Deploying the New System

Testing

- Testing activities are a key part of implementation and deployment activities, testing is the process of examining a component, subsystem, or system to determine its operational characteristics and whether it contains any defects.
- To conduct a test, developers must have well-defined specifications for both functional and non-functional requirements. From the requirements specifications, test developers develop precise definitions of expected operational characteristics
- If the results indicate a shortcoming or defect, then the project team cycles back through earlier implementation or deployment activities until the shortcoming is remedied or the defect is eliminated.
- An important part of developing tests is specifying **test cases** and **test data**
- A **test case** is a formal description of the following:
    - A starting state or condition
    - One or more events to which the software must respond
    - The expected response or ending state
- The starting states and the events are represented by a set of **test data.** For example, the starting state of a system may represent a particular set of data, such as the existence of a particular customer and order for that customer

Test type	Core process	Need and purpose
Unit testing	Implementation	Software components must perform to the defined requirements and specifications when tested in isolation—for example, a component that incorrectly calculates sales tax amounts in different locations is unacceptable.
Integration testing	Implementation	Software components that perform correctly in isolation must also perform correctly when executed in combination with other components. They must communicate correctly with other components in the system. For example a sales tax component that calculates incorrectly when receiving money amounts in foreign currencies is unacceptable .
System and stress testing	Deployment	A system or subsystem must meet both functional and non-functional requirements. For example an item lookup function in a Sales subsystems retrieves data within 2 seconds when running in isolation, but requires 30 seconds when running within the complete system with a live database.
User acceptance testing	Deployment	Software must not only operate correctly, but must also satisfy the business need and meet all user "ease of use" and "completeness" requirements—for example, a commission system that fails to handle special promotions or a data-entry function with a poorly designed sequence of forms is unacceptable.

Figure 1. Types of software test and the purpose of each

- **Unit testing** is the lowest level of and the earliest testing for a new software system, a unit test is a test of an individual method, class, or component before it is integrated with other software
- The primary purpose of doing unit testing is to test a small piece of the code in isolation to make sure that it functions correctly before it is integrated into a larger program.
- **Integration testing** is the next logical extension of unit testing. After small units are tested, they are combined into a larger component and tested together
- The purpose of an integration test is to identify errors that were not or couldn't be detected by unit testing, it evaluates the functional behavior of a group of classes or components when they are combined.
- Figure 2 illustrates this concept. A new class or component is integrated into the existing, growing, and tested portion of the system.

Figure 2. Integration testing— adding new components to tested components

- **System testing** means testing the system as a whole. All the modules/components are integrated to verify if the system works as expected or not
- System testing is done after integration testing. This plays an important role in delivering a high-quality product. Figure 3 illustrates the types of tests that can be included in system testing.

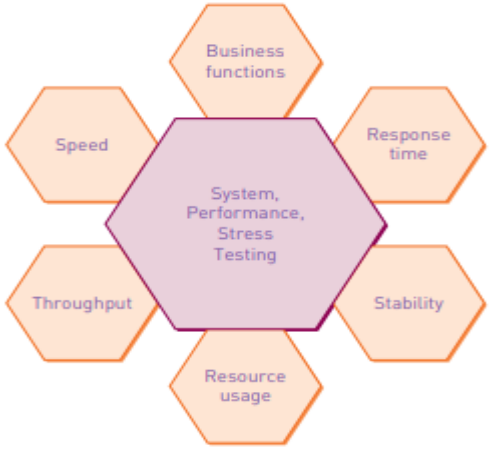

Figure 3. Types of test included in system testing

- A **performance test**, also called a **stress test**, determines whether a system or subsystem can meet such time-based performance criteria as response time. Stress tests examine how the system behaves under intense loads and how it recovers when going back to normal usage.
- **User acceptance testing** (UAT) is the last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications. UAT is one of the final, and critical software project procedures that must occur before newly developed software is rolled out to the market.

Deployment Activities

- Once a new system has been developed and tested, it must be placed into operation. Deployment activities involve many conflicting constraints, including cost, the need to maintain positive customer relations, the need to support employees, and overall risk to the organization.
- **Training Users** is an essential part of any system deployment project. Figure 3 shows representative activities for each role. The two (2) classes of users are the following:
    - **End users** are people who use the system from day to day to achieve the system's business purpose.
    - **System operators** are people who perform administrative functions and routine maintenance to keep the system operating.

End-user activities	System operator activities
Creating records or transactions	Starting or stopping the system
Modifying database contents	Querying system status
Generating reports	Backing up data to archive
Querying database	Recovering data from archive
Importing or exporting data	Installing or upgrading software

Figure 4. Typical activities of end-users and system operators

- **System documentation** serves one primary purpose: providing information to developers and other technical personnel who will build, maintain, and upgrade the system.
- Documentation and other training materials are usually developed before formal user training begins. Each documentation type is targeted to a different purpose and audience. Documentation can be loosely classified into two types:
    - **System documentation** descriptions of system requirements, architecture, and construction details, as used by maintenance personnel and future developers.
    - **User documentation** descriptions of how to interact with and use the system, as used by end-users and system operators.

Deployment

- Different approaches to deployment represent different trade-offs among cost, complexity, and risk. The most used deployment approaches are:
    - Direct deployment
    - Parallel deployment
    - Phased deployment
- **Direct deployment** is a deployment method that installs a new system, quickly makes it operational, and immediately turns off any overlapping systems.

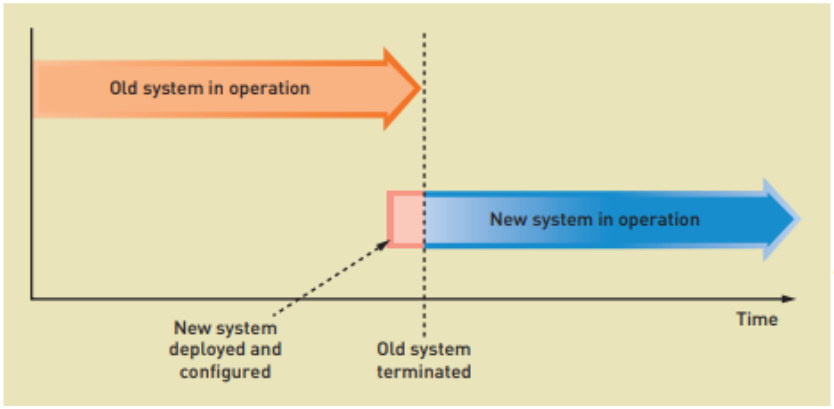

Figure 5. Direct deployment

- **Parallel deployment** is a deployment method that operates the old and the new system for an extended period. Ideally, the old system continues to operate until the new system has been thoroughly tested and determined to be error-free and ready to operate independently.

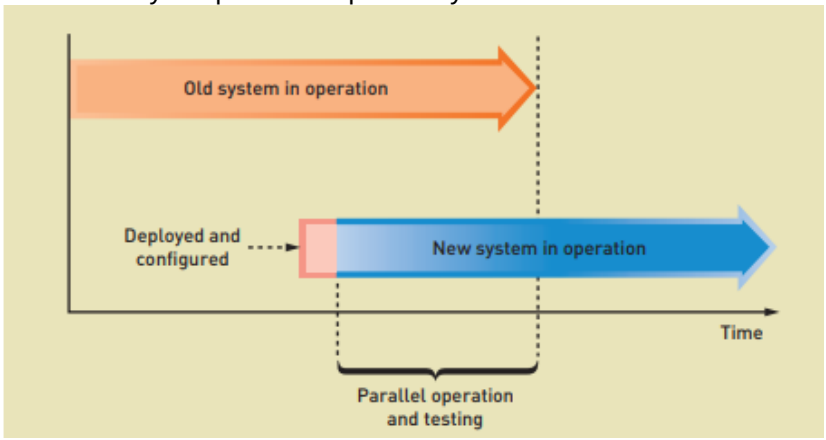

Figure 6. Parallel deployment

- **Phased deployment** is a deployment method that installs a new system and makes it operational in a series of steps or phases. Each phase adds components or functions to the operational system. During each phase, the system is tested to ensure that it is ready for the next phase
- Phased deployment can be combined with parallel deployment, particularly when the new system will take over the operation of multiple existing systems. Figure 7 shows a phased deployment with the direct and parallel deployment of individual phases.

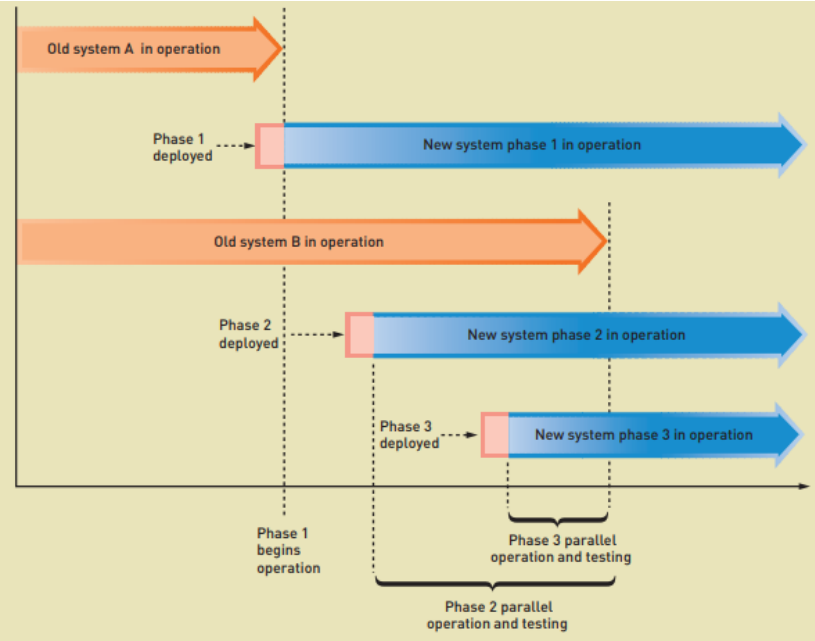

Figure 7. Phased deployment

REFERENCES:

Tilley, S. (2020). *System analysis and design.* USA: Cengage Learning.
Coronel,C. and Morris, S. (2017). *Database systems: Design, implementation, and management* (12th ed.). USA: Cengage Learning.
Satzinger, J., Jackson, R., and Burd, S. (2016). *Systems analysis and design in a changing world-course technology.* USA.