CORONA EMERGENCY LECTURE

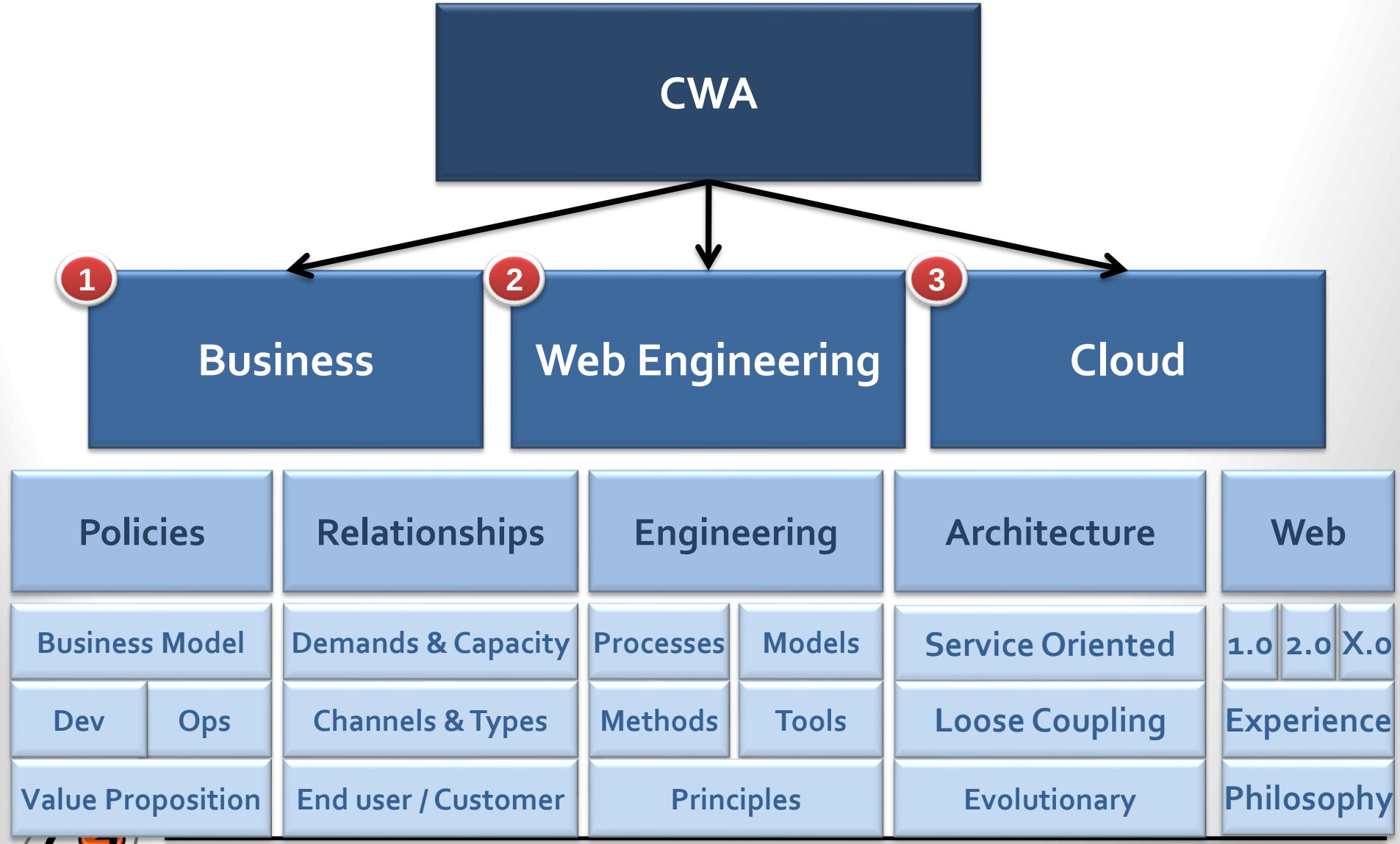
Cloud & Web Anwendungen

Prof. Dr.-Ing. Martin Gaedke
Technische Universität Chemnitz
Fakultät für Informatik
Verteilte und selbstorganisierende Rechnersysteme

PART III – SPECIFIC ASPECTS

- Web Engineering: Methods, Models, Tools

Lecture Outline



Part 3 – Overview

- **Understanding The Problem**

1. Introduction
2. Requirements
3. Design Thinking / Human Centered Design (HCD)
4. Requirements Engineering
5. Managing Requirements & Change
6. Further Readings

CHAPTER://8

- Introduction

Introduction

- Web Engineering projects are likely to fail due to misunderstanding the problem – as a result of "wrong" requirements

- Standish Group's CHAOS Reports:
  - 1994 and 1997 most significant contributors to project failure relate to requirements
- Computer Industry Daily / Sequent Computer Systems, Inc. study:
  - Most frequently named failure: "changing user requirements."

Requirements and the Web

- Common problems:
  - Requirements change
  - Requirements mean different things to different people
  - Requirements are not always obvious
  - Requirements are difficult to express
  - Requirements are related to one another
  - Requirements are driven by different parties
  - Requirements have different levels of detail
  - Requirements are not equally easy to meet
  - Requirements are difficult to gather
  - And there is even more…

Dealing with Requirements

- Web applications differ significantly compared to other software applications
  - Data-intensive, different Performance aspects
  - Highly distributed and heterogeneous environments
  - High demands for security and privacy
  - High demands for accessibility and usability etc.
  - Aesthetic aspects
  - Follow trends
  - High demands for Legal aspects
  - Internet Web application: User / Audience unknown
- Requirements will change at least if a Web application evolves

Dealing with Requirements II

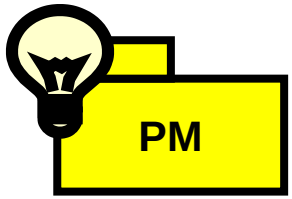
- **Defining and managing** requirements are a **major** success factor
  - Requirements mean different things to different people: Educate **all** project participants about this fact
  - Document every requirement
- This relates to CHAOS success factor No.1 to improve user involvement
  - Who are the stakeholders?

Stakeholders

- **Stakeholder** – an individual who serves as the primary source for some information that can affect the outcome of the project and/or who is affected by its outcome.
  - E.g. Customer, Employee, Marketing/Press, Administration, Customer Support, Content Creators, Domain Experts
  - Usually do not share a common understanding
- Known Stakeholder
  - Intranet & Extranet applications → interview stakeholder
- "Unknown" Stakeholder
  - Internet access → search for statistics
  - Later: user feedback, tracking and profiling for refining requirements

The Requirements Approach

- *Idea* – Apply a systematic approach to define and manage requirements for a desired software product in the WWW



CHAPTER://9

- Requirements Engineering

What is a Requirement?

- **Requirement – is**
  1) a **condition or capability** needed by a user to solve a problem or achieve an objective;
  2) a condition or capability that **must be met or possessed** by a *Web-based software product* or component to satisfy a contract, standard, specification, or other formally imposed documents;
  3) a **documented representation** of a condition or capability as in (1) or (2).

[Source: IEEE: *IEEE Standard Glossary of Software Engineering Terminology*, IEEE Standard 610.12-1990, IEEE, New York, 1983.]

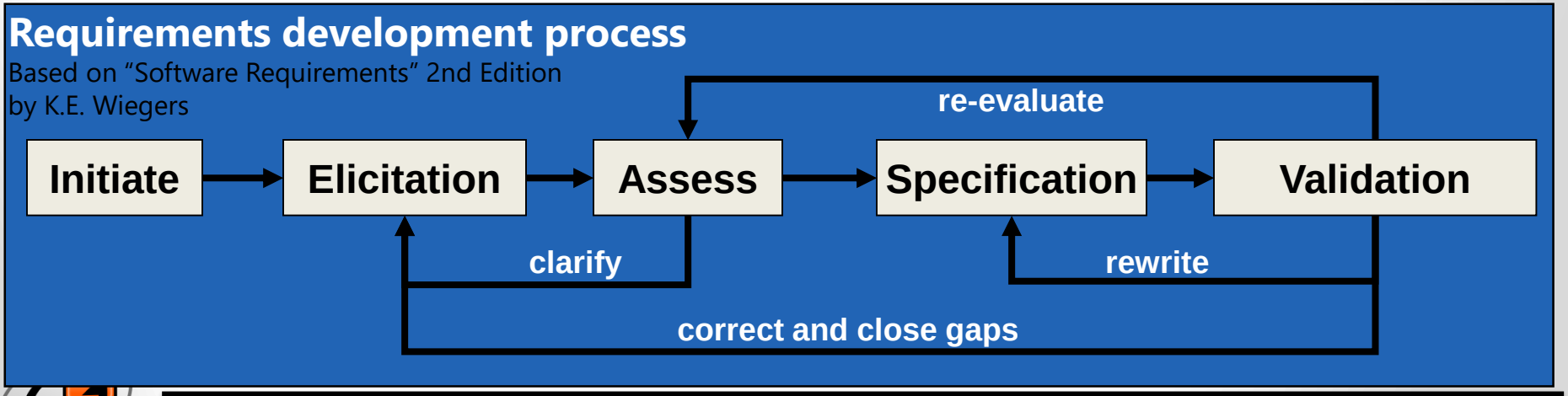
Types of Requirements

- **Functional requirements**
  - Fundamental subject matter of the system, i.e. something the product must be able to do
  - Measured by concrete means
  - E.g. Data values, class diagrams, algorithms
- **Non-functional requirements**
  - Behavioral properties that the specified functions must have, e.g. performance, usability, security etc.
  - Measured by specific means (the "nice-looking and easy to use problem")
  - E.g. Performance: resource response time < 2 sec with less than 100 users

Note: A lot of different notations and terms exist, try to understand the core idea

Requirements Engineering

- **Requirements Engineering (RE) –**
  - **is a recurring process** of the systematic, disciplined, and quantifiable application of approaches to define and manage the purpose and the external behavior of a proposed software product (here: in the WWW), and, which continues throughout the whole life cycle of the product
- RE includes *elicitation*, *analysis*, *specification*, *verification*, and *management* activities
  - Traceability is therefore a MUST
  - Cf. IEEE Std. 830 SRS and IEEE 828 Configuration Management Plans



Performing RE

- Requirements Analyst Role (RA)

  - Responsible for "clear communication" – bridging the gap between vague stakeholder notions and **clear specifications**
- Some skills required
  - Interviewing skills,
  - Listening skills,
  - Facilitation skills,
  - Observational skills,
  - Writing skills

Goal of RE

- The goal of requirements engineering is the production of a good **software requirements specification (SRS)** and the disciplined management of its evolution
- Aspects of a good Requirement *Statements* and *Specifications*:

Modifiable **Consistent** *Feasible* *Necessary* *Verifiable*

Traceable ***Complete*** *Precise* *Prioritized*

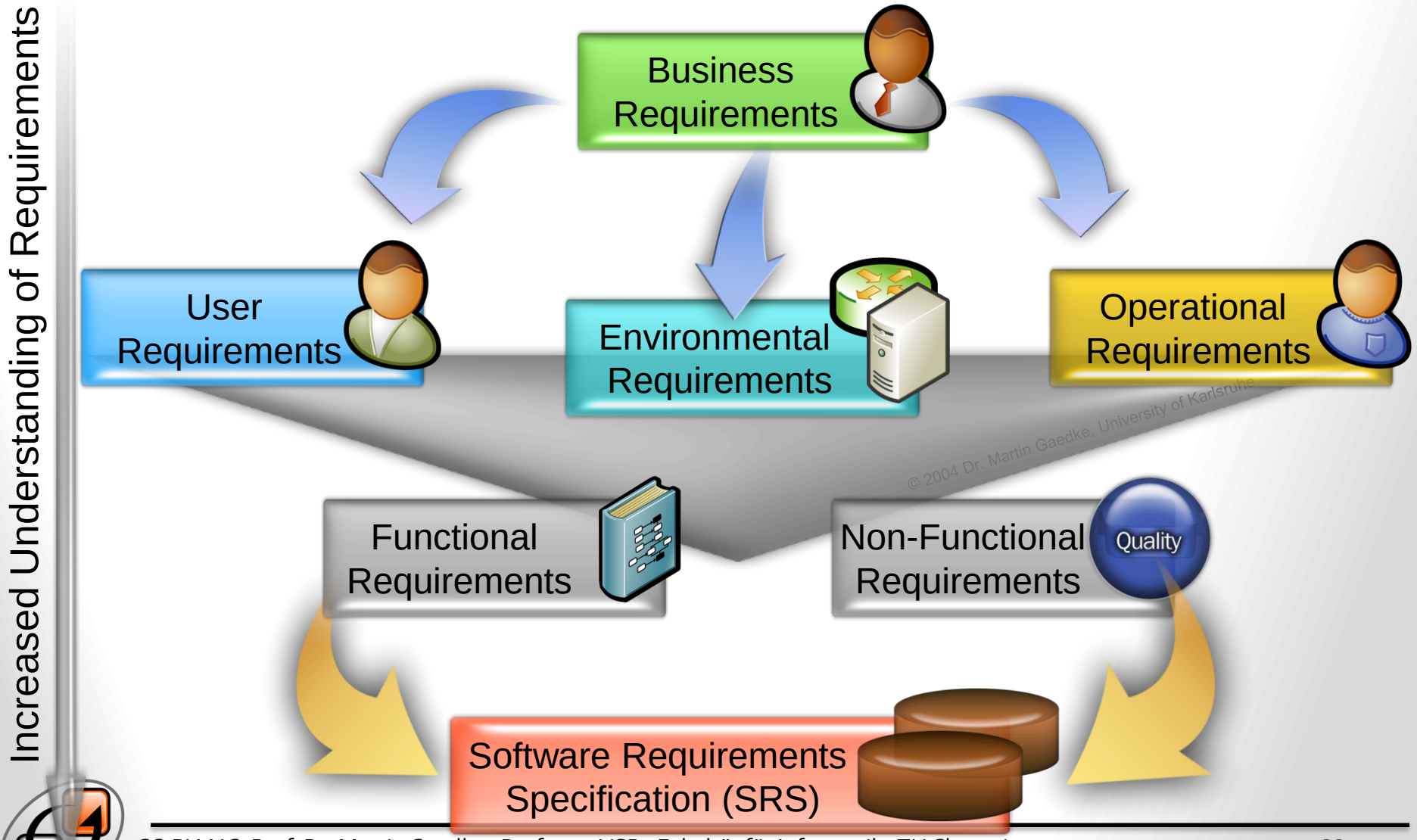
Versioning *Correct* *Unambiguous*

Usable during operations and maintenance

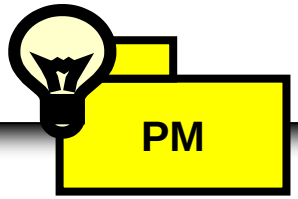
Levels of Requirements

- **Business** Requirements
  - ▶ High-level objectives of the organization or customer
- **User** Requirements
  - ▶ Tasks that users must be able to perform using the new product
- **Operational** Requirements
  - ▶ Tasks that operations staff must be able to perform using the new product
- **Environmental** Requirements
  - ▶ Aspects of the technology available and to be applied as well as the project's ecosystem

The Requirements Big Picture

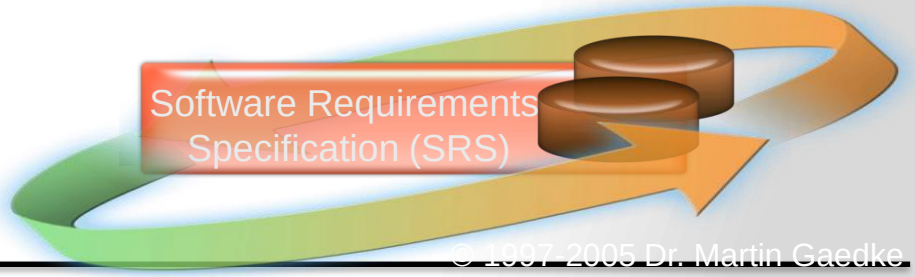


RE & Project Life-Cycle



Project evolves (Change Management)

1. **Prepare for RE activities**
   - Team Mind-Set, Glossary etc.
2. **Initiate Phase**
   - Business Requirements under change control
3. **Elicit Phase**
   - Gathering information – Refine Requirements
4. **Assess Phase**
   - Transition from gathering to analyzing, preparation for further phases (including tests for finalization of the project)
5. **Continuous evolution**
   - **Change Management**

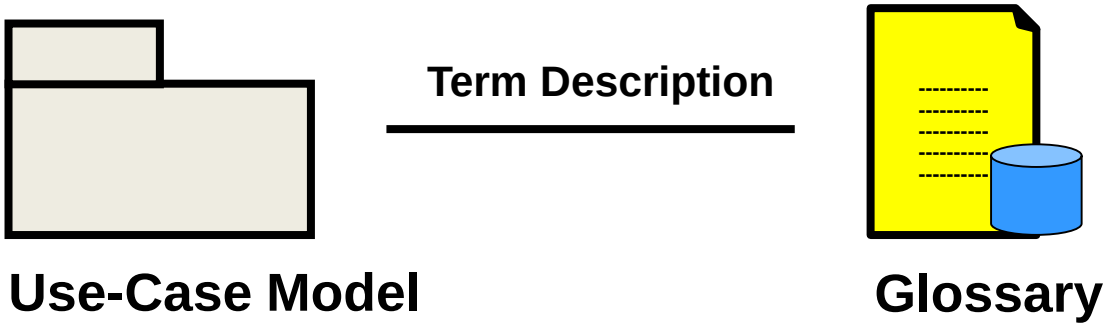


Prepare for RE Activities

- Structure project team adequate
  - Roles, e.g. requirements analyst (RA), quality assurance, standards-specialist, domain-expert, CCB
- Project team must be aware of RE
  - Common vocabulary
  - Procedures for proposing, reviewing, and resolving changes
  - Status reports
- Prepare to elicit
  - Observe your customer/users
  - Train for interviewing (Journalism School?)

Prepare: Glossary

- Defines important and common terms of a domain
- Used by analysts and developer to support same language
- System analyst or use case specifier responsible for use case glossary



SECTION://1

- Initiate Phase

Objective

- **Initiation Phase** – Initiation is the *process of formally authorizing a new project*
  - Links project to the performing organization
  - Formal document and output of this phase:
    Project Charter (issued by manager)
    - Remember: There is no official project if this document is missing
    - Other projects are unofficial ones

- Usually formal initiation after completion of further assessments
  - Functional Specification
  - Feasibility study
  - Etc.

Developing Solution Concept

- Identify business opportunity
  - The first customer meeting
  - Interview with key personnel, senior management (CEO, CTO, CIO)

- Gather business requirements
  - Use Cases and interview techniques
  - Focus on business improving processes, boundaries, external relationships, key business process stakeholder

Use Cases

- **Use Case (UC)** – A (possibly ordered) set of actions, including variants, that a system performs that yields an observable result of value to a particular actor
  - ▶ UC complete from the outside actor's view
  - ▶ Bulleted form, written structured:

Natural Language, nothing else – just that

Note: Be aware of the many different definitions of similar or related terms

Use Cases Realizations & Co.

- **Use Case Realizations**: Different ways to carry out a use case
- **Use Case Scenario**: Single path through a use case
  - A story telling us how something (usually a business process) is done, i.e. a particular combination of conditions within that use case
- **Use Case Diagram**: puts UCs and Actors into a graphical context

- Very effective with Senior Management

Drafting the Vision/Scope

- Analyze and consider business impact
  - Review (current and future) business processes and opportunities
  - Identify Stakeholder
  - Typical categories: common, specific, competitive, etc.
- For Internet-Sites
  - Create market requirements document (MRD)
  - MRD describes requirements of market segments, e.g. audience characteristics / profiles

Drafting the Vision

- Develop a first Vision statement
  - Strategic plan – long-term view on solution
  - Focus on business requirements
  - Aligns all stakeholder in common direction
  - Cf. For-Who-The-Is-That-Unlike-Our Product example
- Major Features
  - Labels and Names for the major capabilities of the product
- Assumptions, Dependencies, Legal Issues
  - Describe assumptions etc. mentioned by stakeholders

Example: Vision eConcierge

- **FOR** guests of the hotel
- **WHO** need assistance in enhancing their stay by choosing restaurants or cultural events
- **THE** eConcierge Service (eCS)
- **IS** a portal
- **THAT** will provide an overview of events, activities and selected restaurants (partners)
- **UNLIKE** the current black-board approach
- **OUR PRODUCT** will provide ubiquitous assistance from the early beginning of the ordering process as well as during the stay by allowing to access the system with different devices

Based on “Software Requirements” 2nd Edition by K.E. Wiegers

Drafting the Scope

- Define a first Scope statement
  - Decomposition of Vision into "Business Feature Sets"
  - Sum of high-level deliverables and services to be provided
  - Manageable chunks
- Scope of Initial Release (Version 1.0)
  - Focus only on those features that will provide the most impact
- Limitations and Exlusion List
  - List what might get in and what is definitely out of scope (core rules for the CCB)

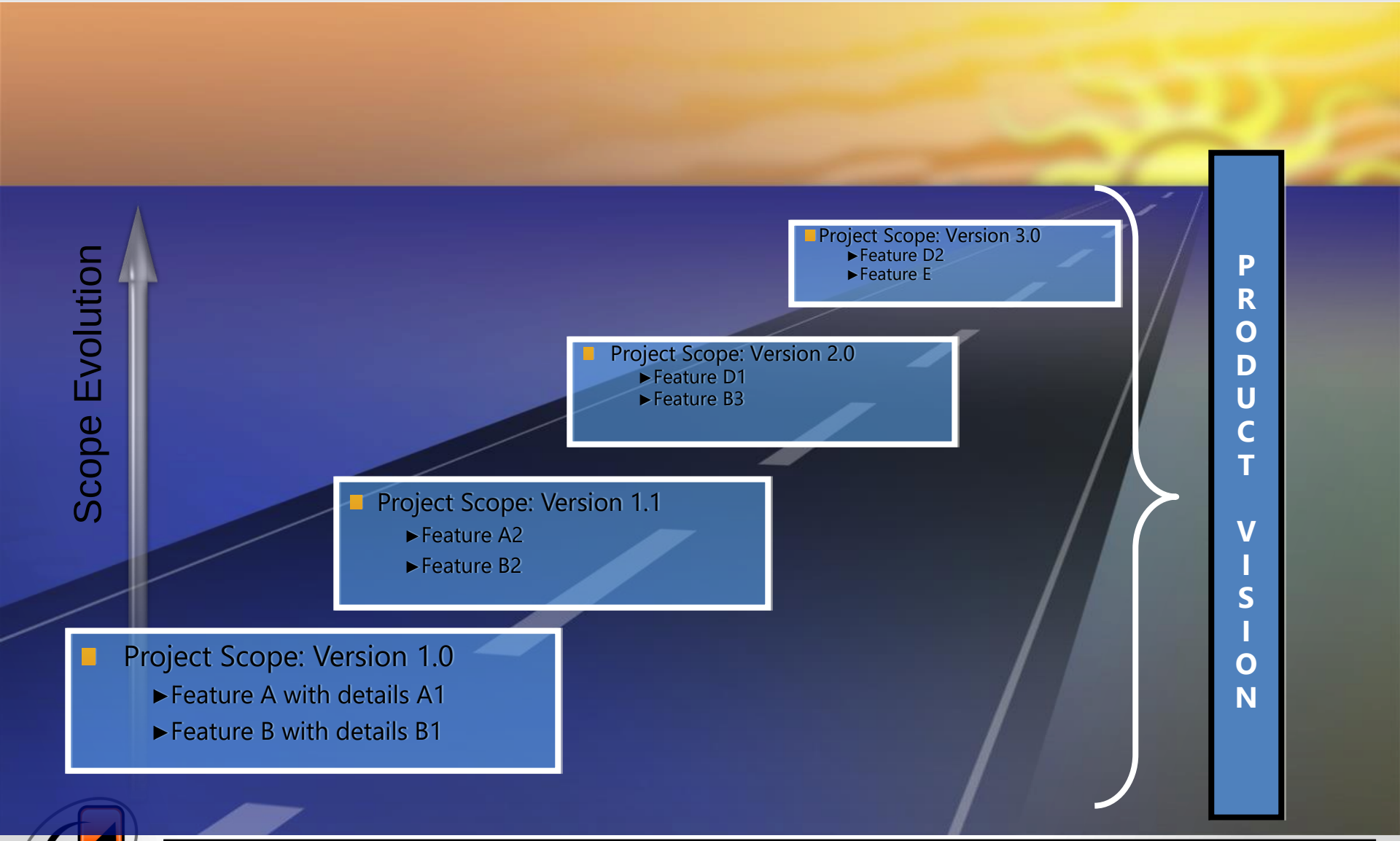
Product Scope

- **Statement of Requirements (SOR)**
  - SOR will be the document against which change control will be exercised
  - Sometimes called Functional Requirements Document (FDS) - set of statement of requirement
- Prepare for iterative approach (multi-versioning)
  - **Baseline** current version
  - Baselined → Put under change control

Project Scope

- Work that must be done to deliver a product with the specified features and functions
- Aka **Statement of Work (SOW)**
  - Narrative description of products or services to be supplied under **contract**
  - Sufficient detail required to allow team to determine if capable of providing the item

Vision and Scope



Prepare Project Initiation

- Prepare Project Initiation Documents
- Prepare risk assessment document
- Check for readiness: Personnel and training needs, possibly expert judgment
- First assignments

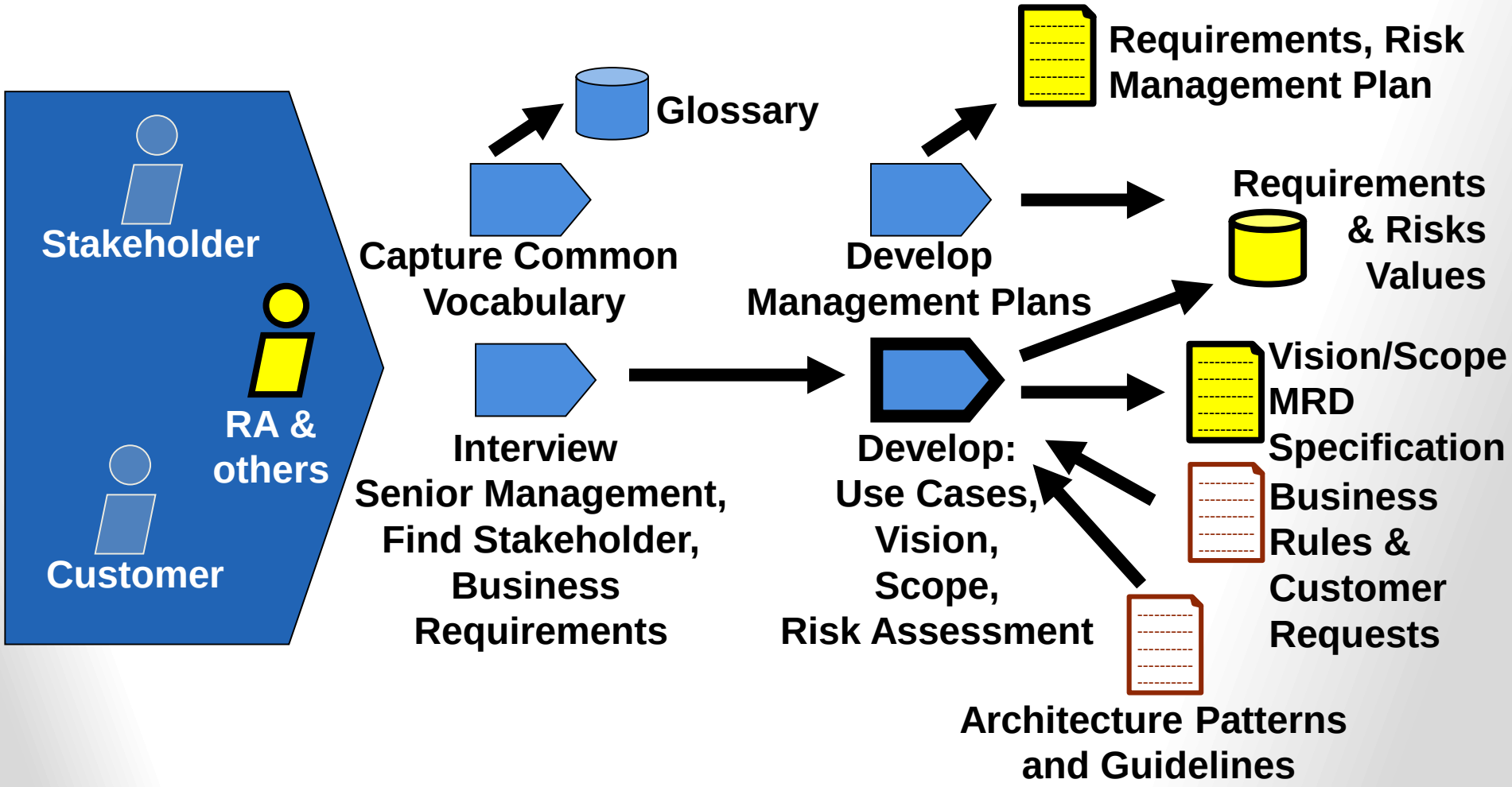
Finalizing Initiate Phase

- Analyze and discuss benefits and first draft documents
  - Vision and Scope Verification by stakeholder
  - Note: ALL Requirements gathered are high-level requirements and **will be further refined or even changed**

M

- Initiated: Commitment to begin the next phase
  - Project Team – Customer:
    - Memorandum of Agreement (Not a contract)
  - Project Team – Management:
    - Project Charter

Initiate Phase – Summary



SECTION://2

- Elicit Phase

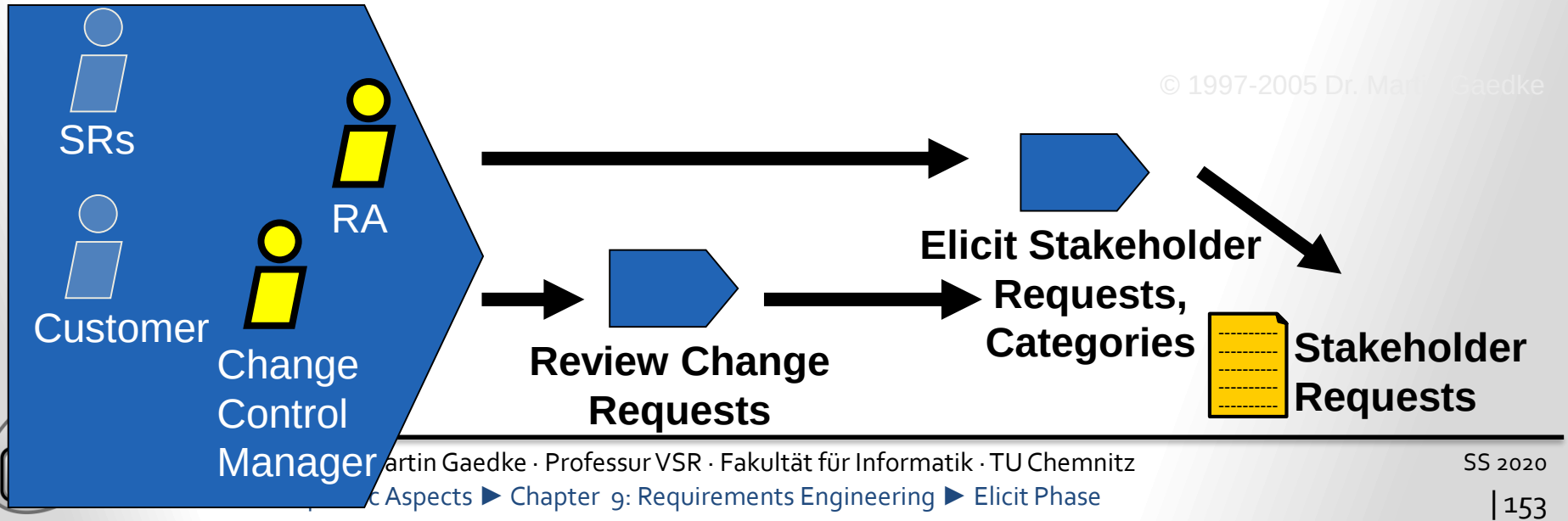
Introduction

- Goal:
  - Refining the business requirements
  - Better understanding the problem
  - Enhancing Vision/Scope Specification by refining the scope

- Refine Requirements for the solution
  - Business Requirements (**why**)
  - Functional Requirements (**what**)
  - Non- Functional Requirements (**how**)

Performing Elicitation

- Identify further stakeholder &
  Find stakeholder representative (SR)
  - ▶ SR aka Product Champion
  - ▶ Elicit stakeholder requests
  - ▶ Categorize information as well as SRs
- Ongoing Process → change control



Better Understanding

- Gather information in a holistic manner
  - Stakeholder (Personnel and Training Needs)
  - User profiles and audiences
  - Organizational structure (Both current and projected)
  - Market / Industry position
  - Organizational political climate
  - Business reach or scope
  - Current and future regulatory requirements
  - Product boundaries, constraints
- Requires finding stakeholder representatives
  - The question to solve: What would X need to do?
  - Be aware of implicit users

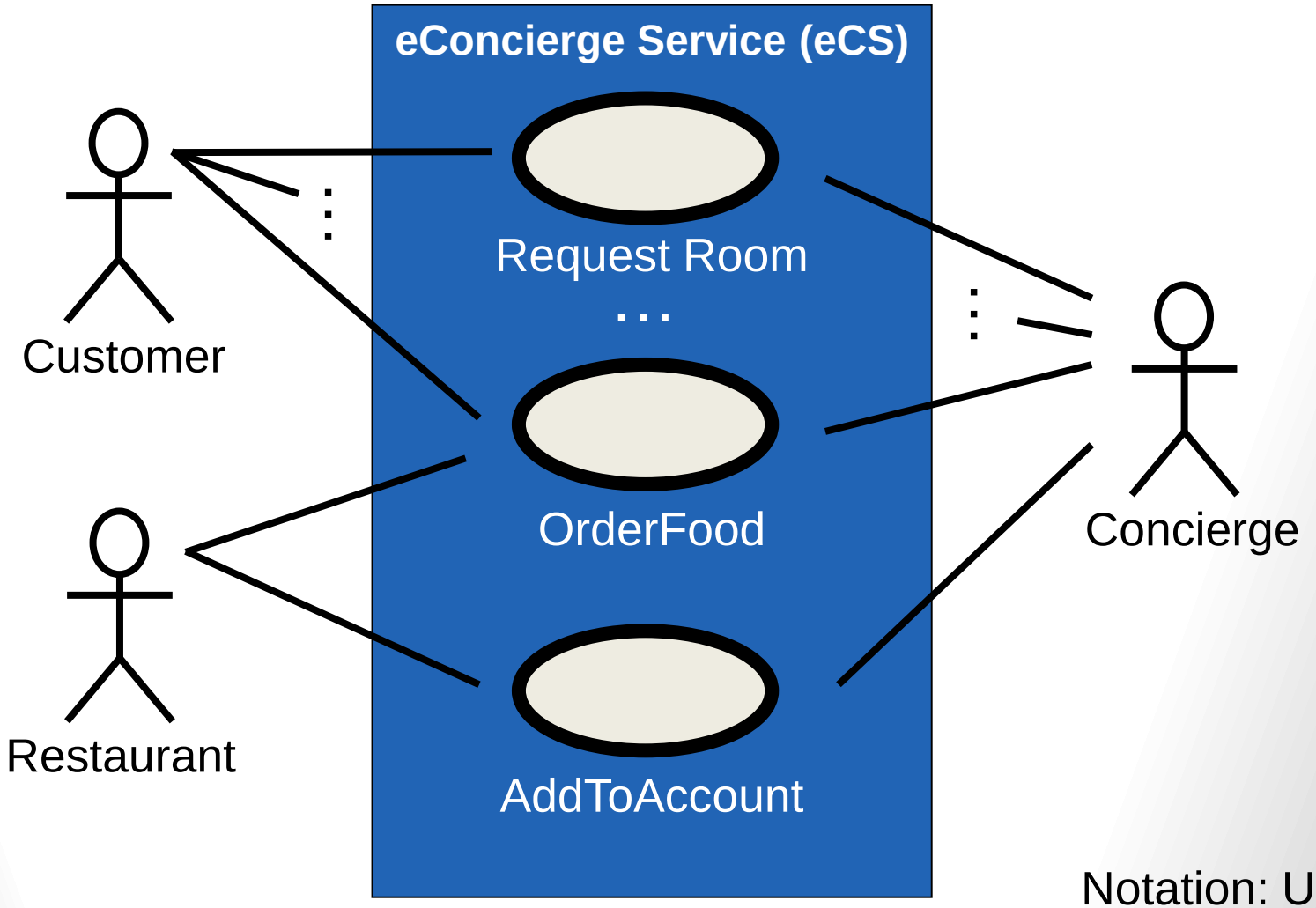
Refining Scope

- Refining high-level business requirements
  - Rewriting Use Cases
  - Focusing on Use Case Scenario
  - Rules of thumb: Describe workflow not just purpose, all possible processes within a business use case, only those inside the business, only relevant ones

- Transforming – Graphical Notation
  - **Use Case Diagram: puts UCs and Actors into context**

Use Case Diagram

Business Requirements



Notation: UML

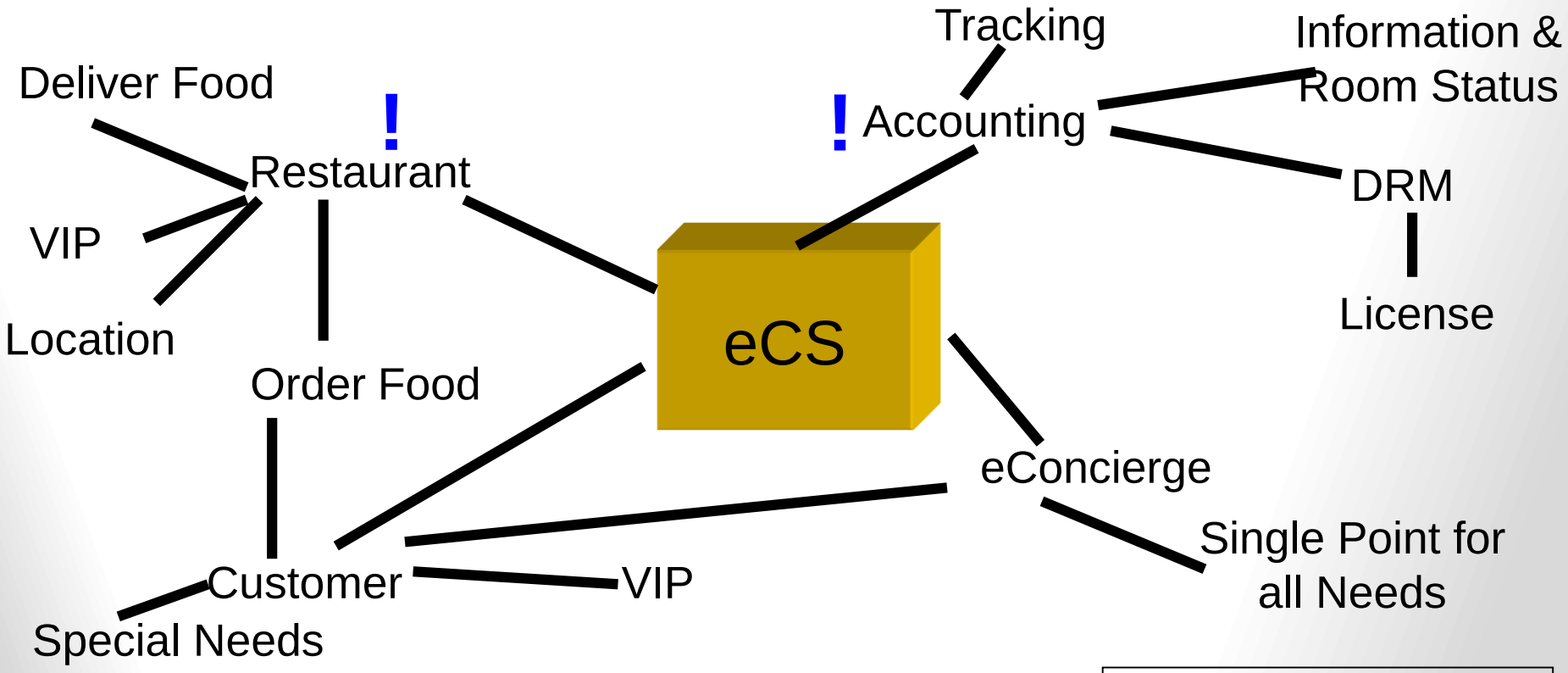
Techniques For Gathering

- Standard approaches:
  - interviewing, shadowing, focus groups / group interview, brainstorming sessions, surveys, user instruction, prototyping, statistics
- Working with Subject Matter Expert (SME)
  - Class Responsibility Collaborator Modeling

Class Name (e.g. Person, Place, Thing, Event)	
Responsibility (what Class knows and does)	Collaborators (classes to work with)

Mind-Maps

- Structuring the results of Brainstorming and Interviewing sessions



DRM – Digital Rights Management

Categories of Information

- Based on existing Business Requirements
- Identify and classify Requirements with focus on
  - ▶ Business
    - Focus on **Data and Semantic (CONTENT)** and how it is used (**Process/Workflow**)
  - ▶ User
    - Focus on **User-Interface Experience (UIX)**
  - ▶ Operations
    - Focus on the **System Management and Operation (SMO)**
  - ▶ System
    - Focus on the **Process and Communication Aspects (DSA)**
- And enhance the accuracy of Vision/Scope and SRS documents

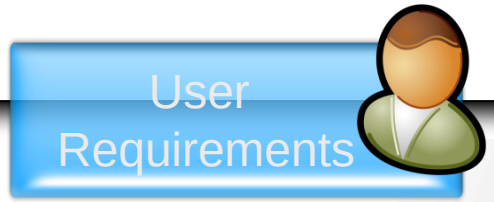
Business

- Example from Business Requirements
  - "I need to order rooms" vs. "Only selected restaurants are allowed to add to the account"
  - This is: What vs. Who/How
- What: Functional Requirement
  - User Requirement: Customer
- Who/How: Non-Functional Requirement
  - Business Rules: Selected restaurants

Business

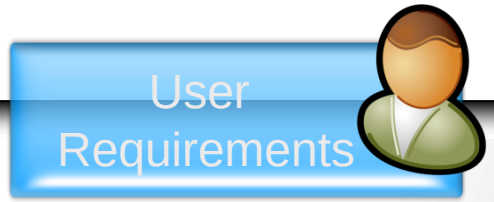
- Prioritize projected business processes
  - Will determine order and construction of planning and development process
  - Identify internal/external dependencies
- Business data (Content)
  - Describe data/entities/objects, refine glossary
  - Describe semantics – check relations to data flow/workflows
- Data flow (Process and Interaction)
  - Specify the flow of data from a business perspective
  - E.g. DeMarco & Yourdon, Gane-Sarson model or UML analysis model
  - This will be important input for DSA

UI Experience



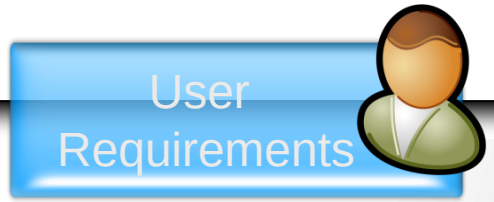
- (User Interface) Design
  - Design is manifested in appearance of the Web solution
  - Visual ornamental characteristics embodied in, or applied to, chunks of information as well as their composition
- Relates to dimensions:
  - Presentation – Deals with appearance of the Web application, i.e. layout, color, fonts, links, etc.
  - Navigation – Application of the hypermedia paradigm to access information or perform tasks
  - Dialogue – Relates to interacting (including manipulating) the information space
- User-Centered Design puts these three dimensions into context with the overall tasks the user has to perform to fulfill his/her business goals

UIX: Experience



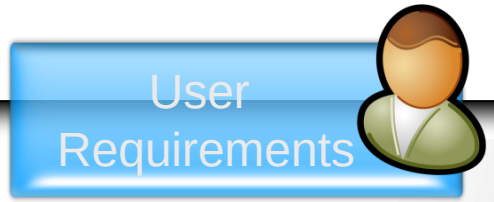
- Focus on Overall Feel of the Site
  - Who will use it? → audiences/user context
  - Where is it used? → location context
  - How is it used? → task/job context
  - Which devices? → technical context (UA restrictions)
- Cultural aspects of using the application
  - I18N, L10N, G11N
  - Identify localization requirements
  - Globalization requirements

UIX: Accessibility



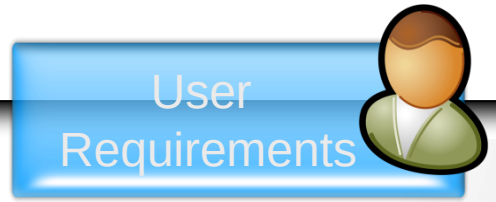
- Accessibility Requirements
  - http://www.w3.org/WAI
    Web Accessibility Initiative
  - http://www.w3.org/TR/WCAG
    Web Content Accessibility Guidelines
  - Web Accessibility Design
- User-Agent/Device specific Requirements
  - E.g. the use of specific handheld devices is required

UIX: Navigation



- Relationship-Navigation Analysis
  - ▶ Michael Bieber et al., NJIT, NJ, USA
- Intensify hypermedia mindset of stakeholders and teams
  - ▶ Promote a hypermedia mindset
- Process RNA steps (simplified)
  - ▶ Element of interest analysis
  - ▶ Relationship analysis
  - ▶ Navigation analysis

UIX: RNA's Taxonomy



Generic Relationship

- Internal
  - Generalization/ Specialization
  - Self
  - Whole-part/ Composition
  - Classification/ Instantiation
    - •Characteristic
    - •Descriptive
    - •Occurrence
    - •Configuration/Aggregation
    - •Membership/Grouping
- External
  - Association/ Dependency
  - Comparison
    - •Ordering
    - •Activity
    - •Influence
    - •Intentional
    - •Socio-organizational
    - •Temporal
    - •Spatial
    - •Equivalence
    - •Similar/Dissimilar

Operations

- Identify Requirements with focus on operation of the solution, e.g.

Efficiency
Availability
Security
Scalability
Testability
Maintainability
Reliability
Interoperability
Deployment
Monitoring & Healing
Robustness
Reusability
Policies
Integrity

- These often relate to **non-functional requirements**, rules or quality attributes
  - Help in specification of SLA, OLA

Technology

- Integration and Performance Requirements
  - What software, data, existing systems (legacy) will be accessed
  - How well or how rapidly a system must perform
- Solution elements of the distributed system
  - **Hardware**: Computer, firewall, router, switch, network etc.
  - **Operation**: Operating system, network topology / VLAN etc.
  - **Hosting**: Web server, messaging server, database server, etc.
  - **Core Services**: Web services, mail, UDDI, IP/STS, location srv
  - **Application**: Application composition, integration and federation
- Analyze the Impact of solution elements on the IT environment and vice versa
  - E.g. Bandwidth, Cache, (required) response time, connections
- Focus on the **Distributed System Aspects (DSA)**

Updating Vision/Scope

- **Update Vision/Scope Specification**
  - Update product scope
  - Draft project scope
  - Possibly: Gain customer agreement
  - Start Assess Phase

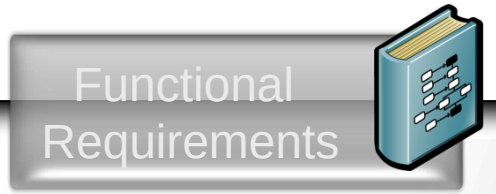
SECTION://3

- Assess Phase

Assessing Requirements

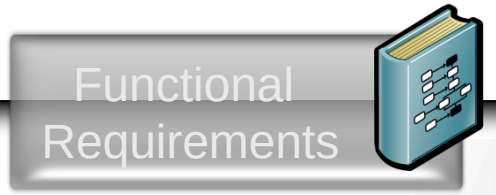
- …is about understanding and organizing requirements
  - In the general software engineering sense
  - With a dedicated focus on the product dimensions of distributed Web-based systems
- Review of functional requirements
  - Focus on product dimensions and features
- Dealing with non-functional requirements
  - Business rules
  - Quality attributes

Product Dimensions

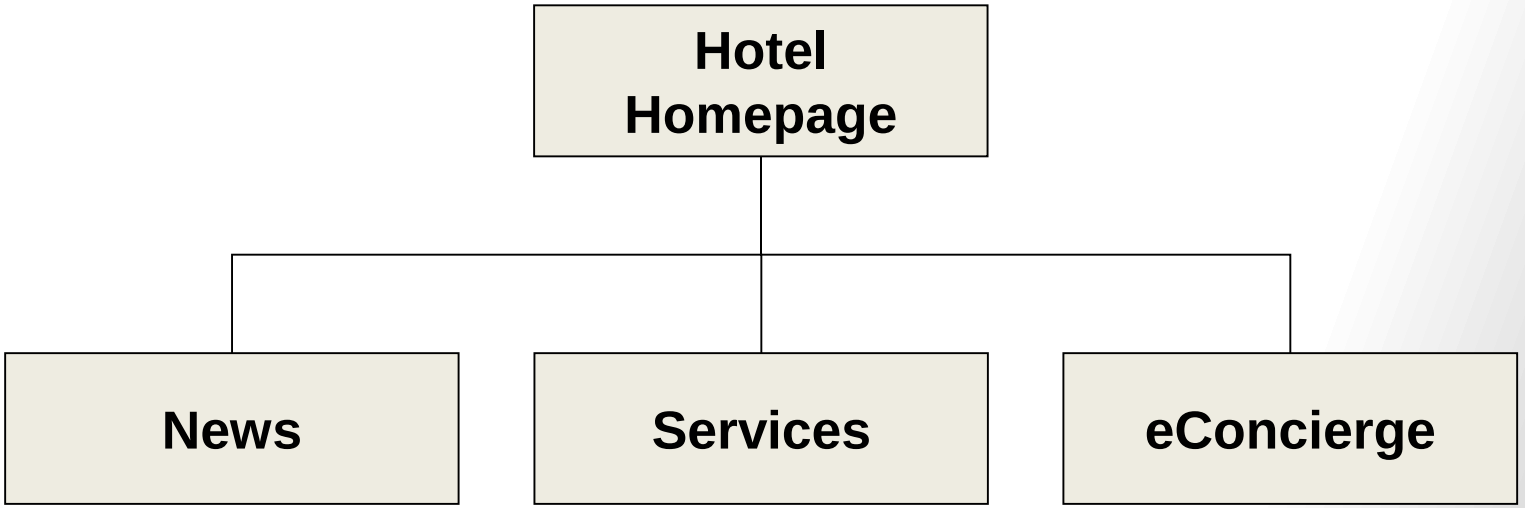


- Review and (if possible)
- Classify Functional Requirements with focus on:
  - **Content**
    - Data
    - Semantics
  - **User Interface Experience**
    - Presentation: Layout & Design
    - User Interaction: Forms & Input Devices
    - Navigation: Links & Well-known navigation structures
  - **Distributed System Architecture**
    - Process (check Business Processes)
    - Communication

Prototyping



- Proof-of-concept
  - Vertical Prototype
  - For some parts: From interface to process components, like the real-system will do
- Mock-ups, Sample-Screenshots
  - Horizontal Prototype
  - Initial navigation / presentation concept

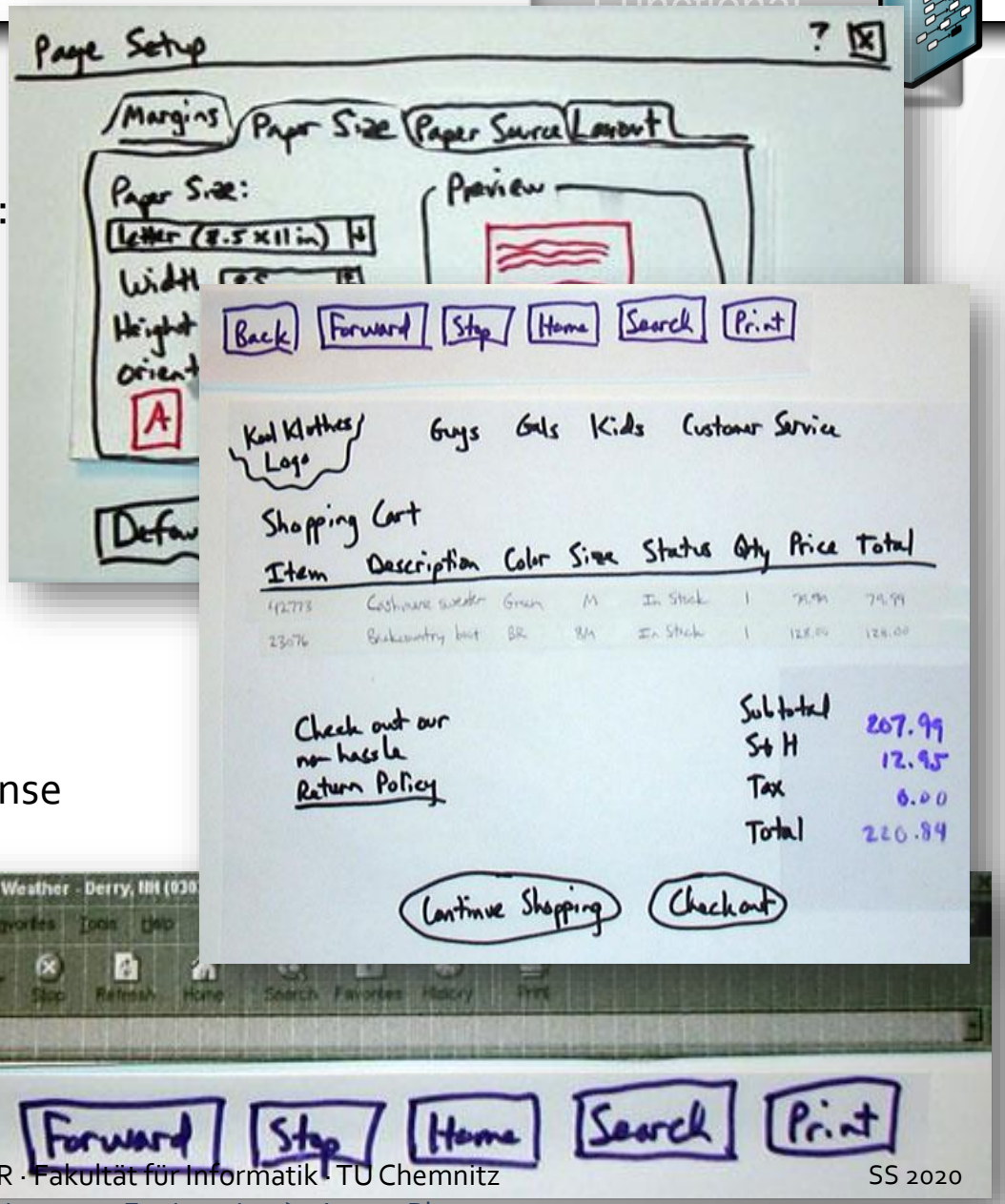


Prototyping

- Prototyping is for Risk Reduction
- Further approaches
  - Throwaway prototypes
  - Paperware
  - Storyboards
  - Illustrations

Paper Prototyping

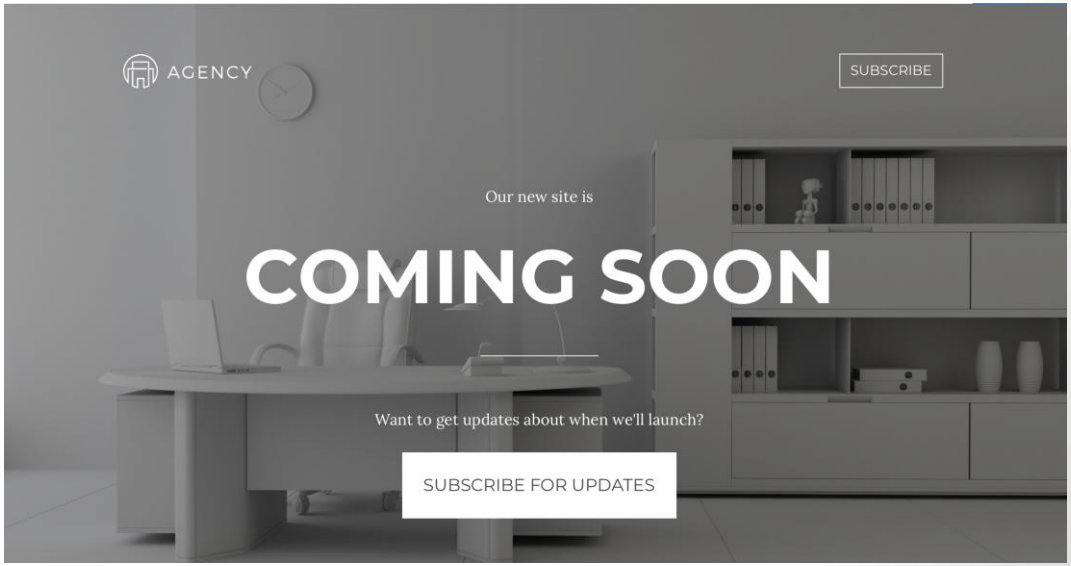
- Usability tool
- Useful for gathering data about:
  - Concepts and terminology
  - Navigation/workflow
  - Does the interface provide the right information to make decisions?
  - Page layout
- Not ideal for:
  - Technical feasibility
  - Download time or other response time
  - Colors and fonts
  - Scrolling



http://www-106.ibm.com/developerworks/library/us-paper/

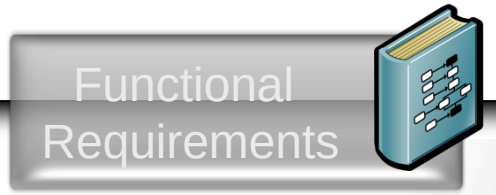
Prototype by using Landing Page

- What is it?
- What is the benefit?
- How do we do it?



Example from: https://www.leadpages.net

Features



- **Feature** – Functionality that the solution must deliver to be complete
  - Based on the functional and non-functional requirements of business and user requirements
  - Describe: benefit, increase customer or user satisfaction
  - Provides a name for a part of the scope
- **Feature Set**
  - Set of features that belong together to support a certain need
  - May evolve and developed in several scopes

Business Rules

- Approach to non-functional requirements:
  Business Rules Statements
  - Define or constrain aspects
  - Assert business structure
  - Control or influence the behavior
  - Define parameter for System Management and Operation (SMO), e.g. for SLA

- **Business Rules Taxonomy**
  - Facts (Invariants), e.g. all pages with terms of use
  - Constraints, e.g. MUST, SHOULD etc. rules
  - Action Enablers (Trigger) , e.g. if x then y
  - Computations, e.g. Orders > 100 EUR → item price decr. 5%
  - Inferences, e.g. if Web Service does not respond within 2 sec. then the services is out of order

Quality Attributes

- Labeled Collection of quality related findings
  - Efficiency, Availability, Scalability, Integrity etc.
  - Focus on the most important ones
  - Usability, security, performance, reliability and reusability are major issues!
- Example:
  - PERF-1: Homepage rendered < 1 sec with following browsers on the following hardware
  - PERF-2: Page download < 2 sec over DSL
  - USAB-1: Concierge and trained staff shall be able to submit all eConcierge forms
  - USAB-2: Restaurant pages must be WAI-compliant

Requirements Prioritization

Feature / Feature Sets	SR1 Benefit	SR1 Penalty	SR2 Benefit	SR2 Penalty	…	Total Value
F1. Order a room	4	7	5	1	…	21
F2. See list of selected restaurants	3	1	7	3	…	40
F3. Add restaurant costs to account	2	5	9	1	…	21
F4. ...	…					

- Requirements Prioritization Matrix a living document
  - Different complex solutions exists
  - E.g. Quality Function Deployment (QFD) or Total Quality Management (TQM) approaches
- Prioritize by VALUE, RISK, READINESS

SR: stakeholder representative

SECTION://4

- Specification and Validation Phases

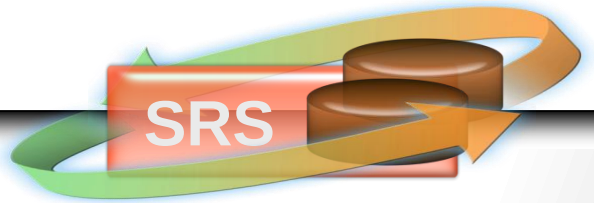
Documenting and SRS

- Documenting Requirements
  - SRS - many people will rely on it
  - Maintainable and ready for change
- Dealing with change
  - Label Requirements: Sequence number, hierarchical numbering or textual tags
  - Prepare for the unknown (To Be Determined / TBD-Process)
- Dictionary helpful to use terms in SRS consistently (use existing Glossary)

Specification of Feature Sets

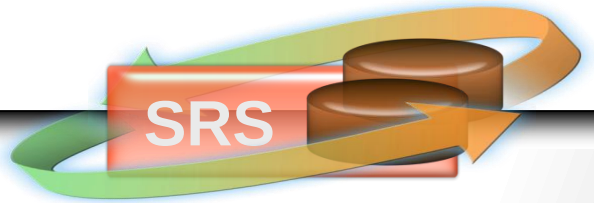
- Feature Sets and the corresponding features
  - Abstract complexity
  - Abstract many different scenarios for using the feature / the system
- **Specification** of the features should include (incomplete list)
  - Name
  - Description (Short and precise – one sentence if possible)
  - Trace: Who raised this feature (who/where/why)
  - Resources: team (responsible, owner), costs, time etc.
  - Priority, like required (in v1), wanted (in v1), wanted (in v2)
  - Dependencies, risks and possible solutions (e.g. existing prototypes)
  - Storyboards explaining the use of a feature
  - History, knowledge (helpful people and material for creating the feature)
  - **Test criteria (when is a feature ready to deploy)**
    - Customer/User/Operation satisfaction and dissatisfaction
    - ***Tests (interface, test programs, complete storyboards with test data&results***

Prepare for Planning



- Check **feasibility** of project direction / vision
- Define tasks, schedules, deadlines, WBS, reports etc.
- Prepare risk assessment document
- Check for readiness: Personnel and training needs, possibly expert judgment
- First assignments

Manage Scope



- Check scope every Major Milestone
- Activities
  - Assign values to requirement attributes
  - Plan further progress with project and development management
  - Focus on highest risk requirements
  - Major refinement necessary?

Vision/Scope Specification

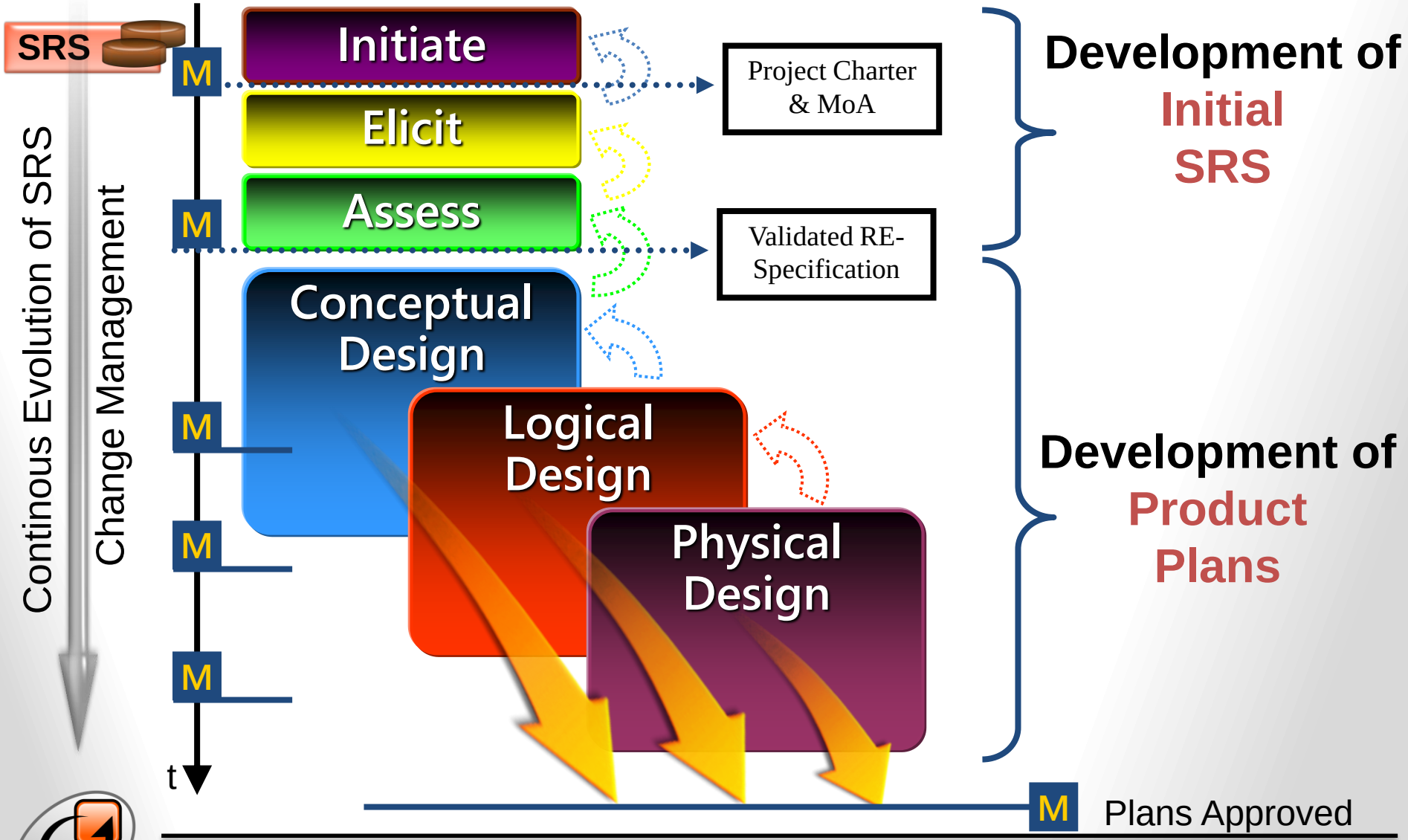
- **Vision/Scope Specification Complete**
  - Inspect with stake holder and gain customer agreement
  - Vision/Scope document baselined
  - Initial SRS developed and baselined
  - Start Planning
- Study: Most successful projects spent 10-30% of total project resources until this milestone

- From problem description to plan
  - Transform customer into developer language

Beyond SRS

- From SRS to Plans, Designs, Codes etc.
- Shift
  - Shift from problem definition to solution design
  - ***Shift from customer language to developer language***
- SRS guides process
  - Provides a framework
  - Source for plan, schedule and build solution
  - Serves as contract between team and customer

SRS and Planning



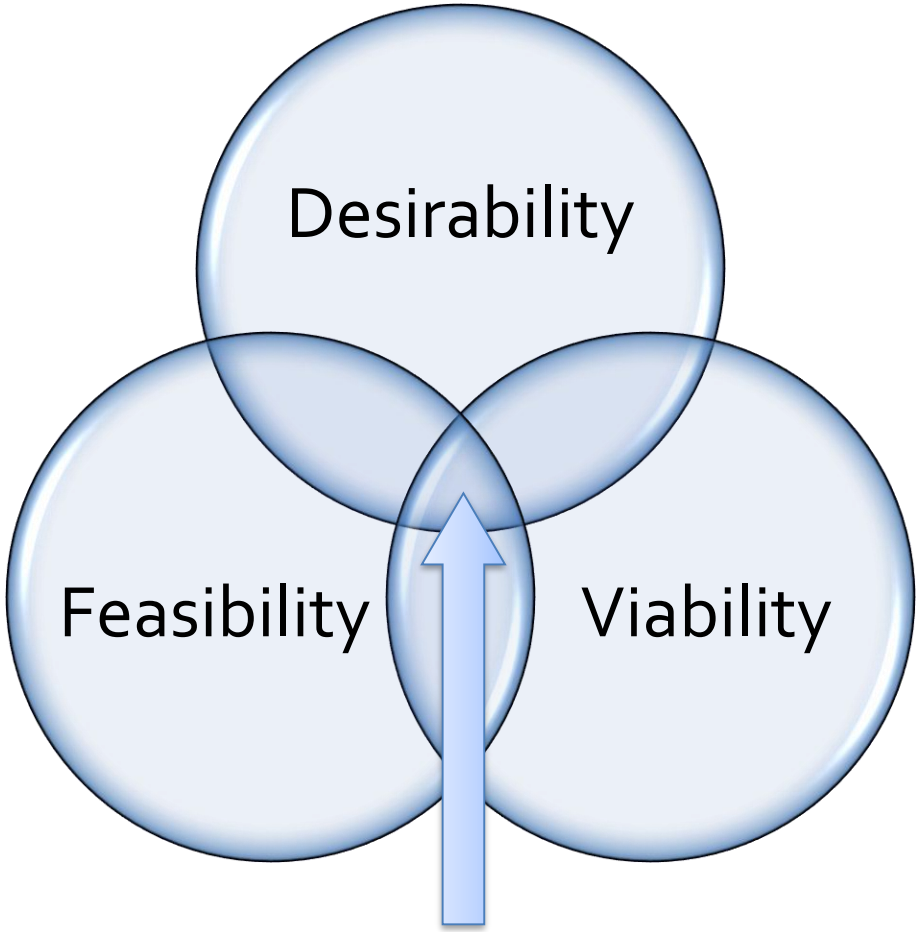
CHAPTER://10

- Design Thinking / Human Centered Design (HCD)

Overview

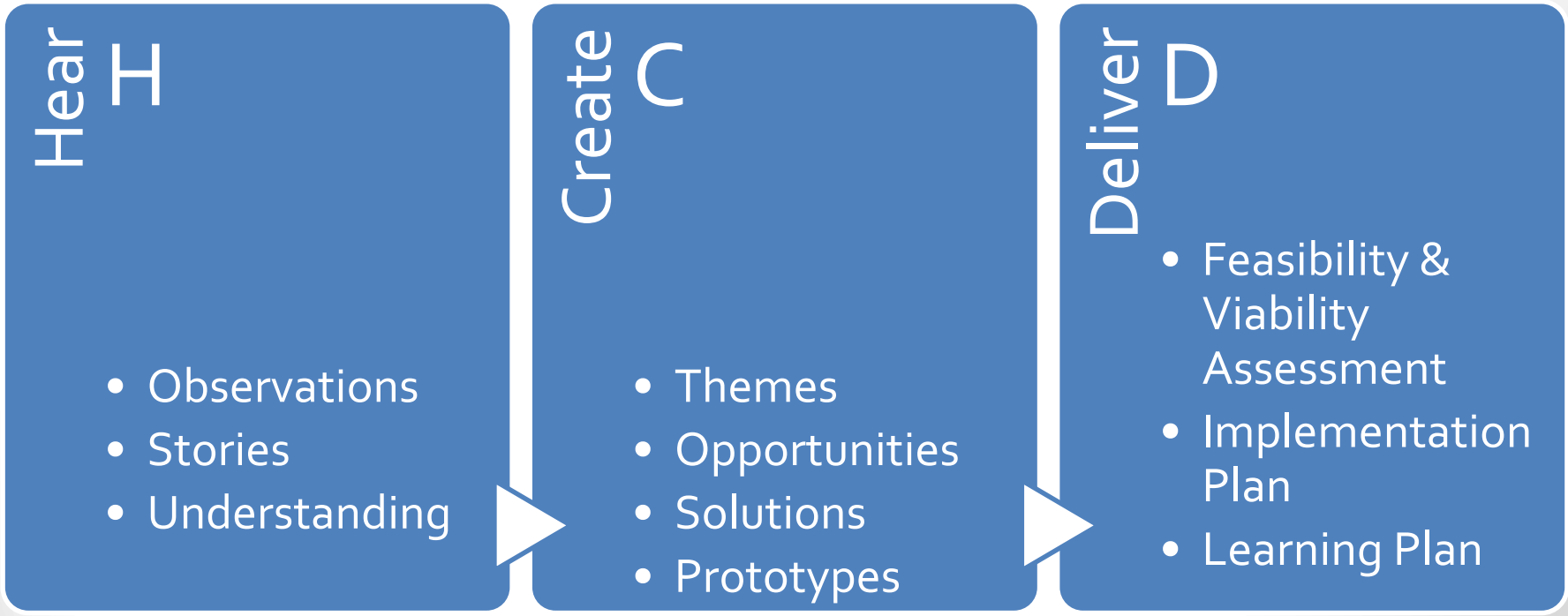
- HCD developed by IDEO
- HCD describes a process and a set of techniques used to create new solutions (*Ideation*)
- Called "human-centered" because it focuses the people solutions are designed for
- HCD process starts examining the needs, dreams and behaviors of the people trying to listen to and understand what they want

The Three Lenses of HCD



Aim: Create solutions which are
Desirable, Feasible and Viable.

HCD Process


Hear
H
• Observations
• Stories
• Understanding
Create
C
• Themes
• Opportunities
• Solutions
• Prototypes
Deliver
D
• Feasibility & Viability Assessment
• Implementation Plan
• Learning Plan

Hear

- Goals
  - Who to talk to
  - How to gain Empathy
  - How to capture Stories
- Outputs
  - People's Stories
  - Observations of Reality
  - Deeper Understanding of Needs, Barriers & Contraints

Hear

1. **Identify a Design Challenge**
2. **Recognize Existing Knowledge**
3. **Identify people to speak with**
4. **Choose Research Methods**
5. **Develop an Interview Approach**
6. **Develop your Mindset**

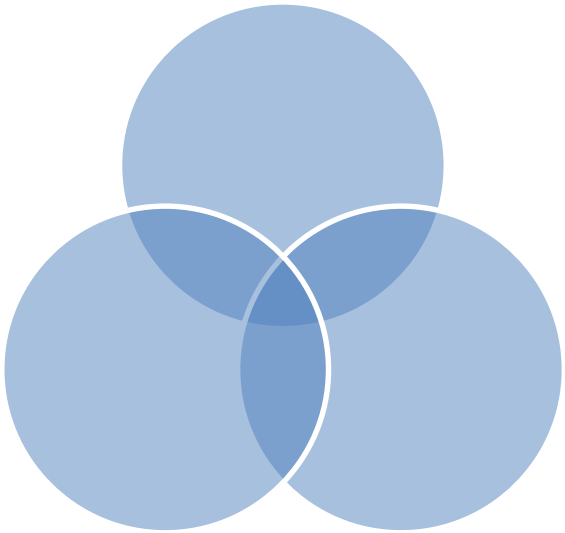
Create

- Goals
  - Making Sense of Data
  - Identifying Patterns
  - Defining Opportunities
- Outputs
  - Opportunities
  - Solutions
  - Prototypes

Create

1. **Develop the Approach**
2. **Share Stories**
3. **Identify Patterns**
4. **Create Opportunity Areas**
5. **Brainstorm New Solutions**
6. **Make Ideas Real: Prototyping**
7. **Gather Feedback**

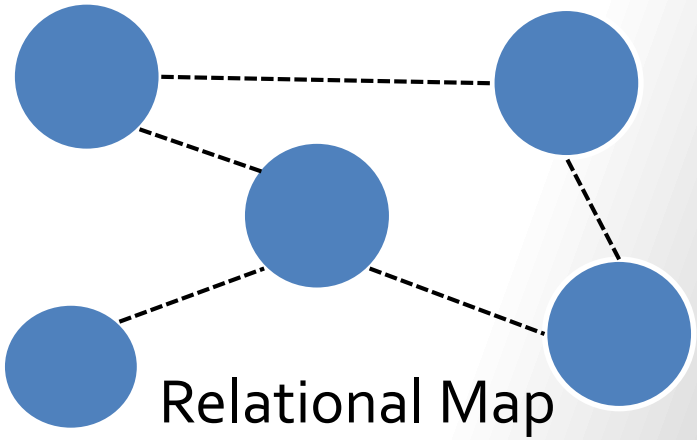
Frameworks



Venn Diagram

Two-by-Two Matrix

Process Map



Relational Map

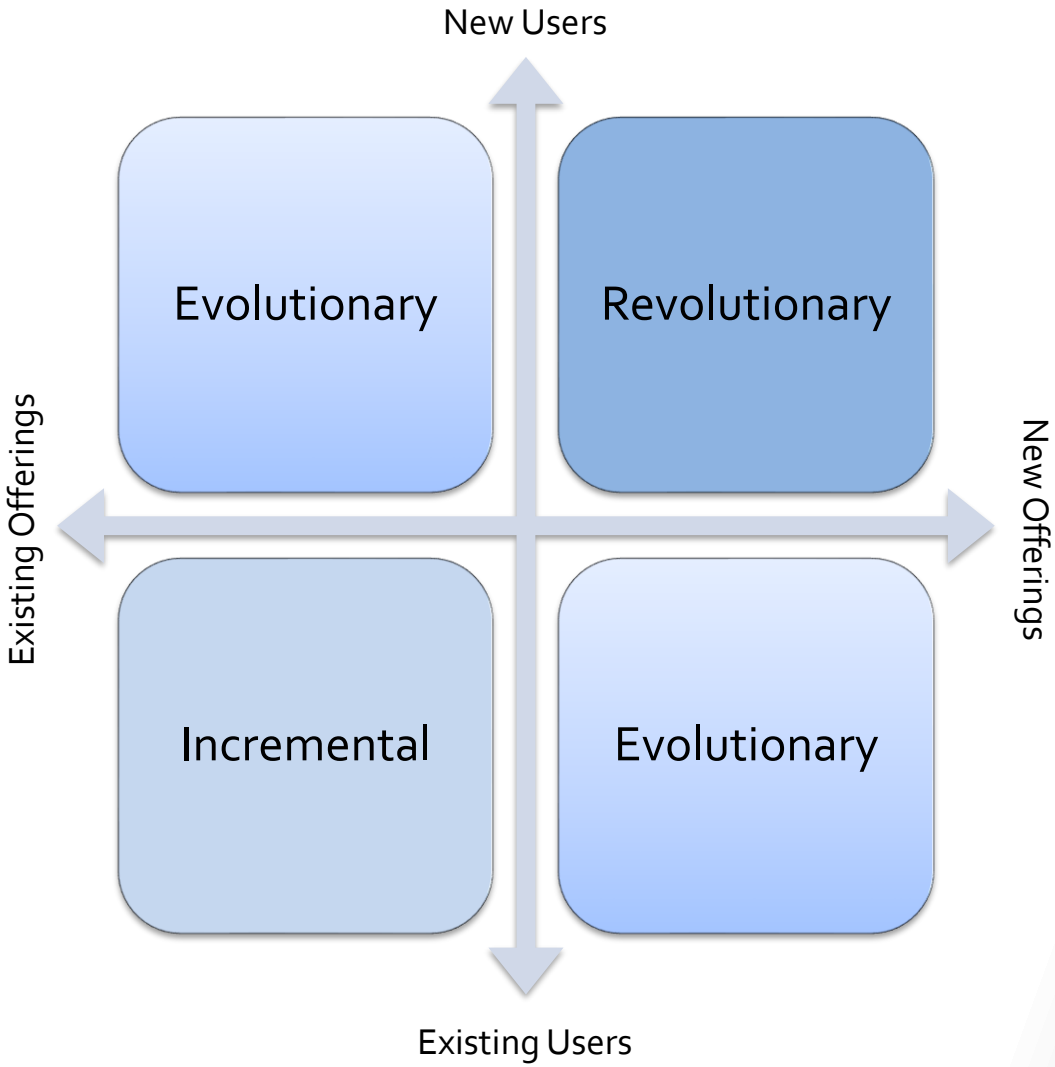
Deliver

- Goals
  - Identify required Capabilities
  - Create a model for Financial Sustainability
  - Develop an Innovation Pipeline
  - Plan Pilots & Measure Impact
- Outputs
  - Feasibility & Viability Assessment
  - Innovation Pipeline
  - Implementation Plan
  - Learning Plan

Deliver

1. **Develop a Sustainable Revenue Model**
2. **Identify Capabilities Required for Delivering Solutions**
3. **Plan a Pipeline of Solutions**
4. **Create an Implementation Timeline**
5. **Plan Mini-Pilots & Iteration**
6. **Create a Learning Plan**

Solution Matrix



CHAPTER://11

- Managing Requirements & Change

Requirements meet Change

- Requirements will change
  - Focus on: **Requirements Management**
- Causes for Change
  - Incident – Event deviates from expected behavior
  - Respond to new/changed business requirements
  - Introduce new and updated components and services
  - → A Web-based system must be treated like a garden
- Key concepts to handle
  - **Incident Management**
  - **Baselining**, "Freeze late", etc.

Managing Change

- **Project's Change Management**
  - **No requirement, feature, function, component etc. added or changed without approval**
  - Responsible Change Control Board (CCB)
  - Cf. Standard IEEE 828
  - Essential to hinder creeping user requirement, gold plating
  - **Should be applied from the early beginning of the project for all assets!** (*not exclusively related to RE*)
- **Change Request (CR)** – A description of a potential improvement to the Web Application, often identified by the users

Simple Change Control

CR due to Issue etc.

Submitted

Impact & Readiness analysis

Evaluated

CCB rejects

Rejected

CCB accepts

Approved

Verification failed

Changing

Change Made

Cancelled

No verification required, deployed

Verifying change

Change failed

Closed

Verified

Change ok, deployed

Change Management

- Remember:
  - Change Management is not a phase of Requirements Engineering
  - Change Management is the starting point for your project and continues from then on

CHAPTER://12

- Further Readings

Literature

- Institute of Electrical and Electronics Engineers. IEEE Standard Glossary of Software Engineering Terminology, IEEE Standard 610.12-1990, IEEE, New York, 1983.
- S. Ward and P. Kroll. Building Web Solutions with the Rational Unified Process: Unifying the Creative Design Process and the Software Engineering Process, Rational Software Whitepaper,1999.
- R. Oberg, L. Probasco, M. Ericsson. Applying Requirements Management with Use Cases, Rational Software Whitepaper,2000.
- Chapter 5: Thomas A. Powell, Web Site Engineering, Prentice Hall PTR
- Chapter 2-6: S. W. Ambler, Process Patterns – Building Large-Scale Systems Using Object Technology, Cambridge University Press

Literature

- Karl E. Wiegers, Software Requirements, Microsoft Press, 2003
- Suzanne and James Robertson, Mastering the Requirements Process, Addison-Wesley, 1999
- Chapter 8.2: David Lowe and Wendy Hall, Hypermedia and the Web – an Engineering Approach, John Wiley & Sons
- Chapter 3: Ian Sommerville, Software Engineering, Addison-Wesley

==========

Further information available at Lecture Web Site

==========

CHAPTER://13

- Conceptual Design

Remember: Vision eConcierge

- **FOR** guests of the hotel
- **WHO** need assistance in enhancing their stay by choosing restaurants or cultural events
- **THE** eConcierge Service (eCS)
- **IS** a portal
- **THAT** will provide an overview of events, activities and selected restaurants (partners)
- **UNLIKE** the current black-board approach
- **OUR PRODUCT** will provide ubiquitous assistance from the early beginning of the ordering process as well as during the stay by allowing to access the system with different devices

Based on “Software Requirements” 2nd Edition by K.E. Wiegers

eConcierge Examples (Hilton Vienna and aloft, Brussels)

Purpose

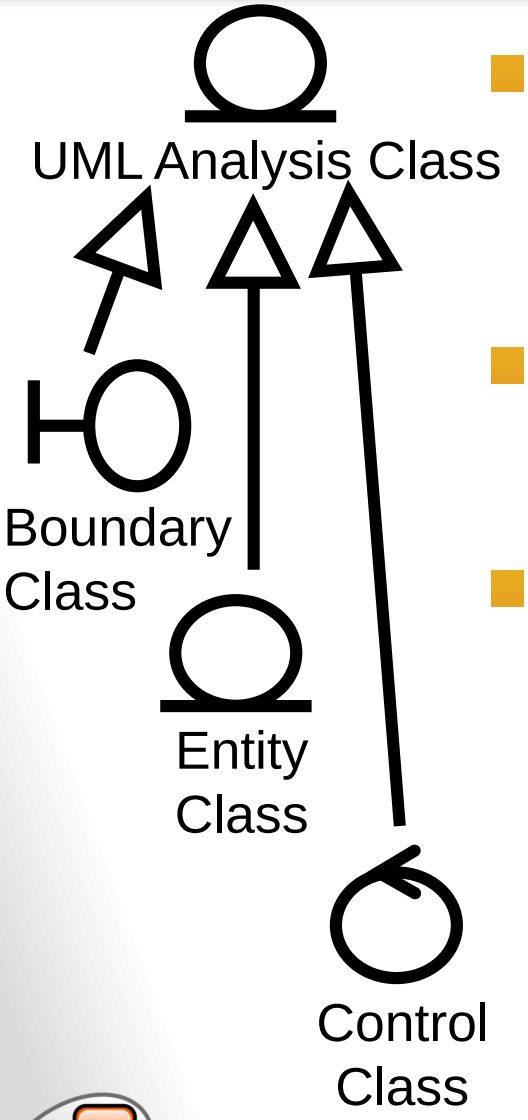
- Analysis of customer problem description
  - Describing the problem in **developer language**
  - Creating a **Conceptual Design**
- When to employ Conceptual Design?
  - Need for concurrent "Design & Implement" phases
  - Provides an excellent overview of the product
  - Easier to study than Design (~ 1:5)
  - Product uses Legacy Systems

* Analysis and Conceptual Design are used as similar terms/concepts

Conceptual Design Activities

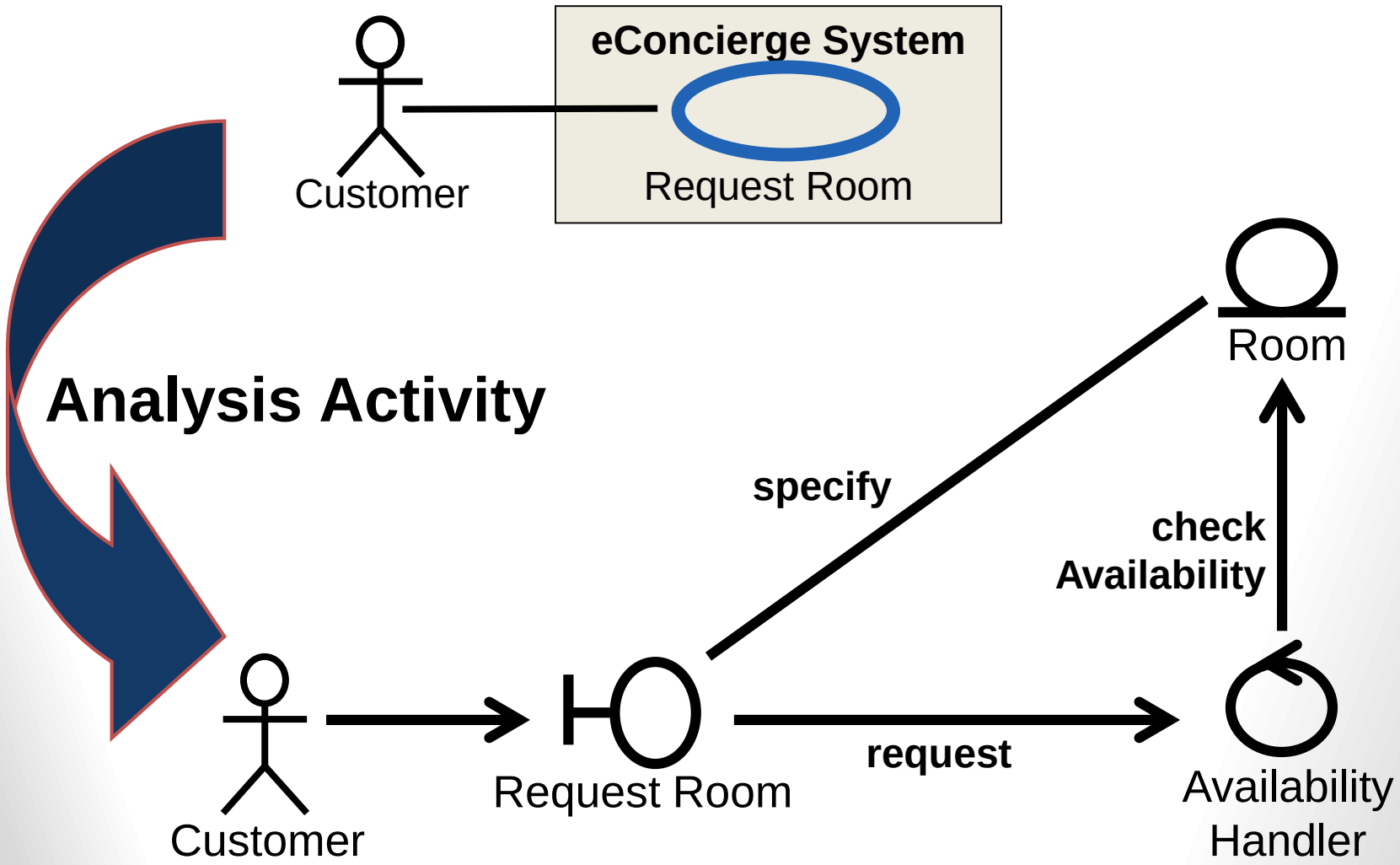
- Activities to perform:
  - **Transform requirements** into conceptual model model
  - Structure model elements wrt **product dimensions**
  - Define **validation tests**

Conceptual Model



- **«boundary»**
  Models Interaction between Product and Actors (e.g. User Interface or Communication Interface)
- **«entity»**
  Models information and associated behavior (e.g. real-life object, event)
- **«control»**
  Models coordination, sequencing, transactions, and control of other objects (e.g. Business Logic)
  - Do not encapsulate interactions with actors!

Transforming Example



Product Dimensions

- Product Dimensions
  - Separation of concerns of product
  - Each dimension subject for conceptual design
- Aspects of **Content**
  - Data - Entities and Relationships the product deals with
  - Semantic annotations/extensions etc. of Data that might be useful for other applications
- Aspects of **User Interaction and User Experience**
  - Presentation
  - Navigation
  - Dialogue
- Aspects of **Distributed Web Systems**
  - Processes
  - Communication
  - Web System Aspects

SECTION://1

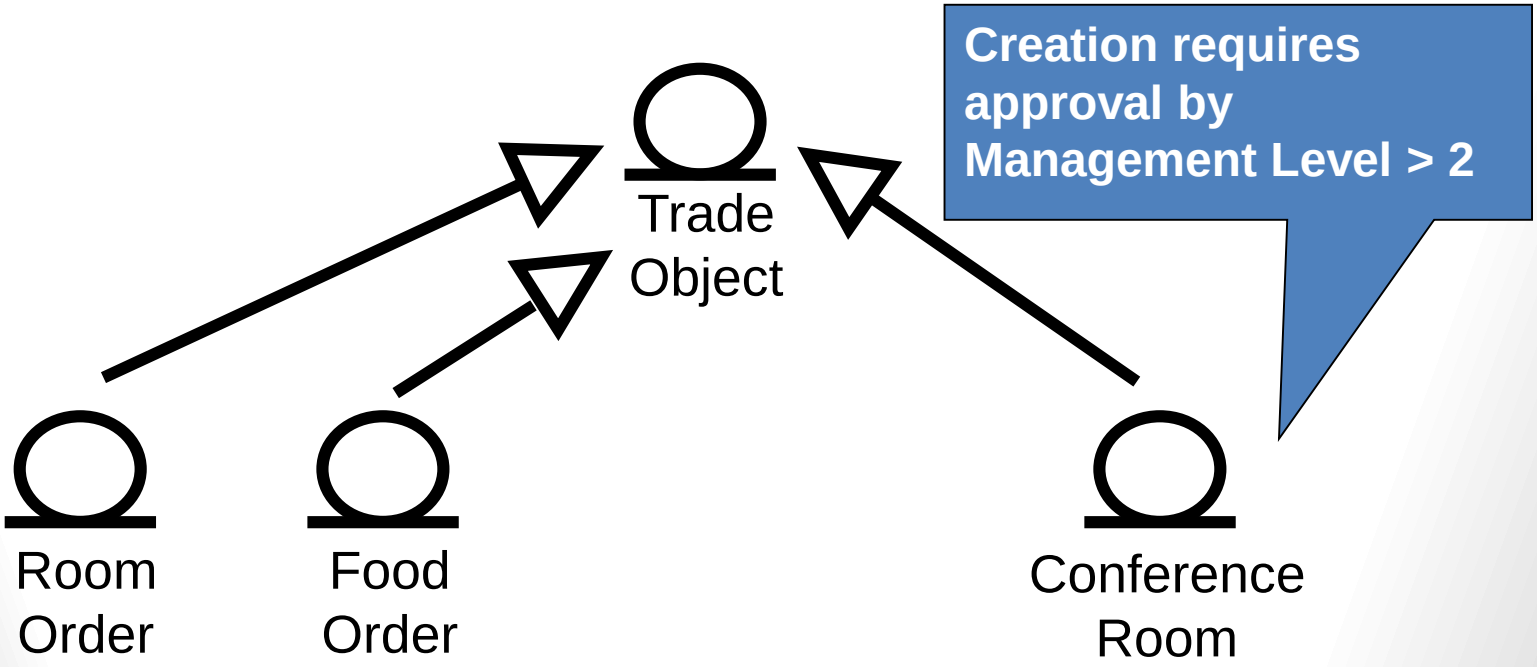
- Content

Data Design

- Define **Entity Classes**
  - Complete, semi-structured, etc.
  - Capture special requirements, e.g. persistency
- Identify relationships
  - Abstractions: refinements / generalization
  - *Roles, responsibilities, security concerns*
  - *Access rules: Creation and management*

Data Example

- Extract shared and common behavior
- Objective: Make solution easier to understand on a conceptual level



Data - Information and Media

- Some Observations and Aspects
  - static vs. dynamic
  - persistent vs. transient
  - passive vs. interactive
  - pull vs. push

- These aspects must be considered in design.

Static vs. Dynamic

- Static
  - Access at different times returns the same result
- Dynamic
  - Access at different times may return different results
- Dynamic Information in the Web
  - Irregular changes
    - Extensions, updates, etc.
  - Regular Changes
    - Periodical issues (e.g. every month, week, day, hour)
    - Periodically generated information (e.g. every day, hour, min, sec)
  - Generated on demand
    - For each access
    - At certain events

Persistent vs. Transient

- Persistent
  - Data can be accessed time independent and repetitive (content is not changing)
    - E.g. books in the library, CD-ROM, DVD, …
- Transient
  - "passing with time"
    - E.g. TV and Radio
- Transient Media on the Web
  - Web is only the transport media, e.g.
    - WebCam – snapshots sent over the web
    - Live media streams like Real Audio, Real Video
    - Results of a calculation (dynamic applications)

Passive vs. Interactive

- Passive
  - Unidirectional information from provider to consumer
  - No (very little) decision for the consumer
- Interactive
  - Information exchange in both direction from provider to consumer and vise versa
  - Symmetric, e.g. video-conferencing
  - Asymmetric, e.g. options to choose from, Wiki principle
- Interactive media on the Web
  - Hyperlinks
  - Forms
  - GUIs

Pull vs. Push

- Pull (Basic Web Mechanism)
  - Services by request
  - Communication is initiated by the user
- Push
  - Service by distribution
  - Communication is initiated by the provider
- Push-Media in the Web
  - Subscribing to information
  - Information distribution, e.g. ActiveChannels (IE4-6)
  - Simulated – automated browser refresh
  - Server push
  - Streaming server (like radio and TV)

SECTION://2

- User-Interface Experience (UIX)

Aspects of Interaction

- Human Computer Interaction (HCI)
  - User is interacting with the computer in order to accomplish something
    - (Dix et al., 1998)
  - … It's not audio. It's not video. It's user control and dynamic experience
    - (T. R. Schussler, Mac World San Francisco conference, 1998)
  - Interactivity is about genuine human engagement.
    - (Nathan Shedroff's, 1994)
- In other words:
  - **If the experience you create is not a compelling one (whether justified by the bounds of the technology or not), you will never find a large audience.**
    - **(Shedroff, 1994)**
  - Teach interaction design like they do in film school
    - (Laurel, 1990)

What is Interaction Design?

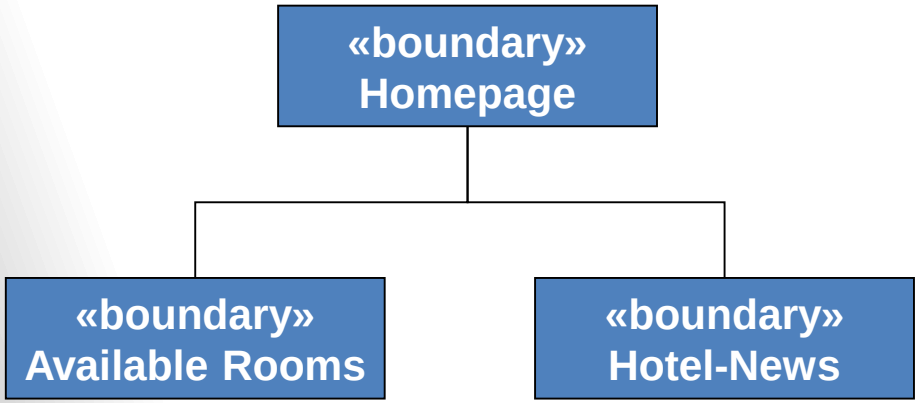
- Design of any direct or indirect communication between a user and computer?
  - **Direct interaction**: dialogue with feedback and control throughout the performance of a task
  - **Indirect interaction**: may involve background or batch processing
- It is about Creating experiences
  - The user should feel good

Model Presentation Aspects

- Presentation – Core component of interaction
  - Explanatory Interactive Assistance
  - User Agents aspects, e.g. screen-size
- Dynamics of content
  - Analyze entity and control classes
  - Identify static elements (e.g. temporary static)
  - Focus on: Access Frequency vs. Change Frequency
- Define layout and design elements
  - Analyze mock-ups and prototypes for reusable layout blocks
  - Define abstract, conceptual elements

Model Navigation

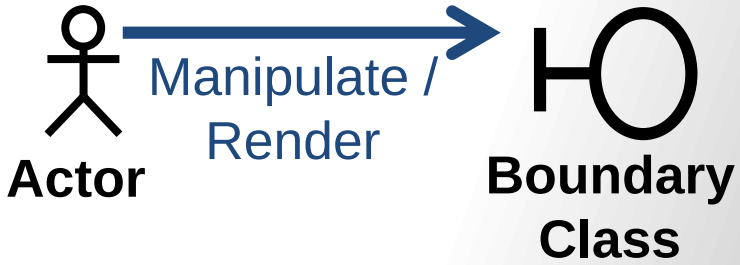
- Analyze Relations
  - Entity to entity relations based on RNA results
  - Set-based relations
- Possible Navigation Patterns to apply



- Types of Relations
  - Direct access to entity
  - Unidirectional / bidirectional
  - Access by type, e.g. to retrieve additional information about an entity
  - Transactional
  - Context-oriented relations
- A lot of research done here, e.g. OOHDM, UWE, OOH, WebML etc.

Model Dialogue

- Define roles for actors interacting with the information space
- Consider special requirements, e.g. browser or bandwidth, as additional dimensions
- Analyze major activities on entities
  - Context, e.g. events, authentication
  - Create, read, update, delete
  - View, Hide
  - Roles



SECTION://3

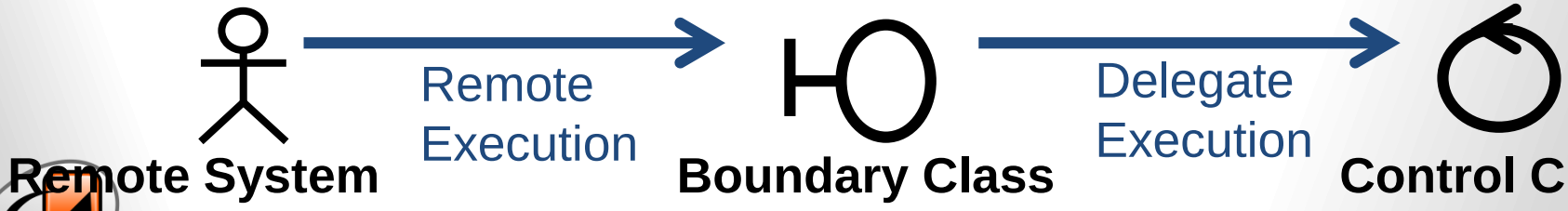
- Distributed Web System

Introduction

- Conceptual Design of overall (distributed) application logic ~ "Business Process"
- **Business Process** – is defined by
  - Two or more autonomous Participants
  - Communicating via Operations
    - Message-based operations
    - Distributed environment
  - No central coordinating Authority
- Each **Participant**
  - Well-defined Behavior
  - Well-defined Relationships with other Participants
  - Concrete implementation is opaque (Black-Box Principle)

Model Business Process

- Design the overall (usually distributed and large scale) business logic of the Web product
- Analyze questions wrt remote execution
- Boundary class
  - Dedicated to system-based actors
  - May represent legacy system access
  - Prepares for wrapper/broker-pattern
  - Focus on Integration Aspects, like EAI, SOA
  - Security and protocol issues



Model Environmental Aspects

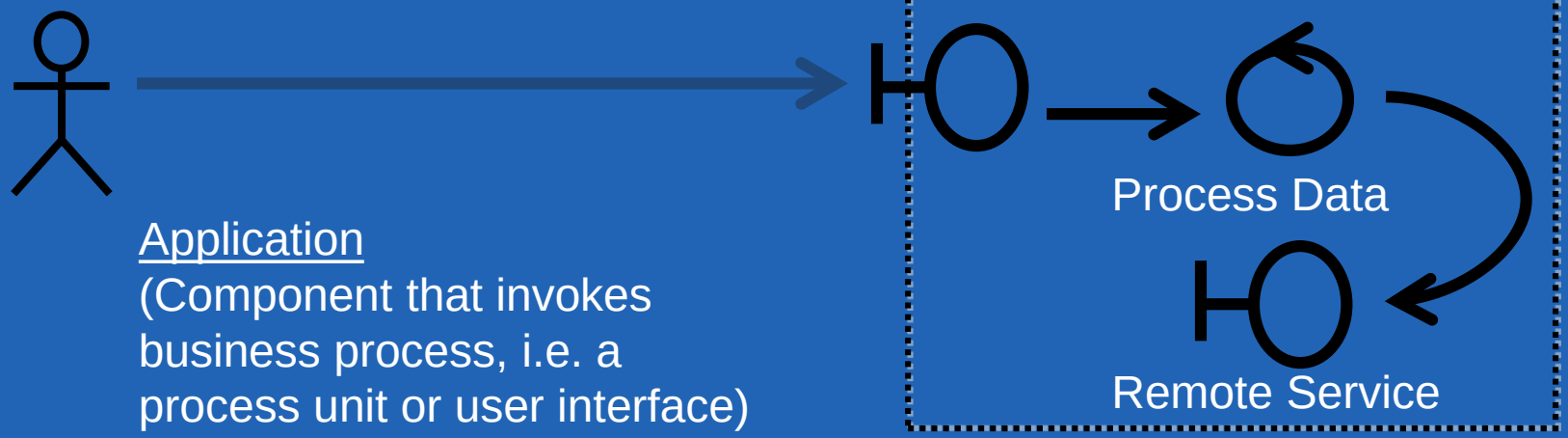
- Architecture is foundation of application
  - **Focus is on its distributed nature**
  - Evolution will happen soon
  - Product might fail due to environment restrictions
  - *Changes are very expensive!*
- Analyze
  - Dependencies between processes – Try to limit
  - Review non-functional requirements – e.g. Bandwidth
  - Optimizing
    - By reducing communication load
    - By increasing communication but reducing processing load
    - Check for caching possibilities
  - Check for security concerns

Idea

- Separation of Concerns
  - Specify Functionality
  - Design reusable process units
  - Package related business process logic (a.k.a. Domain Components)
  - E.g. Use Case related business objects
- Conceptual Design of overall application logic is separated into different views
  - Input for logical design: Process Design, which represents the process layer

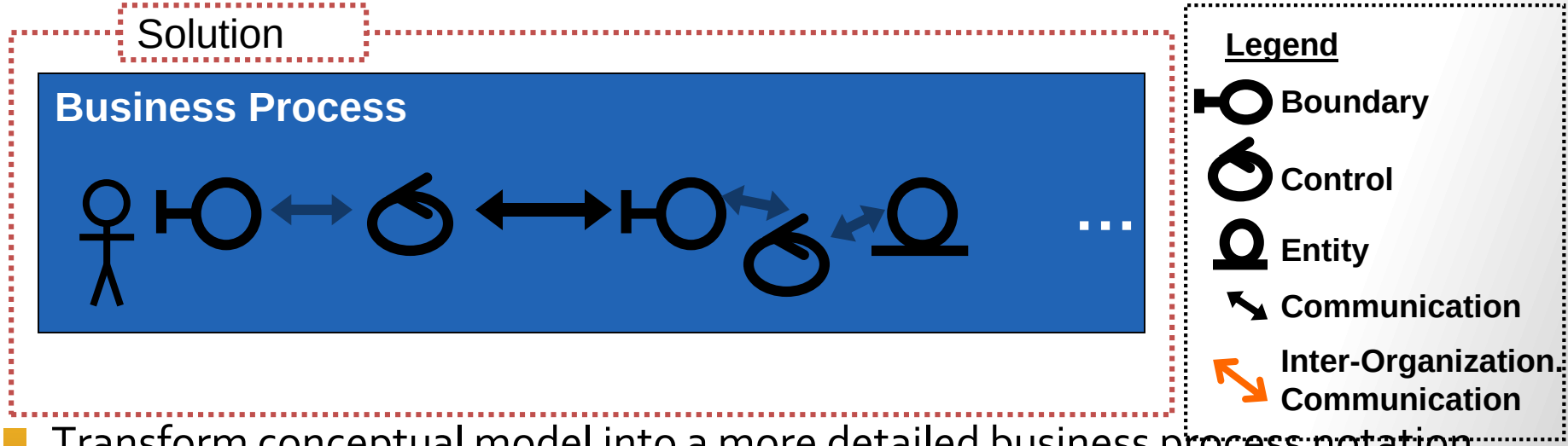
Shift To Process and Service Views

- **Analysis:**



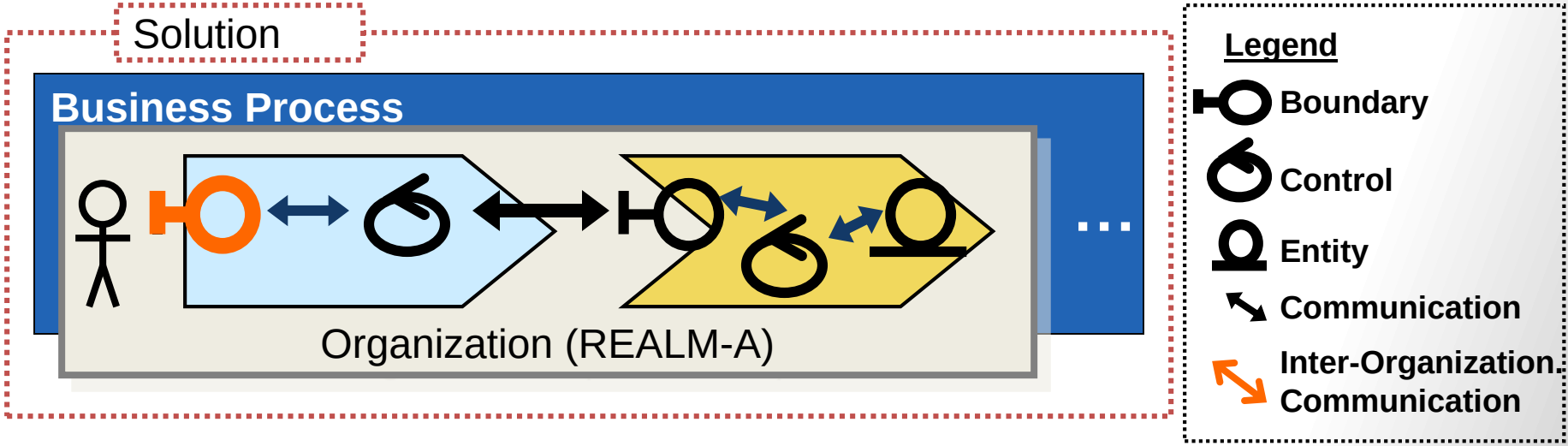
- Many questions to solve…
  - What is the application accessing? And how?
  - What is the interface becoming?
  - What is a processing unit? How is it connected with others?
  - Where is the System and what is it all about?
  - And many, many more…

Process at a Glance



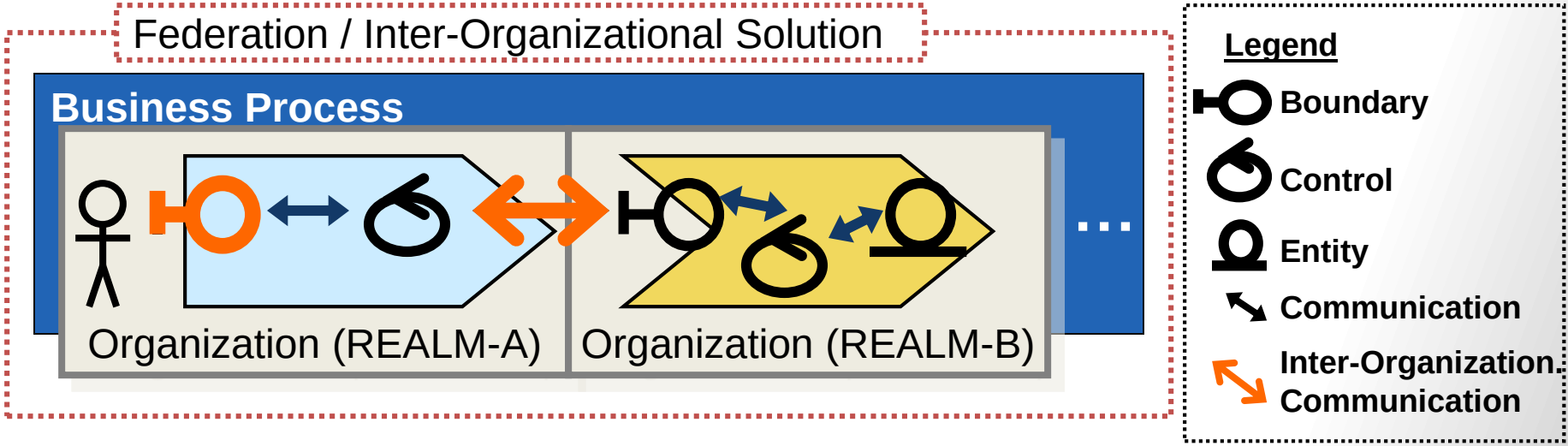
- Transform conceptual model into a more detailed business process notation
  - Helps to map conceptual model to logical models
  - Decompose Business Processes
- Complexity of Business Processes → Need for decomposition

Process Decomposition



- Business Process
  - Business process consists of multiple units
  - Units of a business process are owned by organization
  - Users are managed by organization
- Realm – Represents organizational boundaries
  - Zones of control over the owned systems, i.e. to support IAM, roles, firewall perimeter etc.

Security by Design – Considerations

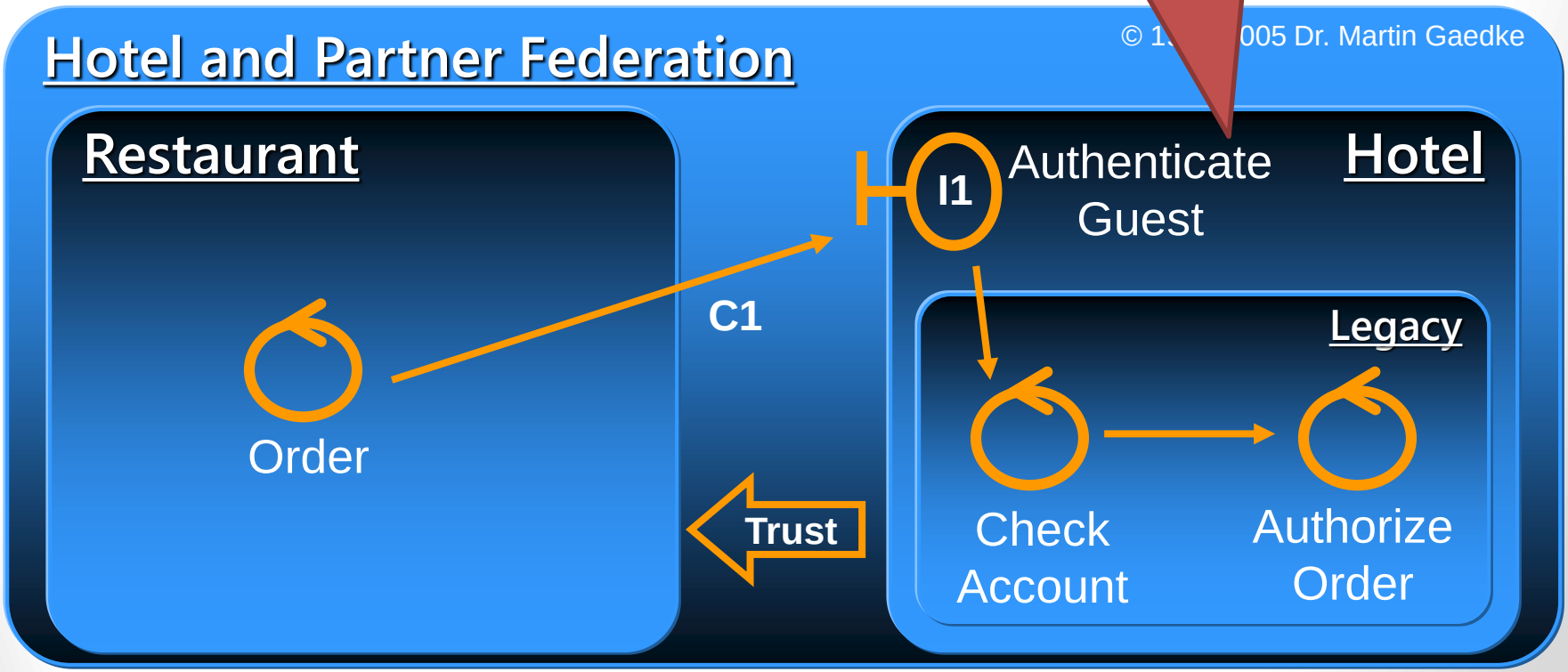


- Federated Business Processes
  - Business process is inter-organizational
  - Different Units of one business process are owned by different organizations
  - Requires dedicated support for identity and access management (User of realm-a acts in realm-b due to a trust-relationship between both organizations)

Example

Each realm might act as a Hosted / Cloud solution

- Example: Order – Discuss: Identity issues



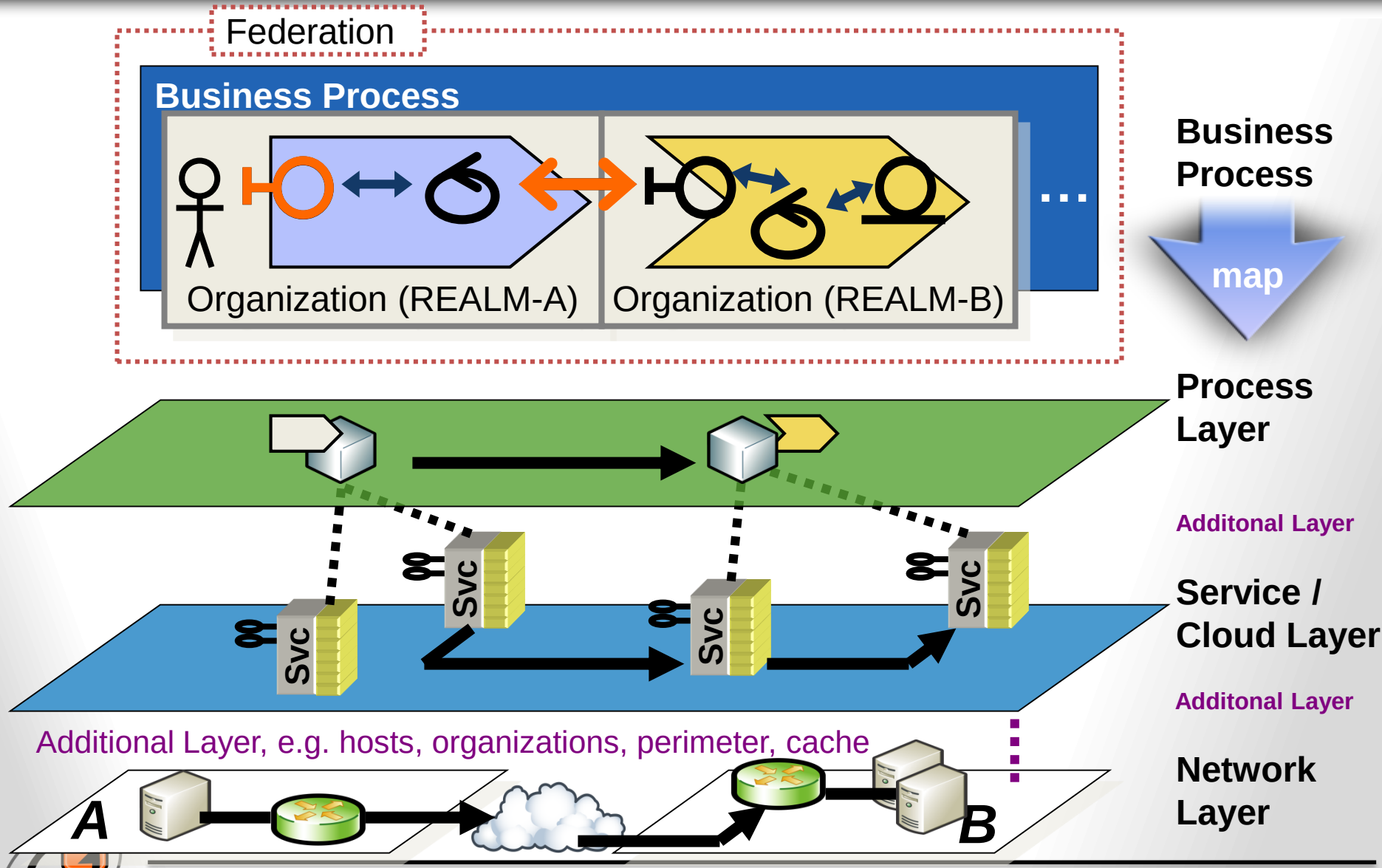
C1:
- Secured Connection
- Speed: 5-54 MBit/s

I1:
- CRUD-Interface
- Policy: Trusted Clients only

Realm

Zone Context
- Security Zone
- Trust Zone, etc.

Multi-Layer View on Business Process



Business Process Notation

- Allows to visualize communication between participants (possibly support for abstractions - business process units)
- Some approaches
  - Business Process Modeling Notation (BPMN)
  - UML Activity Diagrams
  - UML EDOC Business Processes
  - ebXML BPSS
  - Event-Process Chains (EPC)
  - State-Charts
  - Petri-Nets
  - etc.

Summary

- Business Process
  - Spawns conceptual design of the distributed system
- Distributed System is represented by different views (layers)
  - Common approach: Process Layer, Service Layer, and Network Layer
  - Allows for layer of interest, e.g. Federation Layer for inter-organizational processes, hosting layer
  - Apply dedicated approaches (logical design and physical design) for each of these layers

Literature

- S. Ward and P. Kroll. Building Web Solutions with the Rational Unified Process: Unifying the Creative Design Process and the Software Engineering Process, Rational Software Whitepaper,1999.
- Chapter 3,4,8: I. Jacobson, G. Booch, J. Rumbaugh, The Unified Software Development Process, Addison-Wesley, 1999.
- S. Seely and K. Sharkey. SOAP: Cross Platform Web Services Development Using XML, Prentice Hall PTR, 2001.
- W. L. Oellermann Jr. Architecting Web Services, aPress, 2001.

==========

Further information available at Lecture Web Site

==========

CHAPTER://14

- Design – Describing the Solution

Introduction

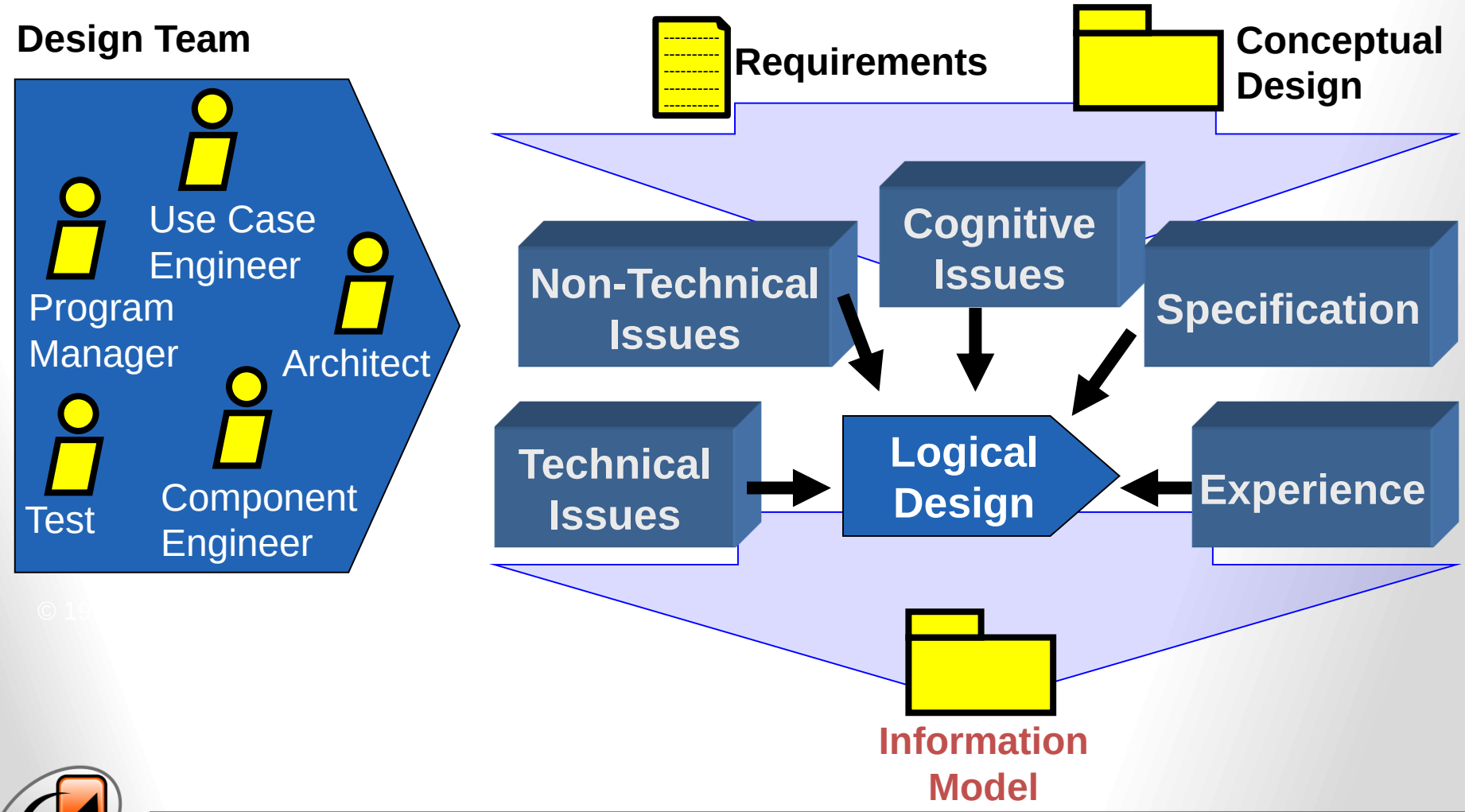
- Shape the product and find its form – Regarding all requirements and conceptual design
  - E.g. data, presentation and hypermedia aspects
  - E.g. architecture and its distributed aspects
  - E.g. emotional aspects
- Achieve Design Model of Web Application
  - Maintainable and detailed description
  - Easy to understand by common notation

- We focus on maintainable and reusable design aspects

Aspects of Design

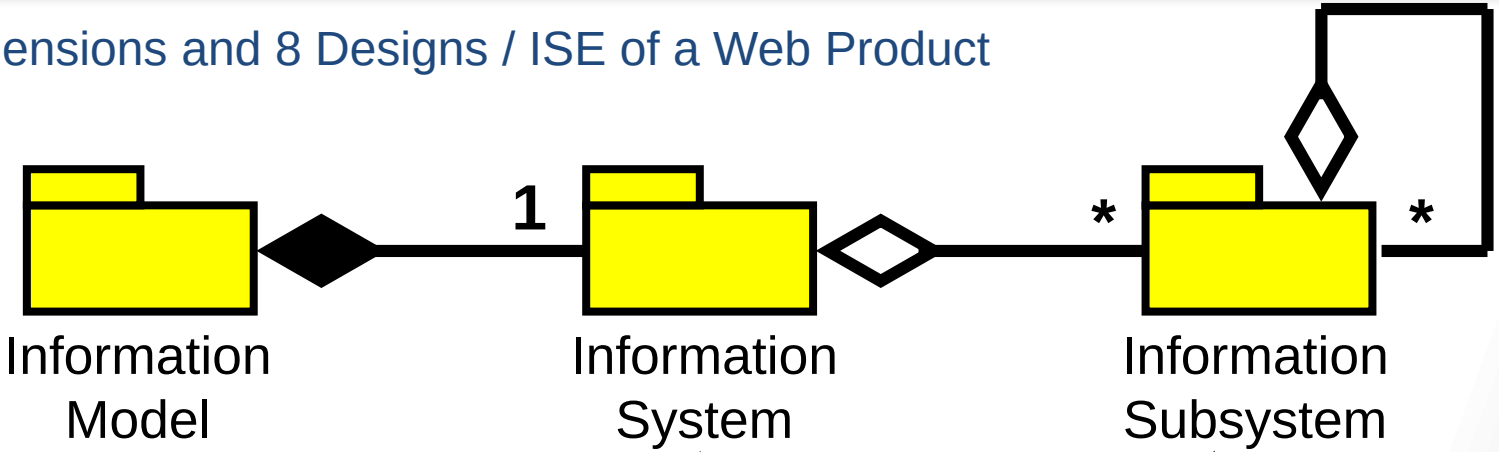
- Design Aspects: Logical and Physical
- Separate logical units of the product
  - ▶ Complete solution independent of physical matter
- Goal: Allow seamless mapping to physical design and implementation
- Possibly implement parts of design in parallel

Design Team and Artifacts

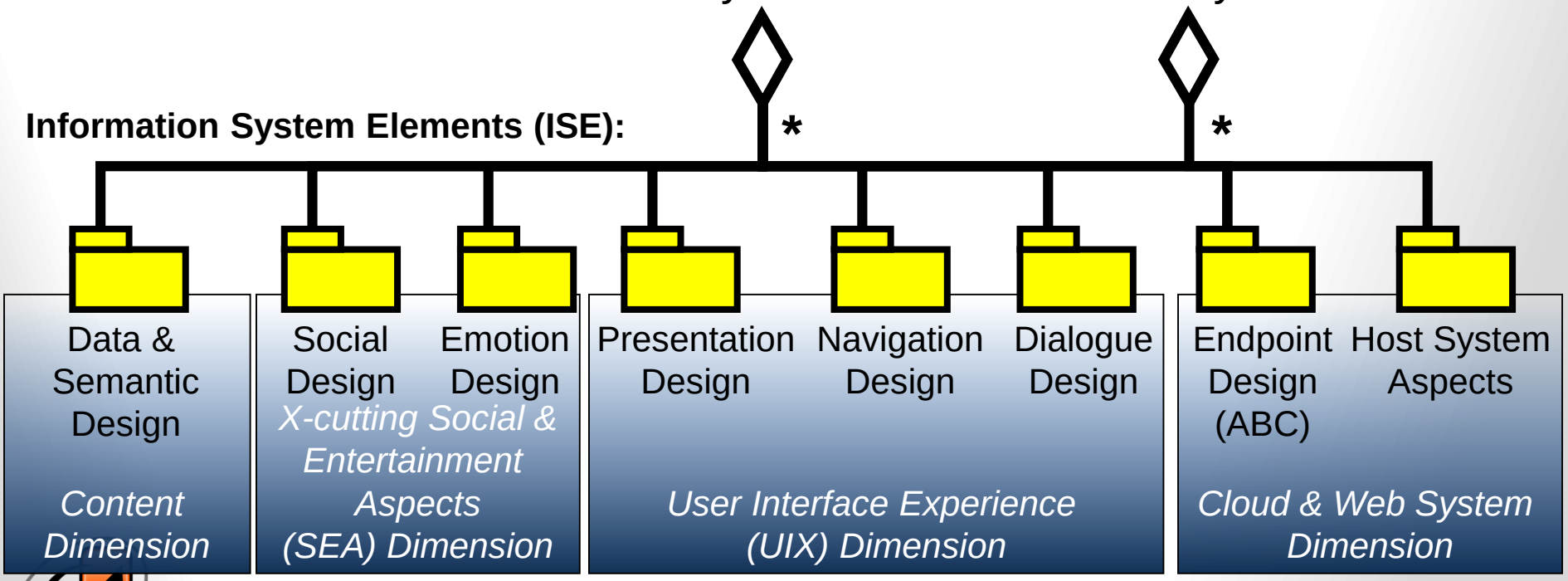


Information Model

4 Dimensions and 8 Designs / ISE of a Web Product



Information System Elements (ISE):

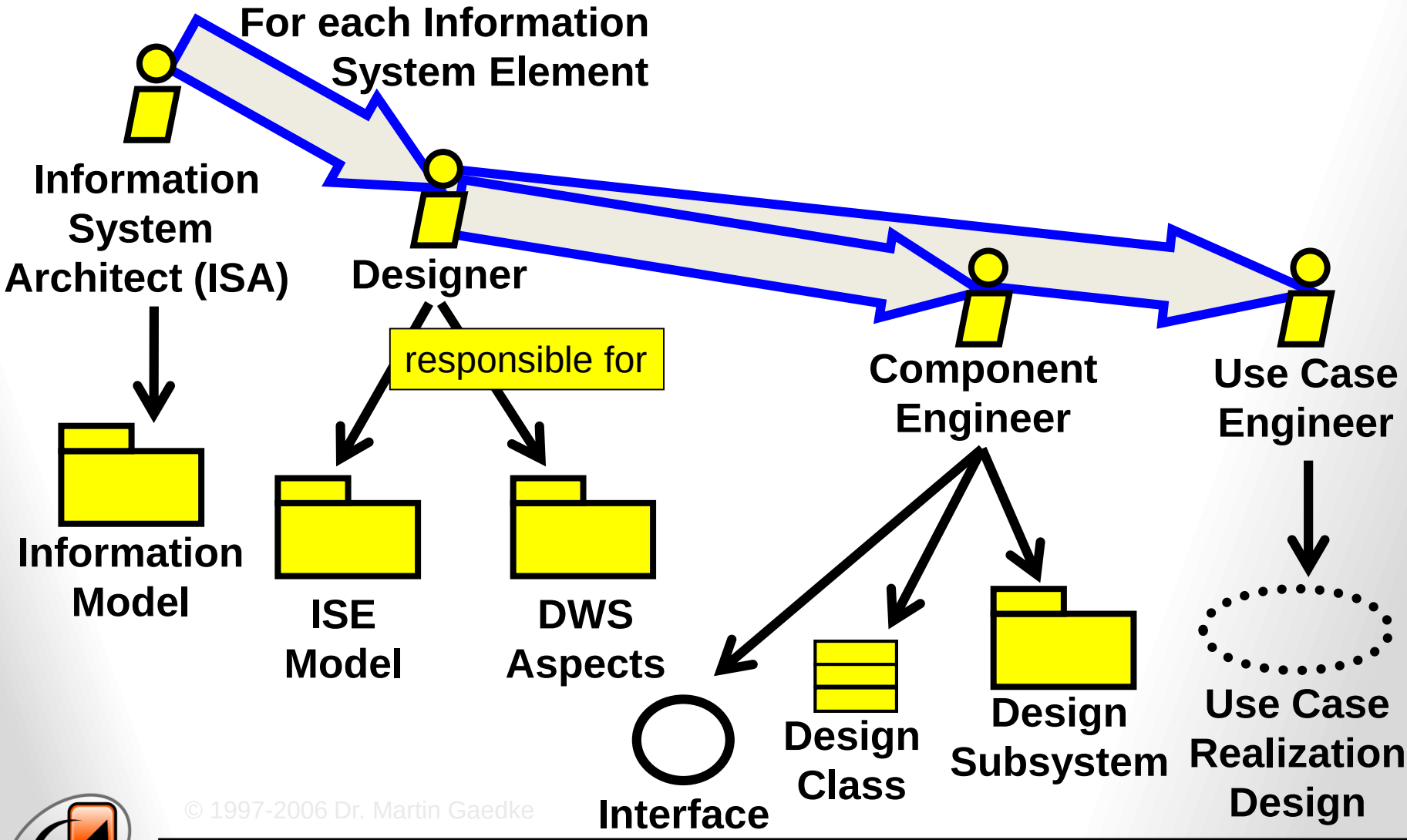


Information Model - II

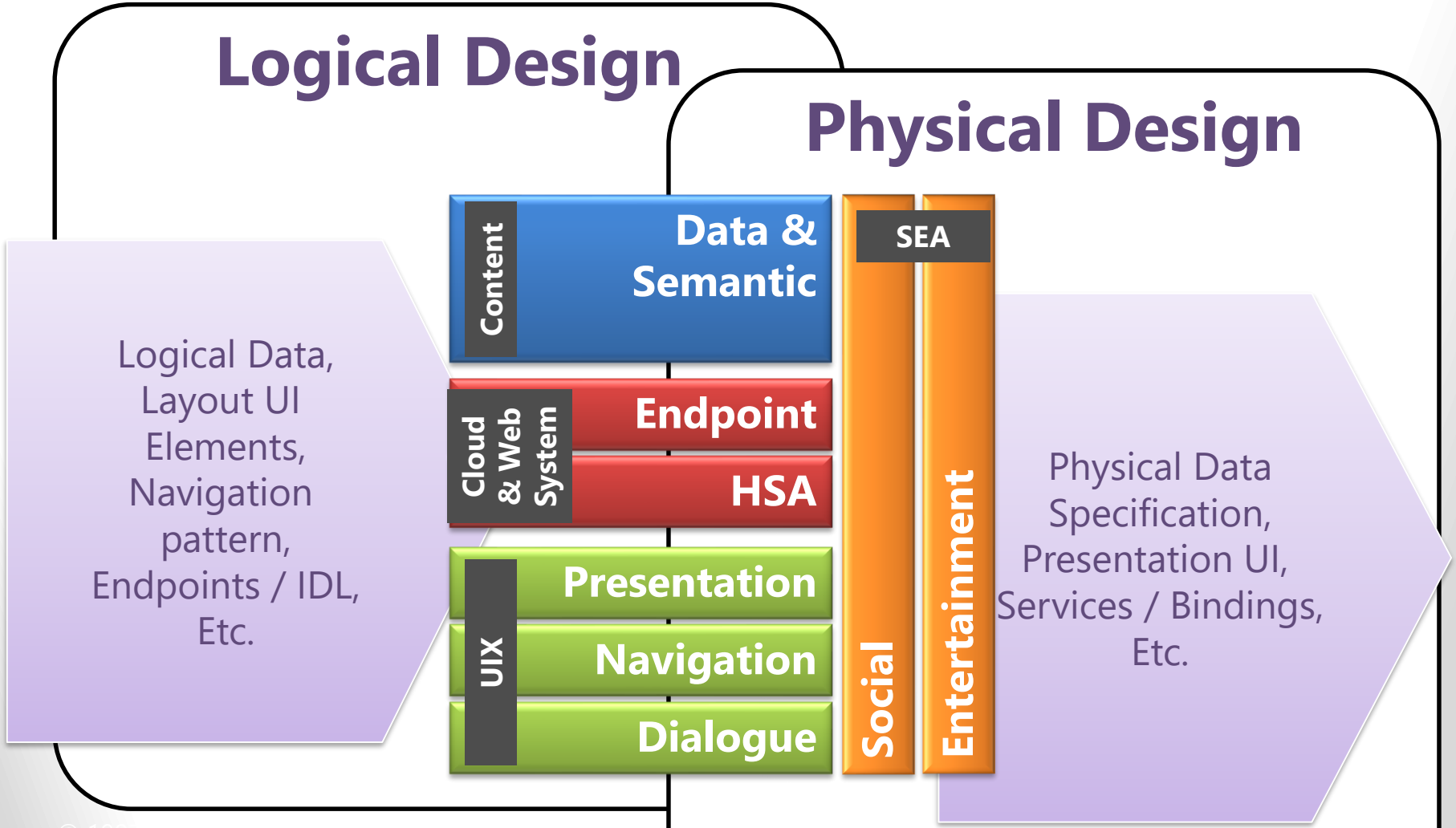
- Activities regarding Information Model Design
  - ▶ Depending on major Product Dimensions
    - Content and UIX
    - Select ISE to focus on
  - ▶ Example for Static Site:
    - Content/Data,
    - Navigation,
    - Presentation Design

- Each information system element is closely related to each other
- There is no single thread / process model through the six designs

Design Team Responsibilities



From Logical to Physical Design



The Social Web and other emotional aspects

- The Social Web
  - Social Web is a set of relationships that link together people over the Web
- The Social Web's potential
  - Social Web term also indicates the **potential**, which could be established by applications on top of these relationships, i.e. addressing all kinds of user-generated content and its use
  - A major **requirement** is a better support (may be also better understanding) of identity
  - Revolutionary **opportunity**: Creation of a decentralized and federated Social Web (**), motivating end users in their social context
  - End user **problems**: portability of their data, identity managemenrt, linkability of social networking sites, privacy
- For further information check W3C's groups on related topics, e.g.
  - ** http://www.w3.org/2005/Incubator/socialweb/XGR-socialweb/
  - http://www.w3.org/2005/Incubator/webid/

CHAPTER://15

- Content

Goal

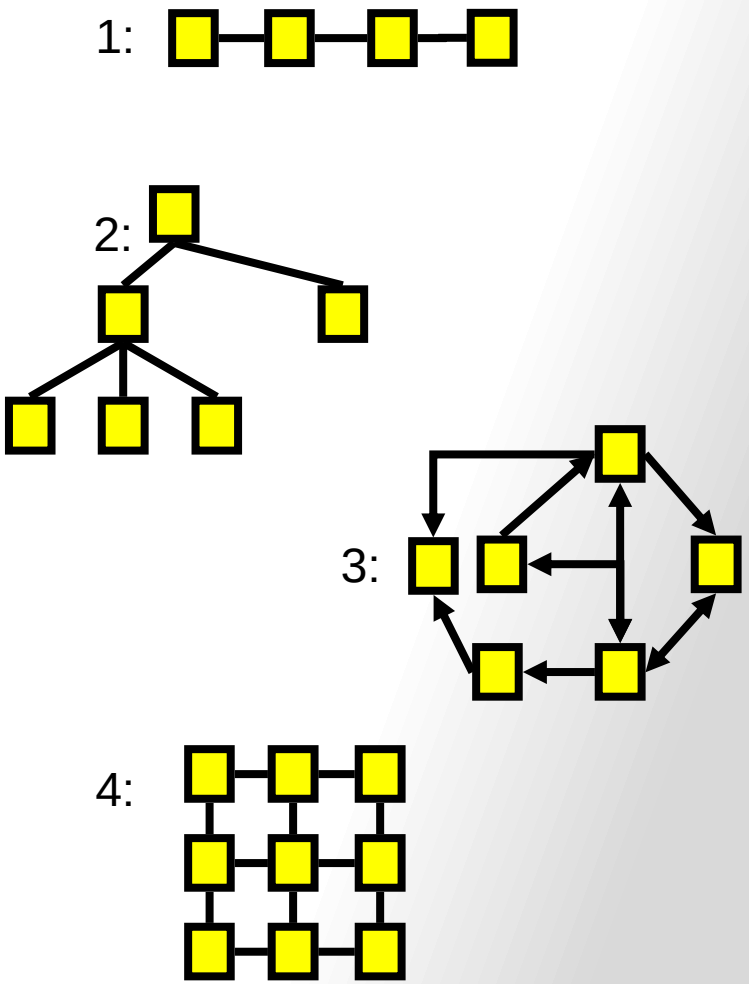
- Aspects of Content Design
- Designing the "**Information Space**"
  - Define structure and structural links (references) of entities (data)
  - Understand impact on distribution, i.e. (physical memory) location of entities
  - Understand and provide semantics, i.e. linking of data
    - E.g. Semantic Web technologies and LinkedData approaches
- Challenges
  - Develop maintainable structure
  - Focus on evolution (growth and changes)
  - Large amount of (multimedia) data

SECTION://1

- Logical Design

Structure and Structural Links

- Linear Structure (1)
  - (ordered) Collections / Sets
  - Pre-caching possible
- Hierarchical Structure (2)
  - Deep or flat structured
  - Requires navigation aid
- Network Structure (3)
  - Extremely expressive
  - Complex Entities
- Grid Structure (4)
  - Collections of related Items
  - Requires uniform data



Logical Design Approaches

- Object-oriented Design approaches
- Hierarchical approaches
- *Web-compliant Data Definitions*
- *Diagramming Technique*
- Related approaches
  - Entity Relationship Modeling (ER-Design)
  - Relationship Management Methodology (RMM)
  - Object Role Modeling (ORM)
  - Object Z
  - Object Constraint Language (OCL)

SECTION://2

- Physical Design

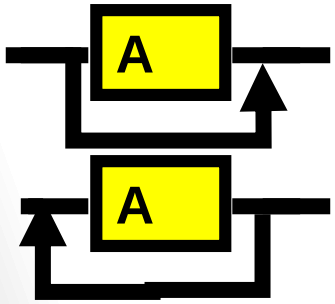
Physical Design

- Transform logical design to physical model
  - Examples:
    - Class source code
    - Database specific table descriptions
- Dedicated approaches for representing data / information in the Web
  - **XML DTD**
  - **XML Schemas**
  - RelaxNG
  - Schematron
  - Resource Description Framework (RDF)
  - XML Information Set
  - Microformats (apply with care – if at all)
  - And many more…

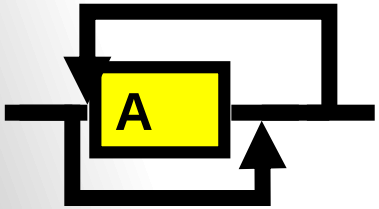
Diagramming Technique for XML

- Component called A

- Component that can not be decomposed called A



- A is optional or occurs once
- A is repeated 1 to n times

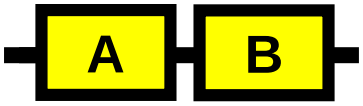


- A occurs 0 to n times

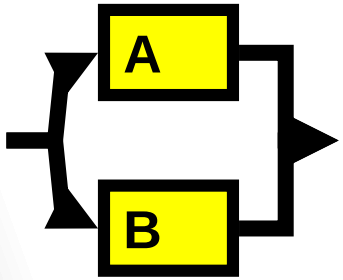
Description	DTD-Notation
Component called A	(A)
Component that can not be decomposed called A	(#PCDATA)
A is optional or occurs once	A?
A is repeated 1 to n times	A+
A occurs 0 to n times	A*

Diagramming Technique for XML

Description

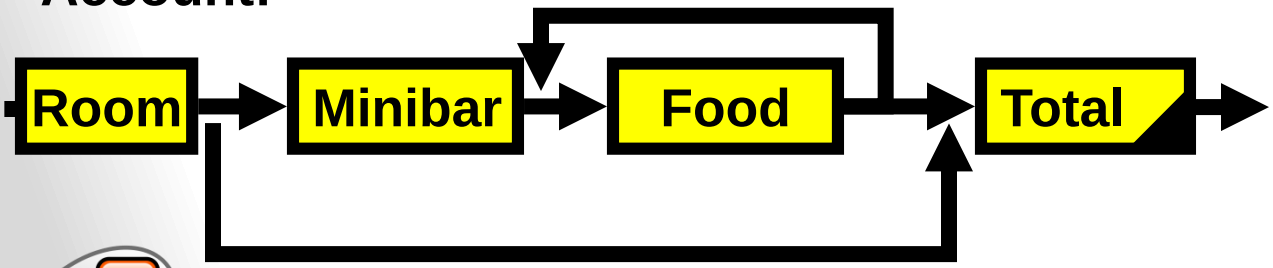


- Concatenation/Series
  B after A



- Selection A or B
- *Example*

Account:



DTD-Notation

- **(A,B)**
- **(A|B)**
- ***Example***

Account= (Room, (Minibar,Food+)?, Total)

Total= (#PCDATA)

Declaration, Definition, Data

Check the Lecture XML Validator Demo

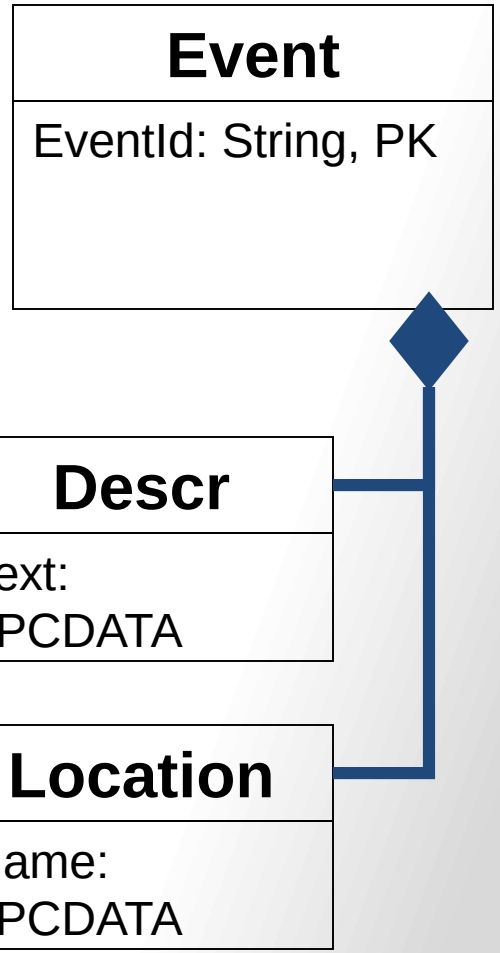
```
<?xml version="1.0"?>
<!DOCTYPE account [
 <!ELEMENT account (room,(minibar,food+)?,total)>
 <!ATTLIST account AccountID ID #REQUIRED>
 <!ELEMENT room EMPTY>
 <!ATTLIST room number NMTOKEN #REQUIRED>
<!ELEMENT minibar EMPTY>
 <!ELEMENT food EMPTY>
 <!ATTLIST food price CDATA #REQUIRED>
<!ELEMENT total (#PCDATA)>
]>
```

```
<account AccountID="a3499bxdz">
    <room number="R101"/>
    <minibar/>
    <food price="10.00"/><food price="15.00"/>
     <total>28.00</total>
</account>
```

Example: Event Entity

- Pay attention:
  - Database design concept
  - But also regarding reuse and change concepts
  - Common formats might exist already

```
<event number="e10">
  <descr>Painter Contest 2008</descr>
  <location>Paris, France</location >
</event>
<event number="m11">
  <descr>Musical Cats</descr>
  <location>Hamburg, Germany</location>
</event>
```



Rethinking: The Event Entity

- Physical Design using XML
  - → Adding semantic support
  - E.g. enhance using Resource Description Framework (RDF)

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns="http://hotel/rdf/syntax#">
    <room rdf:about="http://hotel/event/e10">
        <descr rdf:resource="http://hotel/locations/paris"/>
    </room>
...
```

SECTION://3

- Content Design & SEA

Rethinking Content and Social Web

- Standards for Social Web
- Besides generic standards, e.g. URI/IRI, HTTP, RDF
- Specific Standards
  - Identity related content aspects
  - Social networking related content aspects
  - Content aspects regarding privacy concerns
  - And many more

- For further information check W3C's groups on related topics, e.g.
  - http://www.w3.org/2005/Incubator/socialweb/XGR-socialweb/
  - http://www.w3.org/2005/Incubator/webid/

Identity related content aspects

- Specific standards for profile and relationship information, e.g.
  - XRD (Extensible Resource Description) for identity provider data
  - VCard, IETF for personal address-book data
  - FOAF (friend of a friend ontology), e.g. applicable in RDFa, Linked Data, etc. – works together with other ontologies
  - PortableContacts, "enhanced VCard„ – XML, JSON support
  - OpenSocial, JavaScript APIs allowing Google Gadgets to access profile data (identity attributes)

Social networking related content aspects

- Specific standards for social aspects
  - TAGs – Common Tag Vocabulary
  - Microformats for embedding meta data into UIX code (e.g. using specif CSS names in HTML)
  - OGP (Open Graph Protocol), meta data vocabulary for documents, such as Facebook's like button
  - SIOC (Semantically-Interlinked Online Community) vocabulary for representing user-generated content

Content aspects regarding privacy concerns

- Specific standards for Privacy related aspects, e.g.
  - POWDER (Protocol for Web Description Resources) – W3C language for describing groups of resources
  - P3P (Platform for Privacy Preferences) W3C Recommendation, which allows Web site operators to express their data collection, use, sharing, and retention practices
- And many more important aspects in the context of a Social Web exist, e.g. regarding, Activity / Real-Time Web
  - XMPP (Extensible Messaging and Presence Protocol) is an IETF RFC for the near real-time transfer of XML data
  - Atom and Pubsubhubbub, XML-based IETF RFC Atom ("RSS-like")
  - ActivityStreams is an Atom serialization for activity streams such as status updates

CHAPTER://16

- User Interface Experience

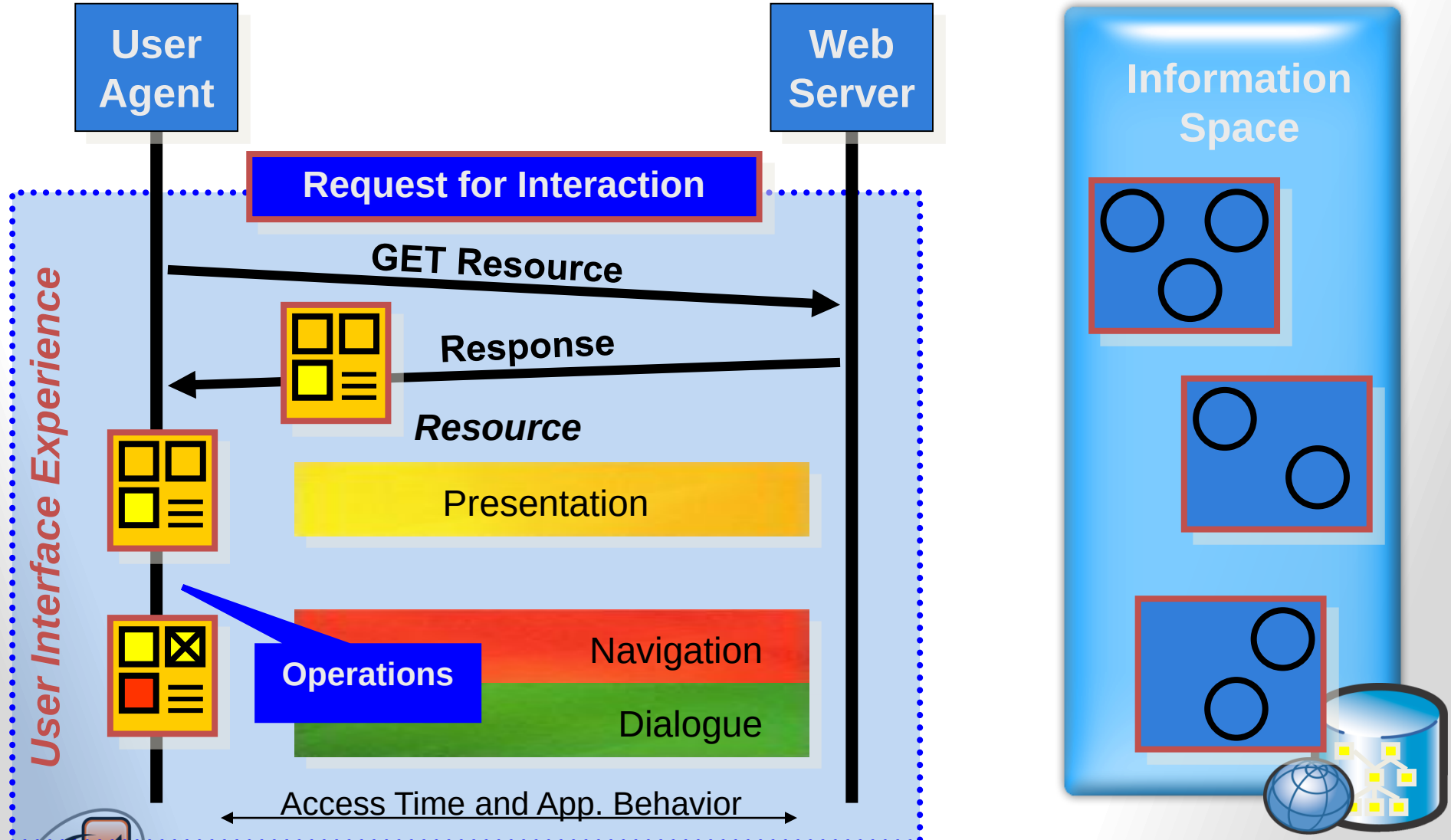
Goal

- Aspects of User Interface Experience (UIX) Dimension is to *Design with Hypertext in mind*
  - 1) Presentation
  - 2) Navigation
  - 3) Dialogue
- Focus on the interaction between human and solution

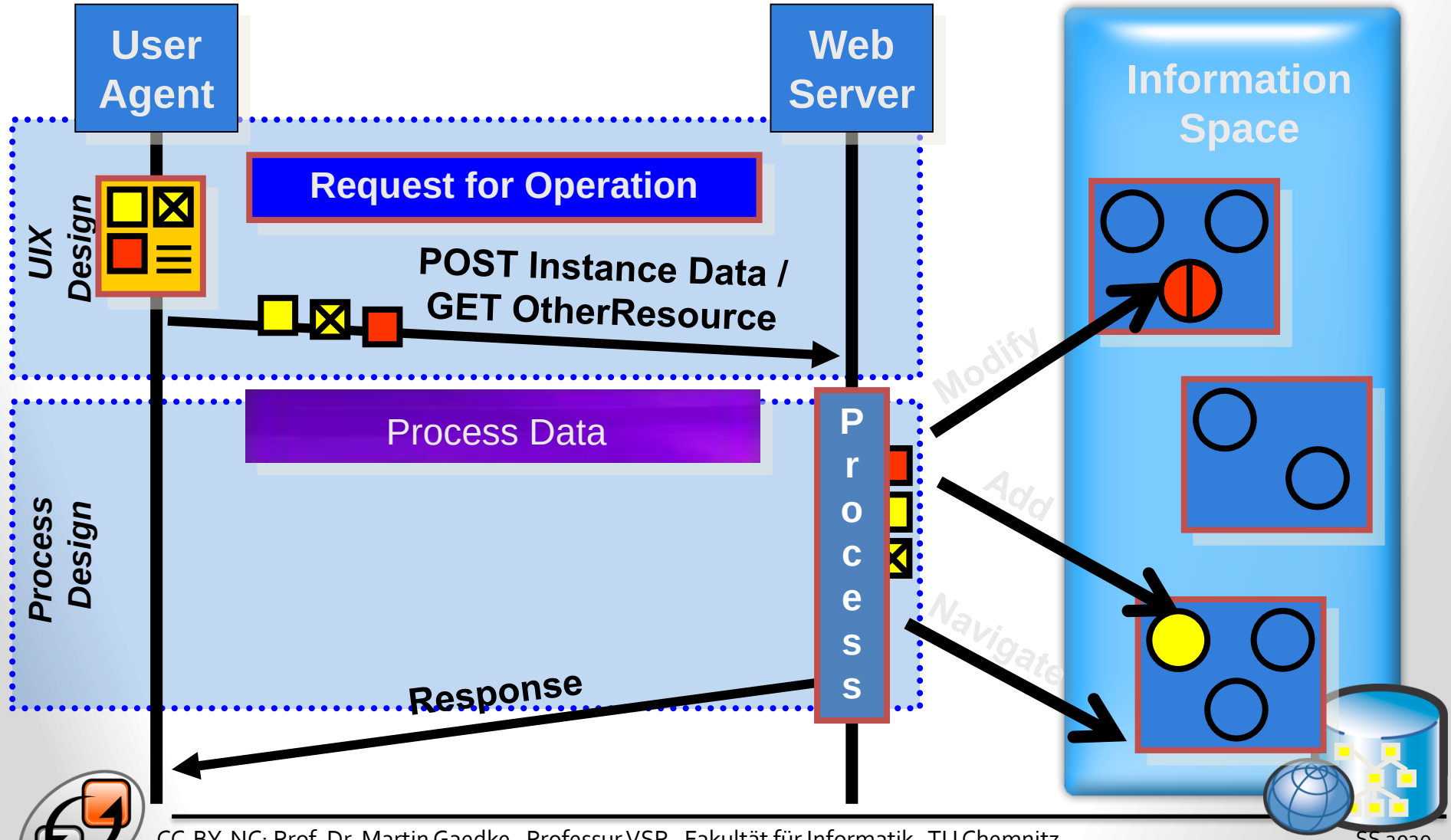
Interaction

- **Interaction** – Process involving a mode of operation in which there is a continual exchange of information between an actor and an Information Space
  - Operations (Actions) include exploring/browsing, modifying, and responding

Web Interaction Model



Web Interaction Model



SECTION://1

- Presentation Design

Goal

- Design the final audio and visual presentation
  - Shape logical presentation's layout
  - Shape physical presentation's 'look and feel'
- Challenges
  - Design maintainable, adaptable, and reusable presentation
  - Enable UI-Reuse → customization
  - Psychological problems
  - Cultural Aspects
  - Create an experience!

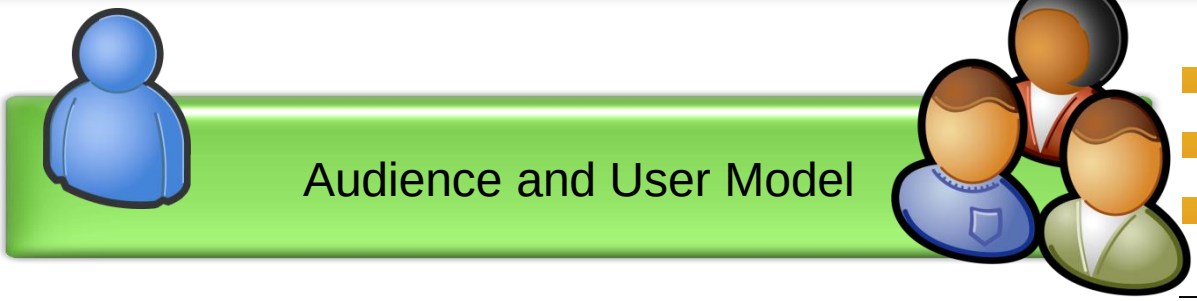
Typical Problems

- Problem – Accepting importance of design
  - Technical oriented people in the team often attack visual elements as wasteful
- Social Aspects
  - Presentation design must add to the site in a meaningful way
  - Transfer a message to the "viewer", e.g. emotions
- Web Engineers are not Graphics Designers, but
  - Bad presentation may make a perfectly engineered solution unusable
  - Badly engineered solution may not become usable by good presentation (but will look good ;-)

Some Facts to keep in Mind

- No bullet-proof rules for Presentation Design exist
- Marketing department will provide some guidelines, e.g. Corporate Identity (CI) Styles
- Corporate Web Sites must promote and reflect the company's brand
- Intranet/Extranet Sites usually pay less Attention about Presentation Design

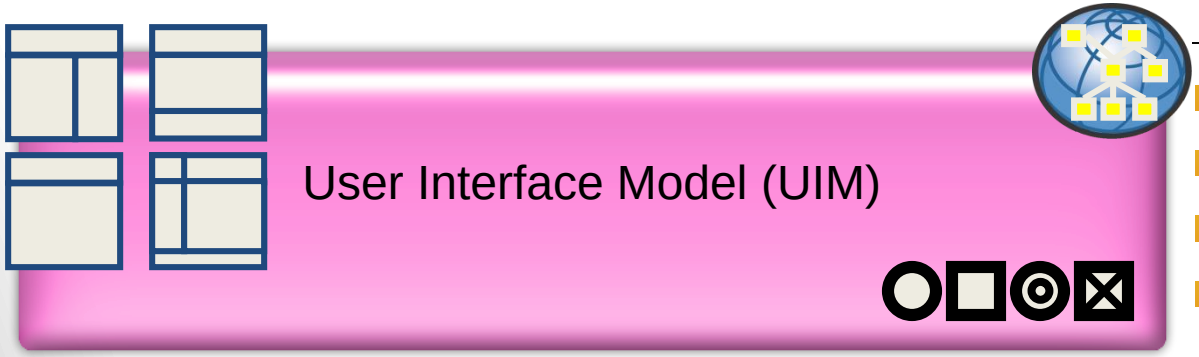
Logical & Physical Aspects



- Goal
- Expectations
- Experience/Device

- Tone, Colors
- CI-Rules, Trends
- Audio/Video

Physical Design

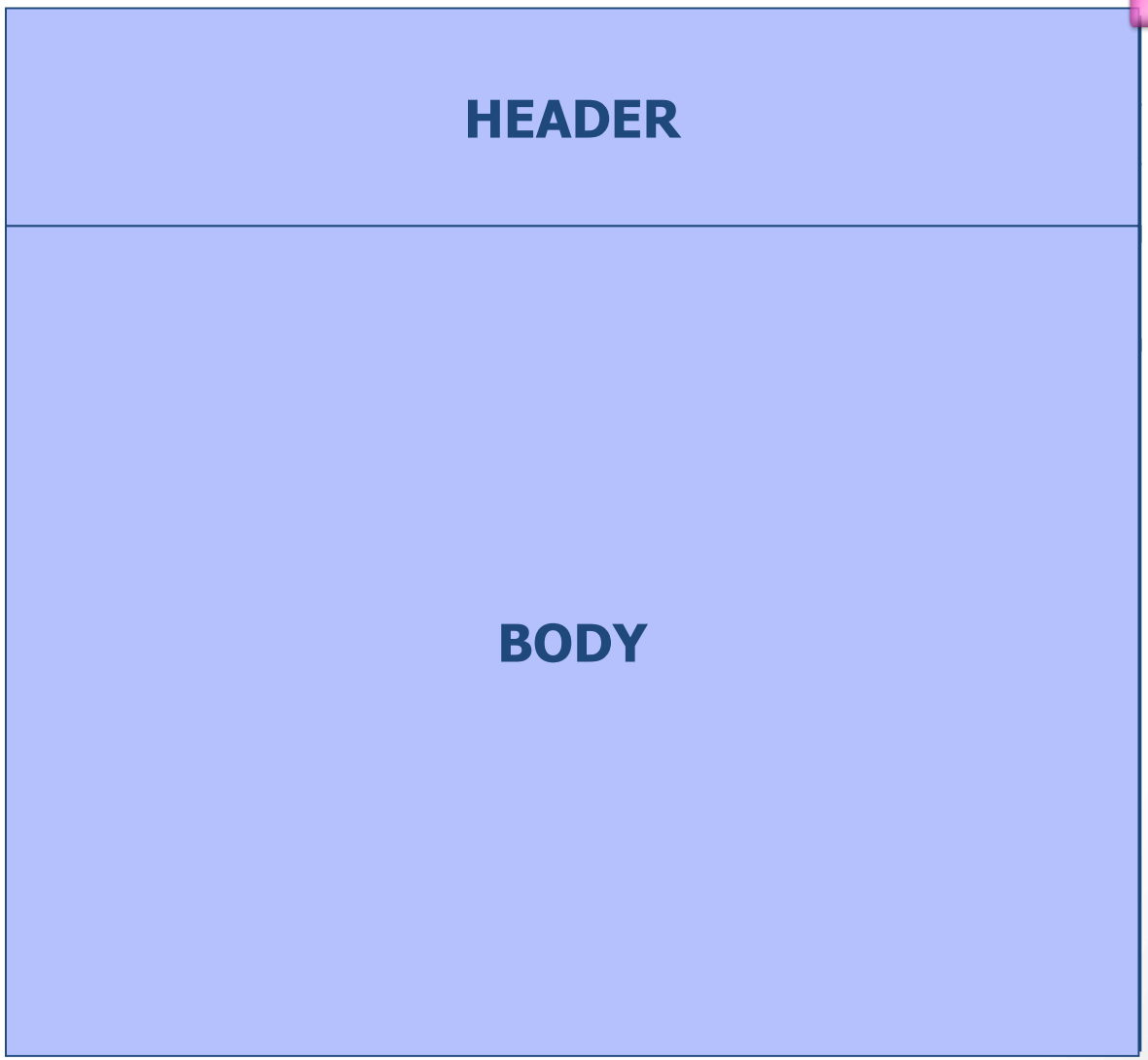


- Layout / Tiles
- Controls, Dialogues
- Exploration
- UI Pattern

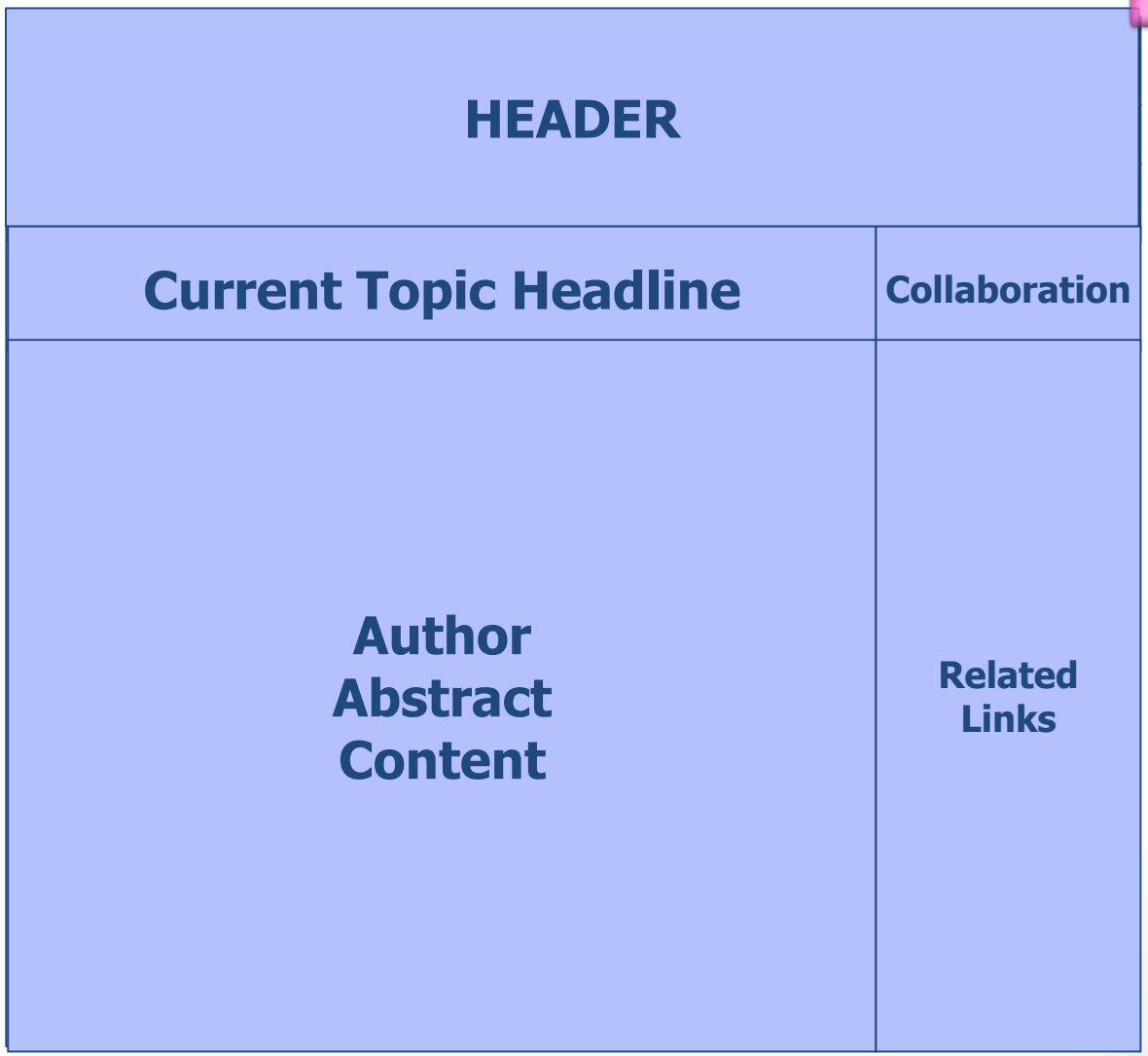
Logical Design

- Distributed Data
- Core App Logic
- Structural Linking

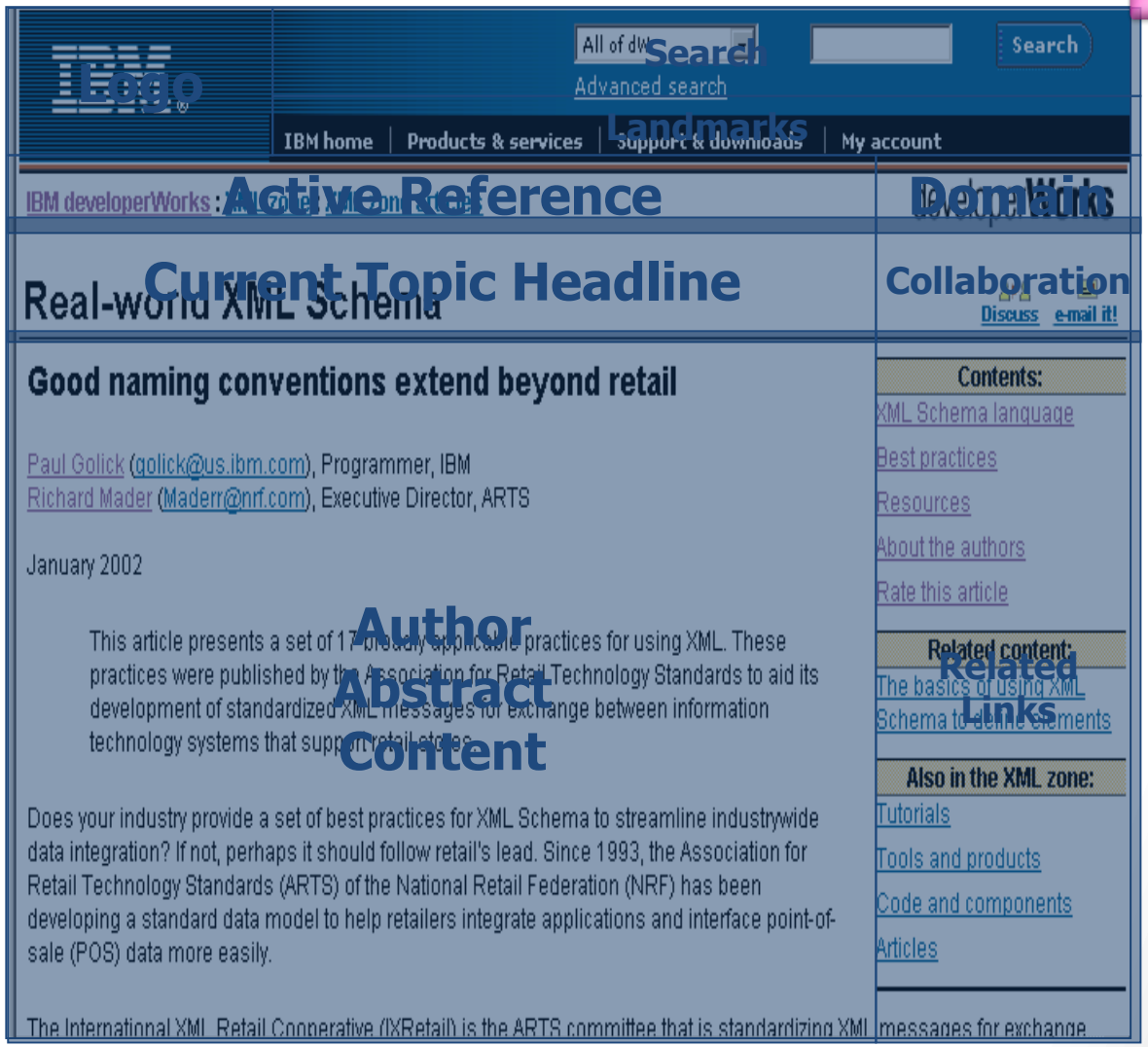
Tiles



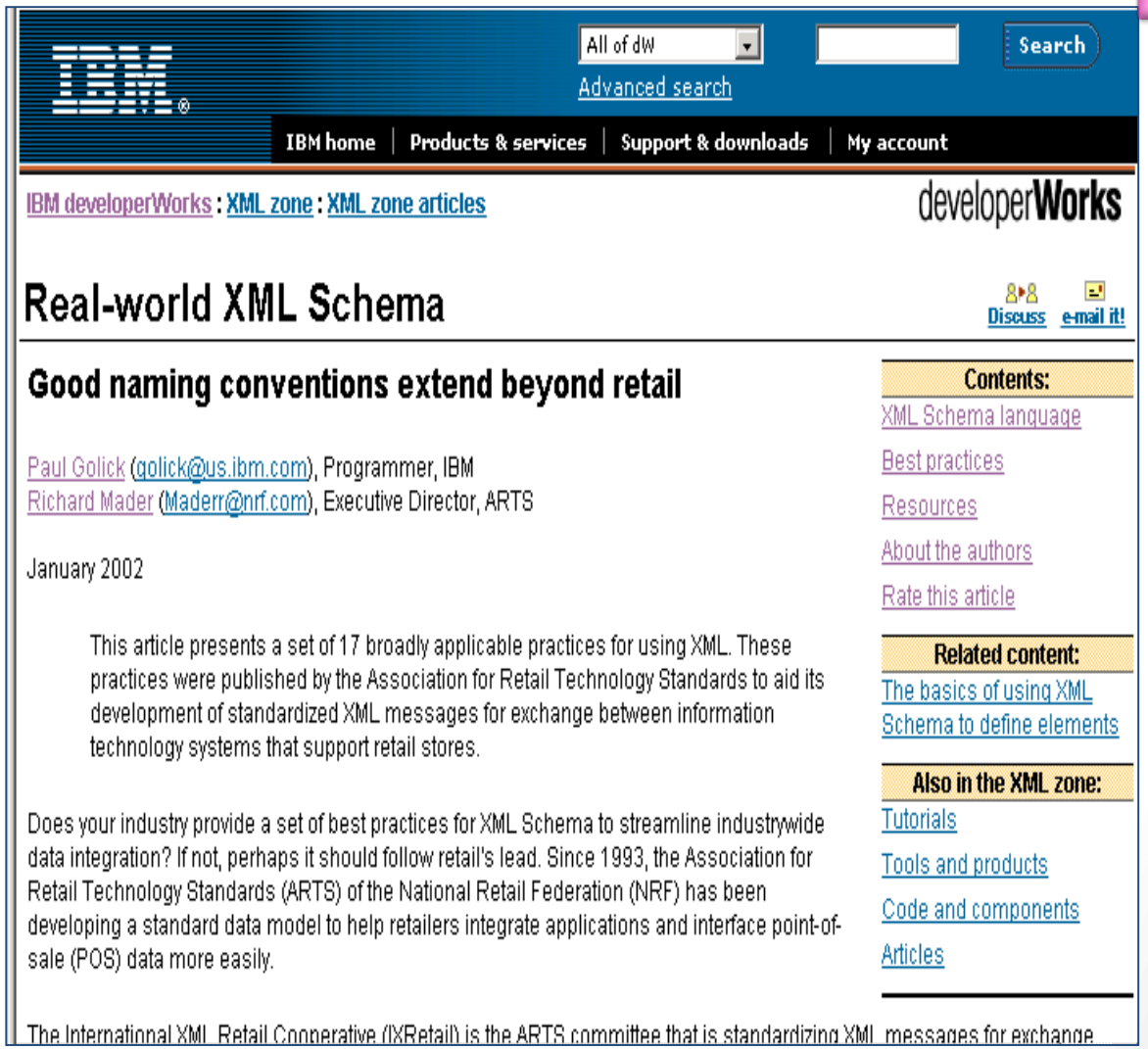
Tiles


HEADER
Current Topic Headline
Collaboration
Author
Abstract
Content
Related
Links

Tiles

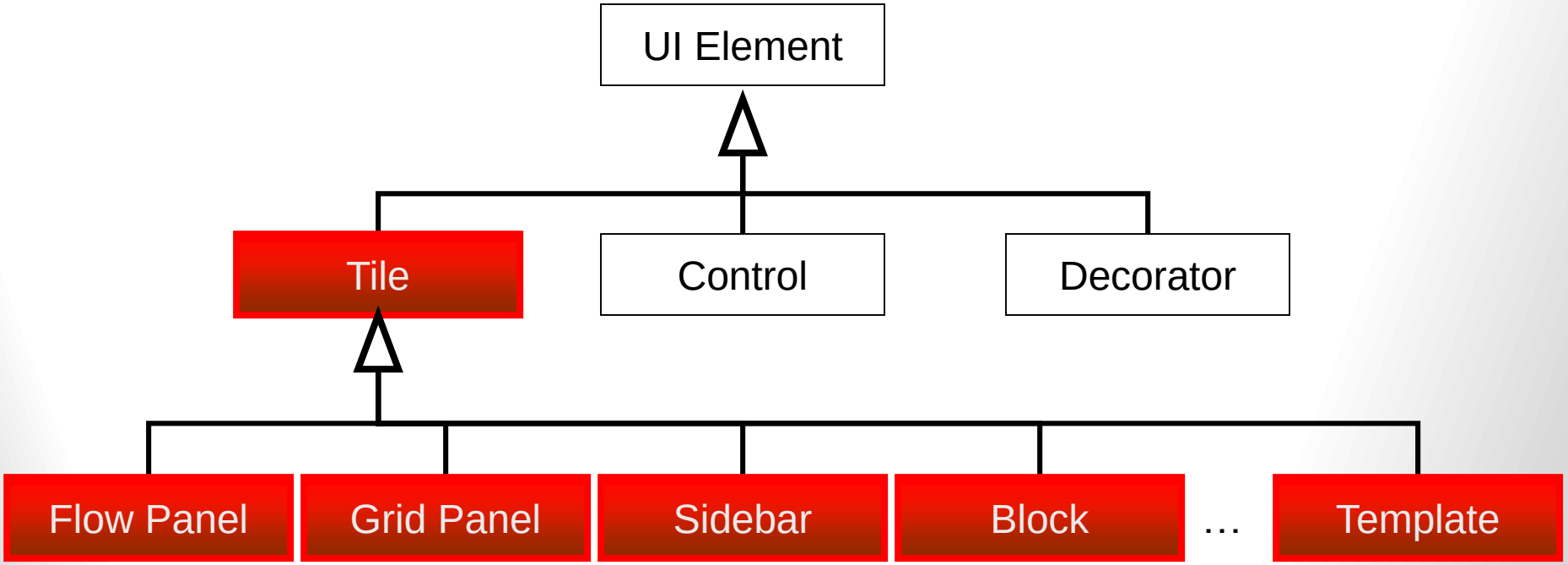

Search
Landmarks
Active Reference
Domain
Current Topic Headline
Author
Abstract
Content
Related Links
IBM home
Products & services
My account
Advanced search
Search
Real-world XML Schema
Good naming conventions extend beyond retail
Paul Golick (golick@us.ibm.com), Programmer, IBM
Richard Mader (Maderr@nrf.com), Executive Director, ARTS
January 2002
Collaboration
Discuss
e-mail it!
Contents:
XML Schema language
Best practices
Resources
About the authors
Rate this article
Related content:
Also in the XML zone:
Tutorials
Tools and products
Code and components
Articles

Tiles


All of dW
Search
Advanced search
IBM home
Products & services
Support & downloads
My account
IBM developerWorks : XML zone : XML zone articles
developerWorks
Real-world XML Schema
Discuss
e-mail it!
Good naming conventions extend beyond retail
Paul Golick (golick@us.ibm.com), Programmer, IBM
Richard Mader (Maderr@nrf.com), Executive Director, ARTS
January 2002
This article presents a set of 17 broadly applicable practices for using XML. These practices were published by the Association for Retail Technology Standards to aid its development of standardized XML messages for exchange between information technology systems that support retail stores.
Does your industry provide a set of best practices for XML Schema to streamline industrywide data integration? If not, perhaps it should follow retail's lead. Since 1993, the Association for Retail Technology Standards (ARTS) of the National Retail Federation (NRF) has been developing a standard data model to help retailers integrate applications and interface point-of-sale (POS) data more easily.
The International XML Retail Cooperative (IXRetail) is the ARTS committee that is standardizing XML messages for exchange
Contents:
XML Schema language
Best practices
Resources
About the authors
Rate this article
Related content:
The basics of using XML Schema to define elements
Also in the XML zone:
Tutorials
Tools and products
Code and components
Articles

Core Tiles

- Tiles – Structure and arrange UI Elements



Controls

- Tiles are filled with controls
  - Tiles provide a structure for rendering controls

- **Control** – Describes a way of accessing instance data of the information space with an interaction handler
  - Abstracts - UI Element treated independent of its presentation
  - Separates presentation of content and its interaction
  - Treats Data as a first class citizen
  - Takes a uniform approach to presentation experience
    - I.e. captions, help text, tabbing and keyboard shortcuts

Control Examples

HEADER
Control

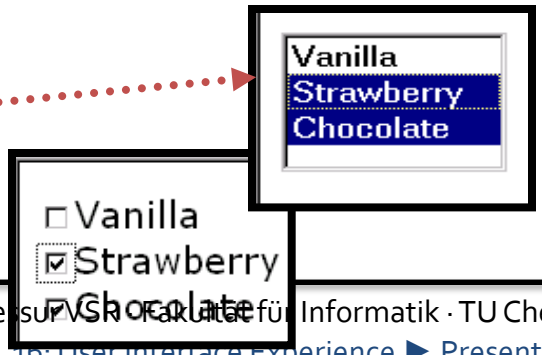
Title

Navigation
Context help

Color - customized by user

Search

Selection
Controls

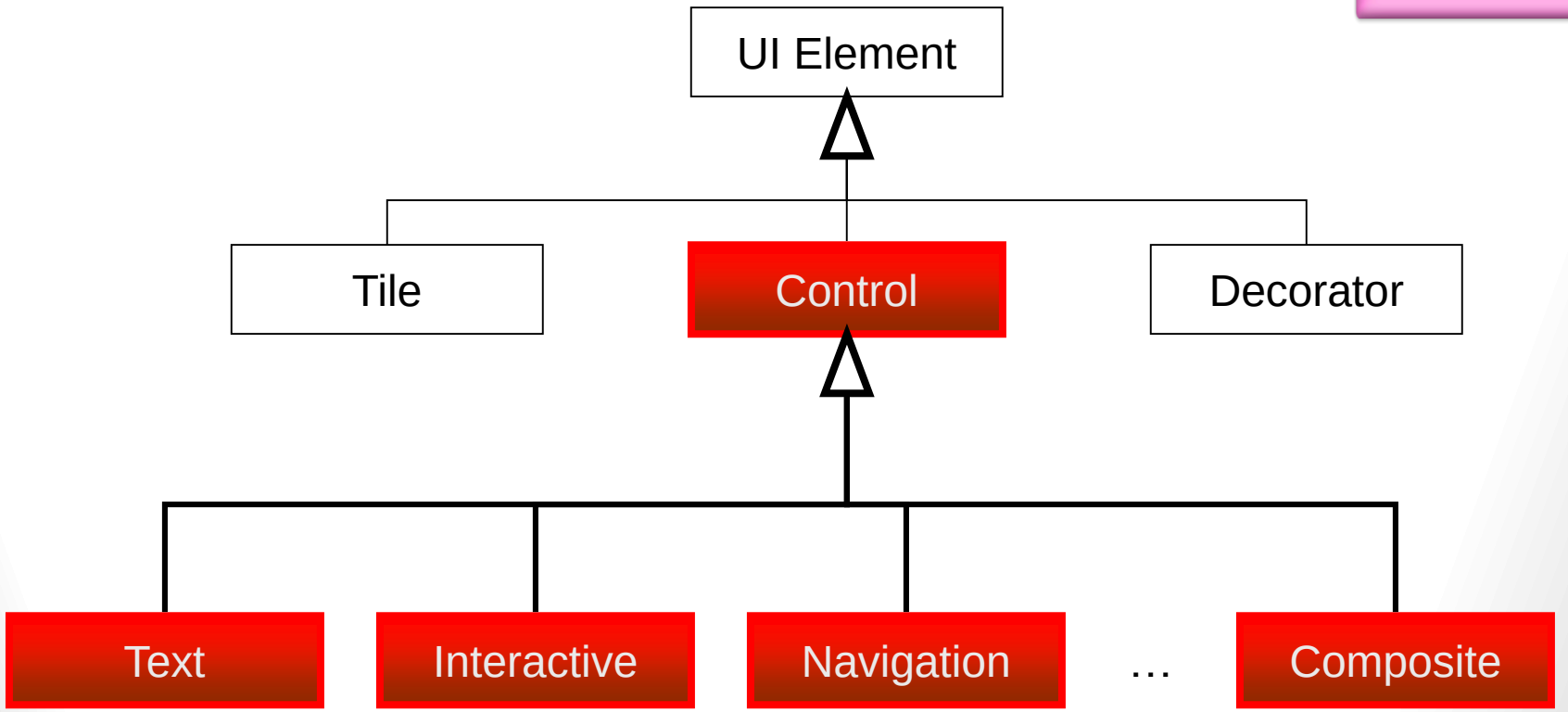


Control Aspects

- Controls are UI Components
  - Allow for early testing, e.g. loading times
  - I.e. bandwidth or kilobytes per control
  - I.e. bandwidth per page = Sum of all Tiles and containing controls

- **Control Model** – Defines a set of abstract User Interface controls
  - Allows to manage abstract controls and their mapping to physical representatives
  - Help for maintenance and reuse
  - Improve consistency of the overall site
  - Object-oriented concepts may be applied
  - Examples
    - Define all icons and their meaning in a consistent way
    - Define text, size, colors, etc.

Simplified Control Model



- Text building blocks
- Inline elements

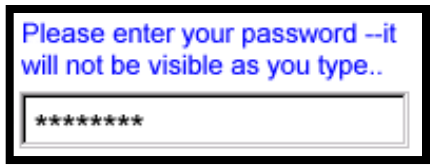
- Form Elements
- PlugIns
- Applets
- ActiveX Control

- Hypermedia structures
- External links
- Internal links
- Navigation pattern

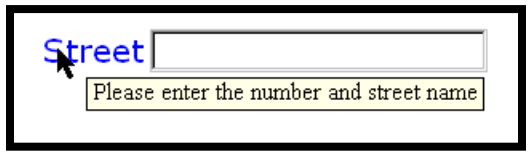
- Composition of UI elements
- consider rules for composing elements, e.g. WAI compliance)

Interactive Controls

Secret



Input



Upload

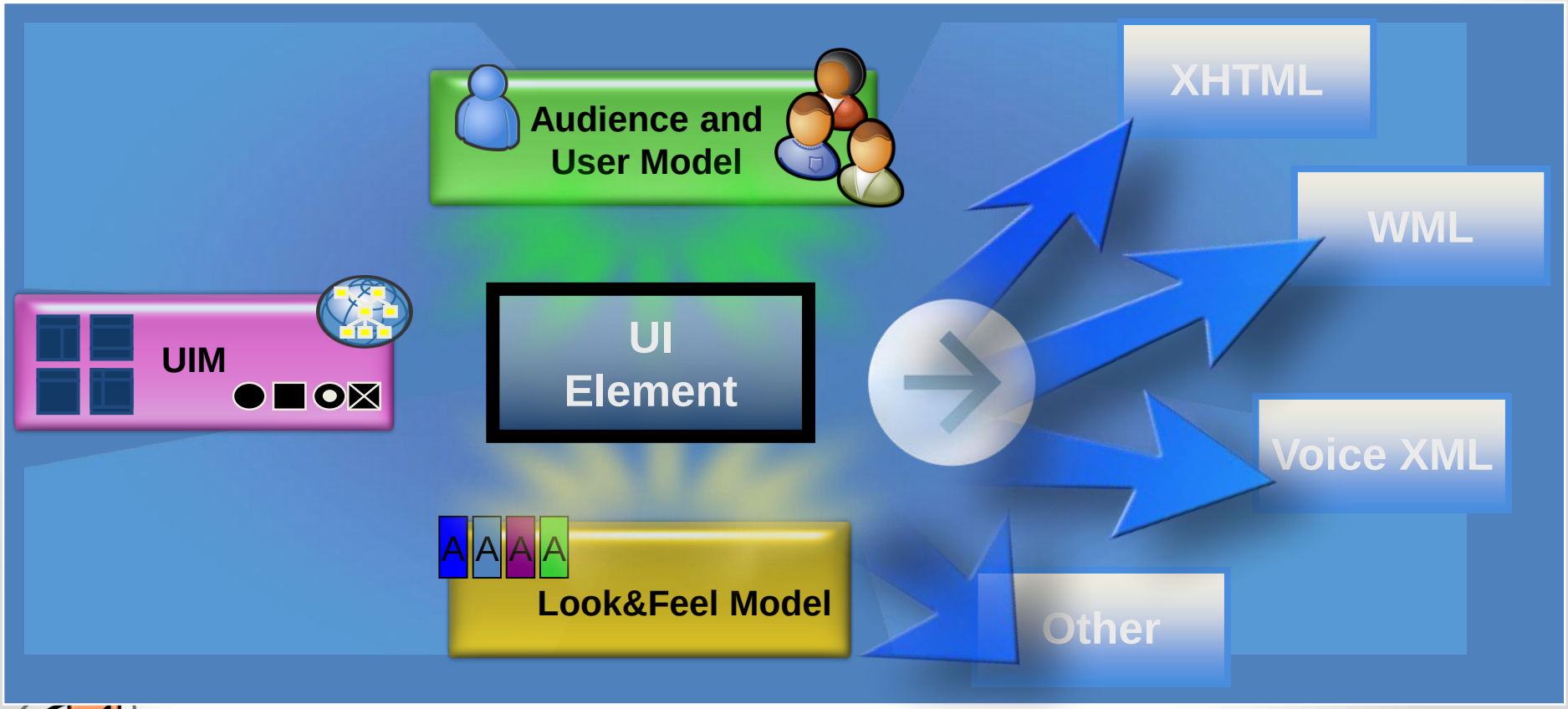
selectOne

selectMany

Pictures Source: XForms 1.0
W3C Working Draft
28 August 2001,
http://w3.org/TR/xforms

Shift to Physical Design

- Develop UI Model representatives (physical UI elements) for each user group or audience



Physical Design Examples

- Map and decorate/style logical elements
  … based on audience / culture

SECTION://2

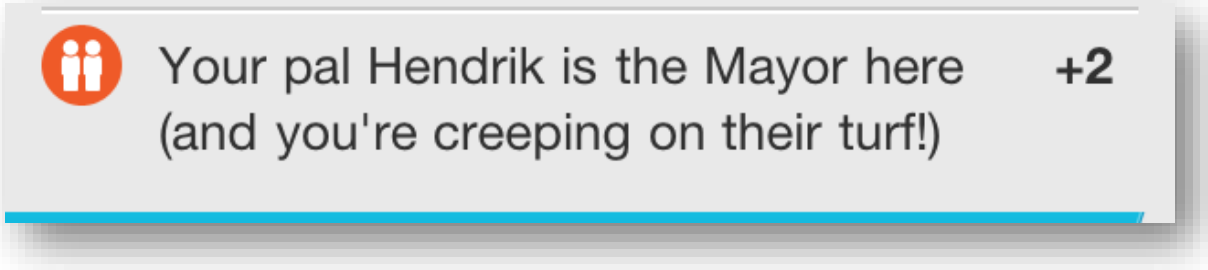
- Presentation Design & SEA

Presenting Data from the Social Point of View

- Which Language to use when presenting personalized data, e.g. Identity Attributes, Personal Digital Items
- **Pattern**: Language of YOUR versus MY
  - Problem
    - Understanding use of possessive pronoun on personalized content
  - Example
    - Your – addresses the fact that the users works with an application, e.g. Your Address, Your tags
    - My – might make the user feel he/she added some data, because the user "owns" the data, but in fact the application did
  - Solution
    - Use: Your (when labelling data, e.g. Your address)
    - Use: User name (when labelling data, e.g. Peter's address)

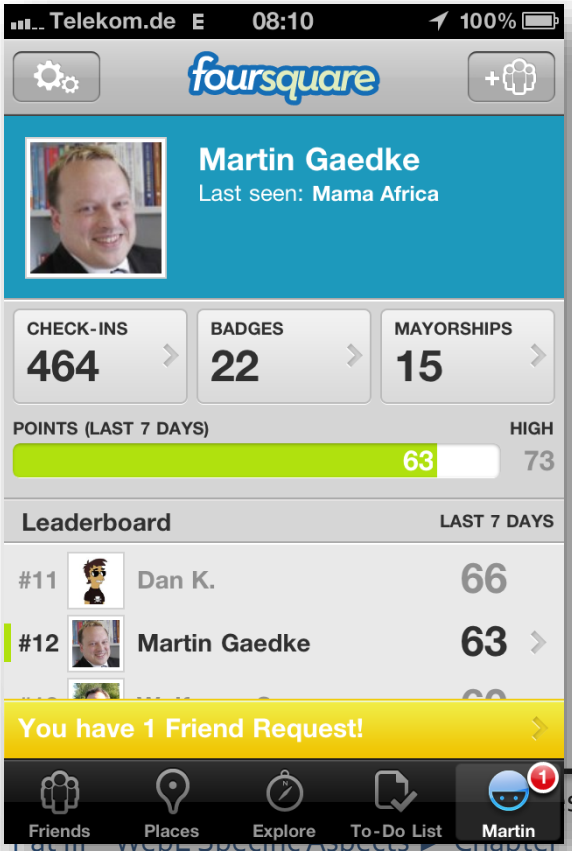
Other social best practices & patterns

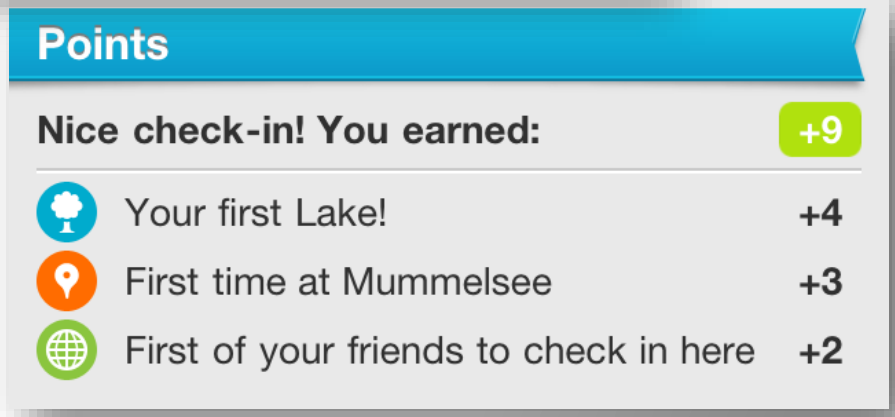
- Several other patterns regarding social/language exists,
  - ▶ Cf. Yahoo! Developer Network / Patterns, e.g. Pattern "Talk Like a Person"



Entertainment / Gamification Aspects

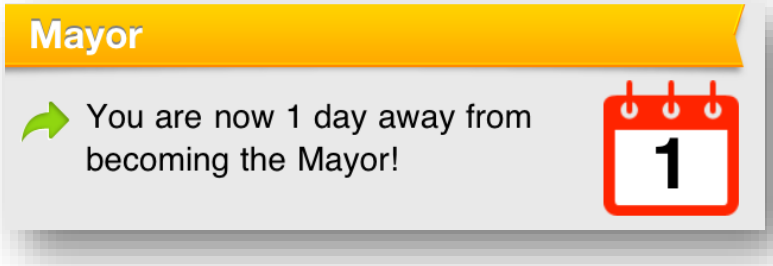
- Presentation / Implementation of gaming mechanics
  - ▶ Points, e.g. scoreboards / scorecards, leaderboards
  - ▶ Rewards, e.g. badges, ranks, levels, leaders

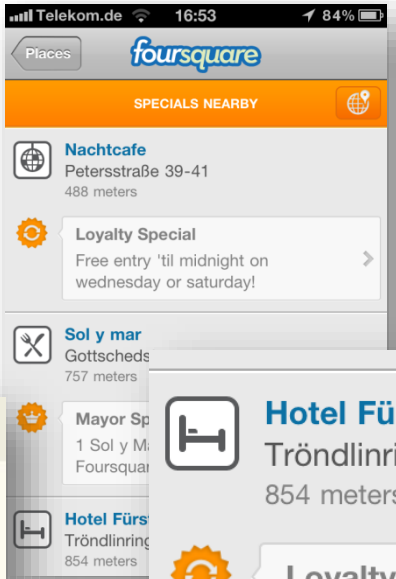


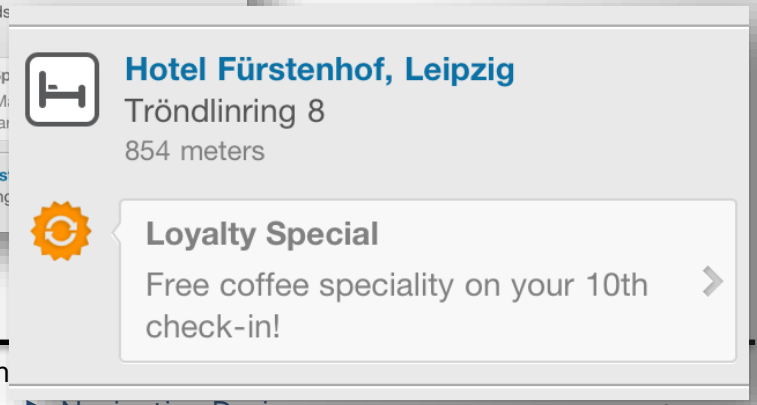


Entertainment / Gamification Aspects

- Pattern: Trainer Language
  - ▶ Use motivating and engaging language
  - ▶ Allow user to "accomplish a mission"
  - ▶ System should announce goals to motivate behaviour change







SECTION://3

- Navigation Design

Goal

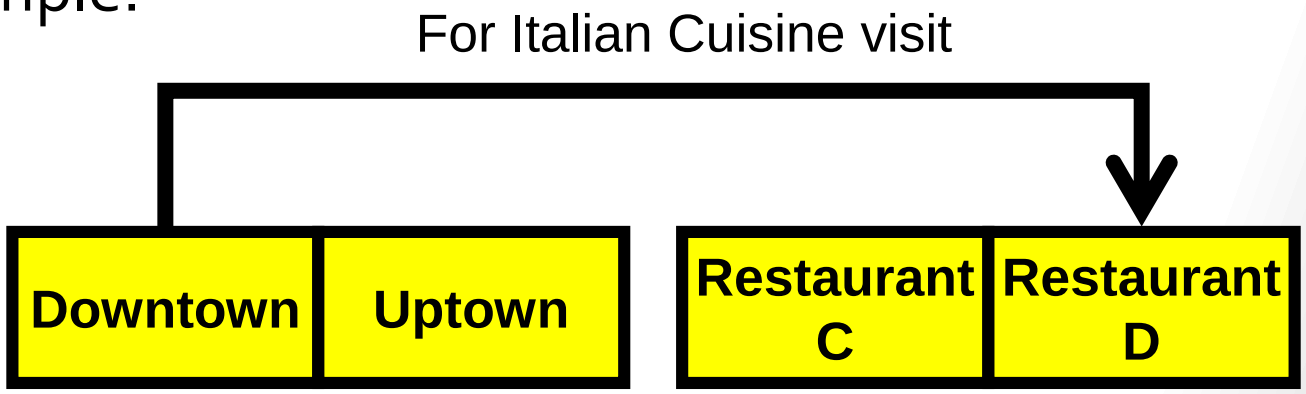
- Improve ease of use and access to information and processes
  - Focus user's attention — unnecessary information are out of the picture
  - Design link structure for "understandable" navigation
  - Develop understanding of semantic associations
  - In other words: *Focus on Hypermedia idea in general*
- Challenges
  - Improve "flow" of the user experience
  - Creating meaningful (Navigation) structure – without cognitive overload
  - Build secure connections between features
  - Reuse of navigational pattern

Hypermedia Linking

- **Link** – Association between a Source Node and one or many Targets Node
- Taxonomy based on type of information
  - Structural links
  - Associative links
  - Referential links
  - Contextual links
- Note: WWW does not provide mechanisms for differentiating link types

Associative Links

- Define **semantic** relationship between information elements
  - "Standard" Hypermedia Idea
  - E.g. for more information on X refer to Y
  - Independent of the underlying data structure (This is not structural linking!)
- Example:



Referential Links

- Provides Link between item and an explanation
- Item at end of Referential Link exists because of the existence of the other item
- Example – Fragment of Room description
  - The business rooms are equipped with WLAN and On-Demand video…
  - Definition: Wireless network adapters connect individual …refer to this as a Wireless Local Area Network (WLAN).

Contextual Links

- Context - “the set of facts or circumstances that surround a situation or event”
  - Synonyms: situation, phase, position, place, point, standing, status, occasion, environment, location, dependence …
- Provides relation between items based on context
  - Often between items in Information Space and real world (cf. Ubiquitous Computing)

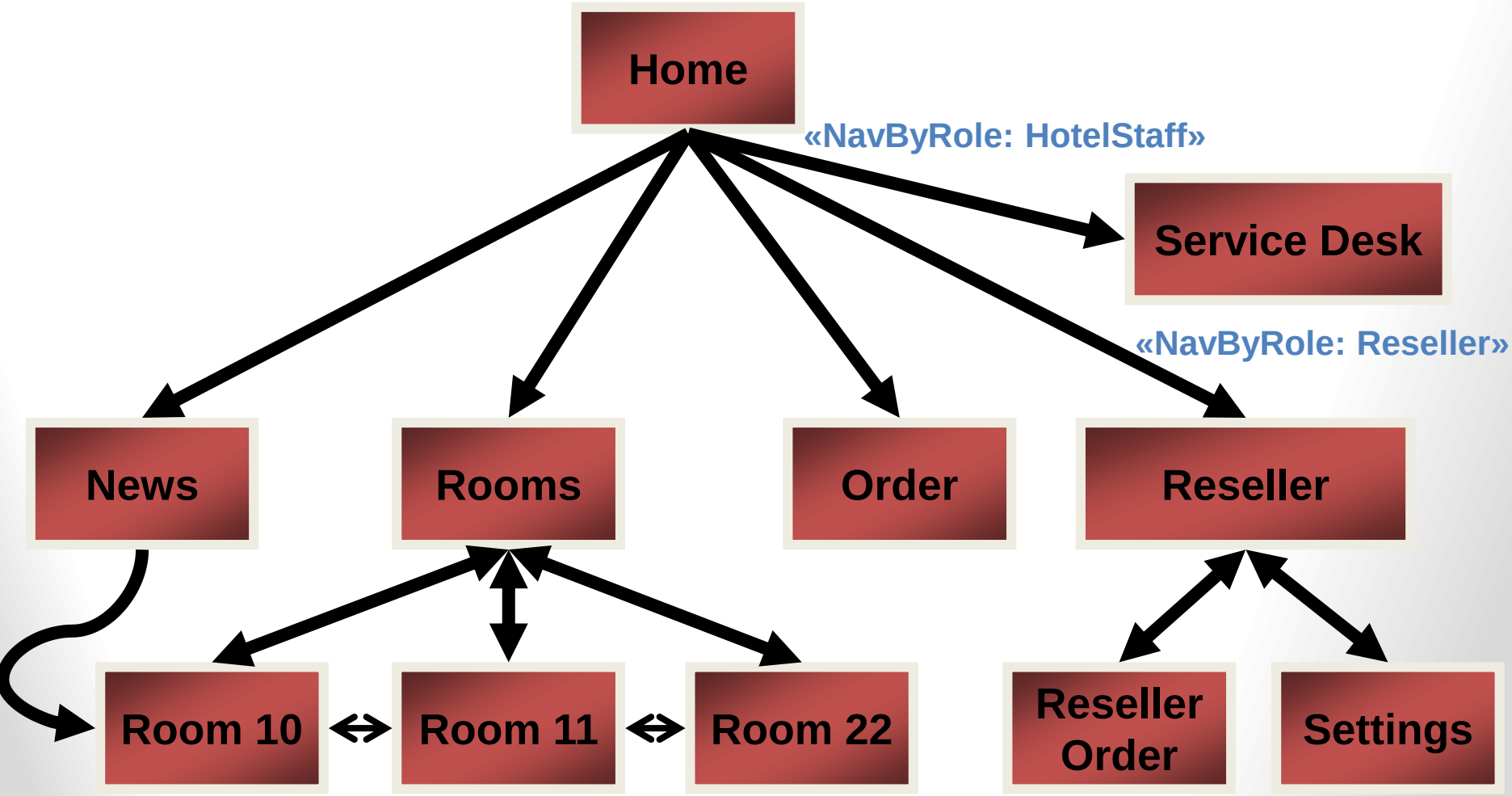
Contextual Links

- Web Applications for Mobile Devices
  - Becomes more important as technology for mobile devices evolves
  - Referring to information *surrounding* the system
  - E.g. Where a computer is used, by whom, who else is around, what are they doing, …
- Example
  - Mobile Devices with Location Information
    - Project at the University of Lancaster, UK
    - Project NUKATH, University of Karlsruhe, Germany

Navigation Design Process

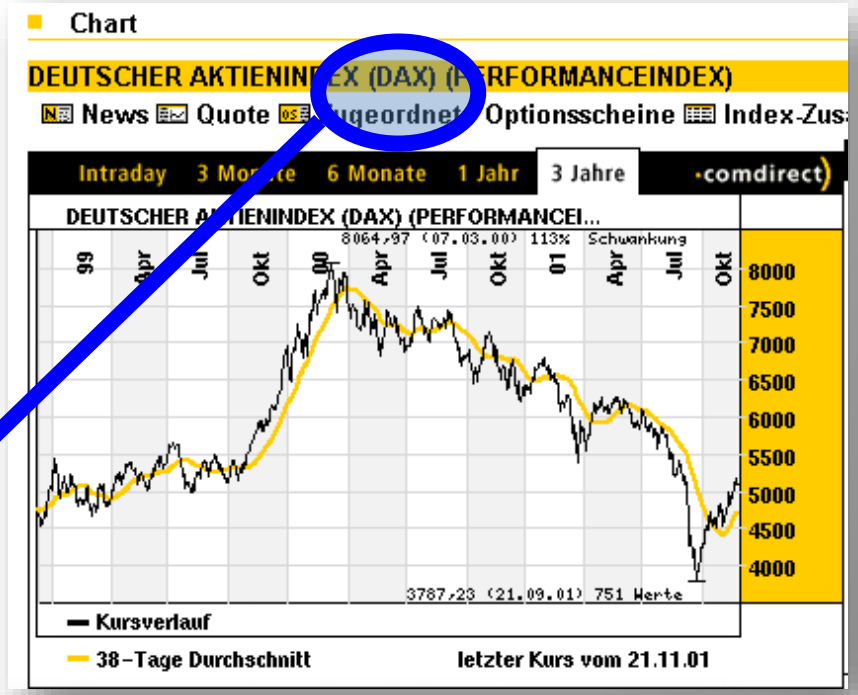
- Navigation Design involves many different types of activities
  - Logical Design connect Business Process (Site Map approach)
  - Domain specific navigation design – enhancing information access and task oriented use of processes
  - Design Navigational context with external resources
  - Enhancing user experience with Navigational Design Pattern
- Monitor and support:
  - Access by Navigation
  - Access by Direct Addressing
  - Access by Search Request
  - Access by Browsing

Site Map


Home
«NavByRole: HotelStaff»
Service Desk
«NavByRole: Reseller»
News
Rooms
Order
Reseller
Room 10
Room 11
Room 22
Reseller Order
Settings

Domain-specific Navigation

- Referential Links to access information/help
- Access processes / features as described in Use Case scenarios – guiding user principle



DAX

Abkürzung für Deutscher Aktien-Index; enthält als gewichteter Index 30 Aktien führender Unternehmen, die an deutschen Börsen amtlich gehandelt werden. Sein Wert wird minütlich an der Börse ermittelt. Der DAX zeigt Kursschwankungen an hinsichtlich des Zieles, für den deutschen Markt international eine höhere Markttransparenz zu erreichen und zugleich die Basis für Terminkontrakte an der Deutschen Terminbörse zu schaffen. Die Auswahlkriterien für die in den DAX einzubeziehenden Gesellschaften sind:

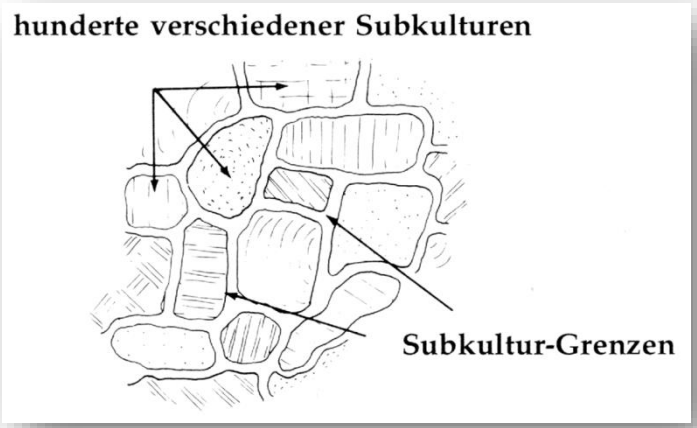
- Umsatzstärke
- Börsenkapitalisierung
- Vorhandensein früher Eröffnungskurse
- Branchenrepräsentativität

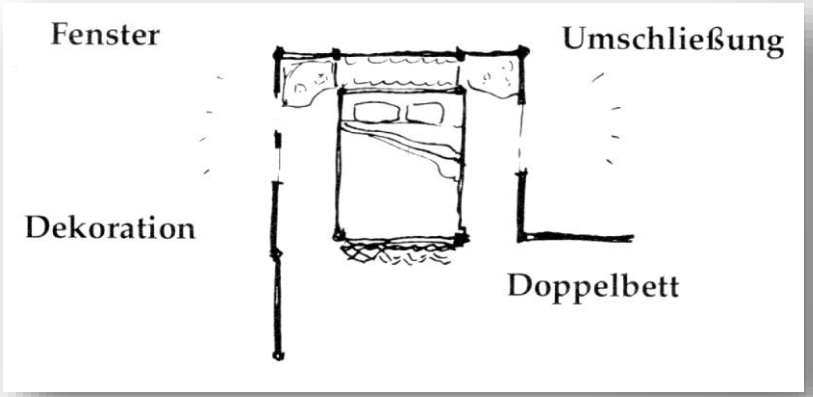
Pattern

- Architect C. Alexander
  - Collect structures of urban development and other architectural artifacts of any granularity
- Book *A Pattern Language* 253 patterns
  - Could be used to build rooms, buildings, places, cities…
  - Over ten years work experience
  - Each pattern is presented in the same format

Patterns: Reusing Experience

- Source: C. Alexander et al.: A Pattern Language, Oxford University Press, New York, USA, 1977





About a Pattern

- Alexander says:
  - Each pattern should be expressed as a solution to a conflict between forces
  - Empirical evidence is needed to show that the pattern works as claimed
  - Patterns should make us feel good
- A Pattern tells us how to go from a bad solution to a good solution

Patterns Applied

- A **Design Pattern** – "describes a particular recurring design problem that arises in specific design contexts, and presents a well-proven generic scheme for its solution."
- Further exists
  - Analysis patterns
  - Architectural patterns
  - etc.

Describing a Pattern

- A **Description Approach**
  - **Name** – Short and indicates what to be solved
  - **Problem** – Description of conflicts within a given environment
  - **Discussion** – Explains the problem in more detail and describes empirical evidence of existence
  - **Solution** – Instructions for solving the problem
- Other Descriptions are in use
- Tip: It is useful to apply this kind of approach for remembering personal experience

Examples

- Examples discussed here are only a portion of patterns available today
  - Note: Pattern description here is not complete

- Valuable sources to visit:
  - Hypermedia Design Patterns Repository
    http://www.designpattern.lu.unisi.ch/
  - Patterns Home Page
    http://hillside.net/patterns/
  - An HTML 2.0 Pattern Language
    http://www.anamorph.com/docs/patterns/

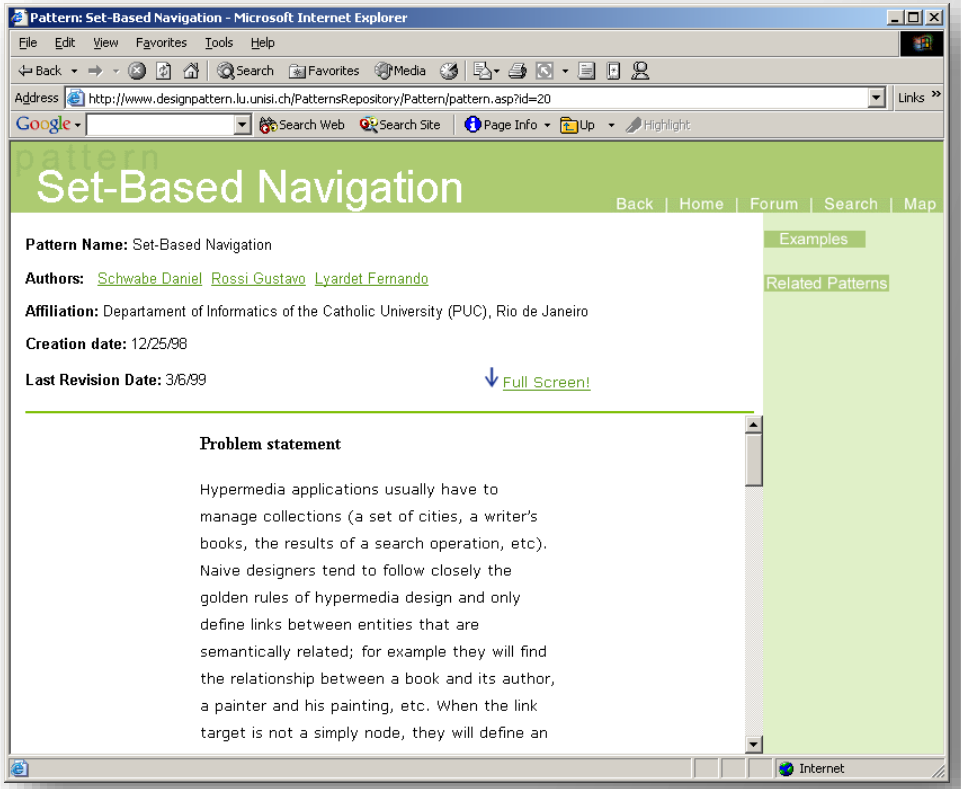
Hypermedia DP-Repository

- Pattern Overview
  - **Active Reference**
  - Advising
  - Analyze Organize Synthesize
  - Behavioral Grouping
  - Behavior Anticipation
  - Collection Center
  - Complex Entity
  - Guided Tour
  - Here I am
  - Hybrid Collection
  - **Index Navigation**
  - Info-Interaction Decoupling
  - Information Factoring
  - Information on Demand
  - Information-Interaction Coupling
  - **Landmark***
  - Navigation Strategy
  - Navigational Context
  - **News**
  - Node as a Navigational View
  - Opportunistic Linking
  - Process Feed-Back
  - Selectable Keywords
  - Selectable Search Engine
  - Selectable Search Space
  - **Set-Based Navigation**
  - **Shopping Process***
  - Simple Search Interface
  - Structured Answer

*** - not part of HDPR**

Pattern: Set-Based Navigation

- Name:
  - Set-Based Navigation
- Problem:
  - Naive Designers tend to follow closely the golden Rules of Hypermedia Design and only define Links between Entities that are semantically related…
  - The user will have to move from the index to a target

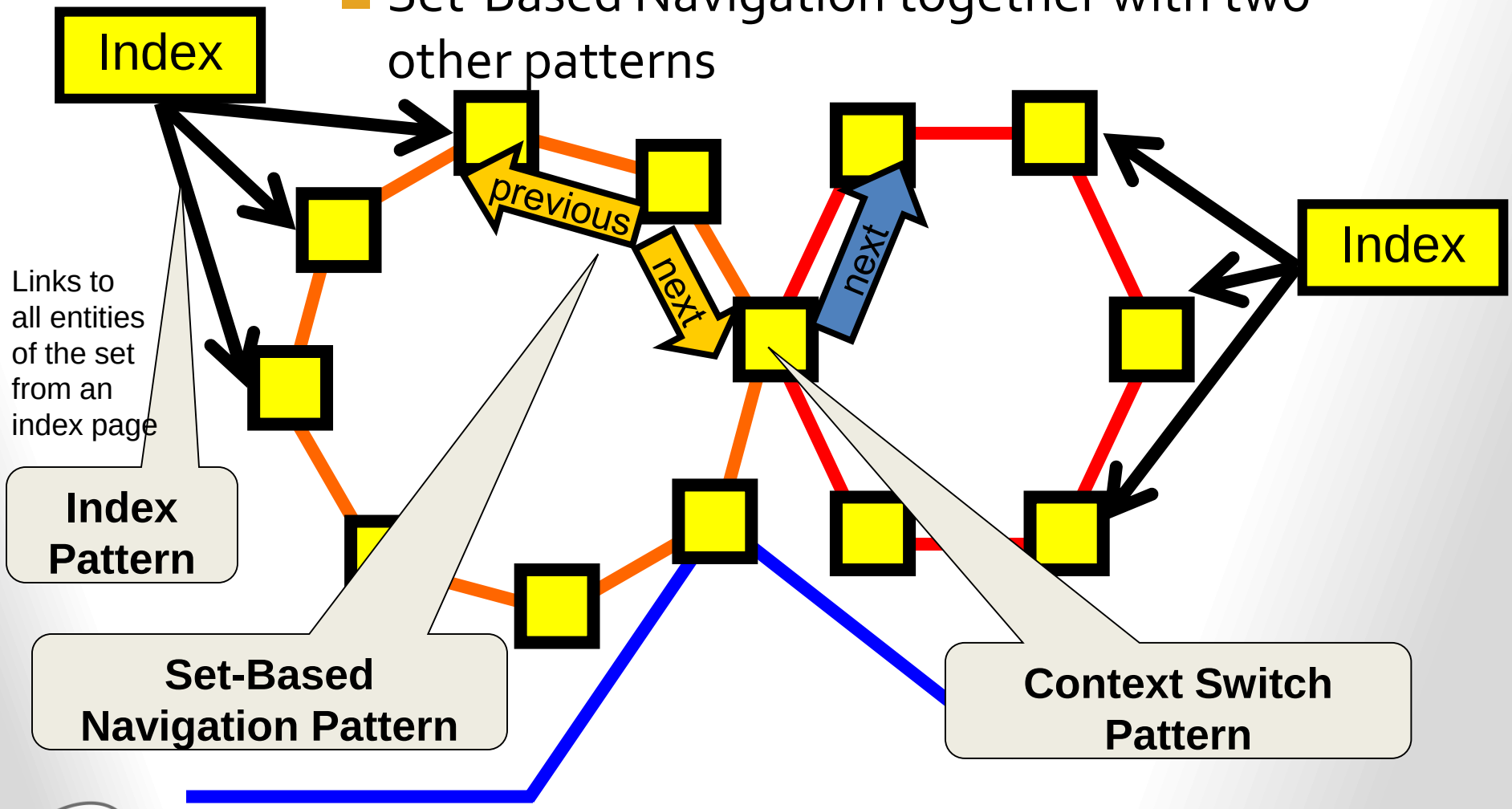


Pattern: Set-Based Navigation

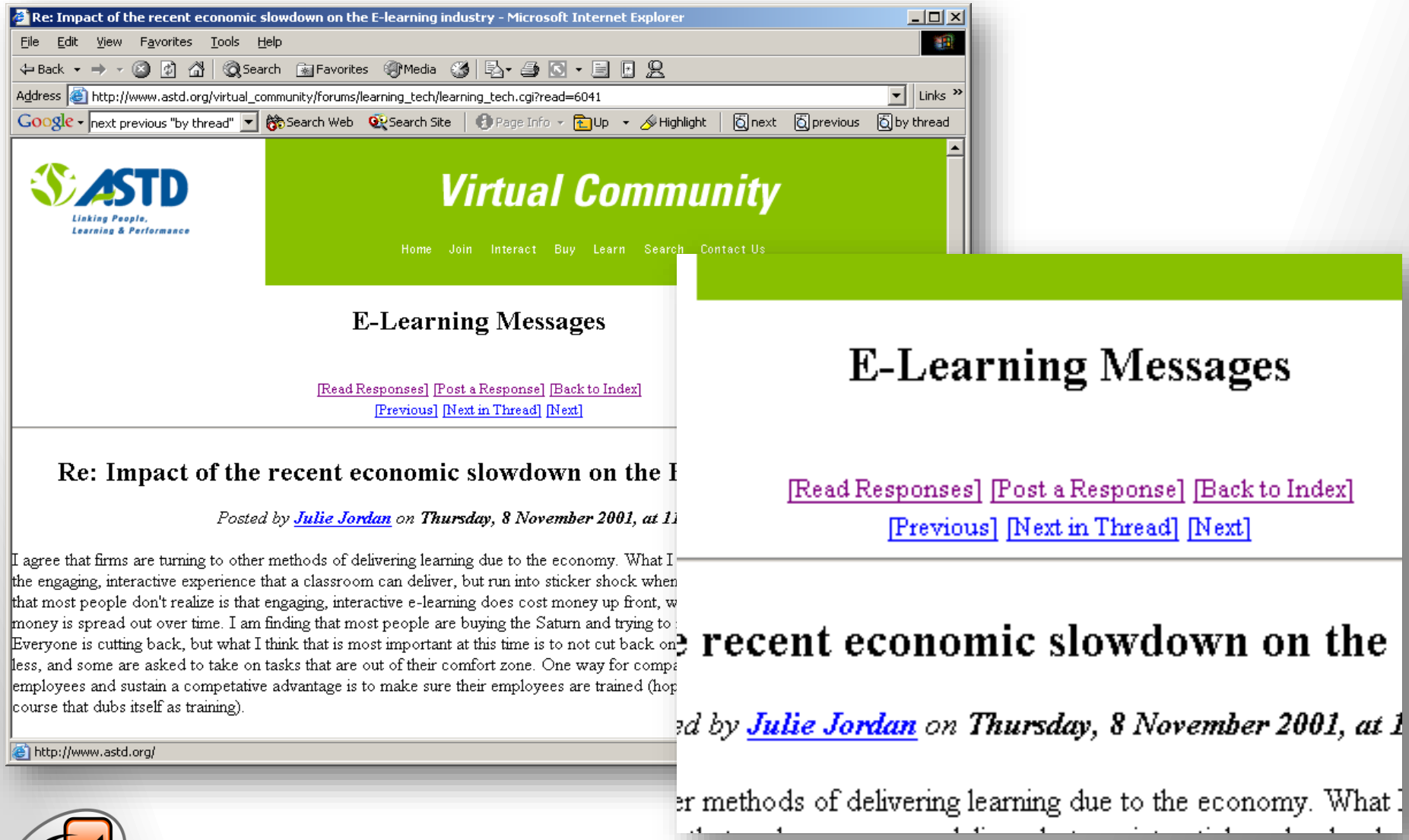
- Solution:
  - **Consider sets as first class entities**
  - Provide intra-set navigation controls
    e.g. "next" and "previous"
  - Combine Set-based Navigation with proper indices to make exploration easier
  - Node may appear in two different sets:
    use the Nodes in Context pattern

Pattern: Set-Based Navigation

- Set-Based Navigation together with two other patterns



Pattern: Set-Based Navigation

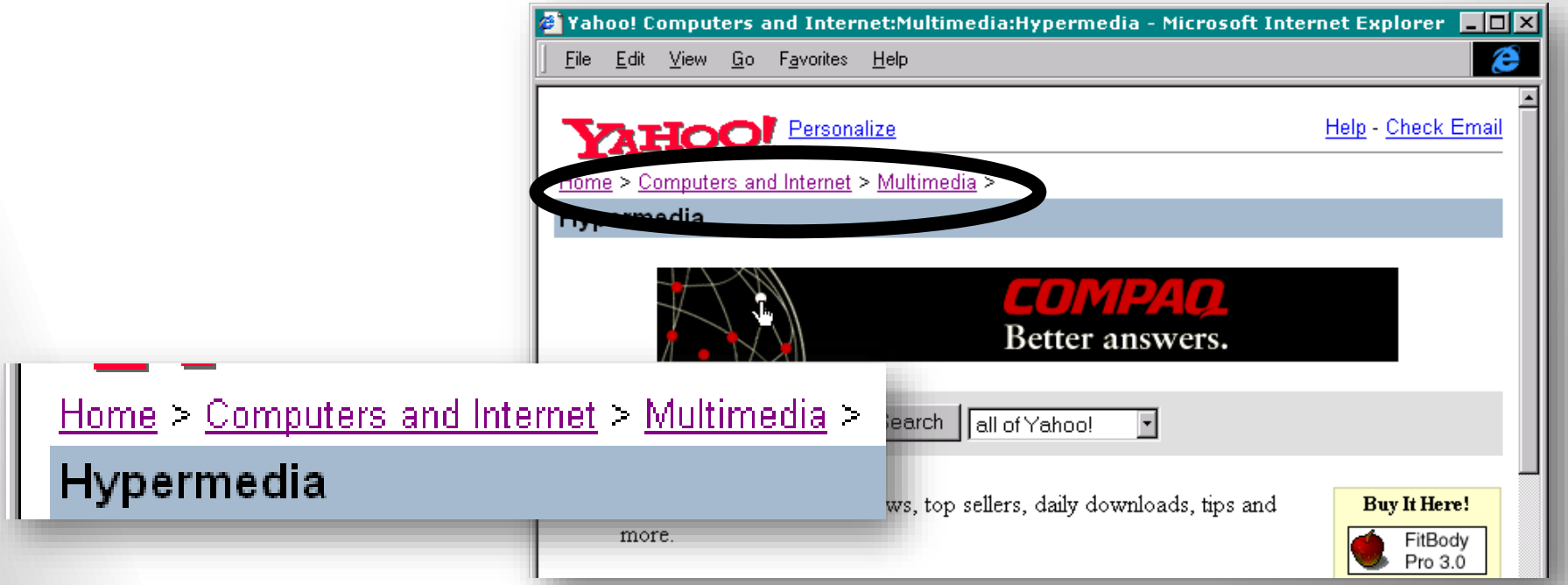


Pattern: Active Reference

- **Name**:
  - Active Reference
  - Also known as: Breadcrumb Navigation
- **Problem**:
  - We need a way to help the user understand where she/he is and where to go next. Indexes or other access structures provide only partial solutions

Pattern: Active Reference

- **Solution**:
  - Maintain an active and perceivable navigational object acting as an index to other nodes



Pattern: Landmark

- **Name**:
  - Landmark
- **Problem**:
  - How to give easy access to different unrelated sub-systems? Web Applications usually contain many interesting entry-points; links to those points do not reflect conceptual relationships, and those links may yield a spaghetti-like structure

Pattern: Landmark

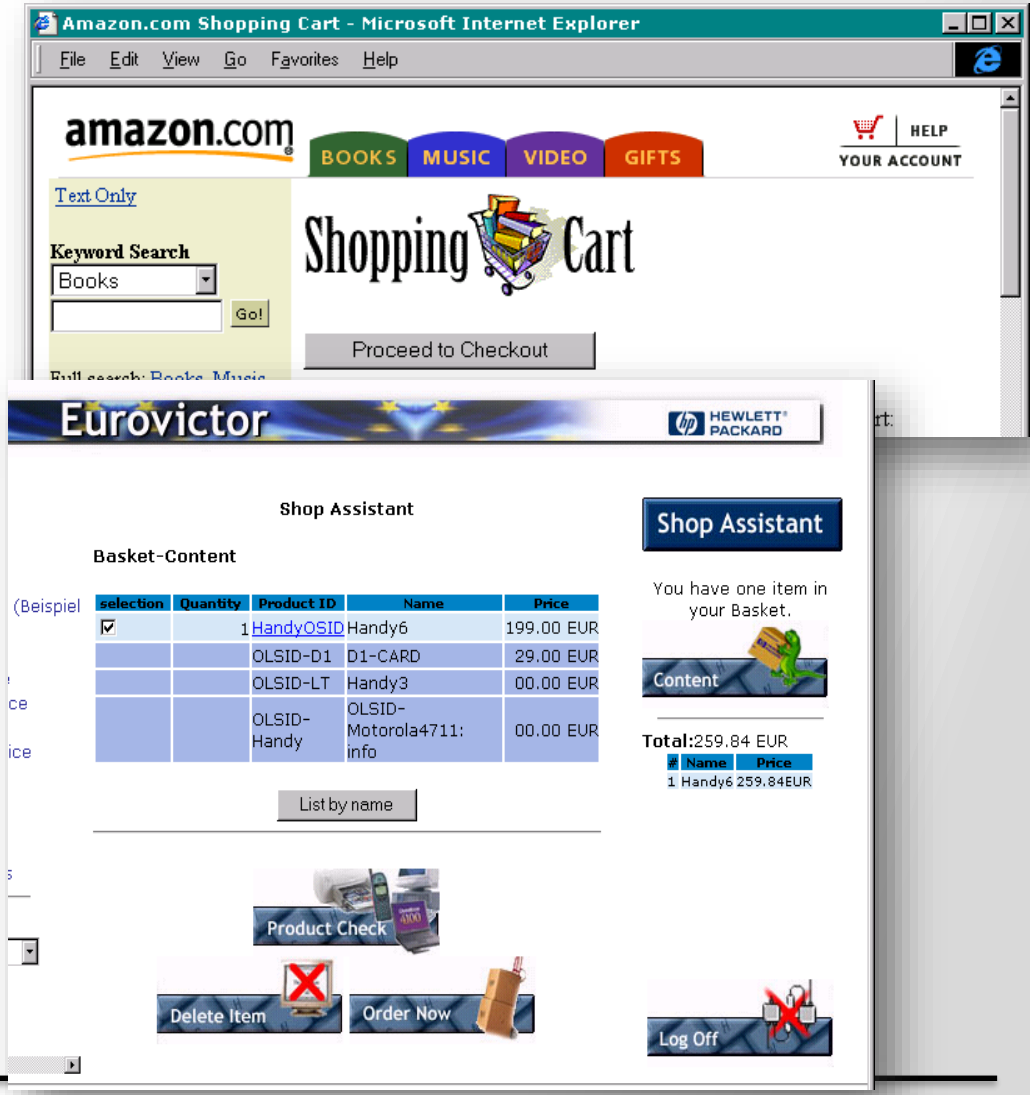
- **Solution**:
  - ▶ Define a set of Landmarks and make them accessible from every node in the application

Pattern: Shopping Process

- **Name**:
  - Shopping Process
  - Also known as: Shopping Basket / Cart
- **Problem**:
  - The Shopping Process must have well defined steps. This is necessary because we need to show the customer where she/he is in the process. The problem is now: How to describe the Shopping Process in a precise way? And how to present the customer a summary of her/his navigational decisions?

Pattern: Shopping Process

- **Solution**:
  - Provide a well-known metaphor to improve navigation
  - Provide a way for selecting items, managing items and check out (complete process)

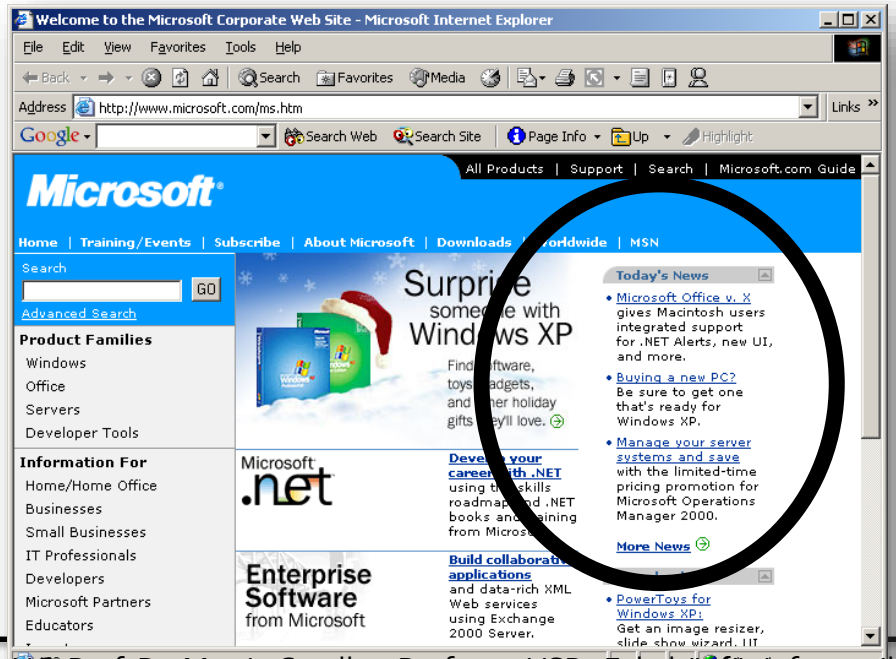


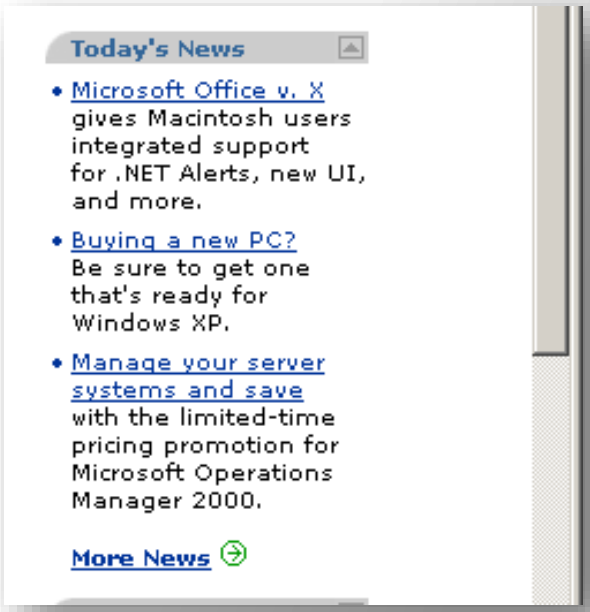
Pattern: News

- **Name**:
  - News
- **Problem**:
  - How to tell users of dynamic Web-Sites that there is new information? Most large Web-Sites are tree-structured. These Information Spaces tend to be large, and are hardly ever completely navigated. New information may not be found.

Pattern: News

- **Solution**:
  - Structure the Home Page in such a way that space is devoted to newest additions. Make those headlines anchors to the new pages
  - Additionally provide data as RSS





Monitor and Support Access

- What if users still do not find …
- (1) **Access by Navigation**
  - ▶ Use common concepts
  - ▶ Solution
    - Navigational Design Pattern
- (2) **Access by Direct Addressing**
  - ▶ Requires unique addresses
  - ▶ Solution
    - Use guessable (common) names / URL
    - Provide URL via marketing channels, e.g. TV

Monitor and Support Access

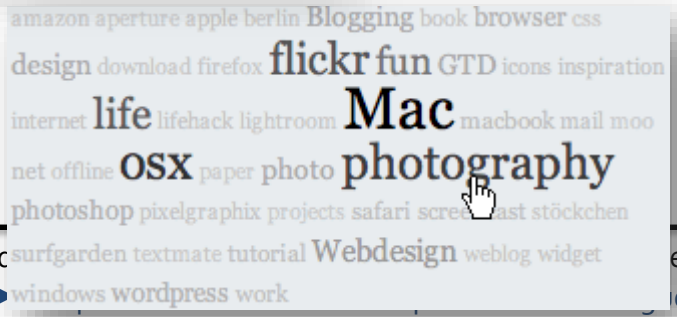
- (3) **Access by Search Request**
  - Provide search facilities
  - Solution
    - Reuse external search engine or implement internal
- (4) **Access by Browsing**
  - Provide support if class of information is unknown
  - Solution
    - Like "browsing" in a library: Register with catalog systems
    - Examples: lycos.com, web.de, ebay.de
    - Requires good classification, keywords, ontology expert

SECTION://4

- Navigation Design & SEA

Tag Clouds

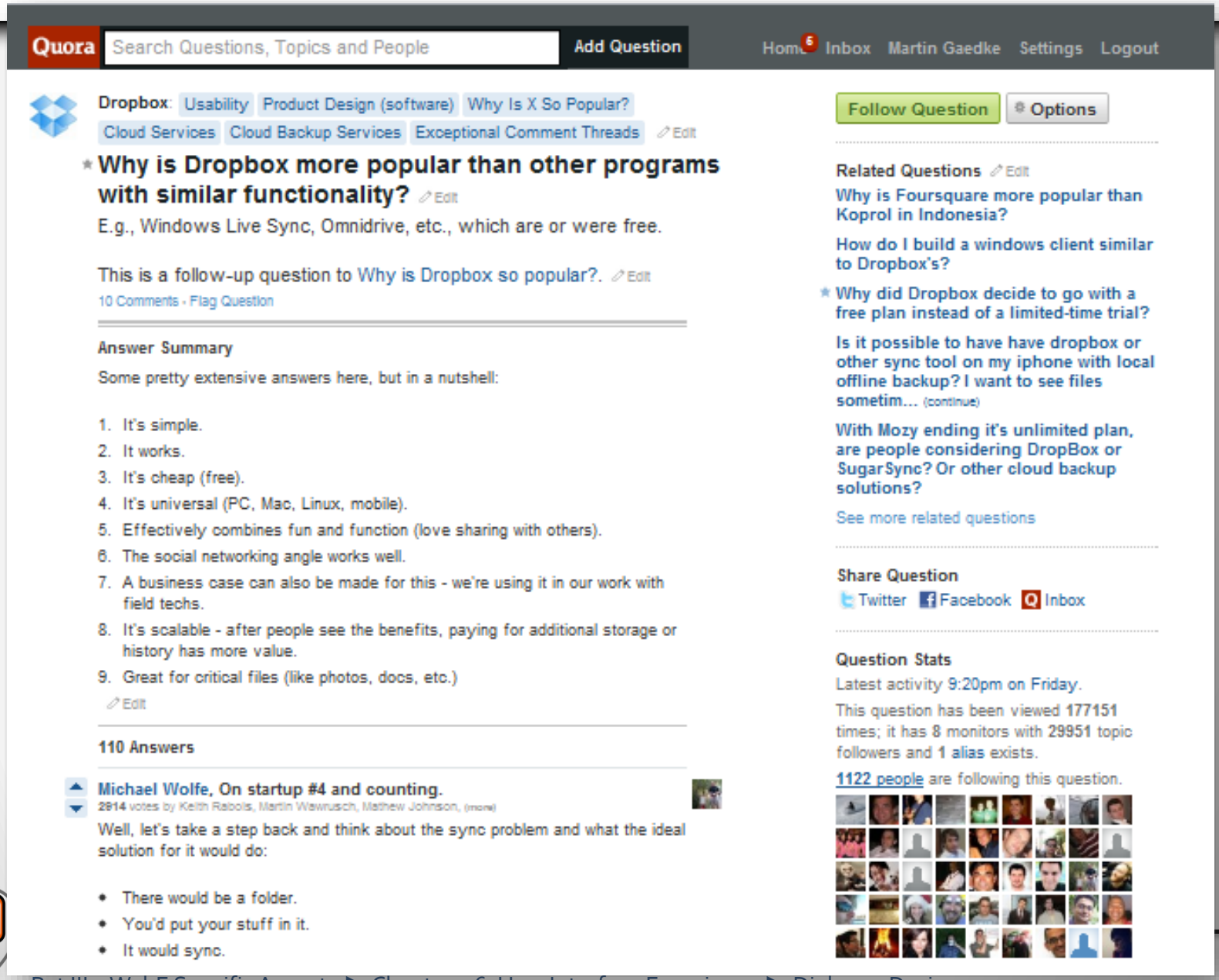
- Name: Tag Clouds
- Problem: Present large amounts of different content targets (links) in a structured way, e.g. items tagged by your social network
- Solution:



Friends / Others did this in your situation

- Navigation based on User-generated content
  - ▶ Amazon's Customers who bought this bought also that
  - ▶ QUORA
- Navigation within your network
  - ▶ Apply Set-based navigation
  - ▶ Set of friends, tags, groups, interests, similar users etc.

Example: Quora.com



SECTION://5

- Dialogue Design

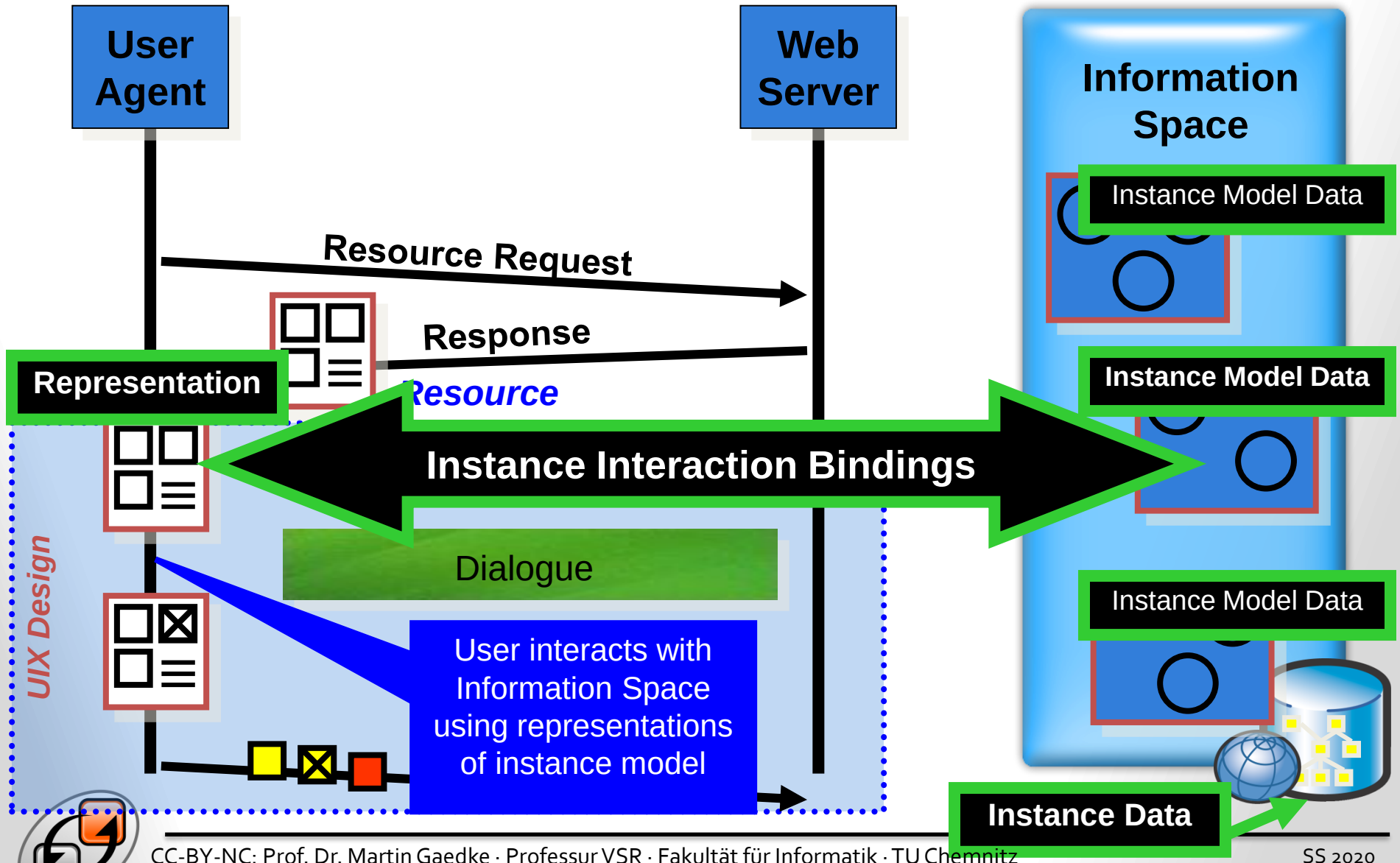
Introduction

- Purpose of Dialogue Design
  - Logical Design of how a user or system may interact with the Information Space of a Web Application
  - Physical Design representing the interaction logic
  - Abstraction of implementation model, e.g. Web, SaaS, other Cloud aspects, serverless etc.
- Challenges
  - Collect relevant information about interaction from Use Cases, e.g. access based on roles or devices
  - Search for specific interaction items not covered by conceptual design
  - Separate interaction and layout concerns
  - Provide a good experience when interacting with the information space, e.g. similar to desktop experience

Instance Model

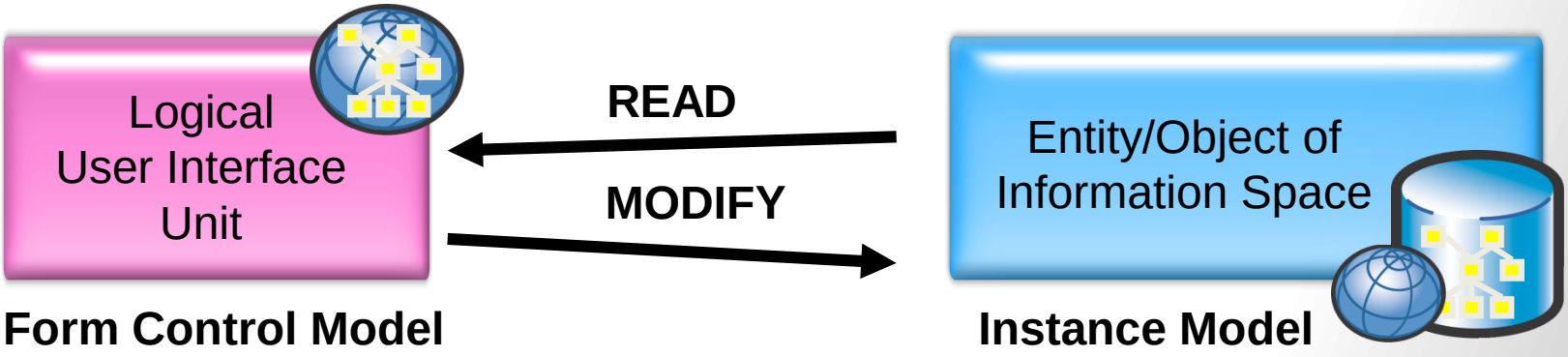
- **Instance Model** – Internal representation of values and state of all the instance data associated with a particular form
  - Usually same representation as used in Information Space – but may be different!
- Internal representation
  - Defines data type and form-specific constraints on a single piece of collected data

Interaction Model in the Web



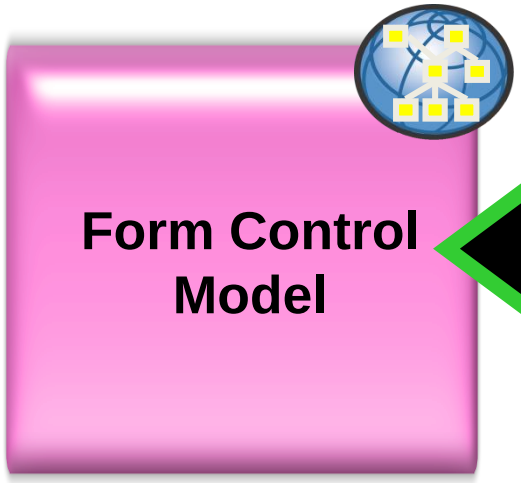
Logical Design

- Interactive user interface item
  - Many Representations – one Semantic
  - Specify security
  - Specify access patterns (Read vs. Write etc.)
  - Specify constraints (select one or many)



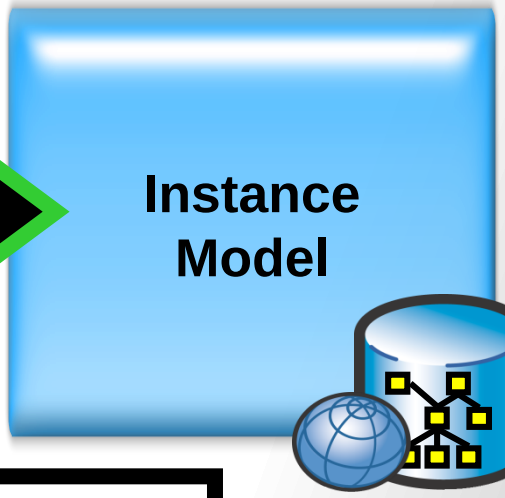
Example

Shift to Physical Design



Form Control Model

Instance Interaction Bindings



Instance Model

«defines»

Data Binding

- Bind control to instance item
- Define model item properties

Presentation Binding

- Bind *abstract control* to specific presentation
- Bind control to *presentation structure / tile – might incl. voice*

Processing Model

- Define Event handling and execut.
- Action – Event mapping (e.g. URL-codes, event-names)

Data Binding Model

- **Data Binding Model** – Set of data bindings. Describes the glue that connects the separate items of the Form Control Model with items of the Instance Model.
- **Data Binding** – Triple defining the connection between a *Form Control* and an *Instance Data Value* and *Item Properties*.

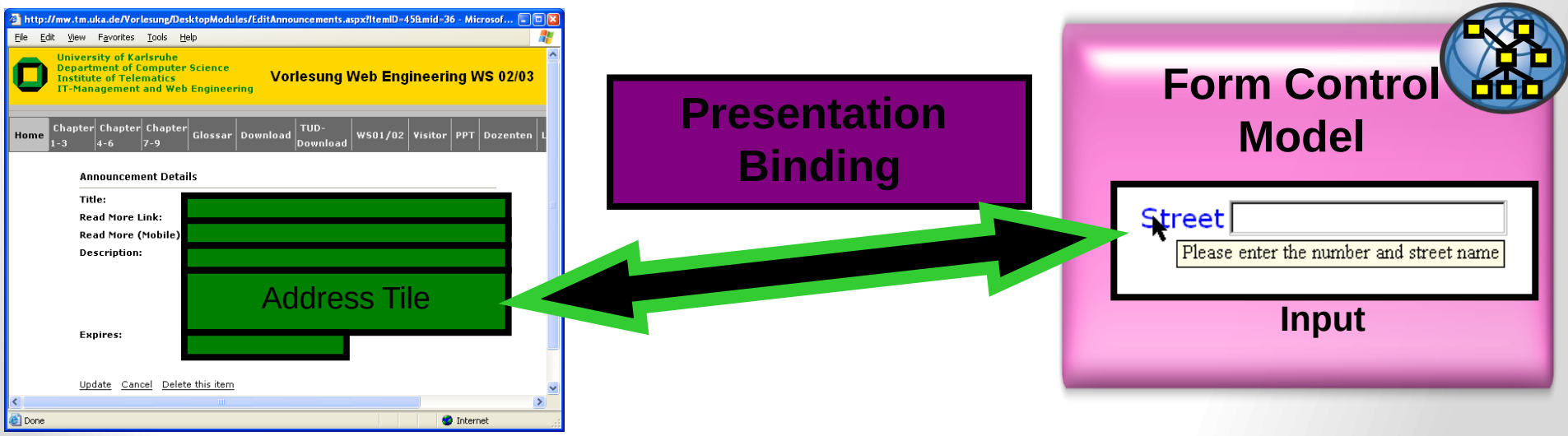
Item Properties

- Describing the logical relationship between Form Control and Instance, e.g.
  - **readOnly** – value is restricted from changing
  - **required** – value is required before the instance data is submitted
  - **relevant** – value is *currently* relevant to the rest of the Model
  - **maxOccurs** – applies only to repeat elements
  - **minOccurs** – applies only to repeat elements

  etc.

Presentation Binding Model

- Defines **Control Presentation**
  - ▶ E.g. SelectOne → Radio Buttons
  - ▶ Usually done in Presentation Design
- Defines "**Where**" in the layout structure, e.g. binding to tiles



Processing Model

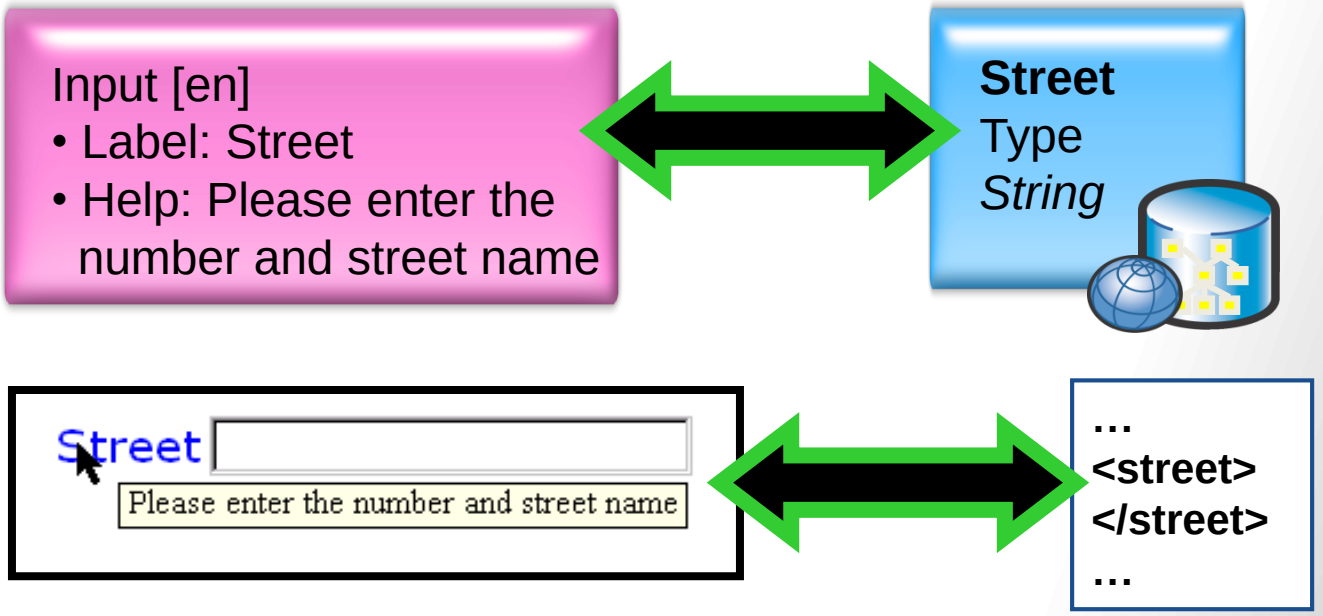
- **Processing Model** – Defines Event Processing at appropriate points within the User Interface
  - ▶ All form controls have a set of common *behaviors* that encourage consistent authoring and look and feel
  - ▶ Provides flexible means in conjunction with the Binding Mechanisms
  - ▶ E.g. send form data to server after pressing submit button *or* after every change
  - ▶ Ask yourself: How would voice be processed?

Processing Model: Examples

- HTML 2.0 Forms Model
  - Instance transfer: Submit
  - Events on instance data: Reset
- Scripting Approach (Advanced technologies)
  - Common approach today (but difficult to maintain, often hand-coded)
  - Use of scripting and standard form-controls
  - Instance transfer: Post initiated by script-code
  - Supporting technologies, like AJAX (Asynchronous JavaScript and XML), Backbase, Rico, DWR
- XForms Action Model
  - Classifies four behavior groups
  - Instance Transfer
  - Actions on Instance Data
  - Actions on Form Controls
  - Actions on Dependencies
- Trend: browser-based support of interface markup languages
  - Mozilla XUL (XML User Interface Language), Microsoft XAML (Extensible Application Markup Language), etc.

Supporting the User

- Goal: Optimize use of application
- Design Tasks
  - ▶ Minimize time user has to interact
  - ▶ Reduce elements prone to wrong inputs
  - ▶ Provide information to support use of input element
- Example:



Selecting One Flavor

- Applying - Instance Interaction Binding
  - Example: Selecting one or only one?
  - Example: What if the User Agent does not support select boxes?

Implies Select Only One

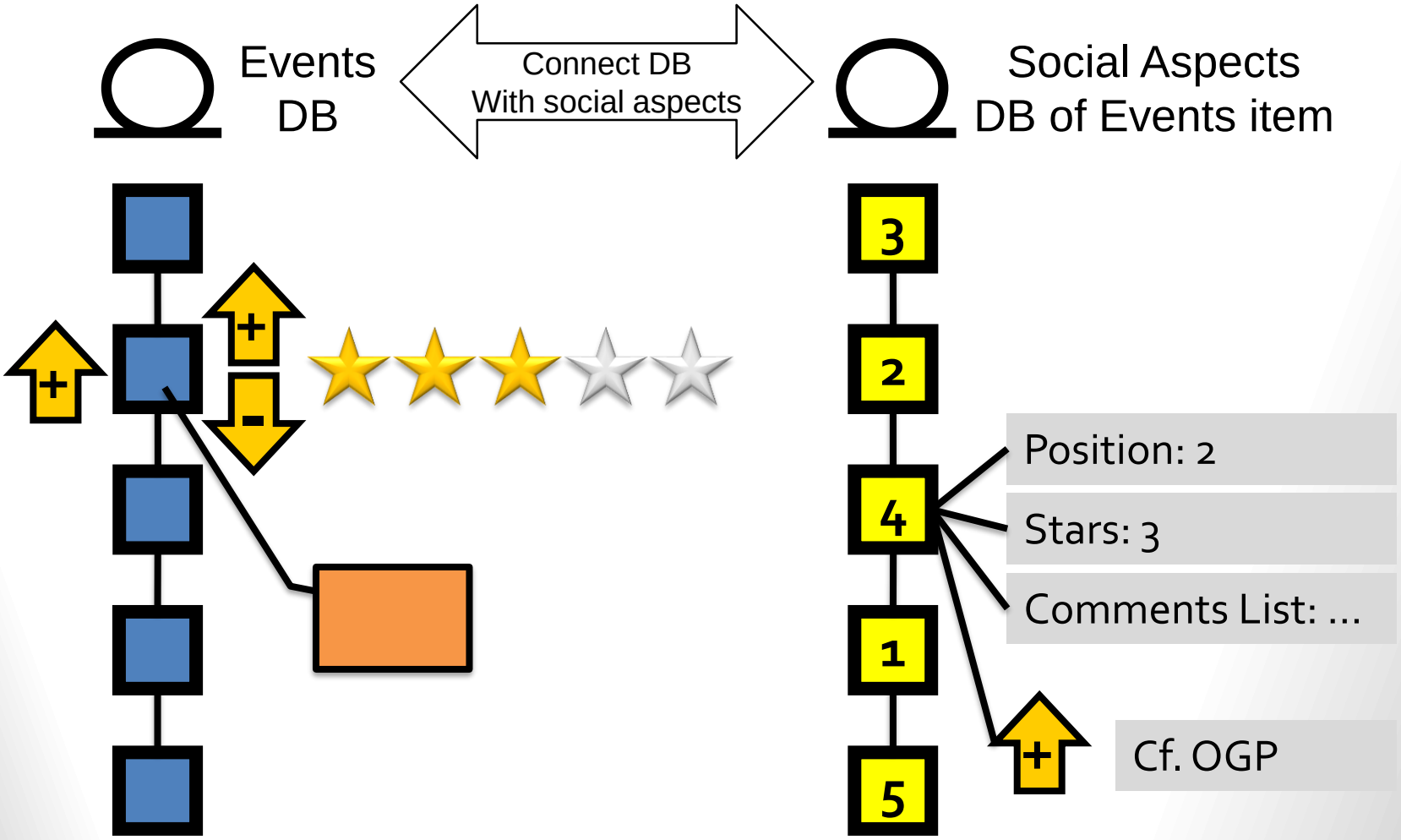
SECTION://6

- Dialogue Design & SEA

Dialogues Patterns of the Social Web

- Typical Dialogue Elements
  - Discussing, e.g. chat, commenting
  - Categorizing, e.g. annotating, keywords, tagging (free, taxonomy)
  - Rating, e.g. rating, voting, fav, 5-stars, thumbs-up and down
- Cf. Custódio: Online Community Design Patterns. http://www.slideshare.net/pecus/online-communities-design-patterns-255635

Applying Social Object Aspects



CHAPTER://17

- Web System

Goal

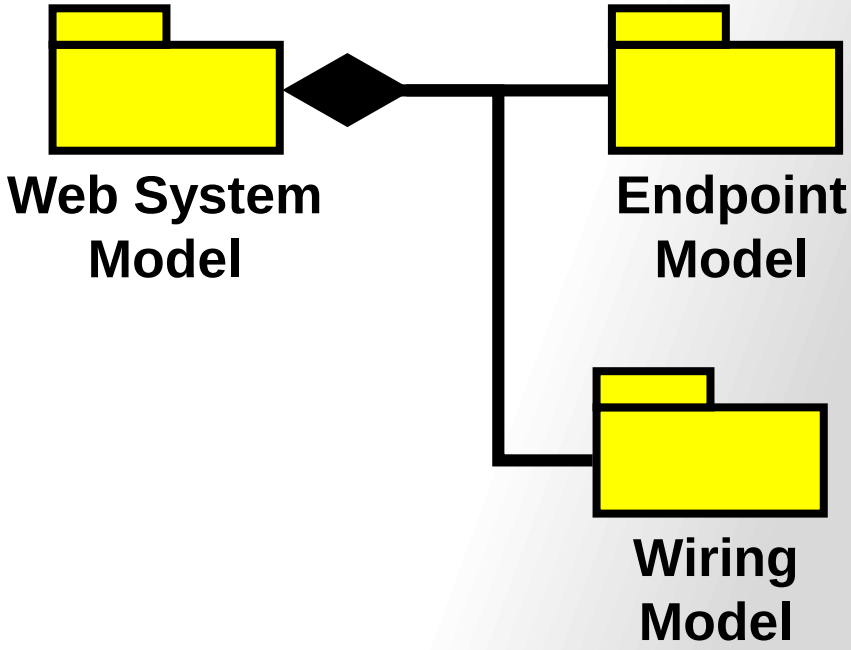
- Aspects of a Distributed Web System
  - Using and understanding the "Environment" to host business processes and applications
  - Define/Reuse business processes and their elements
  - Define *global* (including multiple organizations) wiring of the overall elements
  - Define communication and connectivity aspects, like transport protocols, security, topologies, behavior, constraints and external relationships between all elements
- Challenges
  - Complexity of ultra large-scale distributed systems
  - Maintenance and evolution of all "elements" of the application, its business processes and environment

SECTION://1

- Introduction

Web System

- **Web System Model** – A model that defines the overall product logic as a set of **Endpoints** and their **Wiring** (connections). It abstracts the implementation, e.g. SaaS / Public Cloud, Web on premise etc.
  - **Endpoint Model**
    - Endpoint interface and message definitions
    - Possible Service Level Access Requirements
  - **Wiring Model**
    - Defines wiring between endpoints
    - Depends on communication aspects



Introduction to Endpoints

- **Endpoint** (or service access point) – A specific location for accessing a functional unit operating on messages
  - Accessing an Endpoint is defined by specific protocols and data formats
  - Endpoints are provided by a Service, e.g. a component, XML Web Service, Web/Email server
- **Endpoint Definition** – A definition describing the necessary information for accessing or providing a dedicated type of an Endpoint
  - Abstract Endpoint Definition (logical aspect)
  - Concrete Endpoint Definition (physical aspect)

Wiring Model

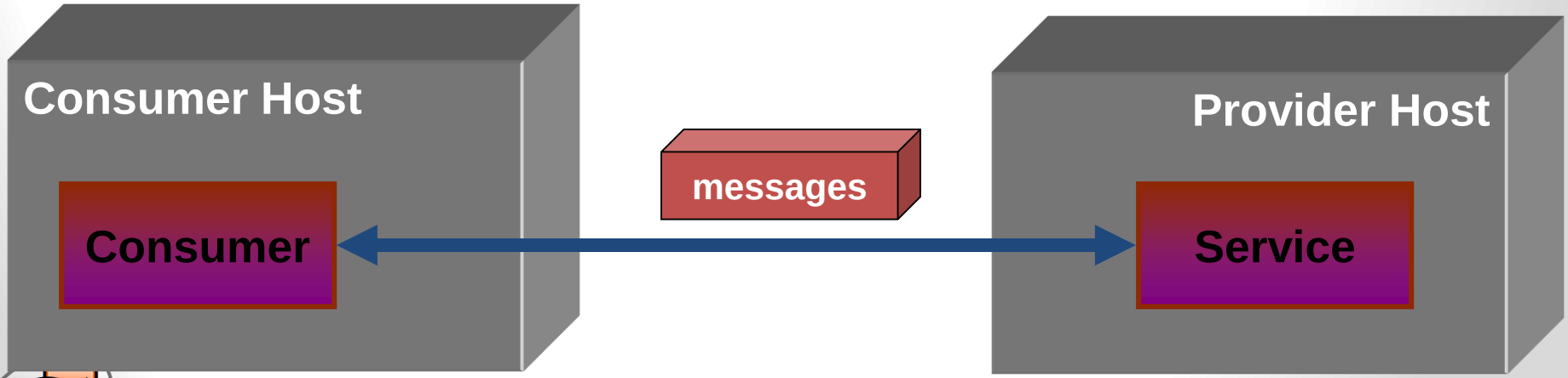
- **Wiring Model (Glue Model, Endpoint Instance Configuration)** – A Model that describes *which* Endpoints are connected and *how* they work together by using **Wire Protocols**.
  - This is more than just define bindings
  - Gluing functionality, requires solid observation and definition of factors influencing behavior
  - Allows for highly dynamic wiring using brokering approaches, e.g. Peer-to-Peer (P2P) networks or broker-proxies

Wire Protocol

- **Wire Protocol** – Defines the format of messages and how specific messages are exchanged between endpoints (respectively service consumer and endpoint).
- This Definition supports distributed systems, i.e. Wire Protocol allows for
  - Message exchange between physical nodes
  - Message patterns using multiple communication protocols
- Examples
  - Wire Protocol put into concrete form by Component Run-Time System DCOM/COM+: Object-RPC, Java: RMI, Internet: Application Protocols / Web Service Bindings etc.

Wiring Distributed Endpoints

- **Distributed Endpoints** – Endpoints are deployed to remote Hosts (Cloud, SaaS)
- **Deployment** – describes distribution of processes and endpoints to processing unit (Hosting node, Host)



Wiring Models Examples

- Wiring Models (Orchestration Models)
  - Business Process Languages
  - XLANG – scheduling language for orchestration of components
  - Web Service Modeling Ontology (WSMO) and related approaches e.g. WSMX (Web Service Modeling eXecution environment)
- Other approaches that might be applied / useful
  - **State-Chart, Petri-Nets, etc.**
    - Petri-Nets are well understood and allow for programmatic optimization
  - **Business-driven Models**
    - ebXML – eBusiness XML by UN/CEFACT and OASIS
    - Cf. http://www.ebxml.org
  - **Architecture Description Models**
    - Support for transactional processes – supports any action, which allows to reverse the step (Undo-Semantic)
  - **Highly Dynamic Approaches**
    - Require modeling Contract-Negotiation
    - E.g. using Broker or Peer to Peer Networks (P2P network)

Factors Influencing Wiring

- Web System Aspects
  - Contracts
  - Dynamic Processes
  - Time-Behavior
  - Payload
  - Security
  - Scalability
  - Optimization
  - Specific Factors (rules) may exist, e.g. law related aspects defined in business rules
  - etc.

  - CF. Lecture **Software Service Engineering**

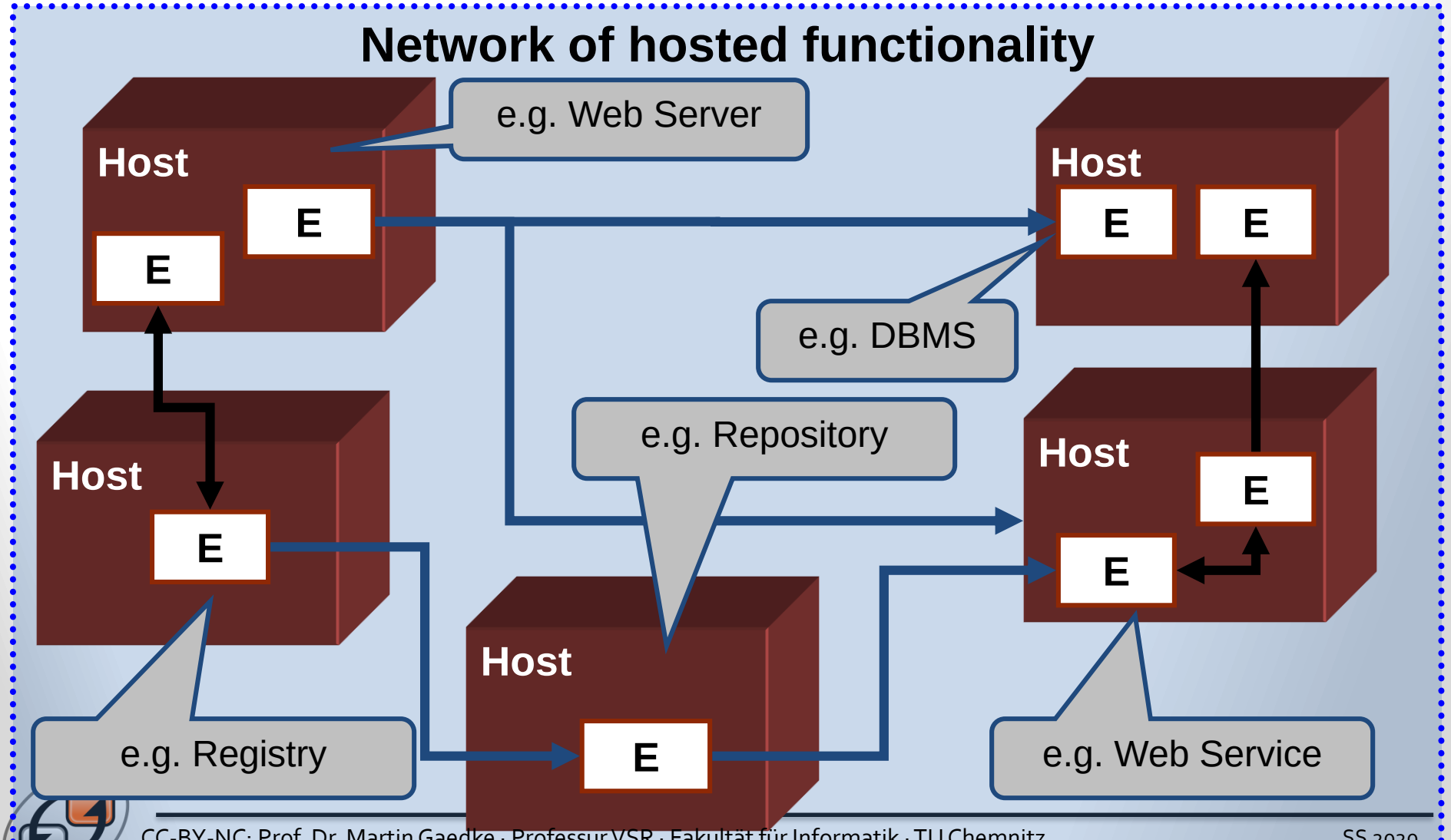
SECTION://2

- Endpoint Design of Processes

Introduction

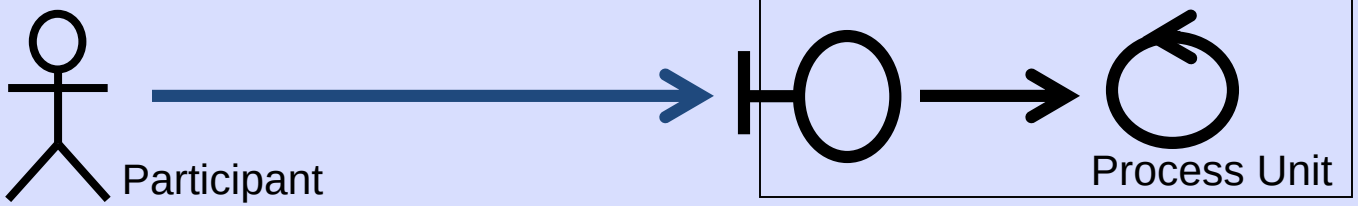
- Design of the Process Layer
  - Logical Design derived from Business Process Model
  - Supports translating into physical model, which might be executable
- Challenges
  - Separation of concerns (functional units and flow between them)
  - Distributed systems hosting the business process
- Advantages
  - Eases considering complex **business rules**
  - E.g. customer related aspects, assurances, such as transactions, reliability, durability, by focusing on **wiring aspects** and **participant behavior**

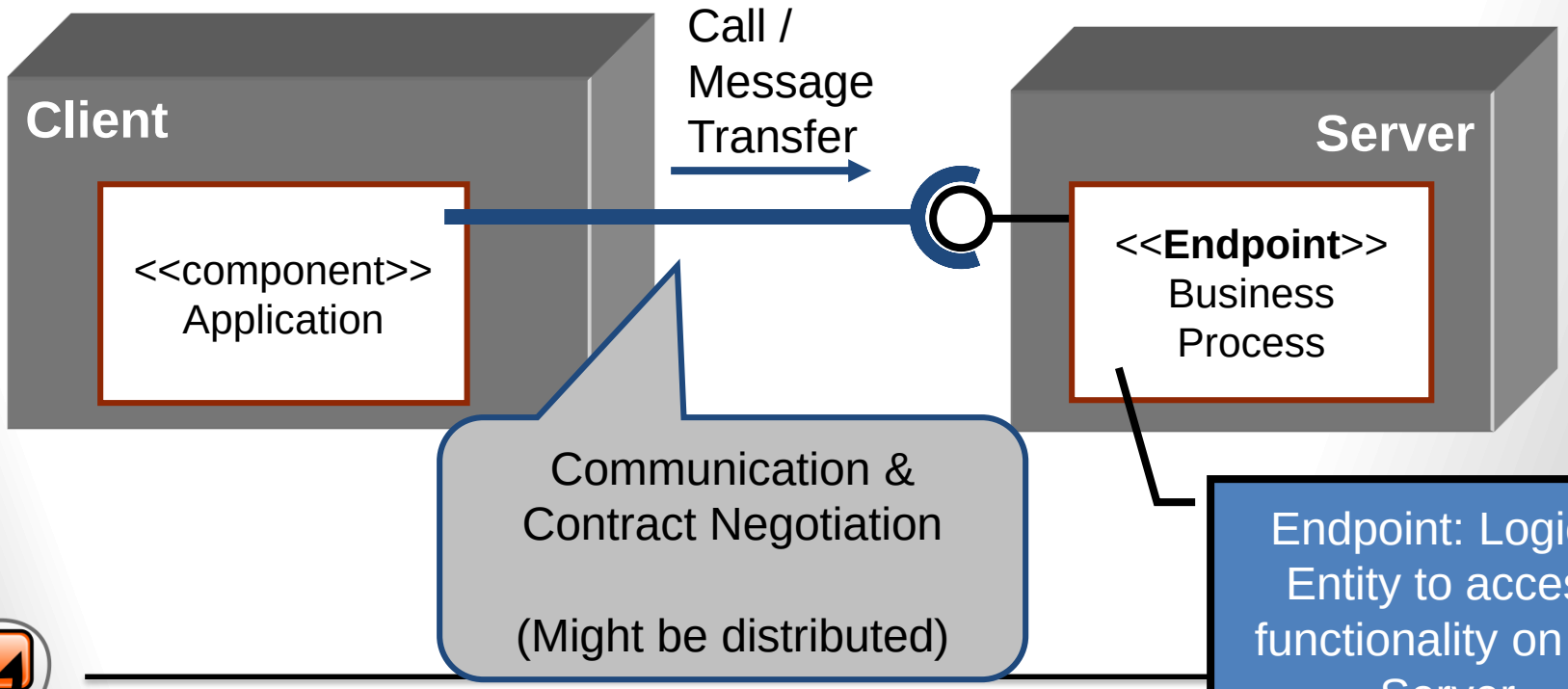
Distributed Business Process



Business Process Deployed

- **Business Process:**

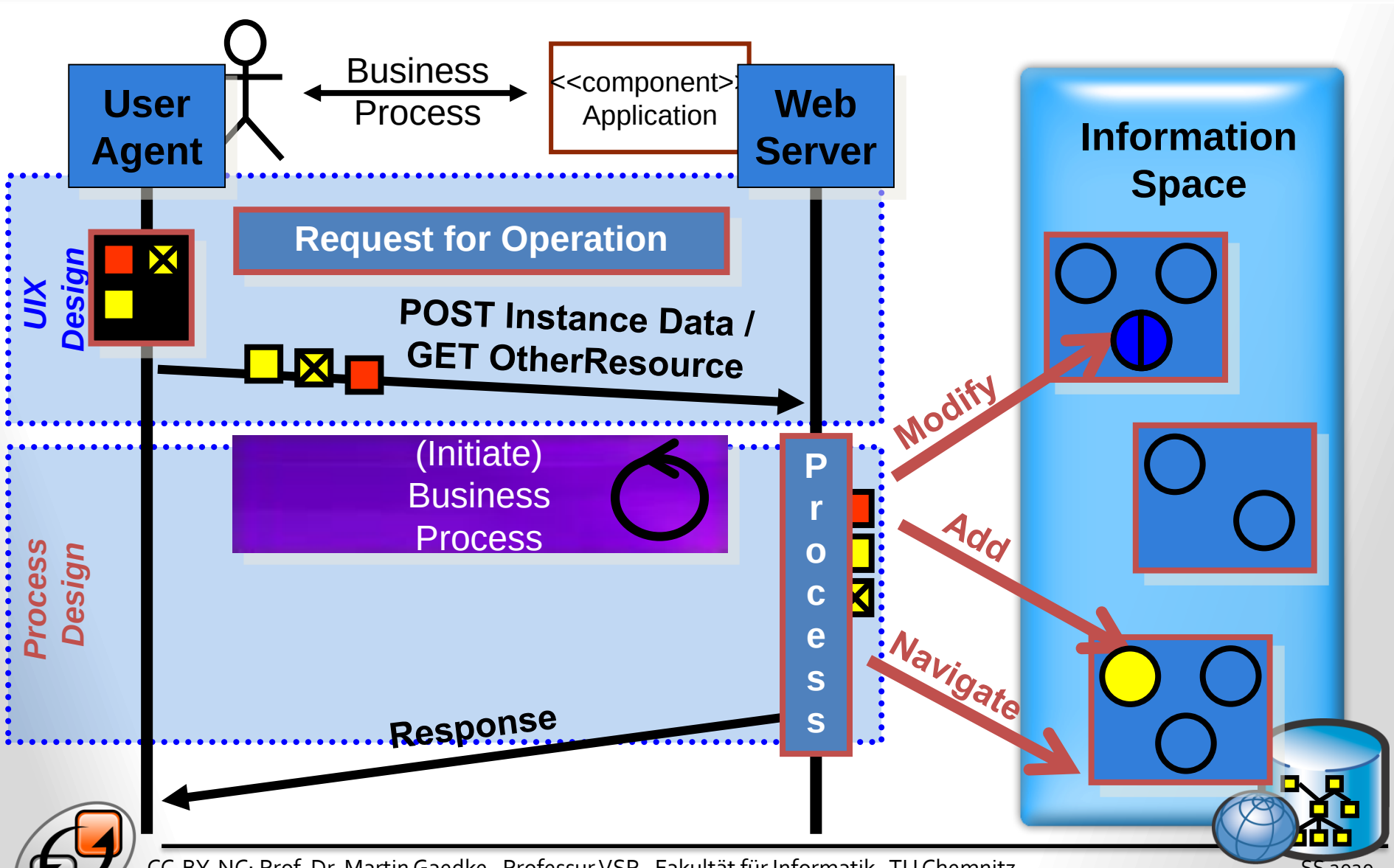




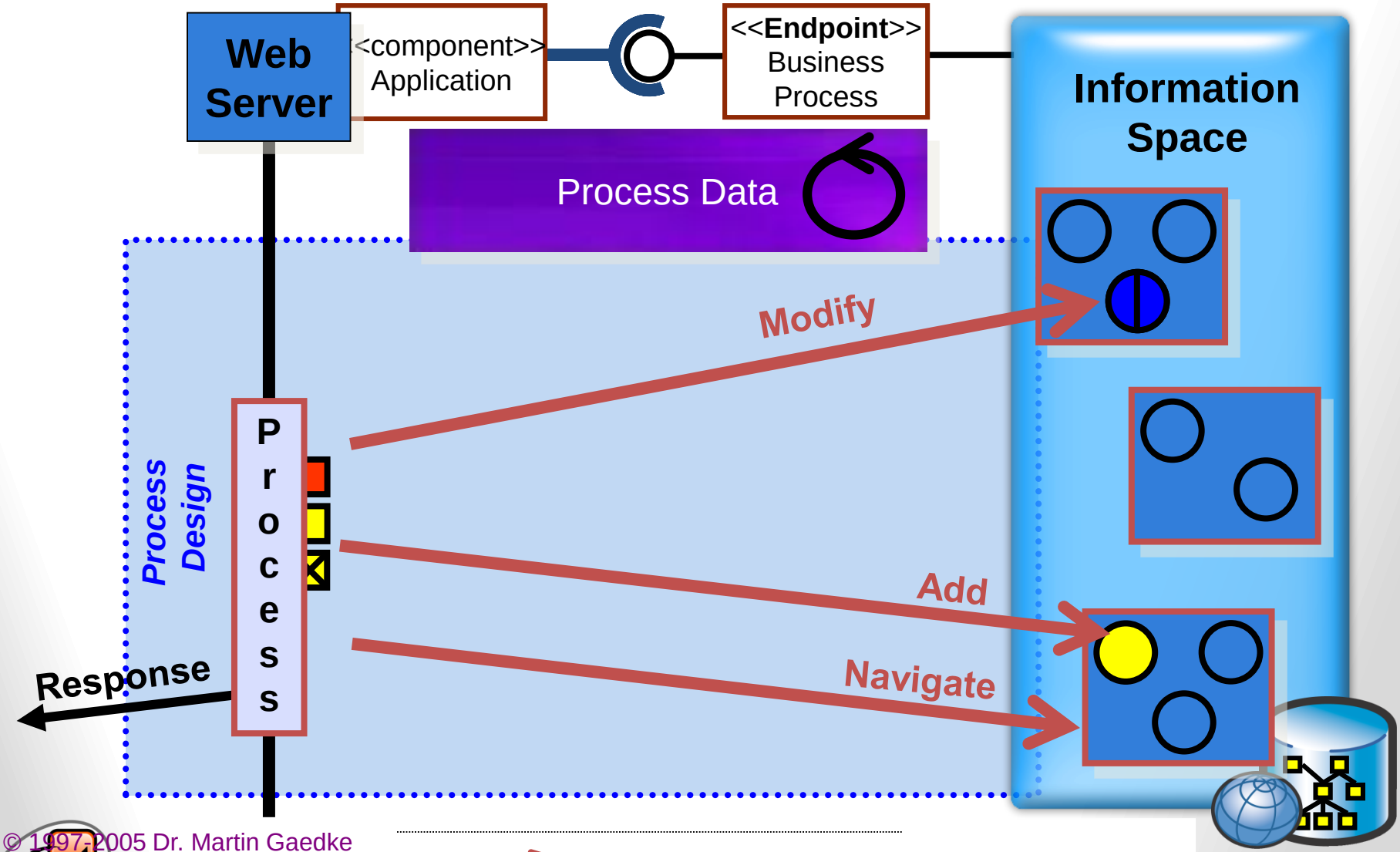
Process Unit

- **Process Unit (Process Component)**
  - Functional representation of a participant of the business process
  - **A unit accessible as Endpoint within the process layer** (and usually provided by several endpoints of the service layer)
- Modeling focuses on **wiring of process units (respectively their endpoints)**
  - Relationships between endpoints are expressed in terms of agreed upon communication patterns
  - Behavior of endpoints is described in abstract terms (Input-/Output-Contracts)

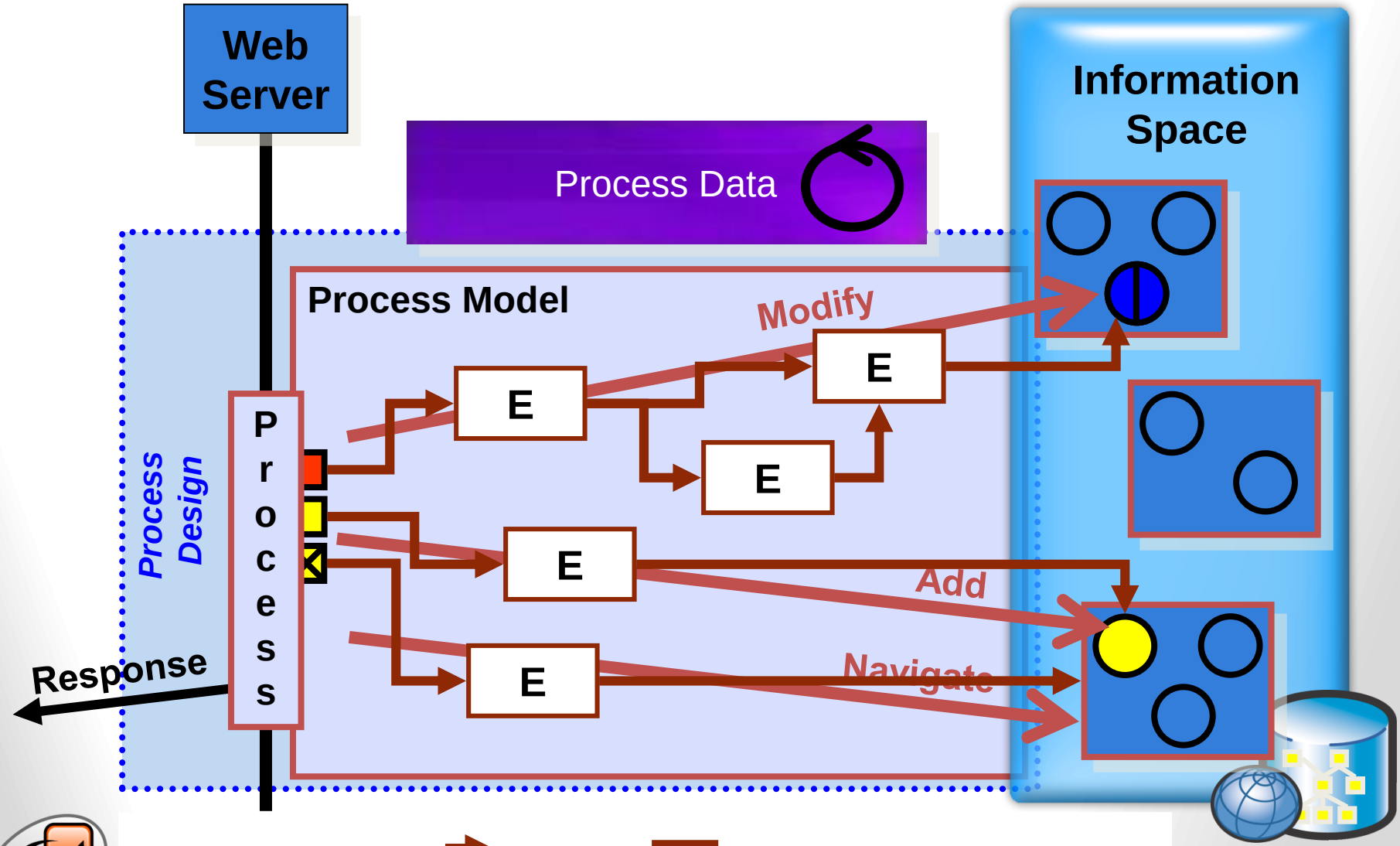
Process and the Web



Process Model and the Web



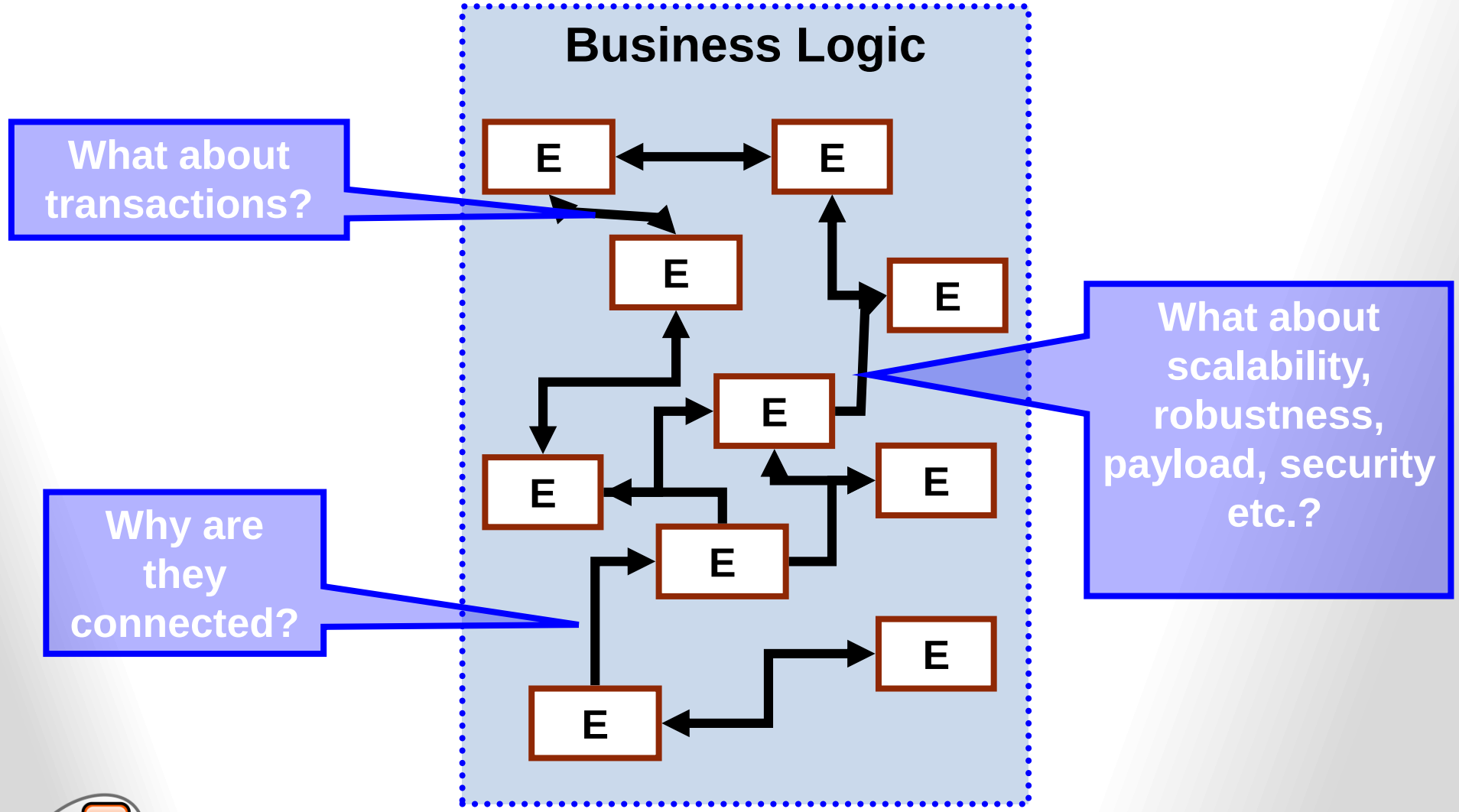
Process Model in Detail



SECTION://2

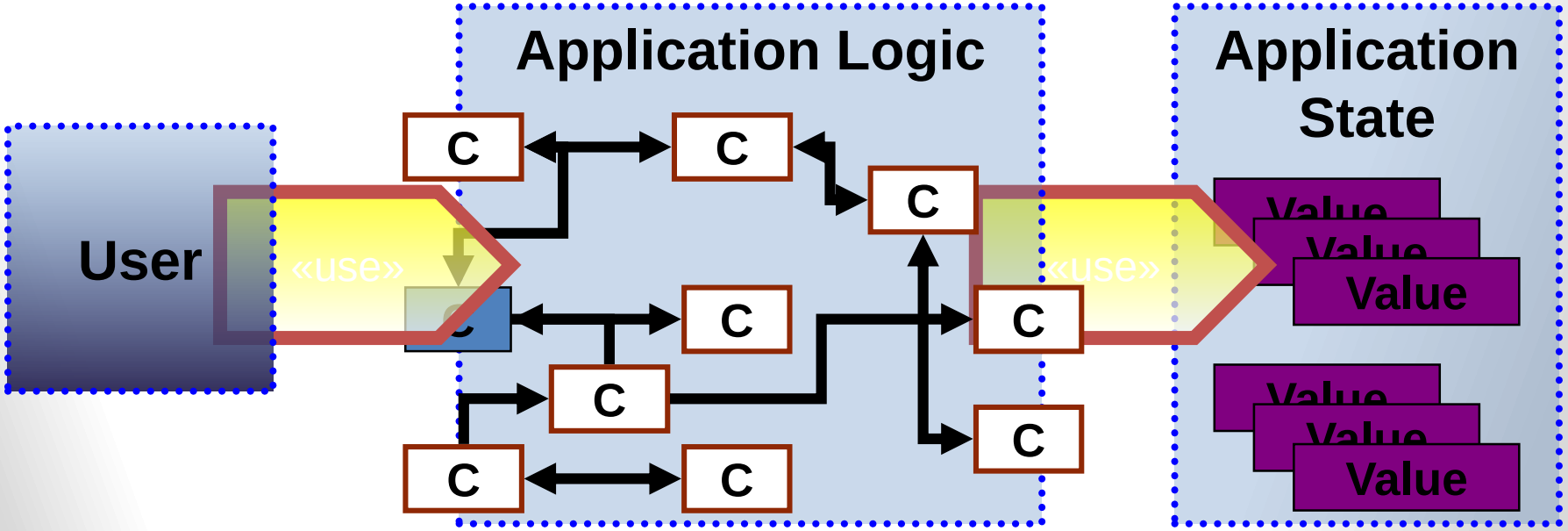
- Endpoints in context

Introduction: Distribution Aspects



Why State Management?

- Core Communication Model of the Web (HTTP) is stateless
- Application requires state when a user traverses the information space of a Web Application



Session Design

- **Session** – Defines a context in which a user communicates with a Web Application in a defined time period
  - One Session per user
  - Assigns application state to multiple requests from one user
- Design Decision / Rules of thumb
  - Use a database to persist state
  - UUID to identify a session/user
- Physical Design: Session identifier exchange
  - Cookie, hidden variable, or encoded into the URL

Dynamic & Transactional Links

- **Dynamic Links** – Describe relationship between endpoints in the Wiring Model that exist based on state and application logic
- **Transactional Links** – Describe relationship between endpoints in the Wiring Model that exist only as a whole

Transactions And The Web

- In some Web Application scenarios you have a series of correlated operations corresponding to consecutive HTTP requests. You need to ensure that if one single operation fails, all related operations fail.
- Example:
  - Booking a flight and a hotel at the destination. The hotel is not necessary if no flight is available for the stay. Transaction processing is the technology that enables you to control the process as a **whole**.

Transactions

- **Transaction** – A unit of work that should either succeed or fail as a whole. A series of operations that behave corresponding to the ACID rules.
  - Series: BEGIN_TRANSACTION, Op1, …, OpN, COMMIT_TRANSACTION
  - ACID Rules define Atomicity, Consistency, Isolation, and Durability
- Characteristics regarding Web Applications
  - Long Running
  - Nested

Atomicity And Consistency

- **A**tomicity
  - Transaction executes *exactly once* and is atomic
  - All the work is done or none of it
- **C**onsistency
  - Transaction preserves the *consistency* of data
  - Transforming one consistent state results in another consistent state of data

Isolation And Durability

- **I**solation
  - Transaction is a unit of *isolation*
  - Concurrent transactions behave as though each was the only transaction running in the System
- **D**urability
  - Transaction is a unit of *recovery*
  - If a transaction commits, the system guarantees that its updates will persist, immediately after the commit.

Effects On Web Services

- Transactional Links requirements
  - Declaration of Web Services to be executed within a transaction
  - Specify transaction properties on every method of a Web Service
- Practical Approach – Transaction Properties describing Context
  - Supported, NotSupported, Required, RequiresNew
  - ITX (identifier) for Internet Transaction
  - Other Approaches are possible

Context Description Approach

- Supported
  - Transaction exists implies the method will run in its context
  - No transaction implies the method will not run within a transaction
- Not Supported
  - The method will not run within a transaction
- Required
  - Transaction exists implies the method will run in its context
  - No transaction implies a new one will be started
- Requires New
  - A new transaction will be started on each call.

Transaction Design

- Some Rules of Thumb
- Use a supporting System
  - ▶ e.g. TP-Manager, Database
- Design Components with transactions in mind
  - ▶ Transactions are powerful but imply overhead
  - ▶ Not every component 'requires' a transaction
- Be aware of the Transactional Semantics of the underlying system or database
  - ▶ Long-lived Locks in the database will kill performance
  - ▶ Look for blocking and deadlocks when testing

Portal Accessing Systems

- **Portal** – Web Application that provides uniform access to different content source
  - ▶ Screen Scrape, WSRP, Uberportal etc.

Some Observations

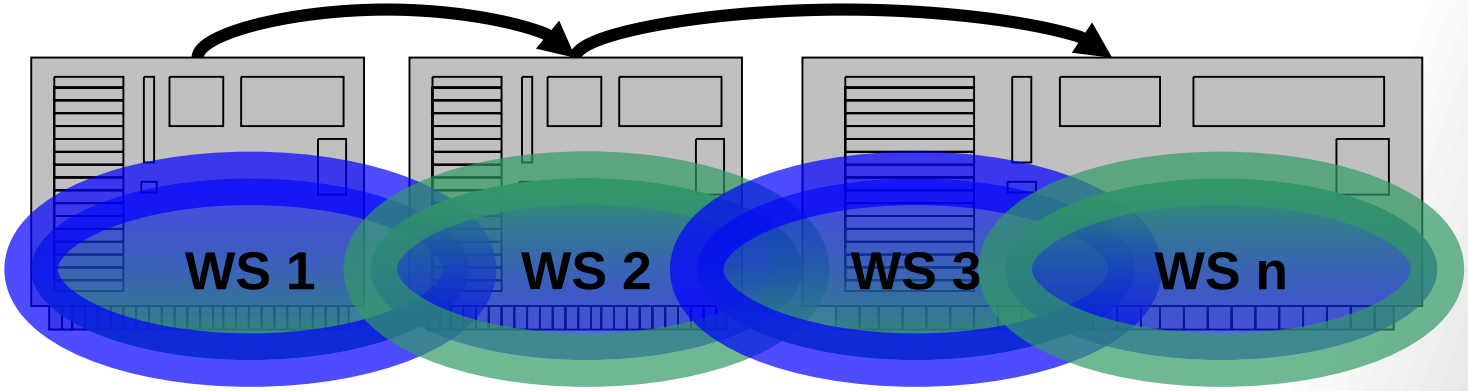
- Communicating Processes may be evolving
  - Hosted on physical machines
  - Mediated by both physical and logical channels
  - Physical hosts and media are subject to failure by various hazards
  - Software and hardware configurations may change during the lifetimes of Business Processes
  - **Business Process Communication must be robust against changes intended (Version Update) or not (Failure)**

Some Observations

- Communicating processes may be mobile
  - Long-lived Business Processes will undergo multiple resource (re-)allocations
  - A process may begin its activities on one resource and continue on another
  - Mobility increases the challenge to Business Process robustness

Simple Web Service Chain

- Web Service WS 1 provides functionality using WS 2, WS 2 provides…
  - Like a chain: The weakest element influences the overall behavior
- **Hops** - Represents the number of network nodes involved from the source WS to the destination WS. Example shows 2 Hops, 4 Web Services



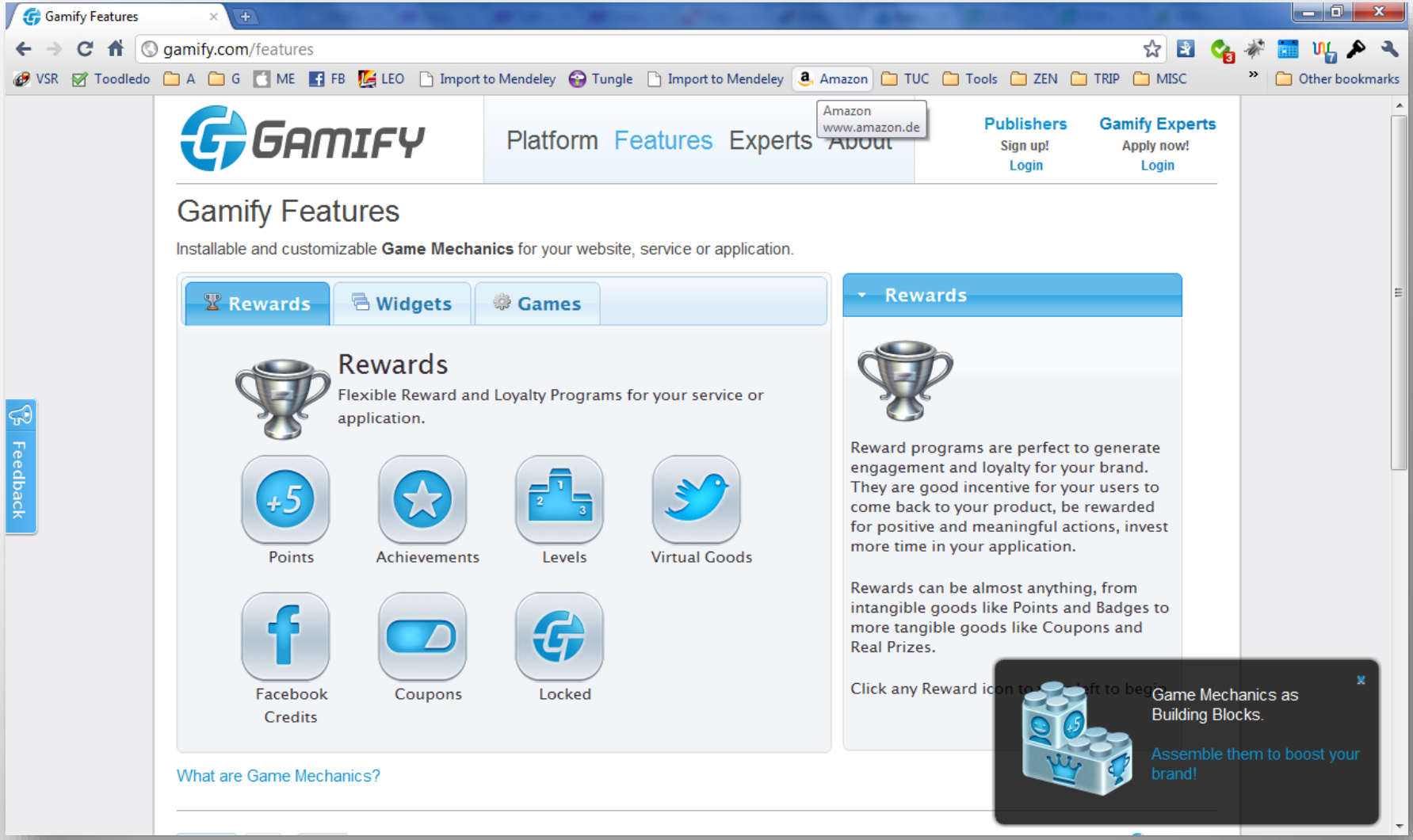
SECTION://3

- Web System & SEA

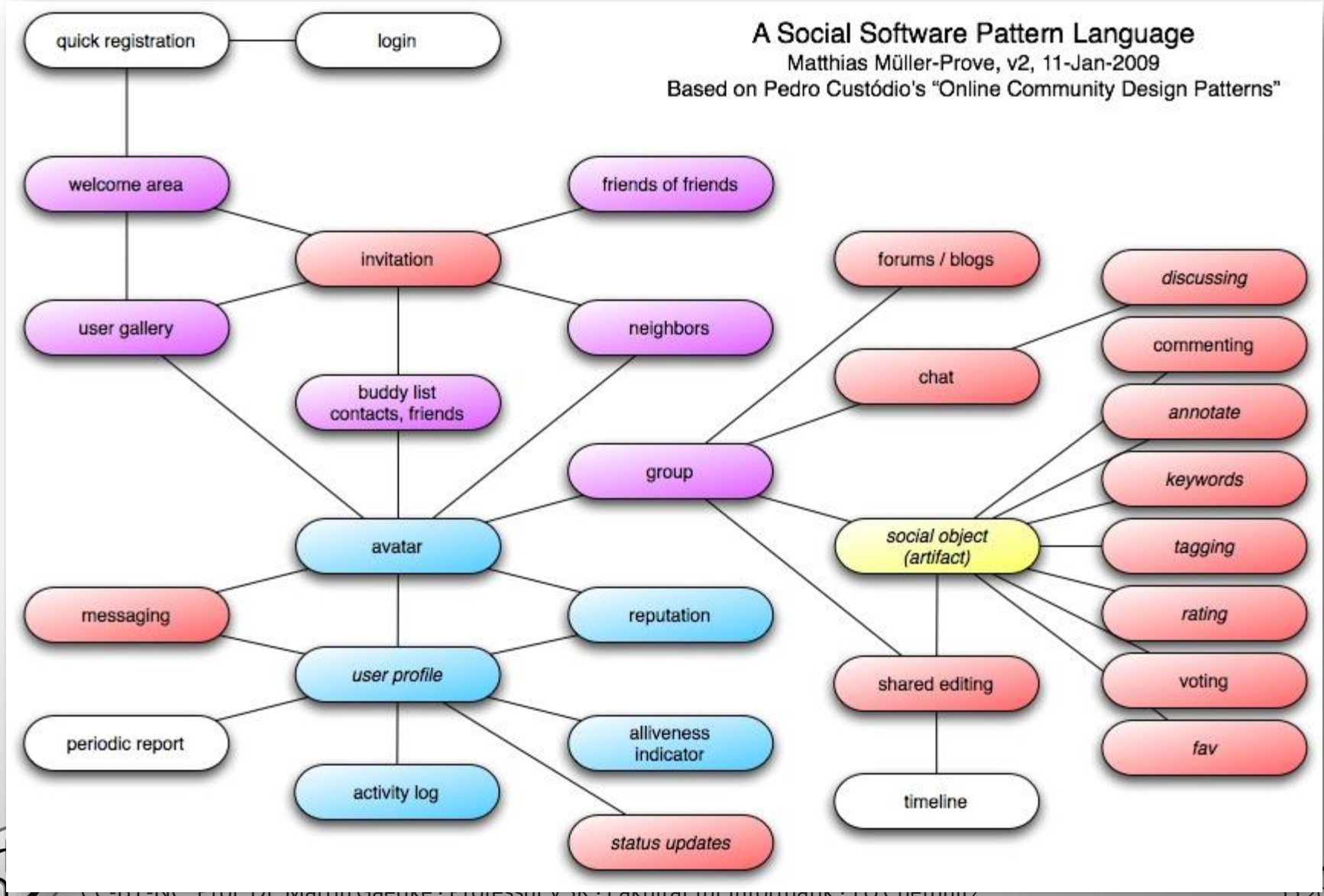
Introduction

- The Social Web
  - Social Web is a set of relationships that link together people over the Web
- Again: Requirement and Opportunity
  - A major **requirement** is a better support (may be also better understanding) of identity
  - Revolutionary **opportunity**: Creation of a decentralized and federated Social Web (**)
- For further information check W3C's groups on related topics, e.g.
  - http://www.w3.org/2005/Incubator/webid/

Example Web Services for Gamification



Discussion – What else do we need?



SECTION://4

- Web System Federations & Modeling

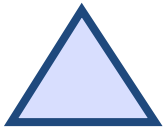
Federated Systems

- Goal: Bring business processes together
  - Globalize the Component-based View
  - Extend processes with external (potentially unknown) partners
- Idea: Federating Web Applications (respectively their Logical Units)
  - Take identity and access management (IAM) into account
  - Define protocols to support inter-organizational information exchange in a standardized way

WAM (Federation Model)

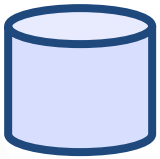
- WebComposition Architecture Model
  - Introduced in 2005 by Gaedke and Meinecke
  - Consists of several models
  - Applies UML-like notation in combination with OCL
- Six core entities
  - Can be connected by protocols
  - Nested within zones/realms
  - Each Connection is *labeled* with a shortcut, which is used for detailed description (cf. OCL) in addition to the graphical notation
  - Labels and their details are stored in a dedicated database, i.e. labels once defined can be reused in later projects

WAM Core Entities (1)



- Service
  - Represents the system's distributed (atomic or composite) components
  - E.g. SOAP Web service

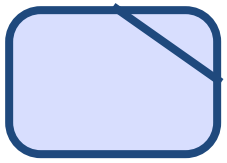
- Application
  - Allows users to interact with the overall system
  - E.g. Web applications or portals



- Data Provider
  - Distinguish between the services and the underlying systems that serve as the actual data sources
  - Connected to service or application with undirected line

- Process Unit
  - Connected systems that perform functionality beyond data management
  - E.g. software that performs computations or triggers events

WAM Core Entities (2)



Name Label

- Security Realm
  - Envelopes applications, services, data provider and process untis as organizational zones of control – as such functions as identity and access management context
  - E.g. defines set of roles and permissions
  - Realms might be nested
  - Implemented e.g. as a Security Token Service
- Identity provider
  - Store for accounts/identities (of known users as well as applications)
  - Allow to authenticate the members of the realm – issues security tokens
  - E.g. through login forms or Web service interfaces
- Name Label
  - These label represent a naming context for each entity
  - Naming-Labels might be used as shortcut for a detailed description of these entities

WAM Core Relations


[i-label]

- Invocation (Communication Profile)
  - Potential accesses on services and applications
  - Labels indicate the designated communication protocols (label acts as a shortcut for a detailed description of the communication relationship)
  - E.g. SOAP via HTTP, SOAP via SMTP, WS-I compliance etc.

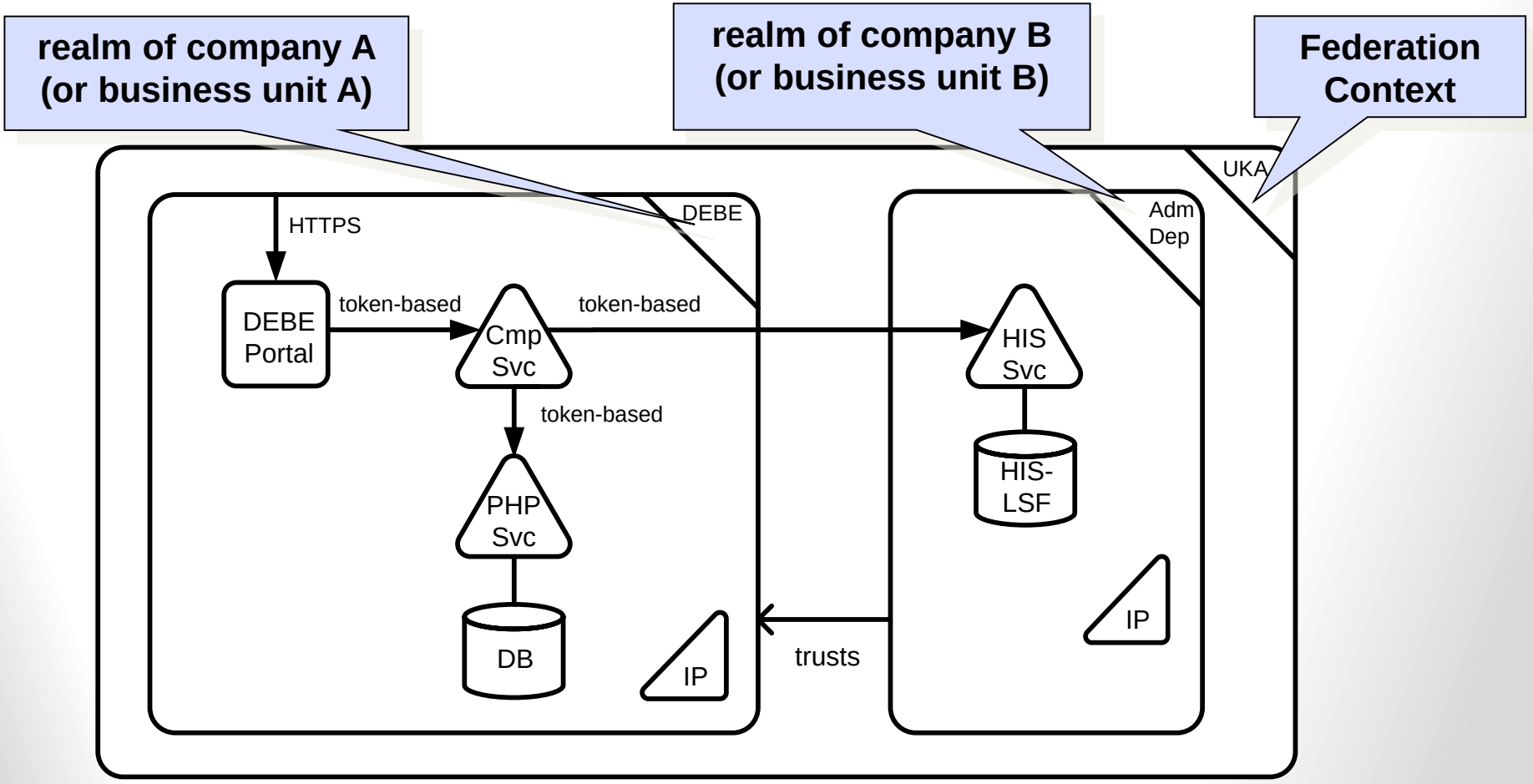

[t-label]

- Trust (Trust Profile)
  - Trust-label separate realms that form a federation
  - STS of the trusting realm accepts the tokens originating from the trusted realm (label acts as a shortcut for a detailed description of the trust relationship)
  - Identities of the foreign requestors can be mapped to tokens that are locally valid – these relationships are defined for the trusts labels

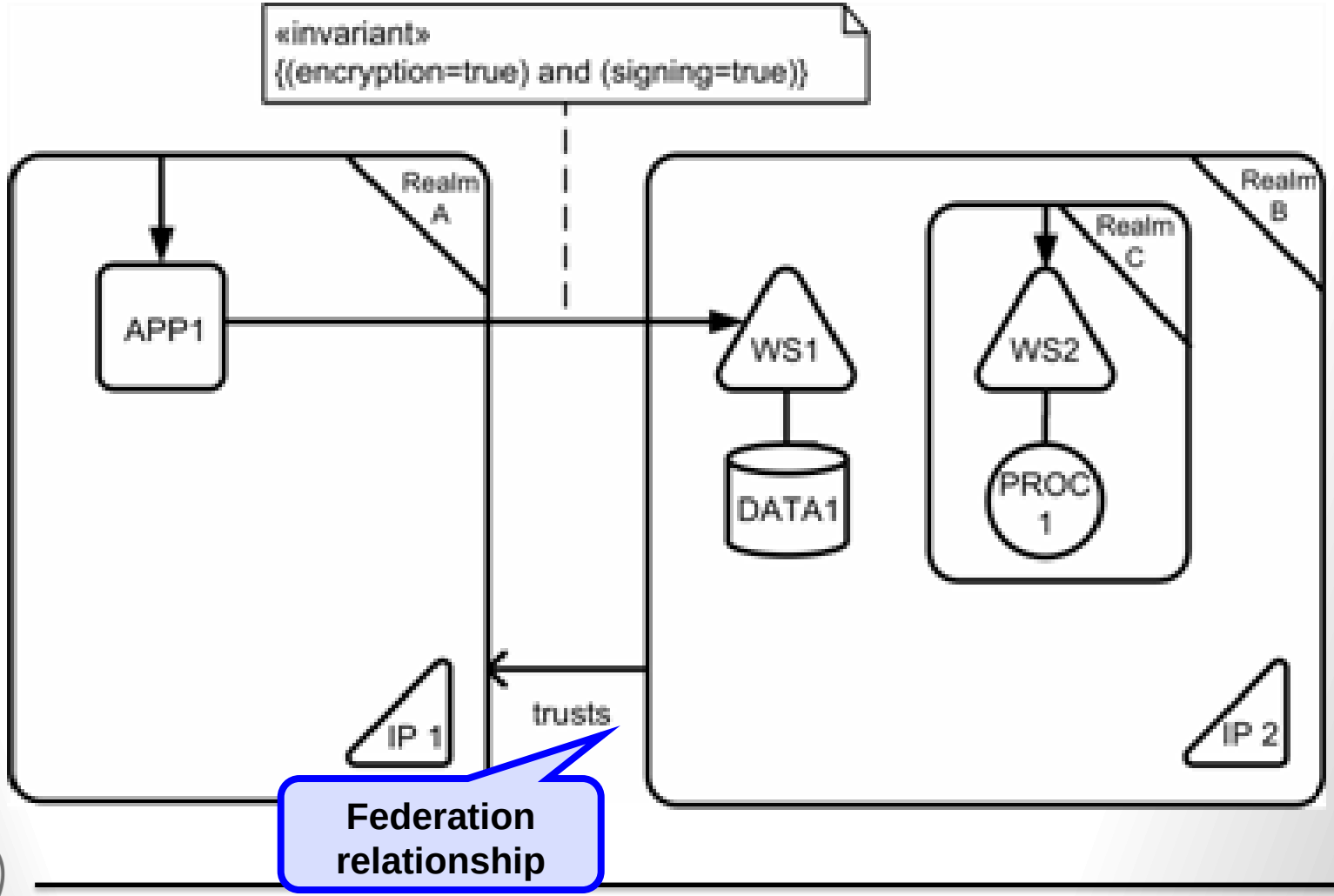

[f-label]

- Functionality (Functionality Profile)
  - Links Web Service technology with functionality
  - E.g. technology in use for calling process unit or data provider
- OCL might be used to describe details of Invocation and Trust

WAM-Modeling Example*



WAM Example


«invariant»
{(encryption=true) and (signing=true)}
Realm A
APP1
Realm B
WS1
DATA1
Realm C
WS2
PROC 1
IP 1
trusts
IP 2
Federation relationship

Identity Technologies for the Social Web

- Browser support for storing Username and Passwort
  - End user support, but not federation
- OAuth
  - IETF standard to share private resources with third parties (without sharing credentials)
- OpenId
  - Authentication technology focussing on Single Sign On (SSO) and secure identity attribute exchange
  - Interesting: Identity is URI – not understood by end users
- WebID
  - Using self-signed Client-Side Certificates and FOAF
  - A.k.a.: FOAF+SSL; technically great approach, but problematic as it requires better browser support for certificate management
- Many more, e.g.
  - Infocard
  - XAuth
  - SAML
  - Kantara Trust Framework
  - Several European projects working on technologies, e.g. PrimeLife

CHAPTER://18

- Further Readings

Literature

- Chapter 3, 6: Thomas A. Powell, Web Site Engineering, Prentice Hall PTR
- Chapter 7, 8, 9, 15: David Lowe and Wendy Hall, Hypermedia and the Web – an Engineering Approach, John Wiley & Sons
- Chapter 10, 11, 12, 14, 16: Ian Sommerville, Software Engineering, Addison-Wesley
- Chapter 1-4: Scott Seely, SOAP – Cross Platform Web Service Development Using XML, Prentice Hall
- Chapter 1, 2, 5: William L. Oellermann, Jr., Architecting Web Services, apress
- Chapter 2, 3, 4, 5: M. Mühlhäuser, A. Schill, Software Engineering für verteilte Anwendungen, Springer
- Chapter 9: I. Jacobson, G. Booch, J. Rumbaugh, The Unified Software Development Process, Addison-Wesley, 1999
- Chapter 7, 8: S. W. Ambler, Process Patterns – Building Large-Scale Systems Using Object Technology, Cambridge University Press

Literature

- K. Fellner und K. Turowski, Approach to Solve Content-related Conflicts in Component-based Business Application Systems, 12th International Conference Software & Systems Engineering and their Applications (ICSSEA'99), Paris, 1999.
- M. Fowler, Kendal Scott: UML Distilled: A Brief Guide to the Standard Object Modeling Language, Addison-Wesley Longman, Inc, 1999
- M. Gaedke, M. Beigl, H.-W. Gellersen und C. Segor, Web Content Delivery to Heterogeneous Mobile Platforms, in Lecture Notes in Computer Science (LNCS), Springer Verlag, vol. 1552(Advances in Database Technologies), S. 205-217, 1998.
- H.-W. Gellersen, F. Lyardet, M. Gaedke, D. Schwabe und G. Rossi, Patterns and Components: Capturing the Lasting amidst the Changing, Active Web Conference, U.K., 1999.
- D. M. Kristol und L. Montulli, HTTP State Management Mechanism, IETF, Network Working Group, RFC RFC 2109, Feb. 1997 1997. ftp://ftp.isi.edu/in-notes/rfc2109.txt
- D. A. Menasce, V. A. E. Almeida, Capacity Planning for Web Services, Prentice Hall, 2002.
- B. Oestereich: Objekt-orientierte Software Entwicklung mit der UML, Oldenbourg Verlag, 1997

Literature

- K. Ostebye, Hierarchical Structure through Navigation Side Bars, Workshop on Hypermedia Development - Design Patterns in Hypermedia, Hypertext Conference (HT99), Darmstadt, 1999.
- G. Rossi, D. Schwabe und F. Lyardet, Improving Web Information Systems with Navigational Patterns, 8th International World Wide Web Conference (WWW8), Toronto, Ontario, Canada, 1999.
- J. Sametinger, Software Engineering with Reusable Components. Berlin: Springer, 1997.
- R. Steinmetz und K. Nahrstedt, Multimedia : computing, communications, and applications. Upper Saddle River, NJ: Prentice Hall, 1995.
- C. Szyperski, Component software: beyond object-oriented programming. Reading, Mass.: ACM Press; Addison-Wesley, 1997.
- A. S. Tanenbaum, Computer networks, 3rd ed. Upper Saddle River, N.J.: Prentice Hall PTR, 1996.

==========

Further information available at Lecture Web Site

==========

Web References

- URLs
  - OOHDM, WebML, WebComposition, etc. via: http://webengineering.org/
  - Patterns Home Page: http://hillside.net/patterns/
  - Patterns for E-commerce applications: http://www-di.inf.puc-rio.br/~schwabe//papers/Europlopoo.pdf
  - Hypermedia Design Patterns Repository: http://www.designpattern.lu.unisi.ch/HypermediaHomePage.htm