Lab 3
System Implementation and Deployment

This lab is concerned with technical features of how to construct and implement a system.

In order to understand what happens in "**System Construction and Implementation**" phase, we need to understand its Context. It consists of two major phases:-

1) **Systems construction:**

The development, installation, and testing of system components.
A common but unfortunate synonym is systems development (more frequently used to describe the entire life cycle.)

Tasks of System Construction:

- Program coding\building:

  - ✓ Build and Test Networks.
  - ✓ Build and Test Databases
  - ✓ Install and Test New Software
  - ✓ Write and Test New Programs
  - ✓ Program and system testing: to ensure that it could be working properly

2) **Systems implementation:**

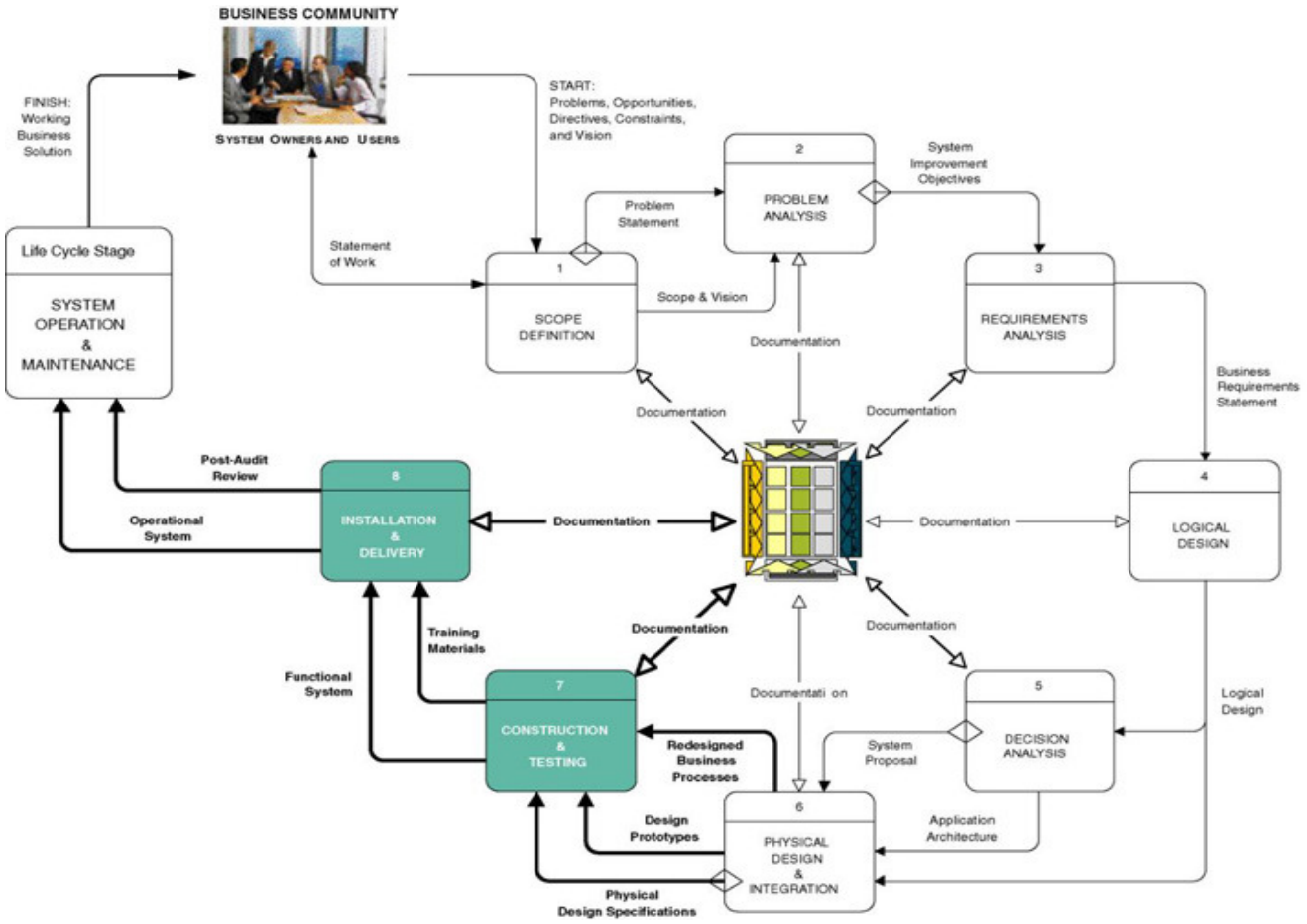
The installation and delivery of the entire system into production. Day-to-day operation.

Tasks of System Implementation:

- Conduction system test
- Prepare Conversion Plan
- Install Database
- Training plan: there is usually a head trainer who will lead the users and direct them on how to use the system by explaining them how to operate it. Sometimes it can be done online, in which they will not need an instructor.

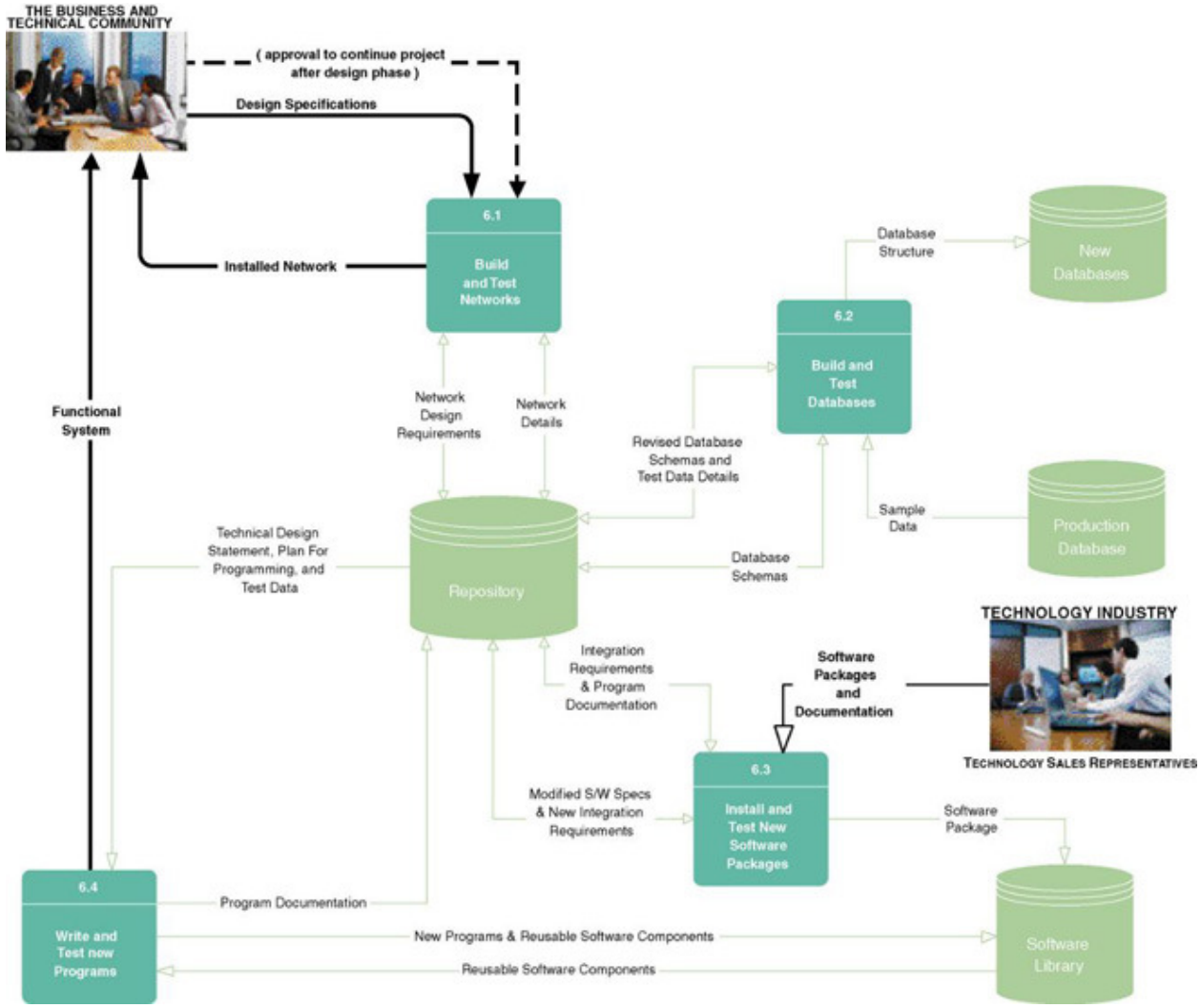
Tasks for completing Construction & Implementation phase:

1-Scope Definition:
Output: Scope, vision and problem statement
2-Problem Analysis:
Output: system objectives
3-Requirment Analysis:
Output: Business Requirements statement
4-Logical Deign:
Output: logical design
5-Decision Analysis:
Output: system proposal and Application Architecture
6-Physical Design & Integration:
Output: Design Prototypes and Physical Design Specification
7-Construction & Testing:
Output: Training Materials and Functional System
8-Installation & Delivery:
Output: Post Audit Review and Operation System



Tasks for Completing the Construction Phase:

1- Build and Test Networks:
Output: Installed Network
2-Build Test Databases:
Output: Database Structure
3-Install and Test New Software:
Output: Software Packages
4-Write and Test New Programs:
Output: Functional system
5- Program and system testing:
Output: List of bugs to be fixed



Construction Phase:

1-Build and test Network:

- Often systems are built around existing networks.
- If system calls for new network functionality, must be built and tested prior to programs that use that it.
- Roles:
  - Network designer
    - Designs LAN and WAN connectivity
  - Network administrator builds and tests
    - Network architecture standards
    - Security
  - Systems analyst
    - Facilitates
    - Ensures that business requirements are not compromised

2-Build and Test Databases:

- Implement database schema
- Test with sample data
- Deliver unpopulated database structure
- Roles
    - System users
        - Provide and/or approve test data
    - Database designer/programmer
        - Build tables, views, stored procedures (if relational database)
    - Database administrator
        - "Tune" database for optimum performance
        - Security
        - Backup and recovery
    - Systems Analyst
        - Ensures business requirements compliance

3- Install and Test New Software:

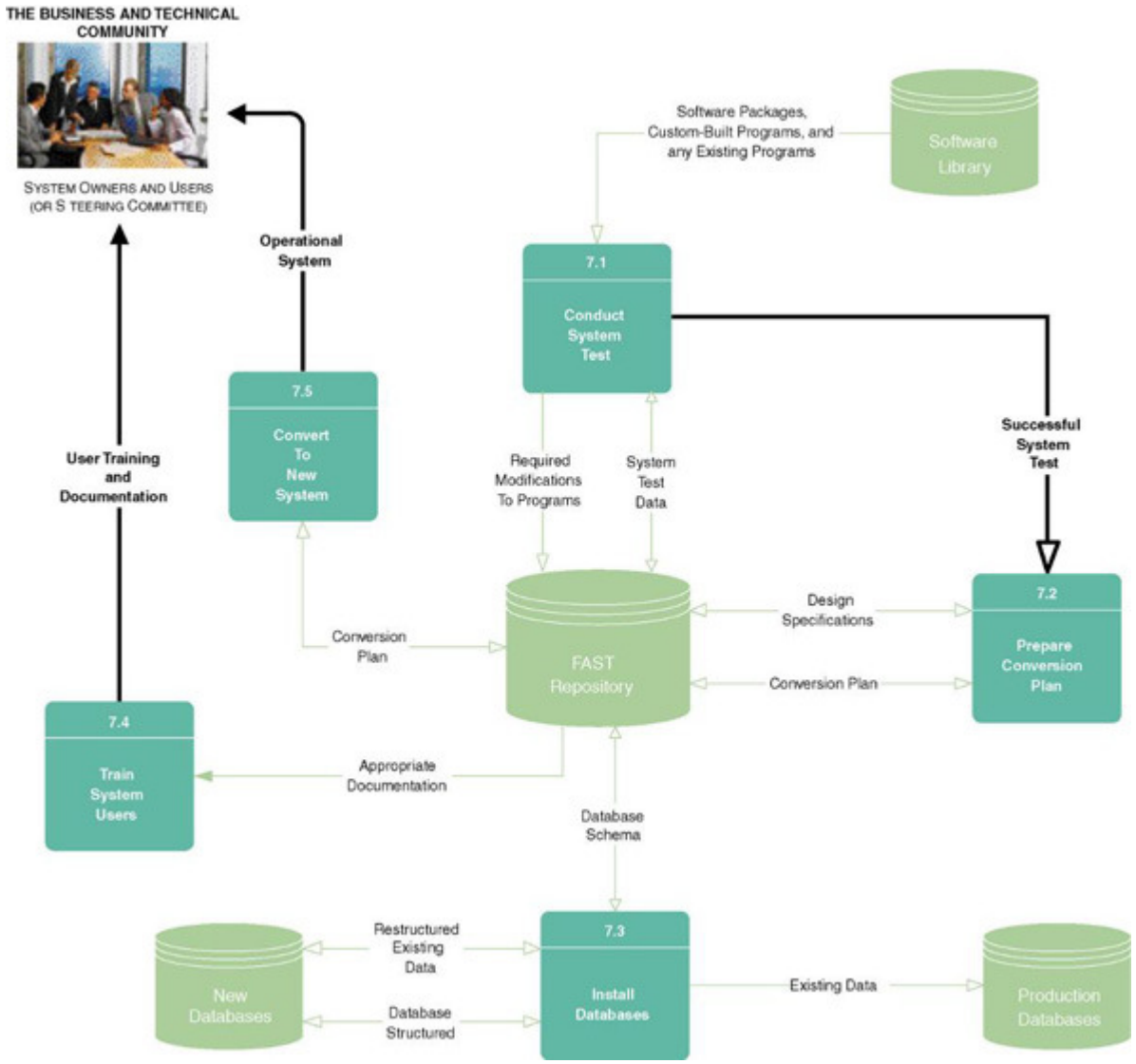
- If the new system requires other purchased or released software, must be installed and tested.
- Roles
    - Systems analyst
        - Clarifies business requirements
    - System designer
        - Clarifies integration requirements between the purchased /released software with current being built system
    - Network administrator
        - Install software package
    - Software vendor/consultant
        - Assist in installation , testing and maintenance
    - Applications programmer
        - Test according to integration requirements

4-Write and Test New Programs:

- Develop in-house programs
    - Reuse available software components in library
    - Write new components
    - Test
    - Document

- Roles
  - Systems analyst
    - Clarifies business requirements
  - System designer
    - Clarifies program design and integration requirements
  - Application programmer (or team)
    - Writes and tests in-house software

Tasks for Completing the Implementation Phase:



Implementation Phase:

1. Conduct System Test:

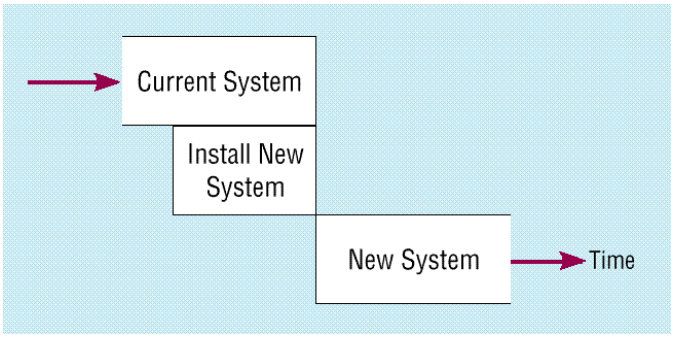
- Test network, databases, purchased software, new in-house software, and existing software to make sure it all works together.
  Note: Next lab we will talk in details about testing and its types
- Roles
  - Systems analyst
    - Develops system test data
    - Communicates problems and issues
  - System builders (database, network, programmers)
    - Resolve problems revealed during testing
    - Fix bugs
  - System owners and users
    - Verify whether or not system operates correctly
- May result in return to construction phase
  - Iterate until successful system test

2. Prepare Conversion Plan:

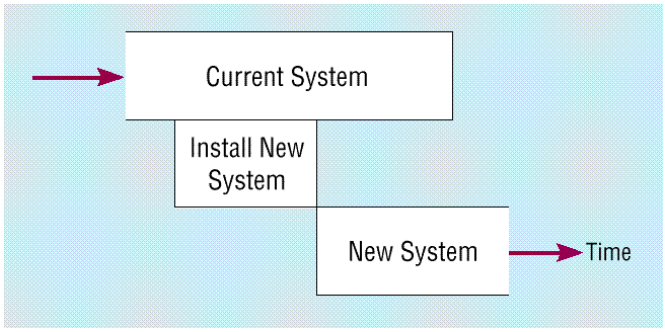
- Conversion strategy: Converting to the new system breaks down into direct implementation, parallel implementation, phased implementation and pilot implementation

- Plan for how to convert from old system to new system.
  - How to install and populate databases
  - How to train users
  - Finalize documentation
  - Conversion issues
- Roles
  - System analyst/Project manager
    - Develop a detailed conversion plan
  - Steering committee
    - Approves plan and timetable

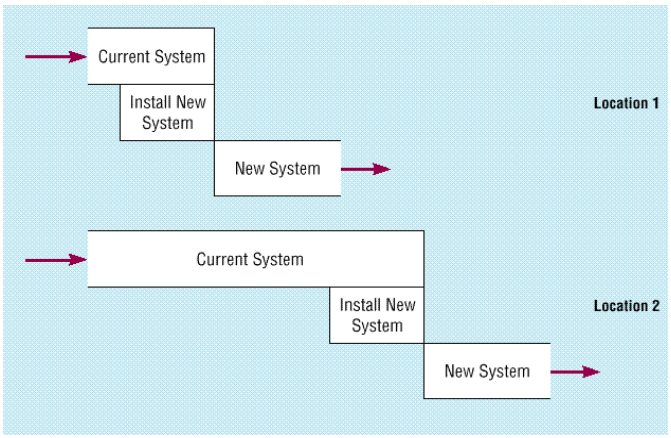
- Installation Strategies
  - Abrupt cutover - Direct Installation



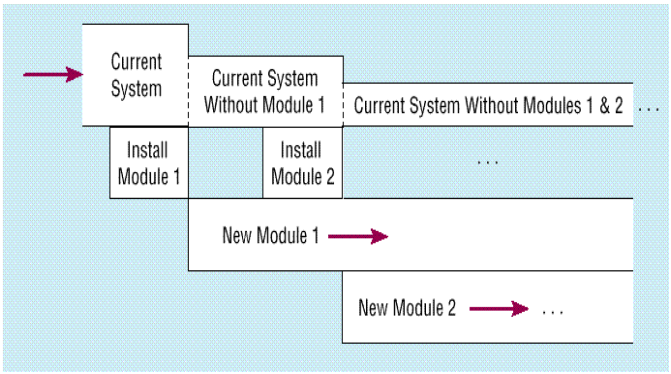
  - Parallel conversion



  - Location conversion - Single Location Installation



  - Staged conversion - Phased Installation



3 .Install Database:

- Populate new system databases with existing data from old system
  - Generally have to restructure data as it is populated
  - Must confirm that data is translated correctly
- Roles
  - Application programmers
    - Write (or use) special programs to extract data from existing databases and populate new databases
  - Systems analyst/designer
    - Calculate database sizes and estimate time for installation

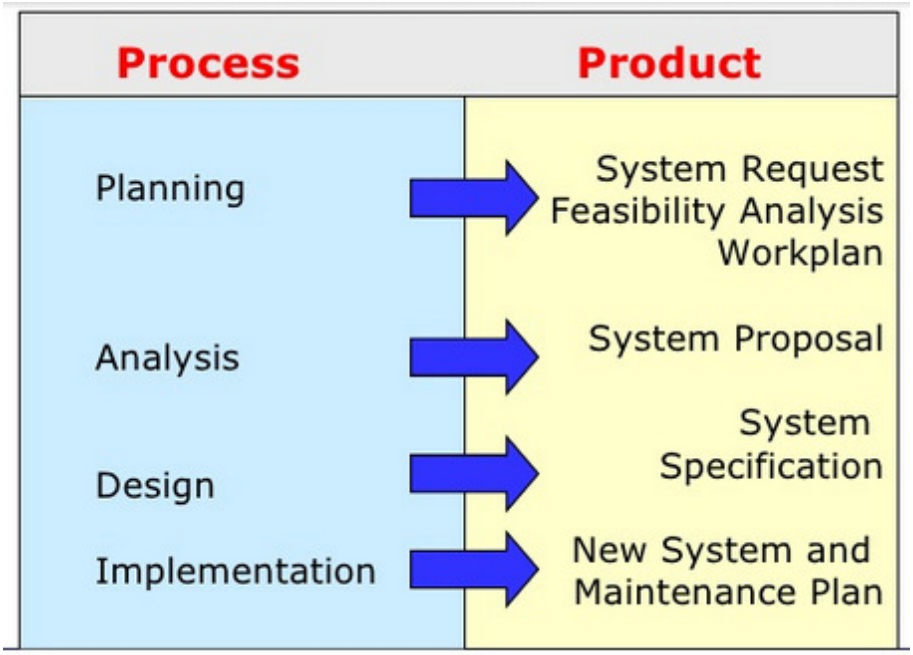
4. Train Users:

- System users trained and provided with documentation
- Outline for a training manual

I. Introduction
II. Manual
A. The manual system (a detailed explanation of people's jobs and standard operating procedures for the new system).
B. The computer system (how it fits into the overall workflow).
1. Terminal/keyboard familiarization.
2. First-time end users.
a. Getting started.
b. Lessons
C. Reference manual (for non-beginners).
III. Appendixes
A. Error messages.

- Roles
  - System analyst
    - Plan trainings
    - Conduct trainings
    - Write documentation
    - Help users through the learning period
  - System owners
    - Approve release time for training
  - System users
    - Attend training
    - Accept system

Processes and deliverables:

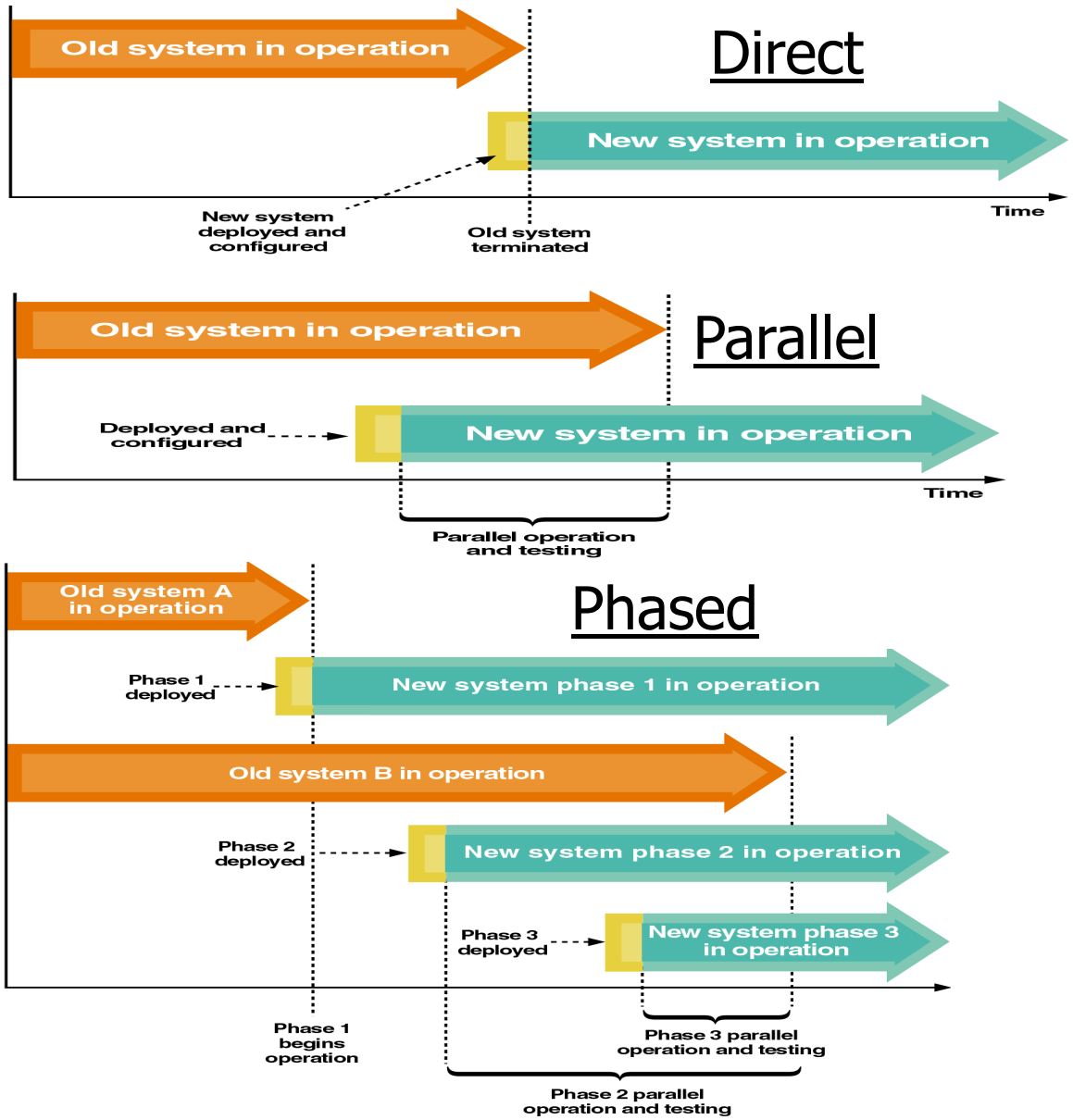


Deployment Phase:

All the activities to make a new system operational

- Includes:
  - Installation and configuration
  - User training
  - Porting and Converting data
  - Deployment strategy

Deployment strategy



Programming Types

1-Web based Languages:

- **HTML**
  - Hyper Text Markup Language.
  - The core language of the World Wide Web that is used to define the structure and layout of web pages by using various tags and attributes. Although a fundamental language of the web, HTML is static - content created with it does not change.
  - HTML is used to specify the content a webpage will contain, not how the page functions.

- **XML**
  - Extensible Markup Language.
  - A language developed by the *W3C* which works like HTML, but unlike HTML, allows for custom tags that are defined by programmers.
  - XML allows for the transmission of data between applications and organizations through the use of its custom tags.

- **JavaScript**
  - A language developed by Netscape used to provide dynamic and interactive content on webpages.
  - With Javascript it is possible to communicate with HTML, create animations, create calculators, validate forms, and more.
  - Javascript is often confused with Java, but they are two different languages.

- **VBScript**
  - Visual Basic Scripting Edition.
  - A language developed by Microsoft that works only in Microsoft's Internet Explorer web browser and web browsers based on the Internet Explorer engine such as FlashPeak's *Slim Browser*.

- VBScript Can be used to print dates, make calculations, interact with the user, and more.
- VBScript is based on Visual Basic, but it is much simpler.

- **PHP**
  - Hypertext Preprocessor (it's a recursive acronym).
  - A powerful language used for many tasks such as data encryption, database access, and form validation.
- **Java**
  - A powerful and flexible language created by Sun MicroSystems that can be used to create applets (a program that is executed from within another program) that run inside webpages as well as software applications.
  - Using Java you can *interact with the user, create graphical programs, read from files, and more*.

2-Procedure-oriented programming:

- A type of programming where a structured method of creating programs is used. With procedure-oriented programming, a problem is broken up into parts and each part is then broken up into further parts. All these parts are known as procedures. They are separate but work together when needed. A main program centrally controls them all.
- Some procedure-oriented languages are COBOL, FORTRAN, and C.

3-Object oriented programming:

- A type of programming where data types representing data structures are defined by the programmer as well as their properties. With object-oriented programming, programmers can also create relationships between data structures and create new data types based on existing ones by having one data type inherit characteristics from another one.
- In object-oriented programming, data types defined by the programmer are called classes (templates for a real world object to be used in a program).

- For example, a programmer can create a data type that represents a car - a car class. This class can contain the properties of a car (color, model, year, etc.) and functions that specify what the car does (drive, reverse, stop, etc.)
- Some object-oriented languages are C++, Java, and PHP.

4-Extreme programming (XP):

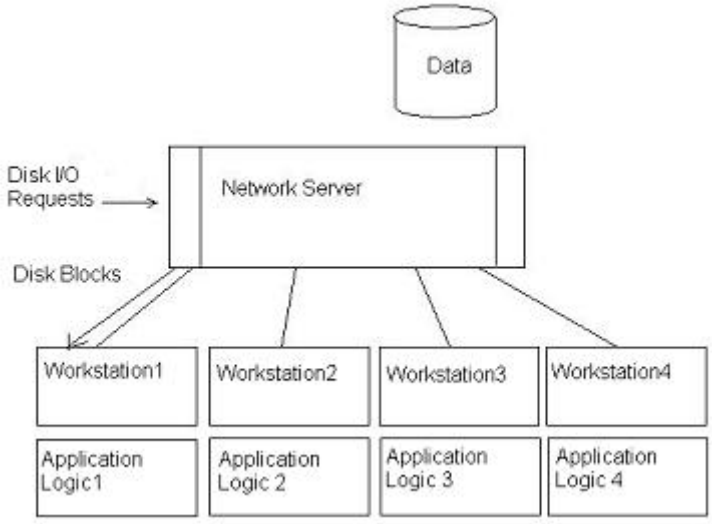
- XP is a software development methodology which is intended to improve software quality and responsiveness to changing customer requirements. As a type of agile software development, it advocates frequent "releases" in short development cycles, which is intended to improve productivity and introduce checkpoints at which new customer requirements can be adopted.
- It works by bringing the whole team together in the presence of simple practices, with enough feedback to enable the team to see where they are and to tune the practices to their unique situation.
- **Why "Extreme"?**

  "Extreme" means these practices get "turned up" to a much higher "volume" than on traditional projects. The result is stable, productive, and very rapid because the practices support each other the more they are used together without interference.

How to choose which programming type is applicable for your chosen architecture? Which programming language?

1-File/Server Architecture

The file server system brought a complete change in implementation of the computer architecture from the mainframe. In this system, the application logic was now executed on the client workstation instead of the server. These servers also provided access to computing resources like printers and large hard drives. The complete File Server Architecture is illustrated in the figure shown below.



- Taking into account the *disadvantages* of the *centralized* system and file server system architectures, the client-server architecture made its advent.
- Any Computer can be configured to be a host and act as a file server.
- In its simplest form: *A file server may be an ordinary PC that handles requests form*

Programming of File\Server:-

- You can use HFS (HTTP File Server) to send and receive files.
- HFS is a small HTTP file server, aimed at mostly sharing files for download
- It is different from classic file sharing, because it uses web technology to be more compatible with today's Internet.
- It also differs from classic web servers, because it's very easy to use.
- Access your remote files, over the network.
- Example: Dropbox

2-Client/Server Architecture

The Server (typically)

- An applications runs on a large computer.
- The server application is usually written in such a way that it does not provide any method to interact with a user. Instead, it waits for other programs to connect (i.e. other programs take the place of a user) and interacts with these programs.
- Typically, a server application has control over large amounts of data, and can access that data fast and efficiently.
- It can also handle requests by many clients (more or less) simultaneously.

The Client (typically)

- An application that runs on a personal computer.
- It has an extensive user interface, but it has no data that it can manipulate or present to the user.
- The client application 'asks' what a user wants, then connects to the server application to obtain that data. Once the data has been obtained, it is presented to the user in a nice format.
- A client usually gets the data from one server at a time, and interacts with one user only.

Example: A client-server database program would work as follows:

The database is stored on the computer where the server application is running. The server application can access each part of the data very fast, but a user can not interact with the server program. A person sits on another, comparatively small computer and starts the client program. It presents a convenient screen, and the user enters, say, search criteria to search for a particular item in the database. Once the criteria has been entered in the client program, it connects to the server program and presents the request. The server program searches for the appropriate data in the database and delivers it (in a machine-suitable, often compressed) format. The client then 'decodes' the information and displays it nicely on the screen.

What is socket?

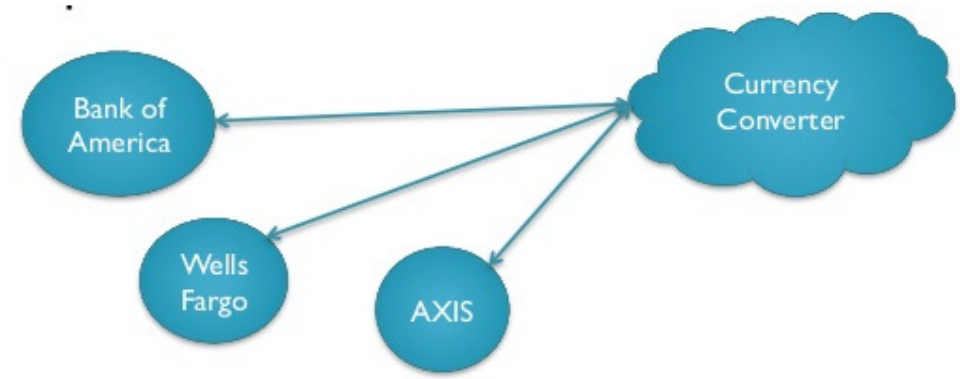
- It is a method for communication between a client program and a server program in a network.
- A server application normally listens to a specific port waiting for connection requests from a client. When a connection request arrives, the client and the server establish a dedicated connection over which they can communicate. During the connection process, the client is assigned a local port number, and binds a socket to it. The client talks to the server by writing to the socket and gets information from the server by reading from it. Similarly, the server gets a new local port number (it needs a new port number so that it can continue to listen for connection requests on the original port). The server also binds a socket to its local port and communicates with the client by reading from and writing to it.
- The client and the server must agree on a protocol--that is, they must agree on the language of the information transferred back and forth through the socket.
- When *your client programs are talking to a more complicated server such as an http server*, your client program will also be more complicated. However, the *basics* are much the same as they are in this program:
    1. Open a socket
    2. Open an input stream and output stream to the socket
    3. Read from and write to the stream according to the server's protocol
    4. Close streams
    5. Close sockets

3-SOA Architecture

Web Services: It is a service to exchange data from different unknown systems.

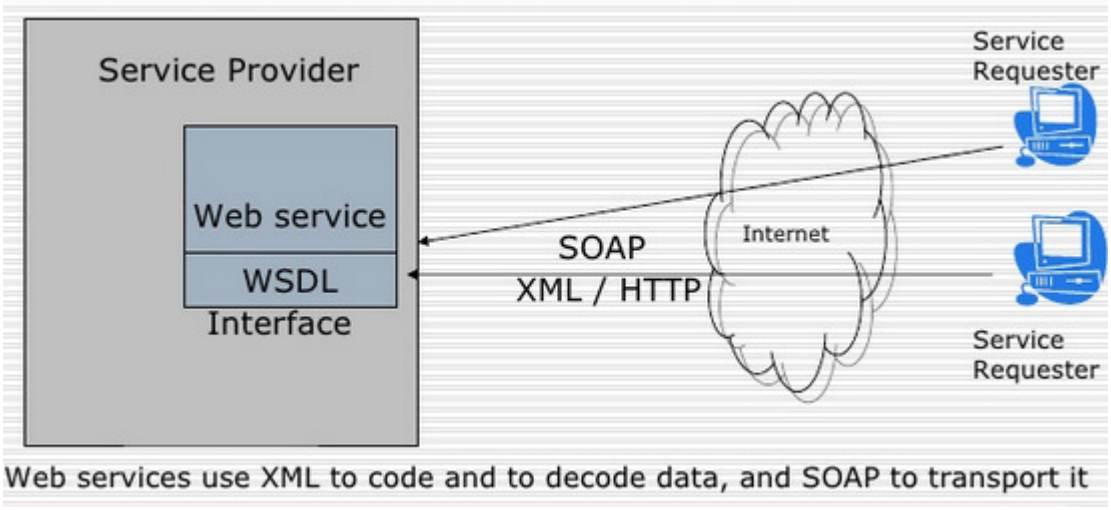
Uses of services:

- Reuse of existing applications
- Exchange or communicate with different platforms



Types of web services:

- **Simple web service**: has only simple data types like integer, string and other types are sent and receive.
- **Complex web service**: are needed to configure the ability to send and receive more than simple data types.



Most popular programming languages used for web services:

1-Java:

Chosen by big players. It doesn't mean that Java is the best one. Java was built around Enterprise model. At the beginning they planned to make business around it (the same .NET). It means that Java is:

- Reliable,
- Expensive. The costs come with overcomplicated solutions.

2-Python / Ruby

- Very fast developing and prototyping. Enjoy when coding, easy to maintain and refactor. Most of the web is driven by Python / Ruby / PHP frameworks (I don't want to talk about the last one). Those languages are built by professional geeks.
- Disadvantage: not as fast as Java. But for serving web content it is fast enough. Because there are implementations with JIT (e.g. PyPy, Numba for Python), this disadvantage is going to be even weaker. JIT gives a huge boost.

3-Node.js

Like the previous one. Brings performance rather then reliability.

4-Go

The newcomer. Takes the best from Java (fast) and Python / Ruby (fun, productive). Have simple yet powerful type hierarchy. Disadvantages: although the project is stable, there are not a lot of tools around it. Some which exists are not as rich / mature as the ones we have in previous technologies.

5-C#

Which one to choose? Choose the one for which you have proficient developers.

Web backend does not require so much complicated tools (those are done through separate workers, if not then it means your app is unsafe and aware of critical bugs, data crash etc...).

4-Cloud Computing Architecture

"Cloud computing" is more of an operating-system-level concept than a language concept.

Example: Let us say you want to host an application on *Amazon's EC2* cloud computing service. You can develop it in any language you like, on any operating system supported by EC2 (several flavors of Linux, Solaris, and Windows), then install and run it "in the cloud" on one or more virtual machines, much as you would do on a dedicated physical server.