Software Engineering

Software Engineering	
Richting	Informatica
Jaar	3BINF

Bespreking

Dit is een vak dat je zeker niet hoort te onderschatten want ieder jaar buizen er veel studenten op het examen. Paradoxaal genoeg is de leerstof helemaal niet zo moeilijk en wordt alles erg gedetailleerd uitgelegd door de prof. Zo goed zelfs dat als je in de les of de praktijk zit je je dikwijls zal afvragen hoe het mogelijk is dat het slaagcijfer zo laag is. Bij de voorbereiding van het examen zul je echter merken dat het wel een pak leerstof is en je vaak abstracte begrippen uit elkaar moet houden (bv error/fault/defect onderscheiden). Het ontbreken van een echte cursus waardoor je van de slides moet leren maakt het er ook niet eenvoudiger op. De laatste twee hoofdstukken (Quality control en Software Metrics) bevatten ook erg veel leerstof en komen helemaal op het einde: ze zijn vaak minder goed gekend op het examen maar zijn wel belangrijk. Moesten je oefeningen op het schriftelijk niet goed zijn, bekijk deze dan zeker nog eens tegen het mondeling want hij durft wel eens een oefening geven dan, om te zien of je deze kan.

Zorg er voor dat je op het schriftelijk examen op zen minst alle "You should be able to answer these questions" vragen kan beantwoorden, hij vraagt deze quasi letterlijk.

Op het mondeling moet je tweemaal met een dobbelsteen rollen, dit bepaalt dan welke hoofdstukken je moet bespreken, vaak moet je dan ook het verband tussen deze hoofdstukken kunnen uitleggen (bijv. het verband tussen Testing en QA).

Puntenverdeling

Schriftelijk examen op 40 punten (theorie 24, oefeningen 16) dat wordt herleid naar een cijfer op 20. Als je 12 of meer haalt dient het mondeling om een punt te bepalen tussen 10 en 20. Als je minder dan 12 haalt dient het mondeling om een punt te bepalen tussen 0 en 10. Let wel, als je op het schriftelijk minder dan 10 haalt je al heel erg je best zal moeten doen om nog een 10 uit de brand te kunnen slepen.

Examen

Schriftelijk

Het schriftelijk examen bestaat uit 12 detailvragen en enkele oefeningen. Voor het oplossen van de detailvragen is een zeer grondige kennis van de slides vereist (reken minstens 6 volledige dagen leren). De motivatie voor deze detailvragen is dat je in het latere leven ook een kant en klare definitie zal moeten klaarhebben voor veel voorkomende begrippen.

Prof. Demeyer plaatst zelf alle schriftelijke examens examenvragen (met oplossingen) van vorige jaren online. Op het schriftelijk examen is het gebruik van een rekentoestel toegestaan, en vaak ook nodig om oefeningen i.v.m. Project Management efficient op te kunnen lossen.

Mondeling

Op het mondeling examen worden uitsluitend inzichtsvragen en creatieve vragen gesteld. Bij de inzichtsvragen wordt naar het inzicht van de student in de materie gepeild. Bij creatieve vragen wordt je een probleem voorgeschoteld dat je in de praktijk kan tegenkomen. Het is jou taak om uit te leggen hoe je dat probleem zou aanpakken.

Vragenlijst

Introduction

You should

How does Software Engineering differ from programming?

> *Software Engineering is a state-of-the-art profession dedicated to designing, implementing and modifying software on time and within budget so that it is of higher quality, more affordable, maintainable and faster to build.*

Why is programming only a small part of the cost of a "real" software project?

> *To ensure quality, one must analyze requirements, design, test and maintain the software project besides implementation.*

Give a definition of "traceability".

- *Can we deduce which product components will be affected by changes?*
- *How to predict impact of changes?*
- *Maintain relationship:*
  - *from component to requirement that caused its presence*
  - *from requirement that must be changed when component is adapted*

What is the difference between analysis and design?

- *Analysis: Model and specify the requirements (“what”)*

- *Design:*
  - *Model and specify the solution (“how”)*
  - *System design (architecture) + detailed design (object design, formal spec).*

Why is the "waterfall" model unrealistic? Why is it still used? Unrealistic for large projects:

- *Complete: a customer cannot state all requirements explicitly*
- *Idealistic: in real projects iteration occurs (but tools and organisation obstruct)*
- *Time: A working version of the system is only available late in the project*
- *Change: it is very difficult and costly to adapt to changes in the requirements*

The waterfall model is popular for upper management, because it is visible: it is easy to control project progress

What's the relationship between iterative development, incremental development and (evolutionary) prototyping?

- *Iterative development: Controlled reworking of a system part to make improvements we get things wrong before we get them right.*
- *Incremental development: Make progress in small steps to get early tangible results always having a running vesion.*
- *An evolutionary prototype is intended to evolve in steps into a finished product*
  - *Grow, don't build: grow the system by redesigning and refactoring along the way*
  - *Combines incremental and iterative developments.*

How do you decide to stop in the spiral model?

After risk analysis.

How do you identify risk? What do you do when projecting a risk? Which risks do you assess as critical?

You can identify risks by via “risk item checklist”

- *For each risk factor, estimate the probability and the impact*
- *Prioritize the list*

Critical risks: high impact + moderate/high probability

What is a sprint in the SCRUM process?

- *2-4 week period*
- *team creates a working (= potentially shippable) product increment*
- *features in increment are chosen from product backlog*

Be able to draw UML diagrams!

Can you?

What is your preferred definition of Software Engineering? Why?

- *State of the art of developing quality software on time and within budget.*
- *The second is too focused on collaboration, the third on difference.*

Why do we choose Correctness & Traceability as evaluation criteria? Can you imagine some others?

- *Correctness: Build right product / product right.*
- *Traceability: Deduce what things will be affected by change.*
- *They ensure quality.*
- *Others: reusability, readability, cost, completeness, ...*

Why is Maintenance a strange word for what is done during the activity?

> *Maintenance: Change system after it's been deployed: adapt to changes; repair defects*

Why is risk analysis necessary during incremental development?

> *To make the go, no-go decision.*

How can you validate that an analysis model captures users' real needs?

> *Exploratory prototype: UI prototype -- validate user requirements.*

When does analysis stop and design start?

> *It Never actually stops. Design starts at the moment that enough requirements have been analyzed.*

When can implementation start?

> *Straight away: when known what to start with.*

Can you compare the Unified Process and the Spiral Model?

- *Unified process: time important, need to know number of iterations + phases + disciplines, global view*
- *Spiral Model: shows steps to take for each iteration, involves risks*

Can you explain the values behind the Agile Manifesto?

> *Experience found: Communication, response to change, interactions are more important than tools, plan, documentation, and contracts.*

Is it possible to apply Agile Principles with the Unified process?

> *?Yes, it seems possible to mix both up as the valued things seem to allow the various iterations.*

Did the UML succeed in becoming the Univerisal Modeling Language? Motivate your answer.

?Yes, uniform notation; complete life-cycle support; adaptable.

Project Management

You should

Why is it necessary to define tasks small?

- *For a reliable time estimation*
- *Allows traceability between plan & requirements/system*

What is a milestone? What can you use them for?

- *A milestone is a verifiable (by customer) goal that must be met after task completion*
- *Clear unambiguous milestones are a necessity*
- *Use: Monitor progress via milestones*

What is a critical path? Why is it important to know the critical path?

- *= path where delay in one task will cause a delay for the whole project*
- *ESD(n) + estimated time (n) = LED(n)*

- *The total time of the project depends on the critical path.*
- *Important to recover from delays (Adding senior staff for well-specified tasks outside critical path to avoid communication overhead)*
- *Analysis allows you to monitor the critical path and allocate the resources right for it.*

What can you do to recover from delays on the critical path?

A combination of following 3 actions:

- *Adding senior staff for well-specified tasks outside critical path to avoid communication overhead*
- *Prioritize requirements and deliver incrementally*
  - *deliver most important functionality on time*
  - *testing remains a priority (even if customer disagrees)*
- *Extend the deadline*

How can you use Gantt-charts to optimize the allocation of resources to a project?

The timeline allows you to easily draw the time of the programmers under the chart which allows you to easily move tasks together with their time.

Shuffle, split, extend tasks in tome. Distribute resources evenly.

What is a Known Known, an Unknown Known and an Unknown Unknown?

- *Known Known = You know the things you know, you can safely make assumptions during planning.*
- *Known Unknowns = You know the things you don't know, you can prepare for these during planning.*
- *Unknown Unknowns = You don't know the things you don't know, can't plan thus be aware and do risk analysis.*

How do you use PERT to calculate the risks of delays to a project?

Determine likely time, optimistic time and pessimistic time for each task.

- *Deduce estimated time for each task: ET(n) = [OT(n) + 4LT(n) + PT(n)] / 6*
- *Standard deviation: S(n) = [PT(n) – OT(n)] / 6*

for each path the degree of uncertainty (high deviation means likely to slip):

SP(n) = max { sqrt{ sum {s(m_i^2} } } m_i is node in path to n Why is it necessary to apply earned value analysis during project management?

- *View progress*
- *Monitor team occupation (with time sheets):*
  Compare the time spent versus the time planned (ask if a delay is expected)
- *Spot incoming delays to prepare for recovery*

Why does replacing a person imply a negative productivity?

- *The person leaving must take extra notes and has a drop in motivation which decreases productivity.*
- *When the new person takes over the productivity is negative as he bothers colleagues (communication overhead) and it will take some time to catch up.*

What's the difference between the 0/100; the 50/50 and the milestone technique for calculating the earned value?

- *0/100*:
  - *0% when task not completed*
  - *100% when task completed*
    - *gives a pessimistic impression*
    - *tasks should be rather small*
- *50/50*
  - *50% when task started*
  - *100% when task completed*
    - *gives an optimistic impression*
    - *tasks should be rather large*

milestone technique = Number of milestones completed / total number of milestones
    - *Tasks are large but contain lots of intermediate milestones.*
    - *Good for summary views on large schedules (otherwise split task in several subtasks and fail back on 0/100)*

Why shouldn't managers take on tasks in the critical path?

- *Because manager's manage and it's difficult to do those tasks at the same time.*
- *They have less time and should have spare time in case something comes up.*

How can you ensure traceability between the plan and the requirements/system?

- *Small tasks*
- *Milestones verifiable by the customer.*

Draw a PERT Chart, incl. calculating the critical path and the risk of delays

...

Draw a Gant chart, incl. allocating and optimizing of resources

...

Draw a slip line and a timeline

...

You can

Name the various activities covered by project management. Which one do you consider the most important? Why?

- *Planning, Organization, Staffing, Directing, Monitoring*
- *Planning makes the other activities more useful.*

Compare PERT-charts with Gantt charts for project planning and monitoring.

- *Gantt:*
    - *:) time management:*
        - *shows taks in timeliness*
        - *optimise resources by managing “slack time”*
        - *monitor critical tasks*
    - *:) resource & staff allocation*
        - *shows resource/staff occupation*
        - *optimize “free time”*
        - *moitor bottle necks*
    - *:(task interdependencies*

- *PERT:*
  - *:) task interdependencies*
    - *shows tasks with estimated timeliness*
    - *link tasks that depend on each other*
    - *optimize task parallelism*
    - *monitor complex dependencies*
  - *:) critical path analysis*
    - *calculate for each task: ESD, LED*
    - *optimize resource allocation to critical path*
    - *monitor critical path*
  - *:(time management*

How can you deal with Unknown Unknowns during project planning?

> *Thorough risk analysis to minimize the impact*

Choose between managing a project that is expected to deliver soon but with a large risk for delays, or managing a project with the same result delivered late but with almost no risk for delays. Can you argue your choice?

> *Manage a project with the same result delivered late but with almost no risks for delays because it is possible that the customer needs the project within a time and can't risk much delay.*

Describe how earned-value analysis can help you for project monitoring.

> *It allows you to compare time spent against the time planned, you can also ask people if delay is expected.*

Would you consider bending slip lines as a good sign or a bad sign? Why? *Bending slip lines indicate bad planning, it would be a bad sign for the manager.*

> *If it bends positive it is a good sign to the customer (early delivery), if not, it would be a bad sign to the customer.*

You're a project leader and one of your best team members announces that she is pregnant. You're going to your boss, asking for a replacement and for an extension of the project deadline. How would you argue the latter request?

- *She will slow down (motivation drop)*
- *Replacement will need to catch up: communication overhead + negative productivity*

You have to manage a project team of 5 persons for building a C++ compiler. Which team structure and member roles would you choose? Why?

> *Concencus: fast transfer of knowledge*

Use Cases

You should

Why should the requirements specification be understandable, precise and open?

- *Understandable: for all parties involved + actors provide an end user perspective*
- *Precise: so that all parties agree what's in and outside the system + to have sufficiently detailed scenarios to test and provide path coverage.*
- *Open: to give developers freedom to pick the optimal solution + as a prevention to over specify scenarios, then it can also allow evolving requirements.*

What's the relationship between a use case and a scenario?

> *A scenario is an instance of an use-case, with a typical example of execution it shows how objects interact to achieve the use case goal.*

Can you give 3 criteria to evaluate a system scope description? Why do you select these 3?

- *short (long is not convincing)*
- *end user commitment (formally approved)*
- *written down (to reference later)*
- *It describes the goals, includes criteria whether we have accomplished those goals and provides context so we know what to solve and what not.*

Why should there be at least one actor who benefits from an use case?

> *If nobody would benefit from an use case, the use case would be useless.*

Can you supply 3 questions that may help you identifying actors? And use cases?

- *Actors*
  - *Who uses the system?*
  - *Who provides information to the system?*
  - *What other systems use this?*
- *Use Cases*
  - *What function will the actor want from the system?*
  - *Who gets information from this system?*
  - *Are there any external events the system must know about?*

Which two basic rules apply to the project plan negotiation? Why?

- *Developers estimate cost; customers do not interfere. Thus schedule slips are the responsibility of the development team.*
- *Customers assign priorities; developers do not interfere. Thus customers decide where their money is spent.*

Which sections should be included into the requirements specification at the end of the inception phase?

The scope definition, the risk factors, the actors, the use cases and a project plan.

What's the difference between a primary scenario and a secondary scenario?

- *Primary: happy day scenario*
- *Secondary: what happens during special cases*

What's the direction of the <<extends>> and <<includes>> dependencies?

They indicate the dependencies, <<includes>> indicates something your object depends on, <<extends>> is something depending on your object.

You should be able to write a requirements specification with scope definition, actors, use casess and a project plan for your year project.

Can you?

Can you explain the difference between the two definitions for use cases?

- *A use case describes outwardly visible requirements of the system.*
- *A use-case is a generic description of an entire transaction executed to achieve a goal and involving several actors.*

The first one talks about the outside and the second one talks about the inside.

Why do use cases fit well in an iterative/incremental development process?

It helps for the Inception and Elaboration phase in the Unified Process.

Why do we distinguish between primary and secondary scenarios?

We add secondary "alternative" scenarios for when we have to do something different than the happy day scenario.

Assume that you work for a company that does not want to apply use cases in their requirements specification. Which principles would you still apply regardless of the use case notation?

System Scope, to at least have an idea of the system.

What would you think would be the main advantages and disadvantages of use cases?

- *Advantages:*
  - *helps validate solution again requirements*
  - *helps validate requirements against user needs*
  - *they form good milestones*
  - *good estimation of development effort.*
- *Disadvantage: Hard to make detailed designs + scenarios restrict evolving requirements*

How would you combine use-cases to calculate the risky path in a project plan?

> *The mile stones and estimation can be used for this, one would use the use cases as the tasks in the PERT chart.*

Do use-cases work well with agile methods? Explain why or why not.

> *Yes, Customers, working software, inviduals & interaction are important (change not so much)*

Domain Modeling

You should

Why is it necessary to validate and analyze the requirements?

- *We must validate the requirements to know if we are building the right system.*
- *We must analyze the requirements to know if we understand the problem correctly.*
- *The customer doesn't know what he wants and will change his mind.*

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

- *Functional: Decompose according to the functions a system must perform.*
- *Object-oriented: Decompose according to the objects a system must manipulate.*

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

- *Functional decomposition*
  - *:) for stable requirements and single functions*
  - *:) a simpler, clear strategy*
  - *:(naive: systems perform multiple functions*
  - *:(not maintanable: systems evolve)*
  - *:(not interoperable: difficult to interface with other systems*
- *Object-Oriented decomposition*
  - *complex and evolving systems*
  - *encapsulation provides robustness against typical changes*

How can you recognize "god classes"?

> *Lots of tiny classes, inheritance is geared towards data and code reuse, only a few large god classes doing the work.*

Why do you want to identify objects, responsibilities and collaborations?

- *To have a good object-oriented design where objects collaborate right and have the right responsibilites.*
- *collaborations lead to associations, responsibilities lead to operations, objects are candidate classes*

What is a responsibility? What is a collaboration?

- *Responsibility are the public services (knowledge, actions) an object may provide to other objects, not the implementation.*
- *Collaborations are with other objects to fulfill a responsibility, empty collaborations are possible.*

Can you supply three issues to take care of when chairing a brainstorm meeting?

> *keep small, heterogeneous, no censorship, round-robin, think fast, as many ideas as possible, all ideas are good.*

What do feature models define?

- *A set of reusable and configurable requirements for specifying the systems in a domain.*
- *Defines features & their dependencies*
- *Defines commonalities & variations between members of a product line*

How does domain modeling help to achieve correctness? Traceability?

- *Correctness:*
  - *System right: through good maintainability via a robust model of the problem domain.*
  - *Right system: model it from user perspective + role-playing helps vaildate use cases + feature diagrams focus on commonalities/variations*
- *Traceability: requirements and system: via proper naming conventions, especially the names of classes and operations.*

Apply noun and verb identification to a part of a requirements specification.

> *...*

Can you?

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

- *Role-playing scenarios help to validate the use cases*
- *Model the problem domain from the customer perspective thus analyzing the requirements.*
- *CRC cards are a requirements validation technique*

What's the problem with "god classes"?

- *Decrease the maintainability of the code*
- *Knows too much, does too much*

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

- *Because a class would then be doing too much, this again decreases the maintainability of the code.*
- *Would/could turn into "god" class*

What is the solution for the "two strings puzzle"? Why is it a good example of creative thinking?

> *Use the scissor for other purpose than cutting. Think out of the box!*

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

> *Create a list of scenarios, assign roles and their CRC Cards to different members, rehearse and revise the scenarios until they are clear. Role-playing makes everything clear, they are a requirement validation technique.*

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

> *Use Cases are a requirements specification technique, CRC cards are a requirements validation technique, they complement each other.*

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

> *It will be a good design, but it might not be the best... The purpose of CRC Cards in the first place is to smooth communication and rehearsing/revising the scenarios.*

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

> *Better communication, better for customer, easier, less official*

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

> *It would have a clear advantage with code reuse, but a disadvantage will occur when the family has to change. (Wired telephone --> Wireless telephone, ...)*

Testing

You should

What is (a) Testing, (b) a Testing Technique and (c) a Testing Strategy?

- *(a) Testing is the activity of executing a program with the intent of finding a defect.*
- *(b) A Testing Technique should have a high probability of finding an as yet undiscovered mistake.*
- *(c) A Testing Strategy is a plans that tells you when you should perform what Testing Technique.*

What is the difference between an error, a failure and a fault?

- *The error (fout) is the input that causes a failure.*
- *The failure (mislukking) is a deviation between the specification and the running system, an inability to perform the required function within limits.*
- *A fault (defect) is a design or coding mistake that may cause abnormal behaviour, thus deviations from the specification.*

What is a test case? A test stub? A test driver?

- *A test case is a set of inputs and expected results that exercise a component with the purpose of causing errors and failures.*
- *A test stub is a partial implementation of components on which the tested component depends.*

  Dummy code: input values and behaviour to run the test cases
- *A test driver is a partial implementation of a component that depends on a tested component.*

  Main function that runs test cases

What are the differences and similarities between basis path testing, condition testing and loop testing?

> *They test paths together, but each checks a different aspect of it. (Path, Condition and Loop)*

What are the differences and similarities between unit testing and regression testing?

- *Unit testing: will encourage you to design the interface right*
- *Regression testing: will ensure it still works after changes*

How do you know when you tested enough?

- *Never done*
- *Statistical testing: Test until the reduced failure rate is under a risk threshold.*

What is Alpha-testing and Beta-testing? When is it used?

- *Alpha testers are end-users that are close, they are invited and the tests are done in a controlled environement.*
- *Beta testers are selected customers, the testing is done in a "real world" setting without developers present.*

What is the difference betwwen stress-testing and performance testing?

> *Stress testing tests extreme conditions while performance tests run-time performance of a system.*

Complete test cases for the Loop Testing example.

> *...*

Rewrite the binary search so that basis path testing and loop testing becomes easier.

> *...*

Write a piece of code implementing a quicksort.

> *...*

Apply all testing techniques (basis path testing, conditional testing [3 variants], loop testing, equivalence partitioning) to derive appropriate test cases.

> *...*

Can you?

You're responsible for setting up a test program. To whom will you assign the responsability to write tests? Why?

> *Every developer should write their own tests (unit tests)* + test team (integrations tests)

Explain why basis path testing, condition testing and loop testing complement each other.

> *Because they tests various aspects of the path.*

When would you combine top-down testing with bottom-up testing? Why?

> *When integrating multiple modules to try to test as much as possible.*

When would you combine black-box testing with white-box testing? Why?

> *When you want to sufficiently test a module. White-box testing is less relevant in OO.*

Is it worthwhile to apply white-box testing in an OO context?

> *Sequence & collaboration diagrams serve a better way to apply white-box testing an an OO context, thus it is worthwhile enough.*

What makes regression testing important?

> *It's a huge help during maintenance, it helps ensure that all things still work when applying changes.*

Is it acceptable to deliver a system that is not 100% reliable? Why?

> *Completely reliable is almost impossible, thus it is more acceptable to deliver a system with a mean time between failures.*

Design by Contract

You should

What is the distinction between Testing and Design by Contract? Why are they complementary techniques?

- *Testing tries to diagnose defects while Design by Contract tries to prevent defects.*
- *They are complementary as you can't prevent all defects and you can't diagnose all defects.*

What's the weakest possible condition in logic terms? And the strongest?

- *Weakest: true*
- *Strongest: false*

If you have to implement an operation on a class, would you prefer weak or strong conditions for pre- and postcondition? And what about the class invariant?

> *A lazy developers wants a strong precondition, a weak postcondition and the invariant doesn't matter.*

If a subclass overrides an operation, what are you allowed to do with the pre- and postcondition? And what about the class invariant?

- *Postcondition: stronger or equal*
- *Precondition: weaker or equal*
- *Invariant: equal*

Compare Testing and Design by Contract using the criteria "Correctness" and "Traceability".

- *Both support traceability if you do it right*
- *Both do not guarantee correctness but sum is more than it's parts*

What's the Liskov substitution principle? Why is it important in OO development?

- *You may substitute an instance of a subclass for any of its superclasses*
- *If you subcontract, you must do the job, no less*

When is a pre-condition reasonable?

- *It must be possible to justify the need for the precondition in terms of the requirements specification only*
- *Clients should be able to satisfy and check it.*

What would be the pre- and post-conditions for the methods top and isEmpty in the Stack specification?

> *...*

How would I extend the contract if I added a method size to the Stack interface?

> *...*

Apply design by contract on a class Rectangle, with operations move() and resize().

> *...*

Can you?

Why are redundant checks not a good way to support Design by Contract?

- *extra complexity: extra code that needs to be verified aswell*
- *performance penalty: extra execution time*
- *wrong context*

You're a project manager for a weather forecasting system, where performance is a real issue. Set-up some guidelines concerning assertion monitoring and argue your choice.

> *Rule of thumb: at least monitor pre-conditions. Make sure that verifying pre-conditions is fast! Do not rely on switching off monitoring to gain efficiency. Profile performance to see where you loose efficiency.*

If you have to buy a class from an outsourcer in India, would you prefer a strong precondition over a weak one? And what about the postcondition?

> *A combination of both: Strong preconditions make a component more reusable, weak precondition will mean less effort to reach the state where you can call the method.*

Do you feel that design by contract yields software systems that are defect free? If you do, argue why. If you don't argue why it is still useful.

> *There will be less defects, but it won't be defect free. The could be put useless, forgotten or wrong ones*

How can you ensure the quality of the pre- and postconditions?

> *Make sure they are reasonable.*

Formal Specification

You should

What does it mean when we say that a formal specification is (a) consistent, (b) complete and (c) unambiguous?

- *(a) A formal specification is consistent when there are no contradictions in the specification.*

- *(b) A formal specification is complete when all properties are defined in terms of known concepts.*
- *(c) A formal specification is unambiguous when misinterpretations are impossible, it could also be minimal.*

What does it mean for a state-chart to be (a) consistent, (b) complete and (c) unambiguous?

- *(a) A state-chart is consistent when every state is reachable from the initial state and the final state is reachable from every other state.*
- *(b) A state-chart is complete when every event (state pair) has a transition.*
- *(c) A state-chart is unambiguous (deterministic) when the same event (incl. Guard) does not appear on more than one transition leaving a given state.*

What does it mean for an algebraic specification to be (a) consistent and (b) complete?

- *(a) An algebraic specification is consistent when it is never possible to deduce contradictions.*
- *(b) An algebraic specification is complete when all query expresions can be reduced.*

Why is an UML class diagram a semi-formal specification?

> *Because it is based on a notation with precise syntax but loose semantics.*

Can you give three arguments against formal methods? Can you counter them?

- *High cost of specification but better verfiability, safety reliability, security.*
- *Requires highly trained staff but only simple math and logic is required*
- *Customers don't understand but depends on customers and specification style, prototypes can be generated*

What's the distinction between a semi-formal and a formal specification?

> *The semantics are not loose.*

In the code of the binary-search routine. If we replace the < in line 10 with an >, can you prove what is wrong with the program?

> *...*

In the discussion of Consistent/Complete/Unambiguous, can you verify that the addition of the axiom implies inconsistency? And can you show that the removal of the axiom implies incompletenss?

> *...*

Extend the Z specification of the storage tank to include a "full" indicator, indicating when a Storage_Tank is close to capacity.

...

Provide the sequence charts for the primary and secondary scenarios of the "Place Order" use case in the chapter on use case. Derive the corresponding state-chart for the class "Order".

...

Given a Z or OCL specification, derive a test model using condition testing.

...

Given a statechart specification, derive a test model using path testing.

...

Can you?

Why is it likely that you will encounter formal specifications?

> *Software projects rely more on "Buy" than on "Build", thus things have to be specified formally.*

Explain the relationship between "Design By Contract" on the one hand and "Input/Output specifications", "Algebraic Specifications", "Logic-Based Specifications" and "State based specifications" on the other hand.

> *Natural pre- and post conditions can be used for formal specifications.*

Explain the relationship between "Testing" on the one hand and "Logic-Based Specificatons" and "State based specifications on the other hand.

> *Formal specification is black box testing but with complete coverage (with highest probability of finding mistakes)*

Why is it necessary to complement sequence diagrams with state charts?

> *Because sequence diagrams don't tell you anything about the states, it's handy to have them both for that reason.*

What does Cleanroom Development have to do with Formal Specificatons?

- *It relies explicitly on it*
- *Projects have shown: cost + time effective to eliminate errors in specification&design and to ensure implementation does not introduce defects*

You are supposed to build an e-commerce system for selling books over the world-wide web. You must guarantee 24 hours operations and secure transactions. You boss asks you to look into formal specs; which ones would you advise and why?

Anything concerning reliability and security. Z-notation

Software Architecture

You should

What's the role of a software architecture?

- *Maps requirements onto system structure*
- *A description of components and the connectors between them, specified in a view to show the relevant functional and non-functional properties.*

What is a component? And what's a connector?

- *A component is an encapsulated part of a software system with a designated interface, either a module/class/object/set/subsystem.*
- *A connector is a connection between two components, either static (in source code) or dynamic (temporary invocations).*

What is coupling? What is cohesion? What would a good design do with them?

- *Coupling is a measure of the strength of a connector. (How strongly it is connected)*
- *Cohesion is a measure of how well the parts of a component belong together.*
- *Both help us evaluate architecture tradeoff's, it's best to mizimize coupling and maximize cohesion.*

What is a pattern? Why is it useful for describing architecture?

- *A pattern is the essence of a solution to a recurring problem in a particular context.*
- *It's easy to recall and customize, it documents existing experiences, it states when to apply and lists tradeoff's.*
- *It allows experts to have deep discussions in few words.*

Can you name the components in a 3-tiered architecture? And what about the connectors?

> *The Database, Domain and Application components. The Database Connection and the Application Logic.*

Why is a repository better suited for a compiler than pipes and filters?

> *Compiler problem: needs flexible integration of partial solutions.*

What's the motivation to introduce an abstract factory?

> *Need for a class hierarchy with abstract roots representing a family of objects*
> *concrete leaves representing particular configurations*

Can you give two reasons not to introduce an Adapter (Wrapper)?

- *If seperately developed classes evolve much, adapter changes with*
- *overhead maintenance&performance*

Assume the ODBC example after applying the bridge pattern. Would it be a good idea for the PrintingOdbc to take advantage of special printing features provided by the Oracle database? Why?

> *...*

What problem does an abstract factory solve?

> *Invoking constructors implies tight coupling with concrete leaves instead of abstract roots.*

List three tradeoffs for the Adapter pattern.

- *How much adapting is required? For 1 class / whole hierarchy?*
- *How do seperately developed classes evolve? Evolve with?*
- *Merge in 1 or 2 directions?*
- *How much overhead in performance and maintenance can you afford?*

What's the distinction between a package diagram and a deployment diagram?

- *A package diagram decomposes a system into packages.*
- *A deployment diagram shows the physical lay-out of run-time components on hardware nodes.*

Define a sensitivity point and a tradeoff point from the ATAM terminology.

- *Sensitivity point: property of 1 or more components that is critical for achieving a quality attribute response*
- *Trade-off point: involves 2 or more conflicting sensitivity points*

Take each of the patterns and identify the components and connectors. Then assess the pattern in terms of coupling and cohesion. Compare this assessment with the tradeoffs.

Can you?

What do architects mean when they say "architecture maps function onto form"? And what would the inverse "map form into function" mean?

> *It means that you map requirements onto system structure, the inverse would be to map system structure back on your requirements.*

How does building architecture relate to software architecture? What's the impact on the corresponding production processes?

> *Architects are the technical interface between the customer and the contractor.*

A poor architectural design cannot be rescued by good construction technology. There are architectural styles or schools. Why are pipes and filters often applied in CGI-scripts?

> *To process data streams where flexibility (and parallelism) is required.*

Why do views and controllers always act in pairs?

- *View creates and initializes the controller + displays + corresponds to notification events*
- *Controller accepts user inputs + translates events into requests + responds to notification events'*

Explain the sentence "Restricts communication between subject and observer" in the Observer pattern.

> *Observers can't update the subject.*

Can you compare a bridge with an adapter as a way to build a layered architecture?

- *An adapter is for all implementations while a bridge is only for a series of variations using implementations.*
- *Bridge splits hierarchy, adapter merges 2 hierarchies*

Can you explain the difference between an architecture and a pattern?

- *Patterns define the essence of the solution, are "expert" knowledge and introduce complexity.*
- *Architecture intends to tackle complexity, implies tradeoffs and brings erosion.*

Explain the key steps of the ATAM method?

> *Analyze architecture + screnarios, check their risks, sensitivity points and tradeoffs, and reanalyze/reform.*

Risk analysis is done on various things, the results are distilled and worked upon before the next risk analysis. How would you organize an architecture assessment in your team?

> *...*

Quality Control

You should

Why is software quality more important than it was a decade ago?

> *Because software became ubiquitous, our society is vulnerable!*

Can a correctly functioning piece of software still have poor quality? Why?

Yes, even if it functions good. It can still have (non-)functional quality problems (like performance, maintainability).

If quality control can't guarantee results, why do we bother?

A quality process leads to a quality product.

What's the difference between an external and an internal quality attribute? And between a product and a process attribute?

- *External quality attributes are derived from the relation between environment and system/process.*
- *Internal quality attributes are derived immediately from the product or process description.*
- *The product is delivered to the customer while the process produces the software product.*

What's the distinction between correctness, reliability and robustness?

- *A system is correct if it behaves according to its specification.*
- *The probability that the system will operate as expected over a specified interval is the reliability.*
- *A system is robust if it behaves reasonably even in circumstances that were not specified.*

How can you express the "user friendliness" of a system?

The degree to which the human users find the system (process) easy to use.

Can you name three distinct refinements of "maintainability"? What do each of these names mean?

- *Repairability, how much work is needed to correct a defect.*
- *Adaptability, how much work is needed to adapt to changing requirements.*
- *Portabilitiy, how much work is needed to port to a new environment or platforms.*

What is meant with "short time to market"? Can you name three related quality attributes and provide definitions for each of them?

- *The ability to deliver the product on time, this is important for marketing.*
- *Any quality attributes are at risk when trying to be fast.*

Productivity, Timeliness, Visibility

Name four things which should be recorded in the review minutes?

What was reviewed, who reviewed it, what where the findings/conclusions, decision

What's the relationship between ISO 9001, CMM standards and an organization's quality system? How do you get certified?

> *The quality standards influence the quality system of the organization. You must ask such a quality organization to get certified, they will then audit the quality system.*

Can you name and define the five levels of CMM?

- *Level 1: Initial, Ad Hoc, no effective QA procedures, quality is luck*
- *Level 2: Repeatable, formal QA procedures in place*
- *Level 3: Defined, QA process is defined and institutionalized*
- *Level 4: Managed, QA process and quantitative data collection*
- *Level 5: Optimizing, improvement is fed back into QA process.*

Can you define a Key Process Area (KPA)?

> *It identifies a cluster of related activities that, when performed collectively, achieve a set of goals considered important.*

Given a piece of code and a coding standard, review the code to verify whether the standard has been adhered to.

Can you?

Given the Quality Attributes Overview table, argue why the crosses and blanks occur at the given positions.

> *...*

Why do quality standards focus on process and internal attributes instead of the desired external product attributes?

> *Because a quality process leads to a quality product.*

Why do you need a quality plan? Which topics should be covered in such a plan?

> *A quality plan is needed to deliver the project on time and within budget.*
>
> - *set out desired product qualities and how these are assessed*
> - *set out which organizational standards should be applied*
> - *define* the quality assessment process

How should you organize and run a review meeting?

> *3-5 people together maximum of 2 hours advance preparation, checklists can be used to evaluate products, meeting must last less than 2 hours and decide whether the product is accepted, provisionally accepted or rejected.*

Why are coding standards important?

> *Coding standards increase readability thus help towards the quality.*

What would you include in a documentation review checklist?

> *project plan, requirements specification, design, code and testing*

How often should reviews be scheduled?

> *There should at least be a review every week.*

Could you create a review check-list for ATAM?

> *Create questions for the topics covered by ATAM.*

Would you trust software from an ISO 9000 certified company? And if it were CMM?

> *No, it should at least be ISO 9001. For CMM it depends on the level, it should at least be Defined (level 3).*

You are supposed to develop a quality system for your organization. What would you include?

> *...*

Software Metrics

You should

Can you give three possible problems of metrics usage in software engineering? How does the metrics theory address them?

> *Preciseness, Representation Condition, Scales and the GQM-paradigm. The metrics theory helps to answer the questions by providing good techniques.*

What's the distinction between a measure and a metric?

- *A measurement is a function mapping of an attribute of a real world entity onto a symbol in a set of known mathematical relations*
- *A metric is a measurement with as range the real numbers and which satisfies $m(x,x)=0$, $m(x,y)=m(y,x)$ and $m(x,z)<m(x,y)+m(y,z)$*

Can you give an example of a direct and an indirect measure?

- *Direct: lines of code*
- *Indirect: module defect density*

What kind of measurement scale would you need to say "A specification error is worse than a design error"? And what if we want to say "A specification error is twice as bad as a design error?"

> *For the first ordinal, for the second ratio*

Explain the need for a calibration factor in SLIM.

- *The calibration factor is needed because projects and teams have different sizes.*
- *To know the parameters that influence the effort*

Fill in the blanks in the following sentence. Explain briefly, based on the SLIM model. "If you want to finish earlier (= decrease scheduled time), you should ... the effort ... ."

> *If you want to finish earlier, you should increase the effort to increase the process productivity.*

Give three metrics for measuring size of a software product.

> *Lines of Code, Function Points, Use Case Points.*

Discuss the main advantages and disadvantages of Function Points.

- *:) measured after the design, independent of implementation language, measures functionality and works well for data-processing.*
- *:(It requires subjective expert judgement and cannot be calculated automatically.*

What does it mean for a coupling metric not to satisfy the representation condition?

> *It means that coupling might not be representable enough to be a sufficient metric.*

Can you give three examples of impreciseness in Lines of Code measurements?

- *Ignores software reuse, code duplication, benefits of redesign.*
- *The lower level the language, the more productive the programmer.*
- *The more verbose the programmer, the higher the productivity.*

What's the difference between "Mean time to failure" and "Average time between failures"? Why is the difference important?

> *The average time between failures is used to calibrate the probability density function of the mean time to failure.*

Given a set of use cases calculate the use case points.

> *...*

Can you?

During which phases in a software project would you use metrics?

> process, system, documentation, *effort estimation*, *quality assurance*

Why is it so important to have "good" product size metrics?

To calibrate the models right (to get a good estimate), these are needed for algorithmic cost modeling.

Can you explain the two levels of calibration in COCOMO (i.e. C & S vs. PM)? How can you derive actual values for these parameters?

C is the complexity factor, S is an exponent and PM the product size metric. You can reach actual values through historical analysis for these parameters to obtain the Effort.

Can you motivate why in software engineering, productivity depends on the scheduled time? Do you have an explanation for it?

- *To finish earlier, you must invest MANY more man months.*
- *To decrease the cost, you must spend A LOT more time.*
- *More productive with less time, because of the pressure but less maintainable, ... code*

Can you explain the cone of uncertainty? And why is it so relevant to cost estimation in software projects?

At the beginning little is known

How can you decrease the uncertainty of a project bid using the SLIM model?

- *Give more time*
- *SLIM is based on 7.200 projects and will thus give a more certain project bid if calibrated right.*

Why do we prefer measuring Internal Product Attributes instead of External Product Attributes during Quality Control? What is the main disadvantage of doing that?

- *external: doesn't require finished product/process, hard to achieve preciseness with external*
- *internal*
  - *:(not reliable, heavy weight approach, difficult to interpret*
  - *:) cheap, good focus*

You are a project manager and you want to convince your project team to apply algorithmic cost modeling. How would you explain the technique?

Choose system model, callibrate, measure attributes, calculate effort

Where would you fit coupling/cohesion metrics in a hierarchical quality model like ISO 9126?

Maintainability --> Simplicity/Modularity

Why are coupling/cohesion metrics important? Why then are they so rarely used?

Because of the disagreement whether it satisfies the representation condition.

Do you believe that "defect density" says something about the correctness of a program? Motivate your answer?

Higher defect density leads to worse correctness.

Refactoring

You should

Can you explain how refactoring differs from plain coding?

Changes the software system in such a way that it does not alter the external behaviour of the code, yet improves its internal structure.

Can you tell the difference between Corrective, Adaptive and Perfective maintenance? And how about preventive maintenance?

Corrective fixing errors, Adaptive to new environments and Perfective with new functionality. Preventive maintenance by refactoring, it is a new category.

Can you name the three phases of the iterative development life-cycle? Which of the three does refactoring support the best? Why do you say so?

Prototying, expansion, consolidation. Consolidation supports this the best, as refactoring during expansion slows down the productivity.

Can you give four symptons for code that can be "cured" via refactoring?

Duplicate code, nested conditionals, large classes/methods and abusive inheritance.

Which four activities should be supported by tools when refactoring?

- *rapid edit-compile-run cycles*
- *reverse engineering facilities*
- *regression testing*
- *version & configuration management.*

Why can't we apply a "push up" to a method "x()" which accesses an attribute in the class the method is defined upon?

Because we need to "push up" the attribute too.

Two classes A & B have a common parent class X. Class A defines a method a() and class B a method b() and there is a large portion of duplicated code between the two methods. Give a sequence of refactorings that moves the duplicated code in a separate method x() defined on the common superclass X.

...

What would you do in the above situation if the duplicated code in the methods a() and b() are the same except for the name and type in a third object which they delegate responsibilities to?

...

Can you?

Why would you use refactoring in combination with Design by Contract and Regression Testing?

> *To maintain software. DbC and Regression Testing make sure no existing functionality causes defects/changes when refactoring.*

Can you give an example of a sequence of refactorings that would improve a piece of code with deeply nested conditionals?

> *Extracting methods, if the conditionals are types one could even use polymorphism.*

How would you refactor a large method? And a large class?

- *Use "Extract Method" to split the large method in multiple smaller methods.*
- *If a class contains a lot of methods then groups of them should be moved to their own class.*

Consider an inheritance relationship between a superclass "Square" and a subclass "Rectangle". How would you refactor these classes to end up with a true "is-a" relationship? Can you generalise this procedure to any abusive inheritance relationship?

> *Make Square a subclass of Rectangle and move the procedures accordingly.*

Conclusion

You should

Name three items from the code of ethics and provide a one-line explanation.

- *Product: Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.*
- *Colleagues: Software engineers shall be fair to and supportive of their colleagues.*
- *Self: Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.*

If you are an independent consultant, how can you ensure that you will not have to act against the code of ethics?

Include in contract a clause to cancel when acting against code of ethics

"Killer Robot"

Which regression tests would you have written to prevent the "killer robot"?

Tests that prevent the killer robot from killing someone should have been ran.

Was code reviewing applied as part of the QA process? Why (not)?

Yes, code reviews were applied during the QA process. Samuels decided not to attempt those code reviews as he was mad.

Why was waterfall process disastrous in this particular case?

*Because there wasn't room for iterative improvements or communcation and thus the flaws couldn't be fixed in time.**

Why was the user-interface design flawed?

Because it violated every Shneiderman's rule, not to mention the console and the comfort for the operator.

The interface was also to slow for an operator to terminate the erratic robot behavior in a fast enough way.

Can you?

You are an experienced designer and you heard that the sales people earn more money than you do. You want to ask your boss for a salary-increase; how would you argue your case?

One could argue that he use your good design to sell it better

Software products are usually released with a disclaimer like "Company X is not responsible for errors resulting from the use of this program". Does this mean that you shouldn't test your software? Motivate your answer.

One should still test his software to prevent defects and to help refactoring for maintenance reasons.

"Killer Robot"

Recount the story of the Killer Robot case. List the three most important causes for the failure and argue why you think these are the most important.

Bad interface design (because of the death), the waterfall model (because of the approach) and faked tests (because of prevention).

Categorieën:

- Informatica

- 3BINF