## Defining the System Architecture

### Architectural Concepts

- **Technology architecture** - the set of computing hardware, network hardware, and topology, and system software employed by an organization
- **Application architecture** - the set of information systems (software applications) that the organization needs to support its strategic plan
- The technology architecture defines the infrastructure that supports application software and the services it provides. Application software is deployed on a technology architecture by distributing application components to specific hardware devices and connecting them via networks and protocols.
- Technology and application architecture are interdependent. Poor technology architecture provides a weak foundation for application software and can comprise its performance, reliability, and other characteristics.
- Good technology architecture enhances those characteristics and provides other benefits, such as minimal operating cost and flexible updating. But high-quality technology architecture can't make up for poor application architecture. The application software must be designed to ensure ease of construction, deployment, operation, and future updates.

### Software

- The **system software** is a software that works behind the scenes to support application software and control or interact with hardware or software resources. Examples include the following:
    - Operating systems (OSs)
    - Database management systems
    - Web browsers and Web server software
- The **application software** is software that performs user- or business-specific tasks. It is typically constructed as an app or Web-based application such as shopping, preparing customer purchase invoices, generating monthly financial statements, or enabling a user to play games against the computer or other users.

### Software as a Service

- A *service* is something that we purchase rather than making or doing it ourselves. For example, we consider our household utilities such as water, electricity, and trash pickup to be services.
- *Software as a service* (SaaS) is a software delivery model similar to a utility, in which application software is accessed via the Internet without locally installed programs
- SaaS follows that same basic idea. If an organization requires some services, for example, bookkeeping and accounting functions, it could build or buy an accounting software system. Alternatively, it could find a firm that provides accounting services and buy only the accounting services it needs
- Many applications employ the SaaS model. Examples include the following:
    - Social networking services such as Facebook, Tumblr, and Instagram
    - Application software suites such as Google Apps and Adobe Creative Suite
    - Specialized applications such as Lazada and Shopee shopping applications

### Protocols

- A **protocol** is a set of languages, rules, and procedures that ensure accurate and efficient data exchange and coordination among hardware and software components
- Modern information systems rely on hundreds or thousands of protocols. Hardware, software, and network components abide by protocols because their owners and users value the results of accurate and efficient data exchange and coordination.
- **Network protocols** enable accurate message transmission among the various computers and software components.
- A network protocol is an established set of rules that determines how data is transmitted between different devices in the same network.
- **Web protocols**: The World Wide Web is built on a small family of protocols for encoding Web documents and hyperlinks, requesting documents from Web servers, and

responding to those requests. The most important protocols include the following:

- **Hypertext Markup Language** (HTML) is a protocol that defines the structure and content of a Web page.
- **Extensible Markup Language** (XML) is an HTML extension that enables the meaning of words, phrases, or numbers to be defined.
- **Hypertext Transfer Protocol** (HTTP) is a protocol that defines the format and content of requests for Web documents and related data communication.

**Web Services**

- A **web service** is a software service accessed over the Internet using Web protocols
- Web services enable software functions developed by organizations to be embedded within the information system of another organization.
- For example, consider the shipping options from lazada.com.ph purchase, as shown in *Figure 1*
- Shipping companies provide Web services that enable companies like Lazada to compute shipping charges. Lazada's Web application passes shipment data to the Web service, and the shipper's software computes the charge and then passes it back to the lazada.com.ph Web application
- In essence, the lazada.com.ph Web application executes the shipper's Web service as a subroutine over the Web.


***Figure 1.*** Shipping option for a Lazada purchase

**Distributed Architectures**

- **Client/server architecture** is a method of organizing software to provide and access distributed information and computing resources. It divides software into two classes: client and server
- A server manages system resources and provides access to these resources through a well-defined communication interface. A client uses the communication interface to request resources, and the server responds to these requests.
- The client/server architectural model can be applied in many ways. A simple example is how desktop computers access a shared printer on a LAN, as shown in *Figure 2*
- An application program on a desktop computer sends a document to a server that dispatches it to a management process for the specified printer.

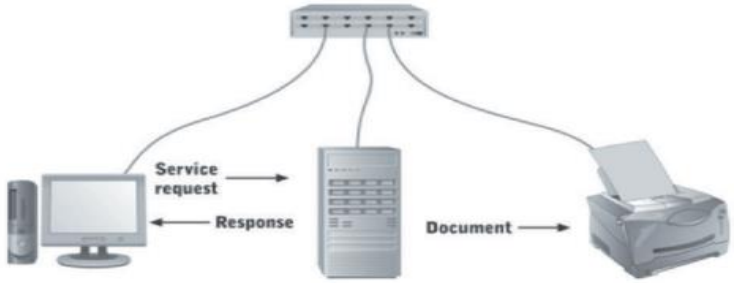


***Figure 2.*** Network printing service implemented with client/server architecture

- **Three-layer architecture** is a variant of client/server architecture used for all types of systems, from internally deployed desktop applications to globally distributed Web-based applications.
- Three-layer architecture divides the application software into three layers that interact, as shown in Figure 3
  - *The view layer* is part of a three-layer architecture that contains the user interface.
  - The *domain layer* contains the programs that implement the business rules and processes.

- The *data layer* is the one that interacts and manages the data.

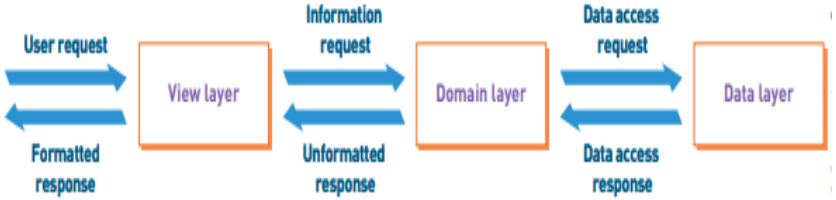


***Figure 3***. Three-layer architecture

**Interoperability**

- *Interoperability* is the ability of a component or system to interact with other components or systems
- To ensure interoperability, system designers must do several things, including the following:
  - Understand and describe the current environment in which the system will operate.
  - Look for existing software components and services that can provide needed functions for the new system.
  - Build components that can't be purchased as software modules or used as a service.
  - Structure and assemble the components in a way that is feasible to build, test, deploy, and operate over the long term.

**Architectural Diagrams**

- *Architectural diagrams* are commonly used to visually describe an information system's environment, components, and deployment
- As with other diagrams used in systems design, architectural diagrams summarize complex details in a way that's easy for people to understand.
- The following sections briefly look at three widely used architectural diagrams:
  - **Location diagrams** are commonly used to show the geographic placement of various system components, including hardware, buildings, and users.
  - **Network diagrams** show how locations and hardware components are interconnected with network devices and wiring.
  - **Deployment diagrams** describe how software components are distributed across hardware and system software components.

**Designing Application Components**

- An *application component* is a well-defined unit of software that performs one or more specific tasks. It hides complex details, including the following:
  - **Size/Scope** - A component could be small, like one method in a class or one script embedded within a Web page.
  - **Programming languages** - Programs, functions, subroutines, and procedures are application components defined by traditional programming languages such as C or FORTRAN. Object-oriented programming languages, such as Java and C#, use classes and methods as application components.
  - ***Build* or *buy?*** - Some parts of an information system are written by an organization's information technology (IT) development staff. Others can be reused from other information systems owned by the organization, purchased separately, or acquired as part of a developer's toolkit. Other components might be deployed by an outside organization and made available via Web services (e.g., Google Maps or Federal Express package tracking service).
- Key decisions made when designing application components include how functions of the system will be grouped or packaged and how they'll interact with one another once built (or acquired) and assembled.

**REFERENCES:**

Tilley, S. (2020). *System analysis and design.* Cengage Learning.
Dennis A., Wixom B., & Roth, R. (2018) *Systems analysis and design 7th Ed.* John Wiley & Sons, Inc.
Satzinger, J., Jackson, and R., Burd, S. (2016). *systems analysis and design in a changing world – course technology.* USA.