Security as an Architecture Quality

Nelis Boucké
Alexander Helleboogh
Johan Peeters

KATHOLIEKE UNIVERSITEIT LEUVEN DistriNet IBBT RESEARCH GROUP

archiwise.com

Setting

- Our background
  - Software architecture
    - Research @DistriNet, Consultancy
    - Security: one concern amongst others
- Perspective
  - How to deal with security in software architecture?

Outline

- What is software/security architecture?
- What drives software architecture and how does security fit in?
- How to describe a (security) software architecture?
- Conclusion

Background of audience

Question

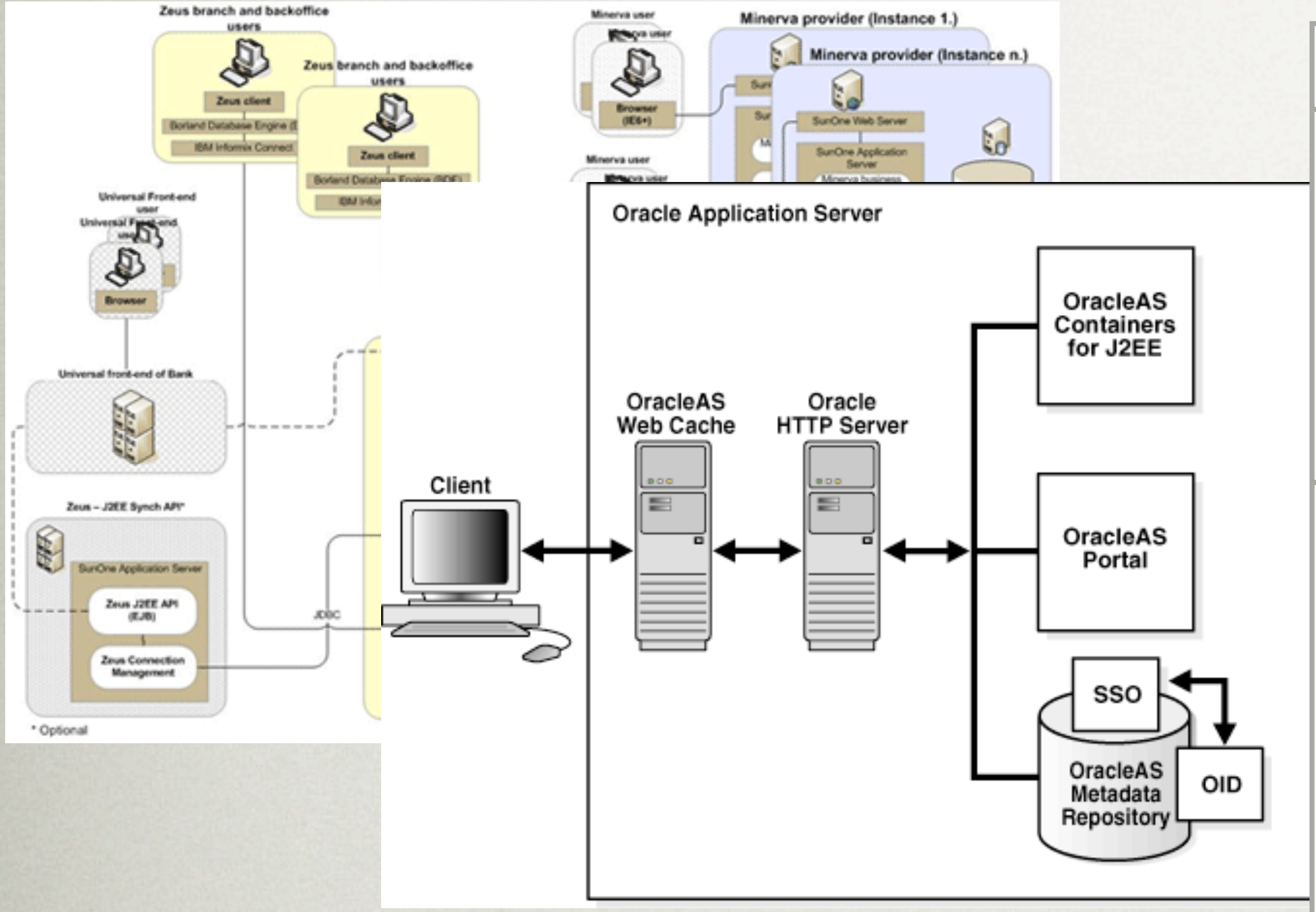
- What do you consider (security) architecture?
- What systems are you working on?

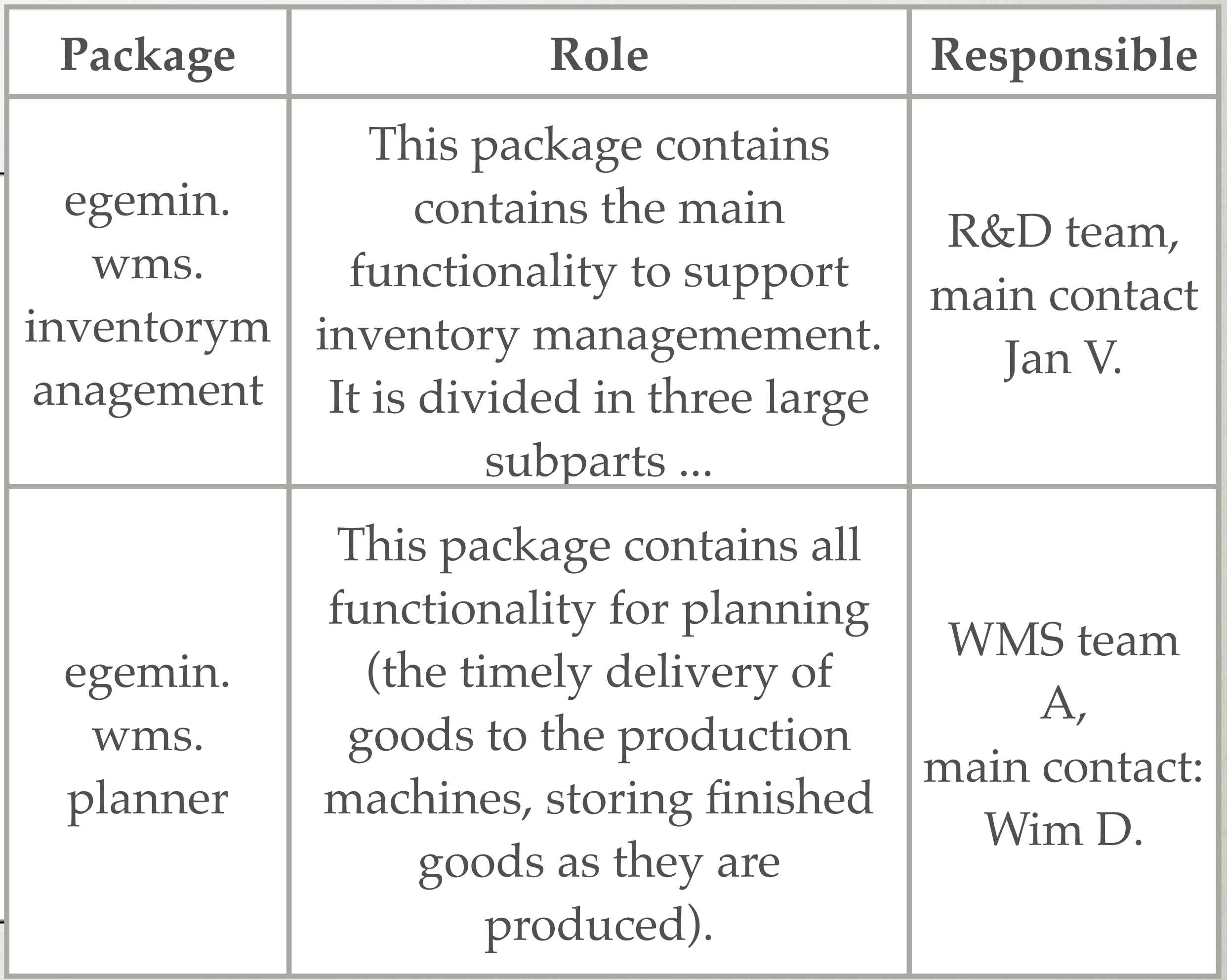
Architecture Definition

- Technical definition: ISO 42010*
  - “The architecture of a system is the **fundamental conception of a system** in its environment embodied in **elements**, their **relationships** to each other and to the environment, and **principles** guiding system design and evolution.”

*** ISO 42010: Systems and software engineering -- Architecture description**

Why do we need architecture?


Zeus branch and backoffice users
Zeus branch and backoffice users
Zeus client
Borland Database Engine (BDE)
IBM Informix Connect
Universal Front-end user
Browser
Universal front-end of Bank
Zeus – J2EE Synch API*
SunOne Application Server
Zeus J2EE API (EJB)
Zeus Connection Management
JDBC
* Optional
Minerva user
Browser (IE6+)
Minerva provider (Instance 1.)
Minerva provider (Instance n.)
SunOne Web Server
SunOne Application Server
Oracle Application Server
Client
OracleAS Web Cache
Oracle HTTP Server
OracleAS Containers for J2EE
OracleAS Portal
SSO
OracleAS Metadata Repository
OID



Package	Role	Responsible
egemin. wms. inventorymanagement	This package contains contains the main functionality to support inventory managemement. It is divided in three large subparts ...	R&D team, main contact Jan V.
egemin. wms. planner	This package contains all functionality for planning (the timely delivery of goods to the production machines, storing finished goods as they are produced).	WMS team A, main contact: Wim D.

- **Communication** amongst stakeholders
- Manage complexity using **abstraction**
- Capture **early design decisions**

Security architecture

- Definition*
  - *"The **design artifacts** that describe how the **security controls** (= security countermeasures) are positioned, and how they **relate to the overall IT Architecture**. These controls serve the purpose to maintain the system's **quality attributes**, among them confidentiality, integrity, availability, accountability and assurance."*

*** http://www.opensecurityarchitecture.org/**

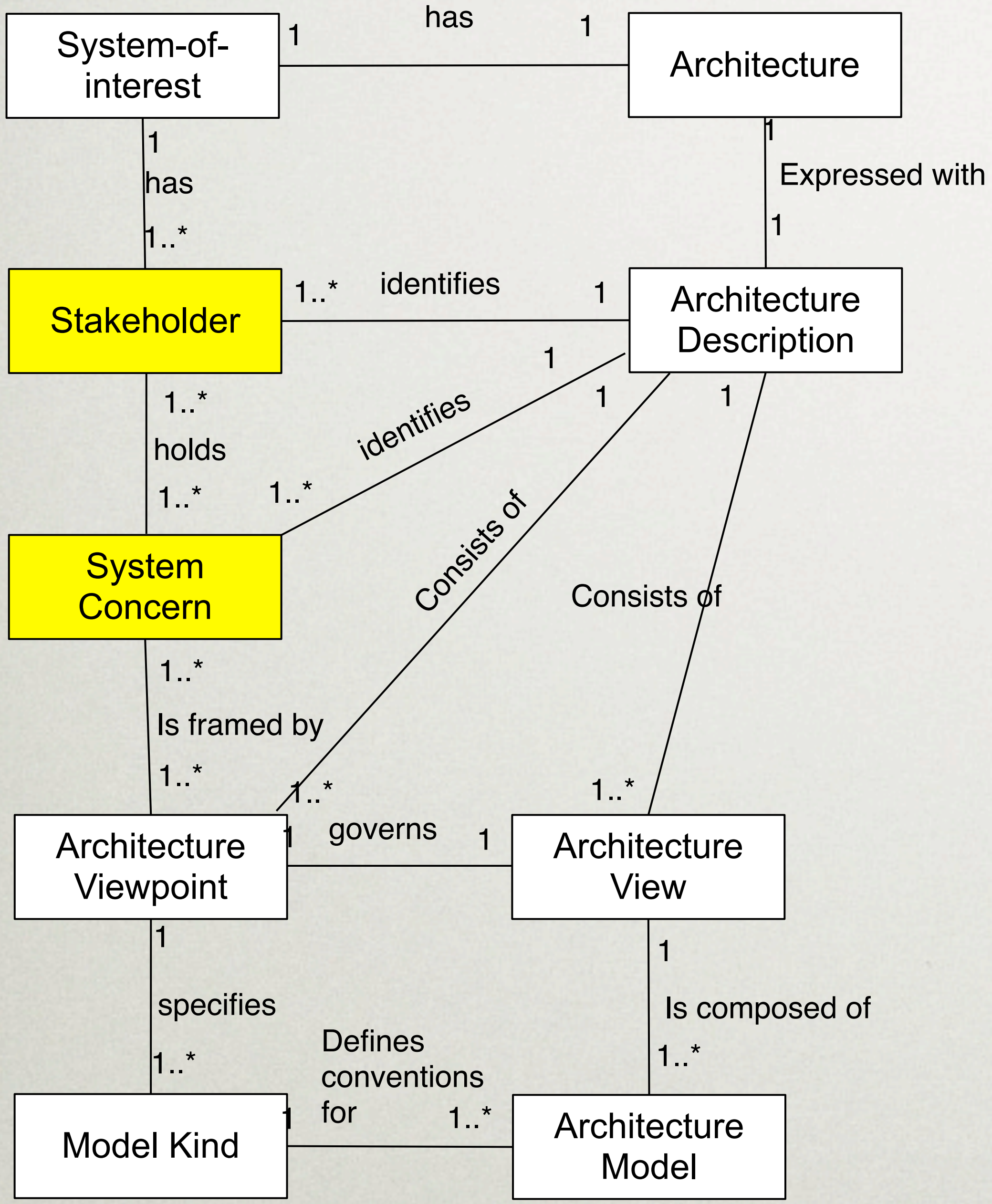
Security vs Software Architecture

- Security architecture suffers from tension w.r.t. software architecture in general:
    - Considered as separate (own methods & representations)
    - Need of being fully integrated with SA in general (guarantees, make trade-offs)
- Addresses non-normative flows
    - Abnormal flows, failure modes, what happens on interrupts, attacks or unexpected events

Outline

- What is software architecture?
- What drives software architecture and how does security fit in?
    - Functional vs. quality concerns
    - How to deal with quality concerns?
- How to describe a (security) software architecture?
- Conclusion

ISO 42010 concepts



- **Stakeholder**:
  - Individual, team, organization, or classes thereof, having concerns with respect to a system
- **Concern**:
  - Area of interest in a system pertaining to developmental, technological, business, operational, organizational, political, regulatory, social, or other influences important to one or more of its stakeholders

archiwise.com

Functional Concerns

- Functional concern/requirement
    - Defines **what** a system should be able to do
    - Example
        - The scheduling system should assign each task to one of the employees.
    - Extensively considered for most software systems
        - E.g. Use case scenarios, feature lists, business process models, etc.

Quality Concerns

- Quality concern[1]
  - Defines the **qualities** the system has to exhibit while fulfilling its function
  - Examples
    - Performance: a system with 12 conveyor belts should be capable of executing 140 transports/hour
    - Security: all transactions on the system can be traced to authenticated users.
    - Availability: After a power down the system should be able to restart in a consistent state
  - Security is a quality concern!

[1] Also known as quality; quality requirement; quality attribute; non-functional requirement

Architecture and Functionality

- **Claim:** Functionality does not constrain the architecture

Exercise   Do you agree? Can you give counter examples?

  - Any **functionality** can be achieved with a single monolithic structure
  - Architecture redesign?
    - NOT because the system is functionally deficient
    - But because it is too slow, crashes, scales bad,...

Architecture and Quality

- **Claim**: quality concerns drive the architecture
  - Example: Modifiability
    - E.g. a feature can be modified more easily if it is encapsulated as a separate component
  - Example: Security
    - E.g. supporting 3rd party plugins in a secure way entails using advanced sandboxing techniques.

Architecture and Quality

- Architectures have trade-offs w.r.t. qualities
  - Changing the architecture to promote one quality often affects the other qualities
  - E.g. security vs. performance
    - Maintaining an audit trail of all transactions improves security => reduces performance
- Architecture *alone* is not enough to realize qualities
  - E.g. modifiability diminished by obscure code

Functional vs. quality concerns

- Orthogonal
  - Example: improving security by adding audit trails => no effect on the functionality
- Not all levels of quality are achievable!
  - E.g. all tasks should be performed within 2 milliseconds
- Functional concerns have implicit quality concerns
  - E.g. security: angry Birds and privacy; e-mail; etc.
  - E.g. performance: reasonable response times, etc.

Outline

- What is software architecture?
- What drives software architecture and how does security fit in?
  - Functional vs. quality concerns
  - ⇨ How to deal with quality concerns?
- How to describe a (security) software architecture?
- Conclusion

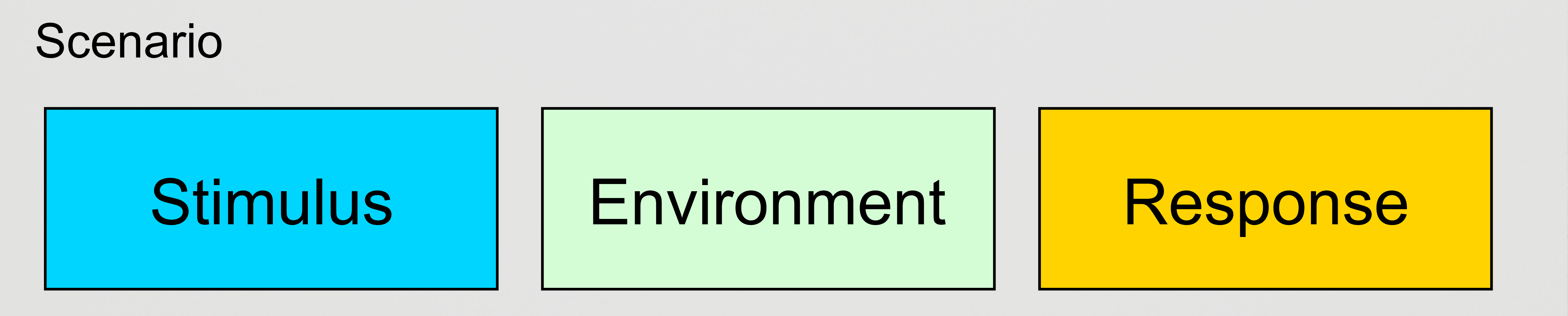
Quality concerns

Exercise

- From your experience
  - (How) do you describe (security) quality concerns?
  - How do you identify which quality concerns are the most important?

Describing quality concerns

- Quality attribute scenarios
  - Capture concrete scenarios for the qualities
    - **Concrete** within the targeted system
    - **Measurable** or **assessable**



Scenario

Stimulus

Environment

Response

Action of the stakeholder or condition

Situation at the moment of the stimulus

How the system responds

Must be measurable or assessable!

Describing quality concerns

- NOT:
  - The system must be secure against denial of service attacks.
- BETTER:
  - Under normal operating conditions, the system that is subjected to a distributed application-level flood attack (500 requests/second from non-authenticated users) should still be able to respond to service requests from authenticated users within 1 seconds.

Exercise Can you identify stimulus, environment, response?

Describing quality concerns

- NOT:
  - The system should detect attacks
- BETTER:
  - Exercise Can you formulate a scenario (stimulus/environment/response)

Example template

Scenario name	
Rationale/business goal	
Quality attributes	
Main stakeholders	
Scenario description (stimulus, environment, measurable response)	
Questions/open issues	

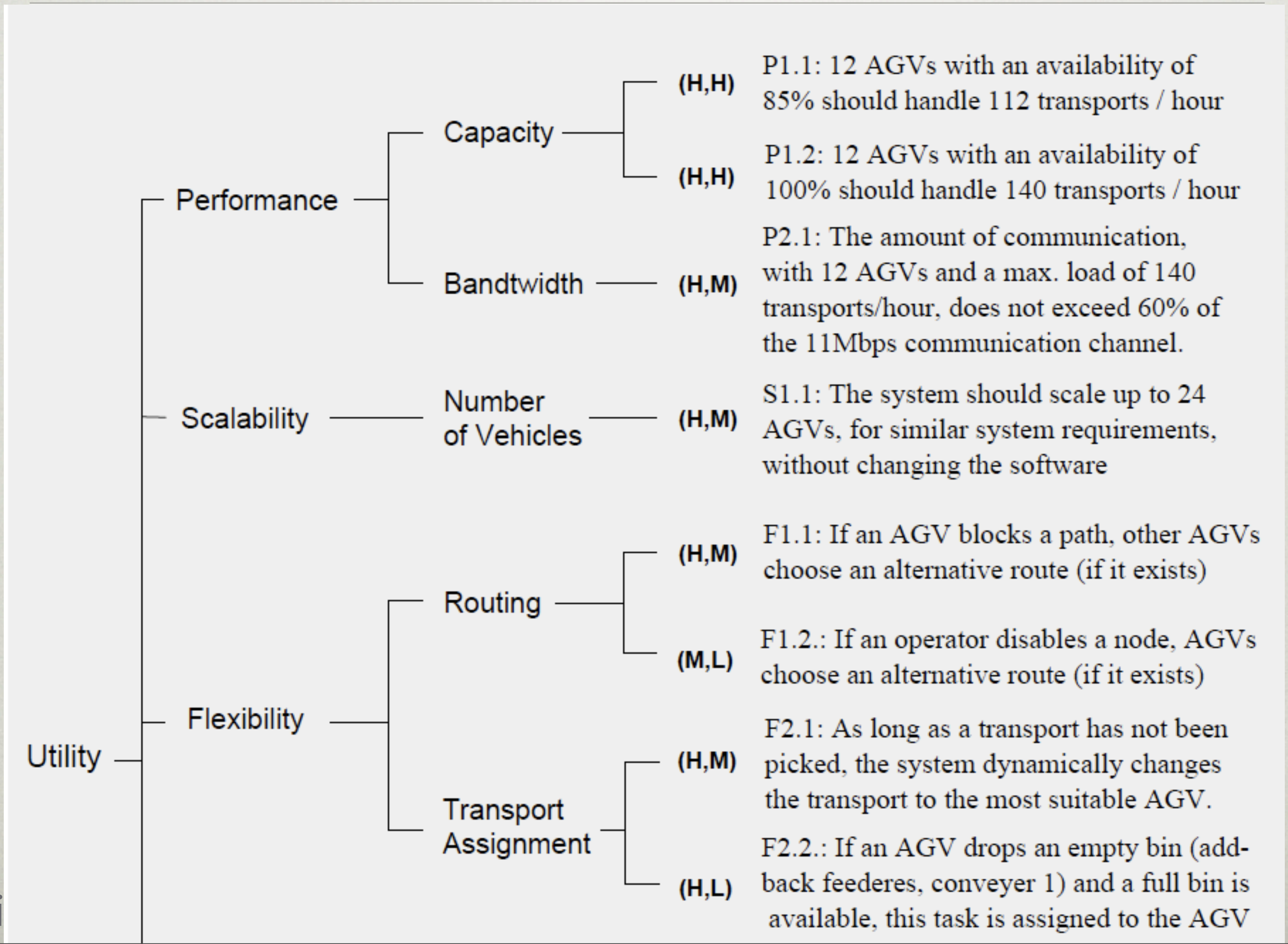
Prioritizing quality concerns

- **Security is one of many quality concerns** architects have to cope with
- Describing qualities as **concrete, measurable scenarios** facilitates
  - mutual understanding and reaching agreement
  - setting priorities and making trade-offs
    - Typical system: 20-60 quality scenarios

Prioritizing quality concerns

- 2 dimensions to rank scenarios
  - **Importance** of the scenario for the system
    - Business stakeholders
  - **Level of difficulty** to realize the scenario
    - Technical stakeholders
- For each dimension and each scenario prioritize using
  - **High, Medium, Low**

Quality attribute tree: tree representation



P1.1: 12 AGVs with an availability of 85% should handle 112 transports / hour

P1.2: 12 AGVs with an availability of 100% should handle 140 transports / hour

P2.1: The amount of communication, with 12 AGVs and a max. load of 140 transports/hour, does not exceed 60% of the 11Mbps communication channel.

S1.1: The system should scale up to 24 AGVs, for similar system requirements, without changing the software

F1.1: If an AGV blocks a path, other AGVs choose an alternative route (if it exists)

F1.2.: If an operator disables a node, AGVs choose an alternative route (if it exists)

F2.1: As long as a transport has not been picked, the system dynamically changes the transport to the most suitable AGV.

F2.2.: If an AGV drops an empty bin (add-back feederes, conveyer 1) and a full bin is available, this task is assigned to the AGV

Quality attribute tree: sheet representation

Difficulty to realize

Importance for the system

Quality attribute

System specific refinement

Measurable scenario

Level 2	*Level 3*	*Level 4*	*I*	*D*
Performance	**P1: The system should (at least) be able to handle normal capacity.**	**P1.1: A system with 12 AGVs and availability of 85% should be able to handle 80% of 140 transports per hour.**	**H**	**H**
		P1.2: A system with 12 AGVs and availability of 100% should be able to handle 140 transports per hour.	**H**	**H**
	P2: The amount of communication should not exceed available bandwidth of the communication channel.	**P2.1: The amount of communication, under all possible scenarios, should not exceed X percent of the bandwidth of the communication channel.**	**H**	**M**

Outline

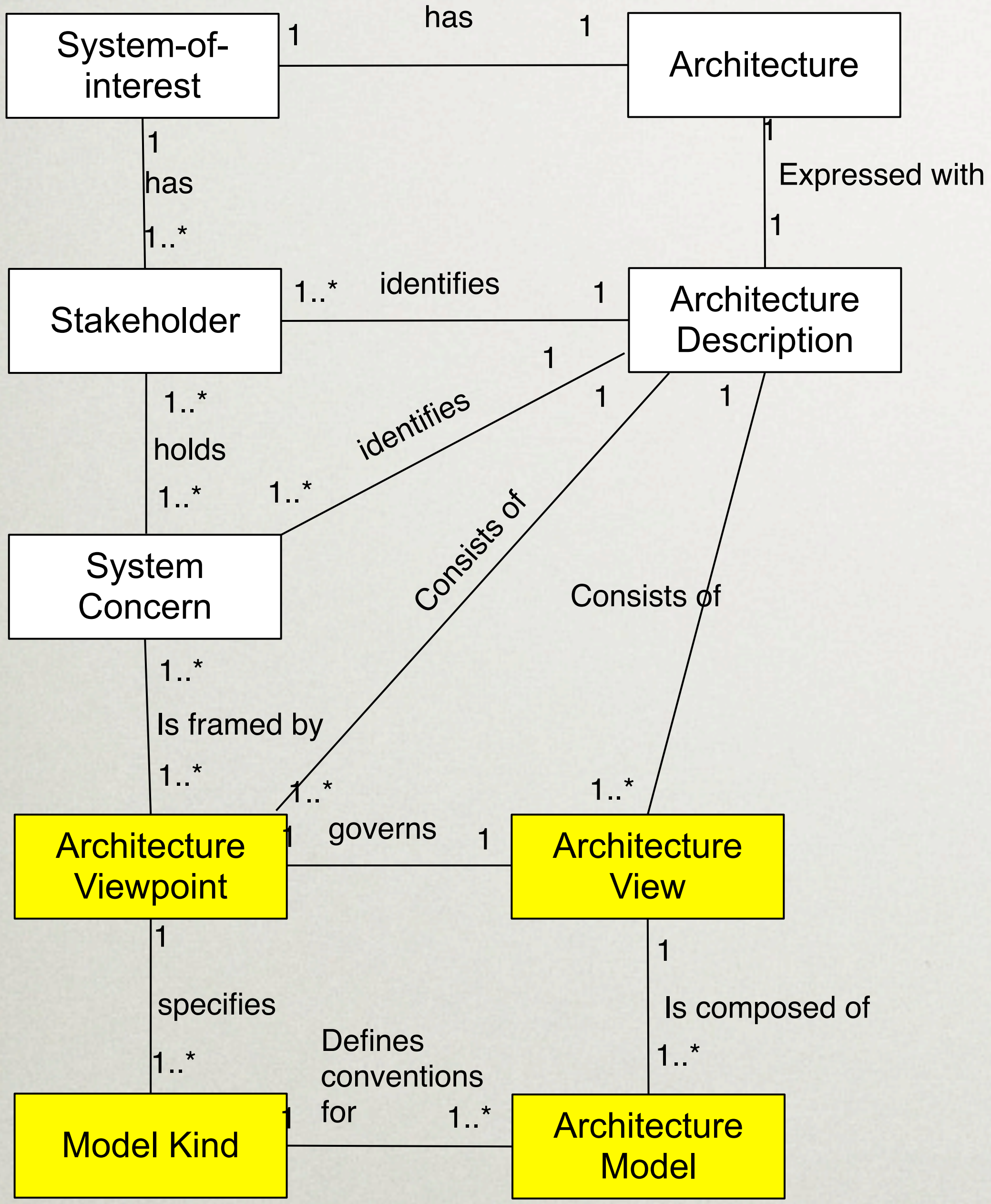
- What is software architecture?
- What drives software architecture?
- How to describe a software architecture?
  - ISO 42010: Viewpoints, views, models
  - What about security viewpoints?
  - Description principles!
- Conclusion

Describing an Architecture?

Exercise

- How does your organization describe software architectures?
  - Security architectures?
  - What is the description used for?

ISO 42010: Views



- **Viewpoint**:
  - establishing the conventions for the construction, interpretation and use of architecture views
- **View**:
  - a representation of the system from the perspective of architecture-related concerns
- **Modelkind:**
  - defining a type of model
- **Model**:
  - logical set of architectural elements and their relations

Example: Development viewpoint

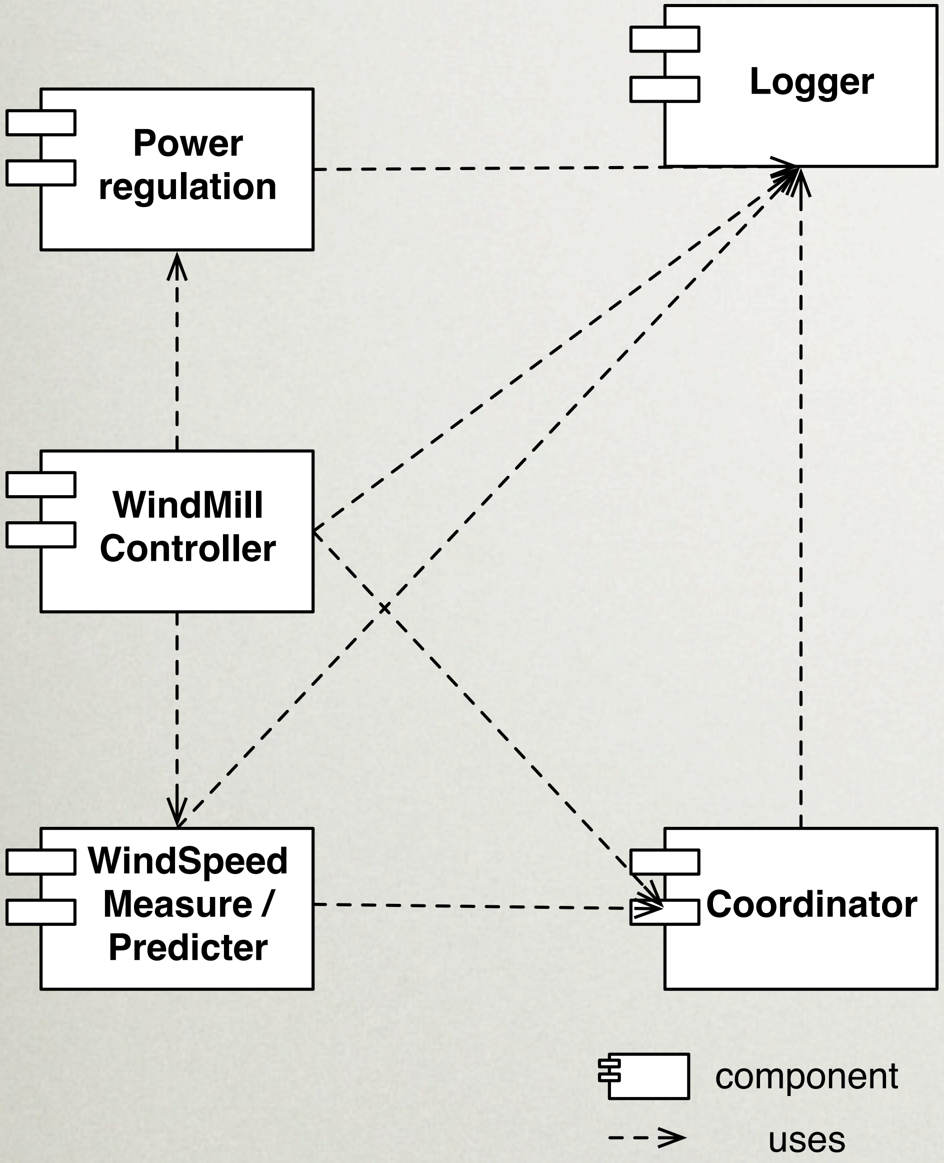
Def	Defines the main components and allocates these to the development teams and project plan
Stakeholders	Architects, development teams, project manager
Concerns	- which software modules must to be developed? - who will develop what? - what are the dependencies between the modules and the teams?
Model Kinds	Component catalog, Component dependency model, Allocation model, ...

View/model template

- **View**: the view contains the several models to cover the concerns of a viewpoint
- **Example view** template:

Primary presentation(s)	Models (Diagram / Table / ...) showing the elements and their relations.
Element catalog	A table based model describing each element.
Rationale	A description of the rationale underpinning the design shown in the primary presentation.

Example: Development view



Component dependency model

Element catalog

Element	Responsibility
Logger	This component is responsible to govern the logfiles and the synchronization of logfiles with the back-end. Currently, the logger component is based on Log4J.

Example: Development view

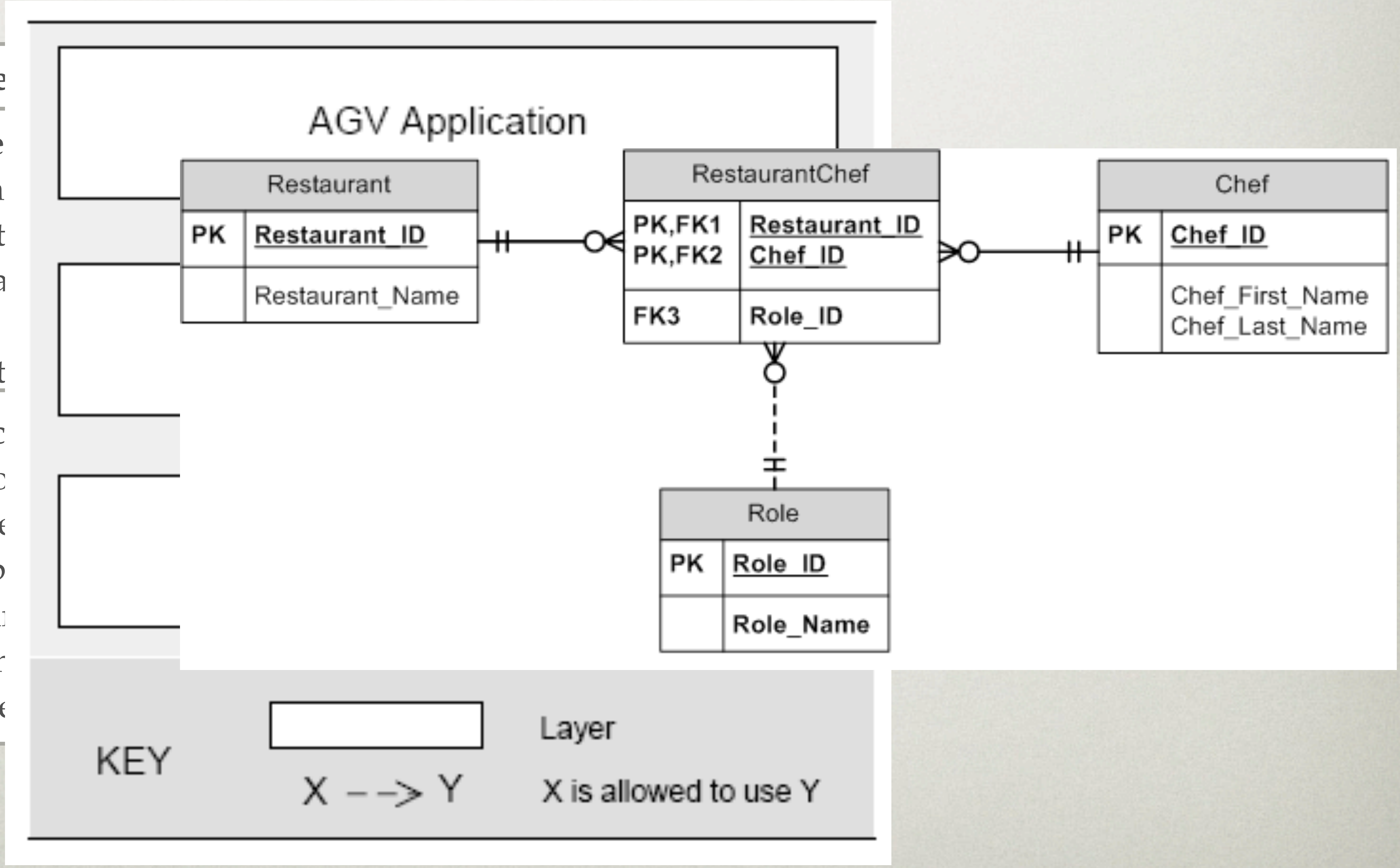
Component	Team	Milestone
Power regulation	Team Energy	deliver at M2
WindMill Controller	Team Controller	deliver at M4
Logging	Platform Team	deliver at M1
Wind Measure/Predicter	Wind specialists team	deliver at M1, update for M2
Coordinator	Team Controller	deliver at M4

Component allocation

Typical categories of models

- Structural models

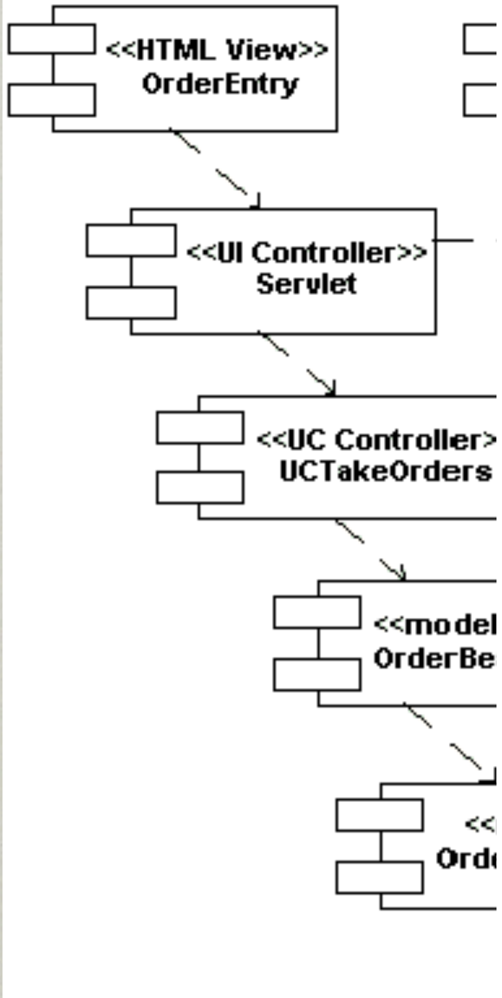
Package	Role
egemin. wms. inventorymanagement	This package contains th functionality t inventory mana It is divided in subpart
egemin. wms. planner	This package c functionality fc (the timely de goods to the p machines, stori goods as tl produce

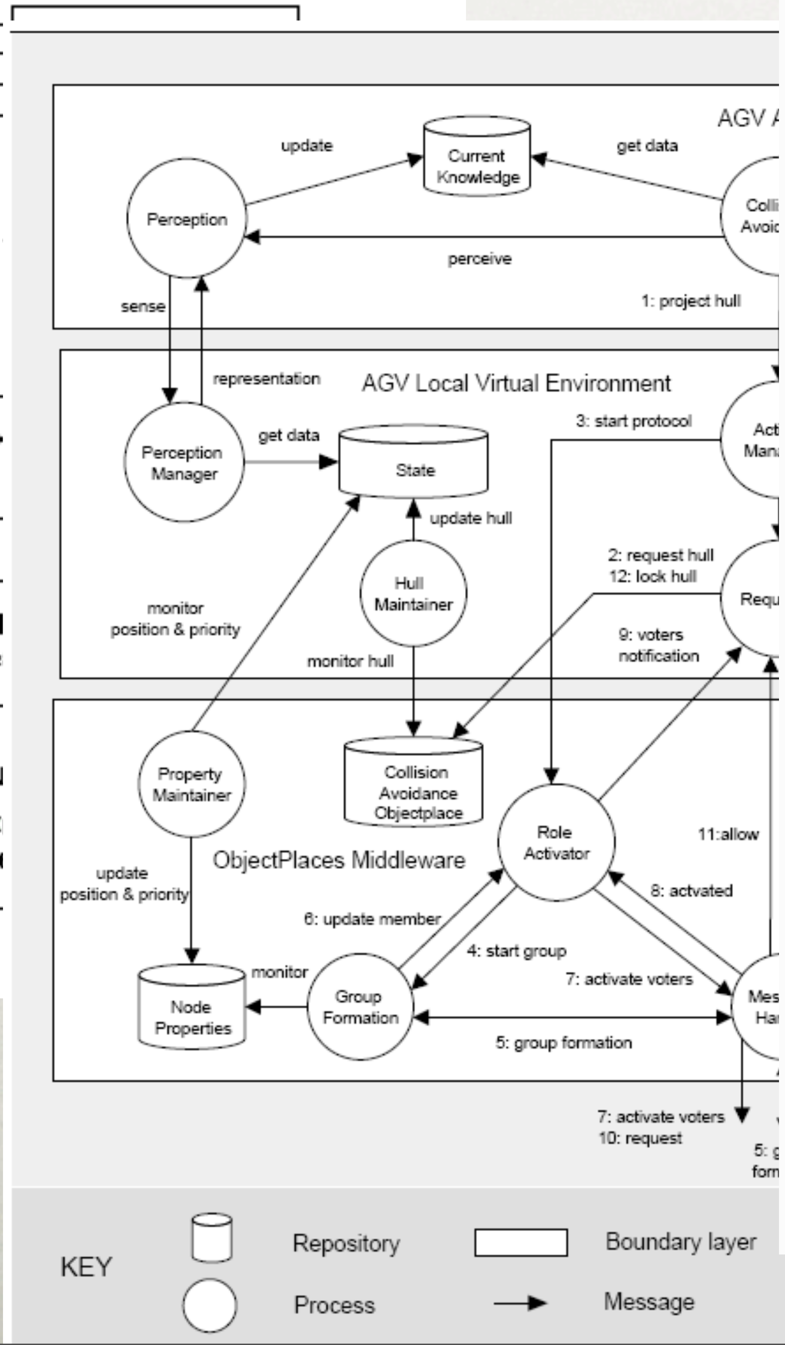


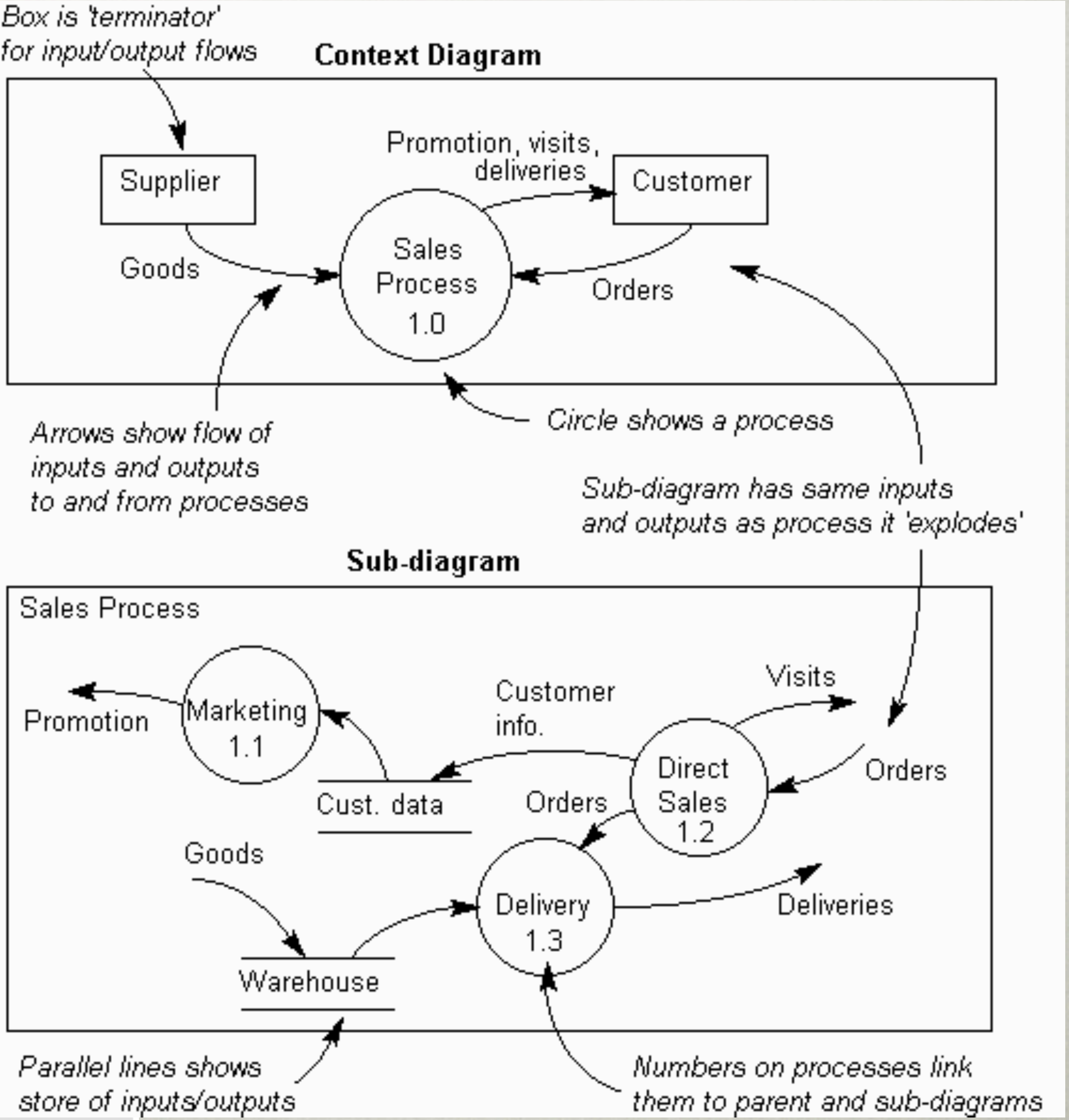
…

Typical categories of models

- Run-time models



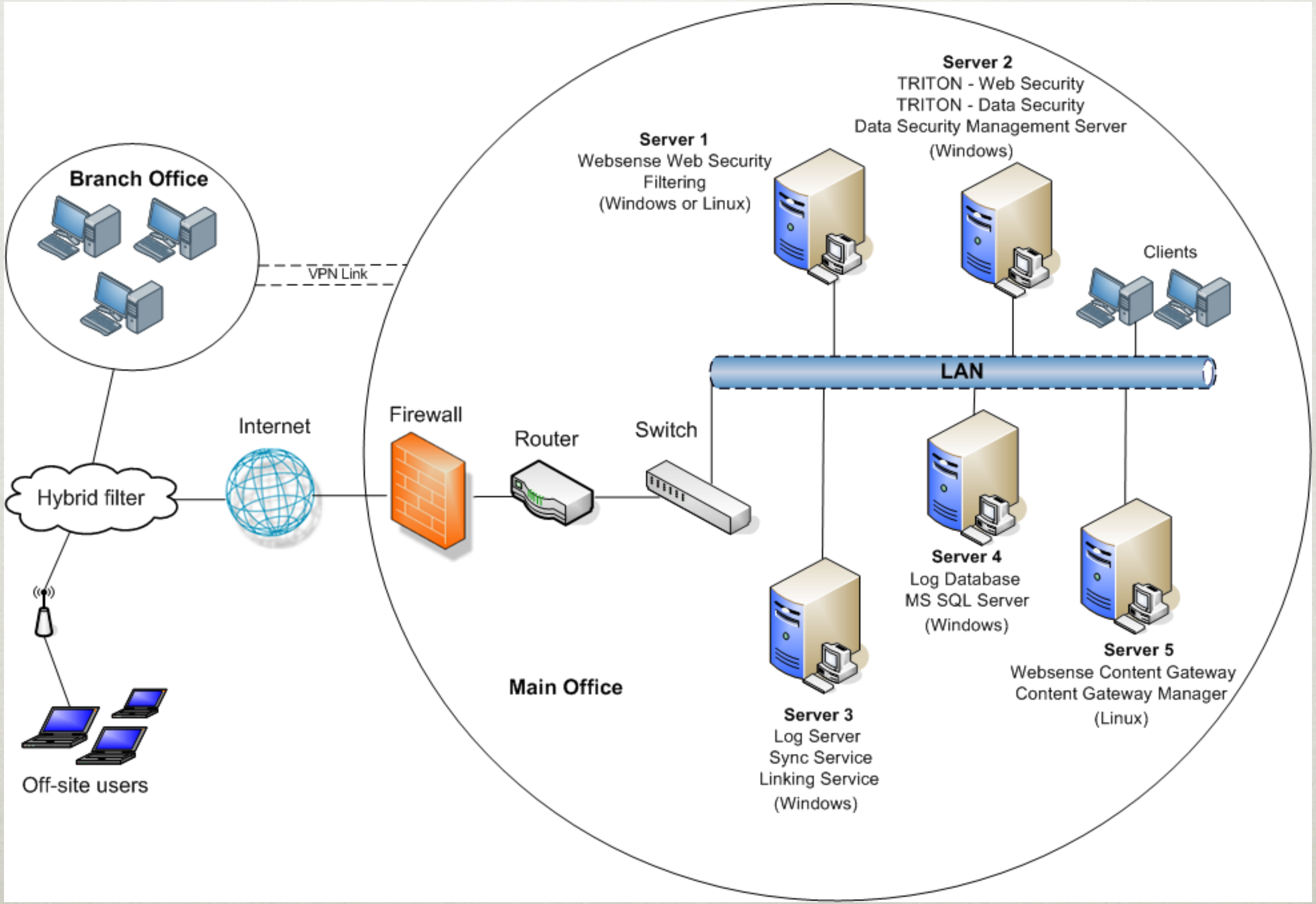




Box is 'terminator' for input/output flows
Context Diagram
Promotion, visits, deliveries
Supplier
Customer
Sales Process 1.0
Goods
Orders
Arrows show flow of inputs and outputs to and from processes
Circle shows a process
Sub-diagram has same inputs and outputs as process it 'explodes'
Sub-diagram
Sales Process
Customer info.
Visits
Promotion
Marketing 1.1
Direct Sales 1.2
Orders
Cust. data
Orders
Goods
Delivery 1.3
Deliveries
Warehouse
Parallel lines shows store of inputs/outputs
Numbers on processes link them to parent and sub-diagrams

Typical categories of models

- Allocation models



In General: Which viewpoints/model kinds?

- **NO fixed set** of viewpoints is suited for each situation
- Guidelines:
  - Choose the best views for each situation,
  - Who are the **stakeholders**, what are their **concerns** and how will they **use the description**?
    - Education, analysis, development
  - What model kinds are known in the **domain?**

Outline

- What is software architecture?
- What drives software architecture?
- How to describe a software architecture?
  - ISO 42010: Viewpoints, views, models
  - What about security viewpoints?
  - Principles of sound documentation
- Conclusion

What about security viewpoints?

- Adds its own **single-purpose components** to an architecture
  - *Localized impact,* single-purpose component
  - e.g. add an authentication component, VPN-component, etc.
- ALSO: seen as a quality of a system, often requiring **dedicated models and views**
  - Security concern with *broad impact*
  - Multiple viewpoints / modelkinds can be used to express security concerns
  - Useful for thorough security-analysis

Example: Trust viewpoint

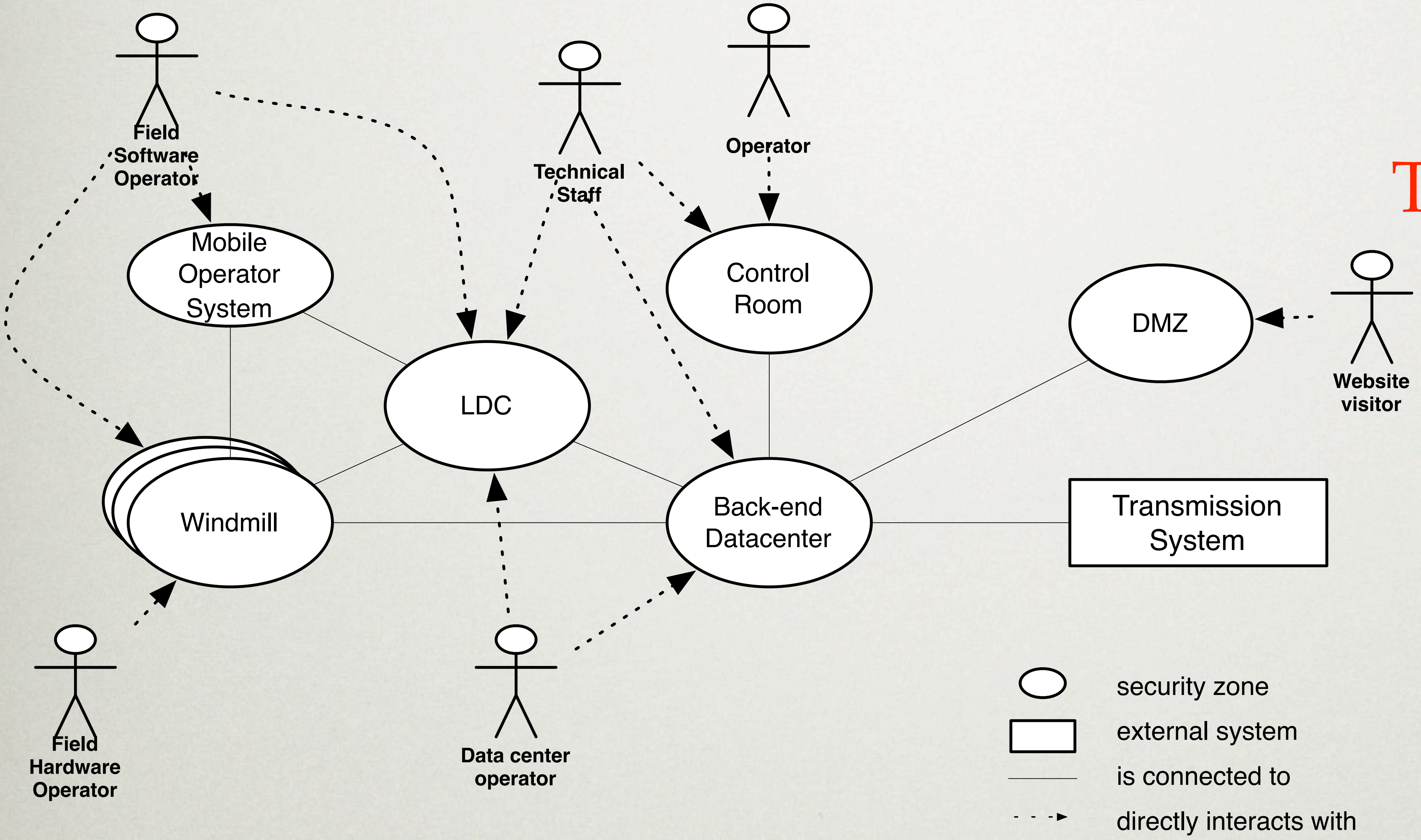
Def	Defines trust level for subsystem for determining control privileges and data validation requirements.
Stakeholders	(Security) architects, operations, management, network team, developers.
Concerns	- Level of trust in a subsystem - Interaction of user groups with subsystem, - Data validation needs - Possible attack paths for a subsystem
Model Kinds	Deployment model, Context model, Information flow diagram, Technology model, Trust zone model

Example: Trust View

- **View**: the view contains the several models to cover the concerns of a viewpoint

- 3 example models
  - Trust-zone model
  - Information flow model
  - Attack surface model

Example: Trust view

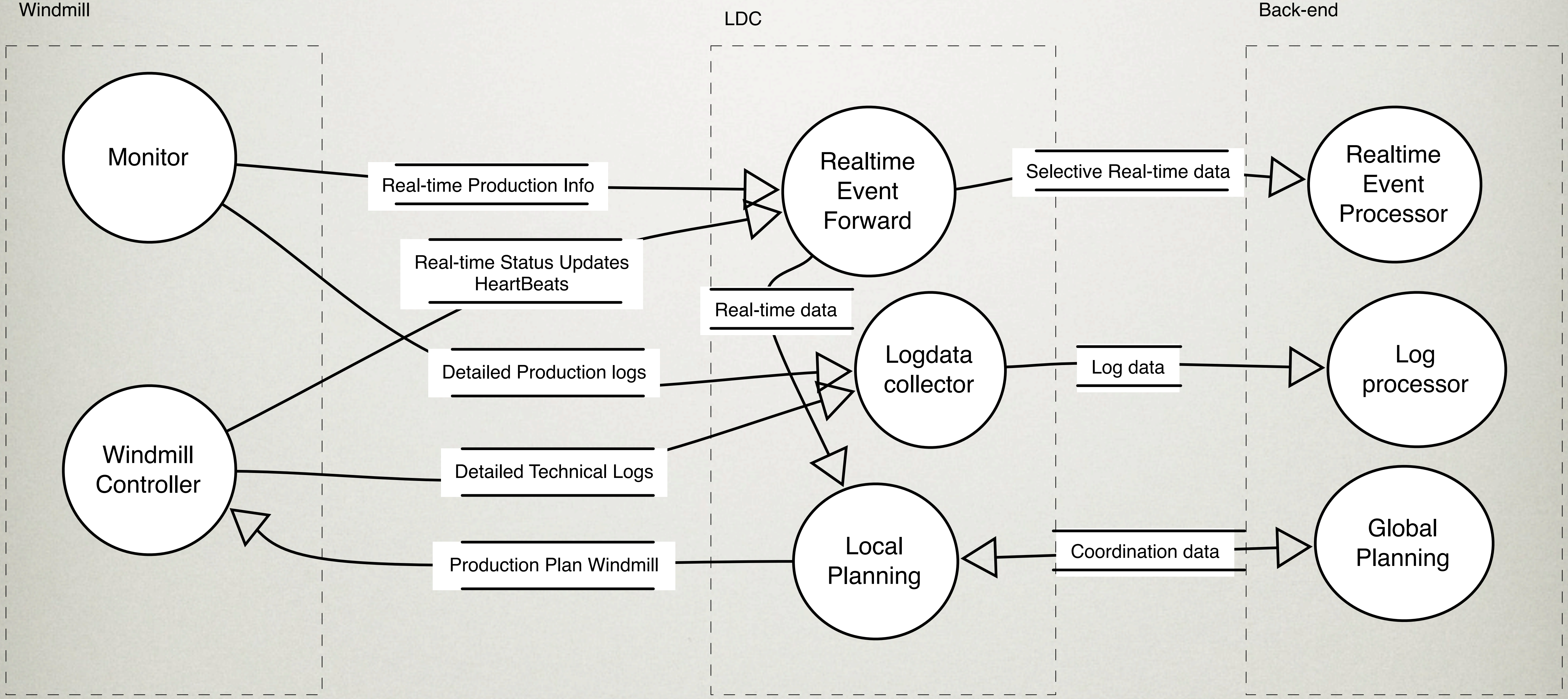


Trust zone model

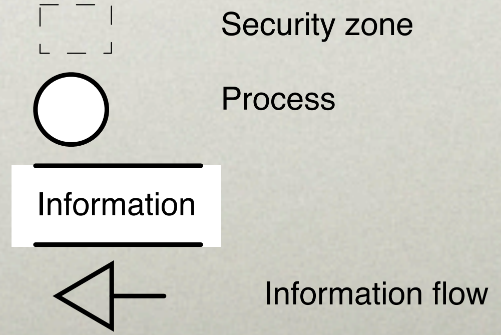
Element	Responsibility	Level
Back-end Datacenter	The Back-end datacenter is responsible for processing and storing all real-time data from the different devices, for controlling these devices and for hosting the applications that are used by	10

Trust zone catalog

Example: Trust view



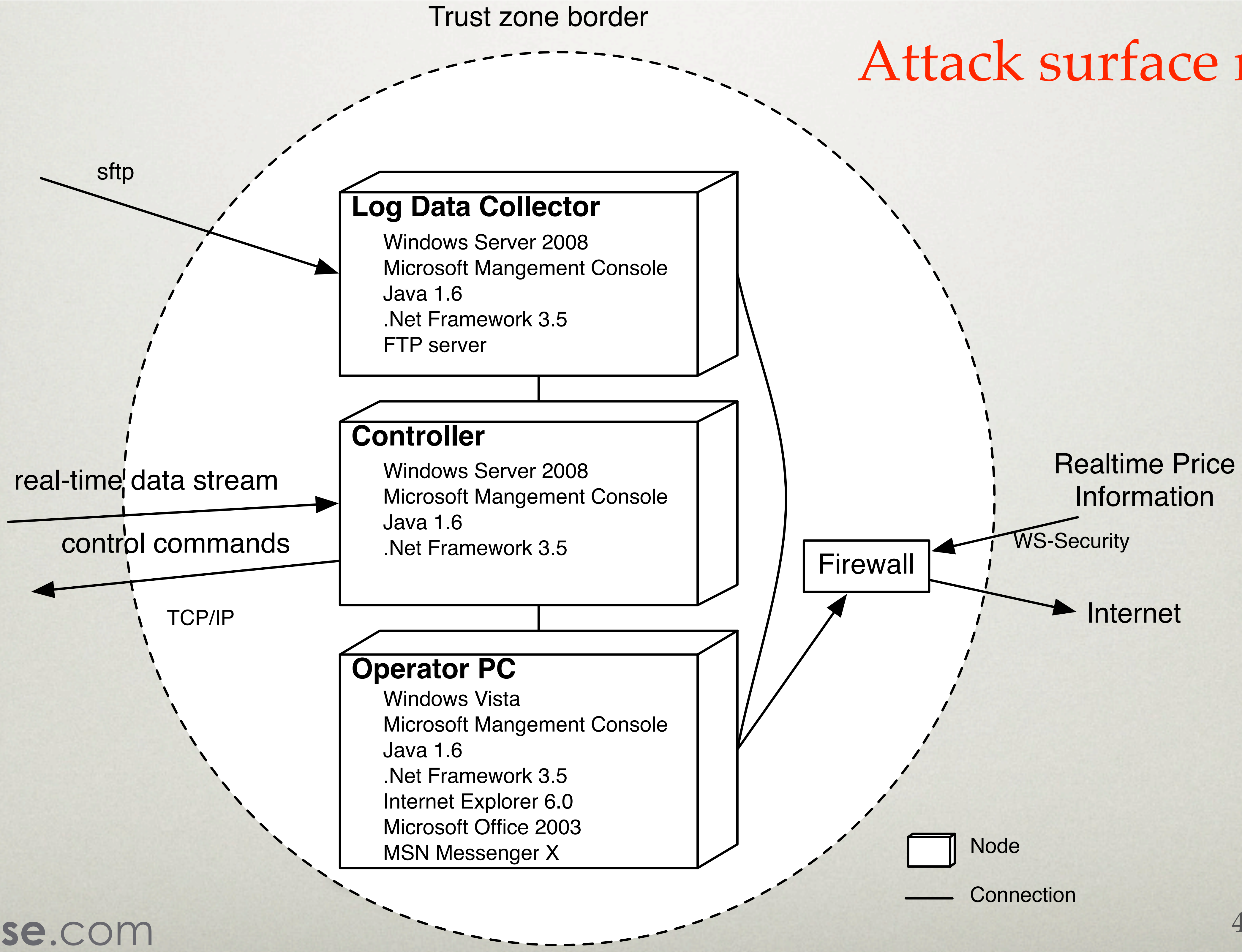
Data flow model



archiwise.com

Example: Trust view

Attack surface model



Trust zone border

sftp

Log Data Collector

Windows Server 2008
Microsoft Mangement Console
Java 1.6
.Net Framework 3.5
FTP server

Controller

Windows Server 2008
Microsoft Mangement Console
Java 1.6
.Net Framework 3.5

Operator PC

Windows Vista
Microsoft Mangement Console
Java 1.6
.Net Framework 3.5
Internet Explorer 6.0
Microsoft Office 2003
MSN Messenger X

real-time data stream

control commands

TCP/IP

Firewall

Realtime Price Information

WS-Security

Internet

Node

Connection

archiwise.com

Other starting points for security models

- Use typical security techniques as inspiration
  - e.g. Talk of John Stevens yesterday on “Threat modelling and architectural risk analysis”
  - Models as input of an analysis (e.g. component, deployment model)
  - Model as output containing decisions (e.g. catalog of security elements and their function).

Other starting points for security models

- Events & failure modes catalog
  - Safe default actions for certain events and failures
  - Log / warn strategy
- Data
  - Data Sensitivity and classification catalog
  - Data lifecycle models
  - Data flow & dissemination models

Other starting points for security models

- Application Architectural models
  - List of security elements
  - List of security standards (+ versions) to follow during implementation
  - Deployment/Technology models
    - Trust zones
    - Attack surface models
  - ...

Views, Model

Exercise

- From your experience, can you give examples of
  - Security viewpoints and model kinds?
  - What concerns are addressed by these viewpoints and modelkinds? What stakeholders?

Outline

- What is software architecture?
- What drives software architecture?
- How to describe a software architecture?
    - ISO 42010: Viewpoints, views, models
    - What about security viewpoints?
    - ➡ Principles of sound documentation
- Conclusion

Principles for sound architecture descriptions

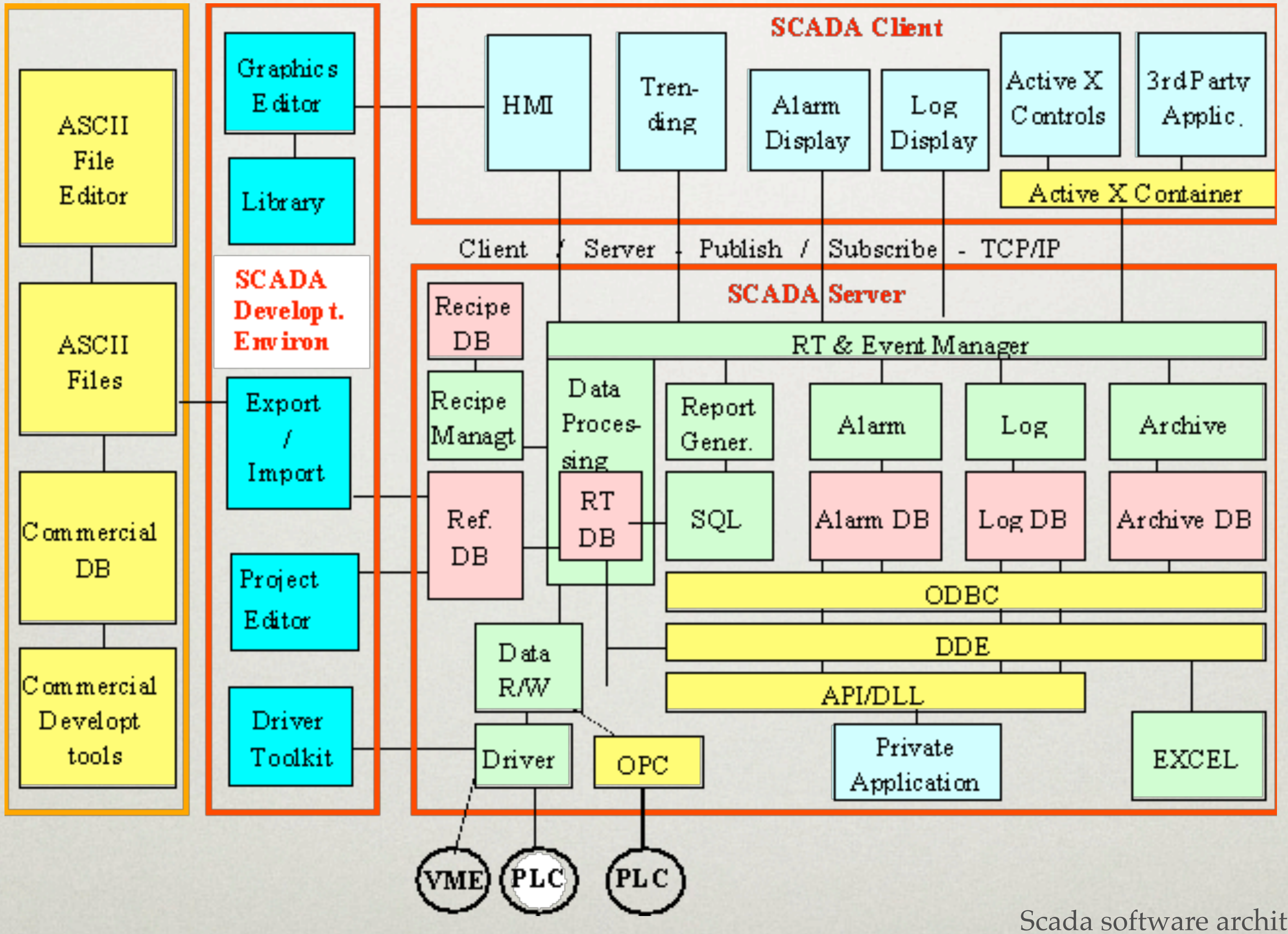
- Keep the target audience in mind
- Avoid ambiguity
- Record your rationale

Principle 1: Keep the target audience in mind

- No target audience = No description
- Write from the reader's point of view, write about what they want to know!
- Provided guidance for the readers to find their way in the documentation

Principle 2: Avoid ambiguity

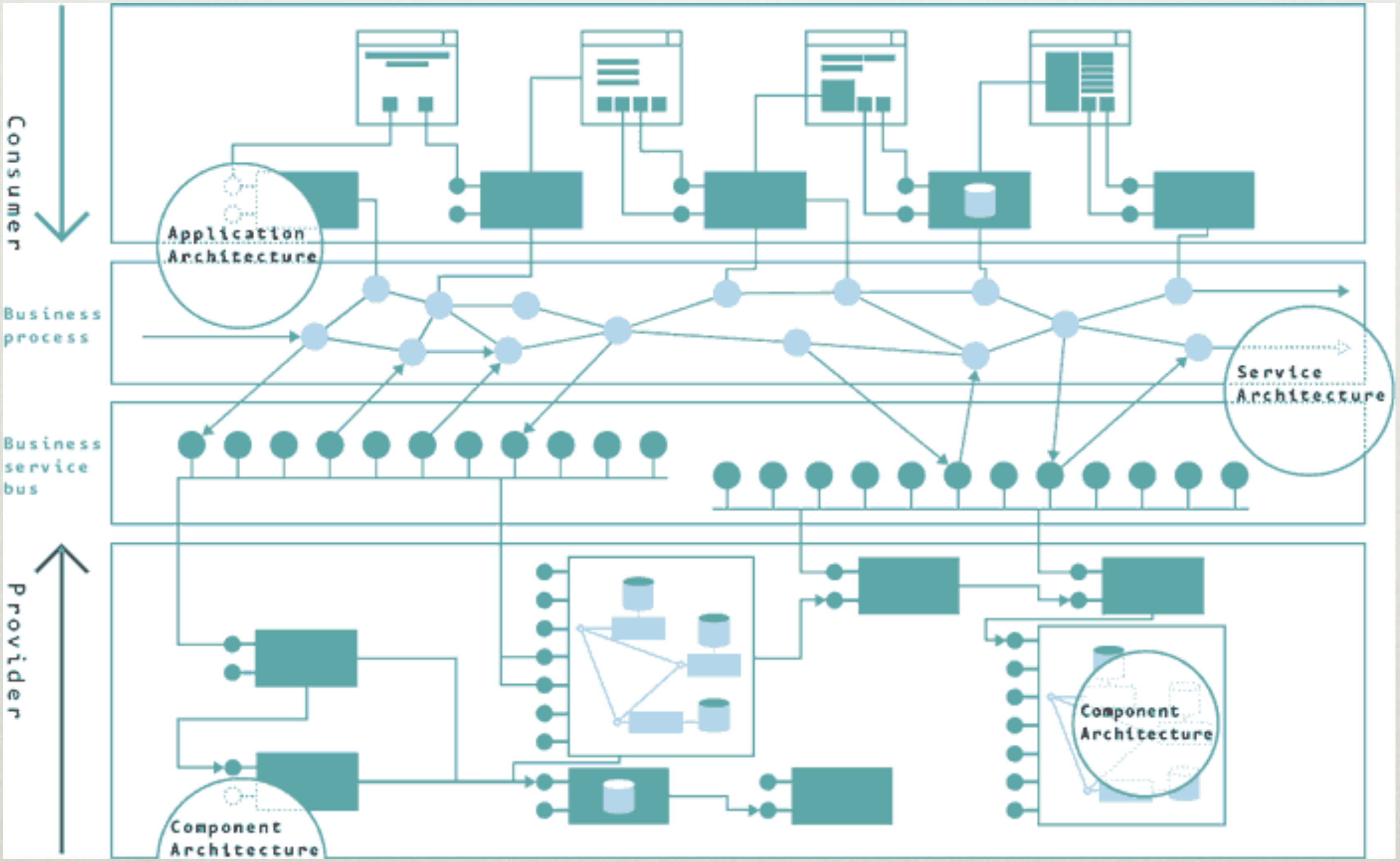
- Always provide a key!



Scada software architecture

Principle 2: Avoid ambiguity

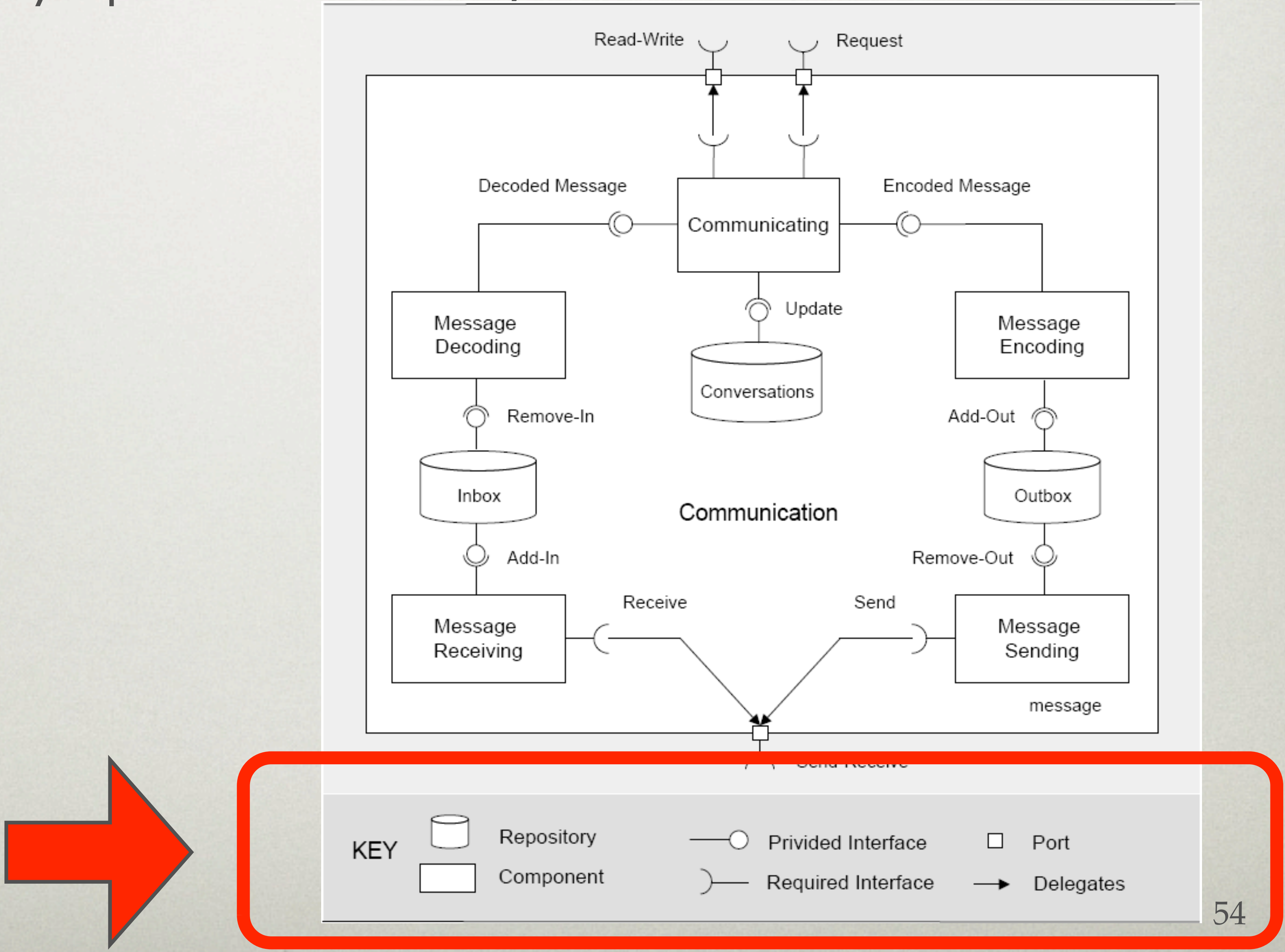
- Always provide a key!



A Service-Oriented Architecture as told in the MS Architecture Journal (January, 2004)

Principle 2: Avoid ambiguity

- Always provide a key!



Principle 2: Avoid ambiguity

- Always provide a key!
  - Precisely defined notations help avoid ambiguity
  - Use standard notations if possible

Principle 3: Record your rationale

- Explain why you made certain design decisions
- Requires discipline but yields in the long run
- Record relevant rejected alternative designs

Outline

- What is software/security architecture?
- What drives software architecture and how does security fit in?
- How to describe a (security) software architecture?
- Conclusion

Conclusion

- **Security** is one of many quality concerns
- Architects need to compare apples to oranges
  - Trade-offs between security, performance, flexibility, etc.

Conclusion

- Software architecture offers principles and practices to deal with multiple concerns
- Stakeholder-orientation is key to
  - agree on **measurable scenarios** for quality concerns
  - decide on **quality priorities**
  - deliver a comprehensible architecture description comprising **multiple views and models**

References

- Software System Architecture (Rozanski & Woods)
- Software Architecture in Practice (Bass et al.)
- Documenting Software Architecture (Clements et al.)
- Evaluating Software Architecture (Clements et al.)
- Enterprise Security Architecture (Sherwood, Clark and Lynas)
- TOGAF 9 (The Open Group)

archiwise.com