Model-Based Testing

(DIT848 / DAT260)

Spring 2015

Gerardo Schneider
Dept. of Computer Science and Engineering
Chalmers | University of Gothenburg

gerardo@cse.gu.se
http://www.cse.chalmers.se/~gersch/

Model-Based Testing

- What is testing?
  - The process of systematically experimenting with an object in order to establish its quality
  - Object "=" software -> software testing
- Why software testing?
  - Most used technique in industry to increase confidence in Sw quality
  - Job possibilities ☺
- What is model-based testing?
  - Generate tests (semi-)automatically from the model of the system under test
- Why model-based testing?
  - Cost saving, systematic approach to testing, automated traceability, early detection of flaws, etc.

Overview course content

- Overview on verification and validation
- Testing in general
  - Black box testing (Junit)
  - White box testing (Coverage analysis)
- Property-Based testing
  - QuickCheck
- FSM / EFSM
- Model-based testing
  - How to select your tests
  - Graph theory in MBT
  - ModelJUnit
  - Making your tests executably

How much of each topic?

- We will discuss today

Guest lectures:

- QuickCheck
- …

Theory and practice

Learning Outcomes

- Explain the distinction between software verification and validation;
- Describe the connection between software development phases and kinds of testing;
- Describe and explain (a number of) different test methods, and use them in practical situations;
- Describe and explain what model-based testing is;
- Construct models in the modeling and specification languages learned in the course;
- Construct appropriate and meaningful test cases, and interpret and explain (to stakeholders) the results of the application of such test cases (using appropriate tools) to practical examples;
- Apply model-based testing on realistic examples;
- Exemplify and describe tools for testing software, and use them and interpret their output;
- Identify and hypothesize about sources of program failures, and reflect on how to better verify the correctness of such programs.

Staff

- Gerardo Schneider – gerardo@cse.gu.se

- Guest Lecturers (on QuickCheck)
  - Koen Claessen (Chalmers) - TBC
  - John Hughes (Chalmers / QuviQ) - TBC

- Course assistant
  - Grégoire Détrez – gregoire.detrez@cse.gu.se

Student representatives

- (baxemyr@student.chalmers.se)
- Priyadarshini Chilaka (chilaka@student.chalmers.se)
- Nakyanzi lynnie Gwosuta (nakyanzi@student.chalmers.se)
- Annika Johansson (annikjoh@student.chalmers.se)

Course organization

- 12 lectures + tutorials
  - Approx. 2h each, in modules of around 45 min
  - Lectures given at the beginning of the course
  - Tutorials to help on the practical aspects
- Mini-projects – 3 HEC
  - **Mandatory!**
  - Meeting with the assistant on predefined dates
- All the information on the course page
  - http://www.cse.chalmers.se/edu/year/2015/course/DAT260/
  - Considered official! (Any message will be written in the News section under Home)
    - Students from Chalmers and GU – different systems
- Individual written exam – 4.5 HEC

Literature

- M. Utting and B. Legeard, **Practical Model-Based Testing**. Elsevier (Morgan Kaufmann Publishers, 2007)
  - An electronic version is available at http://bit.ly/wGlT94 (you must be logged in Chalmers' network to get access)
- Other interesting books
  - P.C. Jorgensen. **Software Testing: A Craftsman's Approach** (Auerbach Publications, 3rd edition, 2008)
  - Sommerville…
- Papers on QuickCheck
  - See course homepage
- Other books and references
  - See list in course homepage

Mini-Projects

- Not fixed days for meetings – Check homepage ("Lectures")
- Each group will choose a software system to be tested (among those proposed by the teachers, or by the group)
- Groups of 3 persons (exceptionally 2 members)
- 3 mandatory reports:
  - 30 Apr, 13:00: Manual testing and testing with Junit
  - 11 May, 13:00: EFSM
  - 26 May, 13:00: Final report (MBT using ModelJUnit)
- 27 May: Group's final presentation
- **Mandatory!** (To pass the course you need to pass both the mini-project and the written exam)
- Information about how to proceed to "register" your group, choose a topic, etc, will be posted in the homepage later this week

Written Exam (Individual)

- Written exam: June 3, 2015 at 8h30 (Johanneberg)
- Re-exam: August 24, 2014 at 14h00 (Johanneberg)

Important!

- The exam is designed to increase the confidence that a student passing the course achieve the Intended Learning Outcomes
- **Strongly recommended to learn when you work on the assignments!**
- So, *most probably* the exam will consist in 5 tasks widely covering the content of the course
- You will need to have at least **50**/100 points for getting ***G*** **(3)** (and at least **65** points for **4**) **and at least a minimum of correct answers for each task** (*e.g.*, **8 points** per task)
- To get ***VG*** **(5)** you will need to have at least **80**/100 points **and at least a minimum of correct answers for each task** (*e.g.*, **12 points** per task)
- *Open book* exam modality

Important: About examination!

For GU students:

- There are 2 "moments"
  - Individual written exam: 4.5 HEC (U, G, VG)
  - Assignment (mini-project): 3.0 HEC (U, G)

For Chalmers students:

- Formally speaking there is only 1 "moment" (the written exam – 7.5 HEC (U, 3,4,5))
  - To avoid discrimination the mini-project will be counted as for GU students
- NOTE: If you don't pass the course you will have to make the mni-project next year again!

The 2 "moments" modality will be applicable to Chalmers students from 2016

Changes w.r.t. last year
Result of course evaluation

- QuickCheck for Haskell: difficult for most students (more than 50% have not background on functional programming)
  - QuickCheck for Java this year
- Change the setting of the weekly meetings as ”open” consultation were not used; e.g. have tutorials on the tools used in the course and have more dedicated meetings
  - Mandatory meetings with each group on specific dates
  - There will be 4 tutorials
- Assignments: last one too long; have less assigments and more complex ones (and more time to work on them)
  - We will have mini-project instead of weekly assignments
  - Lectures have been reorganized so all the theory is there before starting working on assignments
- Some questions in the exam were ambiguous/unclear
  - We will improve the formulation of the written exam

What is your background?

Number of students: 15 (Chalmers) + 11 (GU)

- Knowledge on logic?
  - Propositional (classical): 5 (FOL: 0)
  - Other: 1 (LTL)
- Which functional prog. lang. do you know?
  - Haskell: 5
  - Erlang: 10
  - Other: 1 (Lisp)
- Which imperative/OO prog. lang. do you know?
  - Java: 24
  - C (C++): 24
  - Other: 6 (C# / Python)
- Knowledge on Testing? 11
- Knowledge on automata theory or FSM (Finite State Machines)? 5
- Knowledge on QuickCheck? 7

Preliminary schedule

- Is the content "appropriate" according to your background?
- Remember requirements for the course:
  - General programming knowledge in *imperative/object oriented* (ideally Java) (and *functional* programming)
  - Knowledge of *propositional logic*
  - Have some experience in *testing/debugging* your own programs

Wish list…

- What are your expectations?
- Something you would like to learn on testing (or verification in general) not covered in the programme?
- Are there topics you already know and don't want to see again?

About Registration…

- If you are a **GU student**
  - You need to register through the *Student portal* at GU
- For **Chalmers** students (late registration)
  - Contact the *Studieexpeditionen* (student office) student_office.cse@chalmers.se

Questions?

Check the course page regularly

Hope you enjoy the course!