

Software Architecture



Session 9: Design Decisions and Design Concepts





Outline



- 
1. Design Decisions
 2. Categories of Design Decisions
 3. Types Design Concepts
 4. Important Remarks
- 

What is a design?

What is the meaning of designing?

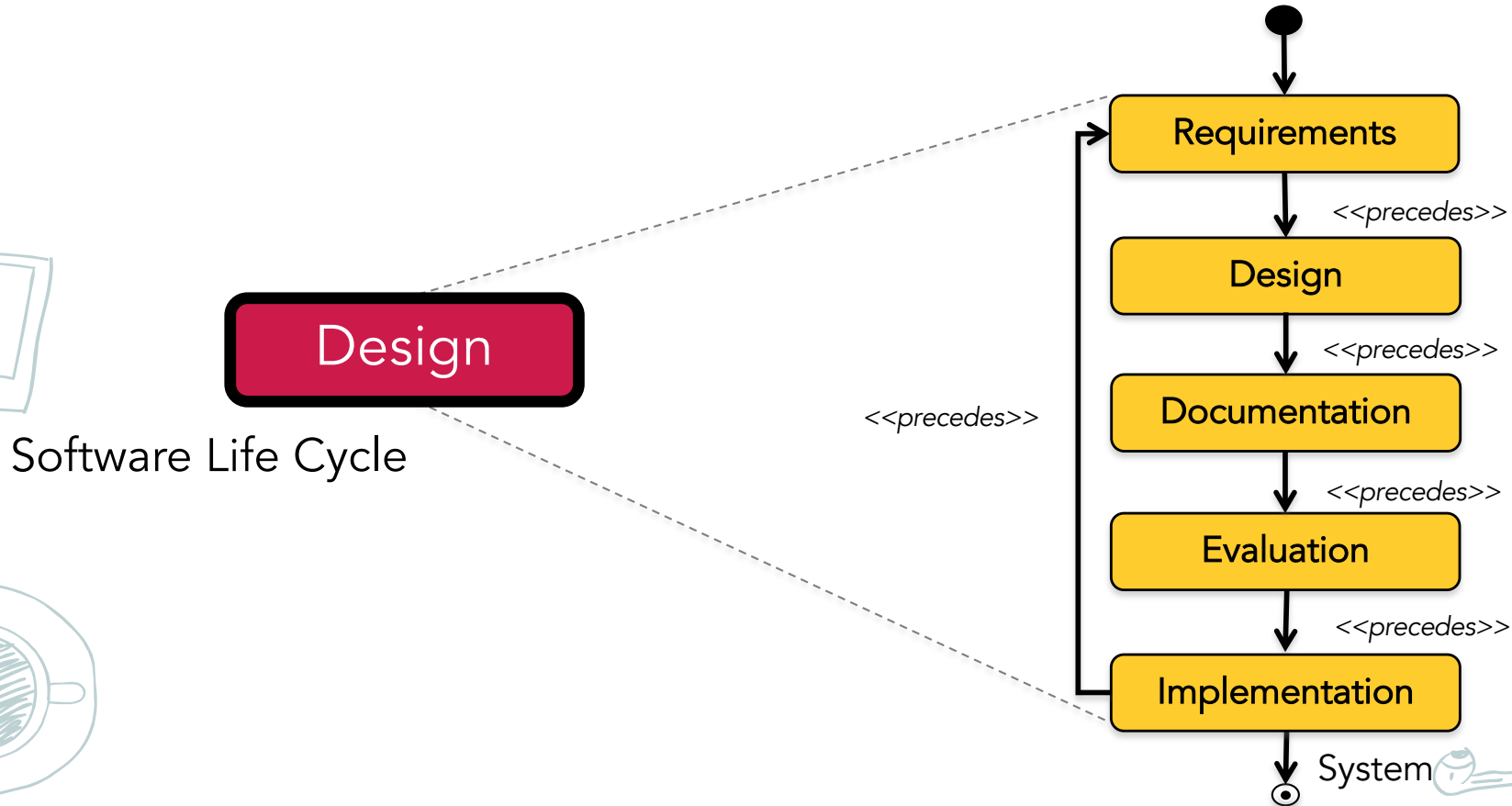




Software System Design

- As a process: it is a **iterative process** through which a designer makes design decisions to build a model that provides a solution to requirements.
- As a noun: it is a **model** that guides the conversion of a specification into executable code, i.e. the means by which a problem description is turned into a problem solution.

Software Architecture Development Cycle





Software System Design

- As a process: it is a **iterative process** through which a designer makes design decisions to build a model that provides a solution to requirements.
- As a noun: it is a **model** that guides the conversion of a specification into executable code, i.e. the means by which a problem description is turned into a problem solution.



Software Architecture

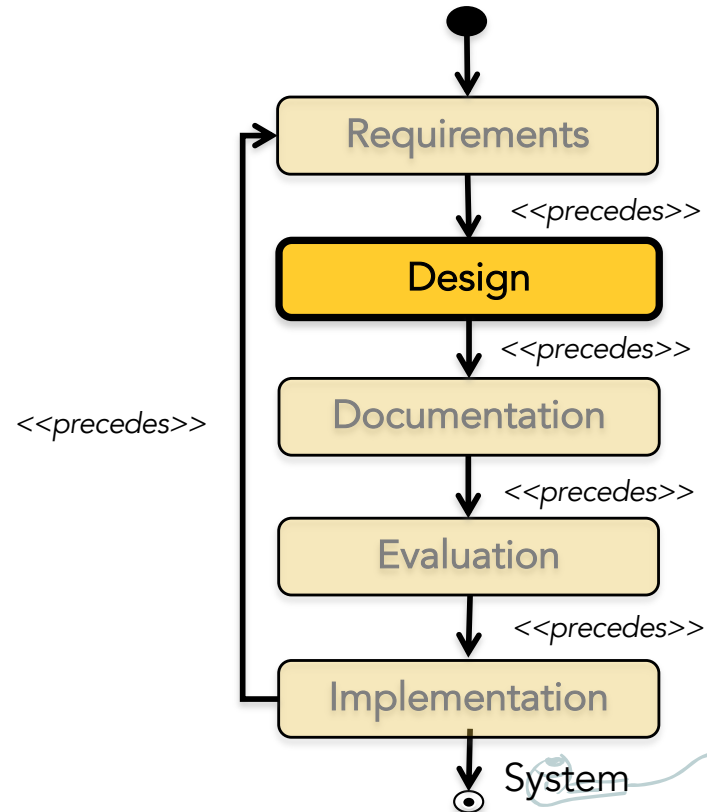
*“The software architecture of a system is the **set of structures** needed to reason about the system, which comprise software elements, relations among them, and properties of both.”*

Len Bass, Paul Clements, and Rick Kazman.
Software Architecture in Practice (3rd ed.). Addison-Wesley Professional, 2012.

Software Architecture Development Cycle

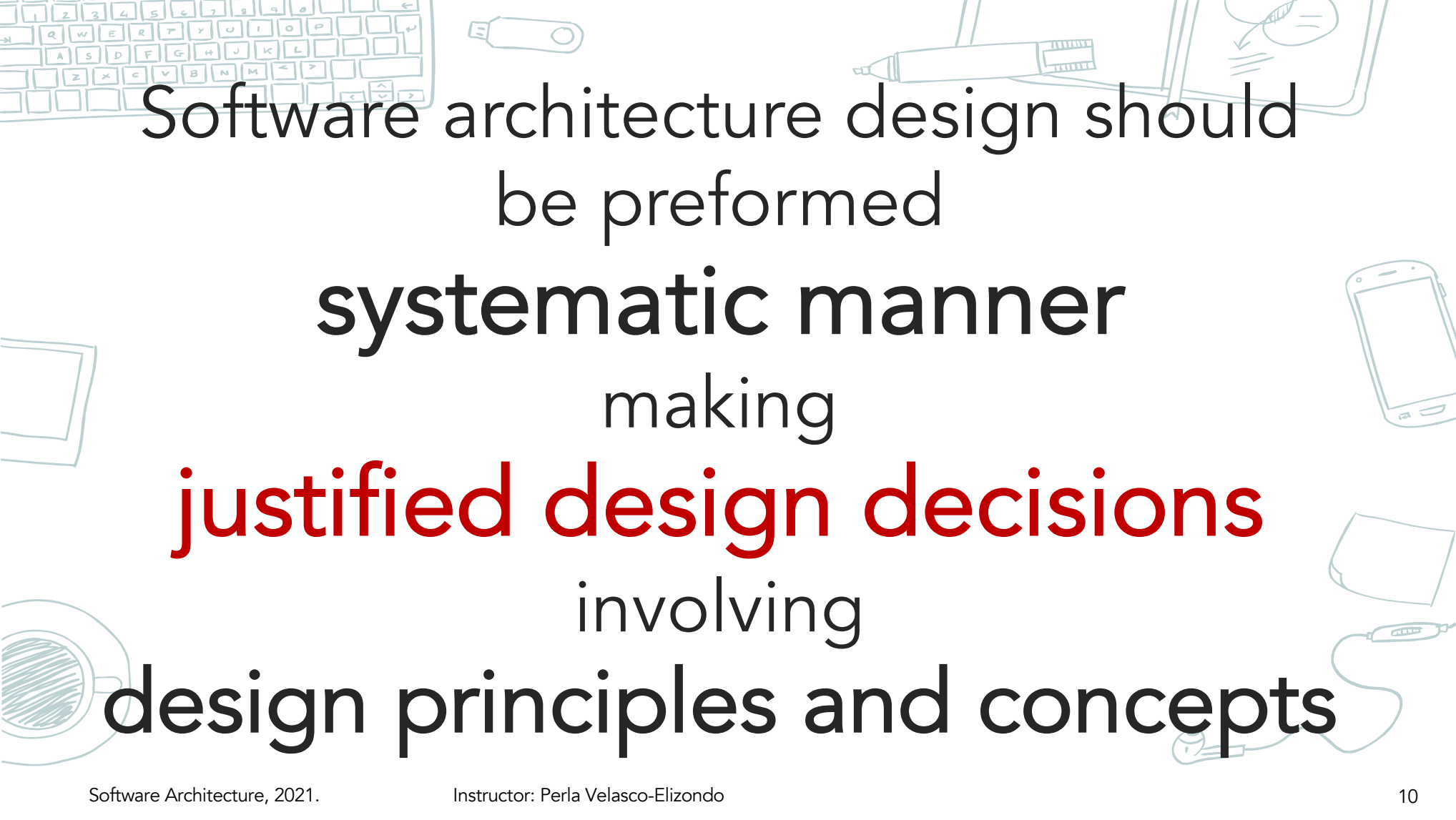
Making design decisions,
creating architectural
structures

A translation, from the world
of needs (requirements) to
the world of solutions, in
terms of architectural
structures.



Software architecture design is not a mystical activity





Software architecture design should
be preformed
systematic manner
making
justified design decisions
involving
design principles and concepts

Categories of design decisions

1. Allocation of responsibilities
2. Coordination model
3. Data model
4. Management of resources
5. Mapping among architectural elements
6. Binding time decisions
7. Choice of technology



1. Allocation of Responsibilities

- Identifying the important responsibilities, including basic system **functions** and satisfaction of **quality attributes**.
- Determining how these responsibilities are allocated to non-runtime and runtime elements (namely, modules, components, and connectors).



2. Coordination Model

- Identifying what **elements** of the system that must coordinate.
- Determining the **properties** of the coordination mechanisms
 - stateful vs. stateless,
 - synchronous vs. asynchronous,
 - guaranteed vs. nonguaranteed delivery,
- Choosing the **communication mechanisms** (between systems, between our system and external entities, between elements of our system) that realise those properties.



3. Data Model

- Determining how the main **data items** are created, initialized, accessed, persisted, manipulated, translated, and destroyed.
- Determining whether any **metadata** is needed for consistent interpretation of the data.
- Determining whether the data is going to be **stored** in a relational database or other.



4. Management of Resources

- Identifying what **resources** that must be managed and determining the limits for each.
- Determining which system element(s) **manage each resource**.
- Determining how resources are **shared** and the sharing strategies employed.
- Determining the impact of **saturation** on different resources.



5. Mapping among Architectural Elements

- The mapping of modules and runtime elements to each other.
- The assignment of runtime elements to processors.
- The assignment of items in the data model to data stores.

6. Binding Time Decisions

- In computer programming “Binding time” refers to the time when the association of an attribute and value is performed.
- Common binding times are:
 - a. Compiling time
 - b. Build time
 - c. Loading time
 - d. Run time



6. Binding Time Decisions

- Allocation of responsibilities, ej. build-time selection of modules via a parameterized makefile.
- Resource management, ej. accept new peripheral devices plugged in at runtime.

7. Choice of Technology

Every architecture decision must eventually be realized using a specific technology.





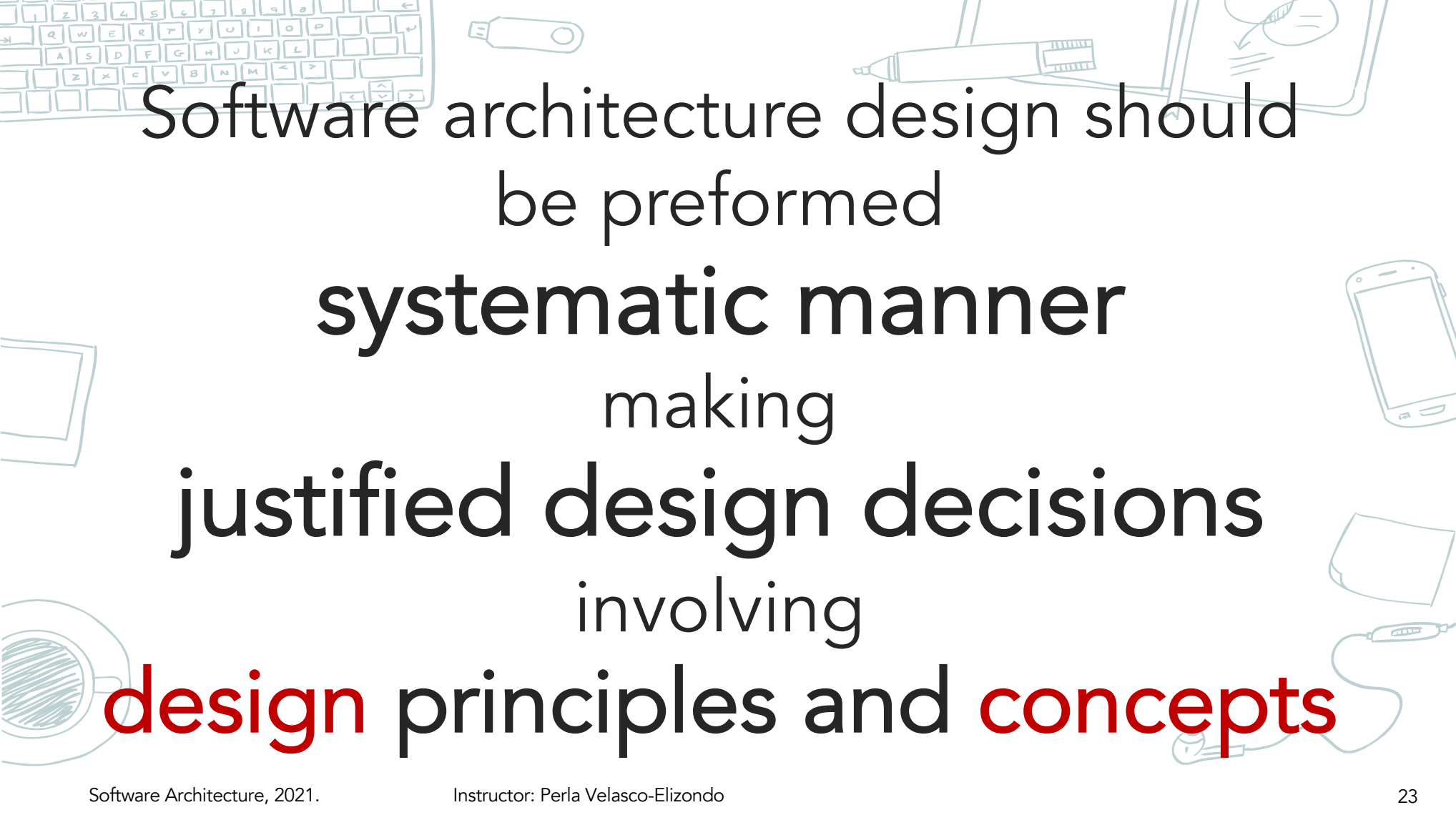
7. Choice of Technology

- Deciding which technologies are **available** to realize the decisions made in the other categories.
- Determining whether the **available tools** to support this technology choice
- Determining the extent of **internal familiarity** as well as the degree of external **support** available for the technology
- Determining the **side effects** of choosing a technology, such as a required coordination model or constrained resource management opportunities.
- Determining whether a new technology is **compatible** with the existing technology stack.

Facing the “blank page”



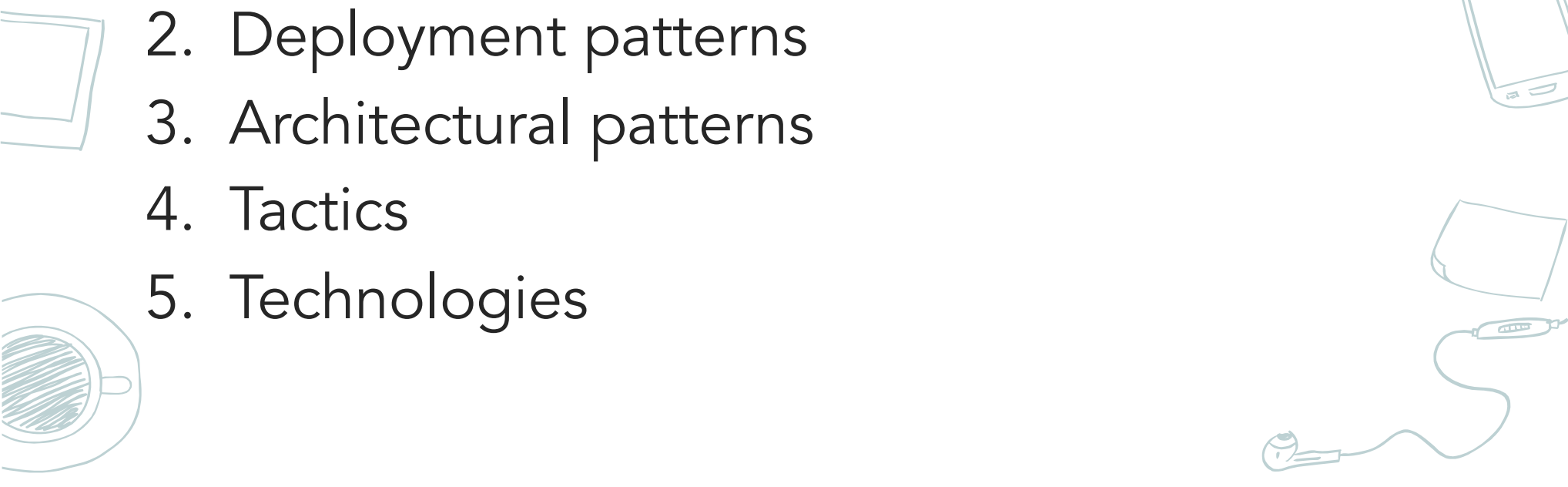
miro



Software architecture design should
be preformed
systematic manner
making
justified design decisions
involving
design principles and concepts



Types of Design Concepts

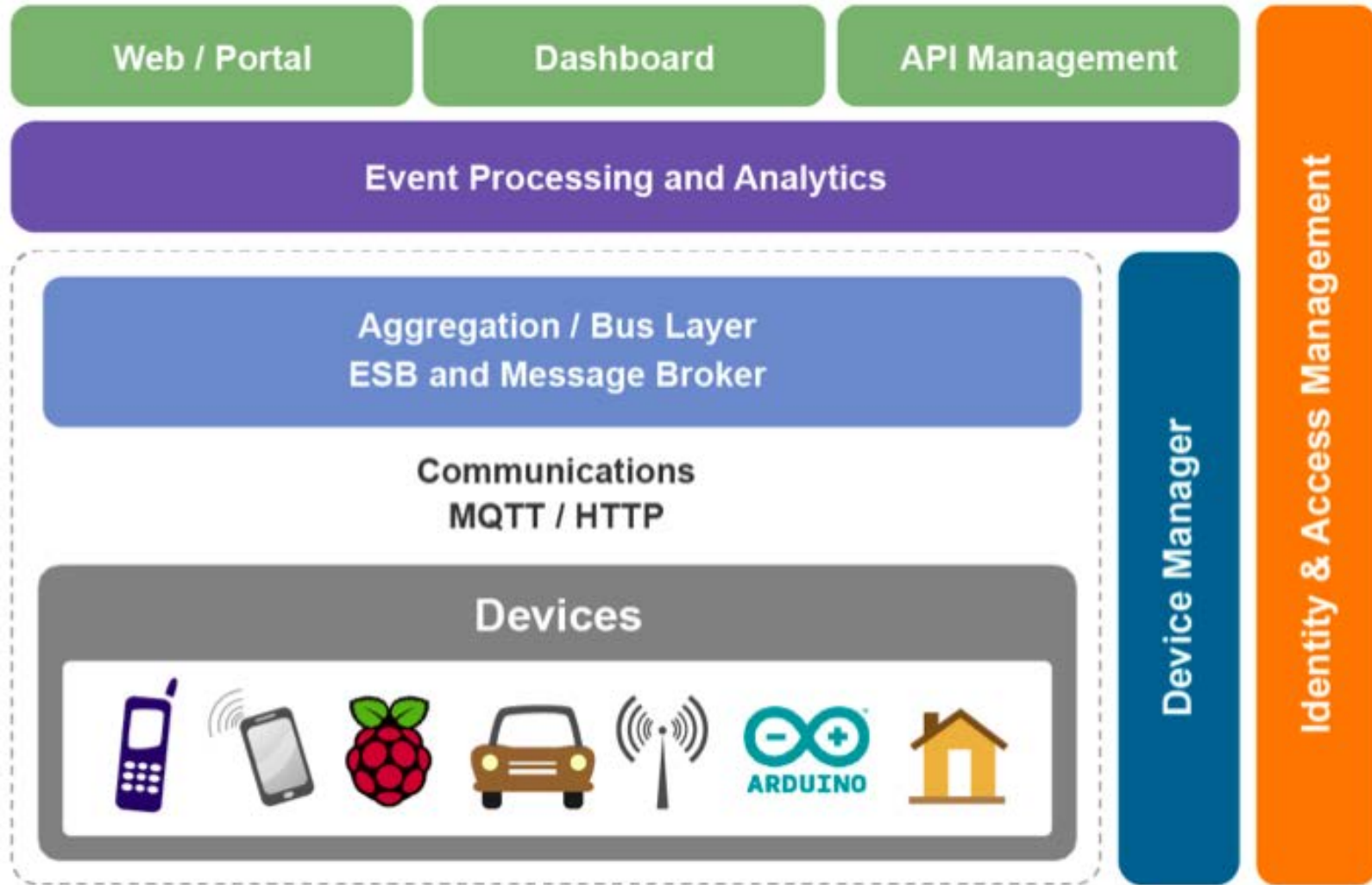
- 
1. Reference architectures
 2. Deployment patterns
 3. Architectural patterns
 4. Tactics
 5. Technologies



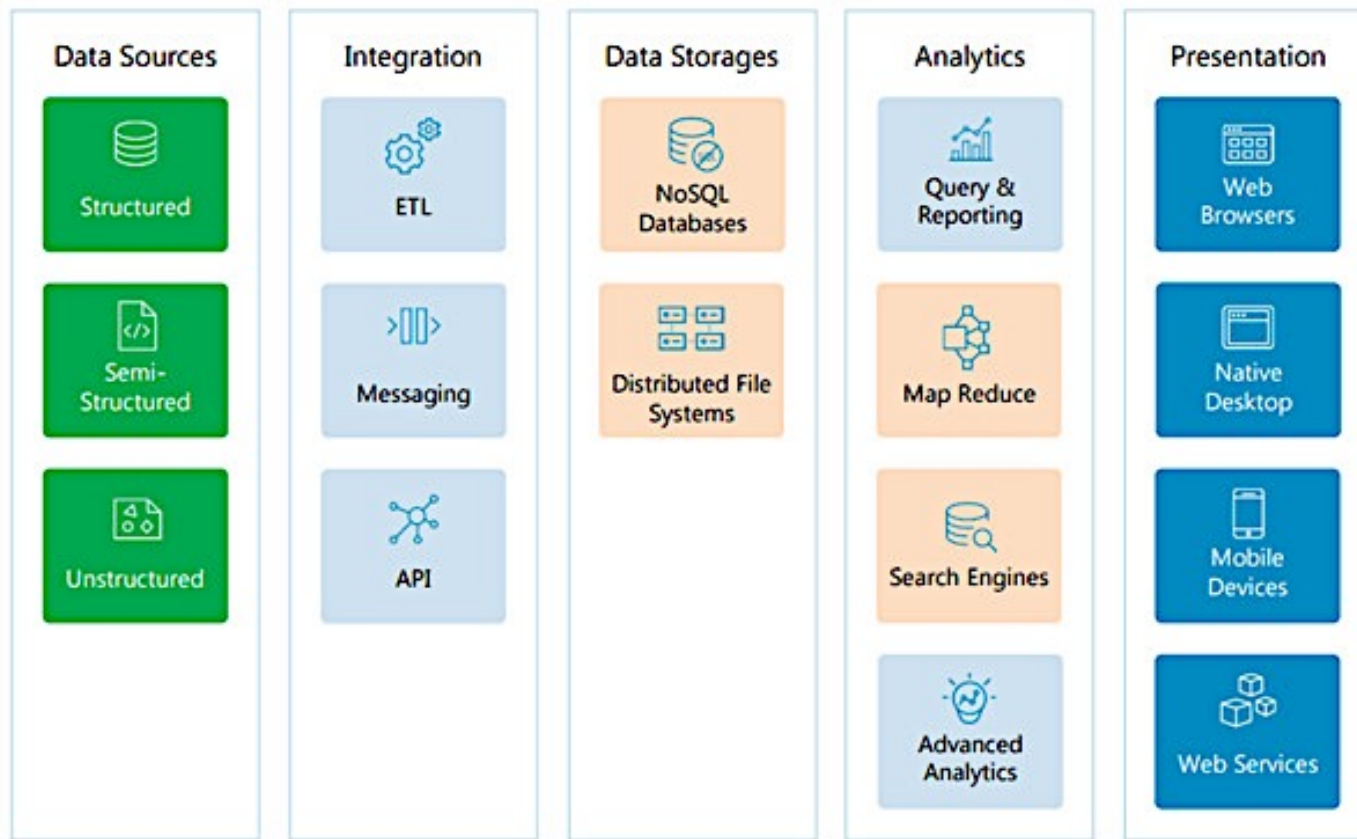
1. Reference Architectures

- Provide a structure for **particular types of applications** (in specific domains); they may embody different architectural patterns.
- They are often more directly connected to recommended structures and integrations of **IT products** and **services** to form a solution.

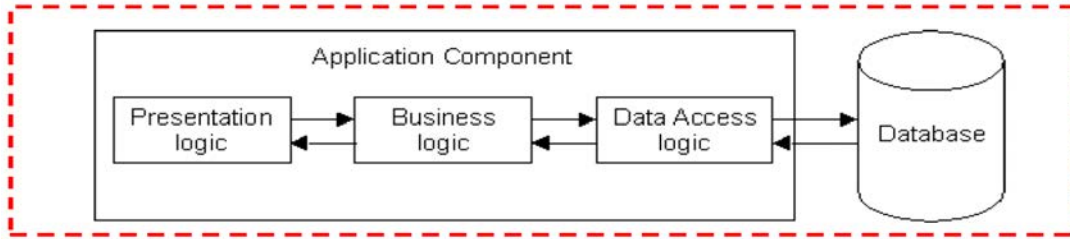
IoT Reference Architecture



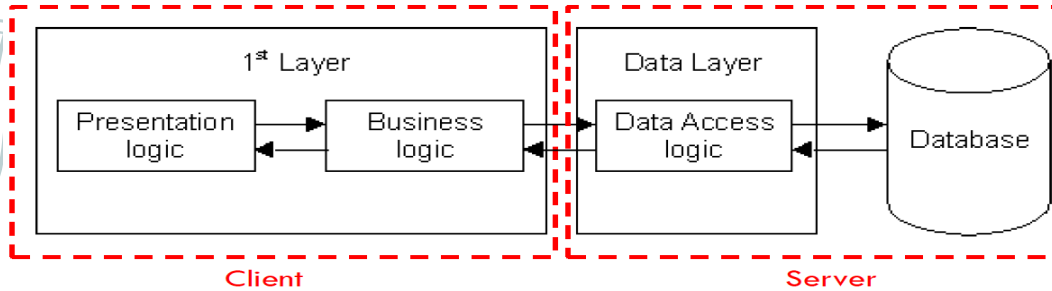
Extended Relational Reference Architecture



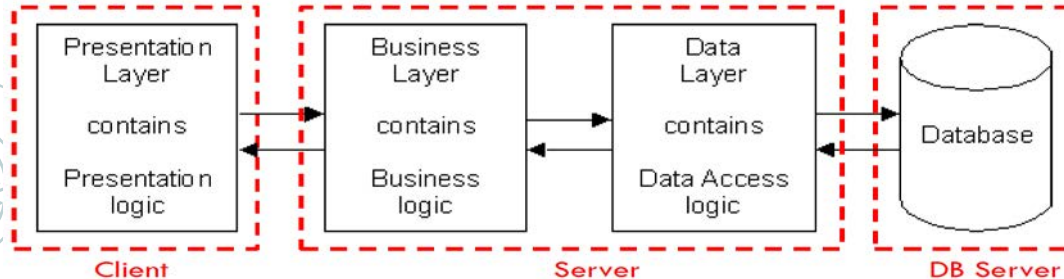
2. Deployment Patterns



3-Layers
1-Tier Architecture

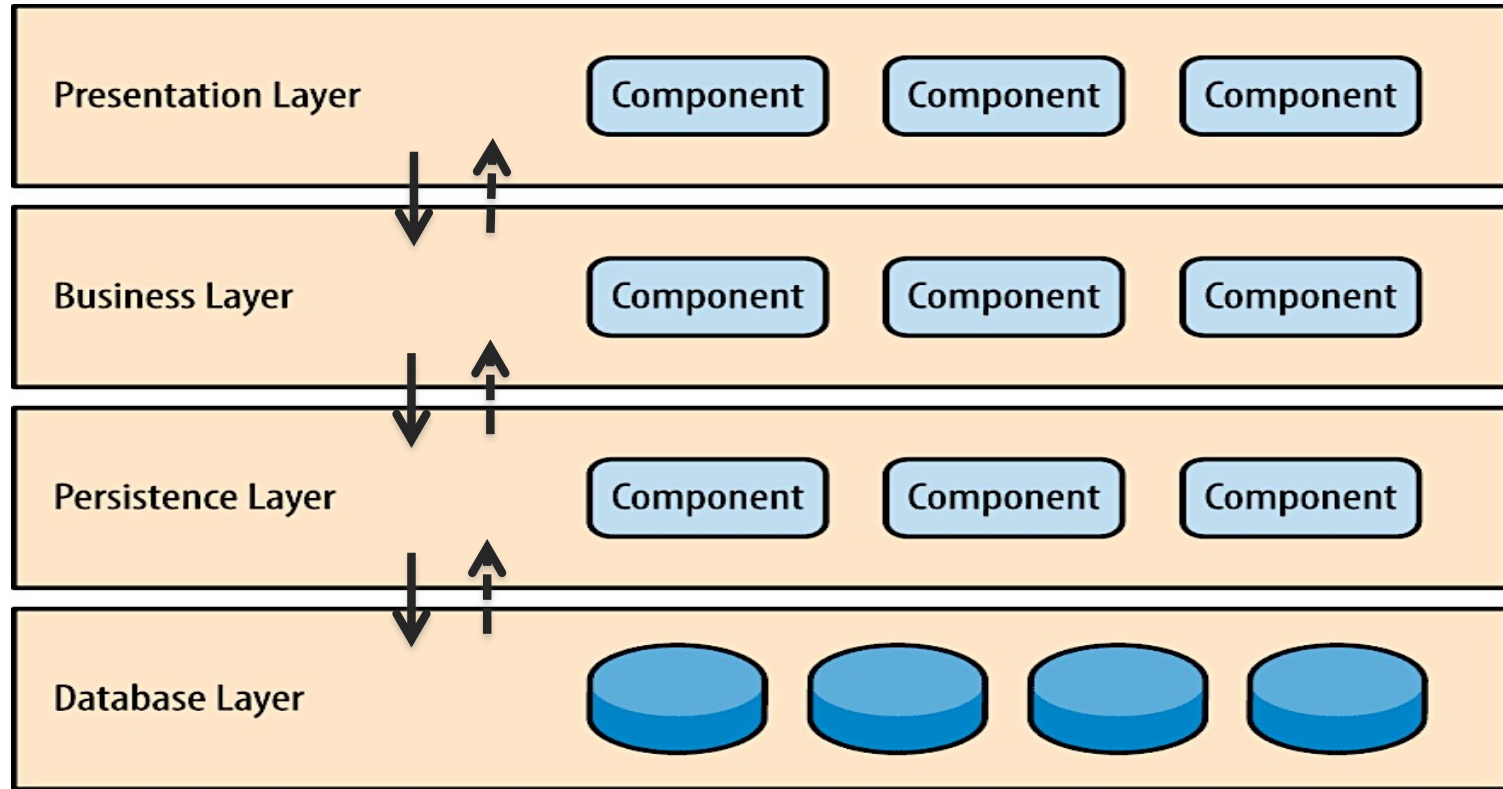


3-Layers
2-Tier Architecture



3-Layers
3-Tier Architecture

3. Architectural Patterns





Architectural Pattern: Layers

1. **The presentation layer:** contains all of the components responsible for presenting the UI to the end-user or sending the response back to the client
2. **The business layer:** represents the underlying domain, mostly consisting of domain entities and, in some cases, services.
3. **The Persistence Layer and Database Layer:** Contains all the components responsible for doing the technical stuff, like persisting the data in the database, like DAOs, repositories.

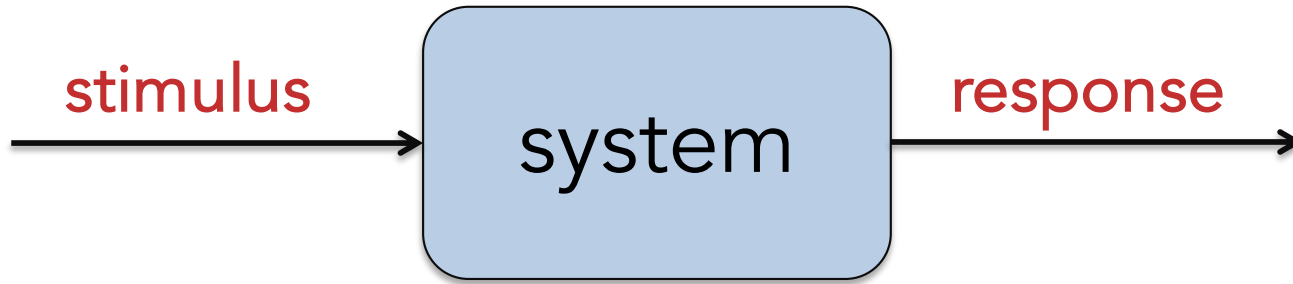


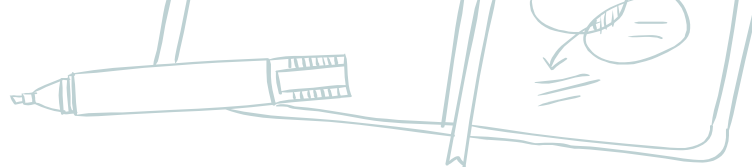
4. Tactics

"Between stimulus and response, there is a space. In that space is our power to choose our response. In our response lies our growth and our freedom."

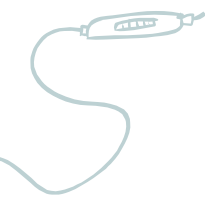
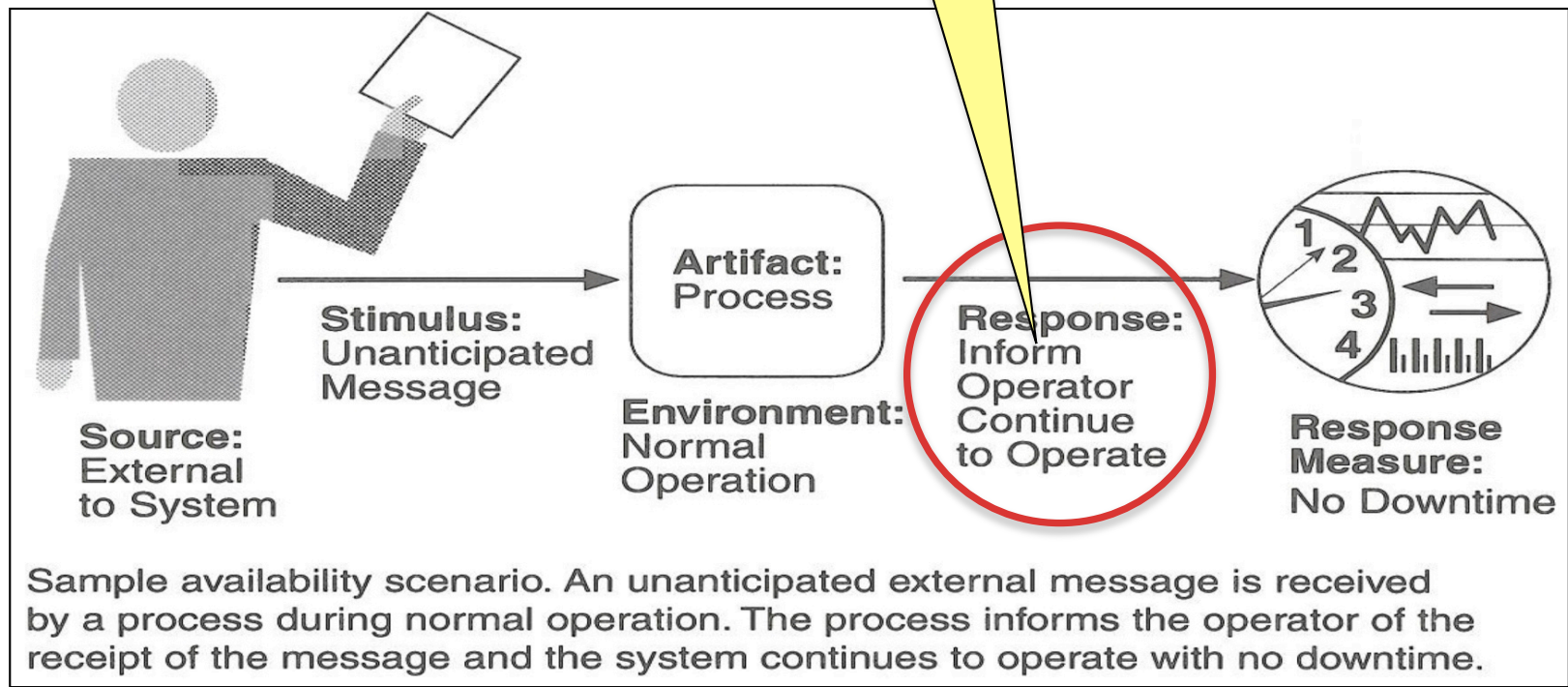
Stimulus, Response, Environment

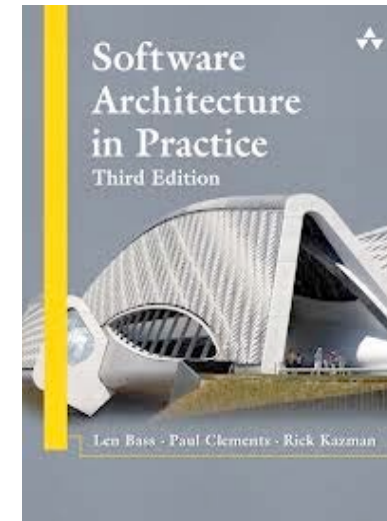
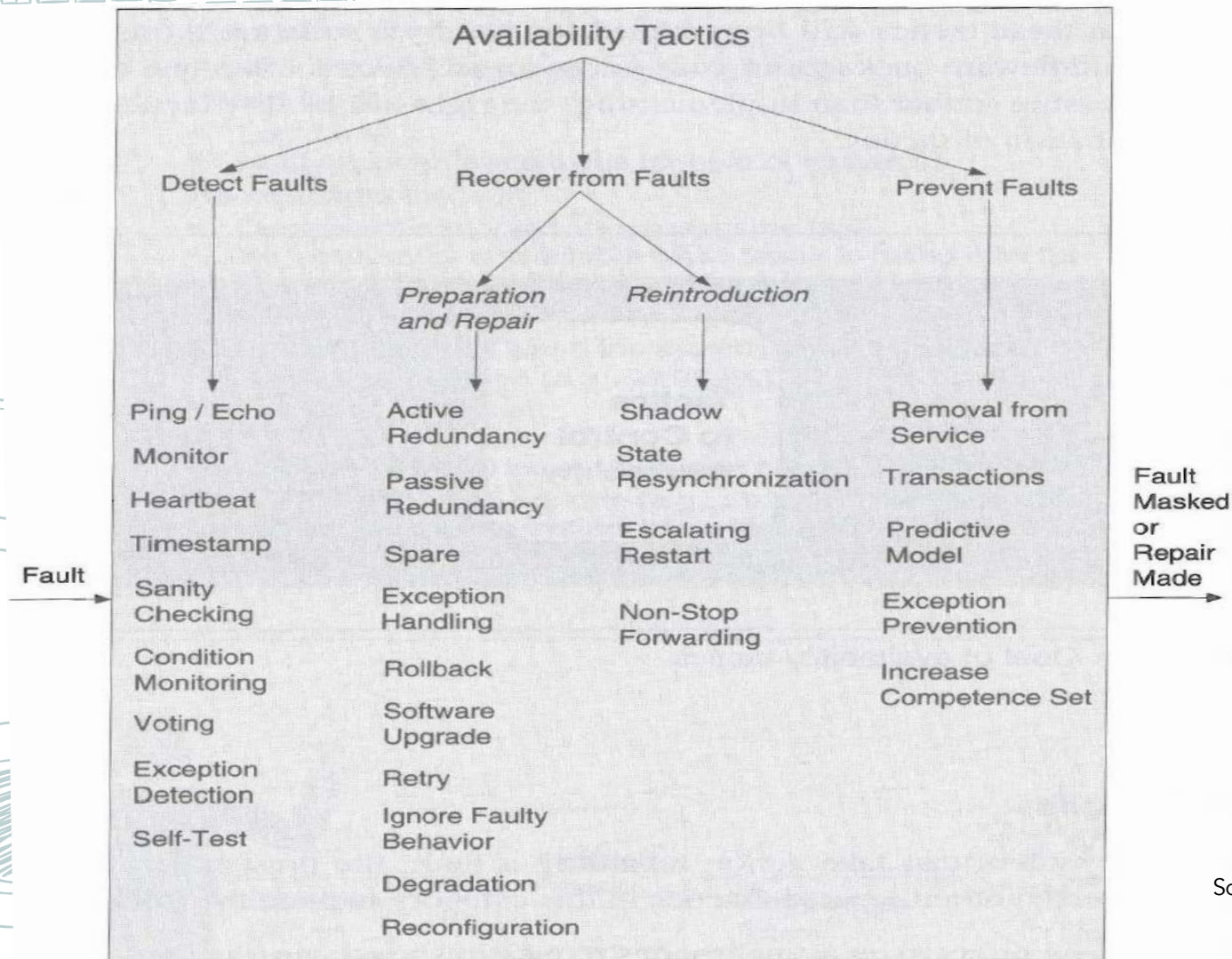
- A good approach to discuss quality attribute requirements.
- In its most basic form, describes the system's response to some stimulus





¿How?





*Len Bass, Paul Clements, and Rick Kazman.
Software Architecture in Practice (3rd ed.). Addison-
Wesley Professional, 2012.



5. Technologies

- Technology families

e.g. Relational Database Management System (RDBMS) or an Object-Oriented to Relational mapper (ORM).

- Products

e.g., Oracle or Microsoft SQL Server .

- Application Frameworks

e.g. Rails, Spring, Yii



Important Remarks

- There is **no single recipe** for designing an architecture. It is a process based on creativity and experience.
- The description of many design concepts is usually
 - a) Informal**: usually expressed in natural language.
 - b) Inconsistent** : different interpretations by different authors.
 - c) Incomplete**: it does not provide a complete solution to the problem.



Important Remarks

- The **selection** of design concepts is **responsibility of the architect**
- The **implementation** of them is **responsibility of the development team** (under the supervision of the architect, sure!)
- Many design concepts are independent of the programming language



Important Remarks

The selection of design concepts must be guided by the **architectural requirements** and their effects that has on other architectural drivers.

Questions? Comments?

