Spring 2023

UP: 4+1 View

Seonah Lee

Gyeongsang National University

Kruchten, Philippe B. "The 4+ 1 view model of architecture." IEEE software 12, no. 6 (1995): 42-50.

목 차

- **Architecture Views**
- **4+1 View**
  - Logical View
  - Process View
  - Development View
  - Physical View
  - + Use Cases/Scenarios

Architecture Views

- Questions
- Architecture Views

Questions

- What views or perspectives are useful when designing and documenting a system’s architecture?

- What notations should be used for describing architectural models?

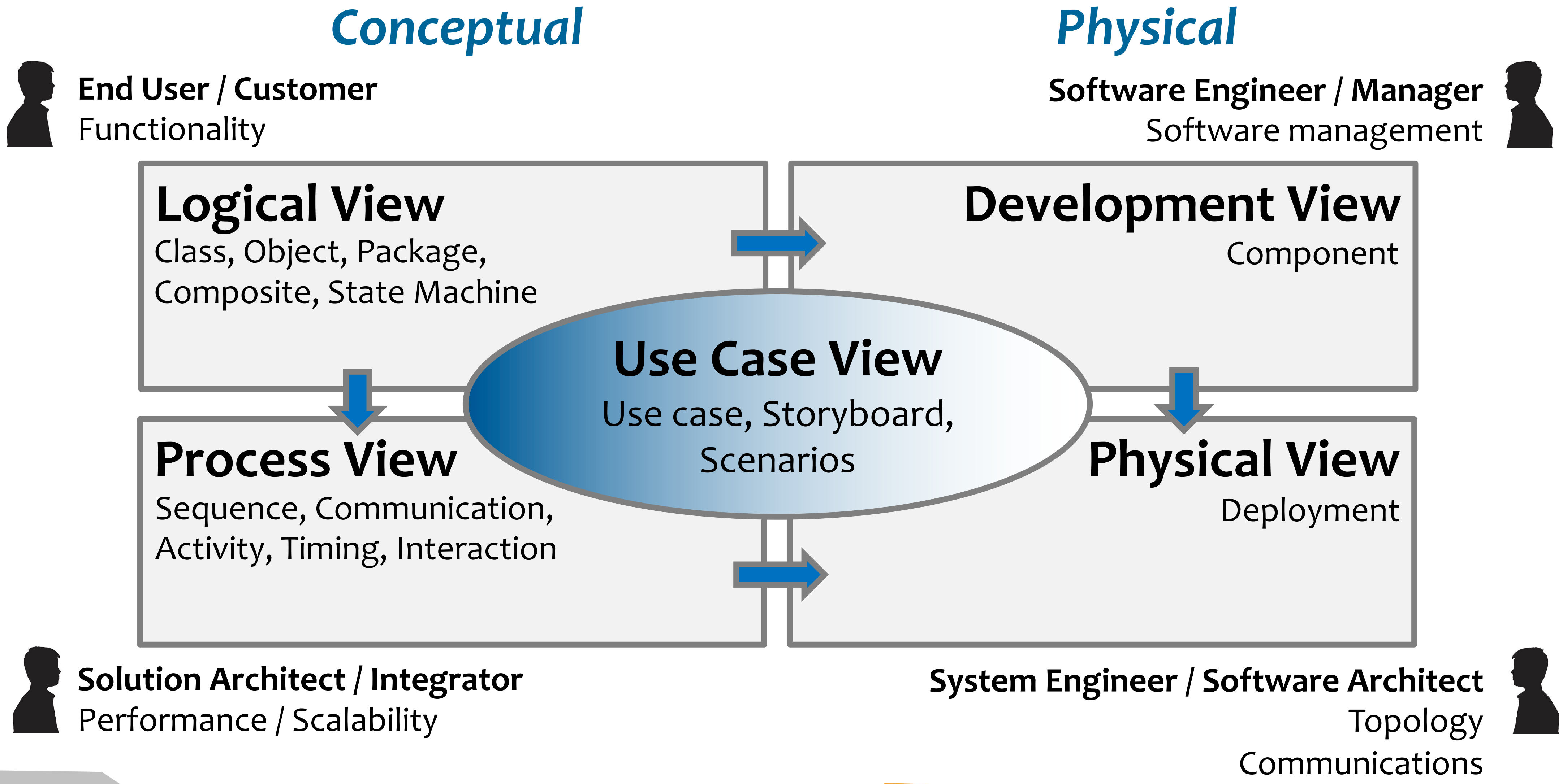
Architectural Views

- **Each architectural model only shows one view or perspective of the system.**
  - It might show how a system is decomposed into modules,
  - how the run-time processes interact or
  - the different ways in which system components are distributed across a network.
  - For both design and documentation, you usually need to present multiple views of the software architecture.

4+1 View

- Logical View
- Process View
- Development View
- Physical View
- + Use Cases/Scenarios

4+1 View



Conceptual

Physical

End User / Customer
Functionality

Software Engineer / Manager
Software management

Logical View
Class, Object, Package, Composite, State Machine

Development View
Component

Use Case View
Use case, Storyboard, Scenarios

Process View
Sequence, Communication, Activity, Timing, Interaction

Physical View
Deployment

Solution Architect / Integrator
Performance / Scalability

System Engineer / Software Architect
Topology
Communications

http://rajmanickam.files.wordpress.com/2010/08/41-view.jpg

Logical View

- A logical view, which shows the key abstractions in the system as objects or object classes.

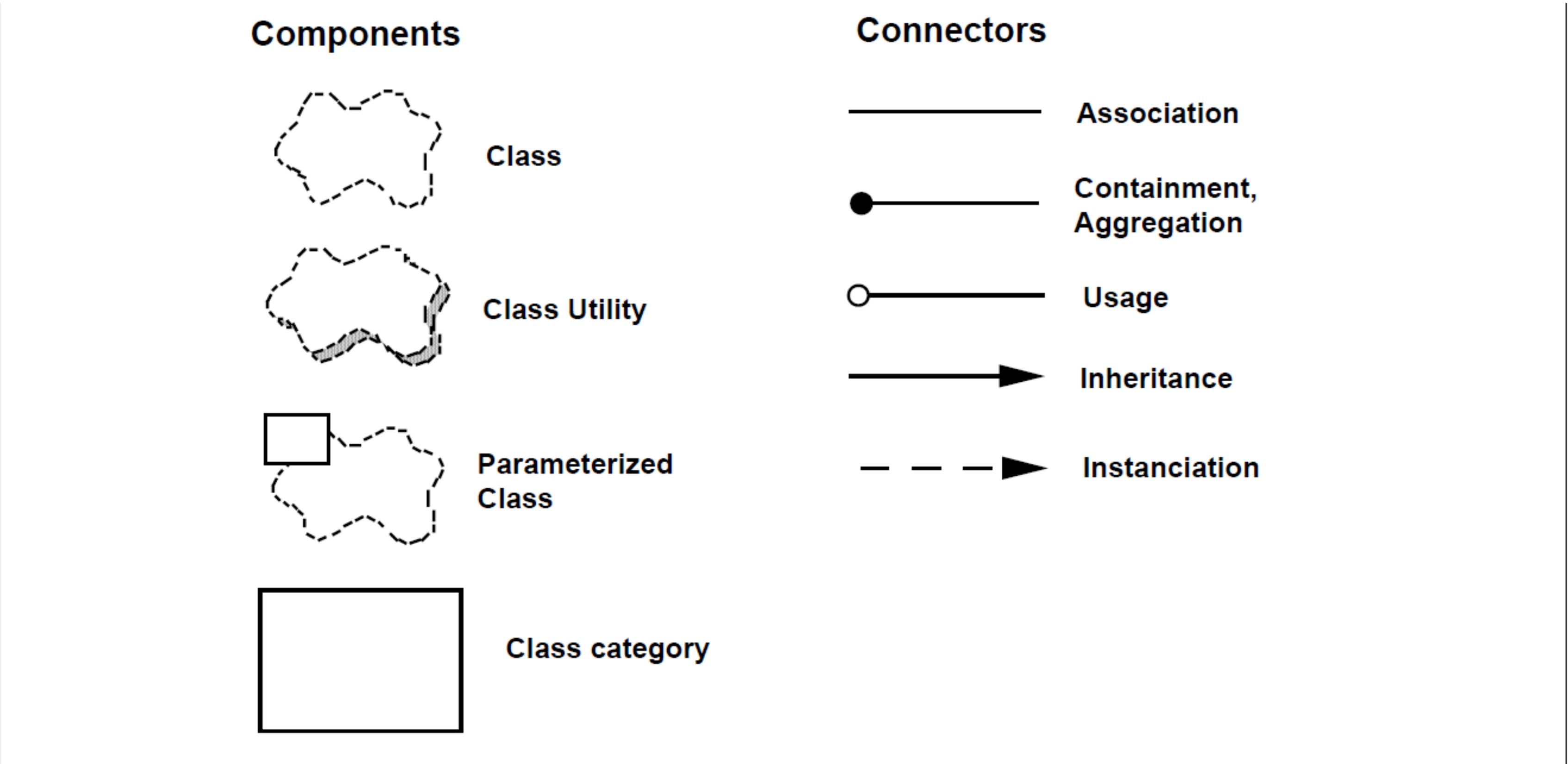


Figure 2 — Notation for the logical blueprint

Logical View

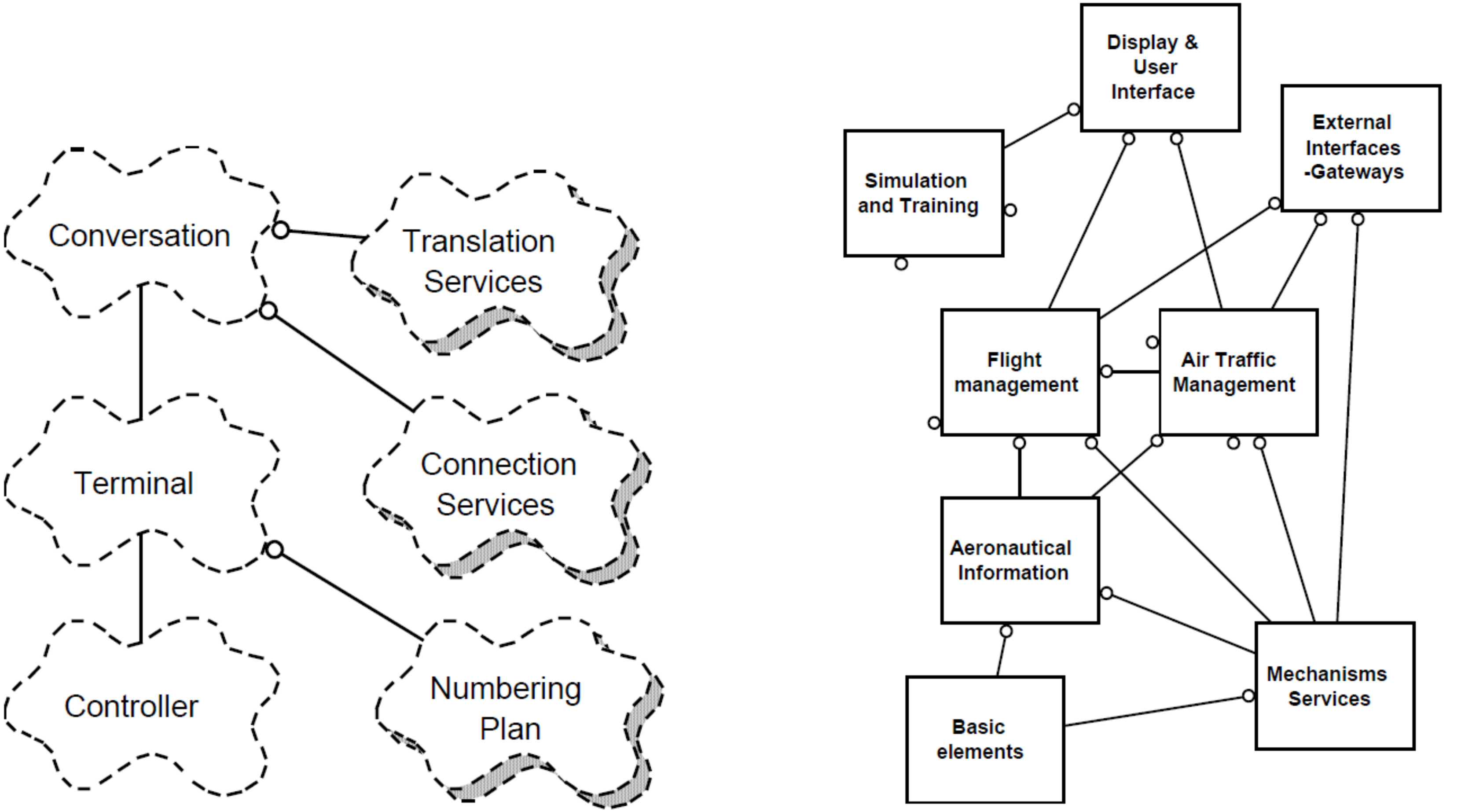


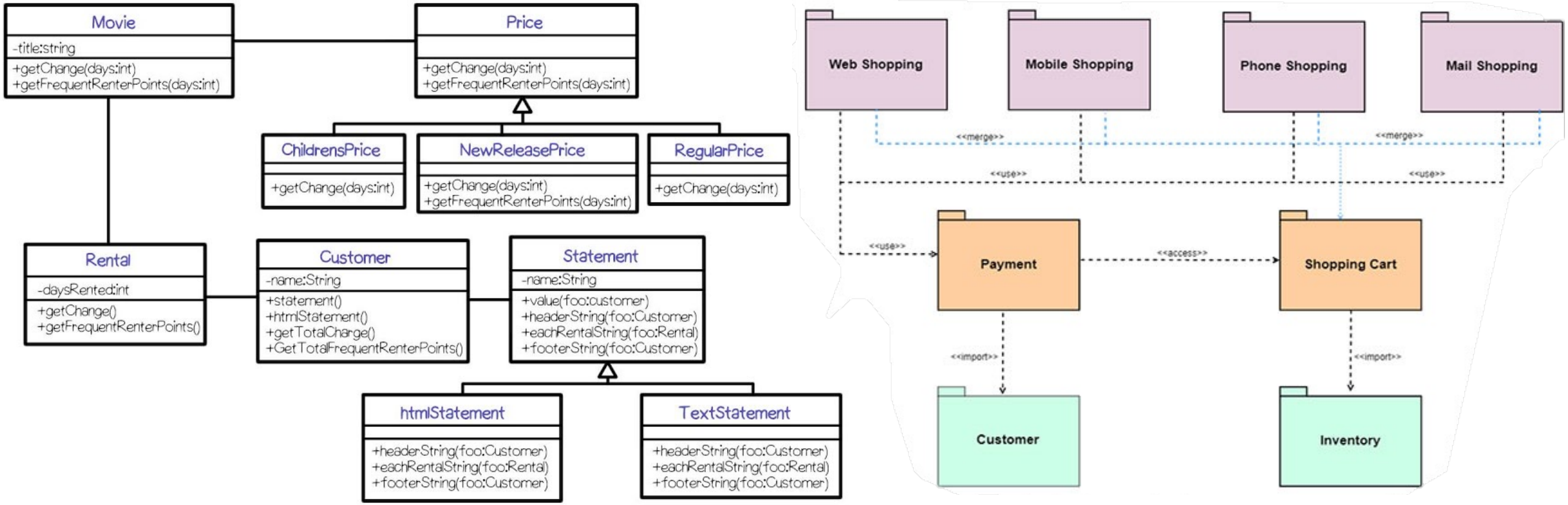
Figure 3— a. Logical blueprint for the Télic PABX . b. Blueprint for an Air Traffic Control System

Logical View

- Stakeholders
  - End users, acquirers, developers, and maintainers of the system
- Concerns
  - Show how the functions are decomposed in terms of classes
  - Assure that the design addresses the intended purpose of the system
- Focus – System Functionality
  - Structural elements
  - Key abstractions and mechanisms
  - Separation of concerns
  - Distribution of responsibilities

Logical View

- **Relevant UML diagrams**
  - Class, Object, Package, Composite, State Machine



Process View

- A process view, which shows how, at run-time, the system is composed of interacting processes.

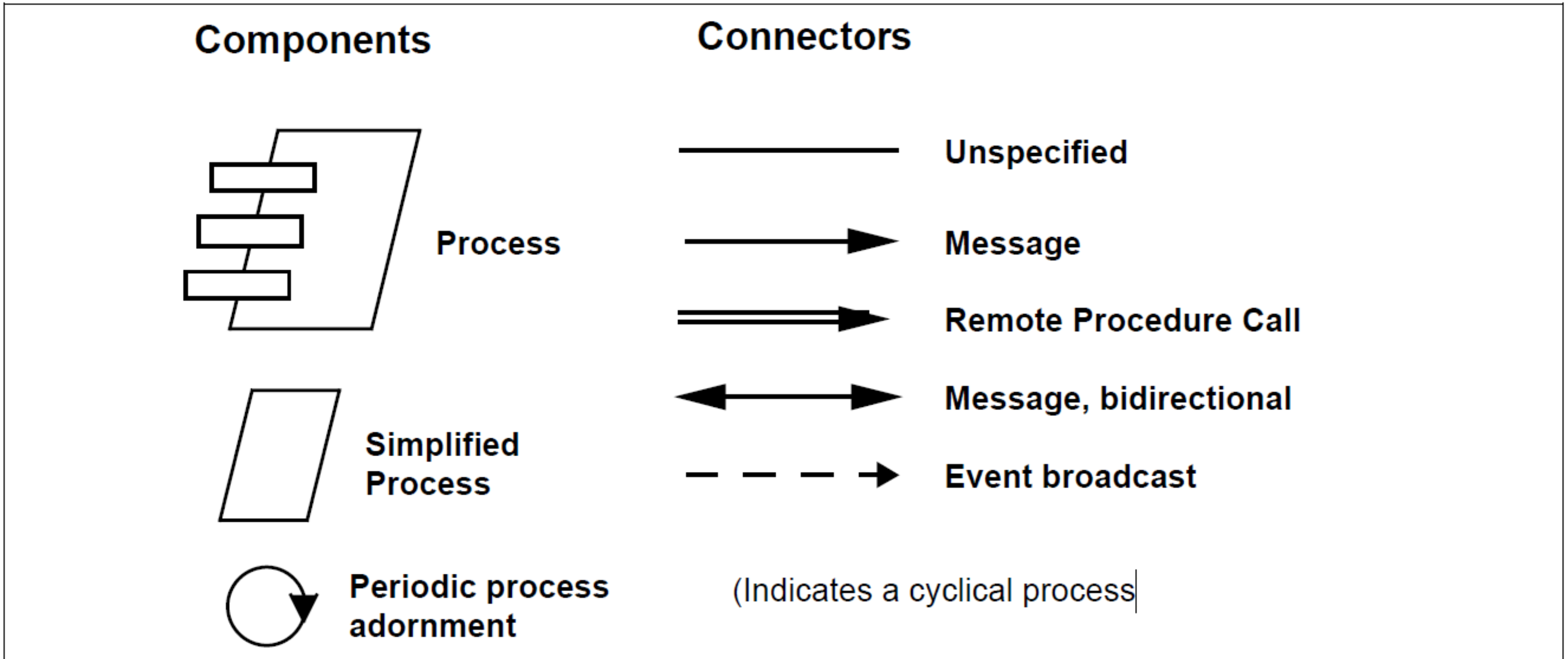


Figure 4 — Notation for the Process blueprint

Process View

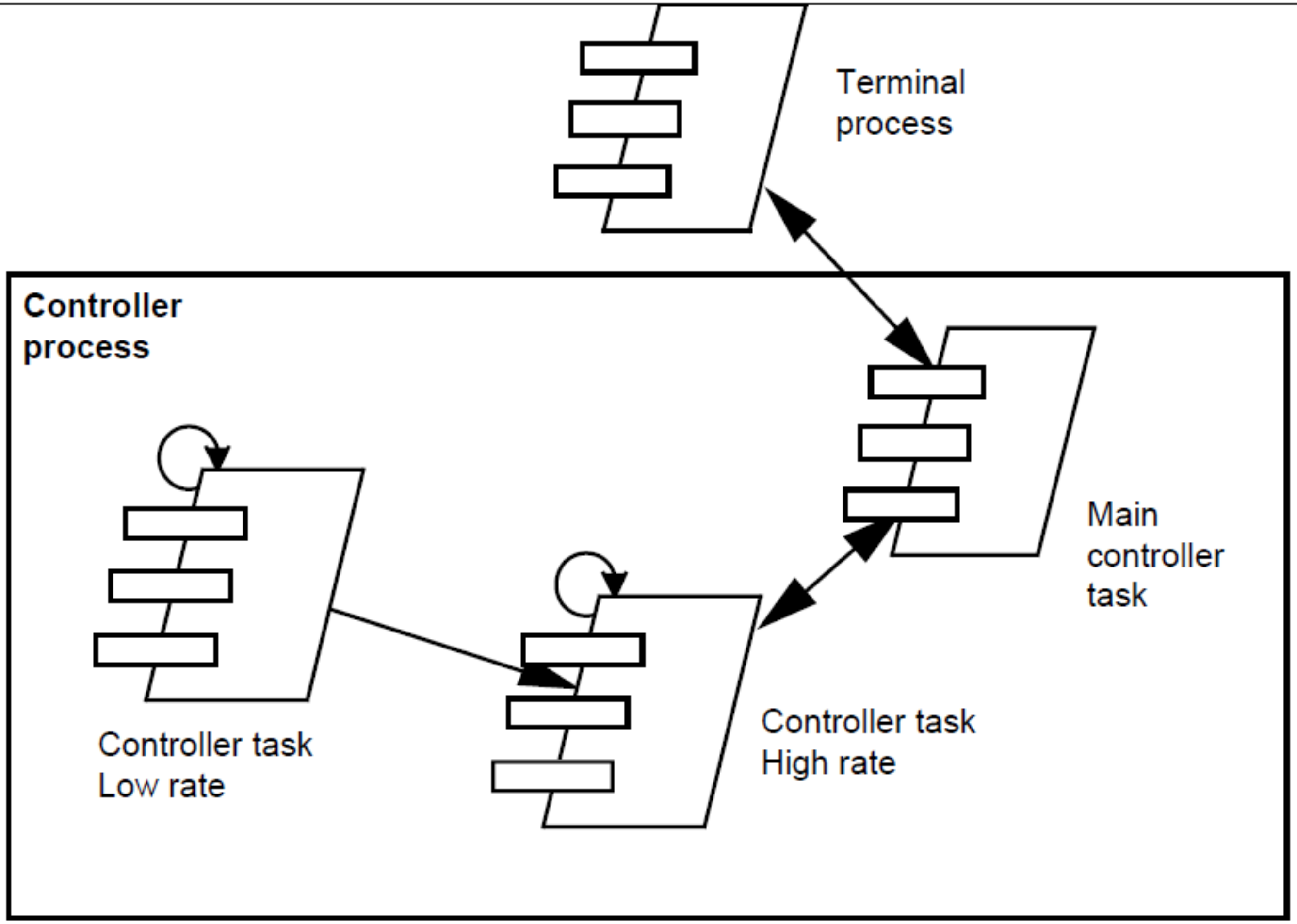


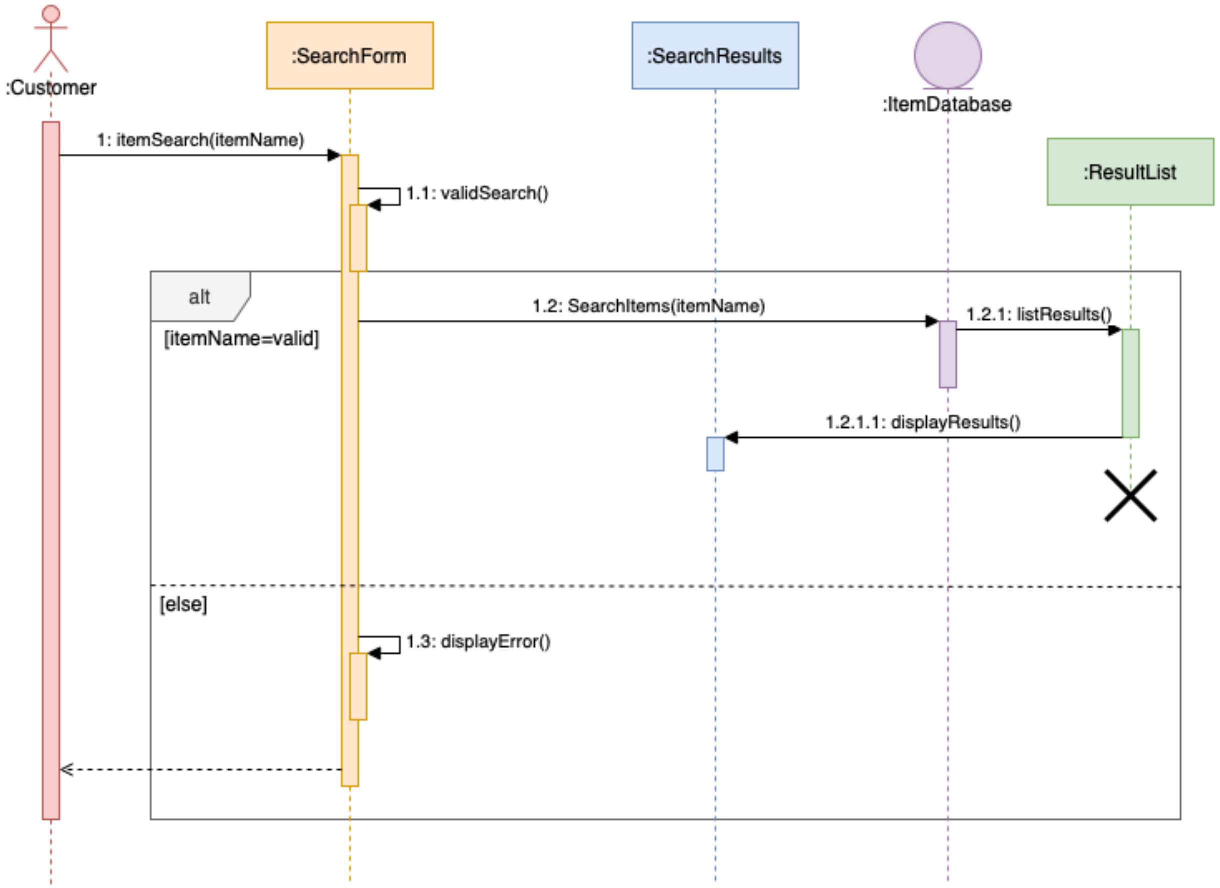
Figure 5 — Process blueprint for the Télic PABX (partial)

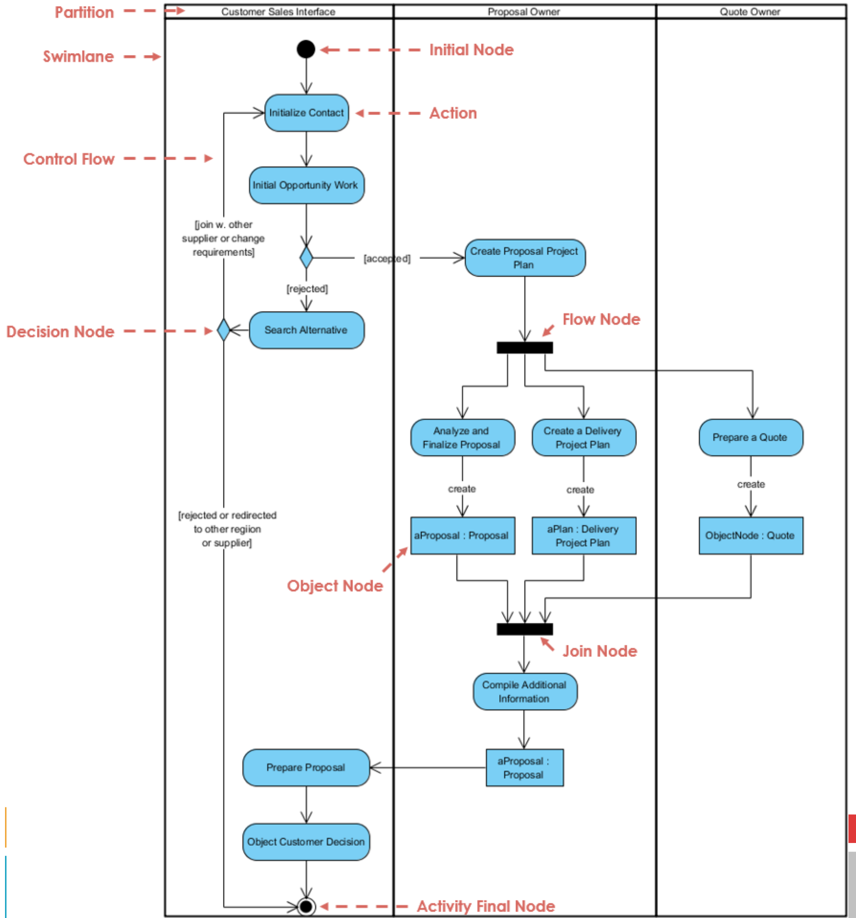
Process View

- **Stakeholders**
  - System integrators, acquirers, developers, and maintainers
- **Concerns**
  - Represent design solutions to nonfunctional requirements such as performance, availability, and fault tolerance
  - Assure that the design will satisfy these nonfunctional requirements
- **Focus – System Tasks**
  - A process is a group of tasks that form an executable unit
  - A software system is partitioned into sets of tasks
  - Each task is a thread of control that executes with collaboration among different structural elements (from the Logical View)

Process View

- **Relevant UML diagrams**
  - Sequence, Communication, Activity, Timing, Interaction





Development View

- A development view, which shows how the software is decomposed for development.

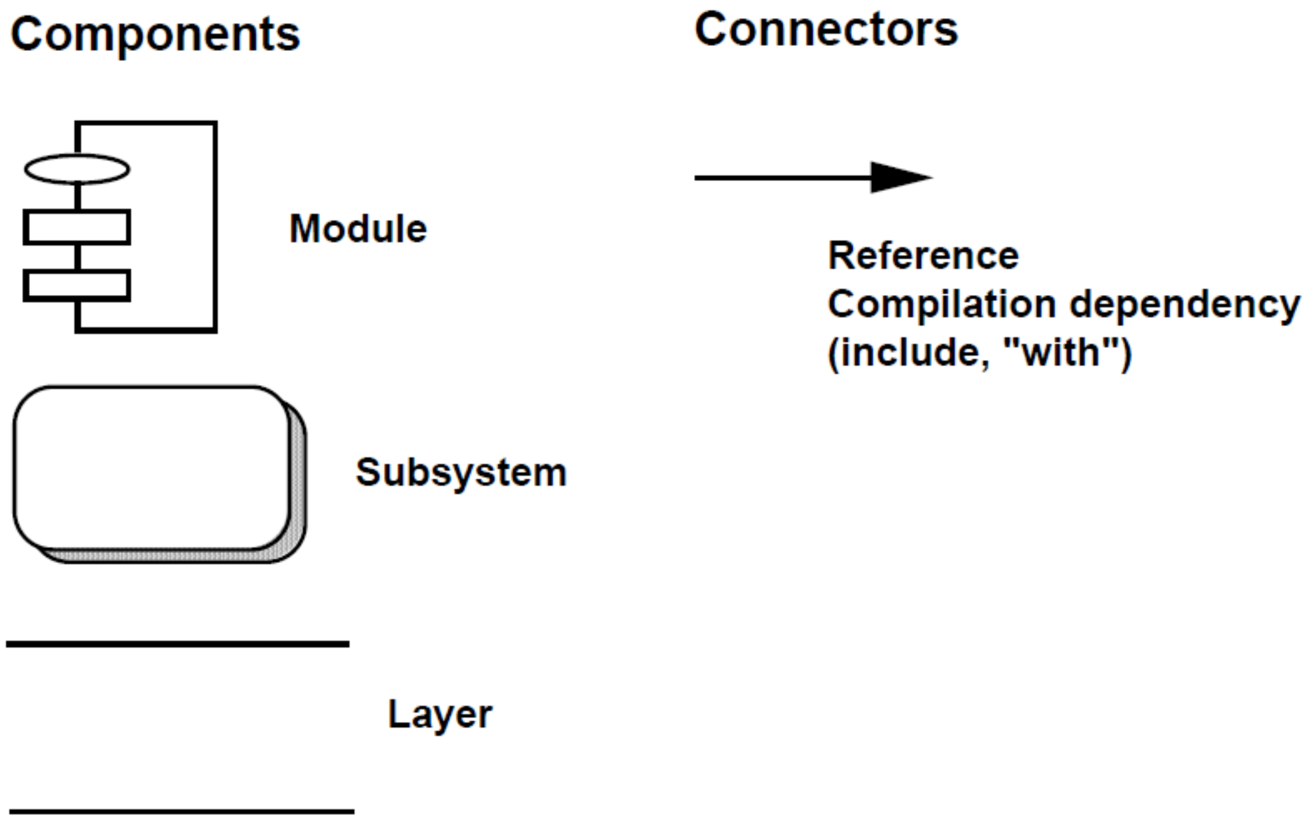
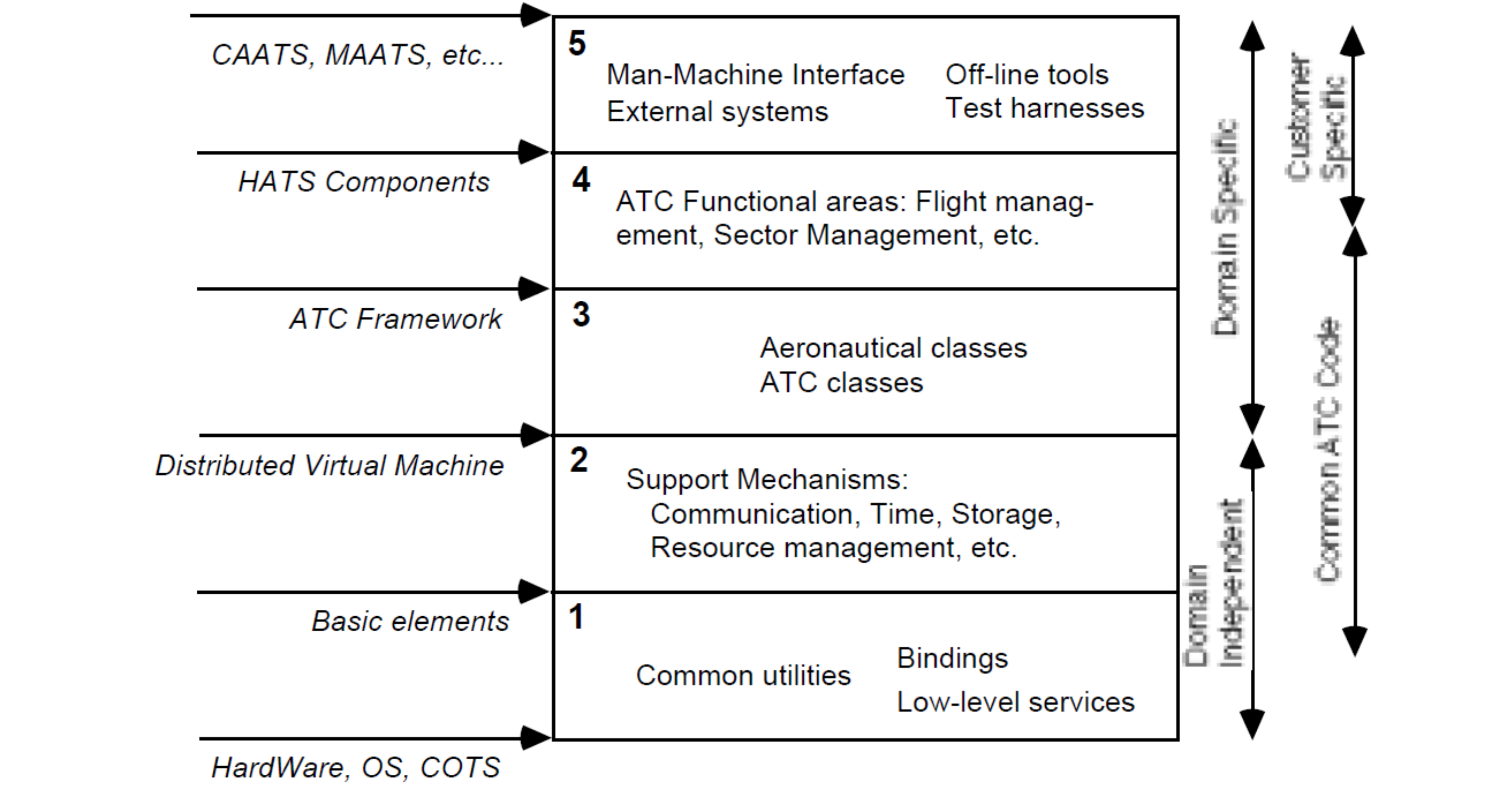


Figure 5 — Notation for the Development blueprint

Development View



CAATS, MAATS, etc...

5

Man-Machine Interface

External systems

Off-line tools

Test harnesses

HATS Components

4

ATC Functional areas: Flight management, Sector Management, etc.

ATC Framework

3

Aeronautical classes

ATC classes

Distributed Virtual Machine

2

Support Mechanisms:

Communication, Time, Storage, Resource management, etc.

Basic elements

1

Common utilities

Bindings

Low-level services

HardWare, OS, COTS

Domain Specific

Domain Independent

Customer Specific

Common ATC Code

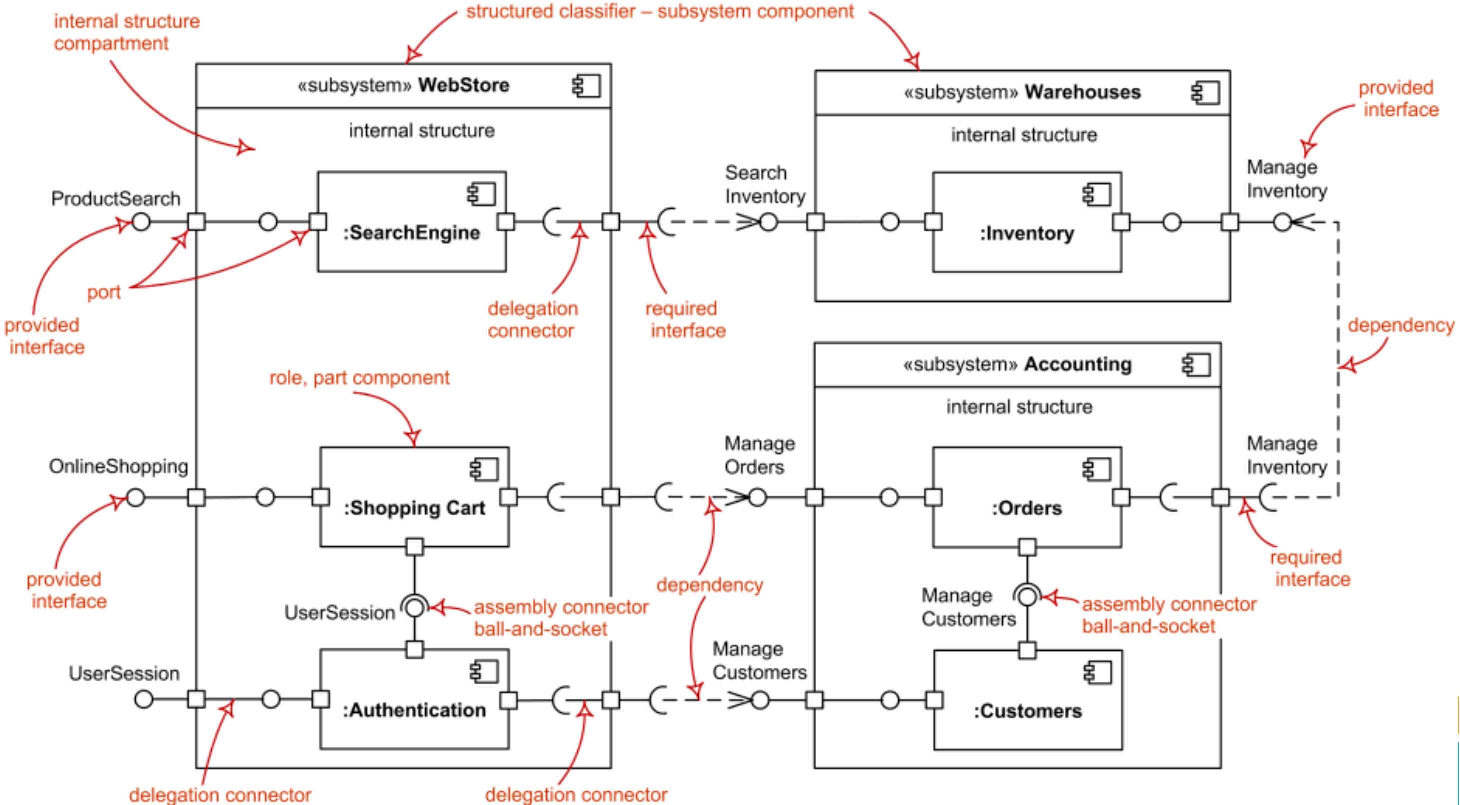
Figure 6 — The 5 layers of Hughes Air Traffic Systems (HATS)

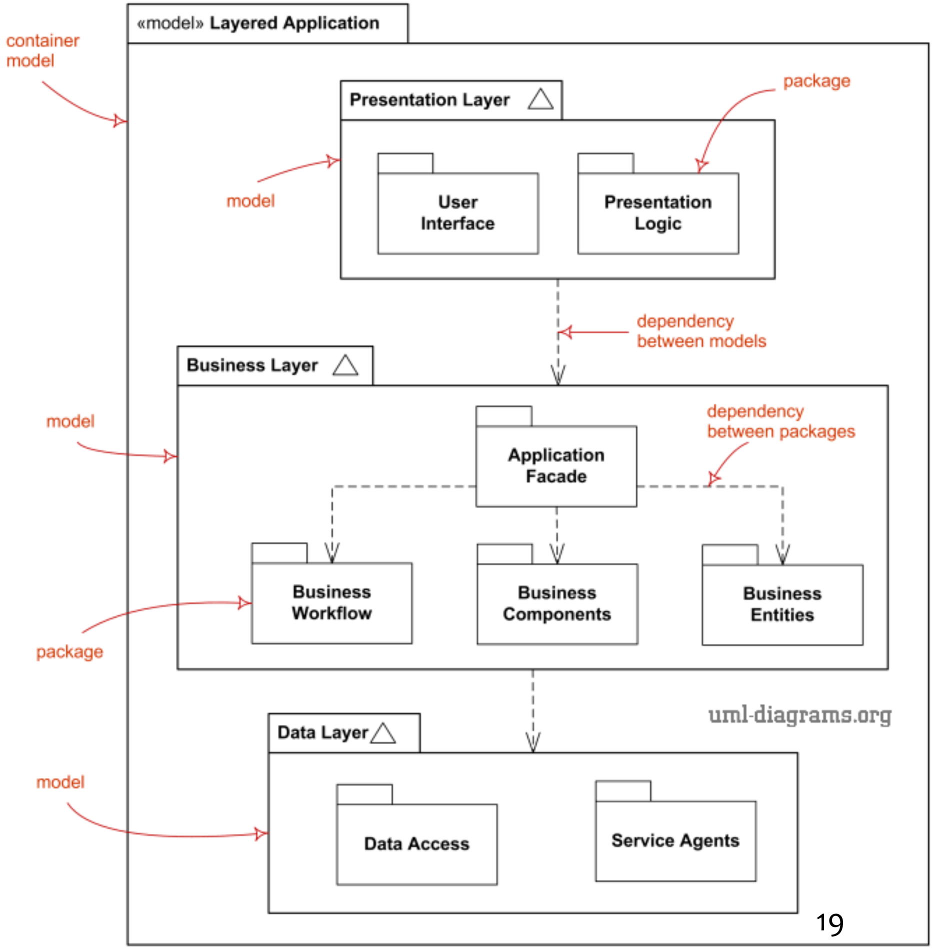
Development View

- Stakeholders
  - Software engineer, programmer, project manager
- Concerns
  - Software configuration management and concerns such as maintainability and reusability
  - Assign functionality to subsystems in support of development
- Focus – Configuration Management
  - Packages used
  - Execution environments
  - Class libraries and subsystems utilized

Development View

- **Relevant UML diagrams**
  - Component, class, package,
  - Layered architecture style can be used





Physical View

- A physical view, which shows the system hardware and how software components are distributed across the processors in the system.

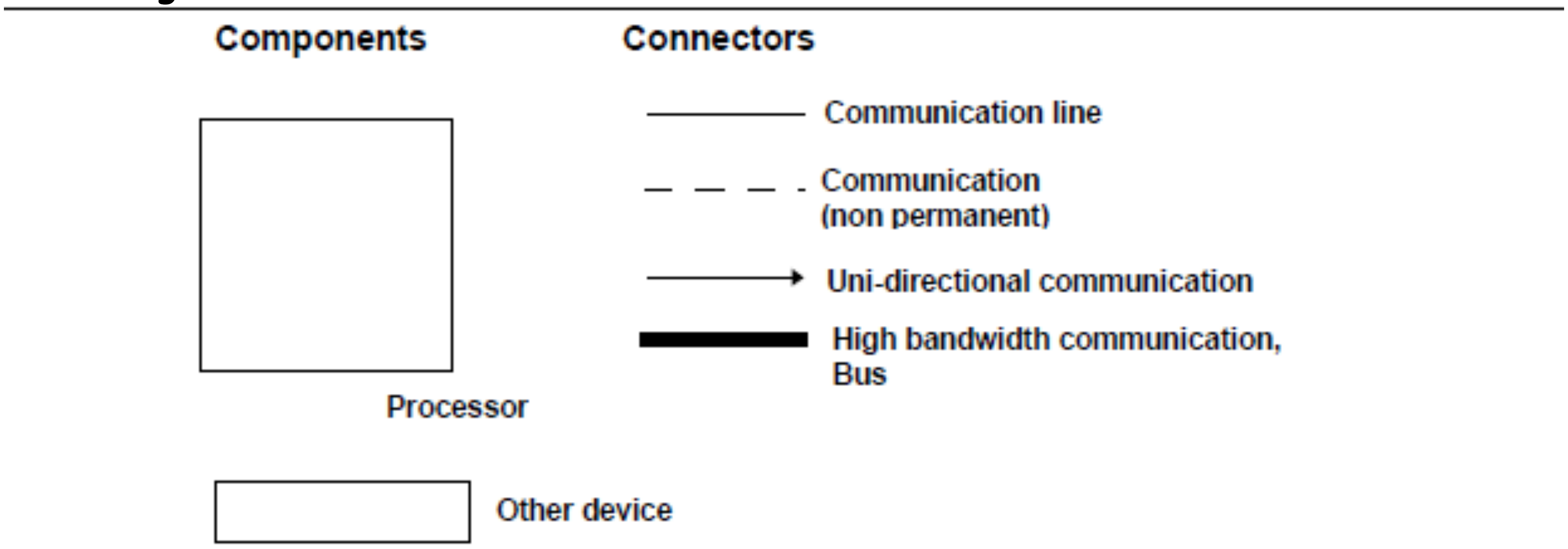

Components
Connectors
Communication line
Communication (non permanent)
Uni-directional communication
High bandwidth communication, Bus
Processor
Other device

Figure 7 — Notation for the Physical blueprint

Physical View

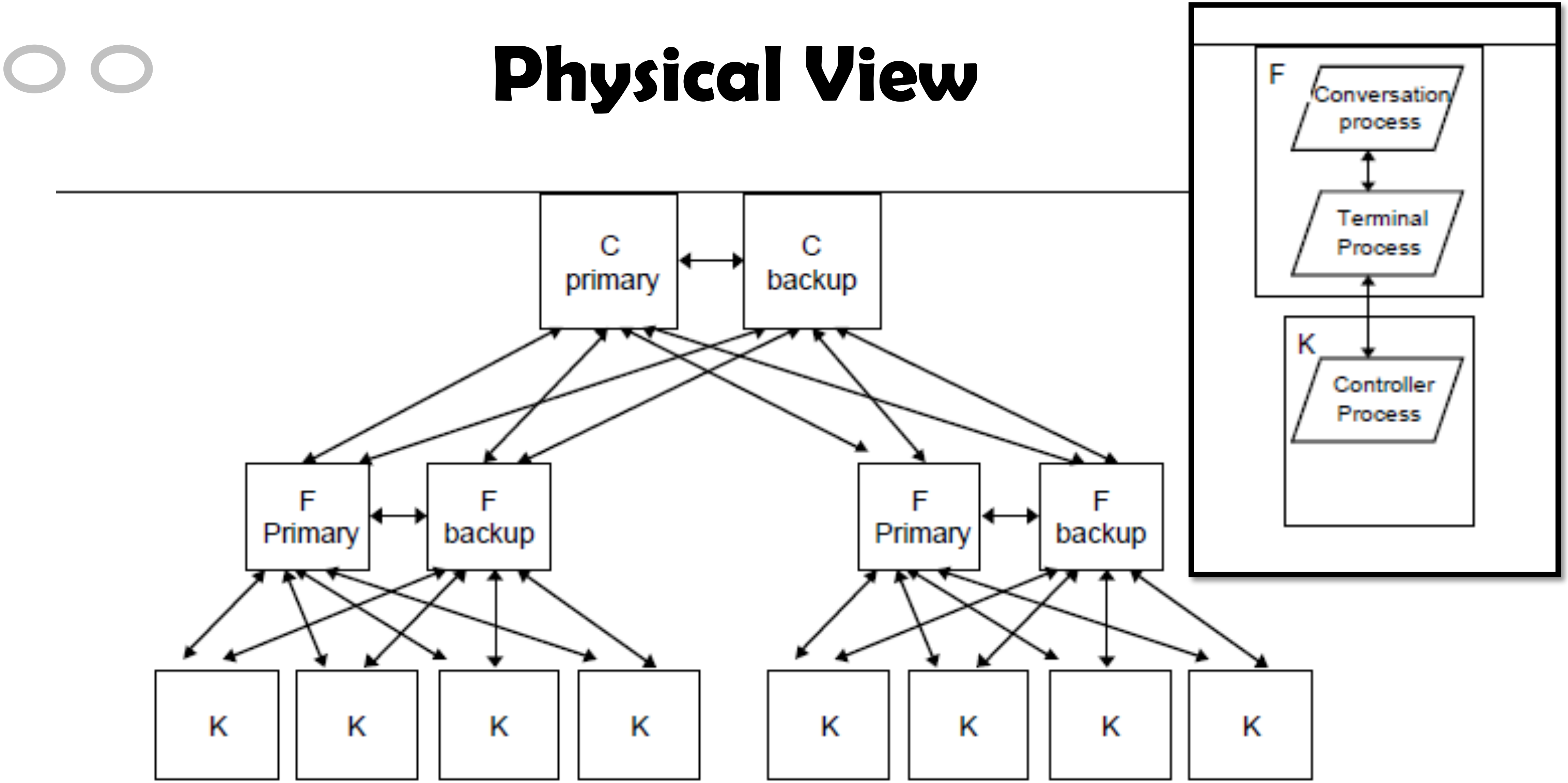


Figure 8 — Physical blueprint for the PABX

Physical View

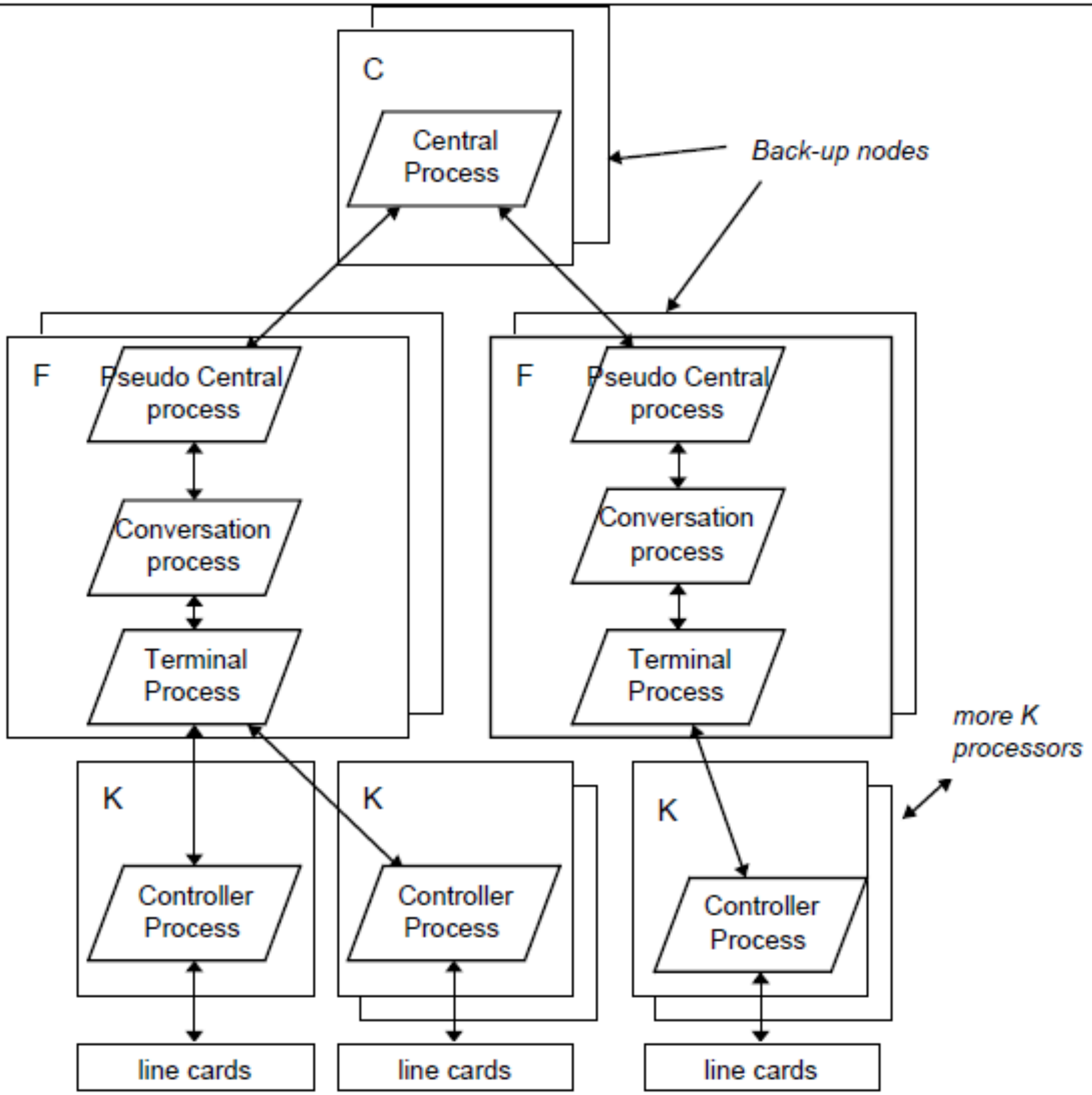


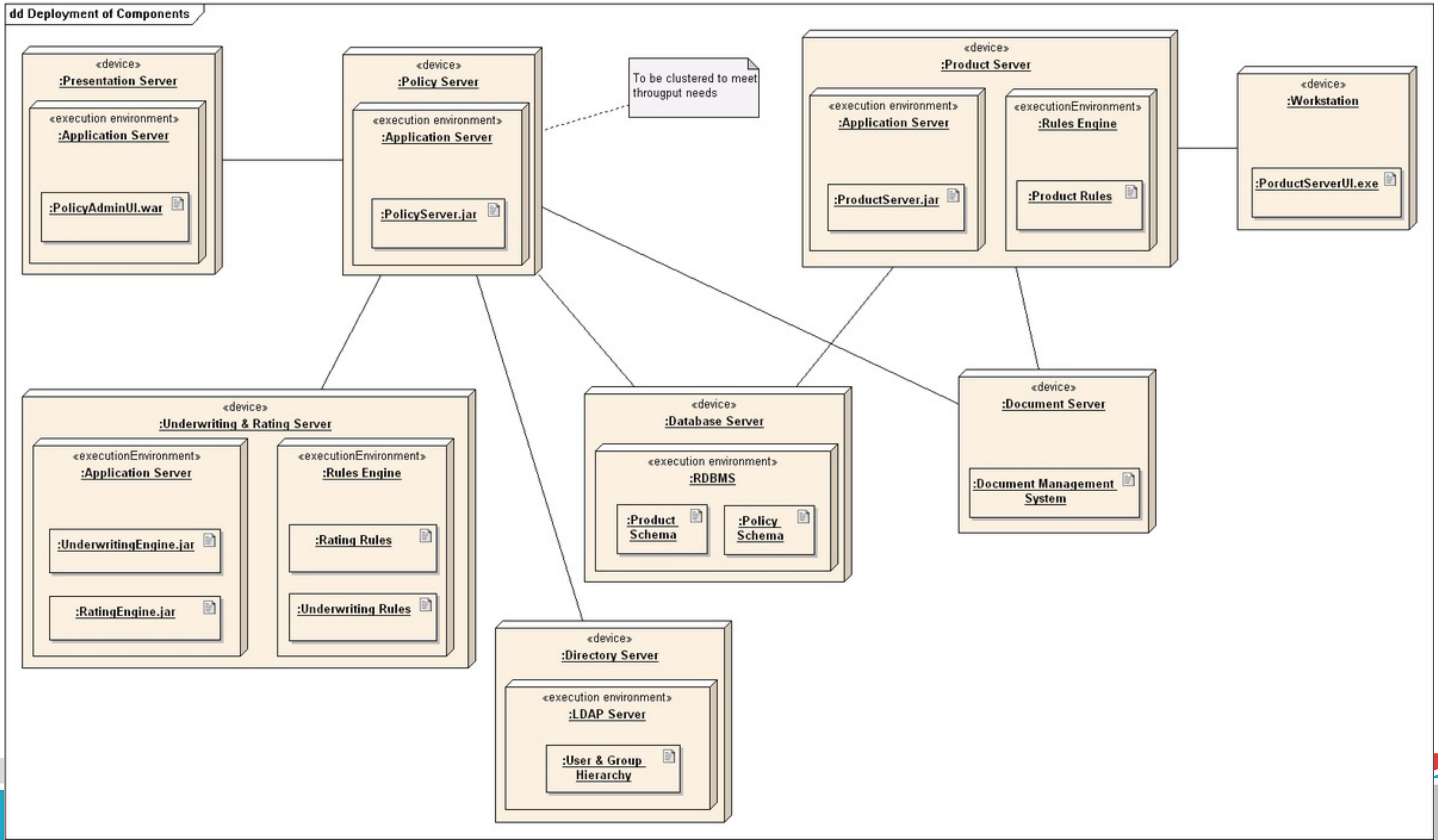
Figure 10 — Physical blueprint for a larger PABX showing process allocation

Physical View

- Stakeholders
  - System Engineer, Software Architect
- Concerns
  - Show how processes are allocated to the various computing machines
- Focus – Allocation to computing machines
  - Processes
  - Nodes (hardware systems)

Physical View

- **Relevant UML diagrams**
  - Deployment



4+1 View

- **Related using use cases or scenarios (+1)**

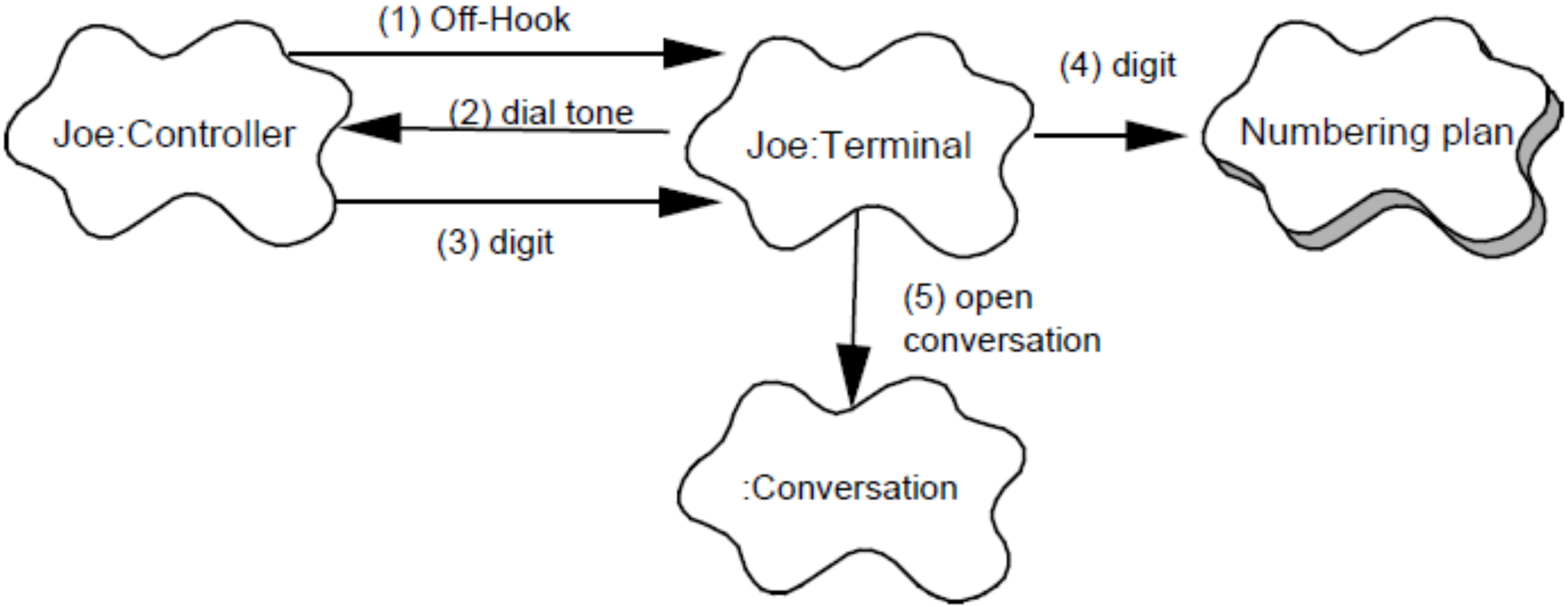
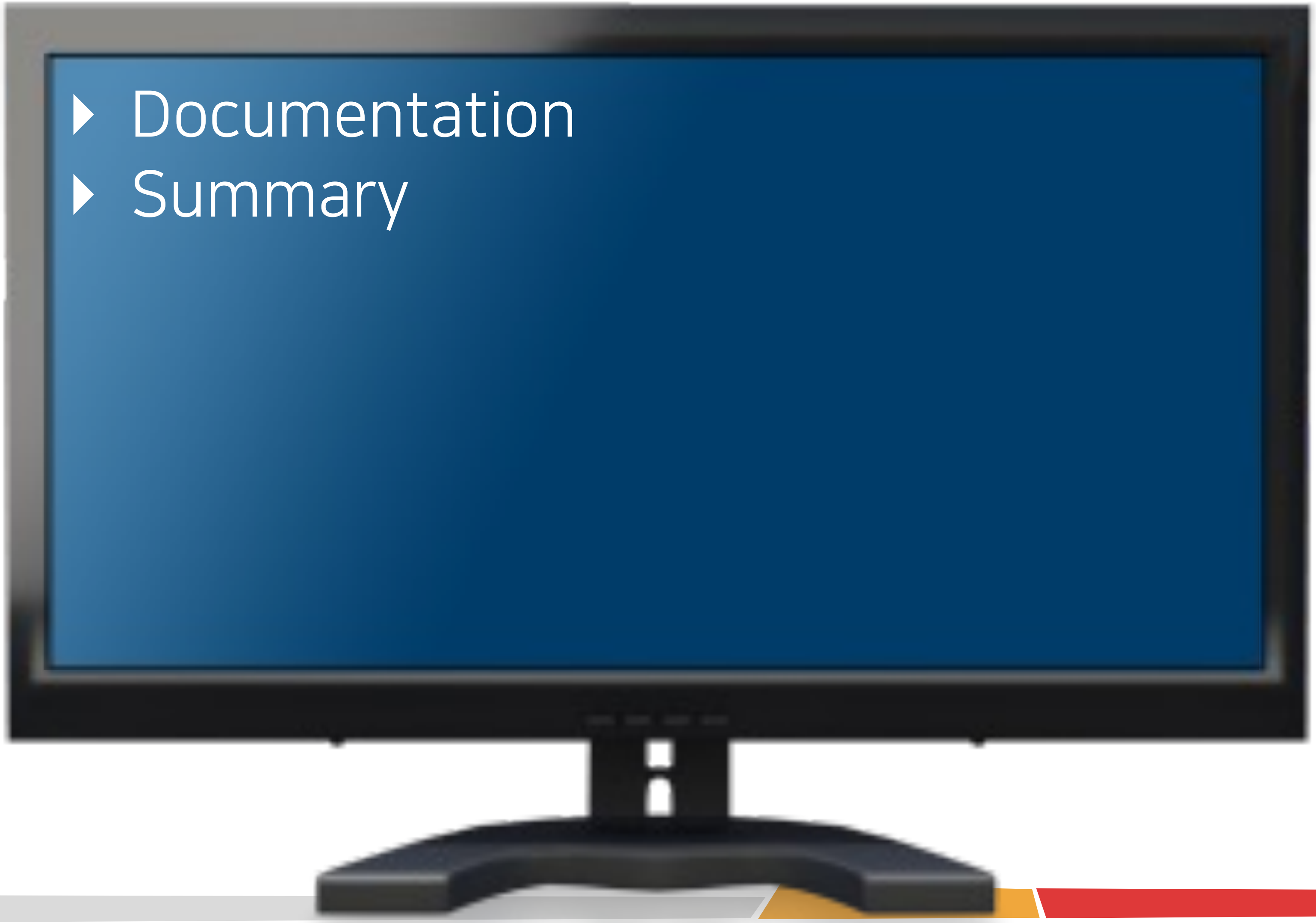


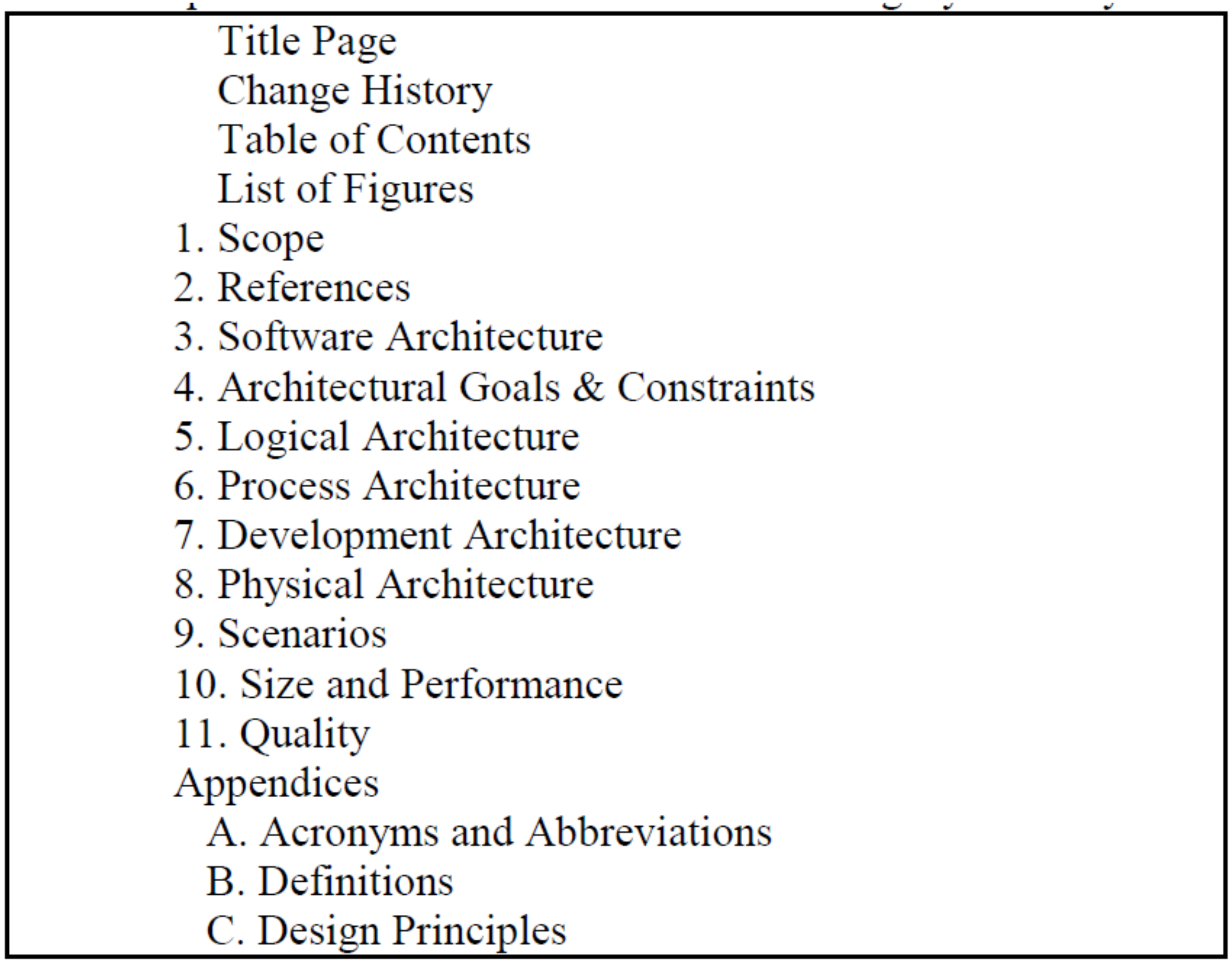
Figure 11 — Embryo of a scenario for a local call—selection phase

Summary



- Documentation
- Summary

Documenting the 4+1 Architecture



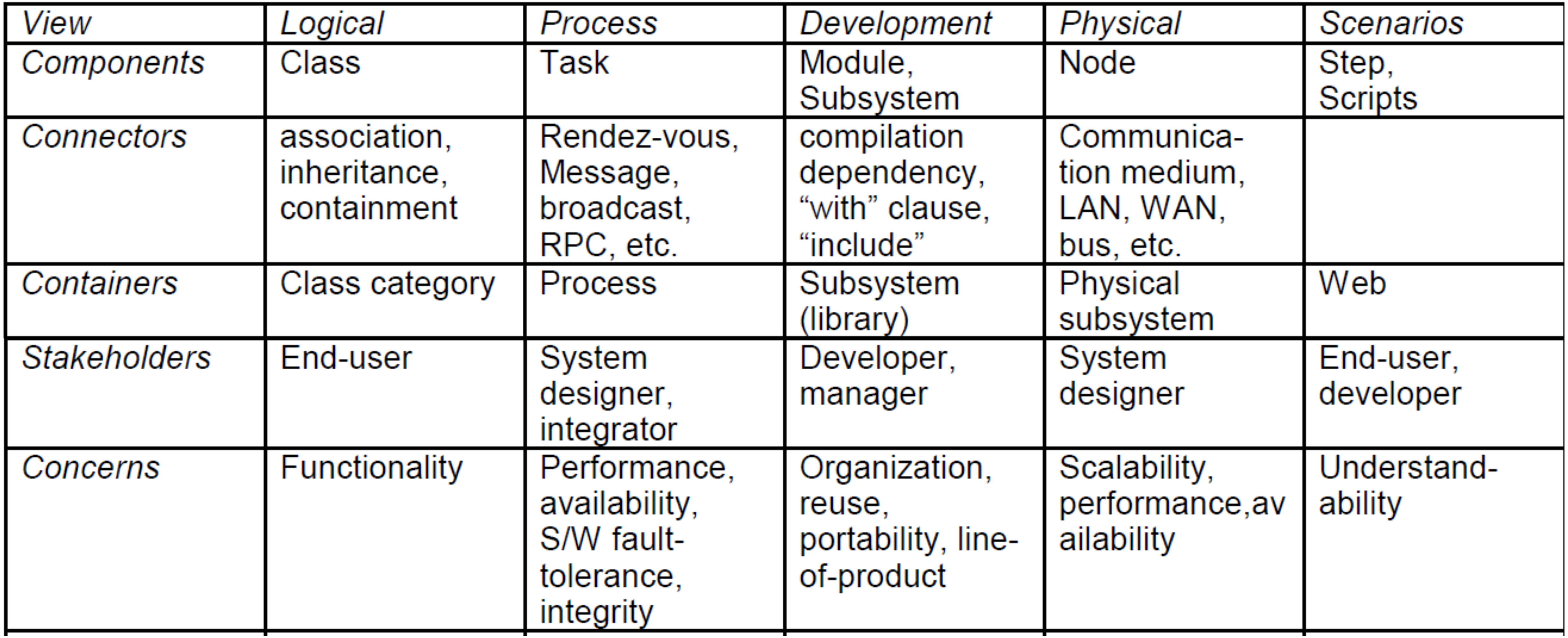
Title Page
Change History
Table of Contents
List of Figures

1. Scope
2. References
3. Software Architecture
4. Architectural Goals & Constraints
5. Logical Architecture
6. Process Architecture
7. Development Architecture
8. Physical Architecture
9. Scenarios
10. Size and Performance
11. Quality

Appendices

A. Acronyms and Abbreviations
B. Definitions
C. Design Principles

Summary



View	*Logical*	*Process*	*Development*	*Physical*	*Scenarios*
Components	Class	Task	Module, Subsystem	Node	Step, Scripts
Connectors	association, inheritance, containment	Rendez-vous, Message, broadcast, RPC, etc.	compilation dependency, “with” clause, “include”	Communication medium, LAN, WAN, bus, etc.	
Containers	Class category	Process	Subsystem (library)	Physical subsystem	Web
Stakeholders	End-user	System designer, integrator	Developer, manager	System designer	End-user, developer
Concerns	Functionality	Performance, availability, S/W fault-tolerance, integrity	Organization, reuse, portability, line-of-product	Scalability, performance,availability	Understandability

Question?

Seonah Lee
saleese@gmail.com