



System Vision, System Idea & Views

Architectural Thinking for Intelligent Systems

Winter 2019/2020

Prof. Dr. habil.Jana Koehler





Agenda

- Formulate the Vision
- Develop a first System Idea
- How to work with Views & Viewpoints
- Importance of the operational model





Tutorial Assignment 6

- We formulate the vision for our system and sketch a first idea of its architecture.
- Furthermore, we work out the context view,
- think carefully about the operational model and
- dive deeper into system behavior.
- We capture our findings in runtime and distribution views.





- 4 Things an Architect must do Right
 - **1. Vision** Where do we want to go?
 - 2. Context In which environment do we want to be successful?
 - **3.** Architecture overview What do we build?
 - also often AS-IS vs. TO-BE
 - **4. Operational Model** How will the system "live"?
 - more than just the distribution view (costs, evolution,...!)





The Context View

- How is the system embedded into its environment?
 - Interfaces to to all(!) neighboring systems
 - Interfaces to all users
 - All events that can enter/leave the system
 - Any data the system sends to /receives from environment
- The system itself is a black box
- Our attention concentrates on the environment and how the system must interact with all elements in it
- All incoming and outgoing data and events
 - Interactions of the system with any stakeholder
 - Relevant parts of the external infrastructure





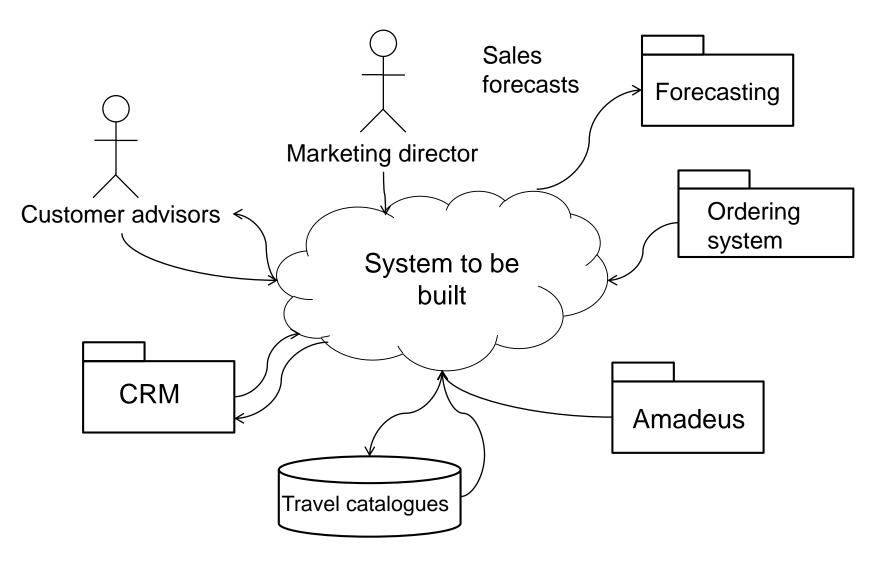
Importance of the Context View

- As an overview diagram of the system environment
 AND
- As the view for resource planning and risk management
 - Number & type of external interfaces
 - Number & type of dependencies on external systems
 - Number & type of stakeholders involved
- Context views with different levels of detail
 - Always create a detailed view when negotiating the scope of the project!





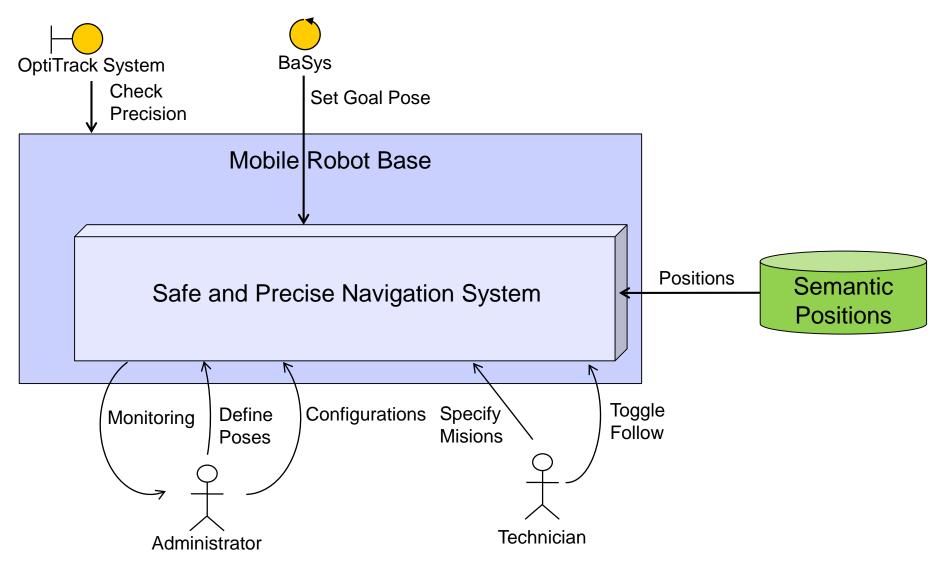
Example Context View - What is missing here?







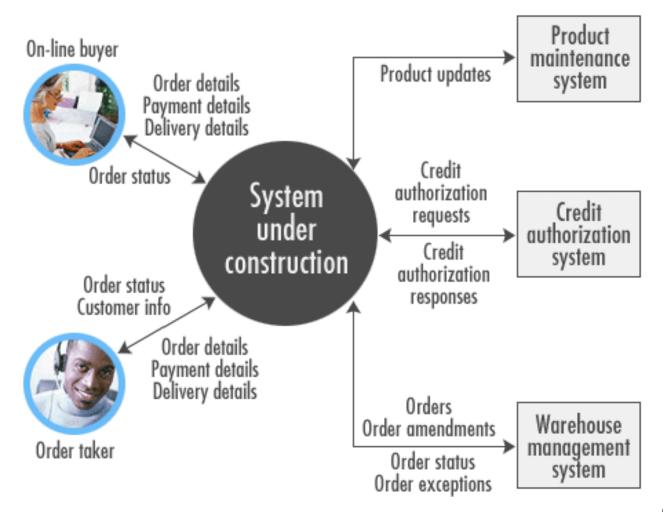
What can we improve in this Context View?







What can we say about the System based on this Context View?

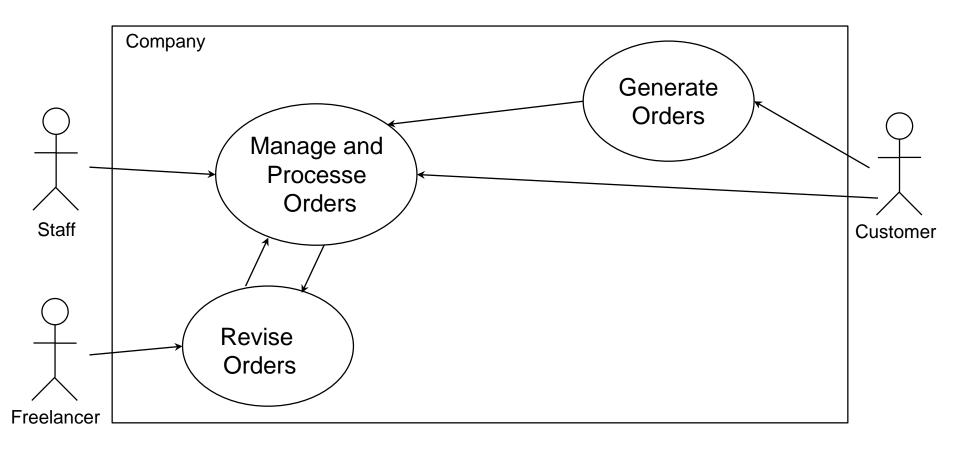


Quelle: IBM





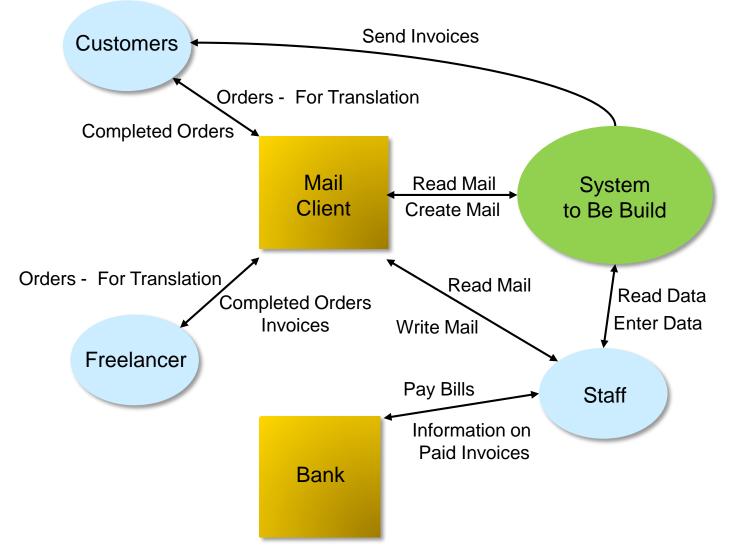
First Draft of a Context View







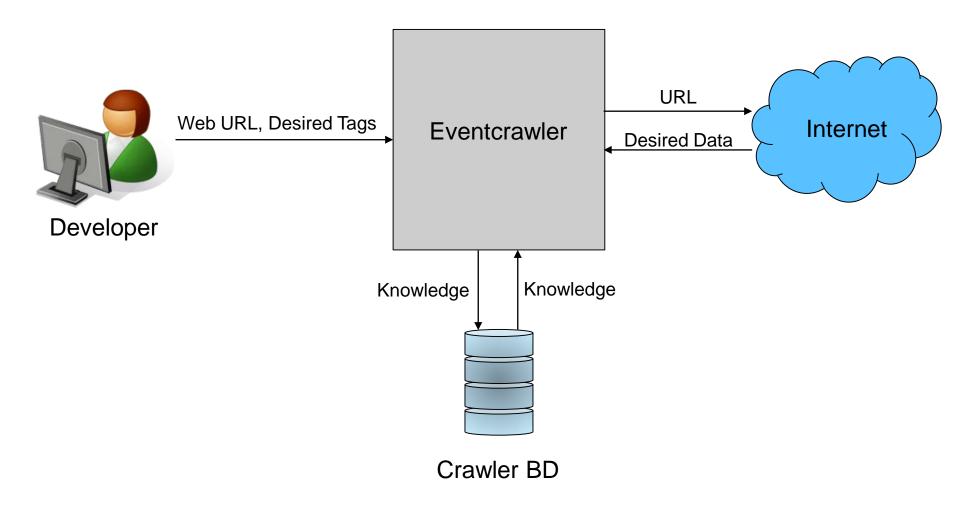
2nd Iteration







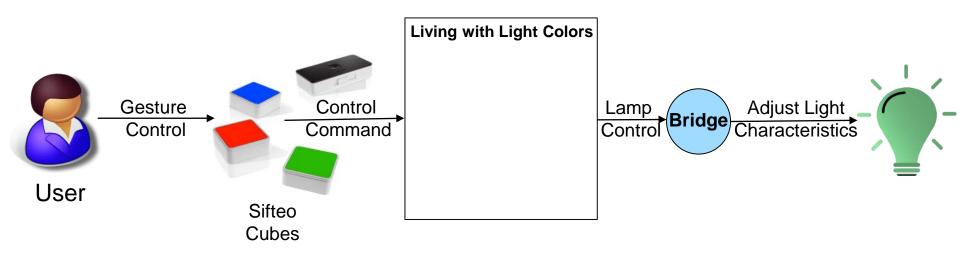
What's not good here?







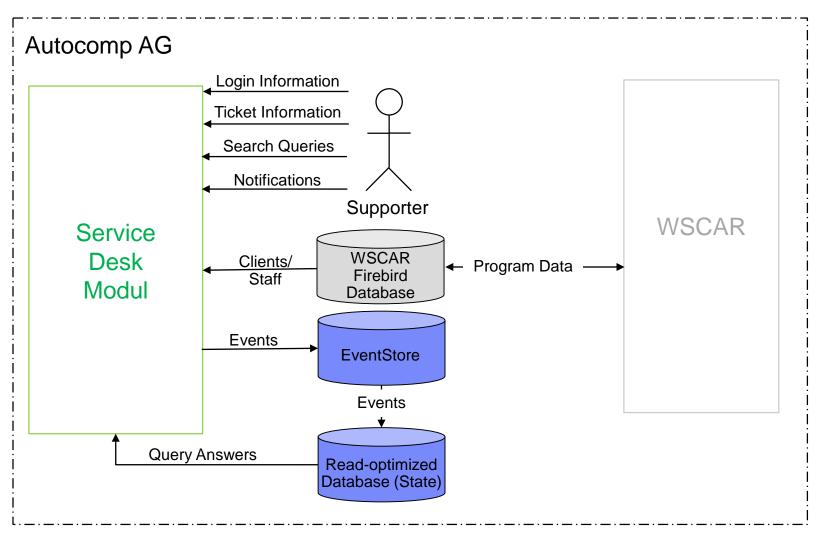
What's the Problem here?







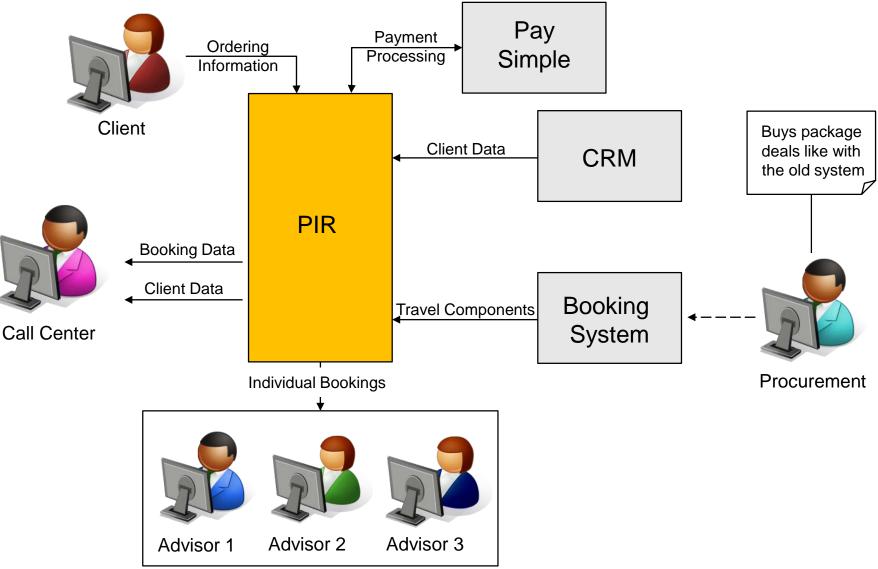
Are System Boundaries clear?







Are System Boundaries clear?



Architectural Thinking for Intelligent Systems: System Context, Vision & Views





Developing a VISION – Port Technology Example

- High-rise building with genuine mixed use containing office, residential and public areas
- Individual & efficient mobility
- Offer passengers an optimal route through the building to the desired destination. The seamless journey starts upon entering the building with the assigned elevator already waiting at the ground floor.
- The intelligent combination of traffic control and access control creates maximum efficiency and meets the requirements of the individual user groups in the building.
- Whether it is a design improvement that enables a new building to achieve far more than originally envisioned or an upgrade to give an existing building a new lease on life, The PORT Technology can act as a major enabler of positive change.

https://www.theporttechnology.com/page/solutions.html https://www.schindler.com/com/internet/en/media/behind-the-scenes/customer-projects/2018/omniturm-frankfurt.html





Writing a Vision Statement

- For [customer]
- **Who** [needs... or has opportunity]
- The [product]
- Is [category]
- That [major capability, benefit, ...]
- **Unlike** [alternative, current system, current practice]
- **Our Product** [differentiation, advantages, ...]





Developing *The* System Idea

- We find the system idea by applying principles and tactics to our architectural problem and develop adequate solutions
- Creative process
- There is more than one solution
- Find compromises
- Ask the right questions and reuse proven solutions





Find clear answers to these questions

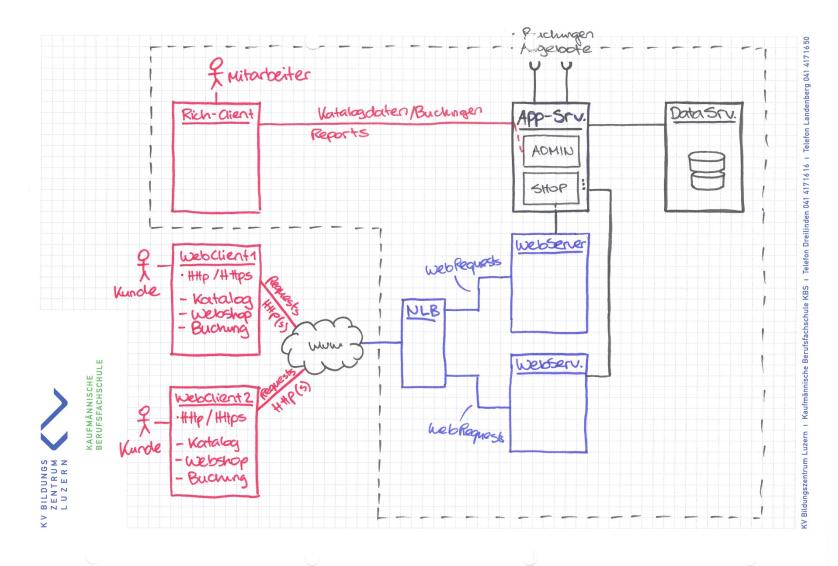
- Which NFR are the most important ones for the architecture?
- Which use cases at level "white" should we put into the main focus?
- What will be the core task of the system?
 - What are the most important elements of the business domain?
- How will the system be used?
- How will the system be controlled?
- Who uses the system?
 - What kind of user interface will the system need?
 - Which business processes and user interactions will the system support?
- What interfaces to other systems will the system need?
- How will data be managed & processed?
 - What kind of data access is necessary?

Which structural components and relations can best implement the answers to these questions?





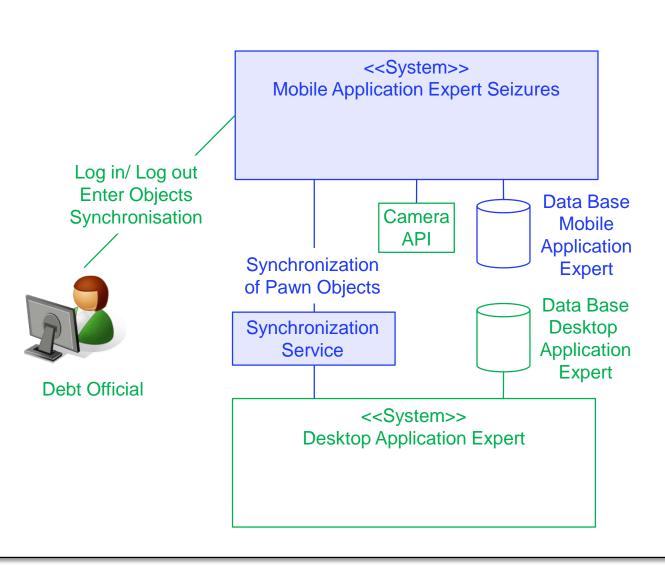
Example System Idea







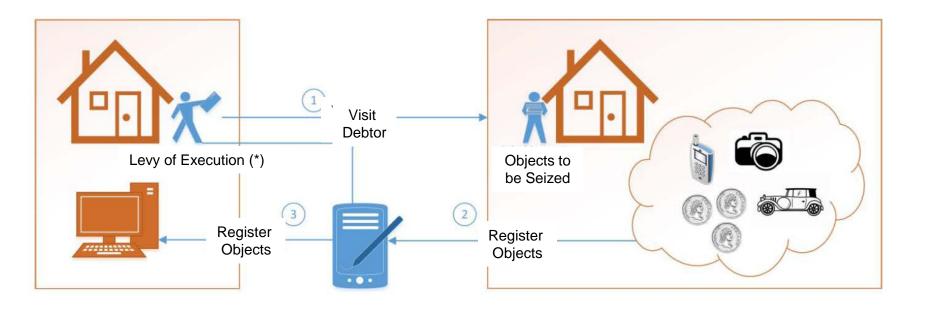
Example System Idea







Second Iteration of the System Idea

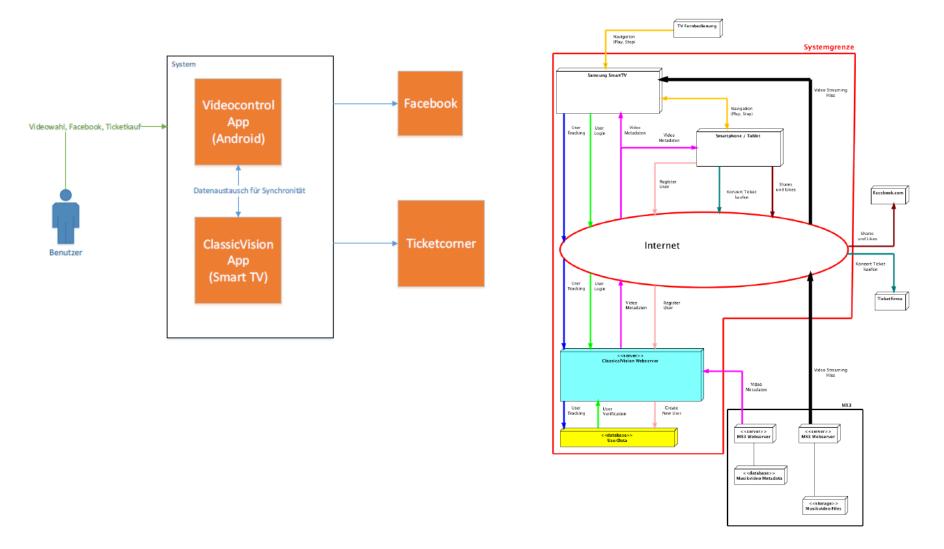


(*) Zwangsvollstreckung



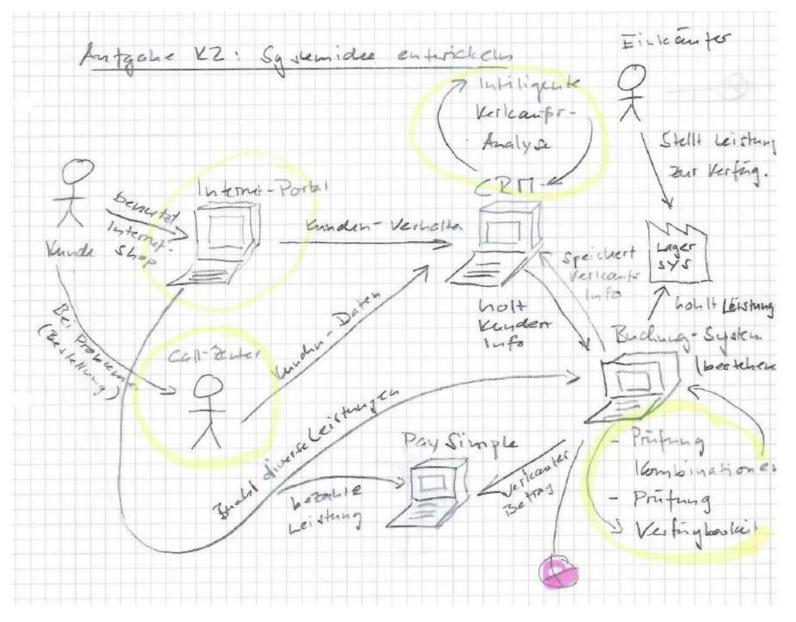


Proposal for Context View and System Idea





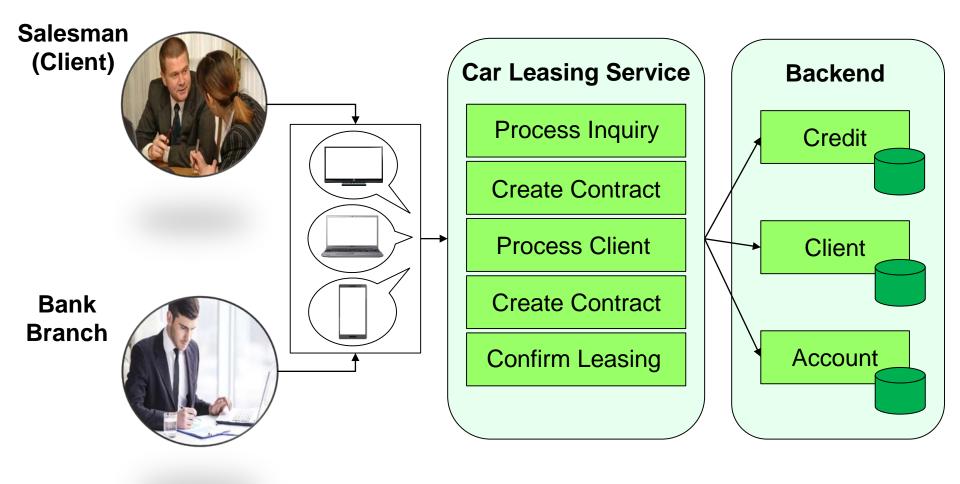








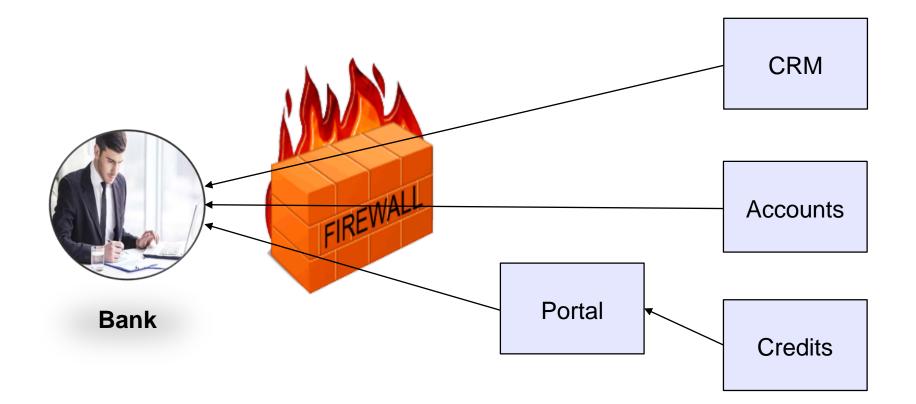
System Idea & Architectural Overview





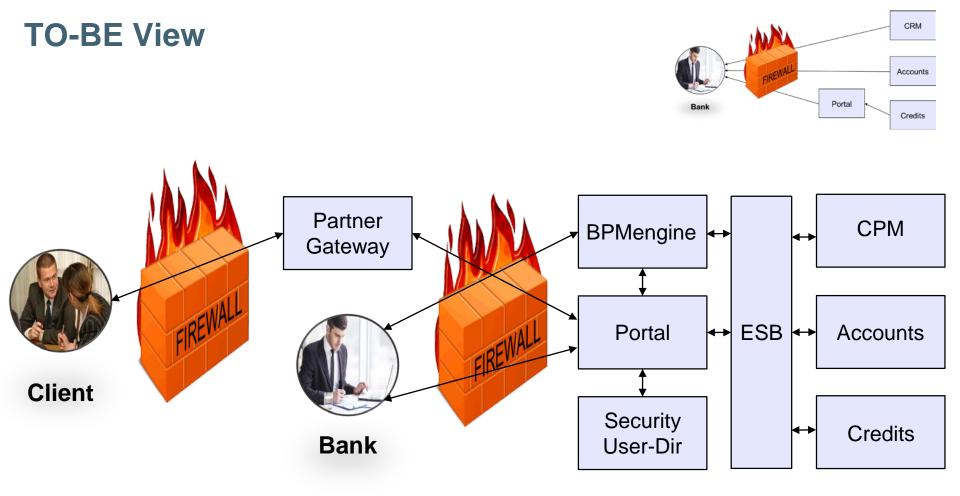


AS-IS View













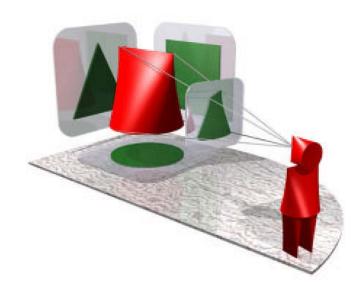
Why Views are Important

 System structure and architectural decisions must be documented and communicated



- For whom?
- Why?
- How?

"If it is not written down, it does not exist." Philipp Kruchten







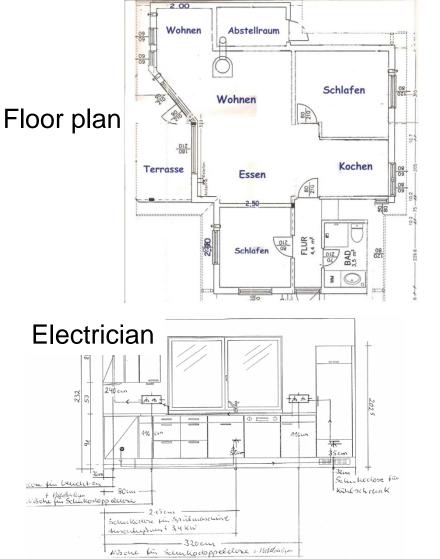
Views and Viewpoints

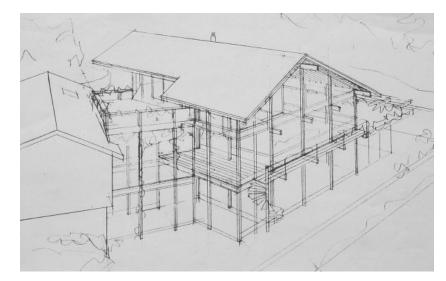
- A view represents a complete system from the perspective of a set of interrelated interests. [ISO/IEC/IEEE 42010:2011]
 - View as a set of related architecture-relevant elements that is created and used by the stakeholders of a system
- A viewpoint is a collection of patterns, templates and conventions for constructing one type of view. It defines the stakeholders, guidelines and principles and template models for constructing its views.





Views in Building Architecture





Vertical plan



Architectural Thinking for Intelligent Systems: System Context, Vision & Views





Why Views?

- Modern software is often too complex to be viewed and understood as a whole
- We need to focus our attention on one (or a few) parts of the system structure to facilitate understanding
- In order to be able to communicate clearly, we need to make clear which structures, relationships or behavior focus on during discussion
- VIEW as a representation of the selected structures
- Architects develop structures and document these structures through views





Creating GOOD Views

- Meeting stakeholder's needs
 - What information does a stakeholder need?
- As little formalism as possible, as much as necessary!
- The more risk involved with a decision, the more detailed the view illustrating it
- Work top-down/bottom-up and in iterations to refine and revise views





How can we Assess the Quality of a View?

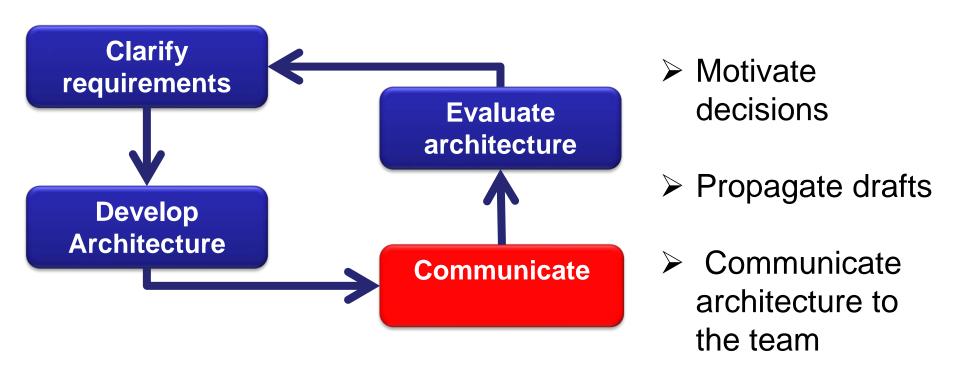
- Need to ask "good for which purpose?"
- What are we trying to achieve?
- What do we need to communicate to whom?
- If the view delivers our intended message correctly and successfully to the stakeholder ...
 - ... and we reach the goal of our message







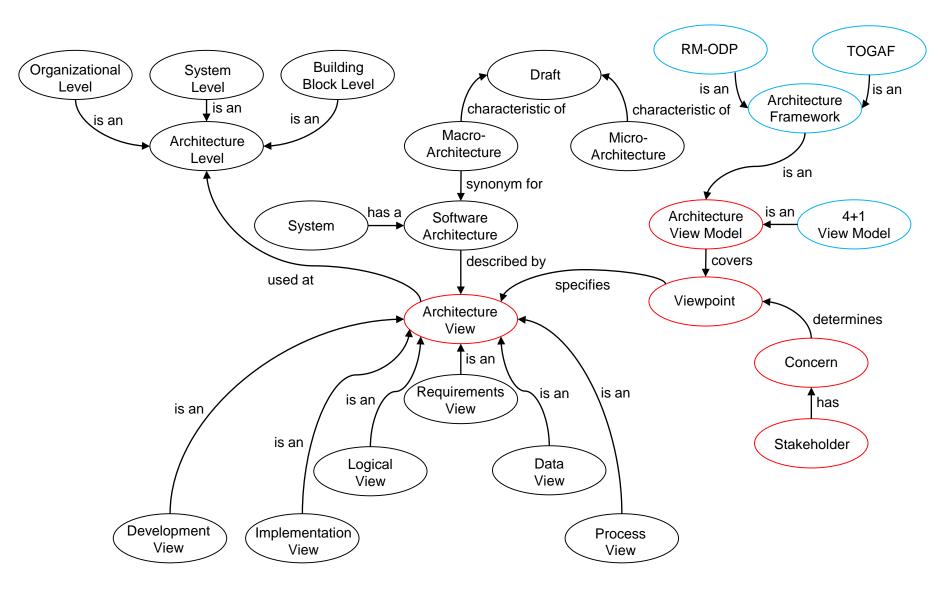
Views in the Communication of the Architect



Different views (abstractions) for different participants





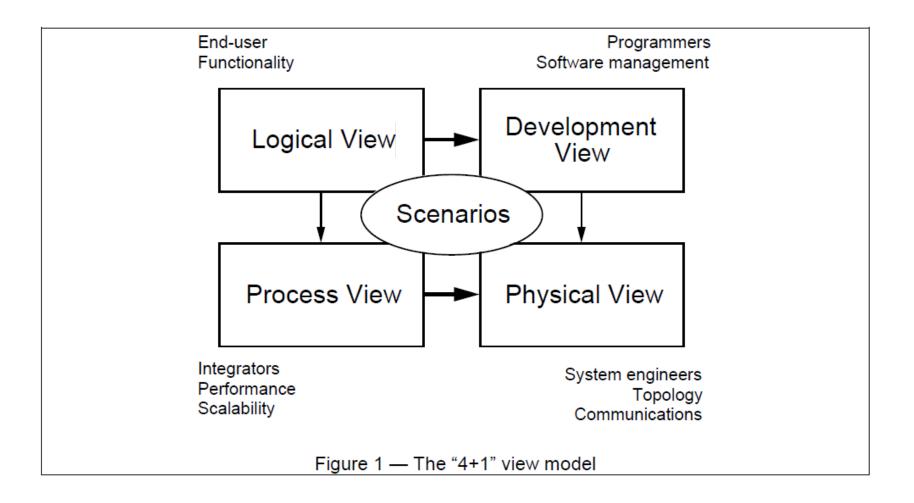


Vogel et al: Software Architektur





The 4+1 Views Model by Kruchten







Views in TOGAF

- A system is a collection of components organized to accomplish a specific function or set of functions.
- The architecture of a system is the system's fundamental organization, embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution.
- An architecture description is a collection of artifacts that document an architecture. In TOGAF, architecture views are the key artifacts in an architecture description.
- Stakeholders are people who have key roles in, or concerns about, the system; for example, as users, developers, or managers. Different stakeholders with different roles in the system will have different concerns. Stakeholders can be individuals, teams, or organizations (or classes thereof).
- Concerns are the key interests that are crucially important to the stakeholders in the system, and determine the acceptability of the system. Concerns may pertain to any aspect of the system's functioning, development, or operation, including considerations such as performance, reliability, security, distribution, and evolvability.





Views in TOGAF

- A view is a representation of a whole system from the perspective of a related set of concerns.
- In capturing or representing the design of a system architecture, the architect will typically create one or more architecture models, possibly using different tools. A view will comprise selected parts of one or more models, chosen so as to demonstrate to a particular stakeholder or group of stakeholders that their concerns are being adequately addressed in the design of the system architecture.
- A viewpoint defines the perspective from which a view is taken. More specifically, a viewpoint defines: how to construct and use a view (by means of an appropriate schema or template); the information that should appear in the view; the modeling techniques for expressing and analyzing the information; and a rationale for these choices (e.g., by describing the purpose and intended audience of the view).

http://pubs.opengroup.org/architecture/togaf8-doc/arch/chap31.html





View vs. Viewpoint

- A view is what you see. A viewpoint is where you are looking from - the vantage point or perspective that determines what you see.
- Viewpoints are generic, and can be stored in libraries for reuse. A view is always specific to the architecture for which it is created.
- Every view has an associated viewpoint that describes it, at least implicitly. ANSI/IEEE Std 1471-2000 encourages architects to define viewpoints explicitly. Making this distinction between the content and schema of a view may seem at first to be an unnecessary overhead, but it provides a mechanism for re-using viewpoints across different architectures.





To address the concerns of the following stakeholders			
Users, Planners, Business Management	Database Designers and Administrators, System Engineers	System and Software Engineers	Acquirers, Operators, Administrators, & Managers
the following views may be developed			
Business Architecture Views	Data Architecture Views	Applications Architecture Views	Technology Architecture Views
Business Function View	Data Entity View	Software Engineering View	Networked Computing/ Hardware View
Business Services View			
Business Process View			
Business Information View			
Business Locations View			Communications Engineering View
Business Logistics View	Data Flow View (Organization Data Use)	Applications Interoperability View	
People View (Organization Chart)			Processing View
Workflow View			
Usability View			
Business Strategy and Goals View	Logical Data View	Software Distribution View	Cost View
Business Objectives View			
Business Rules View			Standards View
Business Events View			Standards view
Business Performance View			
	System Engineering View		
Enterprise Security View			
Enterprise Manageability View			
Enterprise Quality of Service View			
Enterprise Mobility View			







- 4 Views We Focus on
 - Context View
 - Component View
 - Runtime View (behavioral model)
 - Distribution View (operational model)





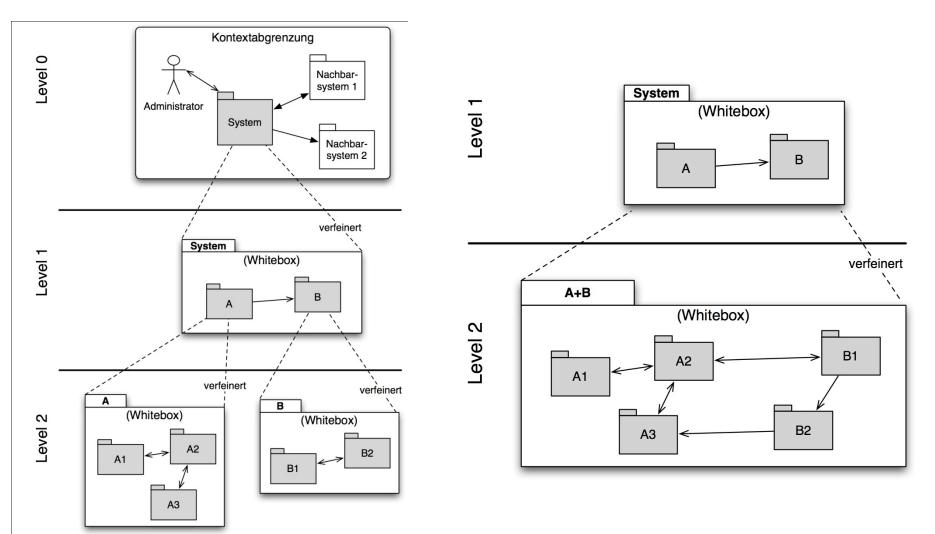
Component View

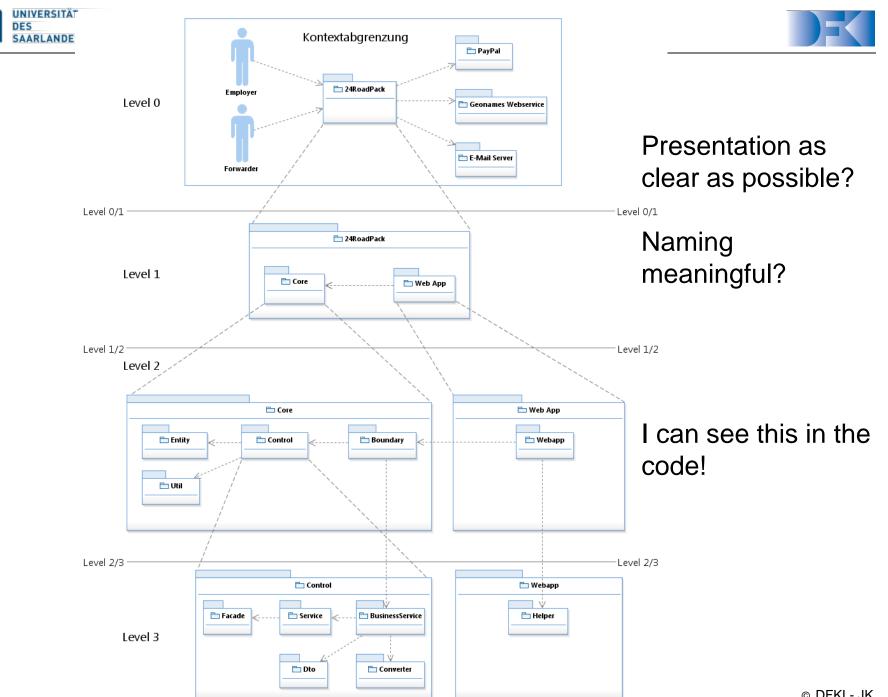
- How is the system structured internally?
 - What is its message?
 - What do I see beyond of what I can see in the code?
- Static structures of the architectural building blocks of the system
 Subsystems, components, their interfaces and relations
- Support project managers and clients in project monitoring
- Allocate work packages (architecture modules) to teams and employees
- Reference for software developers
- Top-down refinement
 - Last (possible) level of refinement is the source code





Examples of Component Views

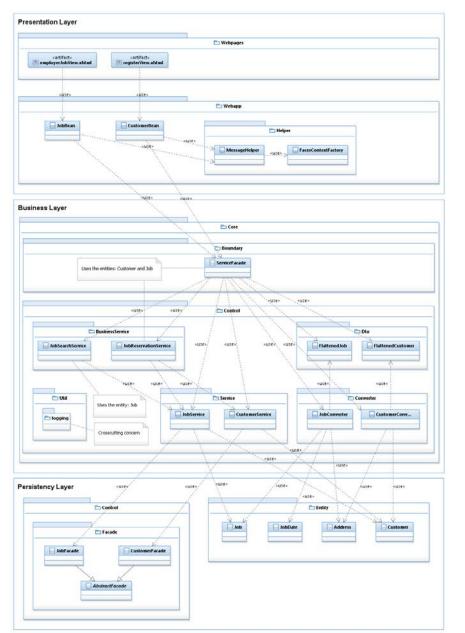




44







Not really clear any more, but you can see weaknesses in the component structure





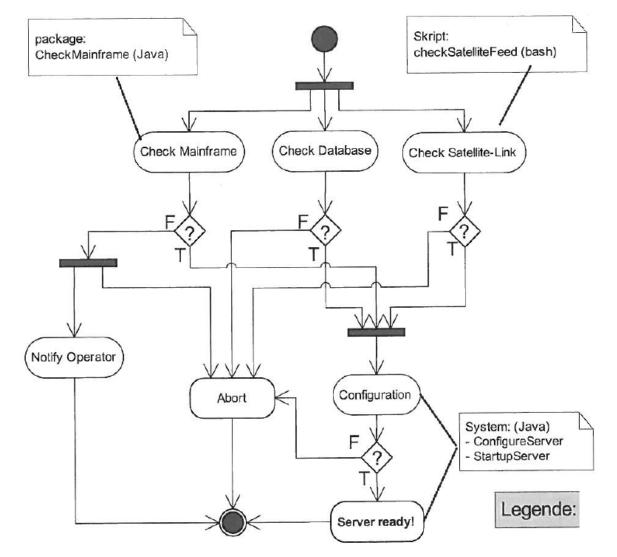
Runtime View

- How do processes run in the system from a user's point of view?
- Dynamic structures and behaviors in the system
 - Which components of the system exist at runtime?
 - How do they interact?
- But also :
 - Where and how do users interact with the system?
 - Where are security risks?
 - How are system qualities such as scalability or performance achieved?





Example of a Runtime View using a UML Activity Diagram

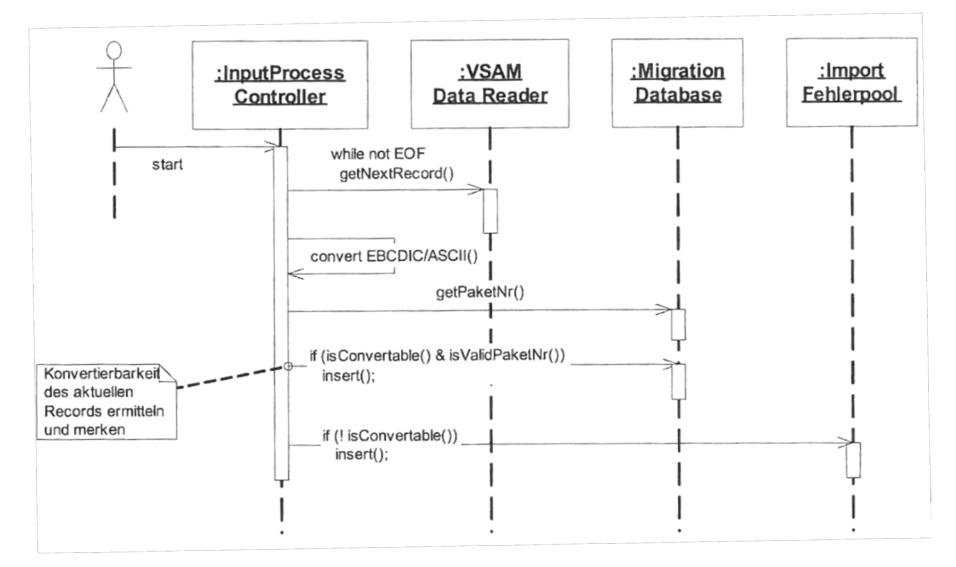


With some effort, I also see this in the code! Where are the architecture-relevant answers to our questions?





Example of a Runtime View as a UML Sequence Diagram







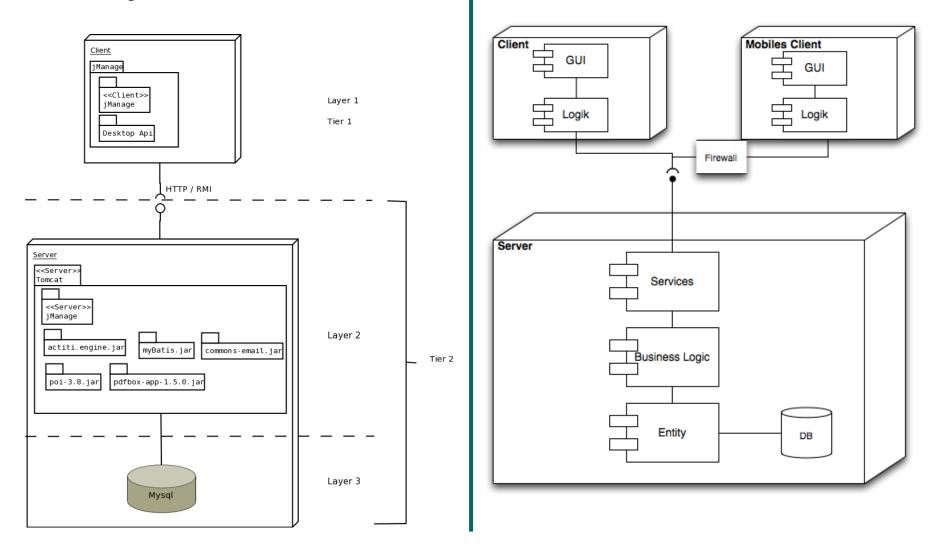
Distribution View - Operational Model - Operational View Infrastructure View

- In which environment does the system run?
 - System from the operator's point of view
- Distribution of the software system to the physical systems
 - Hardware components on which the software runs
 - Computers, processors, network topologies and protocols
- But also:
 - What infrastructure requirements do we have?
 - How expensive will the system be when fully expanded?
 - How do we operate and maintain it?
 - Can it evolve to foreseeable future requirements?





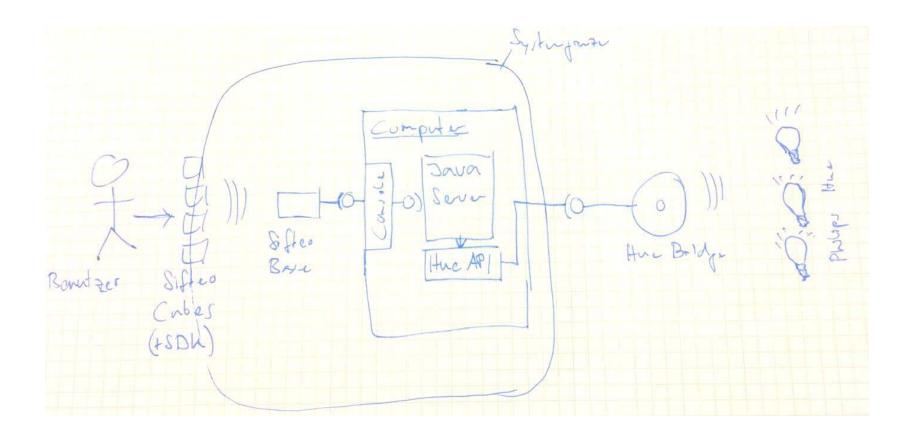
2 Distribution Views of the same System







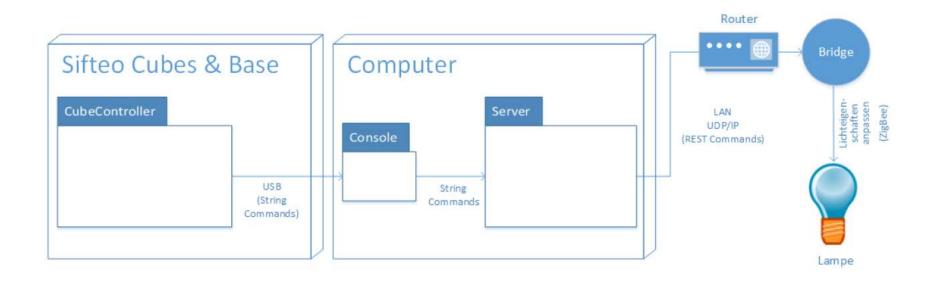
A View?







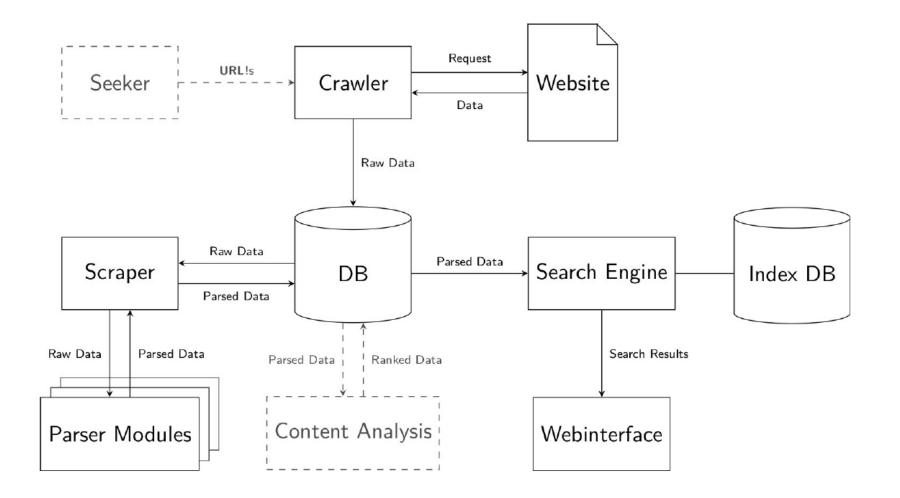
What Type of View is this?





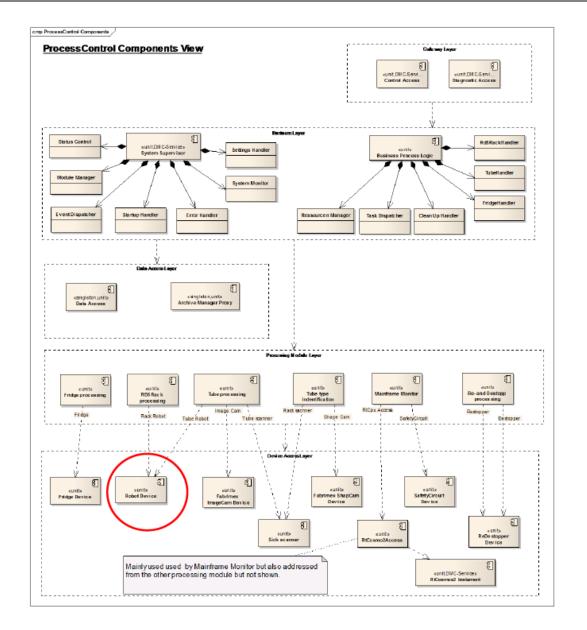


And this?









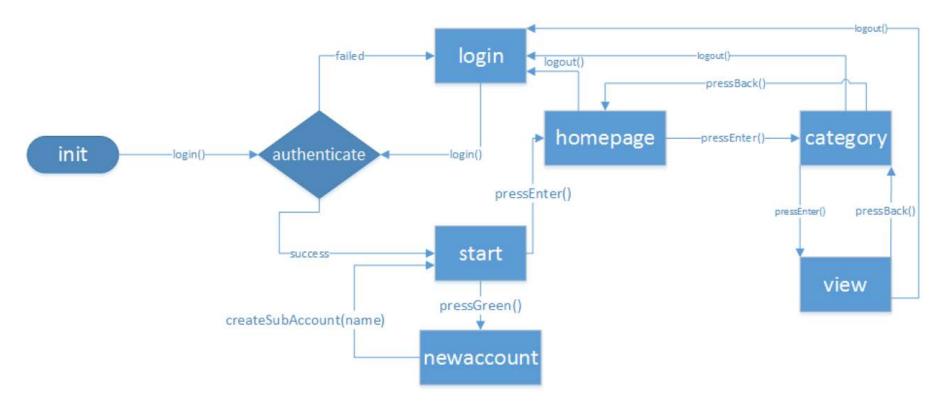
Is this communicated clearly and comprehensibly?





What Type of View is this?

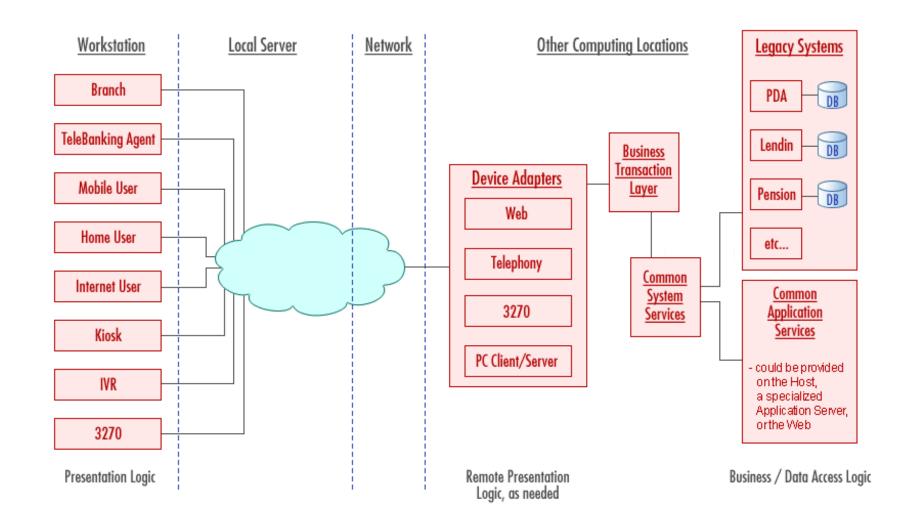
SmartTV Sichten (Scenes)







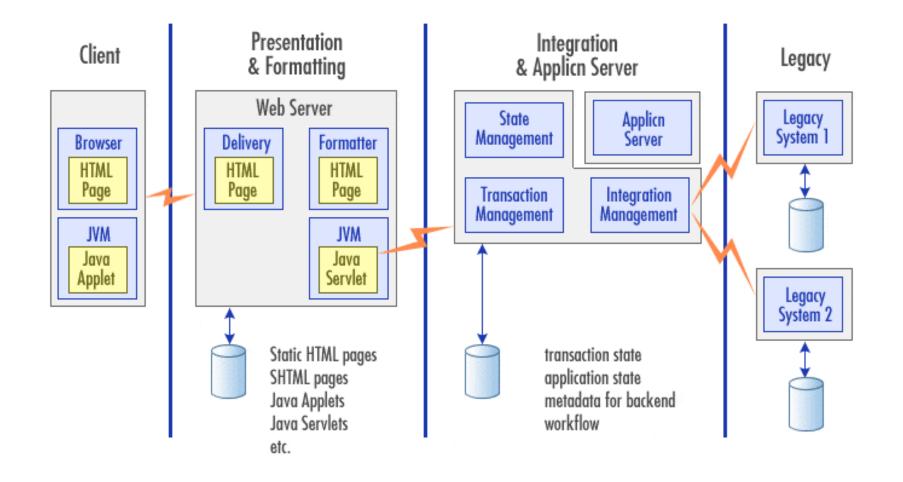
Which View is represented by this Diagram?







Which View is represented by this Diagram?

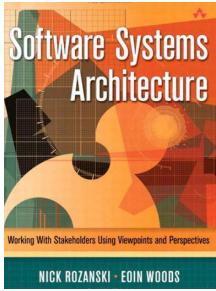






Summary

- Context, Context, Context!!
- Context view is the most underestimated view by novice architects
- Omissions in the system context are the main source of risks
- Spend time in creating a system vision and review it regularly
- Think in alternatives when creating the system idea, the first idea might not be the best
 - Start implementing the most risky parts
- Real projects require other important views:
 - Security view (from different perspectives)
 - Information, business processes, data flows
 - User Experience (GUI)







Working Questions

- 1. How do we document structures and behaviors of a system?
- 2. What do we understand by a view and which views do you know?
- 3. What constitutes a "good" view?
- 4. How much detail needs to be put into a view?
- 5. Why do component views often only represent the structure of the code?
- 6. Why is the operational view one of the most important views alongside the context view?
- 7. Why is it so difficult to create a good context view?
- 8. What is a viewpoint?