



Dr. Hans-Peter Hoidn
Distinguished IT Architect (Open Group certified)

Enterprise IT Architectures

IT Architecture Standards,
TOGAF and OMG in more Detail,
Key Architecture Work Products

IT Architecture Standards

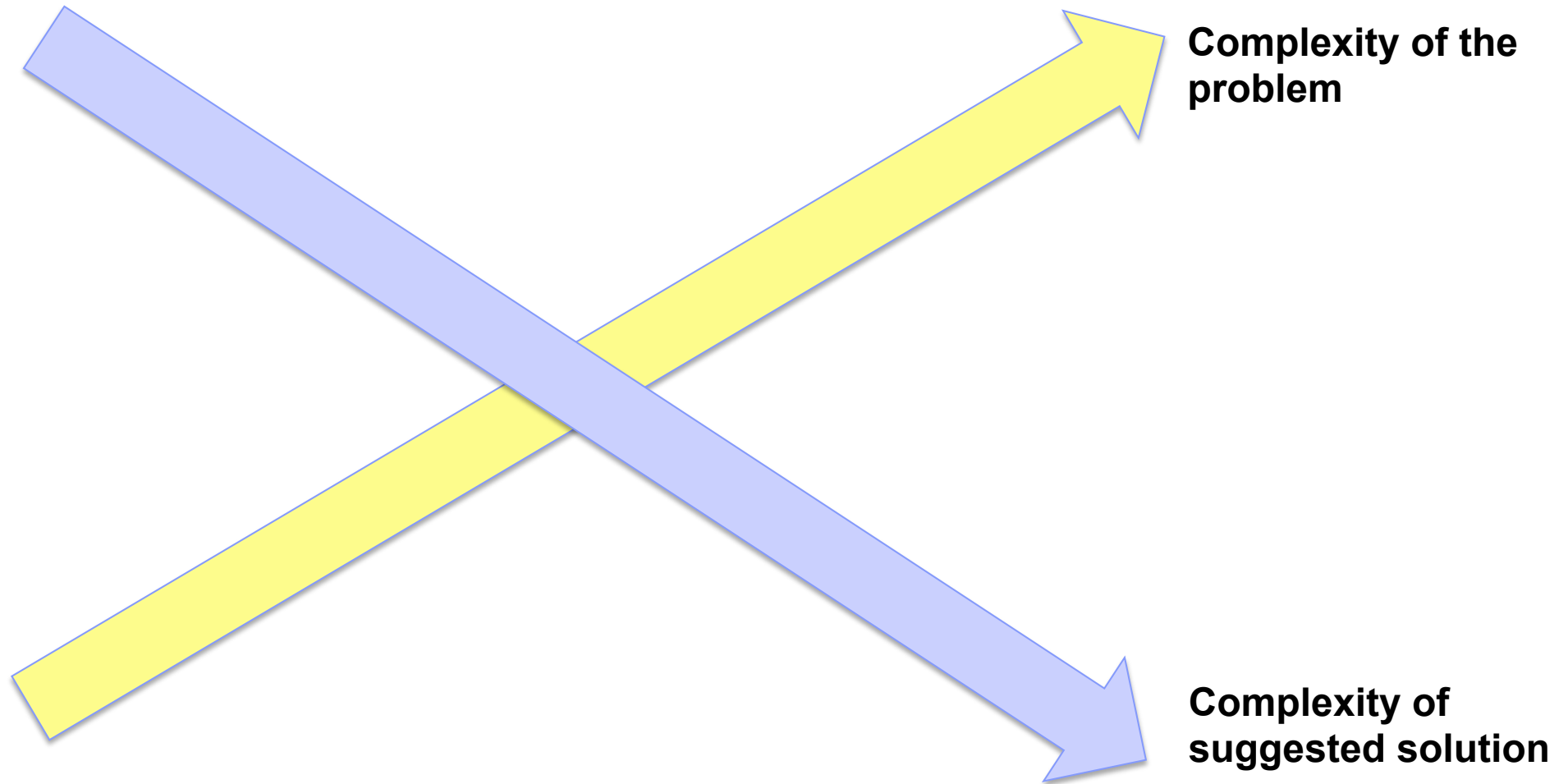
Recap: Architecture

- **Architecture is a term that lots of people try to define**
 - **There is not just one way to state a system's architecture**

- **Architecture provides:**
 - **Managing complexity**
 - **Layers of abstraction**
 - **Separation of Concerns**

- **Architecture covers:**
 - **Breakdown of a system into its parts**
 - **The relationship between the parts (static and dynamic)**
 - **Decisions about the design of a system that are hard to change**

Managing Complexity

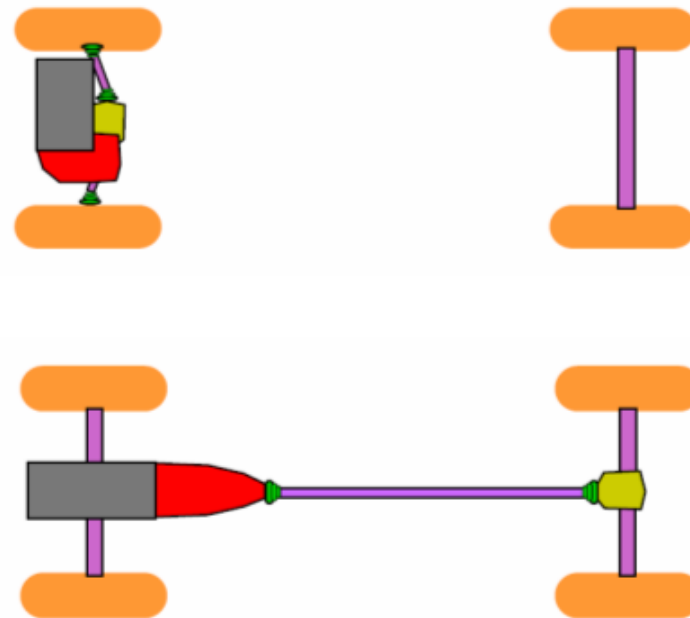


Architectures can be implied, apparent, or explicitly planned

- **Implied architecture**
 - of abstract things such as music or mathematics
 - **Apparent architecture**
 - of natural things, such as geological formations or the structure of biological cells
 - **Explicitly planned architecture**
 - of human-made things such as software, computers, enterprises, and databases, in addition to buildings.
- **Etymology:**
 - Latin: architectus
 - Greek: arkhitekton (αρχιτεκτων) = master builder
 - From arkhi (αρχι) = chief + tekton (τεκτων) = builder, carpenter
 - archon: one of the nine chief magistrates of ancient Athens, 1659, from Gk. arkhon "ruler"

In every usage, an architecture, whether implied, apparent or explicitly planned, may be seen as:

- A *subjective* mapping from **one of many possible** human perspectives
 - to the elements or components of some kind of structure or system,
 - which preserves the relationships among the elements or components.



Models and Semantics

- Addressing ***Content*** and ***Visualization*** of a model or a view; first refers to concepts, second to visualisations
- A ***Symbolic Model*** expresses properties of architectures of systems by means of symbols that refer to reality
- A ***Semantic Model*** is an interpretation of a symbolic model, expressing the meaning of the symbols in that model

Overview: Standards and Frameworks (Selection) – all with Enterprise View

- **IEEE**
 - **Definition and Metamodel**
 - **Uses UML notation**
 - **Provides a number of scenarios**

- **Zachman**
 - **First comprehensive framework**

- **TOGAF – by The Open Group**
 - **Will be our reference**

- **MDA (Model Driven Architecture) – by the OMG**

IEEE 1471: Standard for Defining Architecture

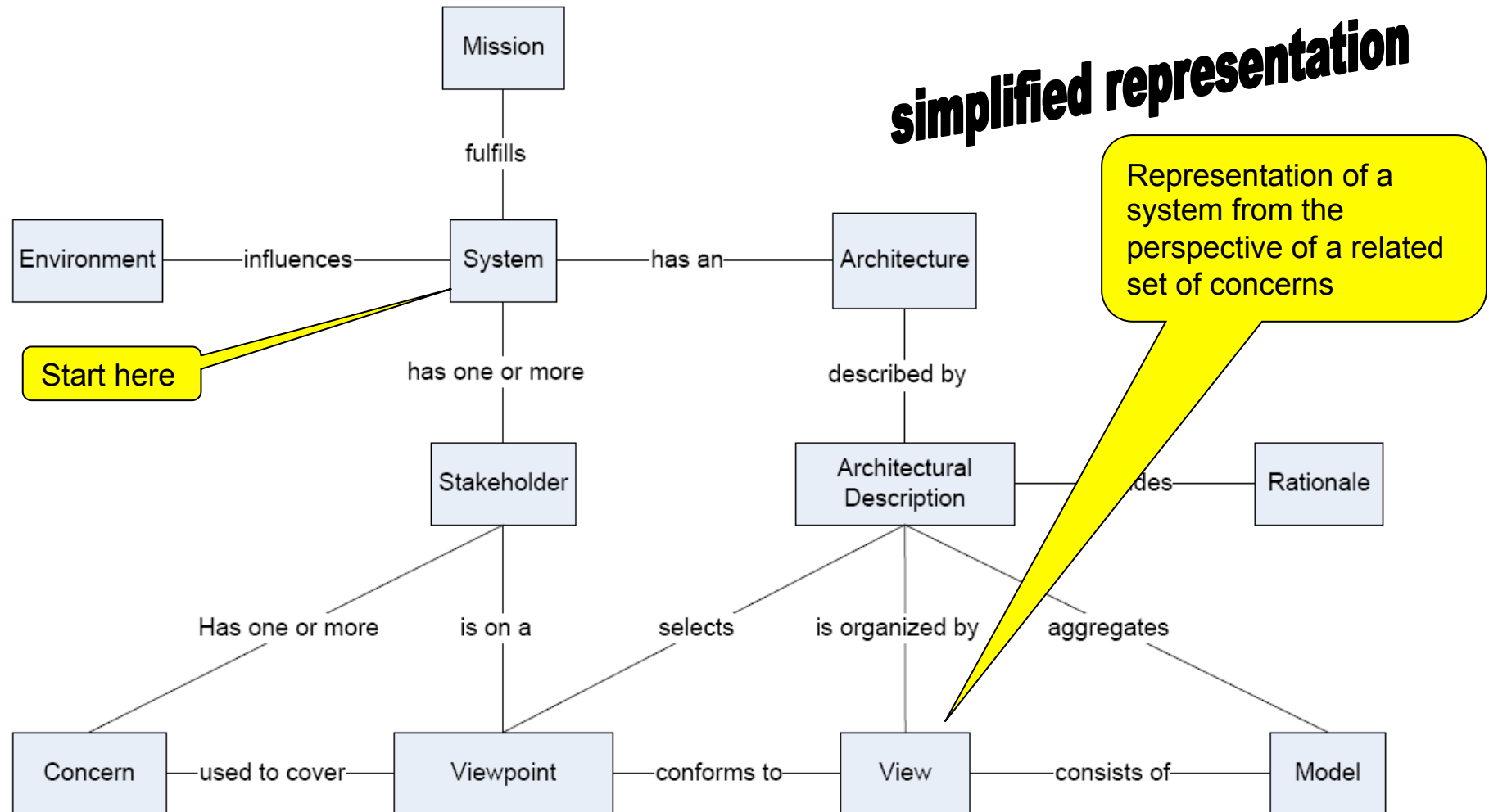
- **IEEE**

- **1471 “Recommended Practice for Architectural Description of Software-Intensive Systems”**
- **Approved by IEEE in 2000, adopted by ISO 2007**
- **superseded by ISO/IEC/IEEE 42010:2011**

- **Standard:**

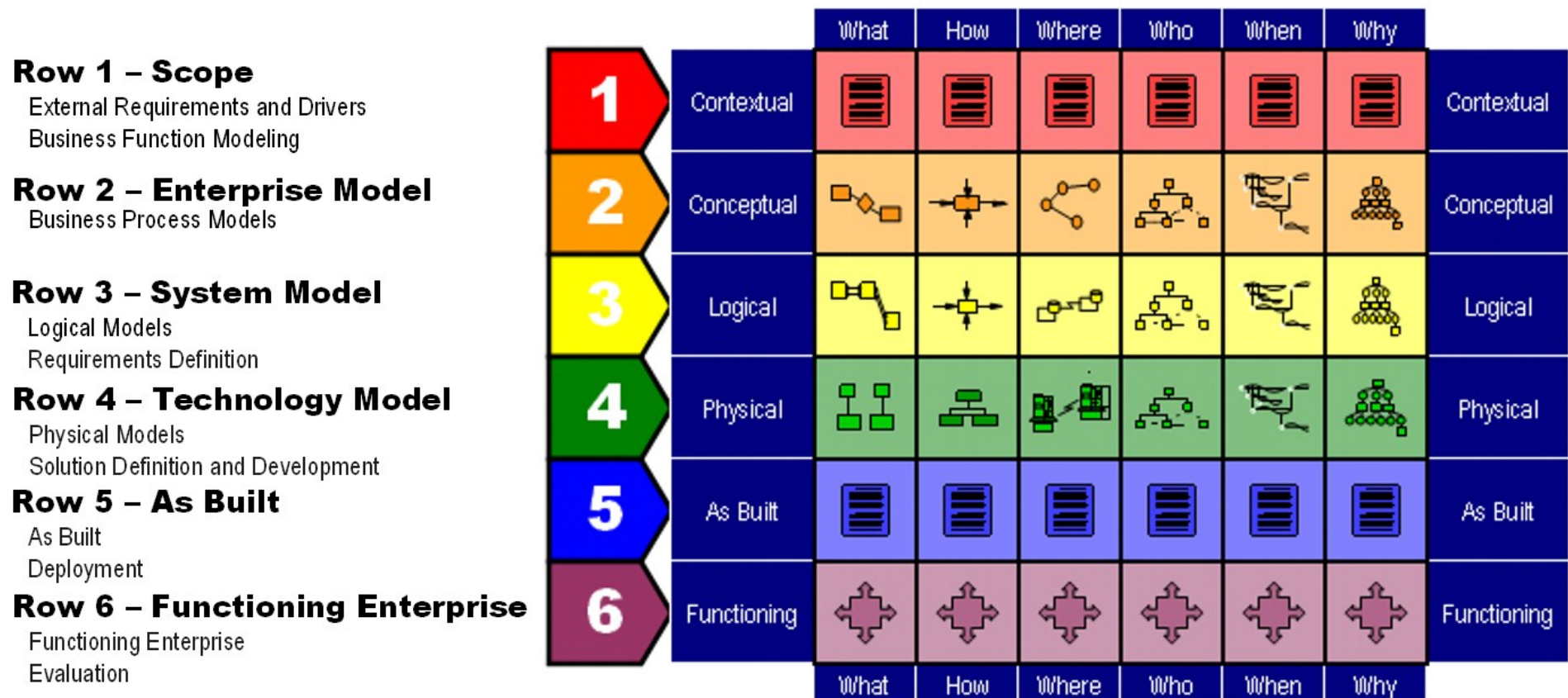
- **It provides definitions and a meta-model for the **description** of architecture**
- **It states that an architecture should address a system's **stakeholders concerns****
- **It asserts that architecture descriptions are inherently **multi-view**, no single view adequately captures all stakeholder concerns**

IEEE 1471: Standard for Defining Architecture



Zachman Framework (see <https://www.zachman.com>)

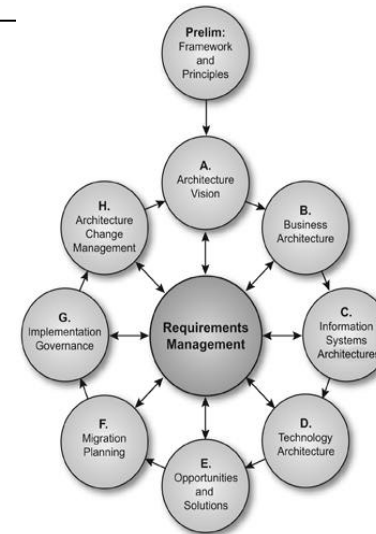
- Zachman Framework
 - Is an ontology and a meta-model
 - Is not a methodology



TOGAF
(The Open Group Architecture Framework)

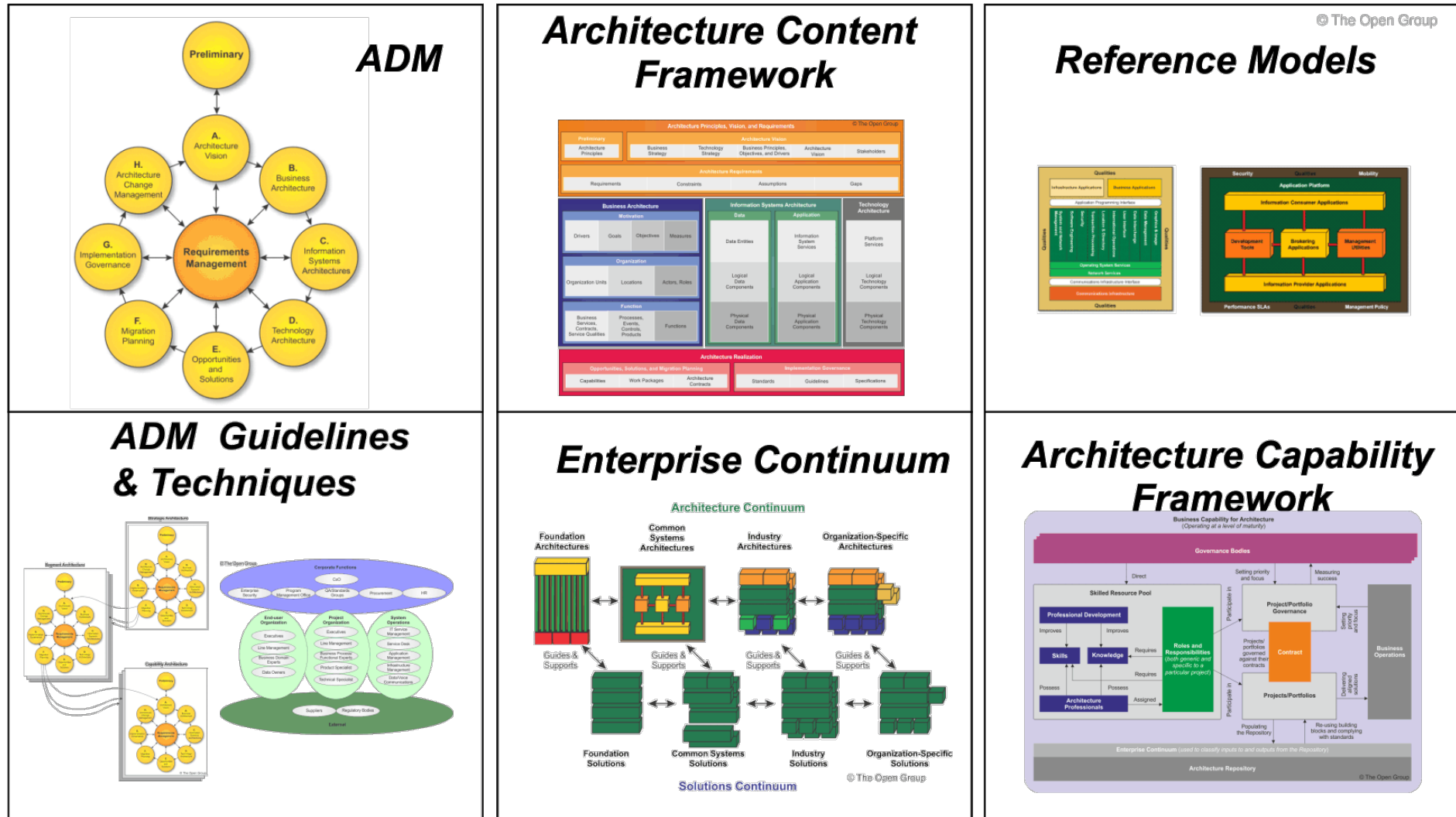
What is TOGAF

- **TOGAF consists of the following :**
 - **Architecture Development Method (ADM)**
 - **Enterprise Continuum**
 - **Resource Base**

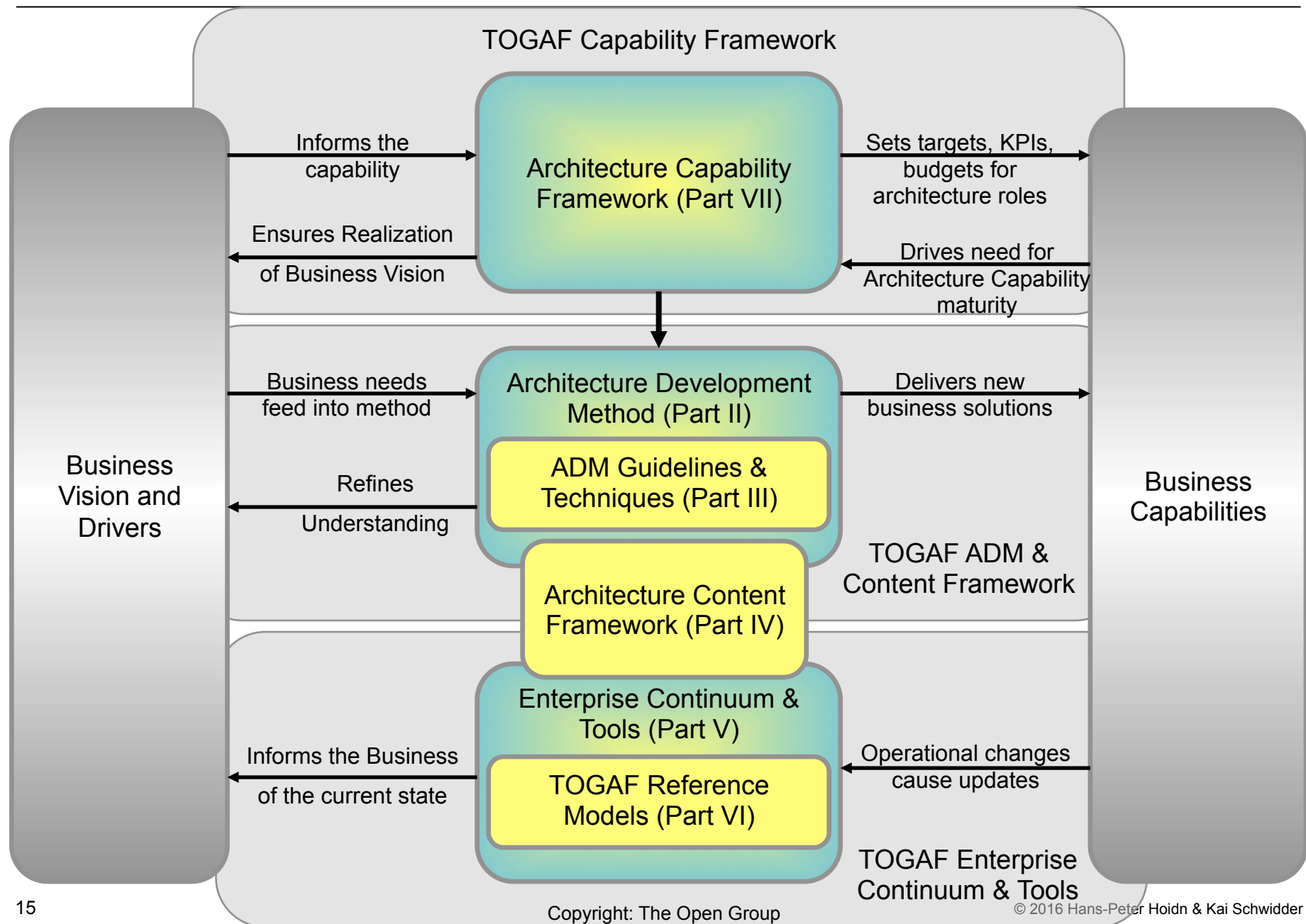


- **The ADM is depicted as the ‘crop-circle’ and represents the core of the TOGAF specification. It is a method for deriving a specific enterprise architecture.**
- **The Enterprise Continuum is a model for structuring a ‘virtual repository’ of architectural assets such as patterns, models, & architecture descriptions.**
- **The Resource Base is a set of ‘good practice’ resources such as guidelines, checklists and templates provided to assist the architect when using TOGAF ADM.**

TOGAF Components (Version 9.1)

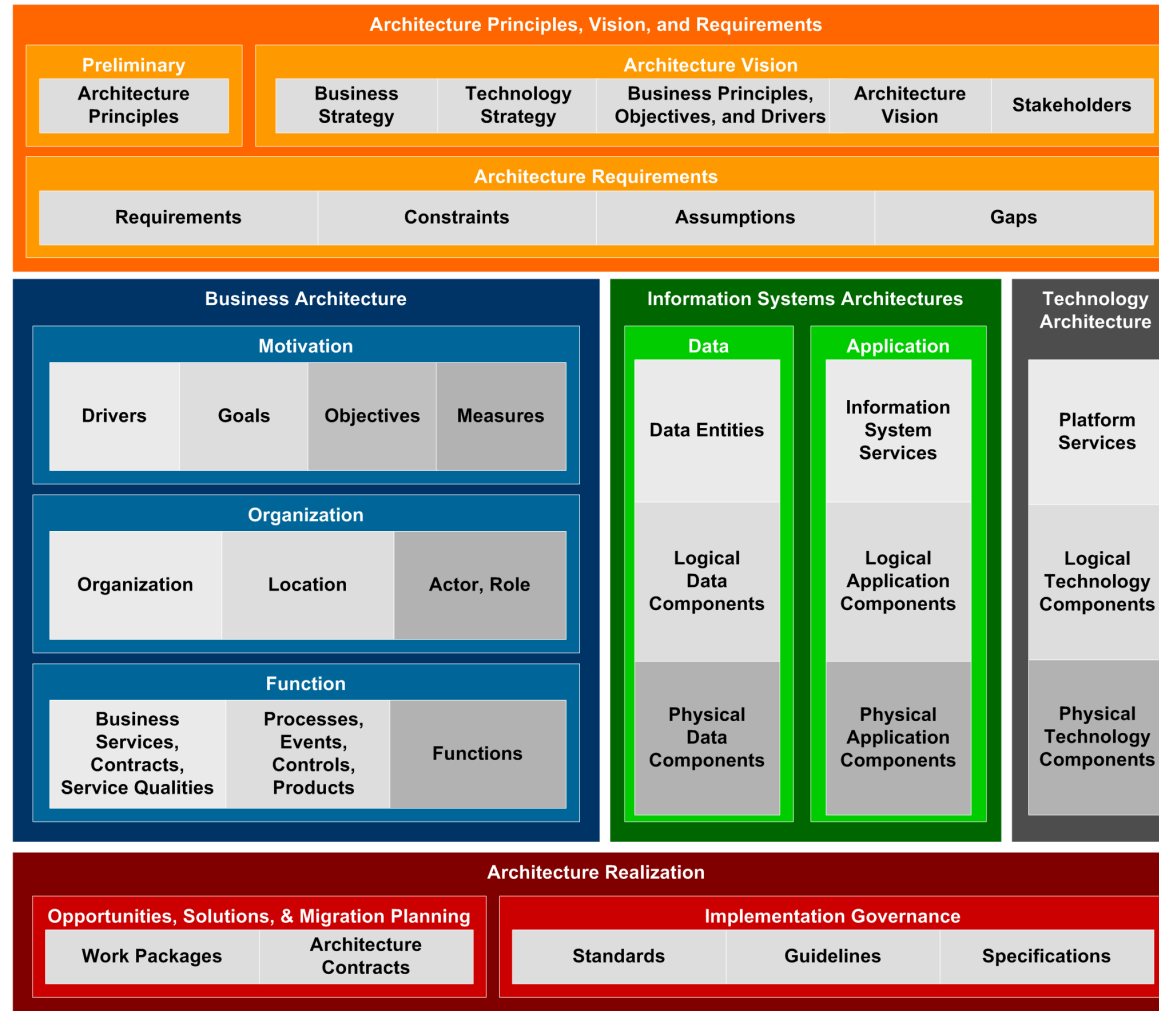


Enterprise IT Architectures



Enterprise IT Architectures

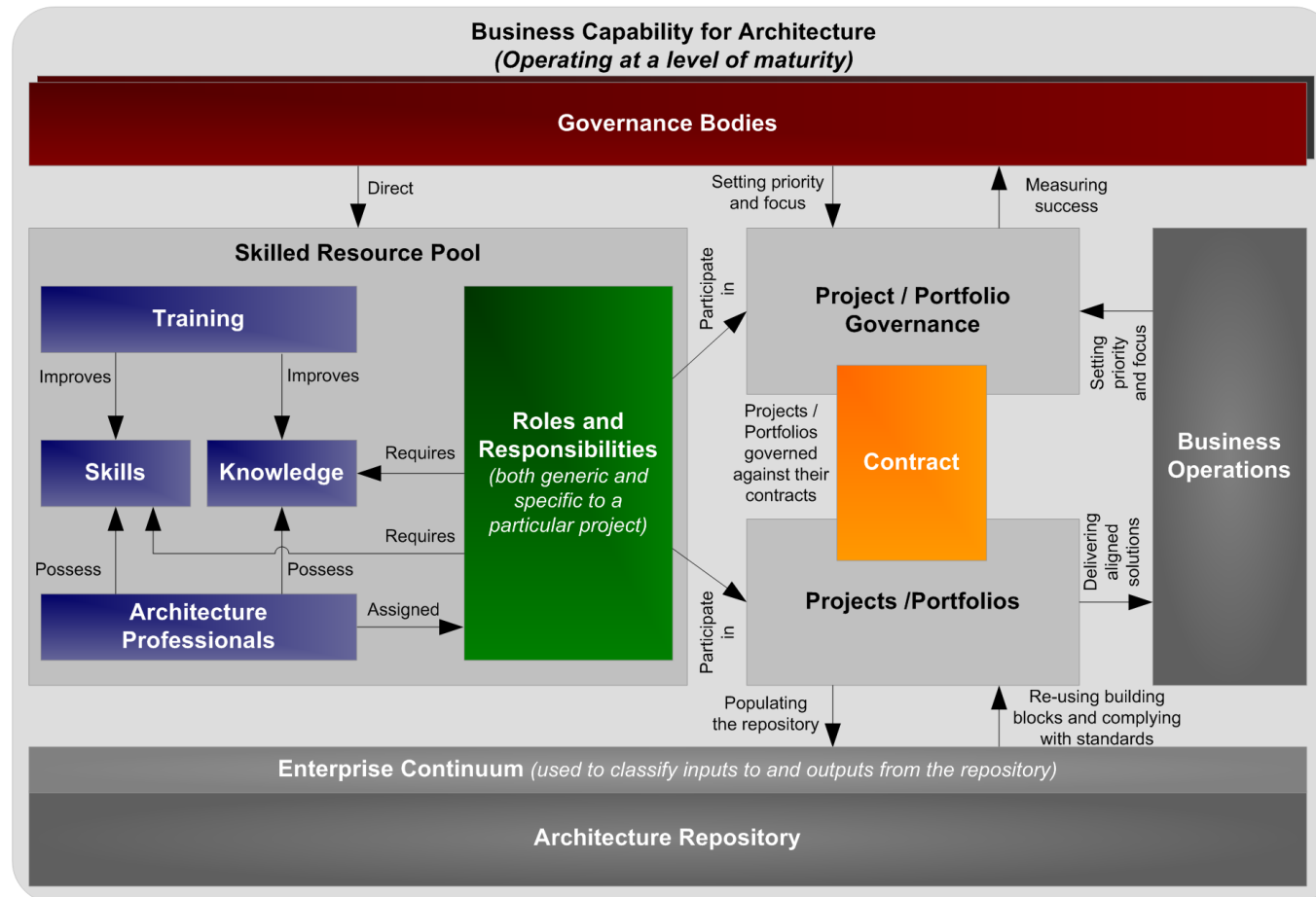
Content Metamodel



Content Metamodel

- **Building blocks continue to be the basic elements of the architecture within TOGAF**
- **The content framework provides a structured model of building block types, relationships and attributes**
- **The content framework can be used informally, or as the basis for configuration of an Enterprise Architecture modelling tool**
- **The content framework features a core and extension concept, with optional building block types, in order to support lightweight and detailed architectures**

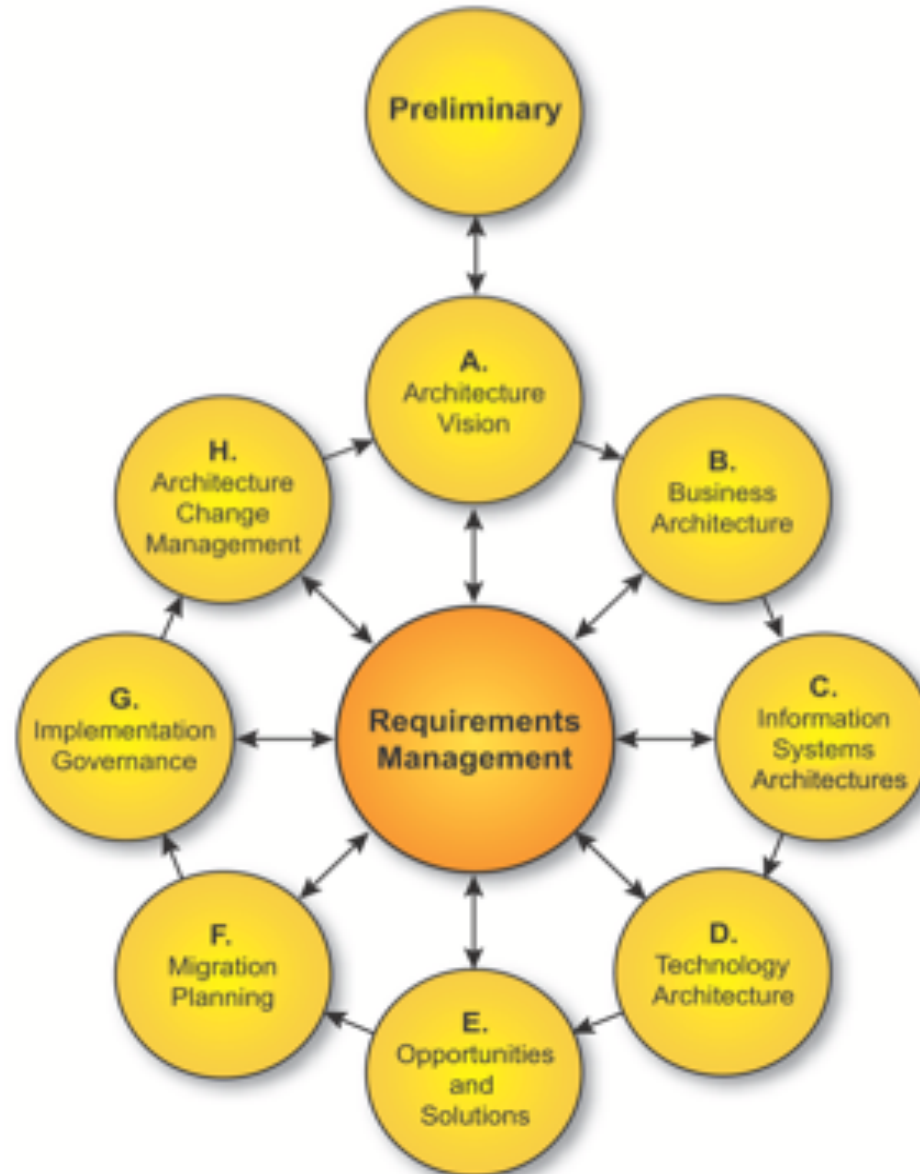
Capability Framework (supporting Governance)



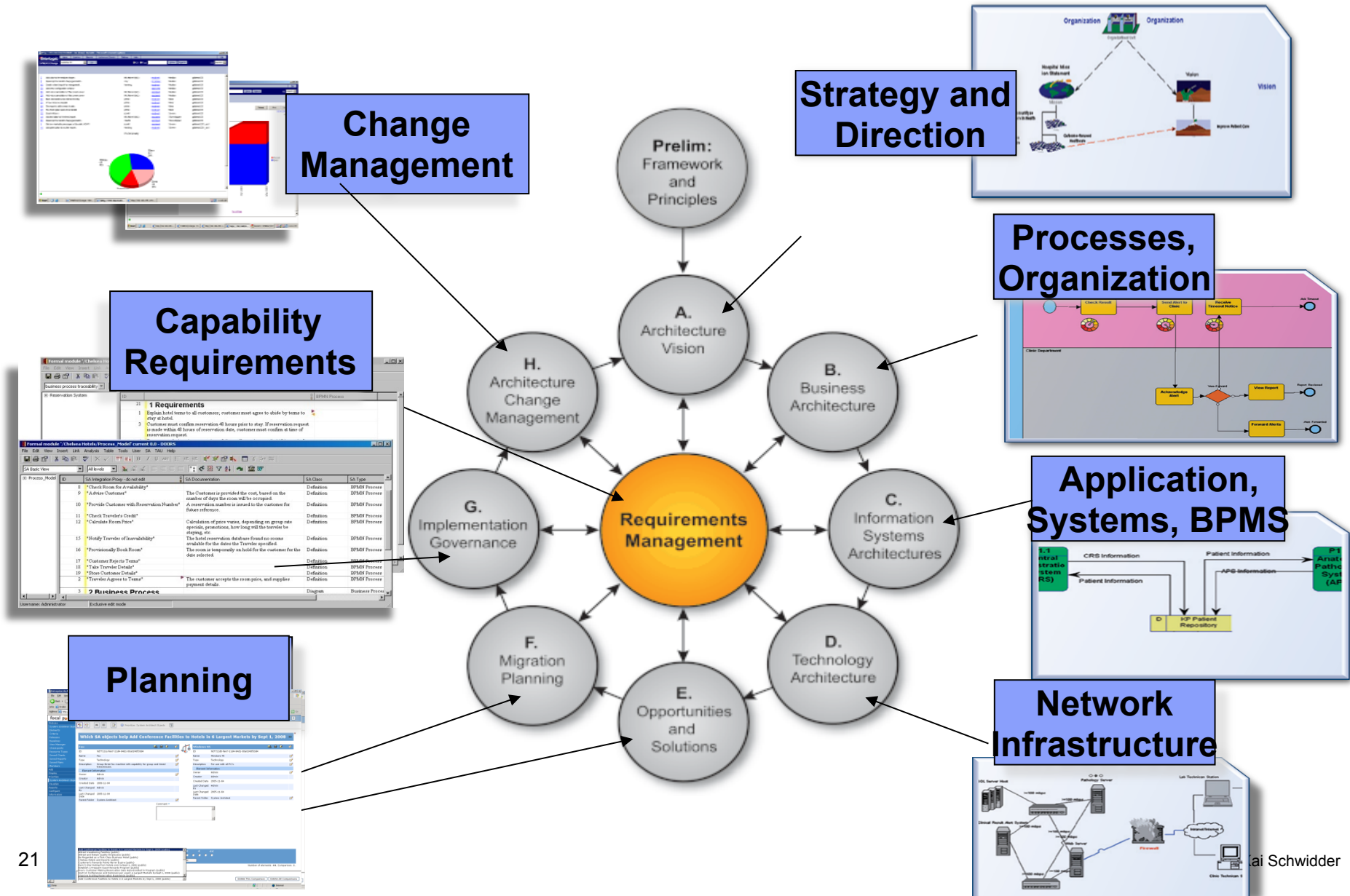
Capability Framework (supporting Governance)

- **A structured definition of the organizations, skills, roles and responsibilities to establish and operate an Enterprise Architecture, including:**
 - **Terms of Reference for an Architecture Board**
 - **Guidance on measuring levels of Architecture Compliance against Architecture contracts**
 - **Processes and organization structures required to operate Architecture Governance**
 - **Techniques for assessing Architecture Maturity**
 - **An overview of the Skills required by practicing architects**

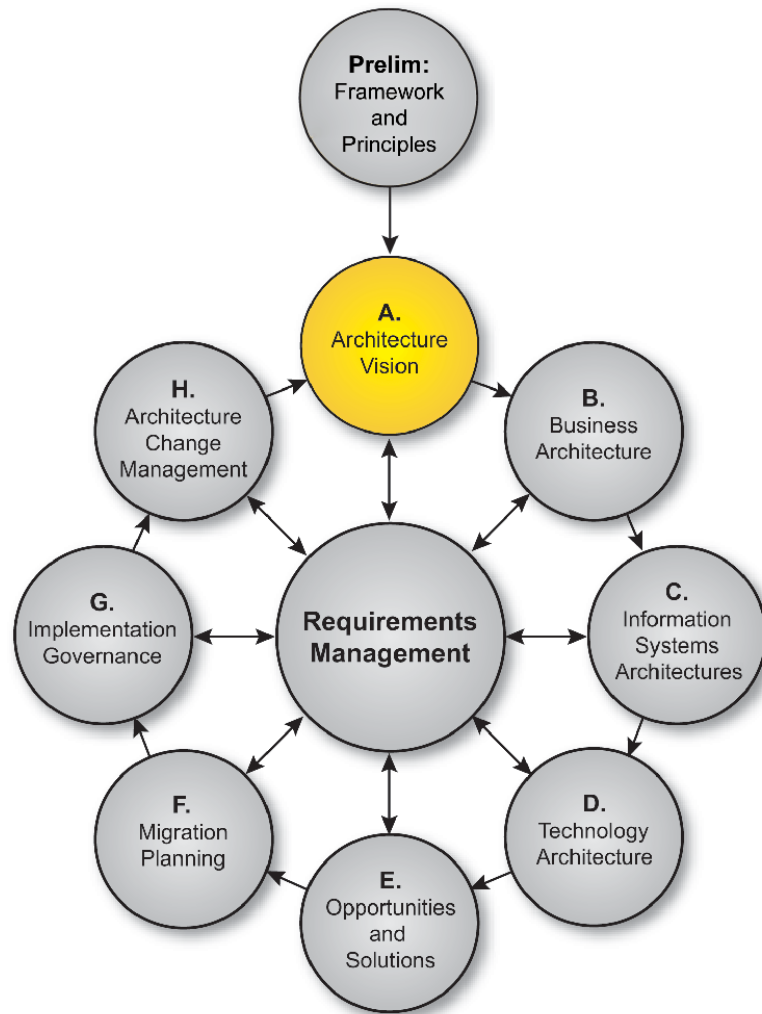
TOGAF ADM (Architecture Development Method)



ADM Detail Overview

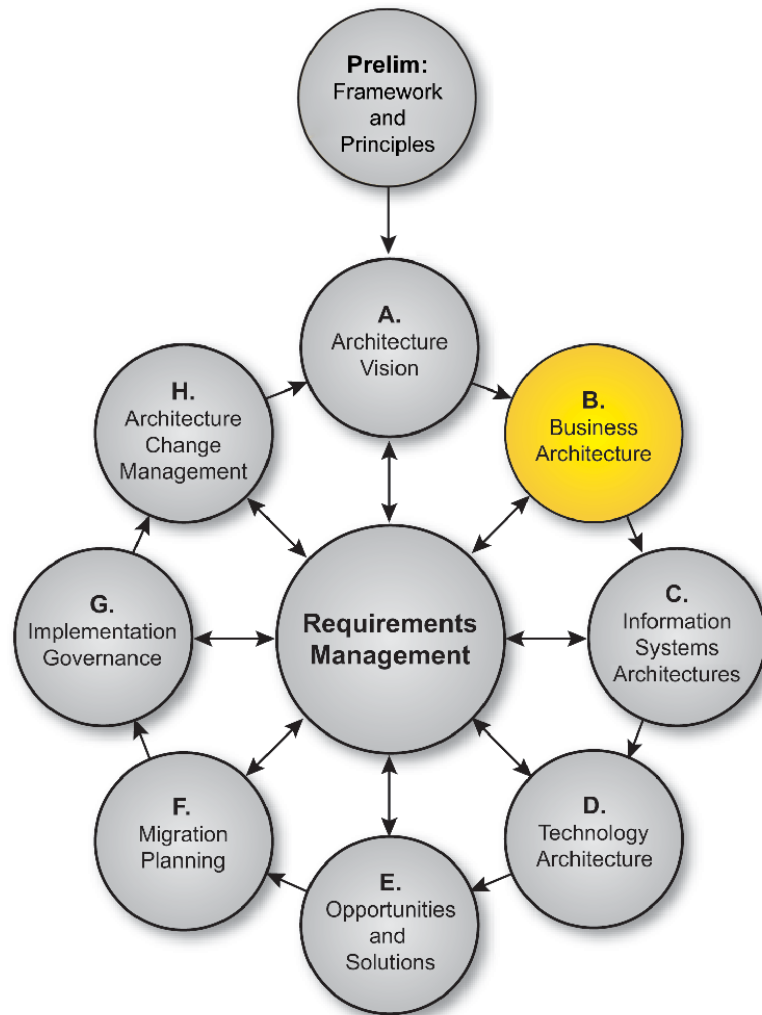


In more Detail: Capabilities Content (Architecture Vision according to TOGAF - augmented)



- **Initiates one iteration of the architecture process**
 - Sets scope, constraints, expectations
 - Required at the start of every architecture cycle
- **Validates business context**
- **Creates Statement of Architecture work**
- **Guiding Principles**
(according to IBM EA Methodology)
- **Architectural Decisions**
(according to IBM EA Methodology)

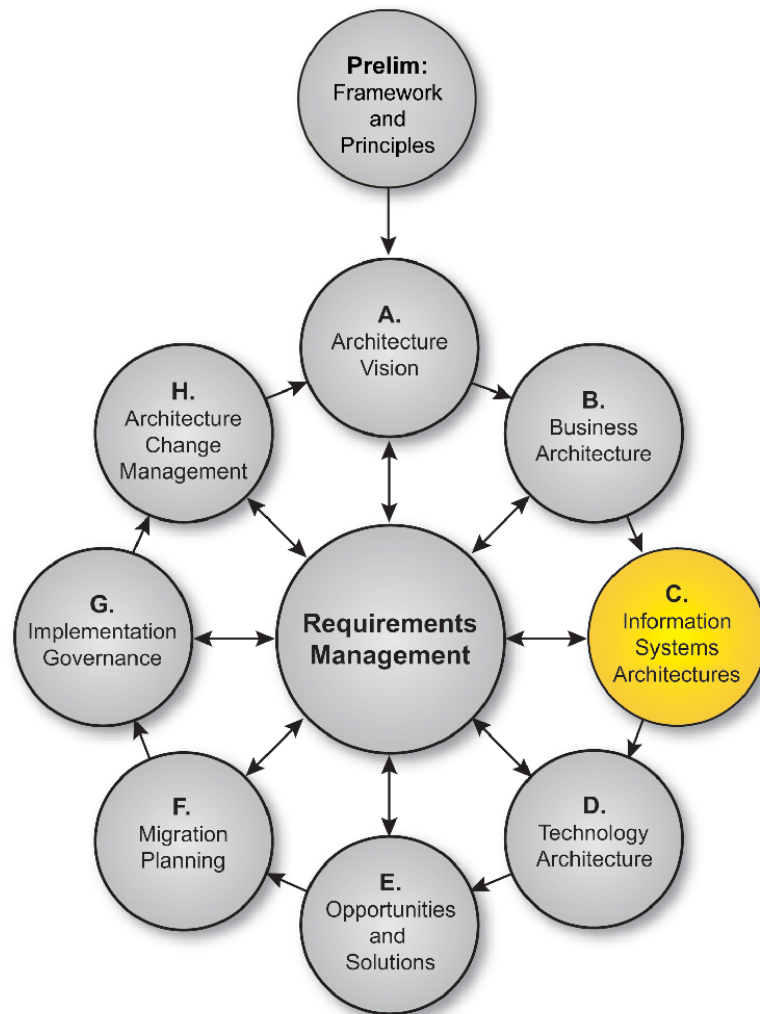
In more Detail: Business Architecture Content (according to TOGAF augmented)



- **Organization structure**
- **Business Goals and Objectives**
- **Business Functions**
- **Business Services**
- **Business Processes**
- **Business Roles**
- **Correlation of organization and functions.**

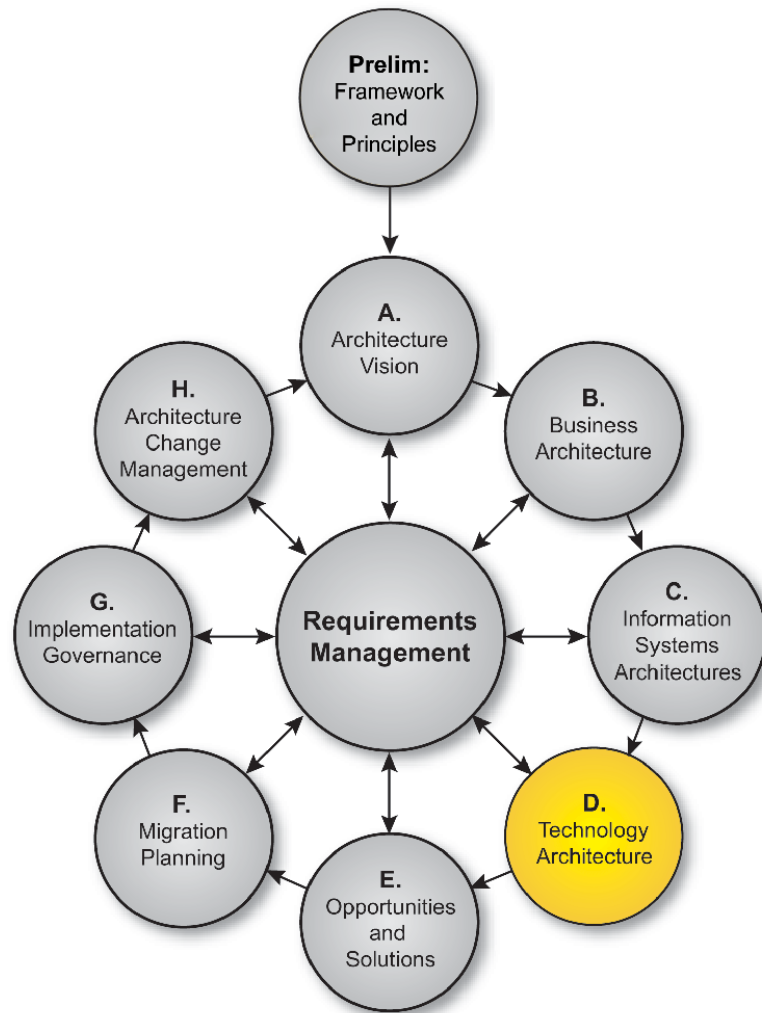
- **Enterprise Information Model (according to IBM EA Methodology)**

In more Detail: IS Architecture Content (according to TOGAF)



- The fundamental organization of an IT system, embodied in
 - relationships to each other and the environment, and the principles governing its design and evolution
- Shows how the IT systems meets the business goals of the enterprise

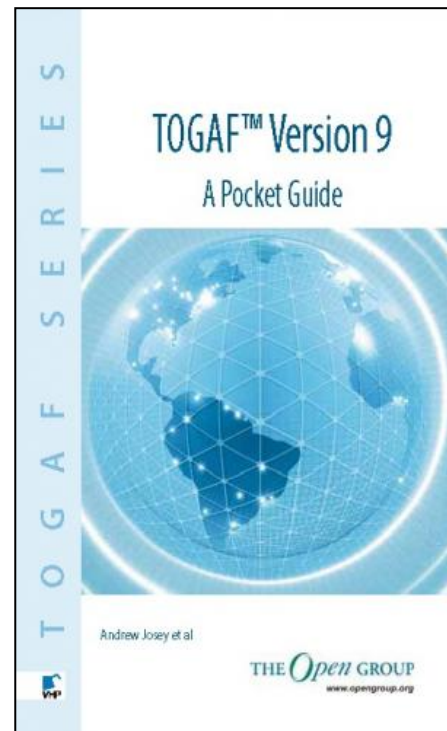
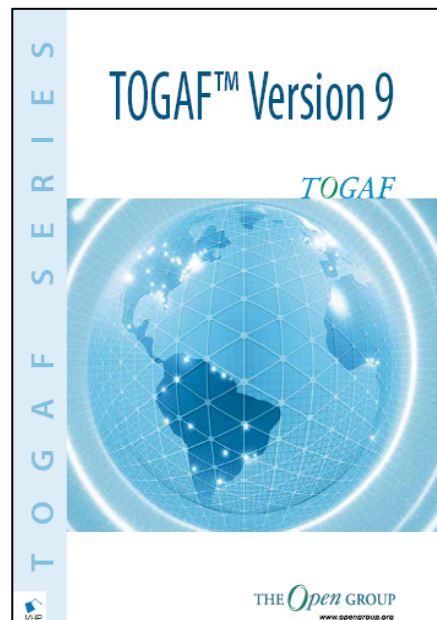
In more Detail: Technology Architecture Content (according to TOGAF)



- The fundamental organization of an IT system, embodied in
 - its hardware, software and communications technology
 - their relationships to each other and the environment,
 - and the principles governing its design and evolution

Further Information on TOGAF

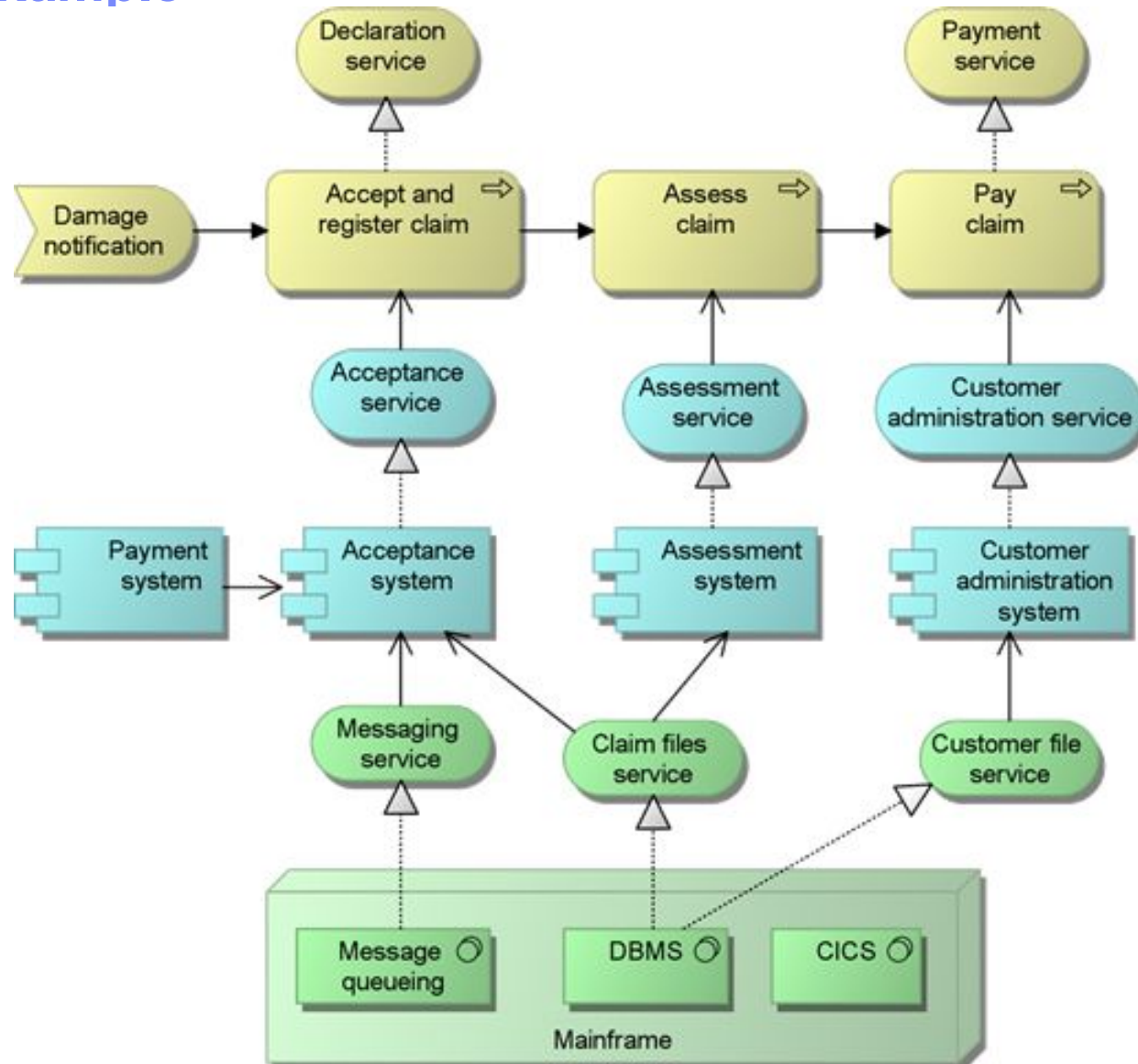
- Detailed information on TOGAF 9 including downloads of the specification, links to white papers, information sheets, reference cards, etc is available at:
 - <http://www.opengroup.org/togaf/>
 - <http://www.togaf.info>



ArchiMate

- **ArchiMate is a modeling technique ("language") for describing enterprise architectures**
 - **ArchiMate is partly based on the IEEE 1471 standard. It was developed in the Netherlands by a project team from the Telematica Instituut in cooperation with several Dutch partners**
 - **Now Open Group Standard – see <http://www.opengroup.org/subjectareas/enterprise/archimate>**
- **ArchiMate offers a common language for describing the construction and operation of business processes, organizational structures, information flows**
 - **Layering: Business, Applications, Infrastructure**
 - **Dimensions: Passive Structure, Behavior, Active Structure**

ArchiMate Example



TOGAF – HPH’s Personal Remarks

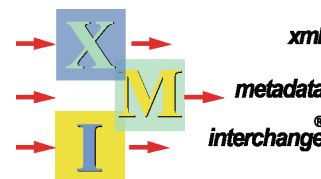
- **TOGAF provides solid and agreed definitions of terms (backed by a large membership of companies)**
- **ArchiMate provides a modeling language for architecture, however we may do some work redundantly, e.g. we will later model business processes on its own**
- **Includes NOT MUCH about deliverables to support architecture work (“work products” / “artifacts”)**
- **Does NOT emphasize iterative work – however IMHO (In My Humble Opinion) ALL work should be done in iterations**

Models – and OMG (Object Management Group)



OMG (Object Management Group)

- “owner” of CORBA, UML, MOF, MDA, CWM, BPMN, ...
- Only organizations (ca. 300 now) can be members, work done by the representatives of the members
- **OMG’s Role**
 - Build *Consensus* for Interoperability over heterogeneous networks
 - and, through *Modeling*, support the design, implementation, and maintenance of the interoperable systems that run on them



OMG Major Successes

Unified Modeling Language

- UML® remains the world's only standardized modeling language

Business Process Modeling Notation

- BPMN™ provides businesses with the capability of understanding their internal business procedures

Systems Modeling Language

- SysML™ supports the specification, analysis, design, and verification and validation of a broad range of complex systems.

Data Distribution Service

- DDS™, Real-time, data-centric, publish-subscribe OMG specification for data distribution

Meta-Object Facility

- MOF™, the repository standard

XML Metadata Interchange

- XMI®, the XML-UML standard

Common Object Request Broker Architecture

- CORBA® remains the only language- and platform-neutral interoperability standard



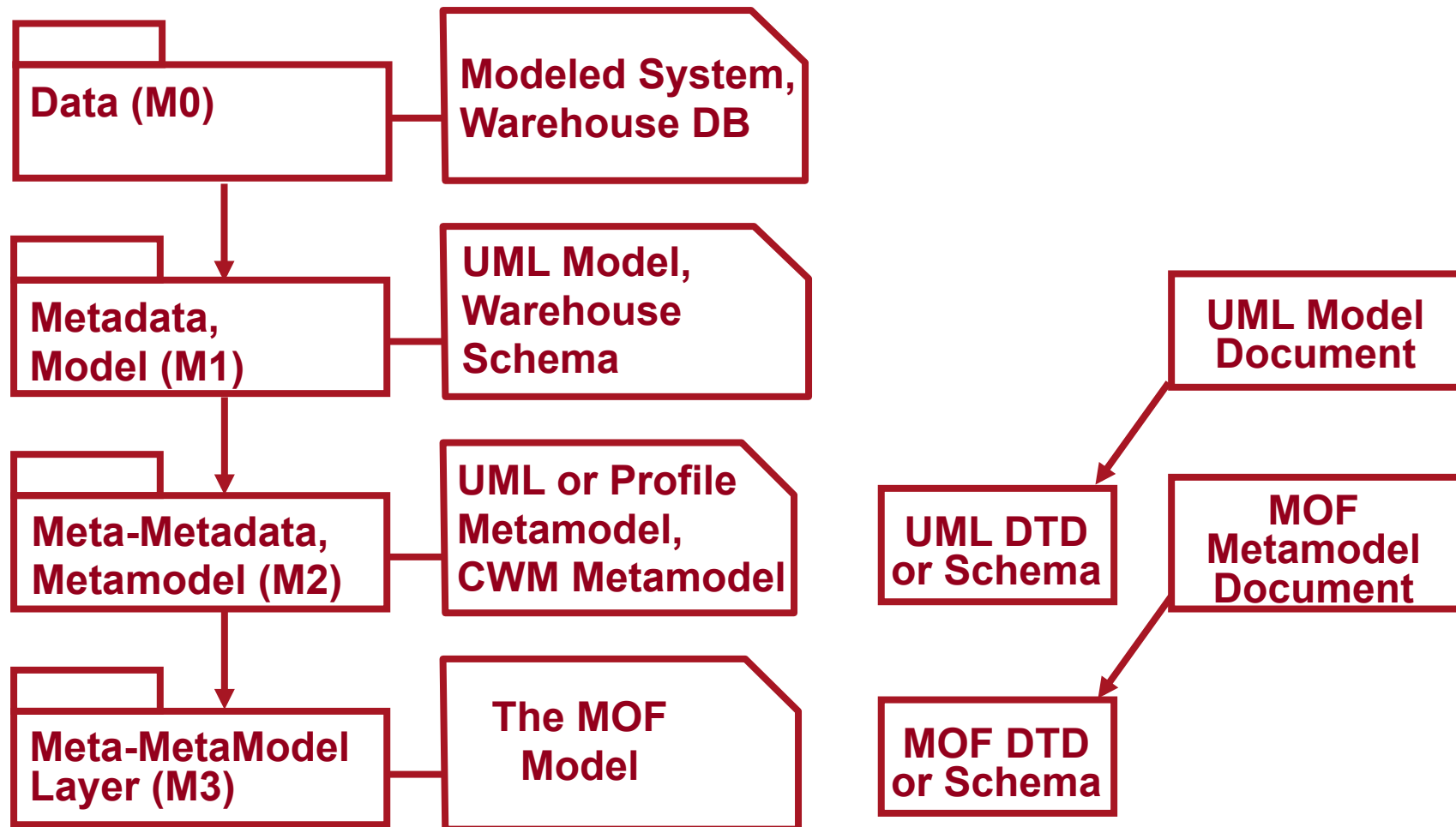
MOF (Meta-Object Facility)

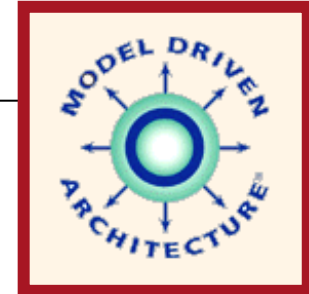
- **Is a meta-meta-model OR a model that defines the constructs for meta-models (a meta-model defines the structure of models, e.g. the meta-model for UML Class diagrams, Use cases, ...)**

- **Advantages (quote from OMG):**
 - **MOF standardizes the format of Metadata for compliant Languages and Models**
 - **If you know the Meta-Model for your Modeling Language, you can specify a Transformation**
 - **And a Tool can apply the Transformation to your Input Model, and produce an Output Model**
 - **Based on MOF, all diverse model elements can share repositories and interchange models among compliant tools**



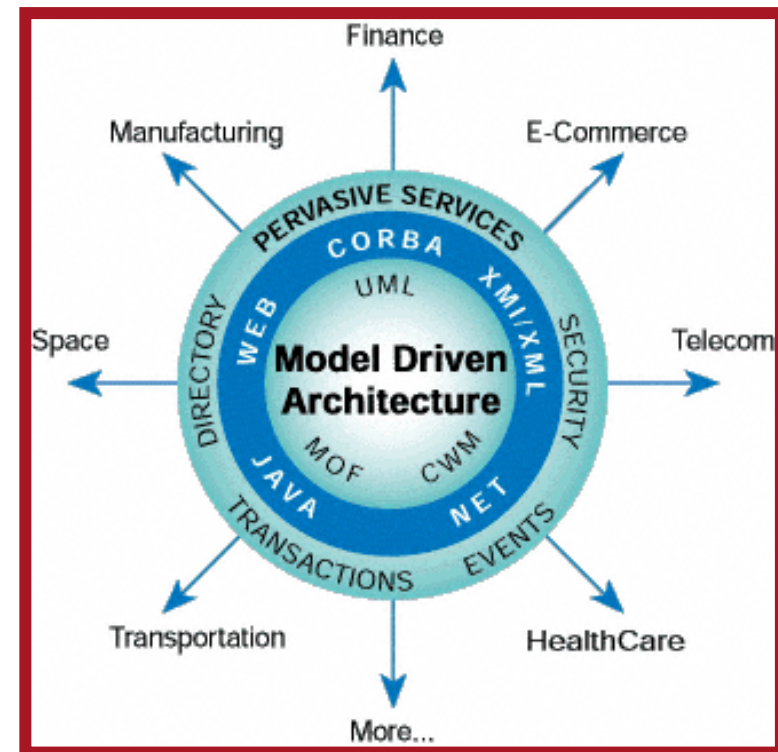
Interchanging Metadata

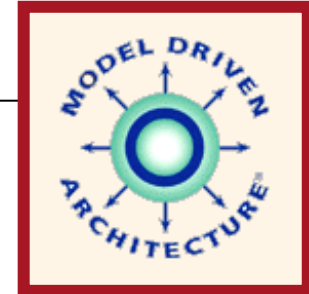




MDA (Model Driven Architecture)

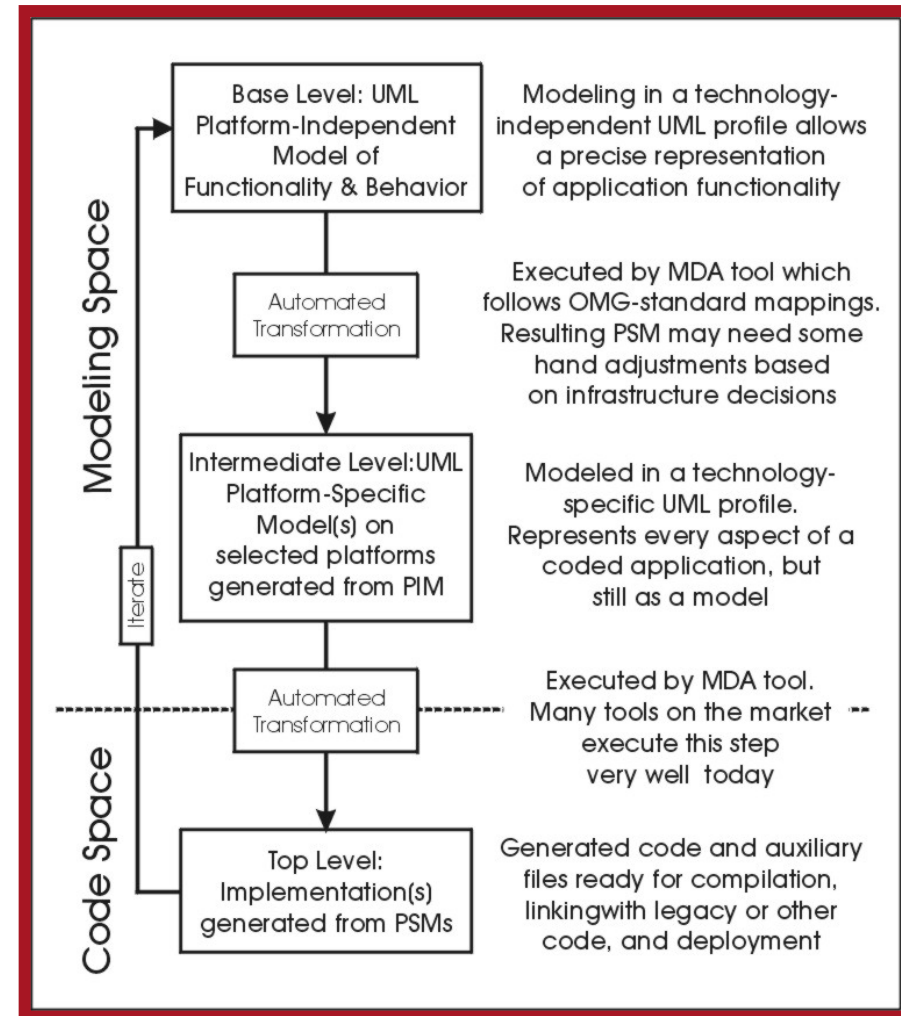
- **A Way to Specify and Build Systems**
 - Based on Modeling and UML
 - Supports full lifecycle: A&D, implementation, deployment, maintenance, and evolution
 - Builds in Interoperability and Portability
- **Reference:**
 - MDA Guide rev. 2.0 OMG Document ormsc/2014-06-01
 - See <http://www.omg.org/mda/>
- **For**
 - Unifies modeling and implementation into a synergistic environment

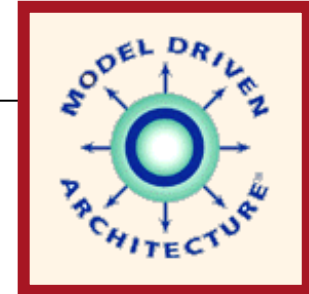




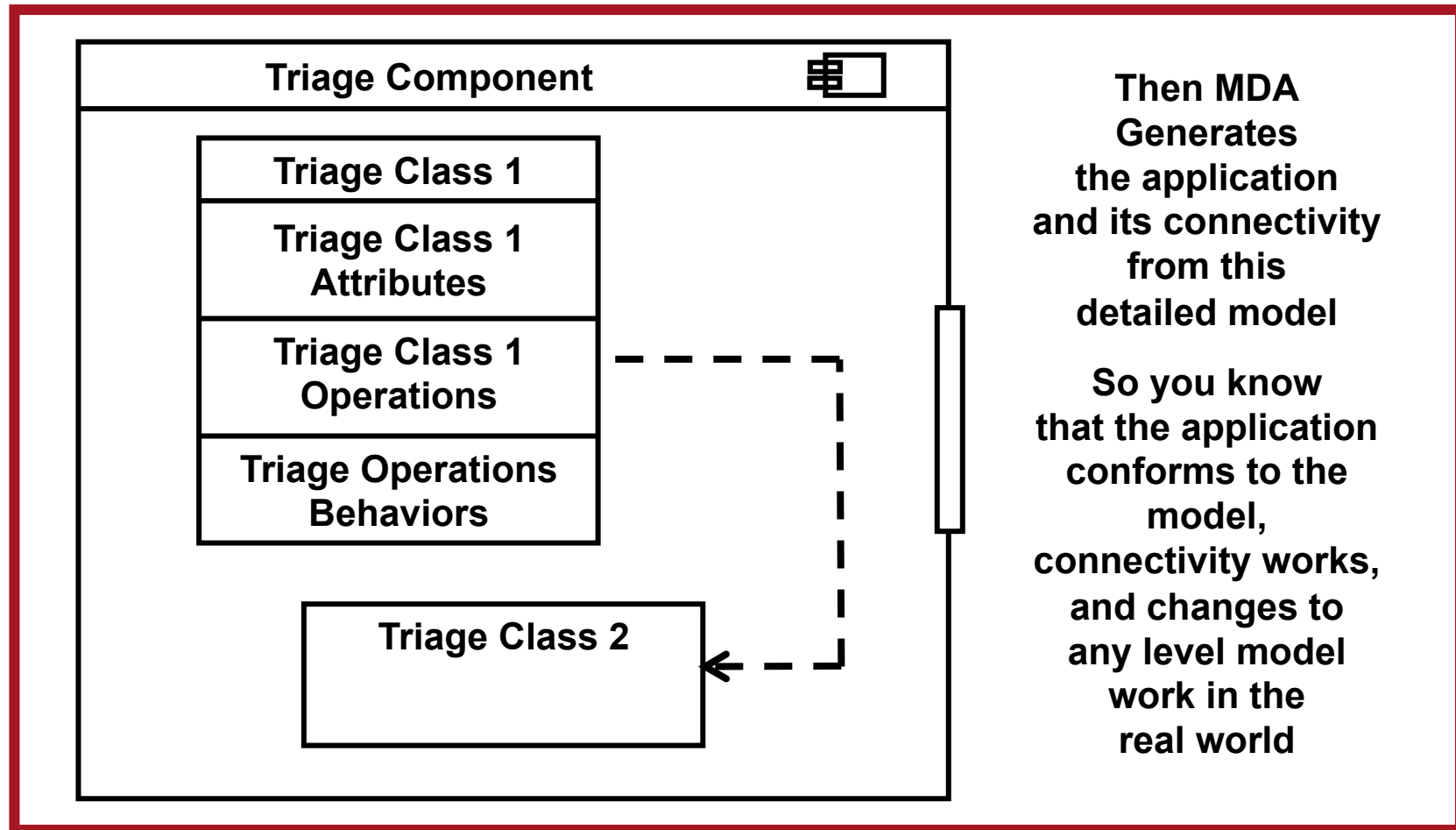
Models designed for Business – MDA Model Transformation and Execution

- **Structure is a Spectrum progressing from Modeling at the Top to Code development at the bottom**
- **Input and Investment concentrate at the business zone at the top**
- **Automated tools take over coding IT infrastructure towards the bottom**





Component Model and MDA



**Then MDA
Generates
the application
and its connectivity
from this
detailed model**

**So you know
that the application
conforms to the
model,
connectivity works,
and changes to
any level model
work in the
real world**

OMG – HPH's Personal Remarks

- **OMG is heavily supporting the modeling approach. Modeling is a very strong approach because of the semantics that allow to verify models as well as use the model for execution (as done by BPMN)**
- **The standardization process is slow and sometimes interests of member companies drive too much the development (as it happens with UML 2)**
- **Models are suitable for IT professionals – not always consumable for stakeholders (thus the tendency to replace models by ad hoc visualizations)**
- **HPH's bias: Served as primary representative for UBS and PwC at the OMG 1999 – 2002 (including alternate board member 2001)**

Questions



**“Work Products”
following (a bit) IBM’s Methodology**

Techniques and Methods

- **Architectural Thinking**
 - **Business – Information Systems – Technology**
 - **Taking decisions and documenting them**
(**“what is not written down does not exist”**)

- **Assessments and Evolutions**
 - **Assessment of Current IT Environment**
 - **Existing goals, functional and non-functional requirements**

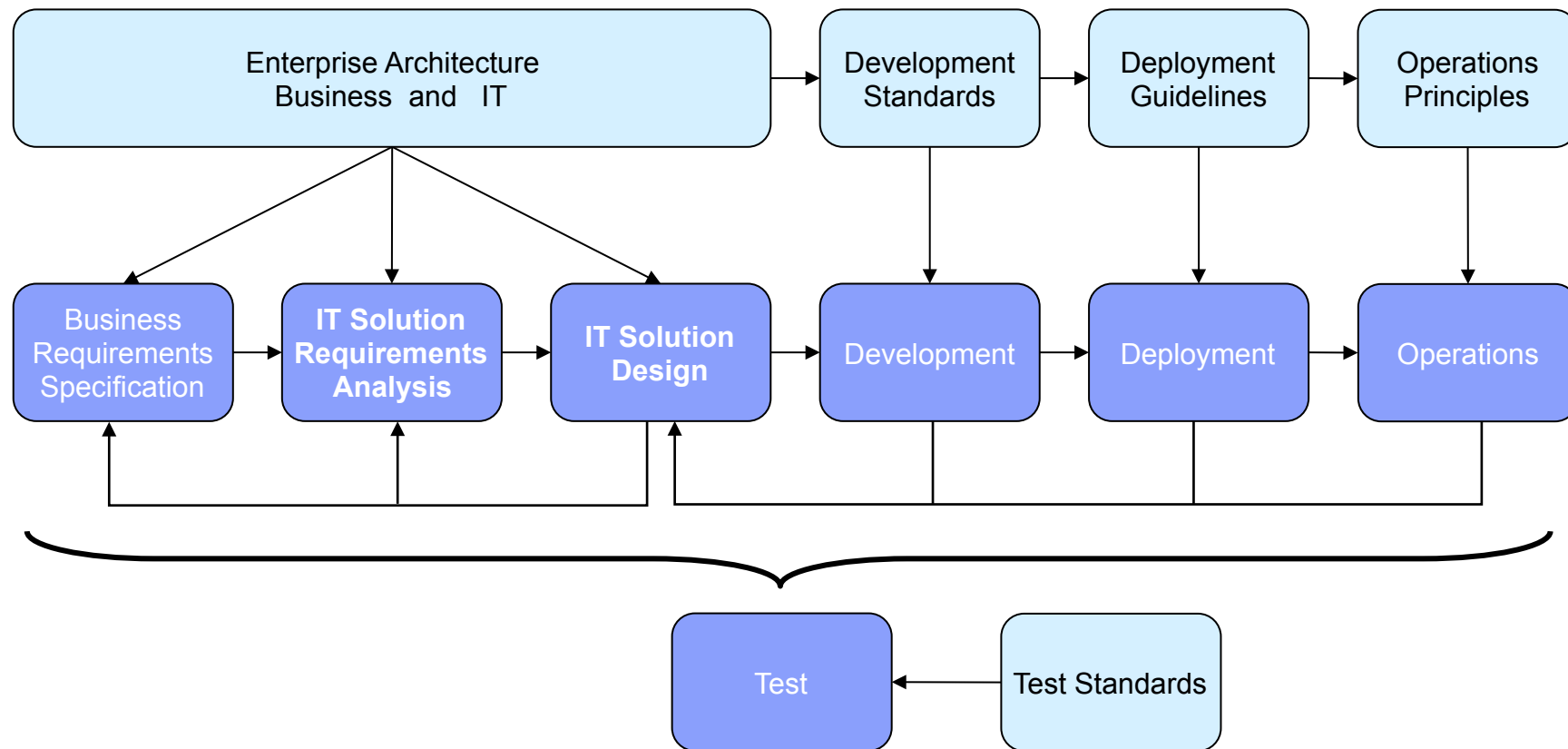
- **Major artifacts (work products):**
 - **Architecture Overview**
 - **Component Model**
 - **Operational Model**

Architectural Decisions: Making Decisions Transparent

Attribute	Value
Subject Area	BPM (Business Process Management)
Architectural Decision	Use of a BPM Engine for the management and control of business processes
Issue	Business Processes are too slow and nobody has an overview of where the processes are
Assumptions	Tools that implement Standard BPMN are available and run on the existing IT infrastructure
Motivation	Using existing tools and software would provide an immediate benefit by managing existing business processes. Processes can be shorter, bottlenecks can be removed, monitoring allows control of the whole processes
Alternatives	Hand coding of the process OR no support for business processes and thus no improvement for the business
Implications	Use of BPM Tools

Enterprise IT Architectures

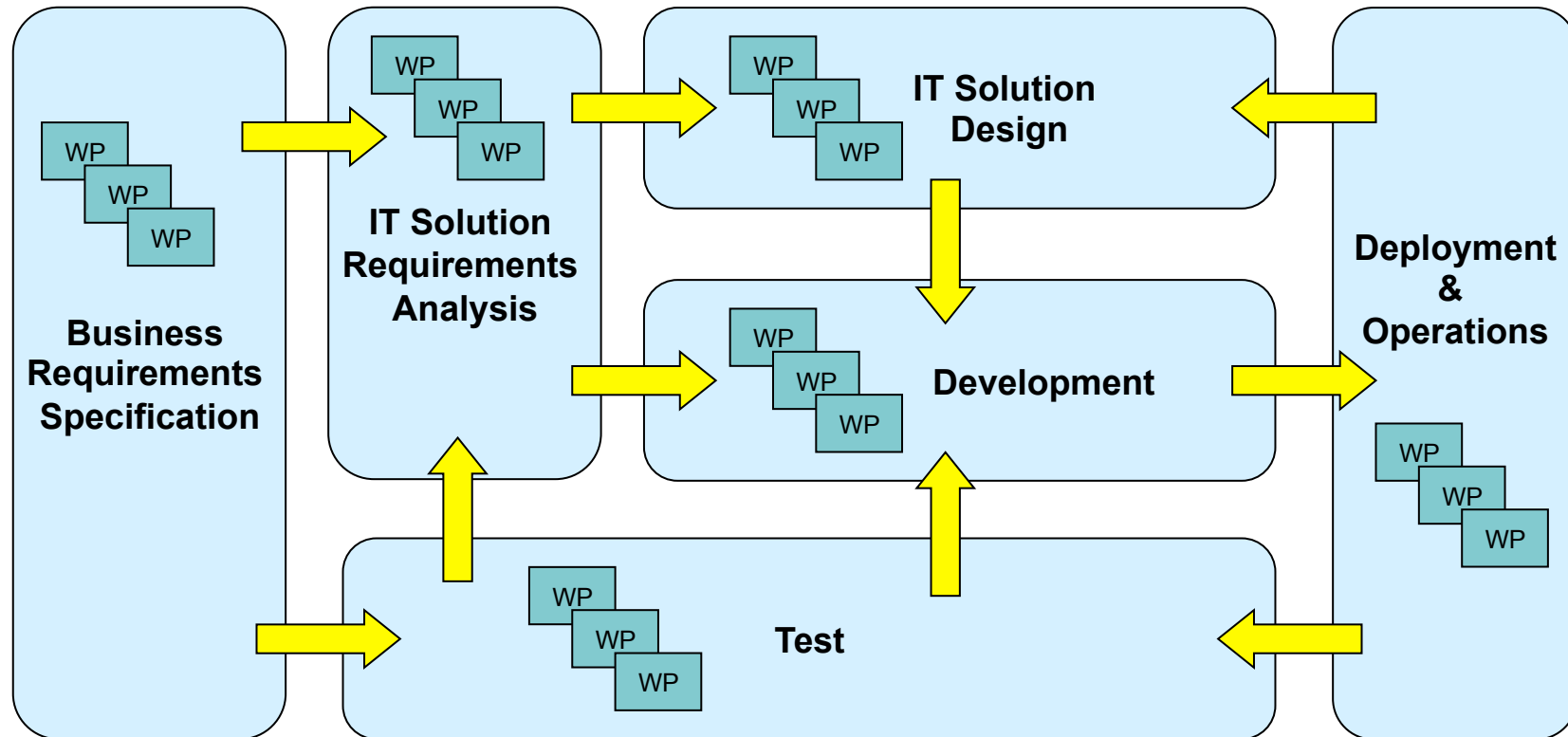
An IT Architect does not work alone – their work is intimately interconnected with that of others in both the business & IT organisations...



Enterprise IT Architectures

...with a strong set of inter-dependencies between all areas

Illustrative dependencies only! Most arrows are two way!



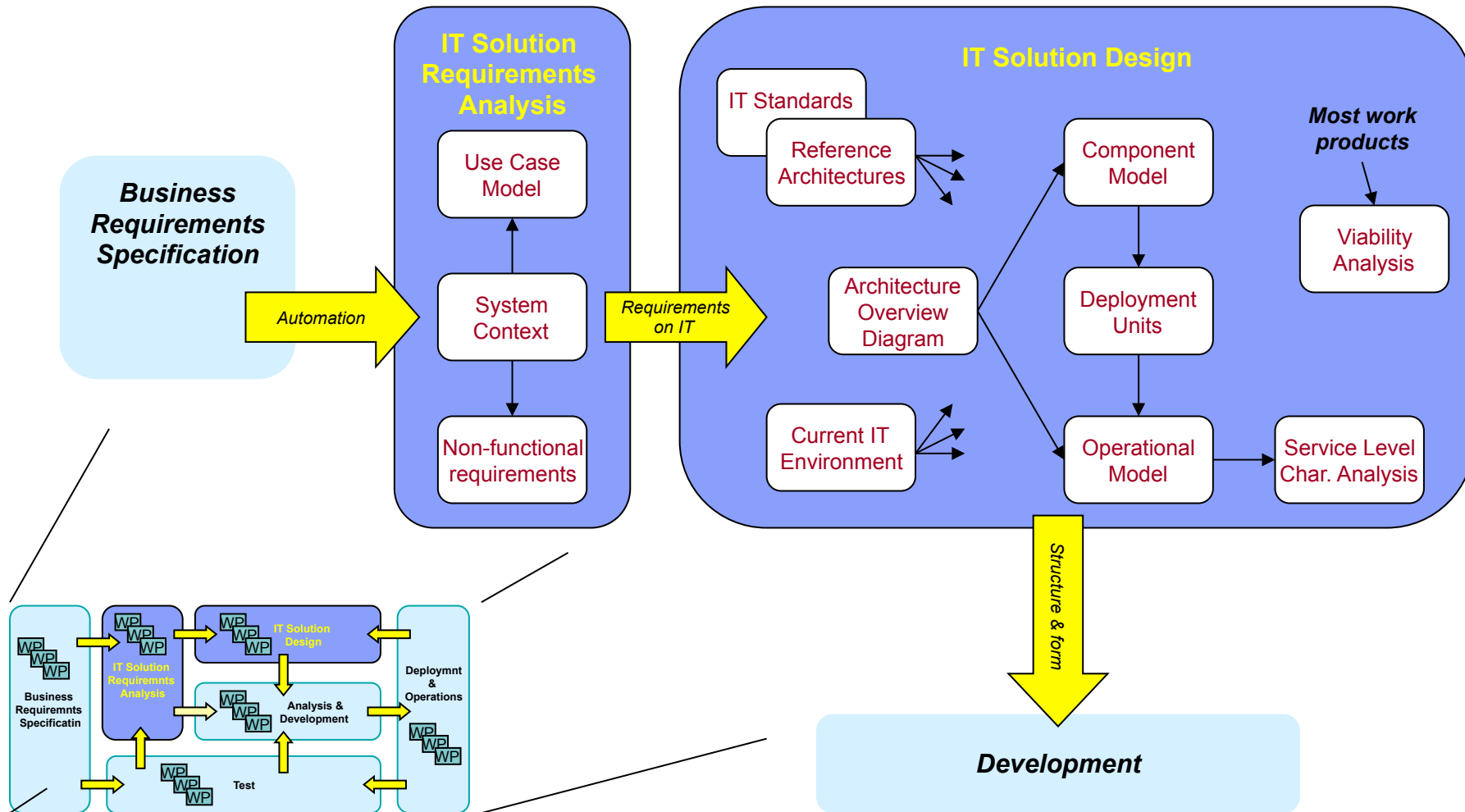
How do we manage this division of responsibilities, while accommodating interdependencies across the life-cycle of a project, without adopting “waterfall development”?

A work product is (usually) a document that defines some specific aspect of the solution and “lives” through the full life of the project

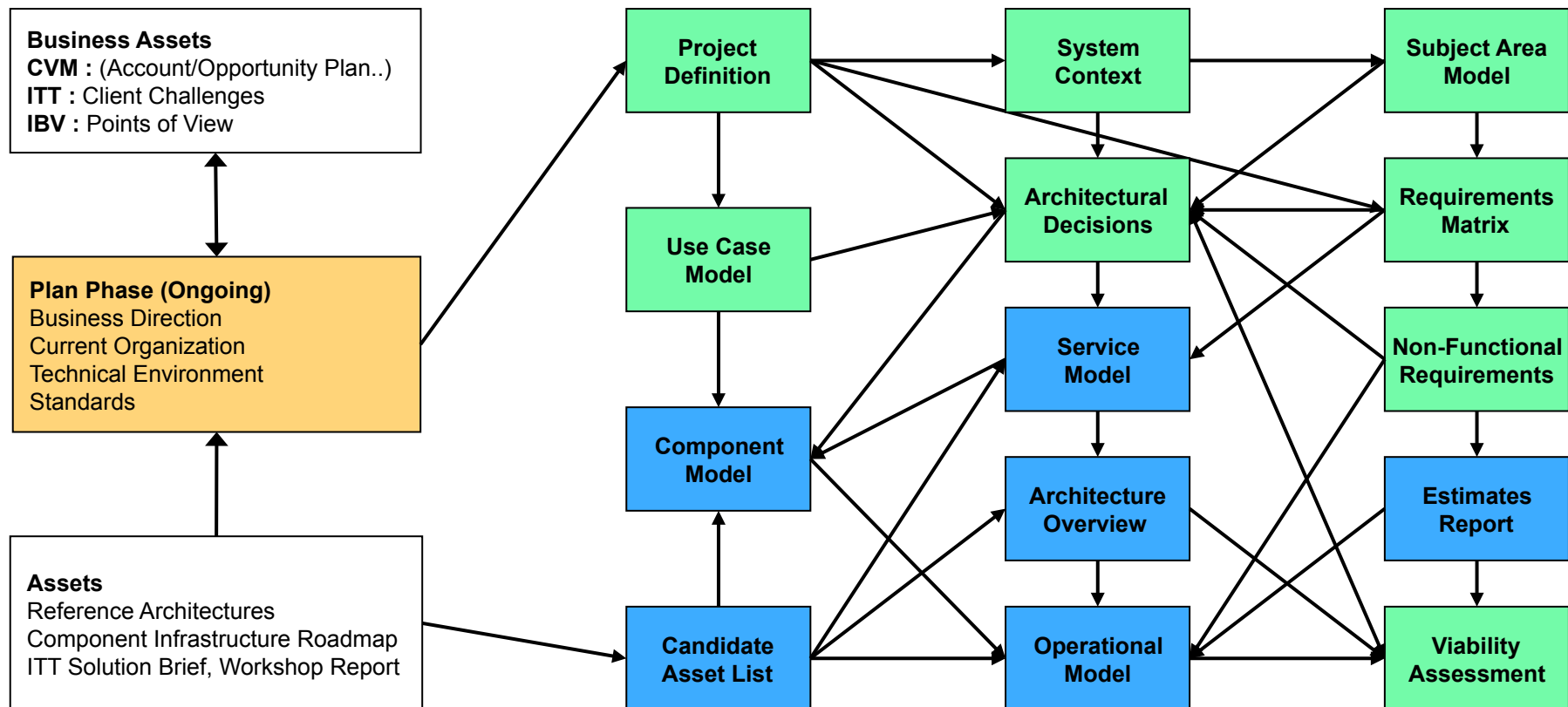
Enterprise IT Architectures

Separation of concerns

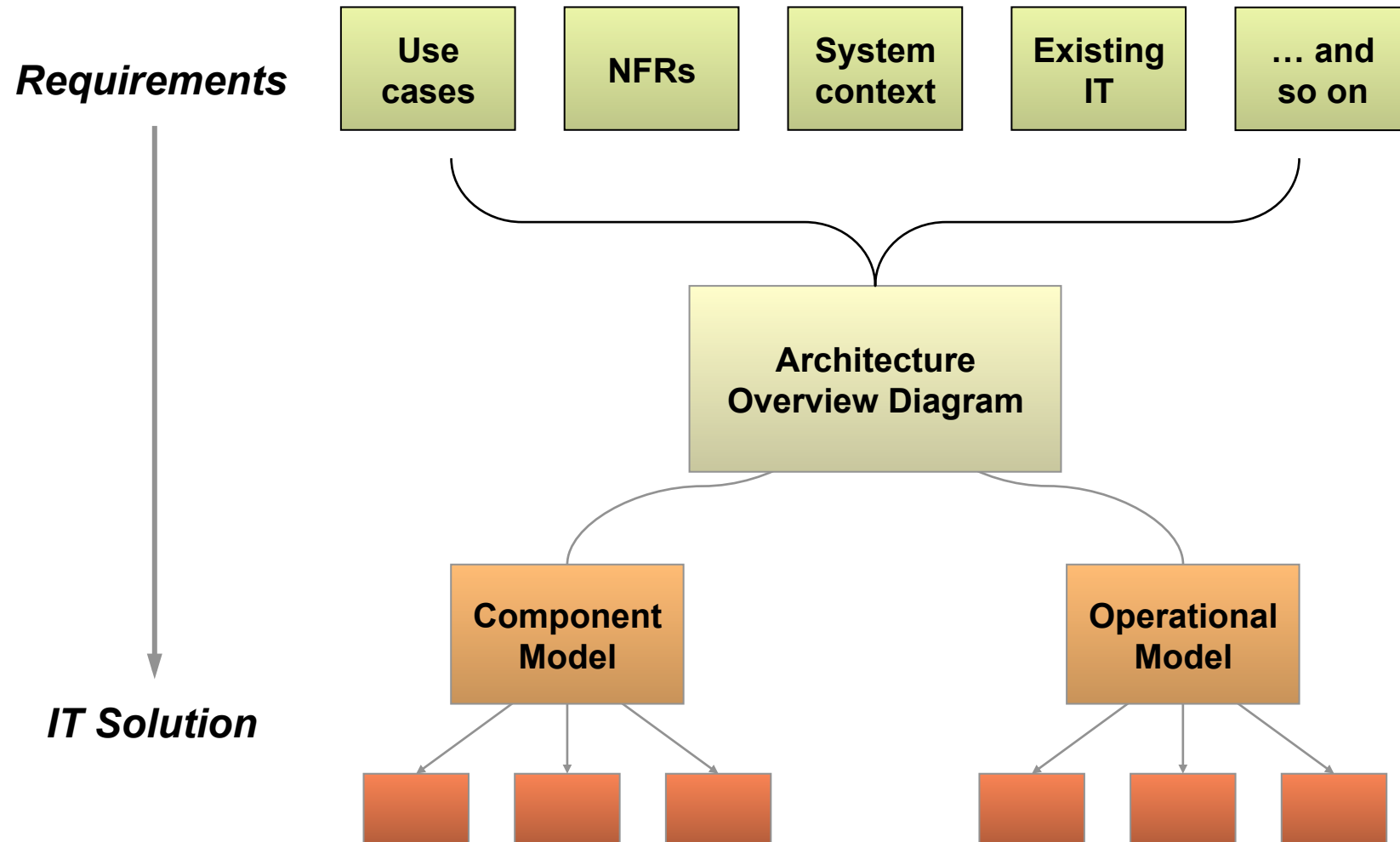
Defining and documenting the various aspects of the IT solution's requirements and design is achieved by using a set of IT Architecture work products, each focused on a specific view of the IT system



IT Architect's Tool Box: Work Product Dependency Diagram – Work Products are the Artifacts of Architecture Work



Overview Work Products – From Requirements to IT Solution

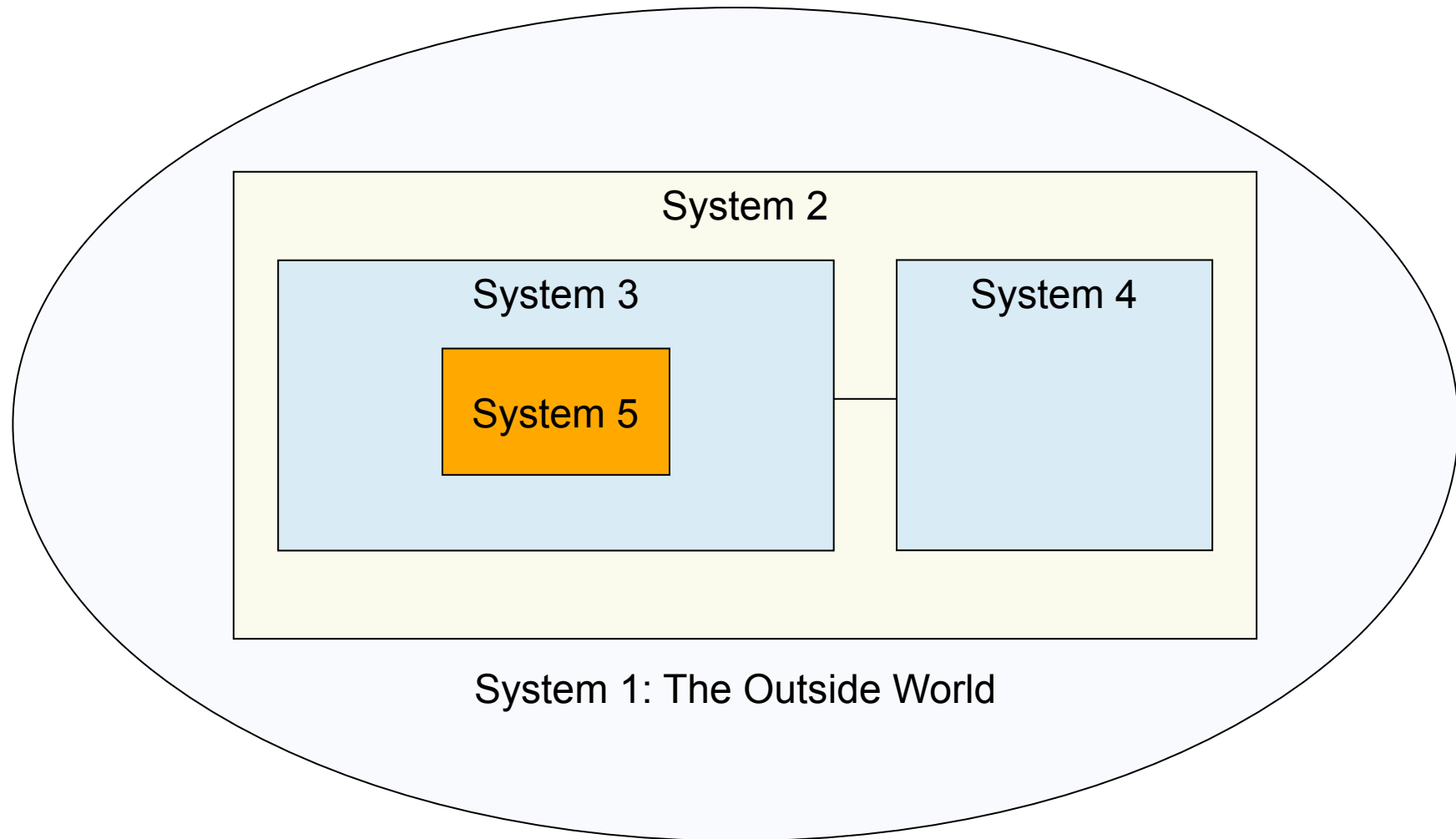


First Work Products (aka Artifacts)

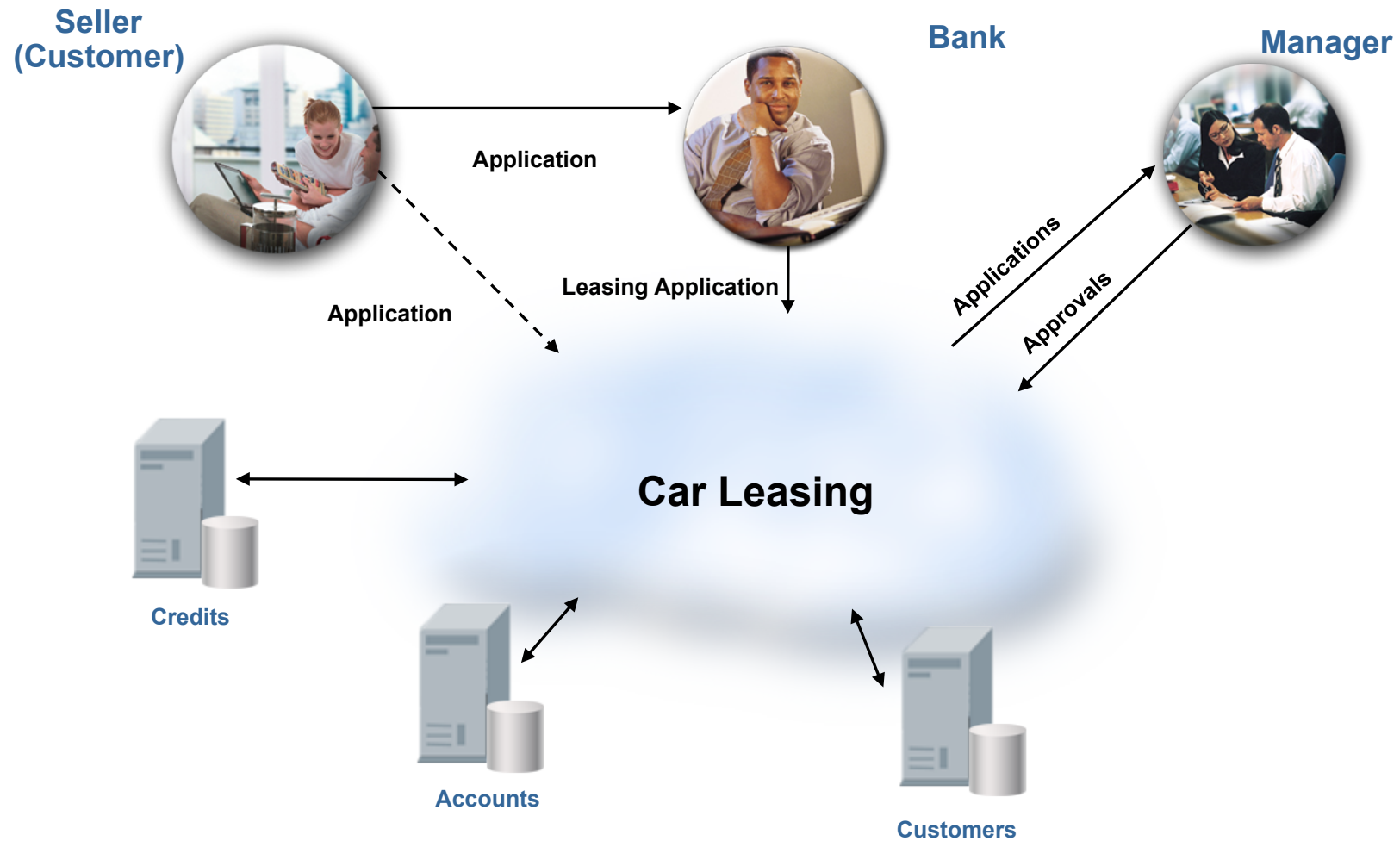
- **The three “C’s”**
 - **Context**
 - **Common Sense**
 - **Communication**

- **Context Diagram**
 - **External view showing the surroundings of the area to be addressed by the new solution**
 - **Helps to clarify the scope and thus is an important part of a project definition**
 - **Puts bounds on the system, identifies external interfaces (users or systems)**
 - **Addresses various stakeholders, can be used as Business Context as well as System Context**

Finding the system boundaries



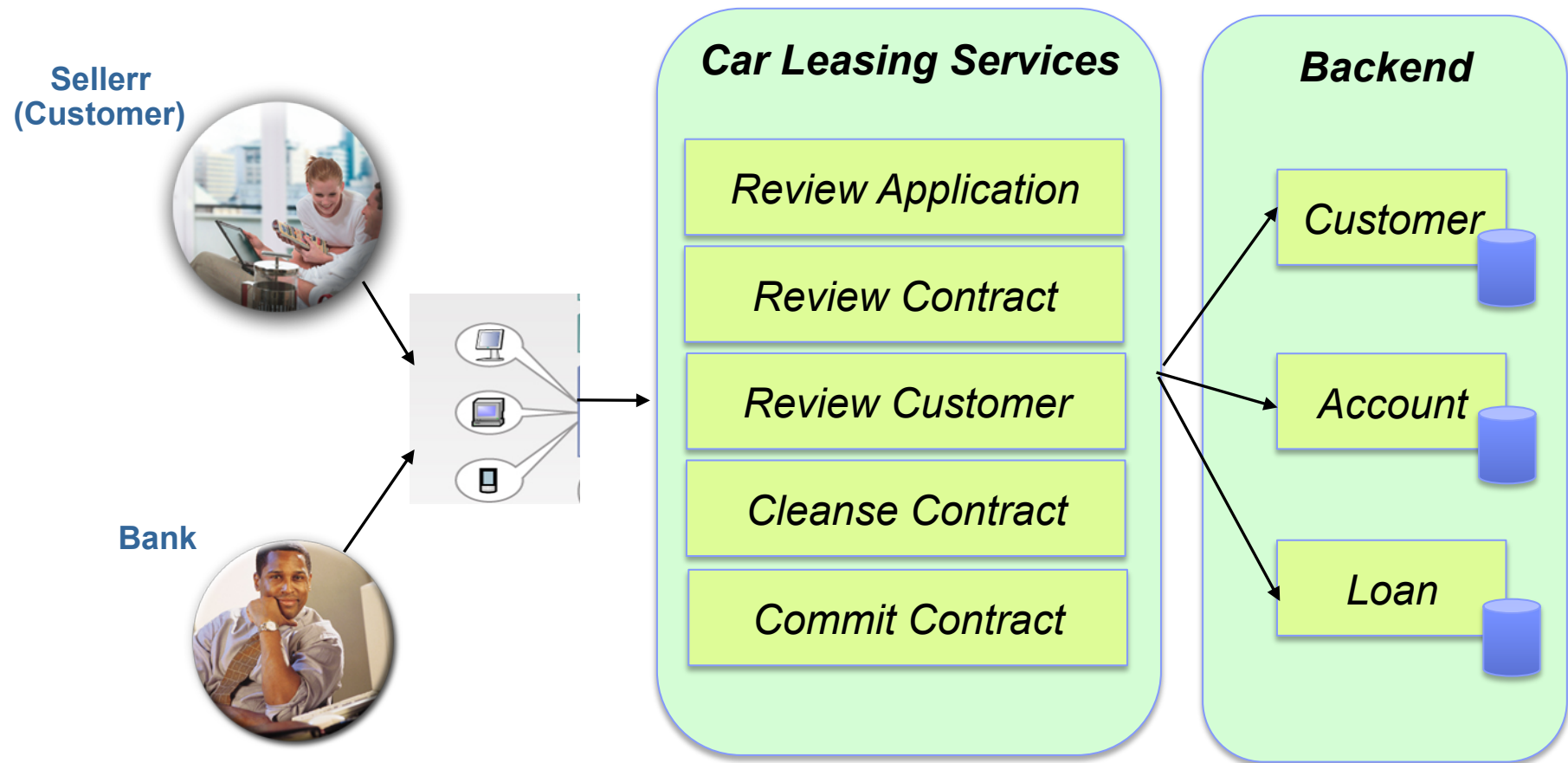
Context Diagram: Example Car Leasing



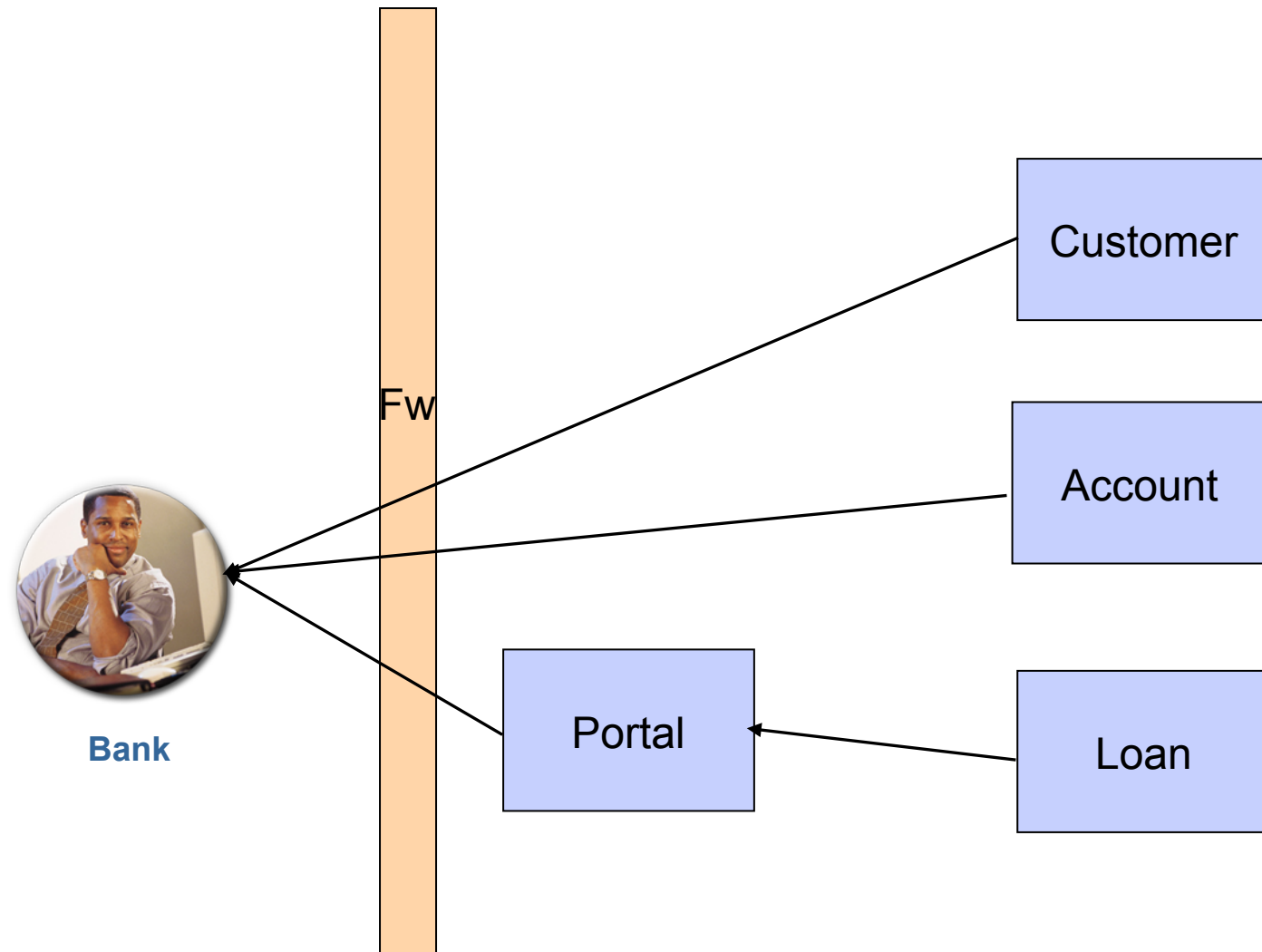
Architecture Overview Diagram (AOD) – Purpose

- To **communicate** to the **sponsor** and external stakeholders a conceptual understanding of the (intended) IT system (thus is a schematic diagram – not a model)
- To provide a **high-level shared vision** of the architecture and scope of the proposed IT system for the development teams (both functional and operational concepts)
- To explore and **evaluate** alternative architectural options
- To enable early recognition and validation of the **implications** of the architectural approach
- To **facilitate** effective communication between different communities of stakeholders and developers
- To facilitate orientation for new people who join the project

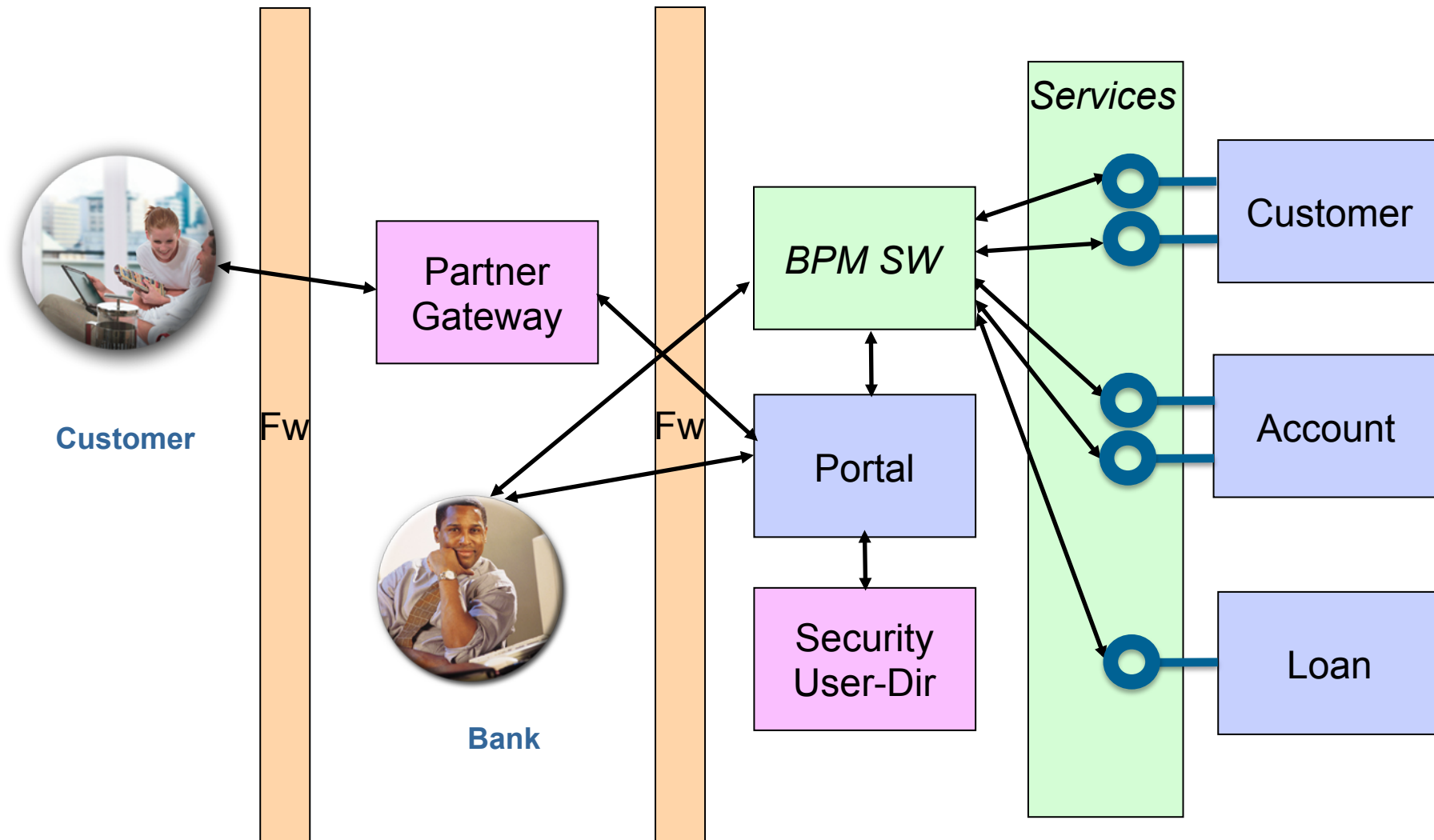
Example Car Leasing: Business-Oriented Architecture Overview



Example Car Leasing: AS-IS Architecture Overview



Example Car Leasing: TO-BE Architecture Overview



Functional Requirements

- **Functional Requirements describe the business functions required, they can be derived from the Business Architecture**
 - Requirements should be traced back to Business Goals
- **Functional Requirements:**
 - Are capabilities needed by users to fulfill their job
 - Answers the question of "what" does the customer want (but not "how" it is achieved)
- **Note: Requirements Engineering is a discipline for Software Engineering**

Non-Functional Requirements

- **Qualities:**
 - Define the expectations and characteristics that the system should support
 - Might be **runtime** (for example, performance or availability) or **non-runtime** (for example, scalability or maintainability)

- **Constraints:**
 - Givens, those things that cannot be changed within the scope and lifetime of the project
 - Other factors, such as mandated technologies, available skills, and budget

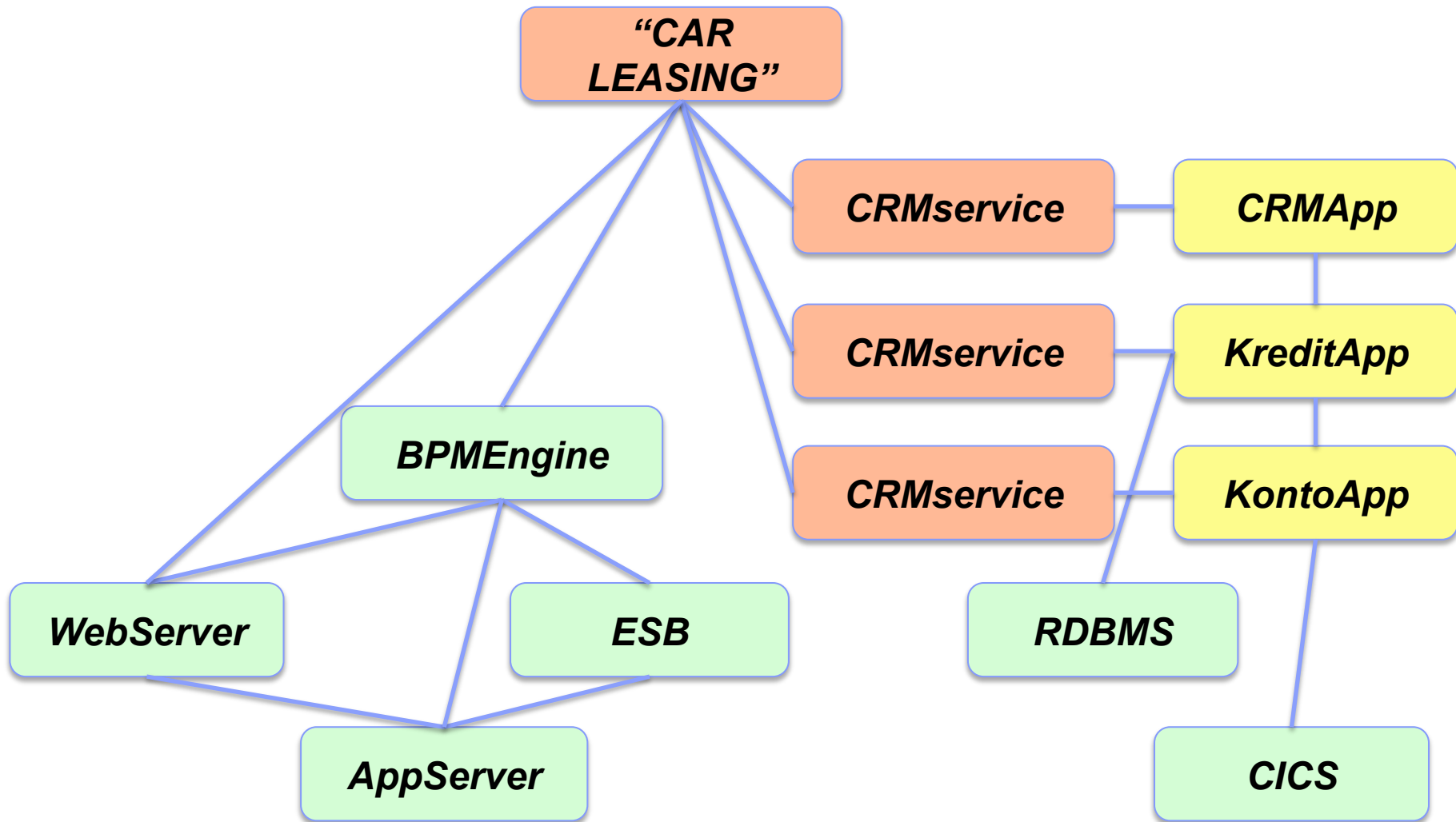
- **Qualities and Constraints are sometimes referred to as “non-functional requirements”**

Component Model

- **Components are defined with appropriate interfaces**

- **Dependencies**
 - **Relationships (can be shown in a relationship diagram)**
 - **Interaction**
 - **Collaboration**

Resulting Component Model “Car Leasing”



Operational Model

- **Focus on how particular systems works**
- **Concerned with the systems that will run in production**
- **Includes guidelines for the physical design and types of hardware**
 - **Nodes**
- **Addresses non-functional requirements**
 - **Performance**
 - **Availability**
 - **Security (e.g. firewalls, zones)**

For all of those: Using Patterns and Reference Architectures (e.g. SOA Reference Architecture)

- **A Pattern is a reusable generalization (or abstraction) that can be used as the starting point in future solutions.**

- **The benefits of Patterns are that they:**
 - **Provide a mechanism to capture knowledge and experience**
 - **Provide a common vocabulary among architects and designers**
 - **Facilitate reuse of approaches that have been successful elsewhere; thus, contributing towards the following aspects of a project by:**
 - Reducing risk**
 - Increasing quality**
 - Improving delivery time**

References

Overview Architecture Literature

- **Plenty of books about**
 - **Architecture,**
 - **Enterprise Architecture,**

- **Typical problems:**
 - **they are very generic, introducing architecture in general (and the technology specific books – e.g. about SOA – are that much “architecture minded”)**
 - **they do not provide concrete means (e.g. advice which diagram to use) because some of them are pattern-specific**
 - **They are very lengthy**

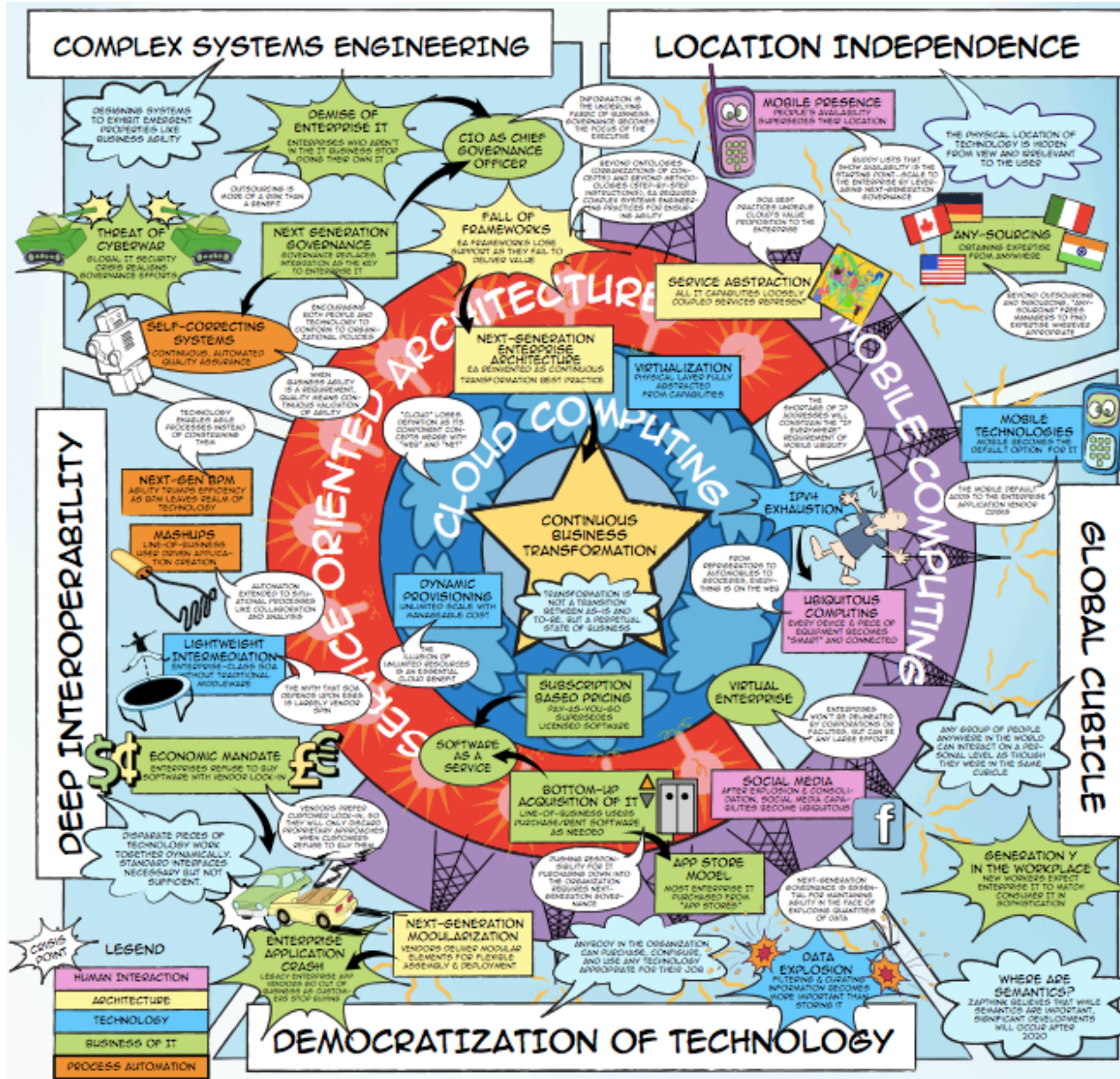
- **Thus: I’m writing my own book (in German)**

References – Books

- **Jason Bloomberg, The Agile Architecture Revolution, 2013, Wiley CIO Series, 280pp**
 - **Connects Service-Oriented Architecture and enterprise as a complex system**
 - **Considers Cloud Computing (does not work without architecture !) and REST (Representational State Transfer) for Service implementation**
 - **Advertises the ZapThink Vision 2020**
- **Peter Eeles & Peter Cripps, The Process of Software Architecting, Addison-Wesley Professional, 2009, ISBN 978-0321357489**
 - **Provides a good overview of work products**
- **Wolfgang Keller, IT-Unternehmensarchitektur, dpunkt.verlag, 2012, (German) – emphasizes Governance, 400 pp**

Enterprise IT Architectures

ZapThink's Vision for Enterprise IT in 2020



**References – Actual
(ZIP-File – will be available for Download)**

- **What do IT Architects do all day? IBM, 2006**
- **TOGAF 9.1 Reference Card, 2011, 2pp**
- **Object Management Group, Model Driven Architecture (MDA), MDA Guide rev. 2.0 OMG Document ormsc/2014-06-01, 15pp**
- **ZapThink's Vision for Enterprise IT 2020**

References – History (ZIP-File – will be available for Download)

- **J.A. Zachman, A framework for information systems architecture, IBM Systems Journal, 1987**
- **M. Maier, D. Emery, R. Hilliard: Software Architecture - Introducing IEEE Standard 1471**
- **IEEE Standard 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems, 2000**
- **Rechtin, Appendix A: Heuristics for systems level architecting**
- **R. Youngs et al., A standard for architecture description, IBM Systems Journal, 1999**
- **M. R. McBride, The software architect, Comm. ACM, May 2007**

Just remember – the future might bring more than you think

“I think there is a world market
for maybe five computers.”

Thomas Watson, chairman of IBM, 1943

“Computers in the future may weigh
no more than 1.5 tons. ”

Popular Mechanics, 1949

“There is no reason anyone would
want a computer in their home. ”

Ken Olsen, founder of DEC, 1977

“Prediction is difficult, especially
about the future”

Niels Bohr, 1957

“640K ought to be enough
for anybody. ”

Bill Gates, 1981

Questions

