What are the reasons for which the legacy systems evolve?

Legacy systems often evolve for one or more of the following reasons:

• The software must be adapted to meet the needs of new computing environments or technology.

• The software must be enhanced to implement new business requirements.

• The software must be extended to make it interoperable with other more modern systems or databases.

• The software must be re-architected to make it viable within a network environment.

Module 2

Briefly elaborate various phases in requirement engineering.

Inception, Elicitation, Elaboration and so on (description given below)

Discuss the questionnaires to be addressed during validating requirements.

• Is each requirement consistent with the overall objectives for the system/product?

• Have all requirements been specified at the proper level of abstraction?

• Is the requirement really necessary or does it represent an add-on feature that may not be essential to the objective of the system?

• Is each requirement bounded and unambiguous?

• Does each requirement have attribution?

• Do any requirements conflict with other requirements?

• Is each requirement achievable in the technical environment that will house the system or product?

Is each requirement testable, once implemented?

• Does the requirements model properly reflect the information, function, and behavior of the system to be built?

• Has the requirements model been “partitioned” in a way that exposes progressively more detailed information about the system?

• Have requirements patterns been used to simplify the requirements model? Have all patterns been properly validated?

Explain the distinct tasks involved in Requirements Engineering.

Inception. How does a software project get started? Is there a single event that becomes the catalyst for a new computer-based system or product, or does the need evolve over time? There are no definitive answers to these questions. In some cases, a casual conversation is all that is needed to precipitate a major software engineering effort.

Elicitation. It certainly seems simple enough—ask the customer, the users, and others what the objectives for the system or product are, what is to be accomplished, how the system or product fits into the needs of the business, and finally, how the system or product is to be used on a day-to-day basis

A number of problems that are encountered as elicitation occurs.

- **Problems of scope**. The boundary of the system is ill-defined or the customers/users specify unnecessary technical detail that may confuse, rather than clarify, overall system objectives
- **Problems of understanding**. The customers/users are not completely sure of what is needed, have a poor understanding of the capabilities and limitations of their computing environment, don't have a full understanding of the problem domain, have trouble communicating needs to the system engineer, omit information that is believed to be "obvious," specify requirements that conflict with the needs of other customers/users, or specify requirements that are ambiguous or untestable.
- **Problems of volatility**. The requirements change over time.

Elaboration. The information obtained from the customer during inception and elicitation is expanded and refined during elaboration. This task focuses on developing a refined requirements model

Negotiation. It isn't unusual for customers and users to ask for more than can be achieved, given limited business resources. It's also relatively common for different customers or users to propose conflicting requirements, arguing that their version is "essential for our special needs."

Specification. In the context of computer-based systems (and software), the term specification means different things to different people. A specification can be a written document, a set of graphical models, a formal mathematical model, a collection of usage scenarios, a prototype, or any combination of these.

Validation. The work products produced as a consequence of requirements engineering are assessed for quality during a validation step. Requirements validation examines the specification5 to ensure that all software requirements have been stated unambiguously; that inconsistencies, omissions, and errors have been detected and corrected; and that the work products conform to the standards established for the process, the project, and the product.

Requirements management. Requirements for computer-based systems change, and the desire to change requirements persists throughout the life of the system. Requirements management is a set of activities that help the project team identify, control, and track requirements and changes to requirements at any time as the project proceeds

---

Discuss about establishing the ground work regarding requirements gathering.

To establish the groundwork for an understanding of software requirements—to get the project started in a way that will keep it moving forward toward a successful solution

Identifying Stakeholders: Stakeholders can be business operations managers, product managers, marketing people, internal and external customers, end users, consultants, product engineers, software engineers, support and maintenance engineers, and others. Each stakeholder has a different view of the system, achieves different benefits when the system is successfully developed, and is open to different risks if the development effort should fail

Recognizing Multiple Viewpoints: Because many different stakeholders exist, the requirements of the system will be explored from many different points of view. Each of these stakeholders (and others) will contribute information to the requirements engineering process. As information from multiple viewpoints is collected, emerging requirements may be inconsistent or may conflict with one another.

Working toward Collaboration: If five stakeholders are involved in a software project, you may have five (or more) different opinions about the proper set of requirements. The job of a requirements engineer is to identify areas of commonality (i.e., requirements on which all stakeholders agree) and areas of conflict or inconsistency (i.e., requirements that are desired

by one stakeholder but conflict with the needs of another stakeholder). It is, of course, the latter category that presents a challenge.

Asking the First Questions: Questions asked at the inception of the project should be "context free"

Who is behind the request for this work?

Who will use the solution?

What will be the economic benefit of a successful solution?

Is there another source for the solution that you need?

Comment on Quality Function Deployment

Quality function deployment (QFD) is a quality management technique that translates the needs of the customer into technical requirements for software. QFD "concentrates on maximizing customer satisfaction from the software engineering process"

QFD identifies three types of requirements

Normal requirements. The objectives and goals that are stated for a product or system during meetings with the customer. If these requirements are present, the customer is satisfied. Examples of normal requirements might be requested types of graphical displays, specific system functions, and defined levels of performance.

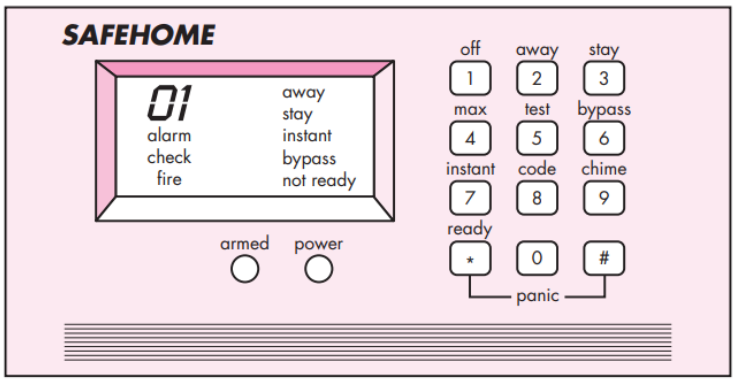
Expected requirements. These requirements are implicit to the product or system and may be so fundamental that the customer does not explicitly state them. Their absence will be a cause for significant dissatisfaction. Examples of expected requirements are: ease of human/machine interaction, overall operational correctness and reliability, and ease of software installation.

Exciting requirements. These features go beyond the customer's expectations and prove to be very satisfying when present. For example, software for a new mobile phone comes with standard features, but is coupled with a set of unexpected capabilities (e.g., multitouch screen, visual voice mail) that delight every user of the product.

Discuss the template for detailed descriptions of use cases considering SAFEHOME system as case study

Template for detailed descriptions of use cases: Use case: InitiateMonitoring

Primary actor: Homeowner.



Goal in context: To set the system to monitor sensors when the homeowner leaves the house or remains inside.

Preconditions: System has been programmed for a password and to recognize various sensors. **Trigger:** The homeowner decides to "set" the system, i.e., to turn on the alarm functions. **Scenario: 1**. Homeowner: observes control panel

2. Homeowner: enters password

3. Homeowner: selects "stay" or "away"

4. Homeowner: observes read alarm light to indicate that SafeHome has been armed

Exceptions: 1. Control panel is not ready: homeowner checks all sensors to determine which are open; closes them.

2. Password is incorrect (control panel beeps once): homeowner reenters correct password.

3. Password not recognized: monitoring and response subsystem must be contacted to reprogram password.

4. Stay is selected: control panel beeps twice and a stay light is lit; perimeter sensors are activated.

5. Away is selected: control panel beeps three times and an away light is lit; all sensors are activated.

Priority: Essential, must be implemented When available: First increment Frequency of use: Many times per day

Channel to actor: Via control panel interface

Secondary actors: Support technician, sensors

Channels to secondary actors:

Support technician: phone line

Sensors: hardwired and radio frequency interfaces

Open issues: 1. Should there be a way to activate the system without the use of a password or with an abbreviated password?

2. Should the control panel display additional text messages?

3. How much time does the homeowner have to enter the password from the time the first key is pressed?

4. Is there a way to deactivate the system before it actually activates?

Use cases for other homeowner interactions would be developed in a similar manner. It is important to review each use case with care. If some element of the interaction is ambiguous, it is likely that a review of the use case will indicate a problem.