System development

Case Study: Boom Constructions (BC) is a growing organisation and presently, various projects are in progress. In a management meeting, the Project Managers had complained to the CEO that the information system is unable to generate accurate and timely information. After extensive discussion among the various stakeholders, it has finally been decided to replace the existing system with a new system.
The management has hired your services to conduct a study and submit a feasibility report in respect of the above.
Required:
(a) Briefly describe the steps involved and the matters which should be considered in developing the feasibility report. (09)
(b) List the key control objectives which BC should aim to achieve while designing information system. (05)

The **steps** that should be taken and the matters that should be considered while conducting **feasibility study** are as follows:

- Understand the users' needs and requirements.
- Identify a range of possible solutions i.e., in-house development, outsourcing development or acquiring an off-the-shelf package.
- Evaluate pros and cons of possible solutions keeping BC's environment, resources and constraints in view, such as cost, time required for implementation, availability of skilled resources, etc.
- Determine approximate cost and resources required to develop/acquire and implement the required solution.
- Define a time frame for the implementation of the required solution.
- Discuss the findings with the management of BC to ensure alignment of views.
- Consider BC's business strategy and try to align your recommended solution with it.
- Based on above, recommend the optimal solution with alternatives if any.

BC should aim to achieve the following **control objectives** while designing an information system:

- Safeguarding IT assets (hardware/network/software)
- Timely updation of information.
- Ensuring the integrity of general operating system environment.
- Ensuring data integrity.
- Compliance with organizational policies and procedures.
- Compliance with applicable laws and regulations.
- Ensuring continuity of business operations.

Prototyping approach VS SDLC:
Analysis of merits and demerits of the two approaches is as follows:
(i) After a quick requirements gathering phase, a prototype application is built and presented to the application users. This saves significant time and cost, as compared to normal SDLC models.
(ii) In prototyping more frequent feedbacks are taken from the users(as against the SDLC approach) which help to improve or add functionality to the application.
(iii) Prototyping makes it possible for programmers to present a mock-up version of an envisaged system to the users before a substantial amount of time and money has been committed. The users can judge the prototype before things have gone too far to be changed. Where as in SDLC model, any meaningful sample of system cannot be seen by the user, as it is based on strict planning followed by consultation, creation, testing, documentation and then launching.
(iv) In case of prototyping early involvement of the user may result in lesser initiative on the part of programmer/system analyst.
(v) In prototyping the developers mainly focus on what the user wants and what the user sees and may miss some of the controls that come out of the SDLC approach such as backup/recovery, security and audit trails etc.
(vi) Prototyping often leads to functions or extras being added to the system that are not included in the initial requirements document. In such cases sometimes the final system ends up being functionally rich but inefficient. With an SDLC model, developers in the beginning would have a clear idea on what is to be built.

Key steps involved in prototyping are as follows:
1. Elicit user requirements – briefly, not as comprehensive as other SDLC models.
2. Plan Prototype.
3. Design prototype – in high level languages.

4. Demonstrate prototype to the users.
5. Obtain users' comments on prototype.
6. Improve prototype.
7. Repeat steps 5 and 6 – until necessary.
8. Build production system (generally, in low level languages).

SDLC audit:
While auditing the feasibility study, functional requirements and technical specifications I would:
(i) evaluate if alternative systems were considered before selecting the proposed system,
(ii) determine if the information needs of strategic management, employees, customers and other business stakeholders have been analyzed,
(iii) evaluate whether the proposed system would be able to meet the business requirements,
(iv)evaluate if the cost justification/benefits are verifiable and based on appropriate parameters,
(v)evaluate the reasonableness of documentation produced during system investigation, analysis and design phases,
(vi)evaluate if the specifications developed for the hardware, software, people, network and the information products satisfy the functional requirements of the proposed system, and
(vii)check if a project management plan has been made and approved by the management.

While **auditing the testing and implementation phase**, I would:
(i) Review the test plan for completeness with evidence that user has participated actively.
(ii) Review the signoff documents to make sure all areas have been tested by right users.
(iii) Interview users for their understanding and level of participation.
(iv) Check if users training have been conducted.
(v) Perform some parallel testing to evaluate users testing results.
(vi) Review system documentation to ensure that all updates from the testing phase have been incorporated.
(vii) Verify all conversions of data to ensure they are correct and complete before the system is implemented.
(viii) Make sure that backup procedure is in place, in case implementation fails.
(ix) Selection of correct timing for implementation considering level of business activity and peak times.

Case Study: IT department of Sana Textiles Mills is headed by its CFO and consists of a senior programmer, a database administrator and two junior programmers. It has recently developed an integrated Information System which has six interconnected modules and is ready to go live. The senior programmer has developed the program and linked all the modules. Junior programmers have assisted him in gathering user requirements, preparing system flowcharts, developing user manual, compiling technical reference manual, designing various forms and reports and unit testing.
Required:
(a) With reference to the composition of the development team, identify any two risks and suggest the steps that may be taken to address those risks. (04)
(b) Explain the term 'unit testing'. Also describe briefly the various types of tests which may be performed under 'whole-of-program testing'. (05)

(a) (i) **Lack of segregation of duties:**
As no programmer other than senior programmer have knowledge of source code, he may leave back doors in the program or may make unauthorised changes in the program later on.
In order to address this risk, STM should:
- get the source code reviewed by an expert before implementation.
- implement strong change management controls.

(ii) **IT department is headed by non-IT professional**
The CFO may be conversant with IT but he may lack thorough knowledge of IT. This may result in poor IT strategic planning, inadequate monitoring of senior developer and database administrator, failure to implement appropriate change management practices and segregation of duties etc.
In order to address this risk, STM should:
- hire a seasoned IT professional as its IT head.
- reassess IT policies, procedures and plans.

(b) **Unit testing** is a testing technique that is used to test program logic within a particular program or module.

Whole-of-program testing focuses on the program, in total, to establish whether it meets the requirements. Following types of test may be performed under this testing:

Function test	**These tests are conducted by programming team to ensure that integrated programs are meeting the user requirements.**
Performance test	These tests are conducted by programming team to check whether the program meets with some performance criteria like fault tolerance etc.
Acceptance test	Acceptance tests are conducted by end-users, which focuses on meeting with the requirements and any performance issues.
Installation test	These tests are performed by programming team in the operational environment. For example, this kind of test might be conducted on actual operating machines and environment which is later used to run the actual program.

Following **areas** should be covered in a **software testing strategy:**
(i) **Strategy approach**: testing strategy should detail the approach to be taken for the testing, tests to be conducted, and tools/techniques to be used.
(ii) **Test plan**: The plan should state what will be tested, in what sequence (when) and the test environment.
(iii) **Test design**: The logic and reasoning behind the design of the tests should be explained.
(iv) **Performing comprehensive tests**: Detailed procedures for all tests to ensure consistency in testing.
(v) **Documentation**: Results of tests must be documented for future reference, including errors and starting point for error correction procedures.
(vi) **Re-testing**: After correction, all aspects of the software should be re-tested to ensure the corrections have not affected other aspects of the software.

Following are the limitations of software testing due to which bugs/errors may have remained undetected in spite of rigorous testing of the software application by AEW's team:
(i) **Poor testing process**
 Testers may not be adequately trained.
 All areas/functionality may not be covered.
 Testing may not be documented.
 Changes made to correct the errors detected during the test may not have been adequately tested subsequent to the change.

(ii) **Inadequate time**
 Due to time pressures, shortcuts may be taken and
 testing time may be reduced.

(iii) **Future requirements were not anticipated**: Range of the test data may have been used to cater the existing requirements. The errors could have occurred had future requirements been tested.

(iv) **Inadequate test data**: Test data may not be selected to test "positively" as well as "negatively", i.e. it does what it should do, and doesn't do what it shouldn't do.

System testing is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. This testing is performed by a technical person, usually in test environment before implementing the system.
User Acceptance Testing (UAT) is a process to obtain confirmation by the owner or client of the object under test, through trial or review, that the new system meets mutually agreed-upon requirements.

The **steps** taken for User Acceptance Testing typically involve the following:
(i) User Acceptance Test (UAT) Planning
(ii) Designing UA Test Cases
(iii) Selecting a Team that would execute the (UAT) Test Cases
(iv) Executing Test Cases
(v) Documenting the Defects found during UAT
(vi) Resolving the issues/Bug Fixing
(vii) Sign Off

Tests performed to ensure that the system will remain available and its efficiency will not be compromised on account of simultaneous log in by a number of users:
Load Testing

It is used to test the expected usage of system (software) by simulating multiple users accessing the system's services concurrently.

Stress / Volume / Bulk Testing

It is used to test the raised usage of system (beyond normal usage patterns) in order to test the system's response at unusually high or peak load.

Performance Testing

It is used to determine how fast the system performs under different workloads.

SDLC risks:

PEOPLE-RELATED RISKS

- Lack of follow-up on the part of top management
- Weak project manager
- Limited stakeholder involvement and/or participation
- Stakeholder conflicts
- Weak commitment of project team
- Team members lack requisite knowledge and/or skills
- Subject matter experts may be overscheduled
- Users' feedback may be inadequate
- Users' resistance to change

PROCESS-RELATED RISKS

- Lack of documented requirements and/or success criteria
- Ineffective change control process (change management)
- Ineffective schedule planning and/or management
- Communication breakdown among stakeholders
- Resources assigned to a higher priority project
- Inadequate or misused methods
- Scope creep
- Poor Testing

Case study: You are employed in a firm of chartered accountants. This is your second year as the audit supervisor on the audit of Greet Bank Limited. The bank has made considerable progress during the year under review which includes introduction of online banking and increase in the number of branches.

This year you intend to adopt "through the computer" approach as against "around the computer" approach followed last year.

Required:

(a) Justify the audit approach adopted last year and explain the reasons for the change in approach for the current year. (08)

(b) Identify the difficulties which may arise while using "through the computer" approach. (02)

(a) The audit approach adopted last year was correct for that period, because:

- Bank's systems were relatively simple.
- A clear audit trail existed.
- Reliance was placed on user controls.
- It was also cost-effective to audit by adopting the approach 'around the computer'.

Following are the **reasons** for adopting the "**through the computer**" approach for the current year's audit.

- The inherent risk associated with the new application systems (online banking application) launched by GBL is high.
- The volume of data being processed through computers this year is greater as compared to the last year, which makes it difficult to undertake extensive checking of the validity of input and output, without the use of audit software.
- Significant parts of the internal control system are embodied in the computer system.
- The processing logic embedded within the application system is complex.
- Because of the cost-benefit consideration (by the bank management), substantial gaps in the visible audit trial are likely to exist in the system.
- Due to introduction of online banking, there may be some regulatory requirement to audit through the computer.

(b) The following **difficulties** may arise while using 'through the computer' approach:
 It may be costly, especially in terms of man hours that must be expended to understand the internal working of an application system.
 Technical expertise may be needed in order to understand how the system works.

Case study: (a) During a review of the newly developed Purchase System of Fun Engineering (FE), the IS auditor noted the following:
the software does not follow the business flow: sequence of data entry fields do not match the Purchase Order (PO), therefore data entry is slow.
users have doubts over data integrity as they encounter various errors in the reports. Two common errors noted in various purchase orders are:
– incorrect total amount
– incomplete address of the supplier
The IS auditor has reported that during the software development process the Systems Analysis, Prototyping and User Acceptance Testing were not carried out properly, thereby leading to these issues.
Required:
(i) Describe the possible weaknesses during the three phases identified by the IS auditor which could have resulted in above problems. (06)
(ii) Discuss whether the common errors identified above are enough to challenge the data integrity. (02)

(a) (i) The problems described in the question might have been resulted because of the following possible weaknesses during Systems Analysis, Prototyping and User Acceptance Testing:
Systems Analysis
 Current business processes may not be carefully observed and/or documented.
 All relevant users may not be thoroughly interviewed about current procedures and/or future requirements.
 Flowcharts and process documents may not be prepared or inaccurately prepared for some of the systems or processes.
 Flowcharts and process documents may not be formally signed-off by the user at the end of this phase.
Prototyping
 Prototype may not contain all forms and processes.
 Prototype may not be tested properly.
 Lack of communication between users and development team during prototyping.
 Users' feedback on the prototype may not be incorporated in the actual system.
User Acceptance Testing (UAT)
 Some key users may not be involved in UAT.
 Tests may not be properly planned.
 Users' recommendations based on UAT may not be incorporated in the system.
 UAT may not be formally signed off by the users.

(ii) The cause of these errors is required to be investigated before challenging the data integrity. As such errors may occur due to coding flaws e.g., incorrect formula` or insufficient text box size in the report (Purchase Order in this case).

Case Study: (b) Though FE's purchase system is capable of electronic approval of the POs, the General Manager (GM) who is authorised to approve POs insists on hard copy of PO for approval as he has the following concerns:
although various controls on password are in place, such as complex structure, periodic change, restriction to re-use the last three password and account lockout after three attempts, he suspects that his password may be known to IT personnel;
the developer (programmer) may make changes in the rejected / approved PO; and
how would he defend himself if somebody claims that the GM has authorised a PO which in fact has not been approved by him?
Required:
As the IT Manager of FE, inform the GM about various controls which are in place to address his concerns. (08)

Concern 1: GM's password may be known to IT personnel
Following controls are in place to address the above issue:
(i) Mandatory change of initial password on first log-in.
(ii) Passwords are stored in irreversible encrypted format and therefore passwords stored in the system are not readable under ordinary situations.

(iii) GM login ID can only be used from a specific network port and IP address.

Concern 2: Developer (programmer) may make changes in the rejected/ approved PO

Following controls are in place to address the above issue:

(i) Once a PO is approved it cannot be edited. If any change in PO is required, the old PO is to be cancelled and a new PO is to be raised/generated.

(ii) Approved POs are digitally signed by the GM; this digital signature becomes invalid as soon as the PO is modified.

(iii) Developers access is restricted to the development environment only i.e., he cannot access/log on to the live system.

(iv) Development tools are disabled in the live environment.

Concern 3: How would he defend himself if somebody claims that he has authorized the PO issue?

Following controls are in place to address the above issue:

(i) Only GM's ID is allowed to approve PO.

(ii) Besides User ID, Network IP, Machine address, approval date and time has been logged on processing PO approval.

(iii) The above log is periodically analyzed both manually and automatically. Procedure is in place to investigate the detected exceptions.

To ensure successful **data migration** following **objectives** should be achieved:

 Completeness: Ensure the completeness of the data conversion i.e., the complete data is converted from source to destination

 Integrity: The data should not be altered by the person or program during transfer to the new system.

 Confidentiality: The confidentiality of the data should be ensured.

 Consistency: Ensure that the data is consistent within the defined ranges of data conversion.

Key steps that should be taken during **data conversion** are as follows:

(i) Establish the parameters/criteria for successful conversion.

(ii) Identify business owners responsible for data conversion validation and signing off.

(iii) Determine what data should be converted programmatically and what, if any, should be converted manually.

(iv) Perform the data cleansing ahead of conversion.

(v) Identify the methods to be used to verify the conversion, such as automated file compressions, comparing record counts and control totals etc.

(vi) Scheduling the sequence of data conversion tasks

(vii) Design audit trail reports to document the conversion, including data mappings and transformations.

(viii) Design exception reports that will record any items that cannot be converted automatically.

(ix) Development and testing of conversion programs, including functionality and performance.

(x) Performing one or more conversion rehearsals to familiarize persons with the sequence of events and their roles and to test conversion process end-to-end with real data.

(xi) Running the actual conversion with all necessary personnel onsite, or at least able to be contacted.

(xii) Final testing of the converted data.

Possible **changeover techniques** for the complete deployment of new system:

Parallel Changeover

This technique includes the running of both existing (old) and new software in parallel and shifting over to the news system after fully gaining confidence on the working of new software.

Phased Changeover

In this approach, the older system is broken into deliverable modules. Initially, the first module of the older system is phased out using the first module of the newer system. Then, the second module of the older system is phased out, using the second module of the newer system and so forth till the last module.

Abrupt / Direct / Plunge Changeover

In this approach the new system is introduced on a cutoff date / time and the older system is discontinued simultaneously.

Pilot Changeover

In this approach, the new system is implanted at a selected location of the company, such as only one branch office (using direct or parallel changeover approach). After the system proves successful at the selected location (pilot site), it is implemented into the rest of the organization.

Changeover to the newer system broadly involves four major **steps**:

(i) Training to the employees or users.

(ii) Installation of new hardware, operating system, application system.
(iii) Conversion of files and programs and migration of data.
(iv) Scheduling of operations and test running for go-live or changeover.

Probable **risks** during changeover process include:
(i) Loss of assets.
(ii) Data corruption / deletion.
(iii) Loss of confidentiality.
(iv) Impairment of system effectiveness.
(v) System efficiency may be affected.
(vi) Resistance from staff.

For effective control of the **change management process** following control measures are recommended.
(i) **Requisition**
The request for change may be raised by users, or by IT department/personnel itself. While making such a request, appropriate justification should be provided.
(ii) **Authorization**
The request should be assessed and authorized for development by a more senior level person or a committee. The person responsible for making the changes should be identified and duly authorised.
(iii) **Development and programmer testing**
The requested change should be developed and tested in test environment to ensure that it does not make any unwanted changes in the associated programs and routines.
(iv) **User Acceptance Testing**
Once the change has been developed, it should be tested adequately by the user to ensure that it achieves the desired objective.
(v) **Approval**
After successful user acceptance testing, the change must be formally approved and documented before being moved/implemented in the live/production environment (transport approval).
(vi) **Segregation of incompatible duties**
The change should be implemented by someone other than the person requesting the change. A developed change should be transported by someone other than the developer. The developers should not have access to the live/production environment.

The company may face the following **consequences** in the absence of proper change management controls:
(i) Increased risk and security vulnerabilities.
(ii) Assessment of impact on other associated areas/programs may be ignored.
(iii) Lack of accountability for unauthorized changes.
(iv) Undocumented changes resulting in poor documentation.
(v) Business interruptions may occur due to uncontrolled changes.
(vi) Poor audit findings.
(vii) Loss of confidence on system security and data integrity.
(viii) Potential fines and other disciplinary measures due to incorrect reporting or submissions to government authorities.

Following **policies** may be implemented to ensure proper change management controls:
(i) All production devices must be monitored for changes.
(ii) All changes should be recorded, explained, and documented.
(iii) Change implementers should not authorize their own changes.
(iv) All changes must be tested in development environment before being implemented live.
(v) All users who may be affected should be notified of the change.
(vi) View point of all related users should be obtained to assess the impact of change on other associated areas/programs.
(vii) Change successes and failures should be tracked.
(viii) The authority levels should be well defined.
(ix) No changes to production assets should be allowed outside scheduled maintenance windows.
(x) All unauthorized changes must be investigated.
(xi) Audit trails should be tracked regularly.
(xii) Exception reports should be designed and reviewed regularly.

Case study: Transparent Glass (TG) has implemented its information systems last year. Since then, various changes have been made on users' request or to overcome errors. Record of such changes consists of copies of any one or more of the following:

- users' emails
- screenshots of errors
- hand written directives of CEO and CFO

According to IT head, each change is implemented in live environment by the program developer after through testing.

The **issues** in the TG's system of **documenting and implementing changes** are as follows:

(i) **Absence of standard format for recording change requests:**
There is no single standard format of recording requests/justification for changes. In the absence of such a change it may be difficult to ascertain that who raised the change request, who authorised the change, who tested the change, extent of testing performed etc.

(ii) **Inadequate evidence of change request authorisation:**
Users email may only contain request for change while errors screen shots may contain only identification of error. All changes must be authorised at an appropriate level and should be recorded in a uniform manner, otherwise, it would be difficult to distinguish between authorised and unauthorised changes.

(iii) **Lack of change assessment procedure:**
TG is continuously making changes in its program without evaluating their risks and benefits. Changes made in such a manner may lead to unnecessary changes that in turn may affect the overall performance of the information system.

(iv) **Absence of fall back/recovery procedures:**
If after a certain change the program starts functioning in an untoward manner, no fall back or recovery procedure is specified to minimise the impact of failure. In such a case, TG's information system may go down for an indefinite period.

(v) **Absence of users' acceptance testing:**
Changes have been implemented after the developer's testing only. No user acceptance testing and sign–off is being obtained. In the absence of UAT it is difficult to ensure that changes are made as intended.

(vi) **Absence of formal approval of implementing the change:**
Instead of the having a formal approval of implementing the change by an appropriate authority, apparently the decision to implement the change rests with the developers who implement the change if he is satisfied with his testing. Thus, he may also implement changes which are not authorised or not fully tested. Besides affecting the functionality such changes may compromise the integrity of the TG's information system.

(vii) **Lack of segregation of duties:**
TG's developer is implementing the change in the live environment himself. This may give rise to the risk of unauthorised changes. The change must be implemented by someone other than the person who developed the change.

(viii) **Absence of monitoring the change after implementation:**
Apparently no procedure in place to monitor the changes for a period of time after implementation. Hence, any untoward or conflicting functionality caused due to a recent change could remain undetected for a long time and may prove more damaging and difficult to address later on.

Key contents of Request for proposal (RFP):

Information given **to** vendors

(i) Broad background of the organization requesting for proposal.
(ii) Details of the information technology environment.
(iii) Requirements of the system for which proposal has been requested.
(iv) How will the proposal be evaluated?
(v) Criteria for the eligibility of the vendors.
(vi) General procurement policies (if any).
(vii) The format of the proposal to facilitate comparative evaluation of the proposal.
(viii) Identifying the timing of submission, including any bonds that may be required and the place and manner of submission.

Information required **from** vendor

(i) Source code availability.
(ii) Minimum hardware requirements for the proposed software
(iii) Availability of the offered product's complete and reliable documentation.
(iv) List of recent or planned enhancements to the product, with dates.
(v) List of clients using the offered product.

(vi) Availability of support status (24 X 7 online help, onsite maintenance etc).
(vii) Provision for staff training.
(viii) Evidence of vendor's financial stability.
(ix) Evidence of relevant experience.

Key activities in ensuring **transparency** in receiving and recording RFPs:
(i) Advising all suppliers of the format (including method of submission e.g. sealed envelopes, by post etc.) and deadline for submissions and the place where the submission should be lodged.
(ii) Ensuring that all vendors have equal and adequate time to submit the proposal.
(iii) Ensuring that all bids are opened at the same time and in the presence of suppliers.

Key activities involved in **short listing** the proposals:
(i) Eliminating proposals from vendors that do not meet the minimum requirements specified in the RFP. The reason for this should be documented and preferably communicated to the supplier.
(ii) Evaluating the remaining proposals so that the relative merits and weaknesses of each solution are documented and compared.
(iii) Eliminating all but a few proposals from further consideration, documenting the reasons for rejection and advising the suppliers who have been short listed.

The project team may arrange the following to **validate the vendors' responses**:
- Walkthrough tests
- Demonstrations
- Benchmark tests
- Visiting or calling the vendors' current clients to verify his claims.

Following **factors** may be considered by the company while choosing an **off-the-shelf software package**:

User requirements: The package should fit the users' particular requirements such as report production, anticipated volume of data, data validation routines, data security and recovery and ease of use.
Processing times: Response time should be fast enough to cater current and future requirements.
Documentation: The software should accompany full and clear documentation. User manuals should be easy to understand.
User friendliness: The software should be easy to use with user friendly menus and clear on-screen prompts/alerts and provide wizards for generating queries and reports.
Controls / Security: Appropriate logical controls should exist in the software such as strong passwords, data validation checks, spelling checks, audit trail etc.
Maintenance: Procedure for corrective, adaptive and perfective maintenance activities should be clearly defined. Also, costs associated with such activities should also be considered.
Modification: It should be checked whether any customisation would be possible. If yes, at what price? If no, how would the organization cope with any changes in its systems and procedures?
Customers' feedback / Market reputation: Take feedback from existing users of the software as regards quality of the software and customer services provided by the vendor. Financial stability of the vendor should also be considered.
Compatibility: Would the software function on the existing hardware or the hardware would have to be upgraded/replaced. Also, how easily the data from old system would be transferred into the new system. It should also be seen whether the system would be able to exchange data with any other existing system of the company.
Support: The type and extent of support service that would be available.
Cost: Cost of software should be compared with other similar software available in the market. However, cost should not be the sole criteria.
Impact on existing staff: The company should consider if the existing staff require any special training for using the new software and if the vendor would provide necessary training to the staff. If some of them become redundant then how such staff would be managed. Also, if the company need to hire additional skilled/expert staff for managing the software, etc.