Unit VI

System Implementation

System Implementation is the critical phase of system development process where you can create executable software.

Implementation is the stage in the software engineering process at which an executable software system is developed.

Implementation is the process of realizing the design as a program.

Systems implementation is the construction of the new system and the delivery of that system into production.

Systems implementation is the process of:

1. defining how the information system should be built (i.e., physical system design),
2. ensuring that the information system is operational and used,
3. ensuring that the information system meets quality standard (i.e., quality assurance).

System implementation is the practice of creating or modifying a system to create a new business process or replace an existing business process.

System implementation constructs the new information system and puts it into operation. During this phase, any new hardware and system software are installed and tested. Any purchased application software and databases are installed and configured.

Installation

Installation (or **setup**) of a computer program (including device drivers and plugins), is the act of making the program ready for execution. Because the process varies for each program and each computer, programs (including operating systems) often come with an *installer*, a specialized program responsible for doing whatever is needed for their installation.

Installation typically involves code being copied/generated from the installation files to new files on the local computer for easier access by the operating system. Because code is generally copied/generated in multiple locations, uninstallation usually involves more than just erasing the program folder. For example, registry files and other deep code in the system may need to be modified/deleted for a complete uninstallation.

Types of Installation

Attended installation

An installation process usually needs a user who attends it to make choices, such as accepting or declining anend-user license agreement (EULA), specifying preferences such as the installation location, supplying passwords or assisting in product activation. In graphical environments, installers that offer a wizard-based interface are common. Attended installers may ask users to help mitigate the errors. For instance, if the disk in which the computer program is being installed was full, the installer may ask the user to specify another target path or clear enough space in the disk.

Silent installation

Installation that does not display messages or windows during its progress. "Silent installation" is not the same as "unattended installation" . All silent installations are unattended but not all unattended installations are silent. The reason behind a silent installation may be convenience or subterfuge. Malware is almost always installed silently.

Unattended installation

Installation that is performed without user interaction during its progress or with no user present at all. One of the reasons to use this approach is to automate the installation of a large number of systems. An unattended installation either does not require the user to supply anything or has received all necessary input prior to the start of installation. Such input may be in the form of command line switches or an *answer file*, a file that contains all the necessary parameters. Windows XP and most Linux distributions are examples of operating systems that can be installed with an answer file. In unattended installation, it is assumed that there is no user to help mitigate errors. For instance, if the installation medium was faulty, the installer should fail the installation, as there is no user to fix the fault or replace the medium. Unattended installers may record errors in a computer log for later review.

Headless installation

Installation performed without using a computer monitor connected. In attended forms of headless installation, another machine connects to the target machine (for instance, via a local area network) and takes over the display output. Since a headless installation does not need a user at the location of the target computer, unattended headless installers may be used to install a program on multiple machines at the same time.

Scheduled or automated installation

An installation process that runs on a preset time or when a predefined condition transpires, as opposed to an installation process that starts explicitly on a user's command. For instance, a system administrator willing to install a later version of a computer program that is being used can schedule that installation to occur when that program is not running. An operating system may automatically install a device driver for a device that the user connects. Malware may also be installed automatically. For example, the infamous Conficker was installed when the user plugged an infected device to their computer.

Clean installation

A clean installation is one that is done in the absence of any interfering elements such as old versions of the computer program being installed or leftovers from a previous installation. In particular, the clean installation of an operating system is an installation in which the target disk partition is erased before installation. Since the interfering elements are absent, a clean installation may succeed where an unclean installation may fail or may take significantly longer.

Network installation

Network installation, shortened netinstall, is an installation of a program from a shared network resource that may be done by installing a minimal system before proceeding to download further packages over the network. This may simply be a copy of the original media but software publishers which offer site licenses for institutional customers may provide a version intended for installation over a network.

System Quality:

It defines the quality of system. Which types of qualities the system has? Whether it has good quality or bad quality? It depends on the reliability, durability, functionality and performance of the system. If these terms are good, the system is called of good quality if not is called bad quality.

What is Software "Quality"?

Quality software is reasonably bug-free, delivered on time and within budget, meets requirements and/or expectations, and is maintainable. However, quality is a subjective term. It will depend on who the 'customer' is and their overall influence in the scheme of things. A wide-angle view of the 'customers' of a software development project might include end-users, customer acceptance testers, customer contract officers, customer management, the development organisation's management/accountants/testers/salespeople, future software maintenance engineers, stockholders, magazine reviewers, etc. Each type of 'customer' will have their own view on 'quality' - the accounting department might define quality in terms of profits while an end-user might define quality as user-friendly and bug-free.

Software Quality Assurance (SQA)

Software quality assurance (SQA) is a process that ensures that developed software meets and complies with defined or standardized quality specifications. SQA is an ongoing process within the software development life cycle (SDLC) that routinely checks the developed software to ensure it meets desired quality measures.

SQA helps ensure the development of high-quality software. SQA practices are implemented in most types of software development, regardless of the underlying software development model being used. In a broader sense, SQA incorporates and implements software testing methodologies to test software. Rather than checking for quality after completion, SQA processes tests for quality in each phase of development until the software is complete. With SQA, the software development process moves into the next phase only once the current/previous phase complies with the required quality standards. SQA generally works on one or more industry standards that help in building software quality guidelines and implementation strategies.

What's difference between QA/testing

The quality assurance process is a process for providing adequate assurance that the software products and processes in the product life cycle confirm to their specific requirements and adhere to their established plans."

Software Testing is the process to find out errors in the system implemented.

Software Quality Assurance Activities

- Application of Technical Methods (Employing proper methods and tools for developing software)
- Conduct of Formal Technical Review (FTR)
- Testing of Software
- Enforcement of Standards (Customer imposed standards or management imposed standards)
- Control of Change (Assess the need for change, document the change)
- Measurement (Software Metrics to measure the quality, quantifiable)
- Records Keeping and Recording (Documentation, reviewed, change control etc. i.e. benefits of docs).

Formal Technical Review (FTR)
A formal technical review is a software quality assurance activity performed by software engineers (and others).
The objectives of the FTR are

- to uncover errors in function, logic, or implementation for any representation of the software;
- to verify that the software under review meets its requirements;
- to ensure that the software has been represented according to predefined standards;
- to achieve software that is developed in a uniform manner;
- to make projects more manageable.

In addition, the FTR serves as a training ground, enabling junior engineers to observe different approaches to software analysis, design, and implementation. The FTR also serves to promote backup and continuity because a number of people become familiar with parts of the software that they may not have otherwise seen.

The FTR is actually a class of reviews that includes walkthroughs, inspections, round-robin reviews and other small group technical assessments of software. Each FTR is conducted as a meeting and will be successful only if it is properly planned, controlled, and attended. In the sections that follow, guidelines similar to those for a walkthrough are presented as a representative formal technical review.

Walkthrough

In software engineering, a **walkthrough** or **walk-through** is a form of software peer review "in which a designer or programmer leads members of the development team and other interested parties through a software product, and the participants ask questions and make comments about possible errors, violation of development standards, and other problems".

Inspection

An **inspection** is, most generally, an organized examination or formal evaluation exercise. In engineering activities inspection involves the measurements, tests, and gauges applied to certain characteristics in regard to an object or activity. The results are usually compared to specified requirements and standards for determining whether the item or activity is in line with these targets. Inspections are usually non-destructive.

System Maintenance

After the systems implementation phase, the maintenance phase takes over.
Systems maintenance is the on-going maintenance of a system after it has been placed into operation. When developing information strategy plans, organizations cannot afford to neglect the fact that systems maintenance is the longest and costliest phase of the systems life cycle. The implications of the maintenance workload upon the information strategy plans for an organization is a subject that deserves special attention. The organization structure needs flexibility to support the maintenance of existing systems concurrently with the implementation of new technologies.
It is important to consider the evaluation and monitoring of a system for needed maintenance and consequently, to lower or contain maintenance costs. Systems maintenance can be categorized into four groups.
Corrective Maintenance: Regardless of how well designed, developed, and tested a system or application may be, errors will inevitably occur. This type of maintenance deals with fixing or correcting problems with the system. This usually refers to problems that were not identified during the implementation phase. An example of remedial maintenance is the lack of a user-required feature or the improper functionality of it.
Customized Maintenance: This type of maintenance refers to the creation of new features or adapting existing ones as required by changes in the organization or by the users, e.g., changes on the organization's tax code or internal regulations.
Enhancement Maintenance: It deals with enhancing or improving the performance of the system either by adding new features or by changing existing ones. An example of this type of maintenance is the conversion of text-based systems to GUI (Graphical User Interface).
Preventive Maintenance: This type of maintenance may be one of the most cost effective, since if performed timely and properly, it can avoid major problems with the system. An example of this maintenance is the correction for the year 2000.

Software maintenance process

- Filling up the request form
- Evaluating the request
- Changing the system according to the request
- Testing the changes

- Documenting the changes

System Testing

Systems testing is an expensive but critical process that can take as much as 50 percent of the budget for program development. The common view of testing held by users is that it is performed to prove that there are no errors in a program. However, this is virtually impossible, since analysts cannot prove that software is free and clear of errors.

Therefore, the most useful and practical approach is with the understanding that testing is the process of executing a program with explicit intention of finding errors that is, making the program fail. The tester, who may be an analyst, programmer, or specialist trained in software testing, is actually trying to make the program fail. A successful test, then, is one that finds an error.