Introduction to Software Architecture and Design

Introduction to Software Architecture and Design

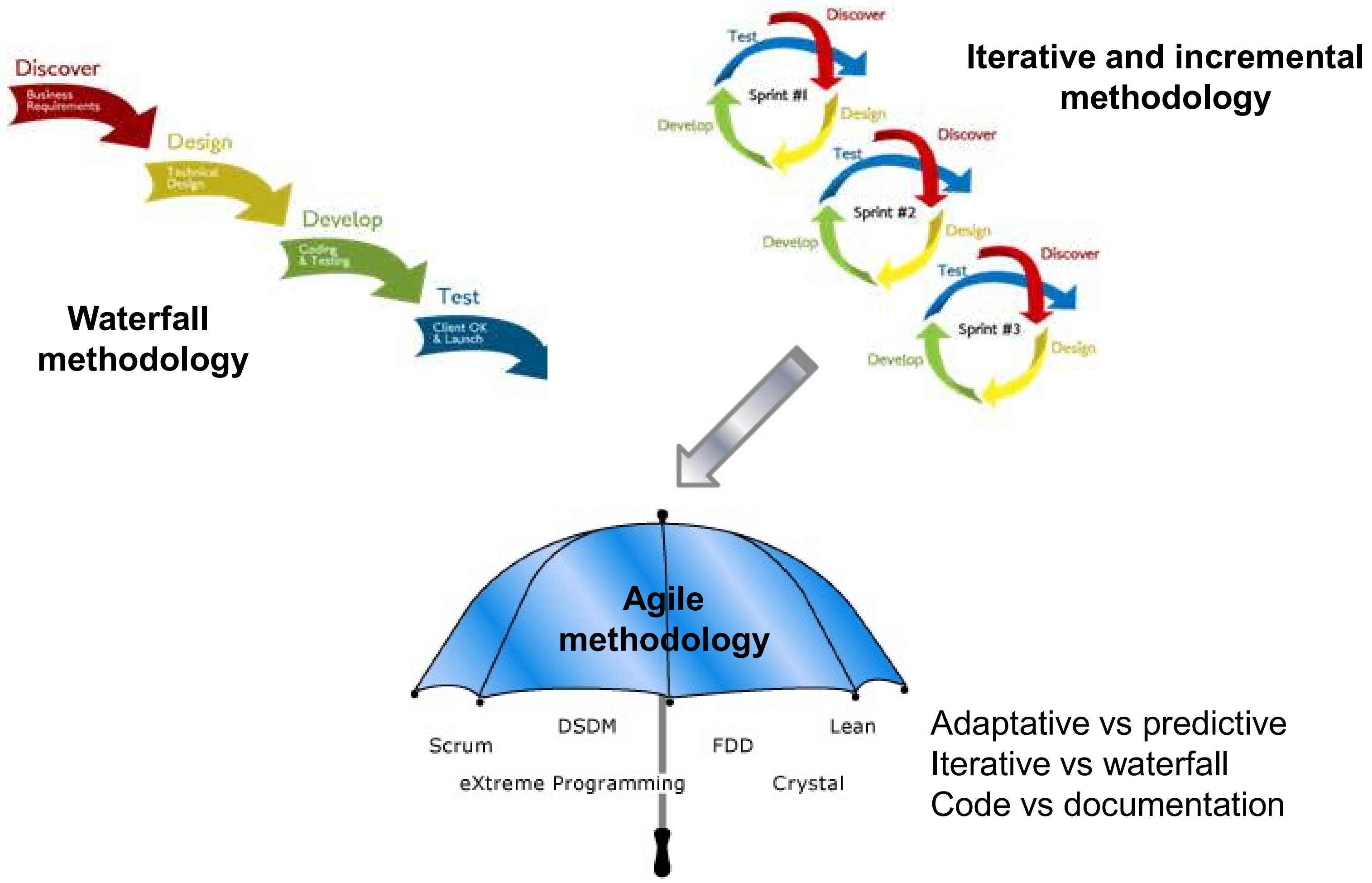
- Software Development Methodologies
- Software Architecture and Design
- Logical and Physical Architecture
- Role of Design Patterns in Software Design
- Software Architecture and Design in Traditional and Agile Methodologies
- References

Software Development Methodologies

- **Software development methodologies** are a specific collection of principles and/or practices applied to develop a software system.

- Methodologies may include the pre-definition of specific deliverables and artifacts that are created and completed by a project team to develop or maintain an application.

Software Development Methodologies



Adaptative vs predictive
Iterative vs waterfall
Code vs documentation

*Images extracted from: http://www.commonplaces.com/blog/agile-v-waterfall-how-to-approach-your-web-development-project/,

Software Design and Architecture

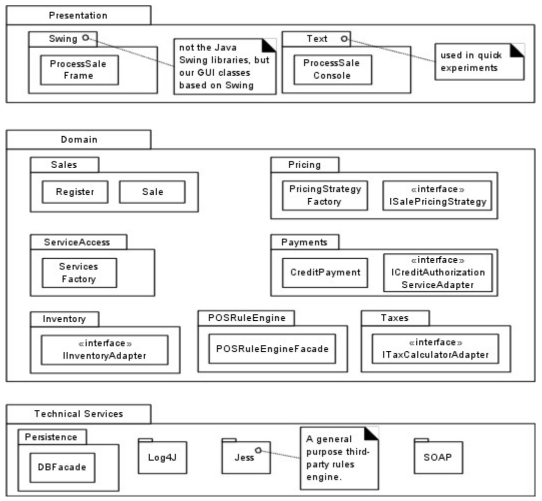
- **Software design** is the activity of applying different techniques and principles in order to define a system up to the level of detail needed to physically build it (i.e., implement it). One of the outputs of the software design is the software architecture.

- **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. (P. Clements; F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, P. Merson, R. Nord, J. Stafford (2010). *Documenting Software Architectures: Views and Beyond*. Addison-Wesley)

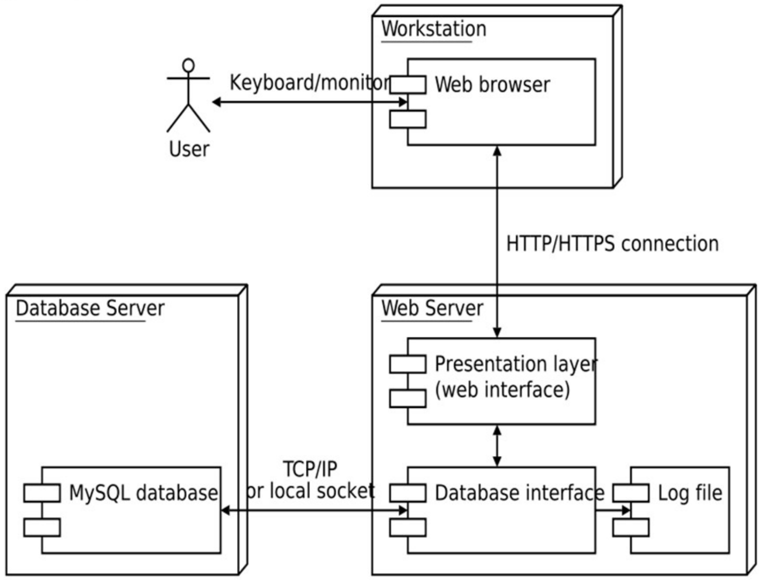
Logical Architecture and Physical Architecture

- **Logical architecture** is the large-scale organization of the software components into packages, subsystems and layers that logically separate the functionality of a software system.
- **Physical architecture** is the organization and distribution of the logical architecture across different computational nodes in a network.

Logical Architecture



Presentation
Swing
ProcessSale Frame
not the Java Swing libraries, but our GUI classes based on Swing
Text
ProcessSale Console
used in quick experiments
Domain
Sales
Register
Sale
Pricing
PricingStrategy Factory
«interface» ISalePricingStrategy
ServiceAccess
Services Factory
Payments
CreditPayment
«interface» ICreditAuthorization ServiceAdapter
Inventory
«interface» IInventoryAdapter
POSRuleEngine
POSRuleEngineFacade
Taxes
«interface» ITaxCalculatorAdapter
Technical Services
Persistence
DBFacade
Log4J
Jess
A general purpose third-party rules engine.
SOAP



Physical Architecture

Role of Design Patterns in Software Design

- **Design patterns** are general reusable solutions to commonly occurring problems within a given context in software design.

- Design patterns may be used in traditional and agile methodologies.

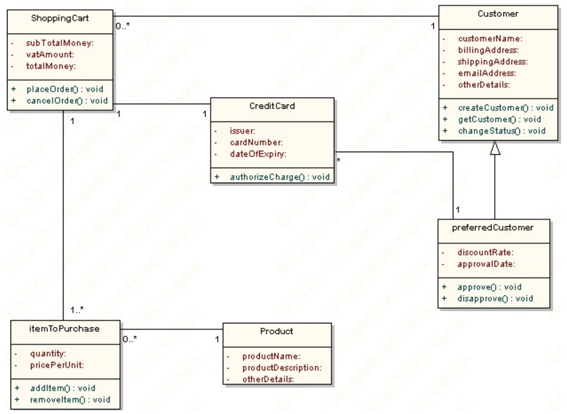
- Two types of patterns used at the design phase:
  - Architectural patterns
  - Design patterns

Role of Design Patterns in Software Design

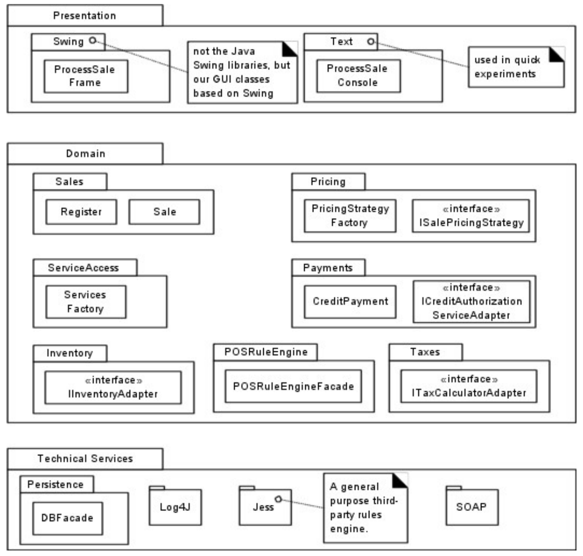
- **An architectural pattern** expresses a fundamental structural organization or schema for software systems. It provides a set of predefined subsystems, specifies their responsibilities, and includes rules and guidelines for organizing the relationships between them. (F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stad. *Pattern-Oriented Software Architecture: A System of Patterns*. Wiley)

- Architectural patterns describe different aspects of applications (deployment aspects, structure and design issues and others communication factors). A typical software system will use a combination of more than one architectural patterns.

Role of Design Patterns in Software Design: Architectural Patterns for the Logical Architecture

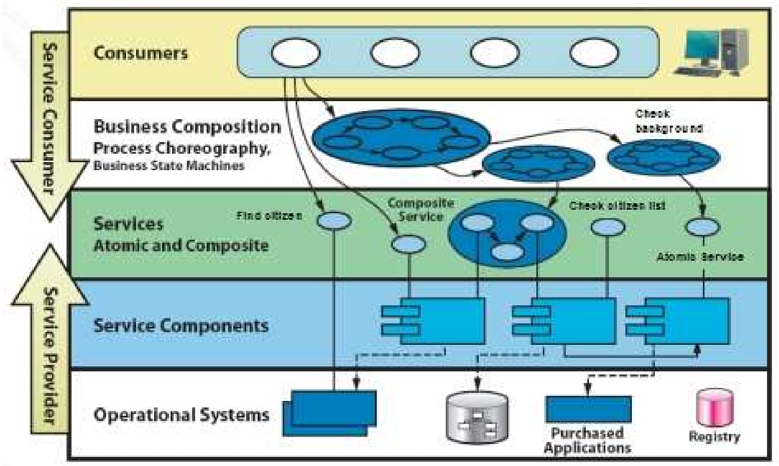


Object-Oriented Architecture



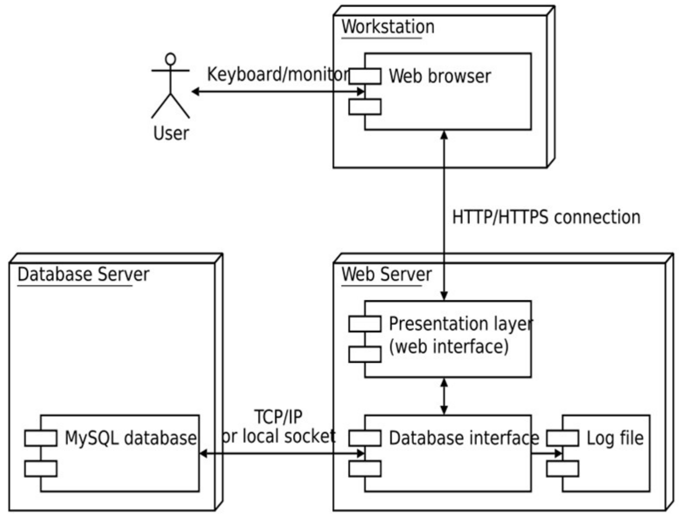
Presentation
Swing
ProcessSale Frame
not the Java Swing libraries, but our GUI classes based on Swing
Text
ProcessSale Console
used in quick experiments
Domain
Sales
Register
Sale
Pricing
PricingStrategy Factory
«interface» ISalePricingStrategy
ServiceAccess
Services Factory
Payments
CreditPayment
«interface» ICreditAuthorization ServiceAdapter
Inventory
«interface» IInventoryAdapter
POSRuleEngine
POSRuleEngineFacade
Taxes
«interface» ITaxCalculatorAdapter
Technical Services
Persistence
DBFacade
Log4J
Jess
A general purpose third-party rules engine.
SOAP

Layered Architecture

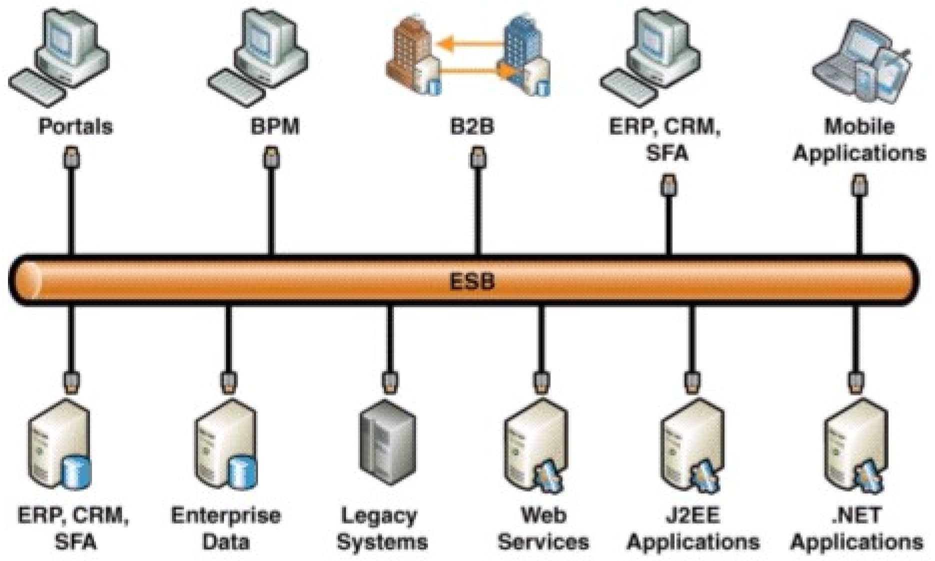


Service-Oriented Architecture

Role of Design Patterns in Software Design: Deployment Patterns for the Physical Architecture



N-Tier/3-Tier Architecture

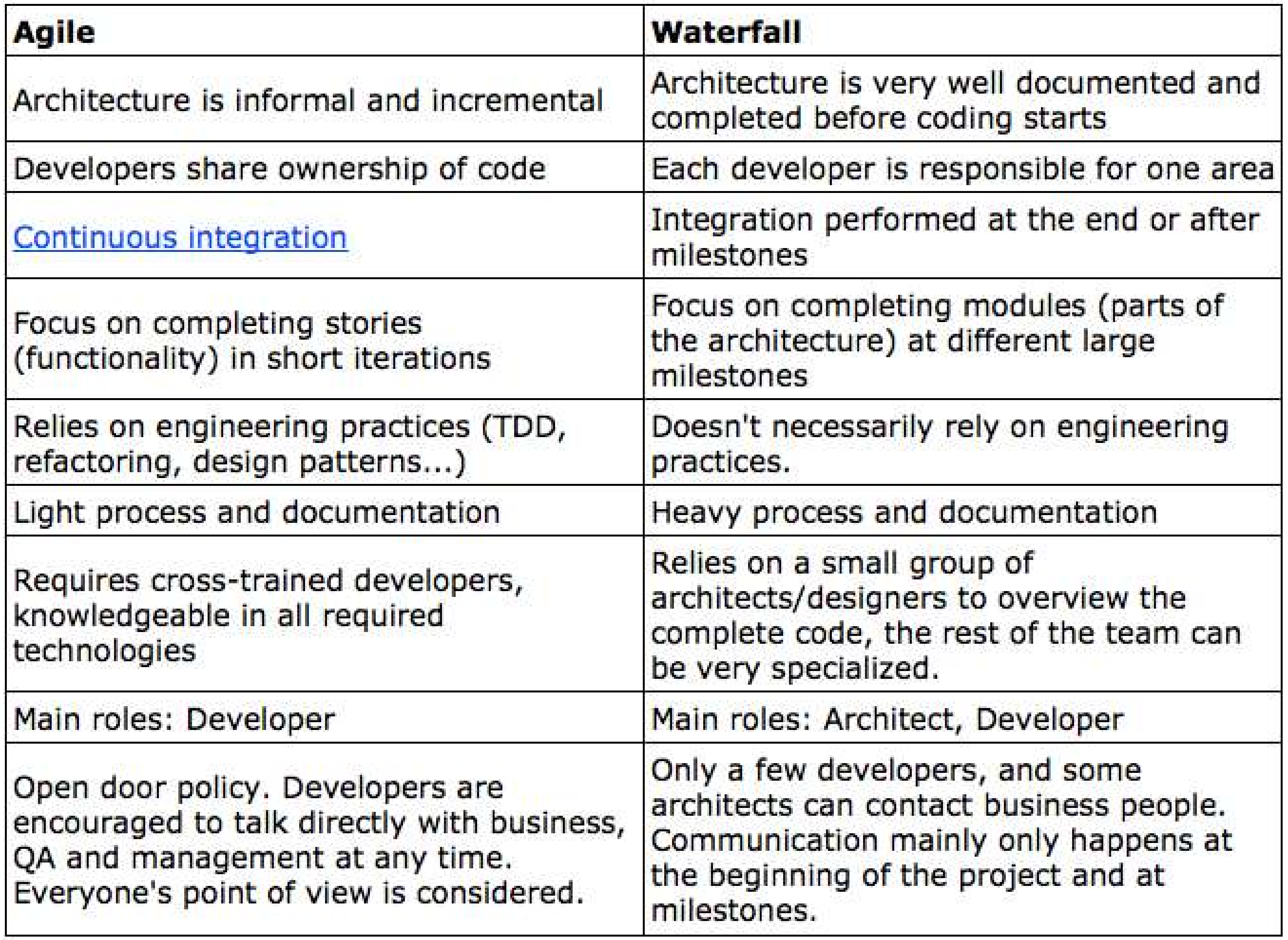


Message Bus

Role of Design Patterns in Software Design

- **A design pattern** provides a scheme for refining the subsystems or components of a software system, or the relationships between them. It describes a commonly recurring structure of communicating components that solves a general design problem within a particular context. (F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stad. *Pattern-Oriented Software Architecture: A System of Patterns*. Wiley)
- Some design patterns
  - State
  - Expert
  - Controller
  - …

Software Architecture and Design in Traditional and Agile Methodologies



Agile	**Waterfall**
Architecture is informal and incremental	Architecture is very well documented and completed before coding starts
Developers share ownership of code	Each developer is responsible for one area
Continuous integration	Integration performed at the end or after milestones
Focus on completing stories (functionality) in short iterations	Focus on completing modules (parts of the architecture) at different large milestones
Relies on engineering practices (TDD, refactoring, design patterns...)	Doesn't necessarily rely on engineering practices.
Light process and documentation	Heavy process and documentation
Requires cross-trained developers, knowledgeable in all required technologies	Relies on a small group of architects/designers to overview the complete code, the rest of the team can be very specialized.
Main roles: Developer	Main roles: Architect, Developer
Open door policy. Developers are encouraged to talk directly with business, QA and management at any time. Everyone's point of view is considered.	Only a few developers, and some architects can contact business people. Communication mainly only happens at the beginning of the project and at milestones.

*Table extracted from: https://dzone.com/articles/waterfall-vs-agile-development-business

References

- *Ingeniería del software. Un enfoque práctico*
  R.G. Pressman
  McGraw Hill, 2010 (Séptima edición), cap. 8, 9 and 10

- *Enginyeria del software: Especificació*
  D. Costal, X. Franch, M.R. Sancho, E. Teniente
  Edicions UPC, 2004

- *Applying UML and Patterns*
  C. Larman
  Prentice Hall, 2005 (3rd edition), ch. 33, 34 and 39

- Microsoft Application Architecture Guide (2nd edition)
  Microsoft
  http://msdn.microsoft.com/en-us/library/ff650706.aspx, ch. 1,2 and 3

- Is Design Dead?
  M. Fowler
  http://martinfowler.com/articles/designDead.html