8 Domain Modelling - exam questions

You should know the answers to these questions

Why is it necessary to validate and analyze the requirements?

- To ensure we are building the right system and to confirm our understanding of the problem domain. Validation focuses on correctness ("Are we building the system right?"), while analysis confirms adequacy ("Are we modeling the problem domain correctly?").

What's the decomposition principle for functional and object-oriented decomposition?

- **Functional decomposition** breaks down the system according to the functions it performs, leading to a centralized hierarchy.
- **Object-oriented decomposition** divides the system based on objects it manipulates, distributing responsibilities across "is-a" hierarchies.

Can you give the advantages and disadvantages for functional decomposition? What about object-oriented decomposition?

Functional Decomposition:

- **Advantages**:
  - Suited for stable requirements or single-function systems.
  - Provides a clear problem decomposition strategy.
- **Disadvantages**:
  - Poor maintainability as system functions evolve.
  - Difficult to interface with other systems.
  - Not adaptable to modern multi-functional systems.

Object-Oriented Decomposition:

- **Advantages**:
  - Better for complex, evolving systems.
  - Encapsulation increases robustness against changes.
- **Disadvantages**:
  - Requires careful design to avoid misuse like "god classes".

How can you recognize "god classes"?

- Symptoms include:
  - A few large classes managing most of the system's operations.
  - Classes with suffixes like "Manager" or "Controller."
  - Many small "provider" classes only offering accessor operations (e.g., `get`, `set` methods).

What is a responsibility? What is a collaboration?

- **Responsibility**: The knowledge or actions an object provides to others.
- **Collaboration**: The relationships between objects needed to fulfill responsibilities. Collaborators play roles to assist in fulfilling a responsibility.

Name 3 techniques to identify responsibilities.

1. **Scenario walk-throughs and role-play.**
2. **Verb phrase identification in requirements.**
3. **Class enumeration and hierarchy analysis**.

What do feature models define?

- They define reusable and configurable requirements for a family of systems, showing commonalities and variations through feature diagrams.

Give two advantages and disadvantages of a "clone and own" approach

Advantages:

1. Saves time and reduces costs.
2. Offers independence in product variants.

Disadvantages:

1. Propagating changes becomes difficult.
2. Adapting the clone is repetitive and prone to bugs.

Explain the main difference between a social fork and a variant fork

- **Social fork**: Created for contributions or improvements, usually reintegrated via pull requests.
- **Variant fork**: Maintained separately, often duplicating effort like bug fixes across variants.

How does domain modeling help to achieve correctness? Traceability?

- **Correctness**: Ensures a robust problem domain model by focusing on the "what" instead of the "how."
- **Traceability**: Links requirements to the system design via proper naming conventions and associations.

You should be able to complete the following tasks

Apply noun identification & verb identification to (a part of) a requirements specification.

- **Example**: From the National Widgets case:
  - **Nouns (Objects)**: Company, Product, Supplier, Catalogue, Customer, Order, Payment, Address.
  - **Verbs (Responsibilities)**: Purchase, Submit, Track, Ship.

Create a feature model for a series of mobile phones.

- A feature diagram could include:
  - **Core Features**: Call, Text Messaging.
  - **Optional Features**: Camera, GPS.
  - **Alternative Features**: Physical Keyboard vs. Touch Screen.

Can you answer the following questions?

How does domain modeling help to validate and analyze the requirements?

- It clarifies the system's goals by focusing on "what" needs to be achieved, ensuring alignment with customer needs. Role-playing scenarios validate use cases and help uncover gaps.

What's the problem with "god classes"?

- They lead to poor design by concentrating too many responsibilities, making the system hard to maintain and extend. They also indicate a flawed decomposition strategy.

Why are many responsibilities, many collaborators, and deep inheritance hierarchies suspicious?

- They increase complexity, reduce clarity, and hinder maintainability. Overloaded classes or excessive hierarchies indicate poor responsibility distribution.

Can you explain how role-playing works? Do you think it helps in creative thinking?

- **Role-playing** involves assigning CRC cards to participants who simulate system operations. It enhances creative thinking by fostering a shared understanding and enabling iterative refinements.

Can you compare Use Cases and CRC Cards in terms of the requirements specification process?

- Use Cases focus on capturing requirements and goals, while CRC Cards validate the design and identify responsibilities and collaborations. They complement each other.

Do CRC cards yield the best possible class design? Why not?

- No, they are a starting point. They help in brainstorming and responsibility assignment but need refinement and validation against system requirements.

Why are CRC cards maintained with paper and pencil instead of electronically?

- They are easy to manipulate, rearrange, and discard during brainstorming sessions, making them accessible to all participants.

What would be the main benefits of thinking in terms of "system families" instead of "one-of-a-kind development? What would be the main disadvantages?

Benefits:

1. Promotes reuse, reducing development time and costs.
2. Simplifies the management of commonalities and variations.

Disadvantages:

1. Higher initial investment in planning and tooling.
2. Difficulties in managing variations effectively.

Can you apply scrum to develop a product line? Argue your case.

- **Yes**. Scrum's iterative cycles can accommodate evolving requirements and manage variability in product lines. However, it may require additional layers for handling product line-specific artifacts like feature models.