SMA: Software Modeling and Architecture

Q&A Session
Categories

6 questions in “General”
10 questions in “Smalltalk”
4 questions in “Reflective Reasoning”
1 question in “Architecture & Design”
3 questions in “Visualization”
3 questions in “Organizational Affairs”

In total 27 questions in 6 categories
Q: What is the difference between type soundness and type safety?

A: Type safety is a property. Soundness describes the quality of the type safety. An unsound type system can accept code that may lead to run-time errors. Example: co-variant arrays in Java.

https://dzone.com/articles/covariance-and-contravariance
Q: Why is symbolic execution considered to be a static (and not dynamic) analysis technique?

A: Symbolic execution is not an execution of code, but a theoretical analysis. Think of it rather as an evaluation of all possible paths (path explosion!) and reasoning about their constraints.

The result of a symbolic execution could be something like “the return value of method getNumber() is a float number between 3 and 4”.
Q: Why is it crucial that model-checkers work with finite state models of software?

A: Model-checking is very complex on its own, thus it is common to use a syntactically simplified (language independent) model which is still logically equivalent to the code.

In other words, code will be transformed into a finite-state diagram/model which is easier to process (no language specific extensions required).

Non-deterministic diagrams/models are less frequently used, since they do not provide that many benefits in most cases (software is mostly deterministic), and finite models are easier to handle (no randomness in the graph traversal, due to the missing Epsilon transitions).
Q: Which machine learning algorithms are mostly used in software data analytics?

A: There is no one fits it all algorithm.

It really depends on the use case. More specifically, the selection depends on many factors, however, the complexity (dimensionality) of the problem is an important one. In general, polynomial algorithms (Stochastic Gradient Descent, SVM, ...) are used for “rather simple” pattern matching, and neural networks are rather used for more complex problems requiring a form of contextual knowledge.
Q: What is the best metric for measuring the complexity of any code? Why is it hard to calculate CYCLO for SMALLTALK?

A:

a) Again, there is no one fits it all algorithm; there exist several different complexity metrics. I assume that most users just choose the built-in algorithm of the analysis framework they use.

b) Cyclo is not that hard to calculate in Smalltalk, in fact, you even used that algorithm for one exercise. Nevertheless, it uses more CPU than lines of code or class cohesion, since the execution paths have to be calculated. The problem arises more on the Pharo platform itself: because it uses only one native CPU the UI might freeze during calculation.
Assignment 12 – E03 – Q06

Q: Could you explain the cyclomatic complexity. We do not understand why the following graph (from wikipedia) results in a cyclomatic complexity of three even though that in our opinion there are four ways through the code: ac ad bc bd.

A: It basically counts the branches. So you have one path through the whole graph and then you have the two branches (each +1). That’s why you end up with three.

McCabe showed that the cyclomatic complexity of any structured program with only one entry point and one exit point is equal to the number of decision points (i.e., “if” statements or conditional loops) contained in that program plus one. However, this is true only for decision points counted at the lowest, machine-level instructions.

\[ M = E - N + 2P \]

3 = 9 - 8 + 2 * 1
Smalltalk
Q: What is the class ProtoObject good for?

A: It provides very low-level APIs, such as “memory scanning”, introspection, debugging, thus some features are rather used by Pharo platform developers.

ProtoObject encapsulates the minimal set of messages that all objects must have.

Generally these are proxy objects designed to read themselves in from the disk, or to perform some wrapper behavior, before responding to a message. ProtoObject has no instance variables, nor should any be added.
Q: Why are Behavior and Class different classes?

A: They are split due to single responsibility principle: results in cleaner code.

Behavior is managing the object life-cycle and configuration (init, cleanup, superclass link, method dictionary), while Class provides most importantly a class name and instance variables.
A: There exist only anonymous classes in the class hierarchy (but not the Object hierarchy). The anonymous classes are always of type Metaclass.

In other words: The instances of class Metaclass are the anonymous metaclasses, each of which has exactly one instance, which is a class.
Q: Why are there no explicit metaclasses?

A: There exist explicit metaclasses: The explicit metaclasses are the class classes, e.g., Object class, Collection class, ....
Q: (L02) What happens to existing instances of a class if you add new fields at run time? (slide 53) Is it correct that existing instances are not changed. Does it depend whether you add instance variables or class variables or instance class variables? [→ should be: class instance variables]

A: You can add to an object new methods and instance variables at run time. However, you cannot add class variables to a object at run time. For that, you would have to inspect the meta class of that object and implement the changes there. BUT: Since the system (Pharo) does not change existing instances, you would have to manually traverse those and add the changes to each object one by one (with the risk of introducing conflicts when variables already have been set, etc.).
Q: (L03) Slide 64: Question: Why don’t metaclasses inherit from Class?
But on slide 51 + 52 it states: Every metaclass inherits from Class and Behavior: Since rule 2 tells us that all classes eventually inherit from Object, we should infer that the same holds for metaclasses, which are also classes. So the metaclass hierarchy does not stop with Object class, but rather with Object. In between, however, we have the special system classes Class, ClassDescription and Behavior. Is the correct answer therefore, they do inherit from class and the question is just a trap :) or should „metaclass“ in the question actually be „Metaclass“ to refer to the class of metaclass, whose superclass is ClassDescription rather than Class.

A: You found a bug in the slides; should be fixed by now.
Of course, the metaclasses inherit from Class as shown in the graph.
Q: (L03) Is it correct that ProtoObject is the only class in Pharo that does not inherit from Object (since its superclass is nil)? (slide 64)

A: Yes!
Q: (L03) The class of a metaclass is Metaclass. The class of Metaclass is Metaclass class. Therefore, the metaclass of metaclass = metaclass class class = Metaclass class. Metaclass class is therefore, also a metaclass. According to rule 5, every metaclass inherits from Class and Behavior. (slide 60) However, according to slide 61, Metaclass class only inherits from ClassDescription i.e. from Behavior and not from Class. Why?

A: My guess: Since instances of Metaclass class are anonymous classes (in contrast to regular classes whose instances are not anonymous) they couldn’t just extend Class since the implementation wouldn’t match. So they chose ClassDescription class instead and augmented the missing parts manually.
Q: (L06) Slide 35: Why is it hard to calculate cyclo for Smalltalk methods? There are also branches in smalltalk programs e.g. (condition) ifTrue: […] ifFalse: […].

A: Cyclo is not that hard to calculate in Smalltalk, in fact, you even used that algorithm for one exercise. Nevertheless, it uses more CPU than lines of code or class cohesion, since the execution paths have to be calculated. The problem arises more on the Pharo platform itself: because it uses only one native CPU the UI might freeze during calculation.
Q: (exam 2016) What is the difference between an instance variable, a class variable and an instance variable on the class side in Smalltalk?

I am still a bit confused with this whole class-side and instance-side story. Let me try to elaborate my current assumptions: instance-side refers to the class i.e. to the methods and variables defined by the class including instance variables and instance methods. Its major purpose is to create new instances of the class. class-side refers to the meta-class i.e. to the methods and variables defined by the metaclass. Since a class is also an instance, it can also be created from a metaclass using the class-side methods. Class-side methods will return an instance of the class and are also used for utility functions, since they don’t require an object to be created first. instance variables belong to the instance-side. Class variables, also belong to the instance-side and class instance variables belong to the class-side. instance variables are defined by the class and initialized by the instance since the class has no access to them (encapsulation boundary is instance itself). Class instance variables are basically instance variables, but are defined by the metaclass. Class variables are defined by the class itself and shared among all instances.
Assignment 12 – E03 – Q16

A: Instance variables and instance variables on the class side refer to the same.

Class variables:
A class variable is accessible by a class, all its subclasses, and all the instances of the corresponding classes. Any method, either a class method or an instance method, can access a class variable by using the name of the class variable.

Class instance variables:
Class instance variables are variables that belong to a specific class instance. Smalltalk itself relies heavily on this feature. For example, each class stores its name in a class instance variable. Just as each instance has its own values for the instance variables, each class has its own values for class instance variables. Unlike class variables, these variables are not shared by all the instances of a class. Only class methods can reference class instance variables (similar to the static keyword in Java). Direct references to these variables are not allowed from instance methods. Instances methods that need the information stored in a class instance variable must send a message to a class method, which then returns the requested information.

In short: class instance variables are referenced by class side methods and instance side variables are referenced by instance side methods only while class variables are shared by both side (class side and instance side).
Reflective Reasoning
Q: Is debugging of a program is introspection? Please explain.

A: No, it is not introspection, but it uses introspection features.
(Introspection is the ability for a program to observe and therefore reason about its own state.)
Q: Is reification needed only for intercession? What is the relation of reification with introspection?

A: 1) Yes. When you use intercession the *whole* execution state has to stored as data to make it accessible. However, this is not the case for introspection.

2) Introspection only needs the *partial* execution state to be accessible.
Q: I am trying to come up with some code examples to understand Reification better. (Also, we have some examples regarding Introspection and Reflection). My question is that can we consider the usage of “Generic Types” in Java as Reification? If not, can you give me another example.

A: No. Java doesn’t support reification at full. For example, you can change field values through reflection methods, but not the class hierarchy (which is considered to be part of the class state).

In Smalltalk you can literally change everything: you could even manually hook up the inheritance chain during run time.
Q: Anonymous classes and Method wrapping - Intercession

A: 1) The only anonymous classes are the instances of metaclass.
2) see slide (especially the note)
3) Intercession is the ability for a program to modify its own execution state or alter its own interpretation or meaning.

MethodWrapper before and after methods

A MethodWrapper replaces an original CompiledMethod in the method dictionary of a class and wraps it by performing some before and after actions.
Q: Question from an old exam: What is the difference between the architecture and the design of a software system? Robert Martin states that there is no difference between software architecture and software design in his book “Clean Architecture”. Thus I am a bit confused. What would the expected answer be? Is the difference only that architecture is usually at a higher level of abstraction than the design?

A: I would say that software design is the process to obtain an software architecture. On the one hand, a bad design leads consequently to a bad architecture, but on the other hand, a nice design does not always lead to a nice architecture.
Visualization
Q: What is information visualization?

A: Information visualization or information visualisation is the study of (interactive) visual representations of abstract data to reinforce human cognition.

In other words, it’s the research of representing data in a more convenient way.
Q: Which techniques allow to visualize software structure?

A: Almost all as the selection of techniques depends on the available data. Since you can select various properties regarding software structure, many options are reasonable. In general, tree structures or graphs have proven quite reliable.
Q: On what visualization features can we map evolution?

A: Line-based plots are preferred, since they clearly emphasize on changes of values over time. That is, because evolution includes different states over time, one way or another.
Organizational Affairs
Assignment 12 – E03 – Q25

Q: Do we need equally to focus on lectures given by professor and guest's lectures? Or we should focus more on the lectures given by the professor? I am asking because some of the guest's lectures have a lot of examples (code & graphs) and not so much of theory. To sum up, should we expect questions regarding guest's topics?

A: Yes, the exam will include the content of guest lectures (except the last guest lecture).

Long answer: However, it's not meant that you learn the presented paper titles and their (paper) content by heart. We rather ask question about general understanding, i.e., rather the high-level concepts. So don't expect any very specific questions like "Which algorithms can you choose in WEKA?", or "What was the test / training dataset ratio used in the BugPrediction experiment?" Nevertheless, questions like "What would be a valid dataset ratio for test / training of an ML model? Justify!", and "List two code-quality related source code metrics." would be perfectly reasonable.
Assignment 12 – E03 – Q26

Q: I have just realized that 5th slide (Moldable Software Exploration) does not exist in the repository. I wonder if you can upload it (or should I assume it is not in the exam?).

A: There exists no such slide deck.

It was Tudor Gîrba's lecture, and he didn't use any slides. He rather told us a story about the potential of future IDEs, followed by a extensive live demonstration of the GToolkit built on top of Pharo. For this lecture we only have an exercise sheet which supports some questions, but I don't think that any code-level GToolkit questions would be appropriate, since the API is currently under heavy change.
Assignment 12 – E03 – Q27

Q: How much of the exam will be about the guest lectures?

A:
Exam has a total of 90 points. 1 point = 1 minute of expected work.

No details, but guest lectures are important:
Expect around 50% guest lecture questions.
... still more questions?
Next Time: Exam
90 minutes / CLOSED BOOK

19-Dec-2018, 10:15am until 11:45am, Room 001
(same building as lecture, but adjoining room)

1. Arrive on time

2. Don't forget your student ID

3. Don't forget a blue or black ball pen

4. Not allowed: pencils, internet, notebook, books, any printouts, additional blank pages, pocket calculators, mobiles, smart anything, ...
Thank you VERY MUCH for your participation!
Good Luck!