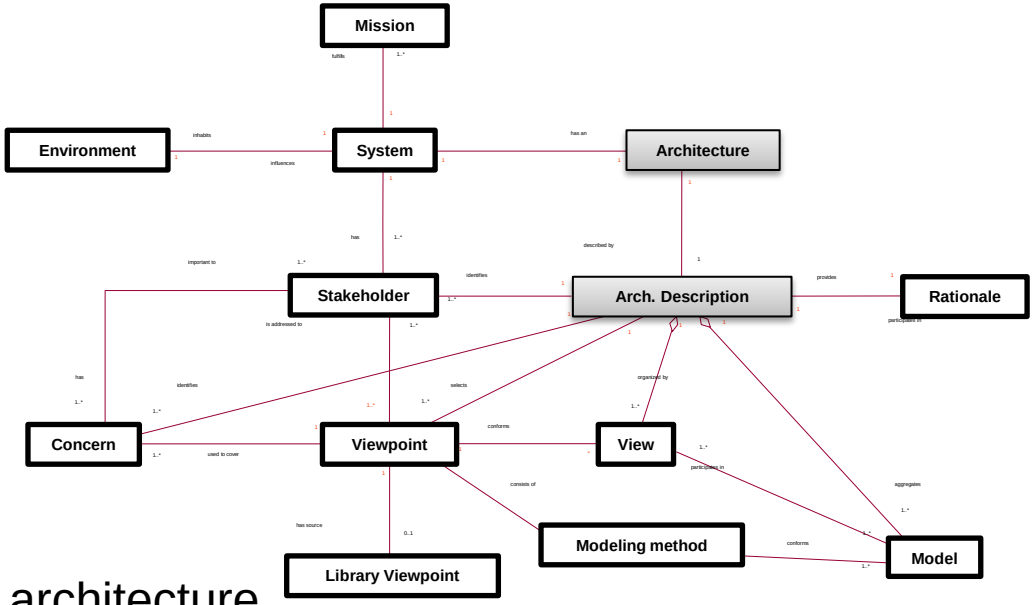
➔ **Delimitation between the system and its environment**

Example: Apple's iTunes store

- System
  - iTunes Store Server
  - …
- Environment
  - client-PCs of the customer
  - …

Notions: Architecture and architectural description

Architecture
Fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.



Architectural description
Collection of products to document an architecture.

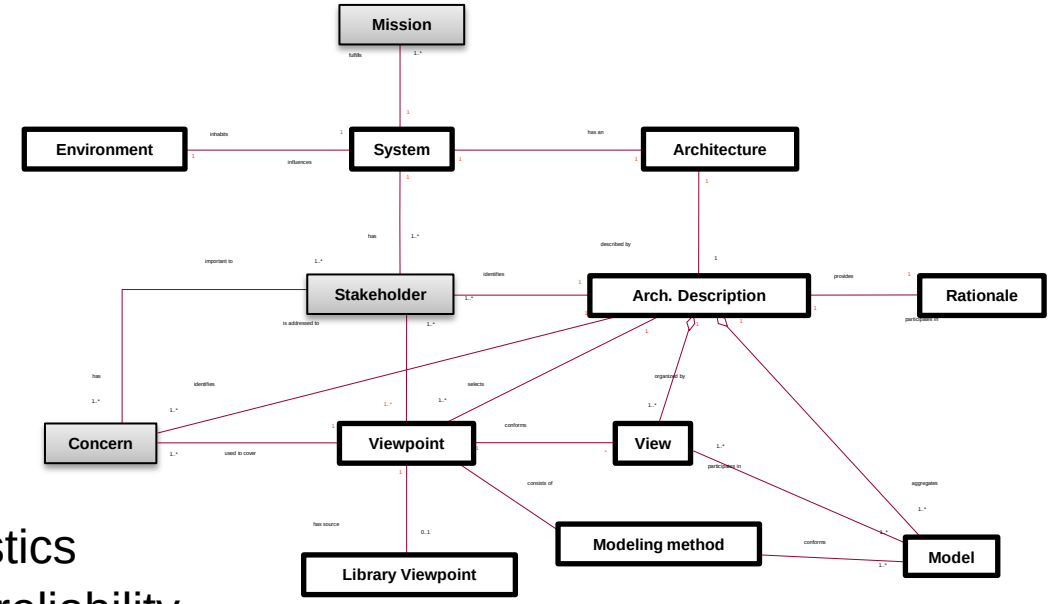
An architectural description selects one or more viewpoints for use. The selection viewpoints typically will be based on consideration of the stakeholders to whom the architectural description is addressed and their concerns.

➔ **Every system has an architecture, whether understood or not; whether recorded or conceptual.**

Notions: Concern, stakeholder, and mission

Stakeholder

Individual, team, or organization (or collections thereof) with interests in, or concerns relative to, a system.



Concern

Those stakeholders' interests, which pertain to the development, operation, or other key characteristics of the system (e.g., performance, reliability, security, evolvability, distribution, …)

Mission

Use or operation for which a system is intended by one or more stakeholders to meet some set of objectives.

➔ **The architectural description has to be aligned with the stakeholders' concerns.**

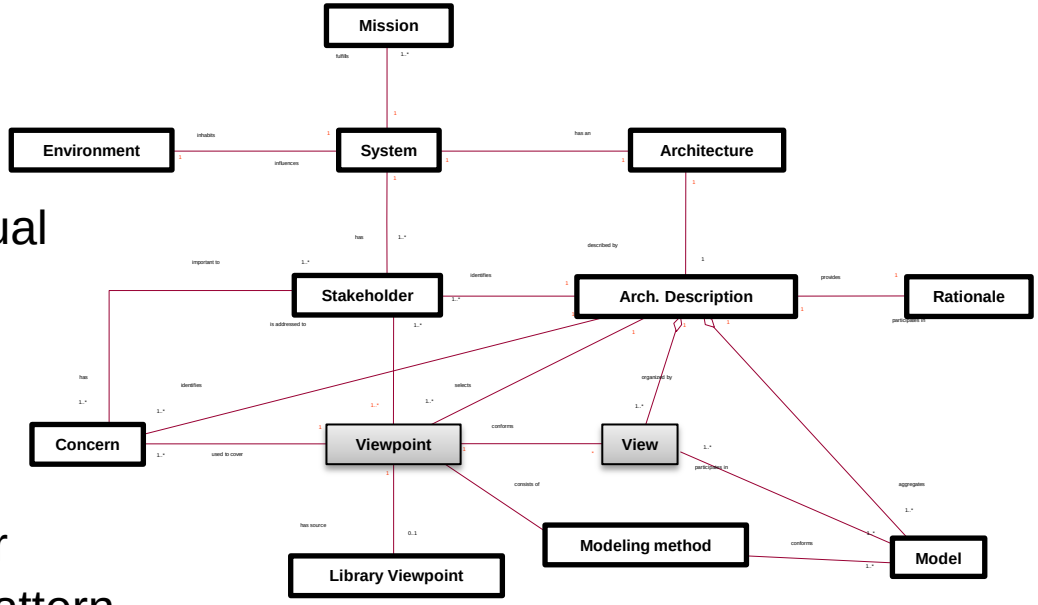
Example: Apple's iTunes store

- Mission
  - Profitable sales of music, videos, and applications by means of an internet platform
  - …
- Stakeholder and concerns
  - Management of the iTunes store Germany
  - Responsible for operating and maintaining the website
  - …

Notions: Viewpoint and view

View

Representation of a whole system from the perspective of a related set of concerns. Views are the actual description of the system



Viewpoint

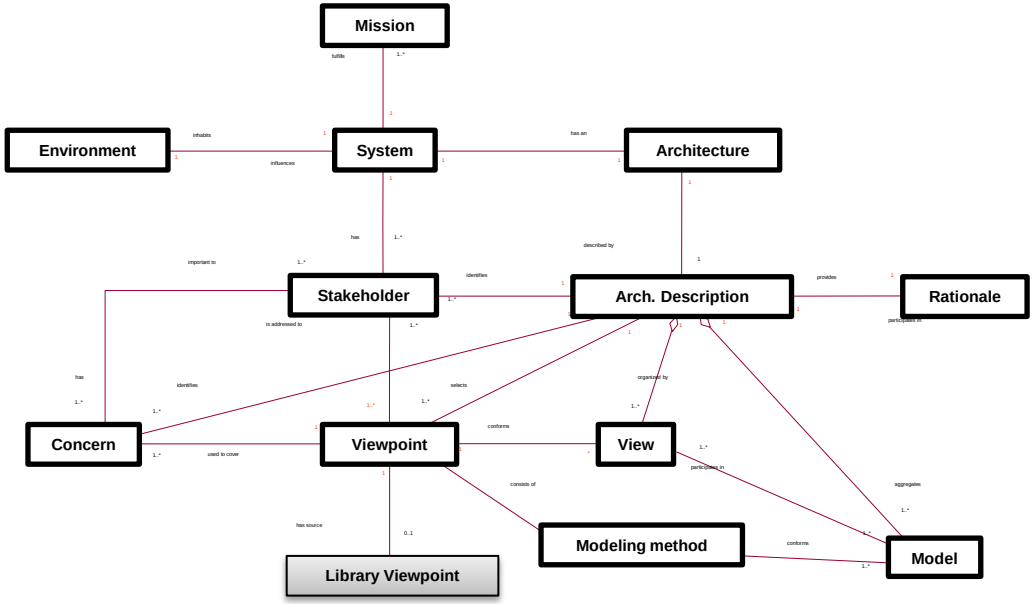
Specification of the conventions for constructing and using a view. A pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis.

➔**Separation between viewpoint and view**

Notions: Library viewpoint

Library viewpoint:

Viewpoint-definition from literature.

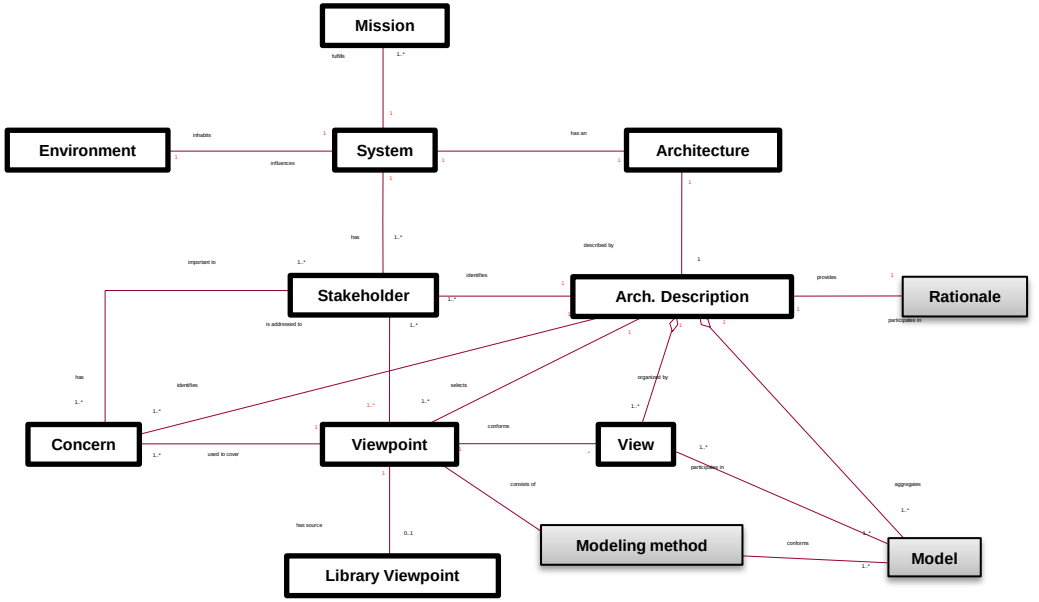


➔ **Reuse of techniques and notations for architectural descriptions in order to avoid ad-hoc notations for "boxes-and-lines everywhere viewgraphs"**

Notions: Rationale and model

Rationale

Describes the reasons, leading to the selection of an architecture as well as the intention an architect pursues with his decisions.



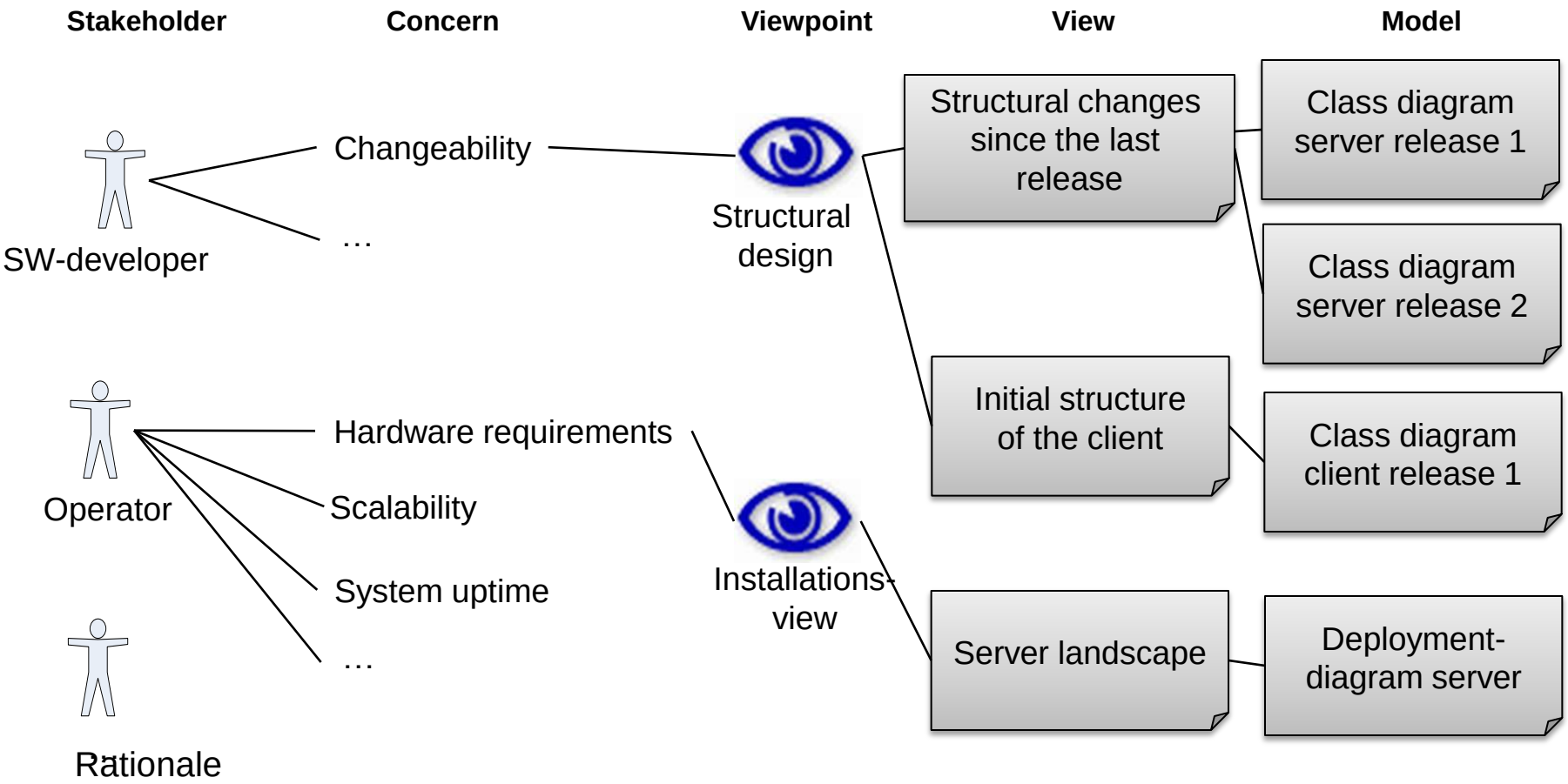
Modeling method

Specification of the conventions for constructing and using a model. The modeling method determines the language to be used to describe the model.

Model

Represents a certain aspect of an architecture, according to a notation defined through a viewpoint.

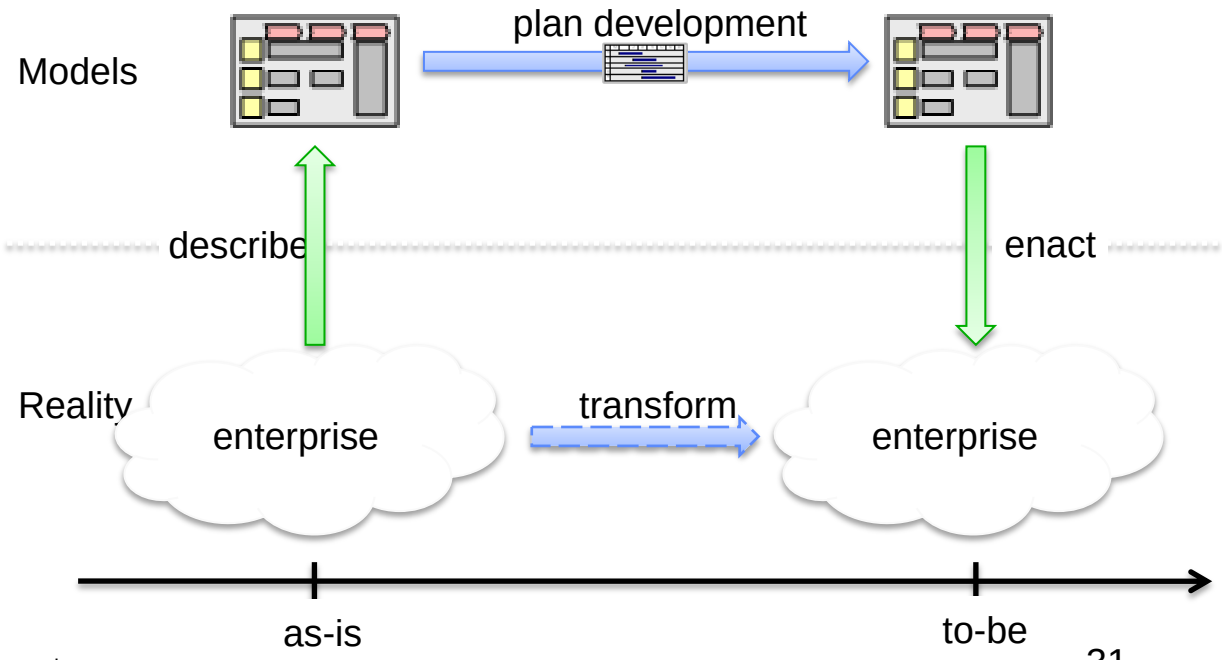
Example: Apple‘s iTunes store



- Rationale
  - Ease of use for the customer
  - It shouldn’t be possible for customers to download registered video and music material without paying it
  - …

Agenda

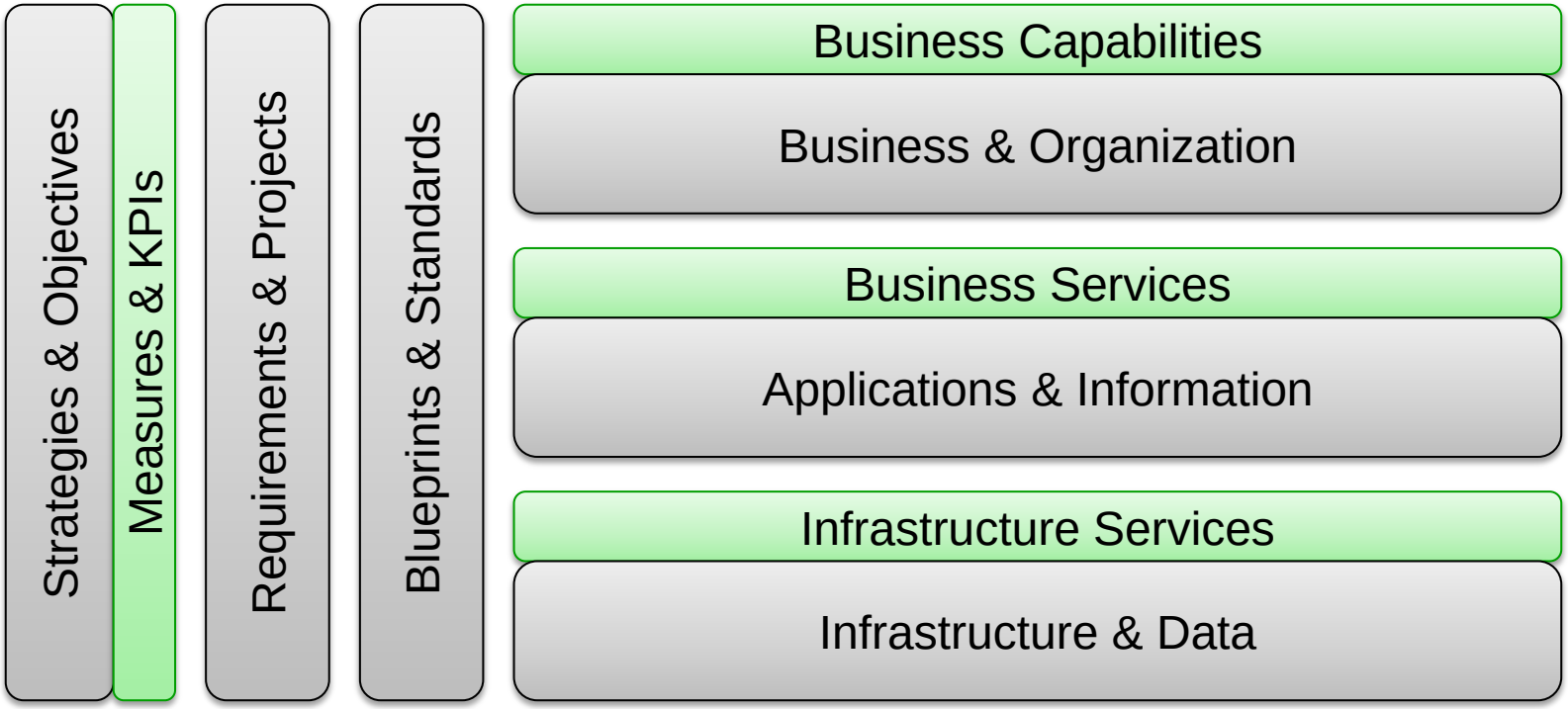
- Software and system cartography
- Architectural descriptions – the ISO Std. 42010
- Challenges for EAM
- BEAMS – Situational EA management



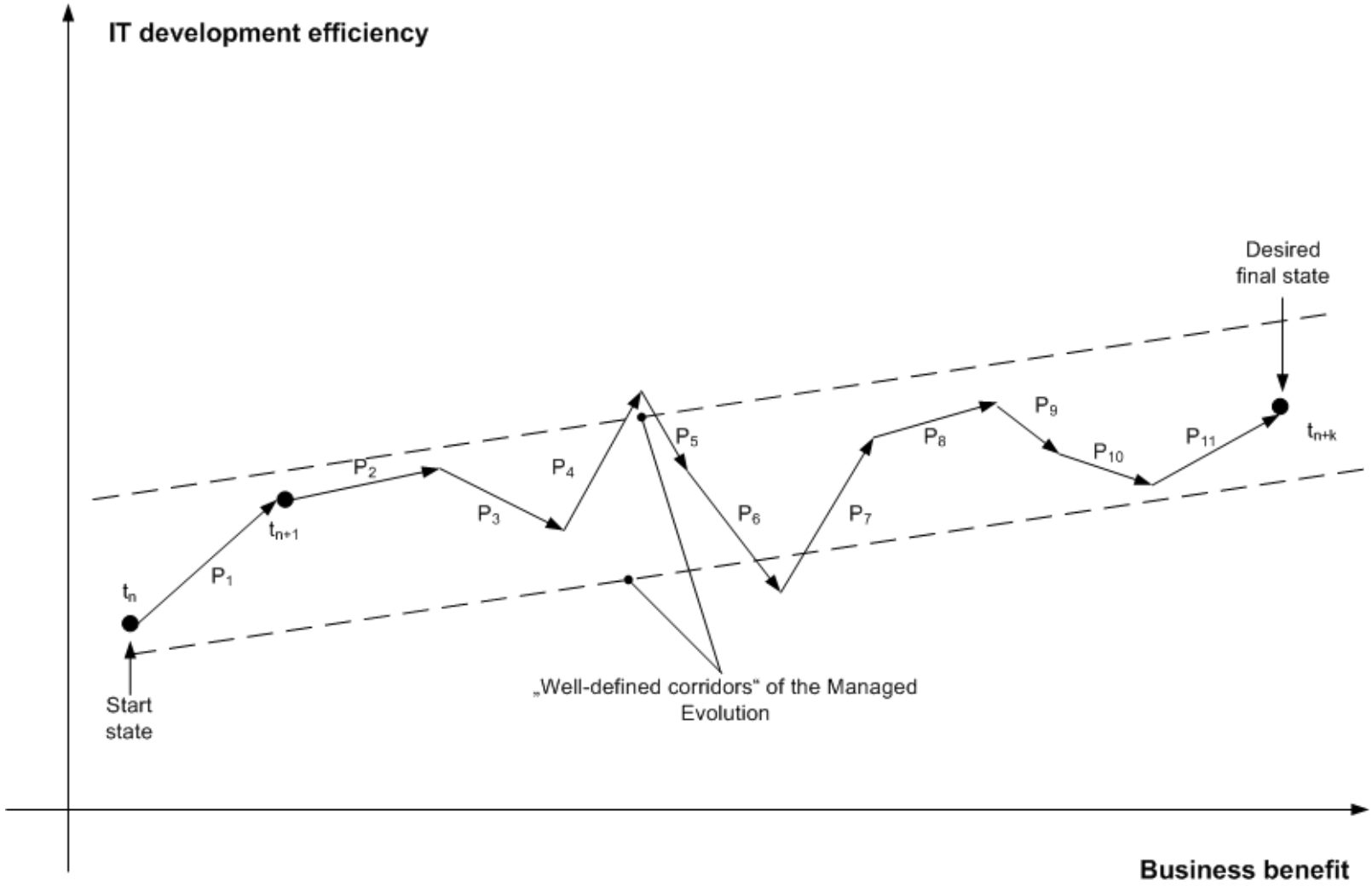
From application landscapes to Enterprise Architectures – a holistic perspective

Fundamental organization of a system [enterprise] embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution. [IS07]

- consists not only of IT but also of business aspects
- can be divided into layers and crosscutting functions



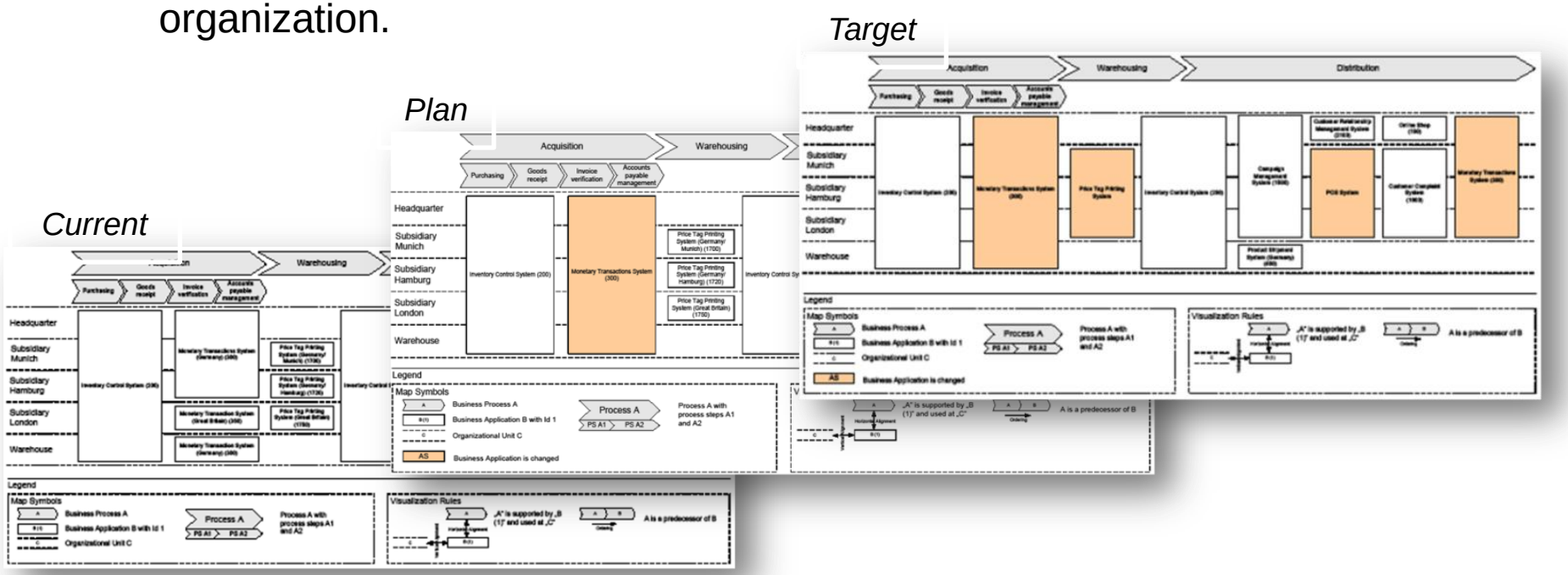
Evolution trajectory of managed evolution



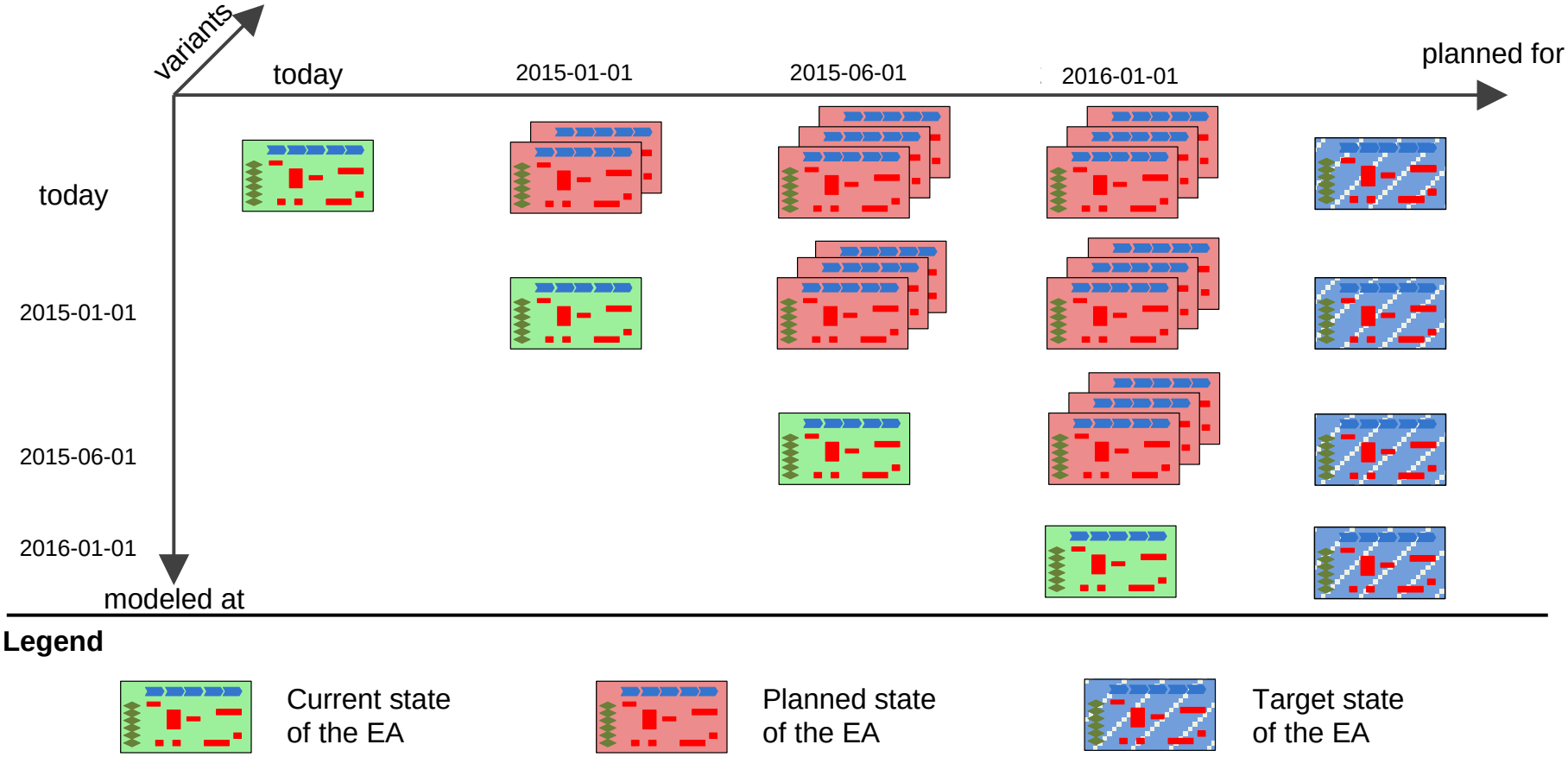
[Mu08]

EAM uses three EA models

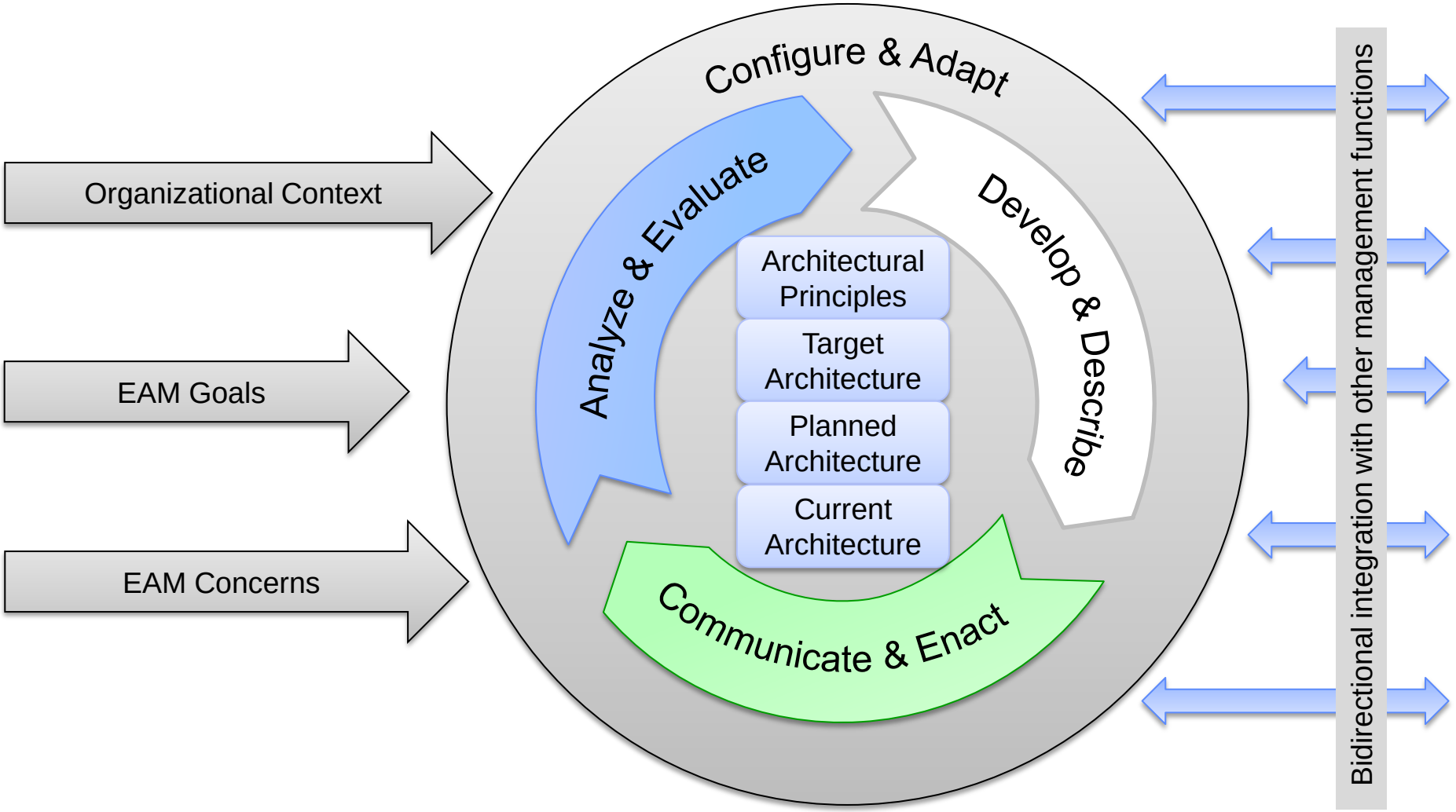
- A **current** (as-is) state of the EA reflects the actual architecture (status quo) at a given point in time.
- A **planned** state of the EA is derived from planned and budgeted projects for transforming the EA until a certain point in time.
- A **target** (to-be, envisioned) state of the EA describes an ideal state to be pursued according to the strategies and architectural principles of the organization.



Every software map has time-references

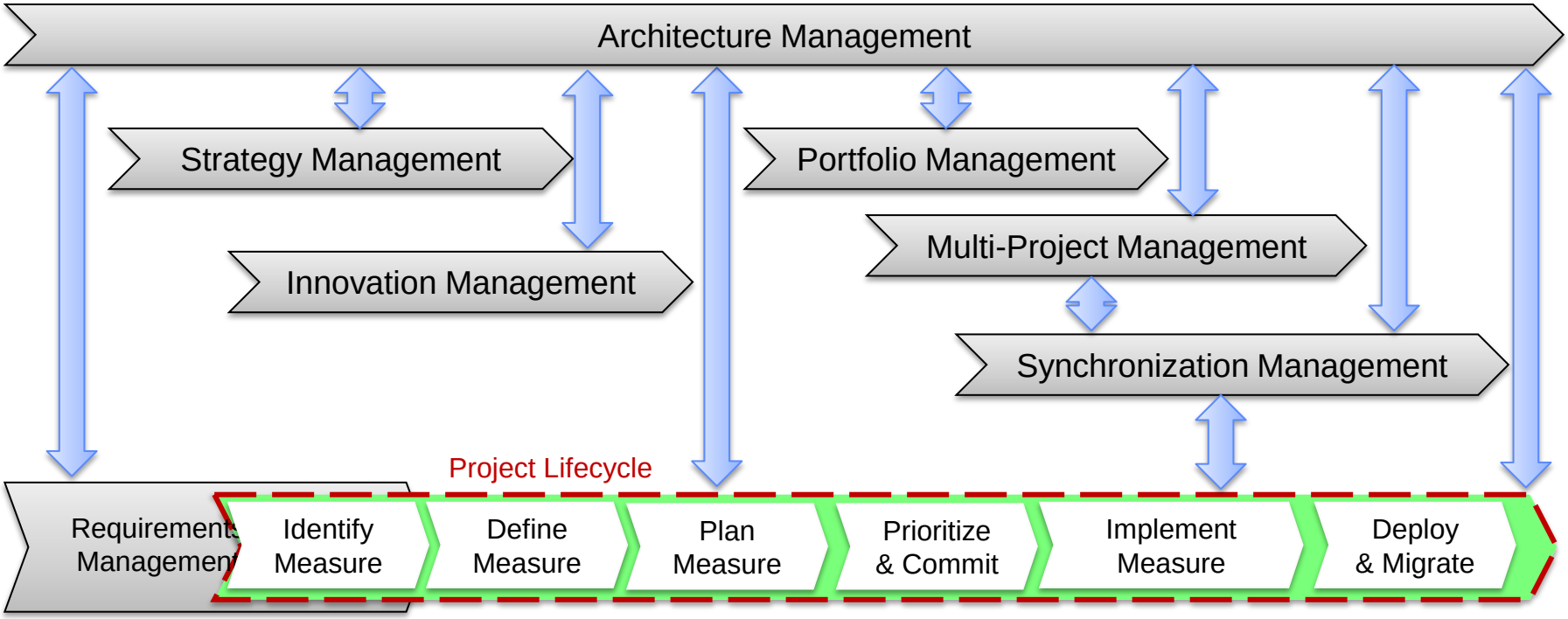


Challenges for EA management – Address the organizational specificities



Challenges for EA management – Integration with other management functions

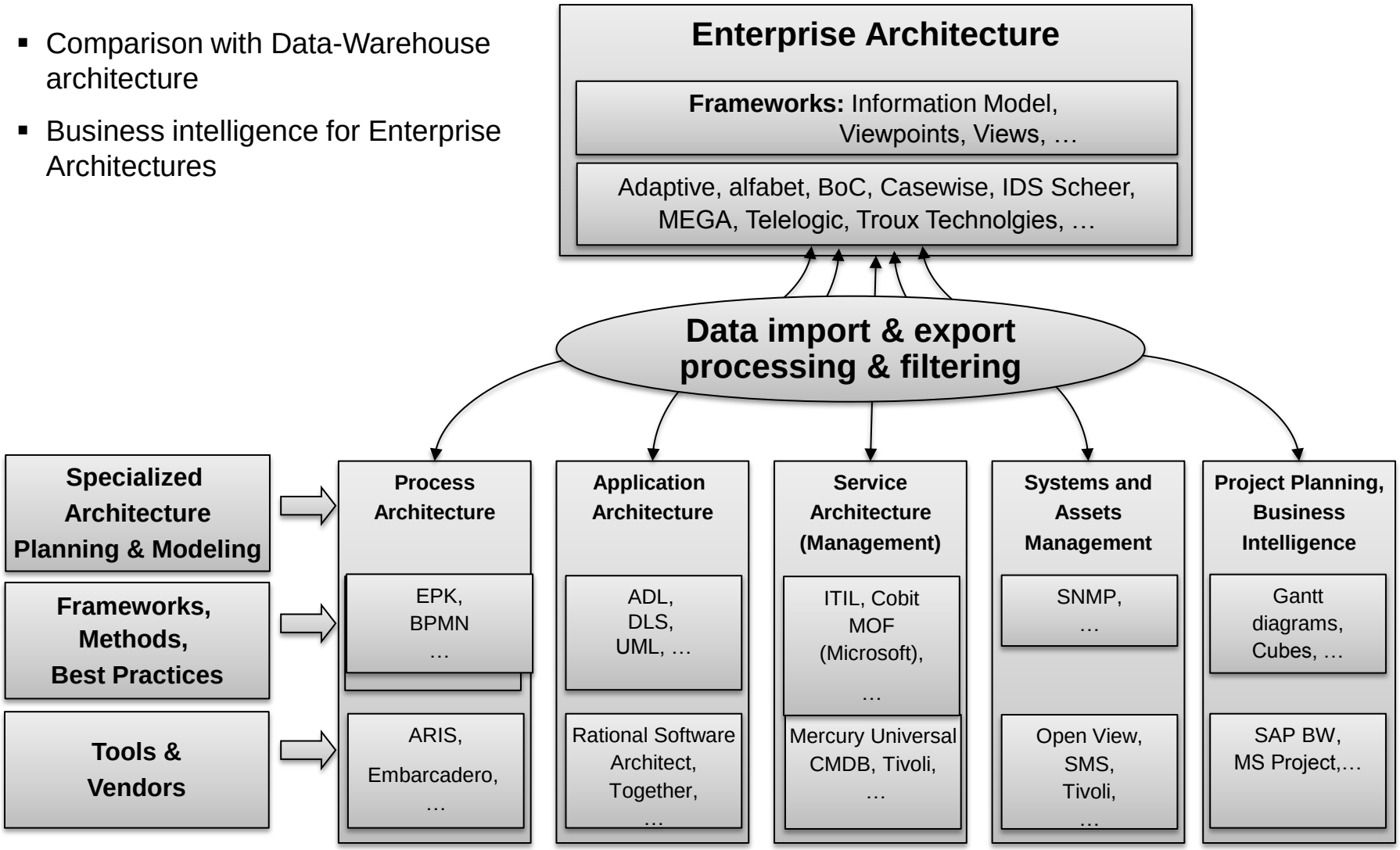
- Example of a mature organization



- All architectural changes are performed through **projects.**
- EA management has to be integrated in the project lifecycle.
- EA management has to exchange information with other enterprise-level management functions

Challenges for EA management – Integration of different information sources

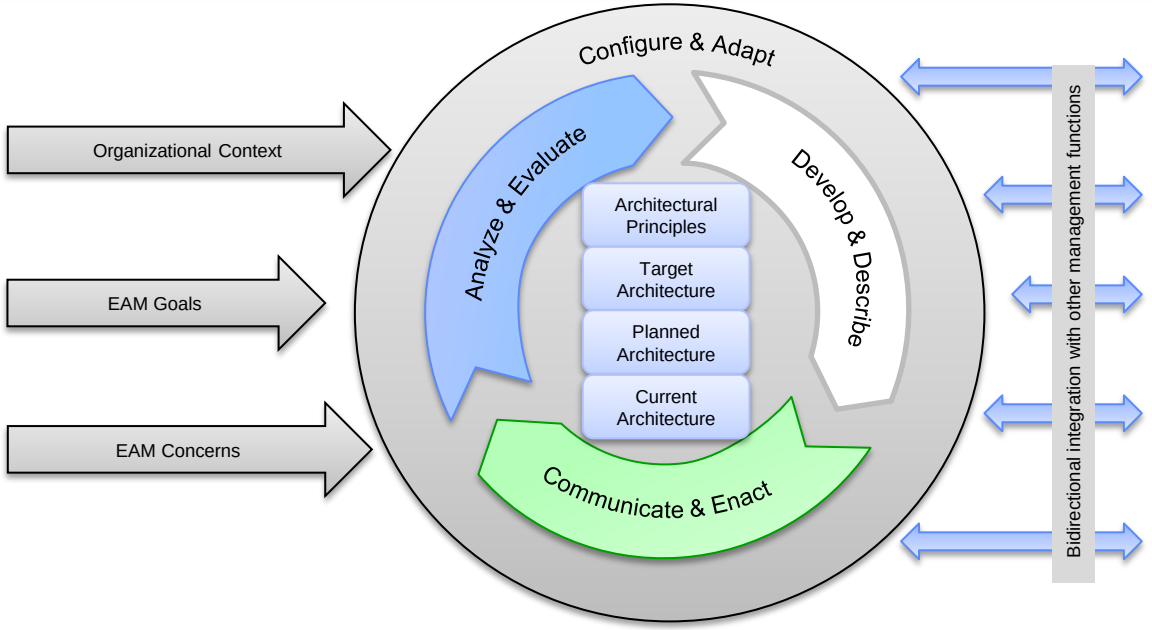
- Comparison with Data-Warehouse architecture
- Business intelligence for Enterprise Architectures



[Ma08]

Agenda

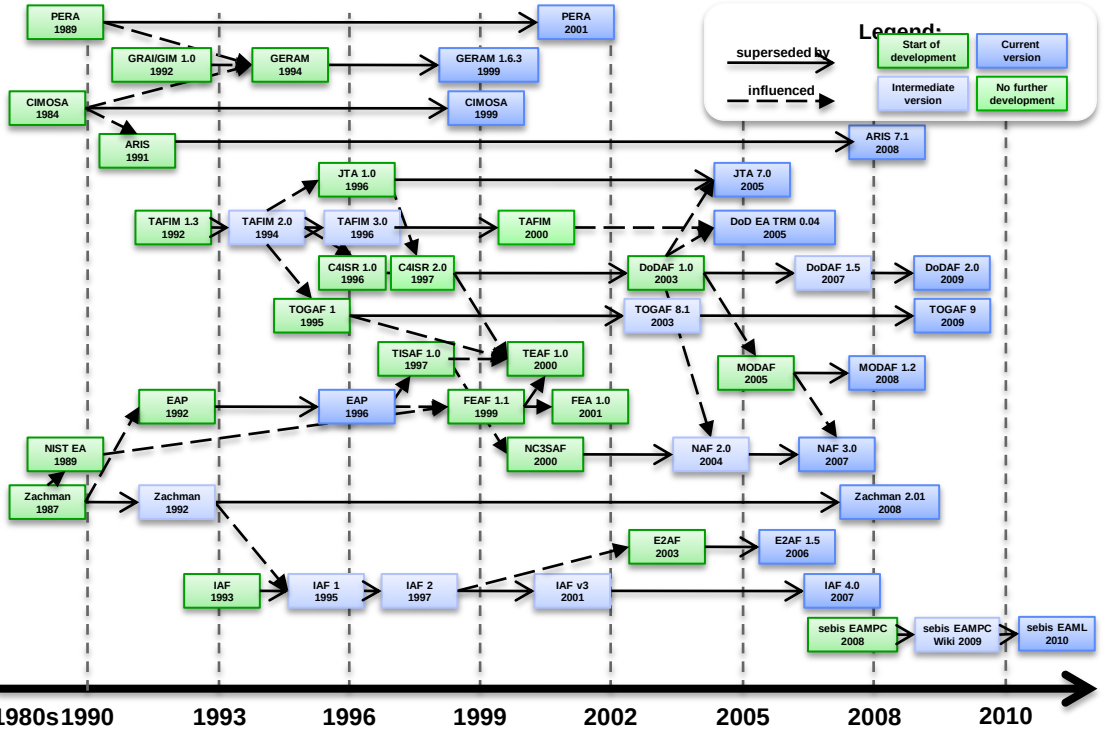
- Software and system cartography
- Architectural descriptions – the ISO Std. 42010
- Challenges for EAM
- BEAMS – Situational EA management



How to start?

- 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



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

Alexander et al. [Al77] (**Architecture**)

> 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.

Buschmann et al. [Bu96] (**Software Architecture**)

> A pattern for software architecture describes a particular recurring design problem that arises in specific design contexts, and presents a well-proven generic scheme for its solution. The solution scheme is specified by describing its constituent components, their responsibilities and relationships, and the ways in which they collaborate.

Gamma et al. [Ga94] (**Software Engineering**)

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

Enterprise architecture management patterns

An enterprise architecture management pattern (**EAM pattern**) is

- a general, reusable *solution* to a common *problem*
- in a given *context*,
- identifies driving *forces*,
- *known usages* and
- *consequences*.

An EAM pattern takes a holistic perspective:

It address problems at the *enterprise* (systems of systems) *level*.
It considers *social*, *technical* and *economic* forces in a balanced manner.

It is *discovered* in working solutions rather than being invented or hoped for.
It uses a *clear*, *accessible* and *informal* language that allows practitioners to describe their knowledge and experience.

[Bu08,Er10]

A catalog of interrelated EAM patterns

- Tailor the EAM function to the specific situation (*context* and *problem*) of the enterprise and follow an incremental strategy based on **EAM patterns** representing proven practices.

- Systematically document the dependencies between

  - methodology patterns (**M-Pattern**),
    Which processes and roles are required to address a problem?

  - viewpoint patterns (**V-Pattern**), and
    Which viewpoints help stakeholders to collaboratively perform the activities?

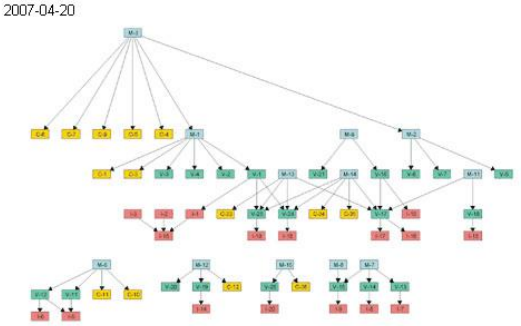
  - information model patterns (**I-Pattern**)
    Which information has to be available to generate a view?

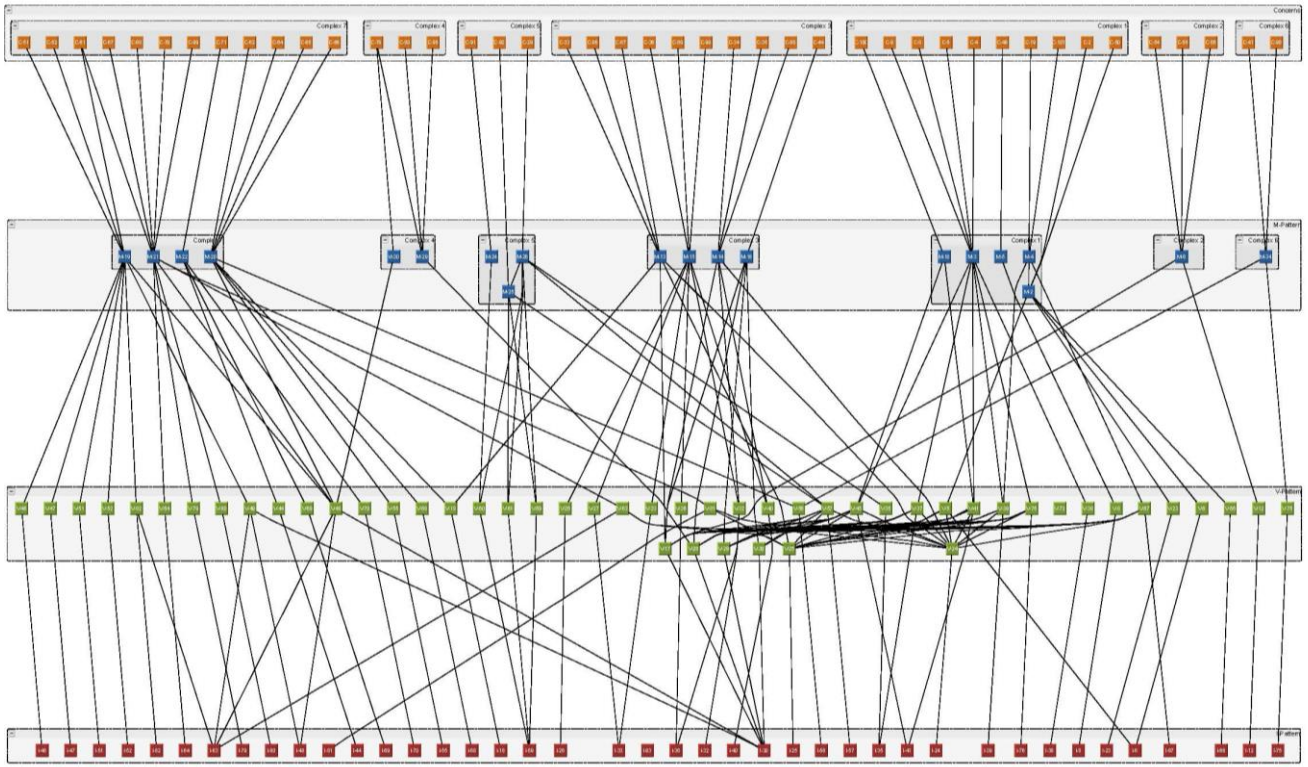
- Draw attention to the consequences implied by a pattern (labor, required information, *political* resistance, …)

Overview of the pattern catalog version 1.0

Basis: literature, experience from *sebis* research projects, structured interviews of 25 enterprise architects
Selection based on relevance and adoption by an extensive online questionnaire

➔ 43 concerns, 20 M-Patterns, 53 V-Patterns, and 47 I-Patterns





Selected literature

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

[Bu96] Buschmann, F.; Meunier, R.; Rohnert, H.; Sommerlad, P.; Stal, M.: Pattern-oriented software architecture: a system of patterns. John Wiley & Sons, Inc., USA, 1996

[Bu08] Buckl, S.; Ernst, A.; Lankes, J.; Matthes, F.: *Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008).* Technical Report TB0801, Technische Universität München, Chair for Informatics 19 (sebis), 2008.

[Bu11] Buckl, S.; Developing Organization-Specific Enterprise Architecture Management Functions Using a Method Base PhD Thesis, TU München, 2011.

[Er10] Ernst, A.: A Pattern-based approach to *Enterprise Architecture Managemen.* PhD thesis, Technische Universität München, Munich, Germany, 2010.

[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.

[Ma08] Matthes, F.; Buckl, S.; Leitel, J.; Schweda, C.: *Enterprise Architecture Management Tool Survey 2008*. Technische Universität München, Chair for Informatics 19 (sebis), München, 2008. 978-3-00-024520-6.

[Wi07] Wittenburg, A.; *Softwarekartographie: Modelle und Methoden zur systematischen Visualisierung von Anwendungslandschaften.* PhD thesis, Technische Universität München, Munich, Germany, 2007.

Fragen?