Exam topics for Concurrent Programming Languages.

Oleg Batrashev

June 4, 2010

These are the topics for the Concurrent Programming Languages course exam. The topics are covered during lectures and most are in the CTM book, except Erlang fault tolerance model and concurrency in other languages.

- For every topic there are several keywords or ideas that I relate to the topic. The more you know about the topic the better. Your experience and thoughts are welcome.
- I'll divide 18 questions into 6 tickets with 3 questions each. Each question is worth 15 points, 50 max in total for one ticket.
- Exam type is oral. You have time to prepare, where you may write down notes or pictures to show me during the examination.
- If you omit something I consider important I'll try to give hints.
- Short examples in any applicable language (even pseudocode) are welcome but no syntactic validity is necessary for the code, just to make sure you grasp the ideas.

Introduction

1. Definition(s) of *parallel* and *concurrent*, where and how parallelism is used.
2. Classifications of concurrency.

Declarative concurrency

3. Declarative programming: declarativeness, iterative and recursive computations, accumulators.
4. Purely functional (immutable) data structures: example structures, behavior with multiple threads, pros and cons.
5. Declarative concurrency: presence or lack of determinisms, presence of race conditions, exceptions.
6. Streams: demand-driven concurrency, problem with several readers/writers, problem with joining 2 streams.

Message-passing concurrency

7. Port concept, why we need it, lock-step execution of declarative model, equivalency with cells.
8. Port objects (agents), reasoning with agents, state diagrams.
9. Basic message protocols, synchronization/callback methods, futures.

Shared-state concurrency

10. Explicit state, programming with explicit state – limiting interleavings, equivalency with ports.
11. Programming with locks, monitors and transactions.
12. Confinments, active objects (actors), difference with (Oz) agents.
13. Which model to use: declarative concurrency, message-passing concurrency, or shared-state concurrency. Expressiveness, readability, and reasoning.

Distributed programming

14. General ideas: network transparency, network awareness, openness, fault tolerance.
15. Mozart distribution model: behavior of different entities, distribution protocols.
16. Mozart 1.4 fault tolerance model: fault stream, fault states for different entities.
17. Erlang fault tolerance model: linking, system process, EXIT signal handling.

Concurrency elsewhere

18. Concurrency in other languages: choose one of Haskell, Erlang, Scala and compare concurrency with that in Oz