

System Building

Key Terms

The following alphabetical list identifies the key terms discussed in this chapter.

Acceptance testing — provides the final certification that the system is ready to be used in a production setting.

Automation — using the computer to speed up the performance of existing tasks.

Benchmarking — setting strict standards for products, services, or activities and measuring organizational performance against those standards.

Computer-aided software engineering (CASE) — automation of step-by-step methodologies for software and systems development to reduce the amounts of repetitive work the developer must do.

Conversion — the process of changing from the old system to the new system.

Customization — the modification of a software package to meet an organization's unique requirements without destroying the packaged software's integrity.

Data flow diagram — primary tool for structured analysis that graphically illustrates a system's component process and the flow of data between them.

Direct cutover — a risky conversion approach in which the new system completely replaces the old one on an appointed day.

Documentation — descriptions of how an information system works from either a technical or end-user standpoint.

End-user development — the development of information systems by end users with little or no formal assistance from technical specialists.

End-user interface — the part of an information system through which the end user interacts with the system, such as online screens and commands.

Feasibility study — as part of the systems analysis process, the way to determine whether the solution is achievable, given the organization's resources and constraints.

Fourth-generation languages — a programming language that can be employed directly by end users or less-skilled programmers to develop computer applications more rapidly than conventional programming languages.

Information requirements — a detailed statement of the information needs that a new system must satisfy; identifies who needs what information, and when, where, and how the information is needed.

Iterative — a process of repeating over and over again the steps to build a system.

Maintenance — changes in hardware, software, documentation, or procedures to a production system to correct errors, meet new requirements, or improve processing efficiency.

Object — software building block that combines data and the procedures acting on the data.

Object-oriented development — approach to systems development that uses the object as the basic unit of systems analysis and design. The system is modeled as a collection of objects and the relationship between them.

Parallel strategy — a safe and conservative conversion approach where both the old system and its potential replacement are run together for a time until everyone is assured that the new one functions correctly.

Phased approach — introduction of the new system in stages either by functions or by organizational units.

Pilot study — a strategy to introduce the new system to a limited area of the organization until it is proved to be fully functional; only then can the conversion to the new system take place across the entire organization.

Postimplementation audit — formal review process conducted after a system has been placed in production to determine how well the system has met its original objectives.

Process specifications — describe the logic of the processes occurring within the lowest levels of a data flow diagram.

Production — the stage after the new system is installed and the conversion is complete; during this time the system is reviewed by users and technical specialists to determine how well it has met its original goals.

Programming — the process of translating the system specifications prepared during the design stage into program code.

Prototype — the preliminary working version of an information system for demonstration and evaluation purposes.

Prototyping — the process of building an experimental system quickly and inexpensively for demonstration and evaluation so that users can better determine information requirements.

Query languages — software tool that provides immediate online answers to requests for information that are not predefined.

Request for Proposal (RFP) — a detailed list of questions submitted to vendors of software or other services to determine how well the vendor's product can meet the organization's specific requirements.

Structure chart — system document showing each level of design, the relationship among the levels, and the overall place in the design structure; can document one program, one system, or part of one program.

Structured — refers to the fact that techniques are carefully drawn up, step by step, with each step building on a previous one.

Systems analysis — specialists who translate business problems and requirements into information requirements and systems and who act as liaisons between the information systems department and the rest of the organization.

Systems design — details how a system will meet the information requirements as determined by the systems analysis.

Systems development — the activities that go into producing an information systems solution for an organizational problem or opportunity.

Systems life cycle — a traditional methodology for developing an information system that partitions the systems development process into formal stages that must be completed sequentially with a very formal division of labor between end users and information systems specialists.

System testing — tests the functioning of the information system as a whole to determine whether discrete modules will function together as planned.

Test plan — a plan prepared by the development team in conjunction with the users; it includes all of the preparations for the series of tests to be performed on the system.

Testing — the exhaustive and thorough process that determines whether the system produces the desired results under known conditions.

Unit testing — the process of testing each program separately in the system. Sometimes called program testing.

Work flow management — the process of streamlining business procedures so that documents can be moved easily and efficiently from one location to another.

Summary

1. *Identify and describe the core activities in the systems development process.*

The core activities in systems development are systems analysis, systems design, programming, testing, conversion, production, and maintenance. Systems analysis is the study and analysis of problems of existing systems and the identification of requirements for their solutions. Systems design provides the specifications for an information system solution, showing how its technical and organizational components fit together.

2. *Evaluate alternative methods for building information systems.*

There are a number of alternative methods for building information systems, each suited to different types of problems. The oldest method for building systems is the systems life cycle, which requires that information systems be developed in formal stages. The stages must proceed sequentially and have defined outputs; each requires formal approval before the next stage can commence. The system life cycle is useful for large projects that need formal specifications and tight management control over each stage of systems building. However, this approach is very rigid and costly and is not well suited for unstructured, decision-oriented applications for which requirements cannot be immediately visualized.

Prototyping consists of building an experimental system rapidly and inexpensively for end users to interact with and evaluate. The prototype is refined and enhanced until users are satisfied that it includes all of their requirements and can be used as a template to create the final system. Prototyping encourages end-user involvement in systems development and iteration of design until specifications are captured accurately. The rapid creation of prototypes can result in systems that have not been completely tested or documented or that are technically inadequate for a production environment.

Developing an information system using an application software package eliminates the need for writing software programs when developing an information system. Using a software package reduces the amount of design, testing, installation, and maintenance work required to build a system. Application software packages are helpful if a firm does not have the internal information systems staff or financial resources to custom develop a system. To meet an organization's unique requirements, packages may require extensive modifications that can substantially raise development costs.

End-user development is the development of information systems by end users, either alone or with minimal assistance from information systems specialists. End-user-developed systems can be created rapidly and informally using fourth-generation

software tools. The primary benefits of end-user development are improved requirements determination; reduced application backlog; and increased end-user participation in, and control of, the systems development process. However, end-user development, in conjunction with distributed computing, has introduced new organizational risks by propagating information systems and data resources that do not necessarily meet quality standards and that are not easily controlled by traditional means.

Outsourcing consists of using an external vendor to build (or operate) a firm's information systems. The work is done by the vendor rather than by the organization's internal information systems staff. Outsourcing can save application development costs or enable firms to develop applications without an internal information systems staff. However, firms risk losing control over their information systems and becoming too dependent on external vendors.

Selection of a systems-building approach can have a big impact on the time, cost, and end product of systems development. Managers should be aware of the strengths and weaknesses of each systems-building approach and the types of problems for which each is best suited. The impact of application software packages and of outsourcing should be carefully evaluated before they are selected because these approaches give organizations less control over the systems-building process.

3. Compare alternative methodologies for modeling systems.

The two principal methodologies for modeling and designing information systems are structured methodologies and object-oriented development. Structured methodologies focus on modeling processes and data separately. The data flow diagram is the principal tool for structured analysis and the structure chart is the principal tool for representing structured software design. Object-oriented development models a system as a collection of objects that combine processes and data. Object-oriented modeling is based on the concepts of class and inheritance.

4. Identify and describe new approaches for system-building in the digital firm era.

Businesses today are often required to build e-commerce and e-business applications very rapidly to remain competitive. New systems are likely to have more interorganizational requirements and processes than in the past. Companies are turning to rapid application design, joint application design (JAD), and reusable software components to improve the systems development process. Rapid application development (RAD) uses object-oriented software, visual programming, prototyping, and fourth-generation tools for very rapid creation of systems. Component-based development expedites application development by grouping objects into suites of software components that can be combined to create large-scale business applications.

Web services enable firms to obtain software application components delivered over the Internet for building new systems or integrating existing systems. Web services provide a common set of standards that enable organizations to link their systems regardless of their technology platform through standard plug and play architecture.

Review Questions

4. What is the difference between systems analysis and systems design? What activities does each comprise?

Systems analysis is the analysis of the problem that the organization is trying to solve with an information system. It consists of defining the problem, identifying its causes, specifying solutions, and identifying the information requirements that must be met by a system solution. Systems design shows how the system will fulfill the information requirements specified in system analysis.

5. What are information requirements? Why are they difficult to determine correctly?

Information requirements involve identifying who needs what information, where, when, and how. They define the objectives of the new or modified system and contain a detailed description of the functions the new system must perform. Gathering information requirements is perhaps the most difficult task of the systems analyst, and faulty requirements analysis is a leading cause of systems failure and high systems development costs.

Information requirements are difficult to determine because business functions can be very complex and poorly defined. A manual system or a routine set of inputs and outputs may not exist. Procedures may vary from individual to individual, and users may disagree on how things are or should be done. Defining information requirements is a laborious process, requiring a great deal of research and often several reworks by the analyst.

6. Why is the testing stage of systems development so important? Name and describe the three stages of testing for an information system.

Testing is critical to the success of a system because it is the only way to ascertain whether the system will produce the right results. Three stages of information system testing are unit testing, system testing, and acceptance testing. Unit testing refers to separately testing or checking the individual programs. With system testing, the entire system as a whole is tested to determine whether program modules are interacting as planned. With acceptance testing, the system undergoes final certification by end

users to ensure that it is ready for installation.

7. What role do programming, conversion, production, and maintenance play in systems development?

Programming translates the design specification into software, thus providing the actual instructions for the computer. Programming constitutes a smaller portion of the systems development cycle than design and perhaps testing activities. Conversion is the process of changing from the old system to the new system. Production is the operation of the system once it has been installed and conversion is complete. The system will be reviewed during production by both users and technical specialists to determine how well it has met its original objectives and to decide whether any revisions or modifications are needed. Maintenance is modifications to hardware, software, documentation, or procedures to a production system to correct errors, meet new requirements, and improve processing efficiency.

8. Compare object-oriented and traditional structured approaches for modeling and designing systems.

The traditional structured methodology focuses on what the new system is intended to do and then develops the procedures and data to do it. Object-oriented development de-emphasizes system procedures and instead creates a model of a system composed of individual objects that combine data and procedures. The objects are independent of any specific system.

These objects can then be placed into any system being built that needs to make use of the data and functions. In addition, in traditional structured methodologies all work is done serially, with work on each phase begun only when the previous phase is completed. Object-oriented development theoretically allows simultaneous work on design and programming. These systems usually are easier to build and more flexible. Moreover, any objects created this way are reusable for other programs.

9. What is the traditional systems life cycle? Describe each of its steps and its advantages and disadvantages for system building.

The traditional systems life cycle is a formal methodology for managing the development of systems and is still the principal methodology for medium and large projects. The overall development process is partitioned into distinct stages, each of which consists of activities that must be performed to fashion and implement an information system. The stages are usually gone through sequentially with formal “sign-off” agreements among end users and data processing specialists to validate that each stage has been completed. Users, managers, and data processing staff have specified responsibilities in each stage. The approach is slow, expensive, inflexible, and is not appropriate for many small desktop systems.

The systems life cycle consists of systems analysis, systems design, programming, testing, conversion, and production and maintenance. Systems analysis is the phase

where the problem that the organization is trying to solve is analyzed. Technical specialists identify the problem, gather information requirements, develop alternative solutions, and establish a project management plan. Business users provide information requirements, establish financial or operational constraints, and select the solution. During systems design, technical specialist's model and document design specifications and select the hardware and software technologies for the solution. Business users approve the specifications.

During the programming phase, technical specialists translate the design specifications into software for the computer. During the testing phase, technical specialists develop test plans and conduct unit, system, and acceptance tests. Business users provide test data and scenarios and validate test results.

During the conversion phase, technical specialists prepare a conversion plan and supervise conversion. Business users evaluate the new system and decide when the new system can be put into production. During the production and maintenance phase, technical specialists evaluate the technical performance and perform maintenance. Business users utilize the system and evaluate its functional performance.

The advantages of using this method for building information systems include it is highly structured; it has a rigorous and formal approach to requirements and specifications and tight controls over the system building process; it is appropriate for building large transaction processing and management information systems and for building complex technical systems. The disadvantages include: it is very costly and time consuming; it is inflexible and discourages change even though requirements will change during the project due to the long time this method requires; it is ill-suited to decision-oriented applications that can be rather unstructured and for which requirements are difficult to define.

10. What do we mean by information system prototyping? What are its benefits and limitations? List and describe the steps in the prototyping process.

Information system prototyping is an explicitly interactive system design methodology that builds an experimental model of a system as a means of determining information requirements. Prototyping builds an experimental system quickly and inexpensively for demonstration and evaluation so that users can better determine information requirements. A preliminary model of a system or important parts of the system is built rapidly for users to experiment with. The prototype is modified and refined until it conforms precisely to what users want. Information requirements and design are determined dynamically as users interact with and evaluate the prototype.

Prototyping is most valuable when requirements are uncertain and cannot be entirely prespecified or when the appropriate design solution is unclear. Prototyping is especially helpful for designing end-user interfaces (screens and reports) and for

determining elusive requirements of decision-support type applications. Prototyping can help reduce implementation costs by capturing requirements more accurately at an earlier point in the implementation process. It is not so useful for a very structured, well-understood, or routine problem.

It is best suited for smaller applications oriented toward simple data manipulation. Large systems with complex processing may only be able to have limited features prototyped. The prototype may be built so rapidly that design is not well thought out or must be reworked for a production environment. The problem arises when the prototype is adopted as the production version of the system without careful analysis and validation. Prototypes are built so rapidly that documentation and testing are glossed over. The system is so easily changed that documentation may not be kept up-to-date.

The steps in prototyping include identifying the users basic requirements; developing a working prototype of the system outlined in the basic requirements, using the prototype, and revising and enhancing the prototype based on the user's reaction. The third and fourth steps are repeated until users are satisfied with the prototype.

11. What is an application software package? What are the advantages and disadvantages of developing information systems based on software packages?

An application software package is a set of prewritten, precoded application software programs that are commercially available for sale or lease. Packages range from very simple programs to very large and complex systems, encompassing hundreds of programs. Packages are normally used when functions are common to many companies, data processing resources, for in-house development, are in short supply, and when desktop microcomputer applications are being developed for end users.

Software packages provide several advantages: (1) the vendor has already established most of the design that may easily consume up to 50 percent of development time; (2) programs are pretested, cutting down testing time and technical problems; (3) the vendor often installs or assists in the installation of the package; (4) periodic enhancement or updates are supplied by the vendor; (5) vendors also maintain a permanent support staff well versed in the package, reducing the need for individual organizations to maintain such expertise in-house, and (6) the vendor supplies documentation.

The usage of software packages has several disadvantages: (1) there are high conversion costs for systems that are sophisticated and already automated; (2) packages may require extensive customization or reprogramming if they cannot easily meet unique requirements, and (3) a system may not be able to perform many functions well in one package alone.

12. What is meant by end-user development? What are its advantages and disadvantages? Name some policies and procedures

for managing end-user development.

End-user development refers to the development of information systems by end users with minimal or no assistance from professional systems analysts or programmers. This is accomplished through sophisticated user-friendly software tools and gives end users direct control over their own computing.

Advantages include improved requirements determination, realizing large productivity gains when developing certain types of applications, enabling end users to take a more active role in the systems development process, many can be used for prototyping, and some have new functions such as graphics, modeling, and ad-hoc information retrieval.

Disadvantages include not being suited for large transaction-oriented applications or applications with complex updating requirements, standards for testing and quality assurance may not be applied, and proliferation of uncontrolled data and private information systems.

End-user development is suited to solving some of the backlog problem because the end users can develop their needed applications themselves. It is suited to developing low-transaction systems. End-user development is valuable for creating systems that access data for such purposes as analysis (including the use of graphics in that analysis) and reporting. It can also be used for developing simple data-entry applications.

Policies and procedures to manage end-user development include the following:

- The organization must establish sufficient support facilities for end-user computing: information centers or distributed end-user computing centers.
- Training and support should be targeted to the specific needs of those being trained.
- End-user application development should not be allowed to be undertaken randomly but should be incorporated into the organizations strategic plan.

Management should develop controls over end-user computing in the following areas:

- Cost justification of end-user information system project.
- Hardware and software standards for user-developed applications.
- Company-wide standards for microcomputers, word processing software, database management systems, graphics software, and query and reporting tools.
- Quality assurance reviews that specify whether the end-user systems must be reviewed by information systems and internal audit specialists.
- Control for end-user developed applications covering testing, documentation, accuracy, and completeness of input and update, backup, recovery, and supervision.

- Critical applications that supply data to other important systems should be flagged and subjected to more rigorous standards.

13. Under what circumstances should outsourcing be used for building information systems?

Outsourcing is the process of turning over an organizations computer center operations, telecommunications networks, or applications development to external vendors who provide these services. Outsourcing is an option often considered when the cost of information systems technology has risen too high. Outsourcing is seen as a way to control costs or to develop applications when the firm lacks its own technology resources to do this on its own. It is seldom used for a system that is strategically important.

14. What is rapid application development (RAD)? How can it help system builders?

Businesses today are often required to build e-commerce and e-business applications very rapidly to remain competitive. New systems are likely to have more interorganizational requirements and processes than in the past. Companies are turning to rapid application design to improve the systems development processes. Rapid application development (RAD) uses object-oriented software, visual programming, prototyping, and fourth-generation tools for rapid creation of systems.

15. How can component-based development and Web services help firms build and enhance their information systems?

Component-based development expedites application development by grouping objects into suites of software components that can be combined to create large-scale business applications.

Web services enable firms to obtain software application components delivered over the Internet for building new systems or integrating existing systems. Web services provide a common set of standards that enable organizations to link their systems regardless of their technology platform through standard plug-and-play architecture.