# SYSTEMS DEVELOPMENT LIFE CYCLE

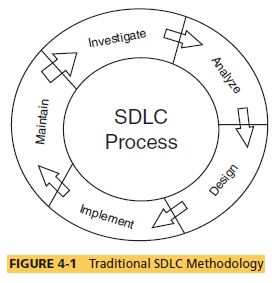
The process of developing new information systems is often called the system development life cycle. It basically includes a systematic process of planning, designing, and creating an information system for organizations. Even though the process of developing a system for individual or personal use can be simple, the task can become very complex when the system has to support thousands of business processes for several hundred users both inside and outside an organization.

For complex systems development projects (e.g., ERP), it is often better to have a structured methodology to avoid mishaps and coordinate the design and development tasks properly among the members of a large systems development team. SDLC uses a systems approach for problem solving that basically states that complex problems need to be broken up into smaller manageable problems using a systems’ hierarchy, and then developing a solution for each problem within the hierarchy. It provides a structured top-down problem identification and bottom-up solution process for managing complex problems. The structured or phased approach is designed to catch problems at an early stage before they become a major risk to the system implementation process. The SDLC process requires both technical and nontechnical problem-solving skills; therefore, the development team must understand technology, as well as the organization’s business processes, culture, and people (or potential end users of this system). For example, a component of an HR system must capture organizational policy on health care benefits and retirement and the process of deducting the premiums from the payroll checks.

Every organization will have some variations that need to be accurately captured and processed by the new system. Capturing these processes and then implementing them in a new system can be difficult for a person with an IT background only; therefore, the development team must be composed of people with a wide variety of IT and business skills for the project to be successful.

**Traditional SDLC**

In the early days of systems development, very few of these projects were successful in the first attempt. There were many reasons for the early failures, chief among them being lack of experience.This led to the systems approach, which we described earlier, and a structured SDLC methodology. The SDLC consists of tasks that are divided into phases or stages.



**Rapid SDLC Approaches**

The SDLC process has several problems, even though it is rigorous in making sure that the new system is complete and successful in the organization. First, developing a new system is time consuming and tedious. In many cases the new system is outdated by the time it is developed. Second, the cost associated with the SDLC process is very high. The cost of recruiting the development team and involving other members of the organization in the development process can be very expensive. Finally, all information systems do not require such a rigorous SDLC process.

For example, the SDLC would be overkill for a small-scale decision-making application; therefore, over the years organizations have used rapid approaches to SDLC that are quicker and less expensive shortcuts to this process. These are called Rapid SDLC approaches.

One rapid development approach is *prototyping* (Figure 4-3). This approach does not go through the analysis and design phases; instead, it implements a skeleton or a prototype of the actual system with a focus on input (i.e., user interface) and output (i.e., screen displays and reports generated with dummy data). The idea is to demonstrate the system functionality as soon as possible to the users and to get their feedback on the prototype. Their feedback is incorporated into the new system and demonstrated back to the users. This approach has proven to be very effective with user-interactive systems because the prototype is eventually converted into a full-scale system. In ERP implementations, many companies install a sandbox system to expose users to the system functionality. ERP sandboxes replicate at least the minimal functionality needed to get user feedback before implementing a full-scale system. The goal of sandboxing is similar to that of prototyping.

Another rapid development approach is *end-user development* (EUD), which lets the end users create their own applications. This process became popular in the 1980s with the advent of personal computers (PCs). In this process the users are trained by the IT staff or professional trainers to develop customized applications (e.g., a small decision-making application with an Excel spreadsheet or a departmental employee tracking system with an Access database). Several other customized approaches have similarly been developed over the years to circumvent the exhaustive SDLC. EUD is applicable in ERP for designing custom reports from the ERP system.

ERP IMPLEMENTATION LIFE CYCLE

ERP applications are prepackaged software developed by commercial software vendors and custom

installed for organizations to automate and integrate the various business processes. Although an

ERP is packaged software, it is very different from PC-based software packages (e.g., Microsoft

Office or other software) that you may have purchased for personal use as shown in Table 4-1. These

are complex software packages costing millions of dollars to develop and maintain that automate

hundreds of business processes in an organization. Furthermore, these applications are mission critical

(i.e., if they fail or break down, the organization will stop functioning).

are fixed and new features are sought for next implementation life cycle.

ERP Implementation Plan

An ERP implementation plan is used to create a roadmap or blueprint to meet cost, scope, and

time constraints of an implementation. There are many different ERP implementation methodologies

promoted by different vendors and consultants. The appropriateness of the plan depends,

in part, on the project, the company, and the reasons for the implementation.

Following are three major implementation plan choices:

**1.** *Comprehensive.* A comprehensive ERP integration plan is the most expensive, lengthy,

and costly approach. It involves implementation of the full functionality of the ERP software

in addition to industry-specific modules. Implementing the full functionality requires

a high level of business process reengineering (BPR) with major changes in the business

processes and customization of legacy systems.

**2.** *Middle of the Road.* A middle-of-the-road ERP implementation plan involves some changes

in the core ERP modules and a significant amount of BPR. The middle-of-the-road

approach is not as expensive as the comprehensive approach or as straightforward as the

vanilla approach.

**3.** *Vanilla.* A vanilla ERP implementation plan utilizes core ERP functionality and exploits

the best practice business processes built into the software. A company following a vanilla

implementation will have to simply align their business processes to the ERP system,

rather than modify the software. By eliminating or minimizing the required BPR, the

project’s costs and time required for the implementation are minimized.

ERP Implementation Methodology

ERP methodology

builds on the theory that an enterprise can maximize its returns by maximizing the utilization

of its fixed supply of resources. Information technology, with its increasing computer

power and the ability to correlate pieces of information, has proven to be the best tool for

business problem solving. Like SDLC, an ERP development life cycle provides a systematic

approach to implementing ERP software in the changing but limited-resource organizational

environment.

There are many different vendor-driven methodologies or approaches that use traditional

ERP development life cycle or rapid ERP life cycles (e.g., Total Solution, FastTrack,

Rapid-Re, Accelerated SAP (ASAP), and business integration methodology (BIM)).

Implementation methodologies are similar in their overall approach with the differences

coming primarily in the staging of the process steps and formality of structure. The traditional

ERP life cycle accomplishes one stage at a time and requires formal milestone approvals

prior to moving to the next stage. In a rapid ERP life cycle, once a company commits to the

implementation, employees are empowered to make the decisions to keep the project moving

forward. They also allow flexibility and quicker feedback loops to accommodate rapid

corrections

Traditional ERP Life Cycle

Like the traditional SDLC, which we discussed earlier, the traditional ERP life cycle approach

has a deliverable at the end of each stage (e.g., a report with supporting documents) that is

reviewed by management and upon which a decision is made either to continue with the project

or not. End-user or people involvement is critical in both SDLC and ERPLC; however, there are

other variations to the traditional SDLC process. The emphasis in ERP implementation is on

customizing the software as well as on changing the organization’s business processes, rather

than determining the user requirements for developing new applications (as in the traditional

SDLC). This may seem like a small deviation, but it requires a major change in the thinking

process as well as team composition and skill level of people involved in the development

process. Furthermore, the ERP life cycle, as shown in Figure 4-5, iterates at a much faster pace

than in the traditional SDLC.

The traditional ERP life cycle includes the following major stages:

**Stage 1.** *Scope and commitment stage.* This is similar to the investigation stage in SDLC

discussed earlier. In addition to conducting the feasibility study, however, one of the

first steps is to develop a scope of ERP implementation within the resource and time

requirement. A number of task parameters or characteristics of the ERP implementation need to be defined at the planning stage.

**Stage 2.** *Analysis and design stage.* In addition to analysis of user requirements, the ERP team

has first to make a decision on the software and decide on consultants and SMEs.

Another key activity is to map the differences between the current business process and

the embedded process in the ERP software or gap analysis and to develop a long-term

plan on whether to change the business processes of the organization or to customize

the ERP software to support existing processes. Using the gap analysis, the team must

develop a design that among other things includes a change management plan, a list of

embedded processes, user interface screens, and reports in the ERP software that will

need customization, design of these changes, and a process of involving subject matter

experts in the design. Other activities include creating plans for data conversion, system

conversion, and training. For a system to be successful, the team must develop a

detailed change management strategy and plan for the release of the new system. By

the end of this stage, the team usually has a sandbox or prototype of the ERP software

installed that is accessible to the entire implementation team, consultants, and SMEs.

**Stage 3.** *Acquisition and development stage.* This stage is similar to the acquisition and testing

stage of traditional SDLC. The organization has to purchase the license for the production

version of the software and build the production version of the system, which is eventually to be made available to the end users. The entire production platform

must be configured and built with the necessary hardware, network, security, software,

database, and real production data.

**Stage 4.** *Implementation stage.* The focus for this stage is on installing and releasing the system

to the end users (i.e., “Go-Live”) and on monitoring the system release to the end

users. This production platform is a mirror of the development version of the system.

Errors found in the production version have to go through the help desk or support

staff. Any changes made to the development version are then retested and migrated to

the production system as regularly scheduled updates. System conversion is a major

activity for the new system and needs to be managed carefully. There are four basic

conversion approaches

**Stage 5.** *Operation stage.* This is often managed by the operation team with assistance from

the implementation team. Knowledge transfer is the major activity as support for the

new system is migrated to the help desk and support staff. Some implementation

team members are very often hired as support staff. The other major activities are

ongoing training of new users to the system as ERP modules are released, as well as

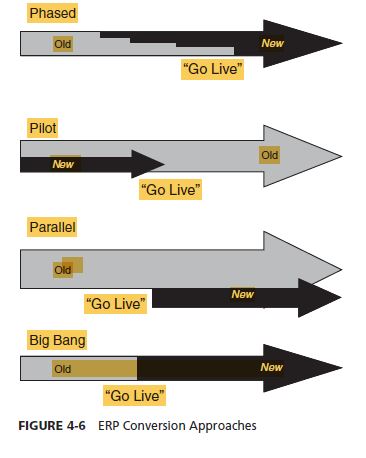
to take a fresh look at the change management strategy. The team has to monitor user

feedback from training and actual system usage carefully and make the necessary

adjustments to the change management approach. Another key activity is management

of new releases of the software, installation of patches and upgrades to the

system, and managing the software contract with the ERP vendor.

, ****