From Domains via Requirements to Software Designs
University of Tokyo, Graduate School Lectures, 2009

Dines Bjørner
Fredsvej 11, DK-2840 Holte, Danmark
E–Mail: bjorner@gmail.com, URL: www.imm.dtu.dk/~db

June 6, 2008

1 Abstract

A plan is suggested for DB's

- three (3) full day,
- four double (two morning and two afternoon) lectures per day
- University of Tokyo, Graduate School of Arts and Sciences,
- computing science lectures, 2009

Contents

2 Lecture Aims and Objectives

2.1 Aims

- The lecturer will survey three views of software development:
  - ⋆ a software engineering view,
  - ⋆ a programming methodological view, and
  - ⋆ a computing science research topics view.

2.2 Objectives

- Course participants should gain new insight into the following aspects of software development:
  - ⋆ the triptych paradigm of software development:
    - ⊕ that before **software** can be **designed**
    - ⊕ the developer must understand the **requirements**, and that before requirements can be expressed
    - ⊕ the developer must understand the application **domain**.
  - ⋆ so that, in an idealistic sense software is developed by
    - ⊕ first researching and developing a **domain description**,
    - ⊕ then developing the **requirements prescription**, and
    - ⊕ finally developing the **software design**.
- Thus the course participants will gain insight into the new **principles, techniques** and **tools** of
  - ⋆ **domain modelling** the
    - ⊕ intrinsics,
    - ⊕ support technologies,
    - ⊕ management and organisation,
    - ⊕ rules and regulations,
    - ⊕ scripts and
    - ⊕ human behaviour

    facets of domain;
  - ⋆ and **requirements modelling** the
    - ⊕ **domain** (functional (?)) **requirements**, that is, the modelling of
      - ⊗ projection,
      - ⊗ instantiation,
      - ⊗ determination,
      - ⊗ extension

        and
      - ⊗ fitting

      of domains onto requirements;
    - ⊕ **interface** ("user" (?)) **requirements**, that is, modelling the
      - ⊗ entity sharing,
      - ⊗ function sharing,
      - ⊗ event sharing and
      - ⊗ behaviour

      of the phenomena and concepts shared between the domain and the machine;
    - ⊕ **machine** (system (?)) **requirements**, that is, the modelling of machine (i.e., hardware + software)
      - ○ performance,
      - ○ dependability,
      - ○ maintenance,
      - ○ platform,
      - ○ documentation,
      - ○ etc.

3 Lecture Plan

- **First Day: Engineering Topics**
  - ⋆ **9:00–10:30 Course Overview**
  - ⋆ **11:00–12:30 Domain Engineering**
  - ⋆ **14:30–16:00 Requirements Engineering**
  - ⋆ **16:30–18:00 Software Design**
- **Second Day: Programming Methodology Topics**
  - ⋆ **9:00–10:30 Domain Modelling I**
  - ⋆ **11:00–12:30 Domain Modelling II**
  - ⋆ **14:30–16:00 Requirements Modelling I**
  - ⋆ **16:30–18:00 Requirements Modelling II**
- **Third Day: Research Topics**
  - ⋆ **9:00–10:30 Domain Engineering Research Topics**
  - ⋆ **11:00–12:30 Compositionality: Ontology and Mereology of Domains**
  - ⋆ **14:30–16:00 Management of Software Development**
  - ⋆ **16:30–17:00 Summary**

4 Practicalities

- The three days need not be adjacent
  - – that is, they can be spread over three consecutive weeks,
  - – or Monday, Wednesday and Friday of, say the second or third week

  of a four week visit.
- The lecturer will provide a complete set of texts and slides.