



Chapter 1: Software Architecture

Architecture Business Cycle

H.S. Sarjoughian

CSE 460: Software Analysis and Design

School of Computing, Informatics and Decision Systems Engineering
Fulton Schools of Engineering

Arizona State University, Tempe, AZ, USA

Copyright, 2022

A Conceptual Roadmap to Software Architecture

Software Architecture Model

Design Pattern Models

typing concurrency persistence

abstraction

hierarchy

encapsulation

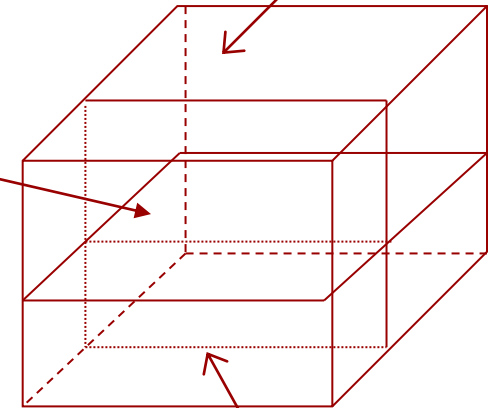
modularity

Object Model

UML Model

Logical/Dynamic:
interaction, ... diagrams

Logical/Static:
Class, ... diagrams



Physical/Static:
Component, ... diagrams

Design Patterns: Specific Problems with Solutions

Software Analysis & Design

Intro. Software Engineering

Software Architecture

Architectural Styles

Design Patterns

Singleton

Observer

Concepts, Principles and Methods

Complexity

Object
Model

...

Structural and Behavioral model specifications (UML Languages):
Strong Coverage

Process Models

Testing

Analysis & Design: ***Light Coverage***

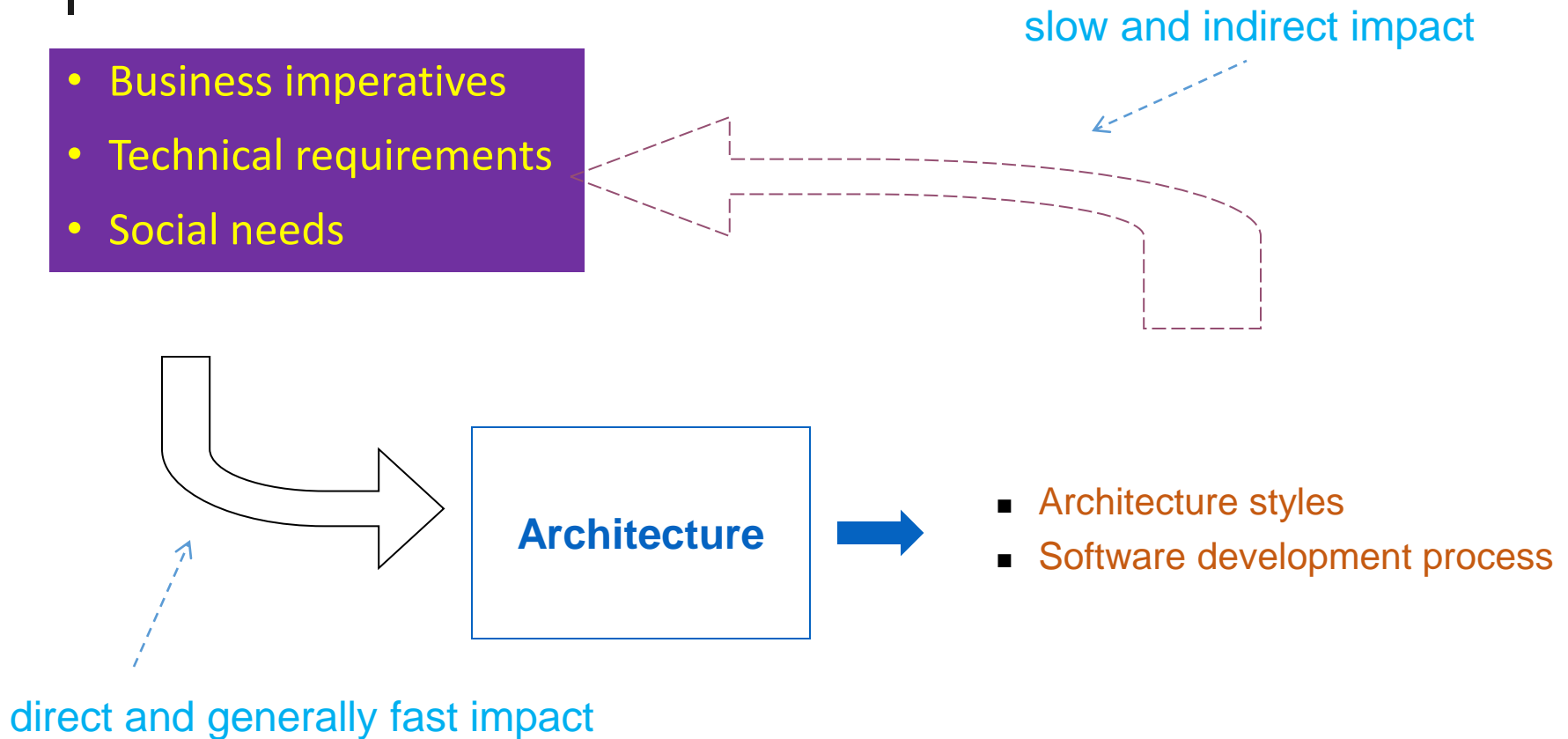
Software Architecture: Some Initial Observations

- Software architect task is to envision the skeleton of a software intensive system based on a wide range of competing and usually conflicting demands
 - Individuals, corporations, and governments are stakeholders in the development of software architectures
 - Architecture represents a set of **business**, **technical**, and **social** decisions
- Lifespan of software architecture by definition is open-ended!
 - Software architecture is expected to be a living artifact – it should lend itself to evolve within some reasonable bounds (e.g., cost, performance, usability, extensibility, etc.)

Architecture Business Cycle

- Architecture is expected to **serve** the following needs
 - Communication
 - Analysis
 - Reasoning
 - Software evolution
- Architecture is shaped due to **influences** from three factors
 - Business imperatives
 - Technical requirements
 - Social needs
- Software architecture **affects** business and social landscapes

Architecture Business Cycle (cont.)



Software architecture is a necessary artifact for building large-scale software-intensive systems

Architects (and therefore software/system architectures) **are influenced by stakeholders**

- **Marketing**
 - Cost
 - Time to market
 - Market share
 - Competitive strategies and practices
- **End-user**
 - General, ordinary individuals
 - Technical individual – e.g., system admin., developer, ...
- **Customer**
 - Generally has conflicting demands with end-user (cost vs. usability)
- **Maintenance organization**
- **Technical environments**
- **Developing organization**
 - Organizational imperatives

Technical environment considerations

- Resources
 - Hardware & software technologies (e.g., system-on-a-chip, J2EE), tools (Eclipse Graphical Modeling Framework, GitHub) , and standards (e.g., UML, MDA, Service Oriented Architecture, ...)
 - Human/intellectual capital – expertise in software engineering architecture
 - Distributed computing
 - Multi-tier architecture
 - Large-scale software development
 - Project management
 - Software safety
 - Ubiquitous/Wireless technologies

Organizational Imperatives

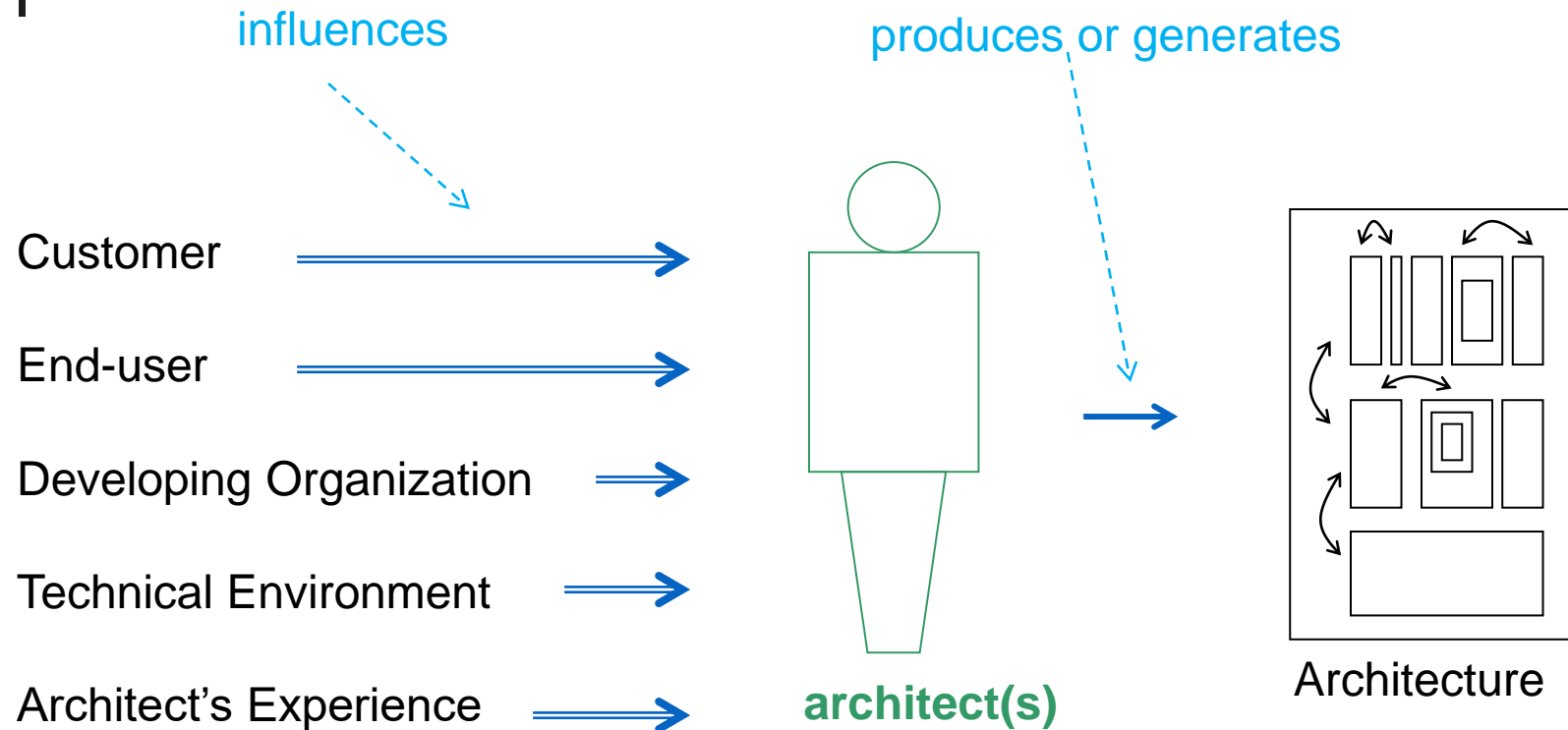
- **Pragmatics and reality of the business world**
 - Immediate, medium- and long-term business strategies
 - Asset reuse
 - Investment – asset building
 - Organizational structure
 - Small
 - Medium (e.g., Epic, Mathworks, ...)
 - Large (e.g., HP, Intel, ...)
 - Very Large (e.g., Amazon, Google, Oracle, PayPal, ...)
 - *Contracting vs. in-house philosophy*
- **Large and medium organizations can impose constraints that affect directly architectural choices**
 - If an organization has expertise in data-flow architectural style, it can be a challenge to choose another such as data-centered

Software Architecture



Organizational Structure

Influences on the Architect(s) and Architecture

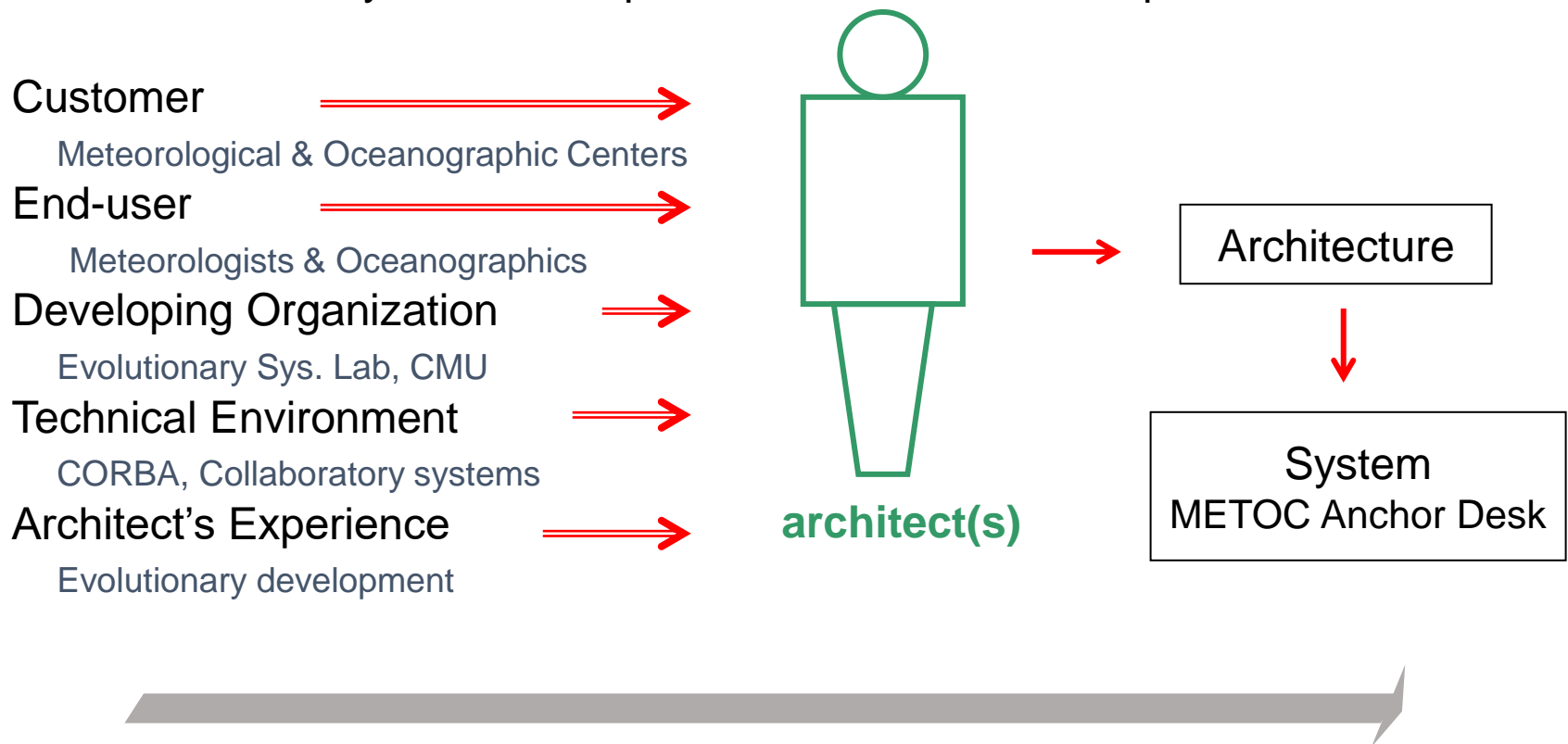


source: adapted from SAP

Influences on the Architect(s) and Architecture (cont.)

Example:

Meteorological and Oceanographic (METOC) Anchor Desk, Ch. 18, SAP
Web-based system developed from off-the-shelf components

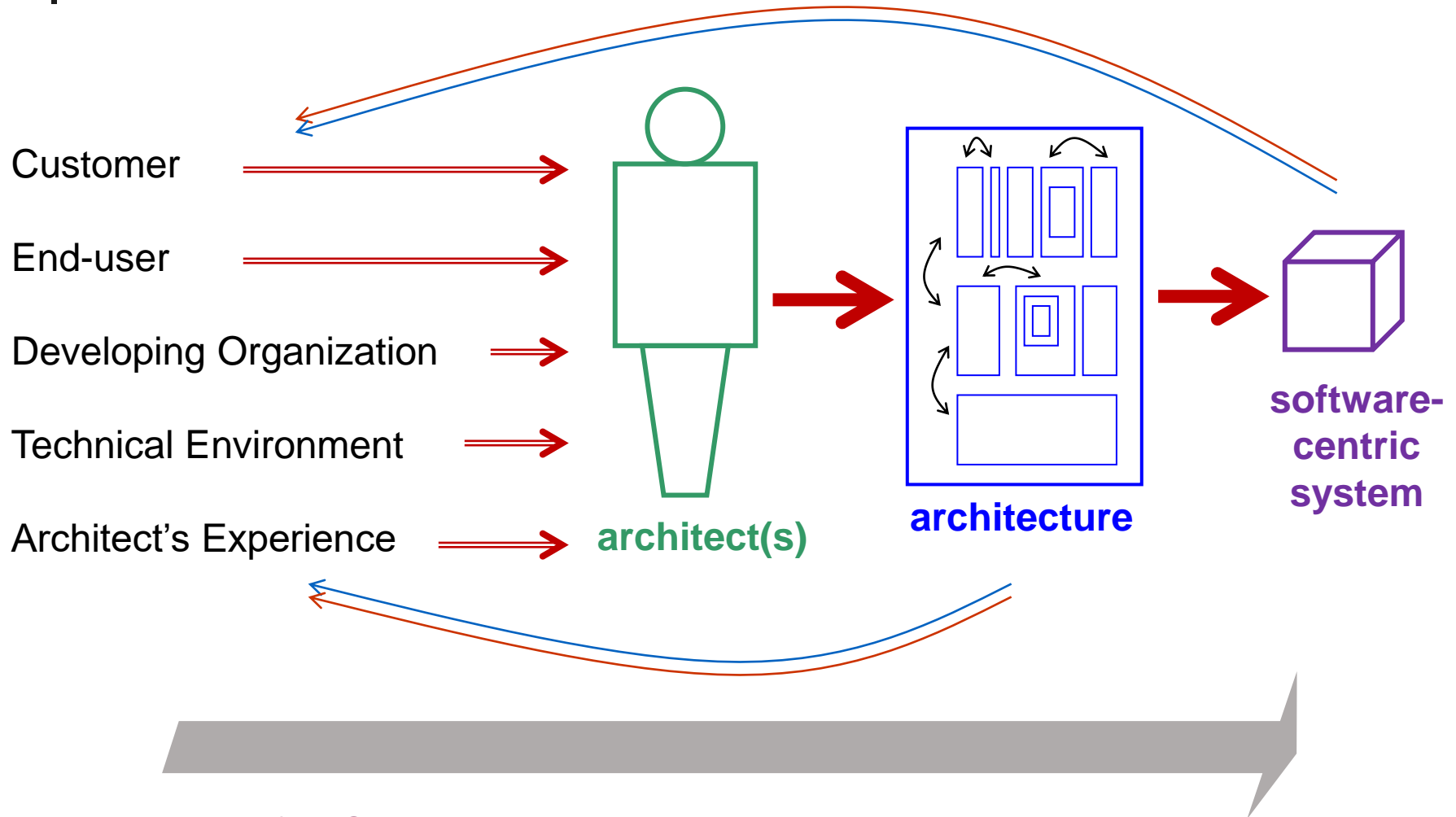


Architect's Skills

Architect's – likes and dislikes

- Education, Practice, and Training
 - **Methods**
 - **Processes**
 - **Tools**
- Experience plays an important role in selecting an architecture from many competing alternatives
 - **Successes & Failures**
- Architect must be able to discover, understand, and formulate often contradictory stakeholders' goals and requirements
 - Technical skill
 - Organizational skill
 - Negotiation and diplomacy
 - Communication – preliminary design review
 - Broad domain knowledge
 - Common sense

The Architecture Business Cycle



source: adapted from SAP

SE Process and Architecture Business Cycle

- Some parts of software engineering process are intuitive – **magic boxes do not exist!**
 - Creating requirements from an idea (e.g., Point of Sale system)
 - Transforming requirements into an architectural design
- A philosophy
 - **Key focus**
 - Motivated workforce – attractive working environment, compensation, ...
 - Competitiveness – tactics and strategies to gain market share
 - Quick time to market
 - **Supplementary focus – “just-good-enough”**
 - Integrability with other vendors
 - Reliability, configurability, etc.
 - Advancing Software Engineering at the expense of profit, ...

Architecture driven activities for creating software/system are

- Creating the **business case** for the system
- **Understanding the requirements**
 - Use-cases, Facilitated Application Specification Techniques, ...
 - Prototype – simulations and/or small-scale prototypes
 - Domain analysis – identify commonalities and variations with similar kinds of systems
- **Creating or selecting the architectures which have conceptual integrity**
 - Architectural styles/design patterns
 - Quality driven and use of prescribed methods and processes

Architecture-Centric Software Steps (cont.)

- Representing and communicating the architecture
 - Representations (texts, diagrams, formal languages, prototypes, ...) should be accessible, informative, and unambiguous to many people with varied background
 - Architect(s) are responsible for behavioral, performance, and quality requirements – i.e., communication is essential
- Analyzing or evaluating the architecture
 - **Non runtime properties** – maintainability, ...
 - **Runtime properties** – performance, behavior, communication, utilization, ...
 - *Architecture Description Languages*
- Implementing software based on its prescribed architecture
 - Suitable infrastructure/environment – tools, standards, ...
- Ensuring that the implementation conforms to the architecture
 - Rigorous supervision and inspection is needed

Choices of Software Architectures

- For a given technical set of requirements, we can expect as many architectures as there are architects – many factors influence the outcome!
- Per se, the **“goodness”** of an architecture is imprecise – a given architecture (client/server) is not inherently the worse or the best choice; it depends on a range of constraints
 - Client/server works well when there exists a large number of independent users
 - Client/server may be unsuitable for hard real-time applications (e.g., critical mission control)
- Nonetheless, **quality** of an architecture can be assessed.

Software Architecture

An architectural skeleton/structure

- specifies **relationships** & **constraints** among **major software components**
 - a hierarchical structure of software components,
 - the ways components interact with one another, and
 - the structure of data
- Known “**design patterns**” are usually used (e.g., Observer also some refer to it as “Publisher-Subscriber”)
- Some common patterns provide a basis for **general architectural designs** that can be specialized to the particulars of a software engineering problem

Good Software Architecture: Some Recommendations

- One – or a few architects (with one being the lead) – should be responsible for the architecture
- Architect (architecture team) should be highly competent – technically, organizationally, leadership, communication, ...
- Architecture need to be well documented – use of standard notations is very important
- Architecture must be analyzed
 - Quantitative measures – performance, correctness, ...
 - Qualitative properties – adaptability, integrability, ...
- ...

- Architecture Business Cycle (ABC) involves a variety of stakeholders and thus conflicting requirements, expectations, etc.
- Non-technical factors are as important if not more as compared with the technical factors
- The process starting from the stakeholders to architects, to architecture, to system, and back to stakeholders is an autocatalytic cycle
 - Autocatalytic cycle is a metaphor for a collection of mutually reinforcing relationships that is capable of sustained existence once each is established and a critical mass has been achieved.

References

- *Object-Oriented Analysis and Design with Applications, 3rd Edition, G. Booch, et. al. Addison Wesley, 2007*
- *Software Architecture in Practice, (SAP), Bass, L. Clements, P., and Kazman, R., Addison Wesley, SEI Series in Software Engineering, 1998*
- *Software Architecture in Practice, 2nd Edition, (SAP2), Bass, L. Clements, P., and Kazman, R., Addison Wesley, SEI Series in Software Engineering, 2003*
- *Software Architecture in Practice, 3rd / 4th Editions, (SAP3, SAP4), Bass, L. Clements, P., and Kazman, R., Addison Wesley, SEI Series in Software Engineering, 2012 / 2021*
- *Eclipse, <http://www.eclipse.org/>, 2022*