# Viva Questions

This document contains a lists of questions I came up with by myself during the viva preparation process, starting mid December. The questions are ordered by their type e.g. whether they are more technical, or related to the overall contribution or deal with consistency / narrative of the thesis, etc.

Note that there are NO written answers to these questions because this list of questions act as a practice guide, where the intention is that the questions are given by freely speaking without lot of formal preparation – after all I do not really know what's coming during the viva.

An important inspiration was the book “How to survive your viva” by Rowena Murray. Another source of inspiration for many questions was a highly critical reading of my own thesis, which proved to be highly valuable. In addition, I incorporated the summary of “How to survive your viva” done by my Supervisor.

An important part of the viva preparation process is to make concious your subconcious thought processes by thinking through my thesis and answering mock questions.

The viva: its purpose

* did you do the work yourself
* have you done the reading
* do you have a good knowledge of the field
* did you write the thesis yourself
* can you do research independently
* can you teach your subject
* can you talk about it professionally
* did you receive any training
* have you contributed to knowledge
* did you learn anything

PRACTICE REPEATEDLY:

* condense my thesis in a 3 Minute answer. focus on narrative. Key: rationale, aims and research questions, Method, approach, outcomes, contributions, implications
* technical chapter explanations of Monads, Transformers, Arrows and MSFs so I don’t forget

# General

OK How you come to do research in this area?

OK Could you summarise the main points of your thesis for us?

OK Who would you say are the key people in your field today?

OK Did your study turned out as you expected?

OK How do you see research developing in the next five years?

OK On a general level, what were the most interesting things to come out of your thesis?

OK What compromises where you forced to make (in your research design) and why (e.g. time). OK How should the research in this area progress? over the next years i want to address the rather ad-hoc Nature of my PhD, building underlying theories, from dynamical Systems, coaltebras, category theory and Dependent Types

list the assumptions on which my thesis Rests  
- state is the root of all evil in non-trivial software engineering  
- OO as in Java, C++, C#, is inherently built on state, which makes it difficult to arrive at correct solutions / requires huge test suites.

OK identify assertions and speculations i made in my thesis

TODO Re-read my Thesis:

* Read each chapter from start to finish
* Write one sentence about it. This will act as a prompt for your thinking and recall
* Summarize every chapter and section
* Put single-word prompts in the margin. This will help you find your way if you lose your place
* Convert the chapter into a series of questions and answers
* Use these in your practise sessions

# Literature Review

OK Why is this research important?

OK Who else thinks it is important?

OK Who has worked in this subject before?

OK What had not been done before?

OK Who has done something similar to what you did?

OK What is your contribution to the field?

OK Who will use your material?

# Method

OK Why did you reject other methods?

OK What are the pros and cons of the methods you chose? What are the limitations in the design?

OK Has the best design been chosen?

OK What are the theoretical components of your framework?

OK What led you to select these models?

OK Why STM and quickcheck in Haskell? they also exist in other languages!  
answer: STM really shines in Haskell because of restricted side effects. quickcheck was invented in Haskell and data-driven programming better suited. also if it would not been for Haskell, i would never had even considered / known / experimented with STM and quickcheck in oop

OK why not oop vs fp? because its not constructive and boring. oop already proved its usefulness for Abs. my thesis is doing it in a constructive way for FP

# Contribution

1. OK Explain what is new / original about your work
2. OK Tell me how your work differs from that of X
3. OK What would you say has been your contribution?
4. OK Who would be most likely to agree/disagree with you?
5. OK How long do you expect your work to remain innovative?
6. OK Do you think your research will influence others?
7. - Invited talk at Sandtable in London. PhD seminar in China. Exchange with Ian Buckley from Canada, Risk Specialist, (<http://csto-btcvm.ca/home.aspx>)

OK What is the original / unique research contribution (to knowledge) of your Ph.D. / thesis?

- Creating an awareness for STM in parallel ABS: my realisation that STM might be a great way to implement the shared environment in ABS and exploit concurrency

- Introducing Property-Based Testing to code testing of ABS implementations: my realisation that randomised property-based testing is a much better / more natural fit to testing ABS implementations than unit testing

- Showed how we can do ABS with FP with focus on purity / referential transparency: systematic development of concepts of how to do it

OK What do you think is the most valuable contribution of your Ph.D. / thesis?

- Introducing Property-Based Testing to code testing of ABS implementations: my realisation that randomised property-based testing is a much better / more natural fit to testing ABS implementations than unit testing

# Coherency and Narrative

You say in the introduction that the central theme of your thesis is purity but in Chapter 8 “Concurrent ABS” you ‘sacrifice’ purity for the sake of parallel ABS by utilising STM. Why did you sacrifice purity? Why do you make purity the central theme but then violate it?

# Technical

What exactly are Monads?

- an algebraic structure, which allows to implement a number of effectful programming in a pure functional programming language.

Please define a Monad.

Give me examples for Mondas?

- Reader: partial function application, Writer: monoid, State: closures, Lists: non-deterministic programming

What are Monad Transformers and what to they solve?

- they solve the stacking of effects / composing of monads

How are effects evaluated in Monad Transformers? What about commutativity?

Why do Monads NOT compose?

- bind operator cannot be polymorphic in both, we need to know one monad because each monad works different, so it is different how to apply subsequent monad

Please define a Maybe Transformer.

What exactly are Arrows?

- an arrow is potentially effectful computation, which is a generalisation of a monad which in addition to be parameterised over its output, is also parameterised over its input, therefore emphasiszing programming with tuples a lot.

Please define an Arrow.

What are MSFs?

- monadic stream function: is a generalisation of Signal Functions, which are Arrows, emphasising a process – oriented approach to computing in FP with the addition of monadic effects to run within such a process, which was previously not possible with Signal Functions.

Please define an MSF.

What are shared-nothing semantics (in messaging)?

# Weaknesses of your thesis

comparison with imperarive approaches: how do i really prove that they are insufficient for verification and validation? Performance comparison is difficult but thats in the end the only Thing that really matters in Simulation. couldnt we have achieved that all not also with oop using some STM and property-based testing library, only with better performance?

Methodology! i never received any useful training on that Part. its more a piecing together of loose bits and pieces into a coherent narrative.

i started out with simple curiosity for the Problem without clear hypotheses / Research questions. This means, that there was never a real-world problem my PhD wanted to solve. I had to “find” real-world problems to which my research seems applicable and a potential solution, otherwise I couldn’t “sell” my results / research / approach.

chapter 5 no publication, how do i know its ok?  
I was thinking about publishing the work of chapter 5 but I thought I’d focus on more interesting stuff e.g. STM and QuickCheck. The problem is that this is probably technically the most advanced chapter in the whole thesis, therefore the only viable audience would have been FP community but the review to Haskell Symposium made it clear that the gatekeepers have not patience for rather applied stuff as I do. I could have developed something in the direction of pure functional objects, following the final tagless approach as I did in section 5.2.5.1. but then I would have needed to look into a lot of theory of coalgebras, codata and so on because with a simple “ad-hoc” approach without some super fancy theory, I would have wasted my time trying to publish at an FP journal / conference.

Chapter 7: should have used criterion as well but then there would not be this differentiation between 3 different output types

the thesis seems to invent a Problem where there was never really one: oop seems to be doing fine for implementing ABS. although the Thesis provides reports of failure, it is doubtful that oo is the sole reason for failure. also the selection of the beneficial topics of parallel and testing abs seens to be rather ad hoc. we hypothesise that this is due to the fact that the researcher simply let curiosity drive his research. however, the actual work is original, highly interesting and very well done with three novel contributions, unique to a pure Functional approach / very unlikely to have emerged in a oop abs research: guaranteed reproducibility at compile time, STM for concurrency and quickcheck for testing Abs implementations

# Who are the major players in my field(s)?

Ivan Perez

Cezar Ionescu

Bezirgianis

Aivika 3

RePast Guys (Ozik, Collier)

* What do you know about the work of X?