BCS Professional Examination 2001
Professional Graduate Diploma

April 2001

Examiners' Report

System Design Methods

Question 1

(a) ***There are a number of general software process models which all incorporate the four fundamental activities of software specification, development, validation and evolution. Explain the purpose of each of these four activities.***

(4 marks)

(b) ***The waterfall model and the spiral model are two general software process models. With the aid of diagrams, provide a detailed explanation of each of these process models. For each process model clearly indicate how the specification, development, validation and evolution activities are incorporated.***

(14 marks)

(c) ***A distinctive feature of the spiral model is its explicit consideration of risk. Explain what is meant by the term "risk" in this context and identify three risks associated with a real-world software development project that you are familiar with. Also indicate strategies that could be employed to reduce each identified risk.***

(7 marks)

Answer Pointers

Expect answers to cover following key points:

a)
- Specification – define functionality of software and the constraints on its operation
- Development – produce the software to meet the specification
- Validation – establish whether or not the software does what the customer wants
- Evolution – software evolves to meet changing customer requirements, and also correct errors etc…

b)
Waterfall points should include:

- Phases include requirements analysis and specification, system and software design, Implementation and unit testing, Integration and system testing, operation and maintenance
- Inflexible partitioning into distinct phases since they overlap in reality

- Diagram of all phases with forward cascade and reverse feedback from any stage to all previous stages
- Not linear – several iterations are involved
- Frequent iterations make it difficult to define clear management checkpoints
- To help stages are often frozen (prematurely)
- Can lead to systems not meeting requirements (since incorrect requirements model frozen early)
- Specification, Development, Validation and Evolution relate directly to phases of model in sequence (design and implementation together encompass "Development")

Spiral model points should include:

- Recognises that risk forms basis of generic software process model
- Diagram: Graphical model spiralling out from Centre
- Each loop in the spiral represents a phase of the software process as defined by management
- No fixed phases but examples could be:
- Inner loop – feasibility study
- Next loop out – requirements specification
- Next loop out – system design
- Management decide how to structure a project into phases
- Often organisation will have a generic phase model, with extra phases for specific projects, of to account for specific problems identified
- Each loop split into 4 sectors: objective setting, risk assessment and reduction, development and validation, planning (expect sectors on diagram)
- Spiral model encompasses other process models (as sub models), hence may include prototyping, conventional WF development and formal transformations in different cycles.
- Specification, Development, Validation and Evolution could relate to different spirals (as indicated above), however may include all within one spiral (if it represents an embedded sub-process model)

c)
- A risk in this context is something that can go wrong and is a consequence of inadequate information.
- Any 3 risks identified, with suitable risk resolution strategy for each. Must be related to a tangible project for credit.

Examiner's Guidance Notes:
For part (a), candidates often made a poor distinction between specification and development. Validation was generally well understood. Many candidates did not demonstrate a clear understanding of evolution.

For part (b), many candidates did not produce clear diagrams. Most produced a reasonable explanation of the waterfall model but many of the spiral models were poorly explained. Many candidates provided long explanations but neglected to clearly indicate how the specification, development, validation and evolution activities were incorporated into each model.
For part (c), many candidates struggled to identify true risks, and associated resolution strategies, indicating a lack of understanding of this concept. Some answers lacked relationship to a real-world project.

Question 2

In literature as well as in industrial practice, a distinction is normally made between the two design paradigms: real-time systems design and object-oriented systems design.

(a) Referring these two design paradigms, discuss the key notions (such as Tasks, Processes and Objects) and their functions. Compare the ways in which concurrency is handled in these two design paradigms.

(12 marks)

(b) Suppose you are a consultant. Your client company intends to develop a temperature control system for multiple warehouses, and therefore needs to decide a suitable method for the system design and implementation. Relating to your discussion in part a), give advice to the company and provide justification for your advice.

(13 marks)

Answer Pointers

a)
Expect to mention some fundamental concepts and principles associated with high-quality software, to show understanding of the notion of quality of software. In addition, the candidate should be able to discuss some of the problems for the designer. These are examples of issues that the real-time design addresses:

- Representation of interruption and context switching
- Concurrency as manifested by multitasking and multiprocessing
- Intertask communication and synchronisation
- Representation of timing constraints
- Asynchronous processing

The candidate should also show understanding of object-oriented design by being able to discuss the following:

- Fundamental components: they are objects with their own private state and operations rather than functions.
- Objects may be designed and further implemented sequential or concurrently. A concurrent object may be a passive object whose state is only changed through its interface or an active object that can change its own state without intervention.
- The process of OO design includes activities to design the system architecture, identify objects in the system, describe the design using different object models (e.g. static –class, and dynamic – sequence, state, etc.) and document the object interfaces.

Comparison and contrast should be made between the two paradigms. Tasks and Processes are the key, unit elements in real-time design, as the main feature of such software is concurrency, manifested by multitasking and multiprocessing.

Similarly, Objects are key, unit elements in OO design. Communication between objects is achieved by sending messages, with or without time constraints. Unlike real-time design, concurrency is not necessarily a main feature.

b)
To answer this part, the candidate must be able to relate the discussion in Part. In general, the real-time design is considered as a suitable design method because of the characteristics of the problem. However, an OO design method with capability of handling concurrency is also possible, but justification must be given for the handling of the real-time features. Having considered this, the student's advice can also include other aspects, such as company's experience and expertise, consideration of reuse, etc.

Examiner's Guidance Notes:
For part a), candidates should show a clear understanding of key concepts of the two design paradigms. Explanation of the notions with reference to particular design methods would attract extra marks.

The answer to part b) builds on the part a) with a justification of the choice. Sometime candidates simply make a recommendation without giving justification, which does not meet the requirements.

Question 3

a) Structured software development methods often include data flow diagrams (DFDs), state transition diagrams (STDs) and entity relationship diagrams (ERDs) to model different aspects of systems. For each of these three diagram types:

1. ***indicate what aspect(s) of a system the diagram type should be used to model***
2. ***illustrate the notation employed***
3. ***provide an example diagram fragment that relates to a system with which you are familiar, and include a short explanation of what the diagram fragment is modelling.***

(15 marks)

b) Consider a software development team that has been using structured methods in their system development activities, but until now has only ever used simple diagram editors and word processors to document its models. The team leader is considering introducing a simple workbench CASE tool which supports the particular method that they are using. Discuss, in detail, the benefits and possible problems that the introduction of such a tool would provide.

(10 marks)

Answer Pointers

a)
Expect coverage of the following:

- DFD
- Used for modelling functional aspects of a system (flow and processing/transformation of data)

- Symbols include: process, data flow, data store, external entity
- Diagram of each symbols and usage
- STD
- Used for modelling time-dependent (dynamic) aspects of a system
- Symbols include: State, transition, condition/event and action
- Diagram to illustrate symbols and usage
- ERD
- Used for modelling static aspects of a system (relationships between data items)
- Symbols include: Entity(different kinds of), Relationship(different kinds of), cardinality mechanism
- Diagram to illustrate symbols and usage

b)
Expect students to include points such as the following:
- Benefits
- Correct method support (rules enforced)
- More efficient than editing
- Consistency checking – hence improve quality
- Problems
- Culture change – reluctance from some staff
- Legacy projects use old system – hence maintenance problem
- Cost of new software (plus platform/infrastructure possibly)
- Training cost

1 mark for each valid (clearly made) point up to maximum available. Expect balance of benefits and problems for full credit

Examiner's Guidance Notes:
For part (a), several candidates did not demonstrate a clear understanding of the different aspects of a system that each of the notations is used to model. Basic symbols described were reasonable in general, however some of the diagram examples were poor, and didn't always employ the symbols presented earlier.

For part (b), many answers started well but became repetative and missed important points. Candidates would benefit from taking time to consider the points they are going to make and "design" their answer in rough first, to ensure that it is well structured.

Question 4

(a) Software reuse can be achieved at different levels (analysis, design, coding) and from different perspectives (application system, component and function). Choosing one perspective, elaborate the notion of reuse at these different levels.

(12 marks)

(b) In the context of software project planning, identify THREE reusable artefacts and describe the mechanism by which reuse of each artefact can be achieved.

(13 marks)

Answer Pointers

a)
Expect the discussion of the reuse at three levels, and each with explanation of the reuse mechanism. Application system reuse: The whole of an application system may be reused either by incorporation it without change into other systems or by developing application families that may run on different platforms or may be specialised to the needs of particular customers.
Component reuse: Components of an application ranging in size from sub-system to single objects may be reused. For example, a pattern-matching system developed as part of a text processing system may be reused in a database management system.
Function reuse: Software components which implement a single function, such as a mathematical function, may be reused. This form of reuse is often realised through the use of library.

Alternative, the candidate could also choose another perspective of reuse: analysis, design and coding. The higher the level of abstraction of the reuse is at, the more significant benefits it will bring in.

b)
Expect to cover both the artefacts as well as mechanisms.
The artefacts are: project plans, cost estimates, architecture, requirements models and specifications, designs, source code, user and technical documentation, human interface, data, and test cases. The mechanism will be related to the artefact. These could include, department policy, good practices, use of template and standards, use of library, etc. The student must explain how the mechanism enable or encourage the use of the relevant artefact.

Examiner's Guidance Notes:
For part a), to understand the notion of reuse is the key. The discussion of reuse can be from either of the two perspectives. However, examples of each reuse are essential. Sometimes, candidates fails to give examples to demonstrate their understandings.

For part b), answers to both artefacts and mechanisms are requires. Reference to the candidates' own experience and supply of examples strengths the answer considerably and would be highly recommended.

Question 5

a) ***Discuss the relationship between software engineering process and product quality using diagrams for illustration. Based on your understanding or experience, you should include key activities in a typical process for quality checks of design documents.***

(13 marks)

b) ***Assume that you are the manager of a software development team and would like to introduce the concept of software process metrics to your team. Write a short report that could be used to introduce the team to the concept of software process metrics. Your report should explain the different classes of process metric that can be collected and should indicate how you will decide what measurements to take and how to use the results.***

(12 marks)

Answer Pointers

a)
This question examines the candidate's understanding of the theory and relating the theory to practice of software engineering. A basic assumption is that product quality is related to the production process. This is especially relevant in automated production systems (i.e. with a heavy use of CASE tools) and "mass production" systems (such as in software houses).
A diagram is required in the answer, in which it must show interaction of people of different roles, and iteration of reviews, revision and final approval. The candidate is expected to use his/her own understanding and experience.

b)
Expect consideration of:

- Different classes of process metric:
- Time taken for process to be completed: could be total, calendar, focussed on particular engineers, etc…
- Resources required for particular process: effort (person-days), travel costs, hardware resources, etc…
- Number of occurrences of a particular event: e.g. events - defect discovered, requirements change request made, etc…
- How decide what measurements to make?
- GQM approach possible
- Goal – consider what trying to achieve (e.g. improve programmer productivity)
- Question – refined goals – specific areas of uncertainty (e.g. How can number of debugged lines be increased?)
- Metrics – measurements that need to be collected to answer the specific questions
- What do with results?
- Analyse data focussing on specific questions posed
- Make sure feed into process improvement
- Re-measure when process updated
- De-focus personal issues – no link to appraisal

Examiner's Guidance Notes:

For part a), the candidate must be able to distinguish the quality of product and process, and discuss the relationship between them. Using own experience or showing examples is an effective way to demonstrate the candidate's understanding and ability of applying the theory.

For part (b), some answers again reflected a lack of thought to structure, and contained large chunks for which no credit could be awarded. Candidates were aware of software metrics in general but many lacked an understanding of "process" metrics.