Question 1: **2.5 marks (1 to 2 pages each)**

1. **Explain** the Business process redesign and **explain** the steps companies go through in practicing business process management.

One of the most important implementations of competitive strategies is **business process reengineering** (BPR), often simply called *reengineering*. Reengineering is a funda- mental rethinking and radical redesign of business processes to achieve dramatic improvements in cost, quality, speed, and service. BPR combines a strategy of promoting business innovation with a strategy of making major improvements to business processes so that a company can become a much stronger and more successful com- petitor in the marketplace.

Many businesses today are trying to use information technology to improve their business processes. Some of these systems entail incremental process change, but others require more far-reaching redesign of business processes.

To deal with these changes, organizations are turning to business process management. **Business process management** provides a variety of tools and methodologies to analyze existing processes, design new processes, and optimize those processes. BPM is never concluded because process improvement requires continual change. Companies practicing business process management go through the following steps:

**1. Identify processes for change:** One of the most important strategic decisions that a firm can make is not deciding how to use computers to improve business processes, but understanding what business processes need improvement. When systems are used to strengthen the wrong business model or business processes, the business can become more efficient at doing what it should not do. As a result, the firm becomes vulnerable to competitors who may have discovered the right business model. Considerable time and cost may also be spent improving business processes that have little impact on overall firm performance and revenue. Managers need to determine what business processes are the most important and how improving these processes will help business performance.

**2. Analyze existing processes:** Existing business processes should be modeled and documented, noting inputs, outputs, resources, and the sequence of activities. The process design team identifies redundant steps, paper-intensive tasks, bottlenecks, and other inefficiencies.

**3. Design the new process:** Once the existing process is mapped and measured in terms of time and cost, the process design team will try to improve the process by designing a new one. A new streamlined “to-be” pro- cess will be documented and modeled for comparison with the old process.

**4. Implement the new process:** Once the new process has been thoroughly modeled and analyzed, it must be translated into a new set of procedures and work rules. New information systems or enhancements to existing systems may have to be impleme

nted to support the redesigned process. The new process and supporting systems are rolled out into the business organization. As the business starts using this process, problems are uncov- ered and addressed. Employees working with the process may recommend improvements.

**5. Continuous measurement:** Once a process has been implemented and optimized, it needs to be continually measured. Why? Processes may deteriorate over time as employees fall back on old methods, or they may lose their effectiveness if the business experiences other changes.

Although many business process improvements are incremental and ongoing, there are occasions when more radical change must take place.

BPM poses challenges. Executives report that the largest single barrier to successful business process change is organizational culture. Employees do not like unfamiliar routines and often try to resist change. This is especially true of projects where organizational changes are very ambitious and far-reaching. Managing change is neither simple nor intuitive, and companies committed to extensive process improvement need a good change management strategy

1. **Explain** the Systems Development Process **covering all the steps**

Systems development is a structured kind of problem solved with distinct activities. These activities consist of systems analysis, systems design, program ming, testing, conversion, and production and maintenance

**Systems analysis** involves analyzing a problem within a firm that requires an information system solution. The process includes defining the problem, identifying its causes, specifying a solution, and determining the necessary information requirements for a system solution. The systems analyst creates a roadmap of the organization, detailing primary data owners, users, and existing technology. Through document examination, observations, and interviews, the analyst identifies problems in current systems and proposes solutions, often involving the development or enhancement of an information system.

Additionally, a feasibility study is conducted to assess the financial, technical, and organizational viability of the proposed solution. This study evaluates whether the system is a sound investment, if the required technology is available and manageable by the organization's specialists, and whether the organization can adapt to the introduced changes. The systems analysis process typically presents multiple alternative solutions, with a written proposal report outlining the costs, benefits, advantages, and disadvantages of each. Management then decides the most desirable alternative based on a mix of these factors.

**Systems design** is the phase that follows systems analysis and focuses on how the information system will achieve its defined objectives. It serves as the overall plan or model for the system, similar to a blueprint for a building. The systems designer outlines specifications that encompass managerial, organizational, and technological aspects, addressing all components necessary for the system solution identified in the analysis phase. The design phase involves creating a unique blend of technical and organizational elements to fulfill user requirements efficiently and effectively within specific constraints such as technical capabilities, organizational needs, financial resources, and time limitations.

**Programming**

During the **programming** stage, system specifications that were prepared during the design stage are translated into software program code. Today, many organizations no longer do their own programming for new systems. Instead, they purchase the software that meets the requirements for a new system from external sources such as software packages from a commercial software vendor, software services from an application service provider, or outsourcing firms that develop custom application software for their clients.

**Testing**

Exhaustive and thorough **testing** must be conducted to ascertain whether the system produces the right results. Testing answers the question, “Will the system produce the desired results under known conditions?” Some companies are starting to use cloud computing services for this work.

The amount of time needed to answer this question has been traditionally underrated in systems project planning (see Chapter 14). Testing is time-consuming: Test data must be carefully prepared, results reviewed, and corrections made in the system. In some instances, parts of the system may have to be redesigned. The risks resulting from glossing over this step are enormous.

Testing an information system can be broken down into three types of activi- ties: unit testing, system testing, and acceptance testing. **Unit testing**, or program testing, consists of testing each program separately in the system. It is widely believed that the purpose of such testing is to guarantee that programs are error- free, but this goal is realistically impossible. Testing should be viewed instead as a means of locating errors in programs, focusing on finding all the ways to make a program fail. Once they are pinpointed, problems can be corrected.

**System testing** tests the functioning of the information system as a whole. It tries to determine whether discrete modules will function together as planned and whether discrepancies exist between the way the system actually works and the way it was conceived. Among the areas examined are performance time, capacity for file storage and handling peak loads, recovery and restart capabilities, and manual procedures.

**Acceptance testing** provides the final certification that the system is ready to be used in a production setting. Systems tests are evaluated by users and reviewed by management. When all parties are satisfied that the new system meets their standards, the system is formally accepted for installation.

The systems development team works with users to devise a systematic test plan. The **test plan** includes all of the preparations for the series of tests we have just described. **Conversion** is the process of changing from the old system to the new system. Four main conversion strategies can be employed: the parallel strategy, the direct cutover strategy, the pilot study strategy, and the phased approach strategy.

In a **parallel strategy,** both the old system and its potential replacement are run together for a time until everyone is assured that the new one func- tions correctly. This is the safest conversion approach because, in the event of errors or processing disruptions, the old system can still be used as a backup. However, this approach is very expensive, and additional staff or resources may be required to run the extra system.

The **direct cutover strategy** replaces the old system entirely with the new system on an appointed day. It is a very risky approach that can potentially be more costly than running two systems in parallel if serious problems with the new system are found. There is no other system to fall back on. Dislocations, disruptions, and the cost of corrections may be enormous.

The **pilot study strategy** introduces the new system to only a limited area of the organization, such as a single department or operating unit. When this pilot version is complete and working smoothly, it is installed throughout the rest of the organization, either simultaneously or in stages.

The **phased approach strategy** introduces the new system in stages, either by functions or by organizational units. If, for example, the system is introduced by function, a new payroll system might begin with hourly workers who are paid weekly, followed six months later by adding salaried employees (who are paid monthly) to the system.

Moving from an old system to a new one requires that end users be trained to use the new system. Detailed **documentation** showing how the system works from both a technical and end-user standpoint is finalized during conversion time for use in training and everyday operations. Lack of proper training and documentation contributes to system failure, so this portion of the systems development process is very important.

**Production and Maintenance**

After the new system is installed and conversion is complete, the system is said to be in **production**. During this stage, the system will be reviewed by both users and technical specialists to determine how well it has met its orig- inal objectives and to decide whether any revisions or modifications are in order. In some instances, a formal **postimplementation audit** document is prepared. After the system has been fine-tuned, it must be maintained while it is in production to correct errors, meet requirements, or improve process- ing efficiency. Changes in hardware, software, documentation, or procedures to a production system to correct errors, meet new requirements, or improve processing efficiency are termed **maintenance**.

Approximately 20 percent of the time devoted to maintenance is used for debugging or correcting emergency production problems. Another 20 per- cent is concerned with changes in data, files, reports, hardware, or system software. But 60 percent of all maintenance work consists of making user enhancements, improving documentation, and recoding system components for greater processing efficiency. The amount of work in the third category of maintenance problems could be reduced significantly through better systems analysis and design practices.

Question 2: **2.5 marks (2 to 3 pages in total)**

Explain **each** of the topics in about **8 to 10 lines**

1. **Object-Oriented Development**

**Object-oriented development** addresses these issues. Object-oriented development uses the **object** as the basic unit of systems analysis and design. An object combines data and the specific processes that operate on those data. Data encapsulated in an object can be accessed and modified only by the operations, or methods, associated with that object. Instead of passing data to procedures, programs send a message for an object to perform an operation that is already embedded in it. The system is modeled as a collection of objects and the relationships among them. Because processing logic resides within objects rather than in separate software programs, objects must collaborate with each other to make the system work.

Object-oriented modeling is based on the concepts of *class* and *inheritance*. Objects belonging to a certain class, or general categories of similar objects, have the features of that class. Classes of objects in turn can inherit all the structure and behaviors of a more general class and then add variables and behaviors unique to each object. New classes of objects are created by choosing an existing class and specifying how the new class differs from the existing class, instead of starting from scratch each time.

Object-oriented development is more iterative and incremental than traditional structured development. During analysis, systems builders document the functional requirements of the system, specifying its most important properties and what the proposed system must do. Interactions between the system and its users are analyzed to identify objects, which include both data and processes. The object-oriented design phase describes how the objects will behave and how they will interact with one other. Similar objects are grouped together to form a class, and classes are grouped into hierarchies in which a subclass inherits the attributes and methods from its superclass. Object-oriented frameworks have been developed to provide reusable, semicomplete applications that the organization can further customize into finished applications.

1. **Computer-Aided Software Engineering**

**Computer-aided software engineering (CASE)**—sometimes called *computer-aided systems engineering*—provides software tools to automate the methodologies we have just described to reduce the amount of repeti- tive work the developer needs to do. CASE tools also facilitate the creation of clear documentation and the coordination of team development efforts. Team members can share their work easily by accessing each other’s files to review or modify what has been done. Modest productivity benefits can also be achieved if the tools are used properly.

CASE tools provide automated graphics facilities for producing charts and diagrams, screen and report generators, data dictionaries, extensive reporting facilities, analysis and checking tools, code generators, and documentation generators. In general, CASE tools try to increase productivity and quality by:

* Enforcing a standard development methodology and design discipline
* Improving communication between users and technical specialists
* Organizing and correlating design components and providing rapid access to them using a design repository
* Automating tedious and error-prone portions of analysis and design
* Automating code generation and testing and control rollout

CASE tools contain features for validating design diagrams and specifica- tions. CASE tools thus support iterative design by automating revisions and changes and providing prototyping facilities. A CASE information repository stores all the information defined by the analysts during the project. The reposi- tory includes data flow diagrams, structure charts, entity-relationship diagrams, data definitions, process specifications, screen and report formats, notes and comments, and test results.

To be used effectively, CASE tools require organizational discipline. Every member of a development project must adhere to a common set of naming conventions and standards as well as to a development methodology. The best CASE tools enforce common methods and standards, which may discourage their use in situations where organizational discipline is lacking.

1. **Prototyping**

**Prototyping** consists of building an experimental system rapidly and inexpen- sively for end users to evaluate. By interacting with the prototype, users can get a better idea of their information requirements. The prototype endorsed by the users can be used as a template to create the final system. The **prototype** is a working version of an information system or part of the system, but it is meant to be only a preliminary model. Once operational, the prototype will be further refined until it conforms precisely to users’ require- ments. Once the design has been finalized, the prototype can be converted to a polished production system.

The process of building a preliminary design, trying it out, refining it, and trying again has been called an **iterative** process of systems development because the steps required to build a system can be repeated over and over again. Prototyping is more explicitly iterative than the conventional life cycle, and it actively promotes system design changes. It has been said that prototyp- ing replaces unplanned rework with planned iteration, with each version more accurately reflecting users’ requirements.

**Steps in Prototyping**

Figure 13.10 shows a four-step model of the prototyping process, which consists of the following:

*Step 1: Identify the user’s basic requirements*. The systems designer (usually an information systems specialist) works with the user only long enough to capture the user’s basic information needs.

*Step 2: Develop an initial prototype*. The systems designer creates a working prototype quickly, using tools for rapidly generating software.

*Step 3: Use the prototype*. The user is encouraged to work with the system to determine how well the prototype meets his or her needs and to make suggestions for improving the prototype.

*Step 4: Revise and enhance the prototype*. The system builder notes all changes the user requests and refines the prototype accordingly. After the prototype has been revised, the cycle returns to Step 3. Steps 3 and 4 are repeated until the user is satisfied.

When no more iterations are required, the approved prototype then becomes an operational prototype that furnishes the final specifications for the applica- tion. Sometimes the prototype is adopted as the production version of the system.

1. **End-User Development**

Some types of information systems can be developed by end users with little or no formal assistance from technical specialists. This phenomenon is called **end-user development**.

In **end-user development**, however, IS professionals play a consulting role while you do your own application development. Sometimes, user consultants may be avail- able to help you and other end users with your application development efforts. This may include training in the use of application packages; selection of hardware and soft- ware; assistance in gaining access to organization databases; and, of course, assistance in analysis, design, and implementation of the business application of IT that you need.

End-user computing also poses organizational risks because it occurs outside of traditional mechanisms for information systems management and control. When systems are created rapidly, without a formal develop- ment methodology, testing and documentation may be inadequate. Control over data can be lost in systems outside the traditional information systems department. To help organizations maximize the benefits of end-user applica- tions development, management should control the development of end-user applications by requiring cost justification of end-user information system projects and by establishing hardware, software, and quality standards for user-developed applications.

# Rapid Application Development (Rad)

Object-oriented software tools, reusable software, prototyping, and fourth- generation language tools are helping systems builders create working systems much more rapidly than they could using traditional systems-building methods and software tools. The term **rapid application development (RAD)** is used to describe this process of creating workable systems in a very short period of time. RAD can include the use of visual programming and other tools for building graphical user interfaces, iterative prototyping of key system ele- ments, the automation of program code generation, and close teamwork among end users and information systems specialists. Simple systems often can be assembled from prebuilt components. The process does not have to be sequen- tial, and key parts of development can occur simultaneously.

Sometimes a technique called **joint application design (JAD)** is used to accelerate the generation of information requirements and to develop the initial systems design. JAD brings end users and information systems specialists together in an interactive session to discuss the system’s design. Properly prepared and facilitated, JAD sessions can significantly speed up the design phase and involve users at an intense level.

**Agile development** focuses on rapid delivery of working software by breaking a large project into a series of small subprojects that are completed in short periods of time using iteration and continuous feedback. Each mini-project is worked on by a team as if it were a complete project, including planning, requirements analysis, design, coding, testing, and documentation. Improvement or addition of new functionality takes place within the next itera- tion as developers clarify requirements. This helps to minimize the overall risk, and allows the project to adapt to changes more quickly. Agile methods empha- size face-to-face communication over written documents, encouraging people to collaborate and make decisions quickly and effectively.

**References:**

1. Kenneth C. Laudon, Jane P. Laudon, 2014 Management Information System 13th
2. [James A O'Brien](https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&field-author=James+A+O%27Brien&text=James+A+O%27Brien&sort=relevancerank&search-alias=books) , [George M. Marakas](https://www.amazon.com/s/ref=dp_byline_sr_book_2?ie=UTF8&field-author=George+M.+Marakas&text=George+M.+Marakas&sort=relevancerank&search-alias=books) July 1, 2017   Management Information Systems