9th lecture – Design of computer system architecture

Agenda

✓ General considerations on computer system design
✓ The principle of the gradual design of information systems
✓ Design of computer system architecture
   A. Client-server architecture
   B. Distributed architecture
   C. Service oriented architecture
General considerations on computer system design

• Computer system design consists of:
  ➢ Establishing the **logical design solutions** and
  ➢ **Physical specifications** for the new system components;

• The design process is mainly based on the results of the previous activities:
  ➢ Defining the **implementation solution** for the new system
  ➢ **Modelling of the new system**
Design process objectives

Analysis activities

- **Objectives**: to understand
- Events and processes of the company
- System activities and processing requirements
- Information requirements and storage

Design activities

**Objectives**: To define, organize and structure the components of the solution system
General considerations on computer system design

• Some methodologies divide system design in two parts:
  A. Architectural design/preliminary design
  B. Detailed design includes:
     i. Use case design
     ii. Database design
     iii. User interface and external system interface design
     iv. System control and security design

• Within these stages, the computer system is logically and physically designed, separately or not.
The principle of the gradual design of information systems

- The following criteria may be taken into account when determining the order of priority in approaching the structures of the information system:
  - Priority of component objectives;
  - Providing connections between components;
  - Availability of resources
    - The limit of funds that can be allocated in time for the implementation of the IT system;
    - The level of technical equipment existing in the conception phase and the level to be reached by the project end;
    - The design forces that the project will involve;
    - The existing and in-service specialists, of the beneficiary unit, necessary for the current implementation and operation of the IT system.
## Key questions for design activities

<table>
<thead>
<tr>
<th>Design activity</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>i. Execution environment design</td>
<td>Have we specified in detail the environment and all the various options in which the software will execute?</td>
</tr>
<tr>
<td>ii. Application and software architecture design</td>
<td>Have we specified in detail all the elements of software and how each use case is executed?</td>
</tr>
<tr>
<td>iii. Design system interfaces with external systems</td>
<td>Have we specified in detail how the system will communicate with all other systems inside and outside the organization?</td>
</tr>
<tr>
<td>iv. Database design</td>
<td>Have we specified in detail all the information storage requirements, including all the schema elements?</td>
</tr>
<tr>
<td>v. Design of system controls and security</td>
<td>Have we specified in detail all the elements to ensure the system and the data are secure and protected?</td>
</tr>
</tbody>
</table>
i. Design the environment

- **THE ENVIRONMENT** represents all of the technology required to support the software application
  - Servers, desktop computers
  - Mobile devices, operating systems
  - Communication capabilities, input and output capabilities
- It is also called the Technology architecture
ii. Design the application architecture and software

- Partition system into subsystems
- Define software architecture
  - Three level or
  - Model-view-controller
- Detailed design of each use case
  - Design class diagrams
  - Sequence diagrams
  - State machine diagrams
iii. Design user interfaces

- Dialog designs **begins with requirements**
  - Use case flow of activities
  - System sequence diagrams
- Design adds screen layout, navigation, look and feel, user experience
- Now we require interface design for many **different environments and devices**:  
  - Smart phones
  - tablets, iPads, etc
iii. Design system interfaces

- Computer system interacts with many other systems, internal and external; integration?
- System interfaces can connect with other systems in many ways:
  - Save data another system uses;
  - Read data another system saved;
  - Real time request for information;
  - Software services.
Identifying system interfaces

- Inputs and outputs with minimal human intervention, highly automated
  - these are captured by devices (scanners, sensors etc) or generated by persons who start a process that proceeds without further human intervention
- Inputs from and outputs to other systems:
  - these are direct interfaces with other information systems, normally formatted as network messages
- Inputs and outputs to external database:
  - these can supply input or accept output from a system
iv. Database design

- Starting with the **domain class model** (or ERD)
- Choose **database structure**
  - Usually relational database
  - Could be ODBMS framework or other NoSQL systems
- Design **DB architecture** (distributed, etc)
- Design **database schema**
  - Tables and columns in relational, usually
- Design referential **integrity constraints**
  - Foreign key references/ other types of constraints
v. Design security and controls

- Protect the organization assets
- Becomes crucial in Internet and wireless environment
- We need:
  - User interface controls;
  - Application controls;
  - Database controls;
  - Network controls.
Analysis models and Design models

ANALYSIS
- Class diagrams
- Use case diagrams
- Sequence diagrams
- Use case descriptions
- State machine diagrams
- Activity diagrams

DESIGN
- Package diagrams
- Design class diagram
- Sequence diagrams
- Database schema
- User interface
- Component diagram
- Deployment diagram
- System security and control

1. Design for Internal deployment
   a. **Stand alone** software systems - run on one device without networking
   b. **Internal network-based** systems:
      - Local area network, client-server architecture
      - Desktop applications and browser-based applications
   c. **Three layer client-server architectures**:
      - Data layer, domain layer and presentation/view layer
      - Desktop applications and browser-based applications
2. **Design for External deployment**
   - Configuration for *Internet deployment*– with advantages and risks
   - Hosting alternatives for *Internet deployment*
     - Colocation
     - Managed services
     - Virtual servers
     - Cloud computing
   - Diversity of client devices with Internet deployment
     - Laptop, tablets and notebooks, smart phones etc
Configuration for Internet deployment

- **Advantages**
  - **Accessibility** – Web-based applications are accessible to a large number of potential users (including customers, suppliers, off-site employees)
  - **Low cost communication**
  - **Widely implemented standards** – Web standards are well known

- **Potential problems:**
  - **Security** – Web servers are target for security breaches because Web standards are open and widely known, accessible to hackers
    - **https** – Hypertext Transfer Protocol Secure
    - **TLS** – Transport Layer Security – Advanced ver. of SSL protocol
  - **Throughput** – When high load occur, throughput and response time can suffer significantly.
Design for remote, distributed environment

- Two interfaces to same Web applications for internal / external access
  - Back end, front end user interface to same Web app
  - Not as secure

- Virtual private network (VPN)
  - Closed network with security and closed access, built on top of a public network (Internet)
User Interface Layer

Browser (with cookies)

request/input data

Internet

reply

Domain Layer
(Business Logic)

Common Gateway Interface (CGI)

Application Server (session mgr)

Response Page

frameset Page

JavaScript
VBScript
Applet
ActiveXControl

PHP
ASP
JSP
Servlets
ColdFusion
Design of system architecture

- The most popular types of networks are:
  - Bus/point-to-point network;
  - Ring network;
  - Star network;
  - Hierarchy network.

- After selecting the network type, the analysis team must identify the communication protocol. The most common communication protocols are:
  - TCP/IP. It is a protocol indicated when using an Ethernet network or when the network computers have different architectures;
  - SNA. It is generally used to connect IBM mainframes.
A. Client/server architecture

- The client / server architecture is a set of three main components: a server, a client, and a network that connects client computers to servers to collaborate on performing tasks.

- The types of client / server applications are:
  - Database systems
  - E-mail systems
  - Groupware systems
  - Legacy systems

- A distributed architecture involves the existence of multiple databases (located on separate computers) and of applications that manipulate data from different local workstations using database management systems (DBMSs).
A. Design for three layer client-server architectures

- Data layer, domain layer and presentation/view layer

- Desktop applications and browser-based applications
- One of the advantages of the client-server architecture is that it easily support software to be developed using three layer architecture applications
Model View Controller (MVC)

- It is a software architectural design for implementing user interfaces and Web frameworks on computers and is a standard design pattern.
- MVC architecture helps to write better organized and more maintainable code.
- This architecture is used and extensively tested over multiple languages (Java, PHP, ASP.NET etc.) and generations of programmers.
- MVC is being used as the powerful framework for building web applications using MVC pattern.
- MVC separation helps to manage complex applications. It is the main advantage of separation and also simplifies the team development.
MVC Model-View-Controller

- Controller is *in charge*, taking care of the data received from Model and injecting it into View

**Model**: central component of MVC, it manages the *data, logic and constraints* of an application. It captures the *behavior* of an application.

**View**: responsible for displaying all or a portion of the data to the user. It formats and presents the data from model to user.

**Controller**: controls the interactions between the Model and View. It acts as an interface between the associated models, views and the input devices.
B. Distributed architectures – Data distribution

a) *Distribution by fragmentation:*

- **Rules:**
  - Completeness
  - Reconstruction
  - Disjointness

- **Method:**
  - Horizontal method: partitions a relation along its tuples
  - Vertical method: partitions a relation along its attributes
  - Mixt method: a combination of horizontal and vertical methods
Data distribution

b) Distribution by replication

- This solution is useful when users from different nodes in the network need information simultaneously, and the database is locally distributed.

- Replication design can be realized:
  - **Fully replicated**: each fragment at each site
  - **Partially replicated**: each fragment at some of the sites
  - **Non-replicated DB (= partitioned DB)**: each fragment resides at only one site
DATA DISTRIBUTION

c) Mixed distribution

d) Distribution by loading

- The technique is used when the data is stable, so it is rarely updated, or when not all users need to have access to the latest data.

- After selecting the distribution type, specific elements will be designed.
C. SOA- Service Oriented Architecture

- Modern applications are built from **software components** based on **interaction standards**:
  - Common Object Request Broker Architecture (CORBA),
  - Simple Object Access Protocol (SOAP), and
  - Java Platform Enterprise Edition (Java EE).

- Each **standard** defines:
  - specific ways in which components locate and communicate with one another.
  - a set of supporting system software to provide needed services (maintaining component directories, enforcing security requirements, and encoding/decoding messages).

- SOA is based on a **classic request/response pattern**: a service consumer invokes a service provider over the network and waits until the provider has finished the operation.
SOA- Service Oriented Architecture

- SOA divides an application in two parts:
  - A service coordinator, representing the user functionality, and
  - Service providers, that implements the functionality.

- While service coordinator tends to be unique for a particular application, a service can be reused and shared by a multiple composite applications.

- The service coordinator explicitly specifies and invokes the required services.
SOA services

- **Services** are autonomous platform-independent entities that enable access to one or more capabilities, which are **accessible by provided interfaces**.
- A **new designed service** has to meet business requirements that are traditionally specified by a **Business Process Diagram (BPD)**.
- Transform the BPD (in BPMN, see figure) into **UML service diagrams**.
- 2 steps of transformation
Business process model for Purchase Order process
1st step: Identify service invocations

1st Step: identify which tasks from the BPD represent service invocations and therefore will be modeled as services in Service diagrams (5 primitive and 2 composite services)

- A service interacts with its environment via interfaces.
- Service can act two different roles: service provider or service consumer - distinguished by means of a port:
  - provider port
  - consumer port
2nd Step: Transformation process
Behaviour of ProcessPurchaseOrder service as a sequence of service calls (composite)
Behaviour of service ProcessPurchaseOrder in the context of used services and internal processes.