ECNU MSc/PhD Course + Project
5–23 November 2018
Domain Analysis & Description
Principles, Techniques and Languages

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

October 31, 2018

Contents

1 Lecture Particulars

1.1 Title: Domain Analysis & Description

- [1] 57 page Paper Domain Analysis and Description
  Principles, Techniques and Languages

- [1] 400 slide **Lectures** **Domain Analysis and Description** **Principles, Techniques and Languages**

1.2 Aims & Objectives

1.2.1 Aims

- The course is **aimed at** 3rd–4th year MSc + PhD student + PostDocs + Et cetera!
- The course **aims to** introduce participants to a new aspect of the Computing Sciences

1.2.2 Objectives

The objective of the course is

- to **enable** participants to **rough-sketch** domain models,
- to **entice** them to **study** domain science & engineering as per [2], and
- to, perhaps, even **lure** them into **research** in domain science & engineering.

1.3 Prerequisites

- **Motivation:** You must be interested in **software as mathematical artifacts**
- **Discrete Mathematics:** Sets, Cartesians, Algebra, ...
- **Basic Knowledge of Logic:** propositional and predicate logic
- **A Smattering of Functional Programming:**
  one of f.ex.: *Coq, Curry, Erlang, F#, Haskell, LISP, [Standard] ML, Ruby, Scala, Scheme, ...*

2 The Didactic Base

By a *didactic base* we shall understand the knowledge "spheres" within which we operate.
Our *didactic base* for software development is outlined below.
That base is also suggested as the *didactic base* for software engineering.

2.1 Method, Methodology and Formal Methods

- **Method:**
  - By a method we shall understand a set of **principles** for **selecting** and **applying** a number of **analysis & synthesis techniques** and **tools** in order to achieve a goal
  - where that goal here is to develop a software specification
  - whether that specification be a
    - a domain description,
    - a requirements prescription,
    - a software design and code,
    - or the first or last two, or all of these.
- **Methodology:**
  - By methodology we shall understand the study and knowledge of one or more methods.
- **Formal Methods:**

  - By a formal method we shall understand a method several of whose techniques and tools
    can be explained **mathematically**, such as, e.g.,
    * refinements,
    * tests, model checks, theorem proofs,
    * specification language syntax, semantics and proof systems.

The present course endows domain analysis & description with a formal method.

2.2 The Computer & Computing Sciences

2.2.1 Computer Science

- By computer science we shall understand the study and knowledge of the **properties** of the kind of phenomena that *"goes on inside" computers*.

2.2.2 Computing Science

- By computing science we shall understand the study and knowledge of how those phenomena (*"inside" computers*) can be **constructed.**

The present course is a computing science course.

2.3 The Triptych Dogma

- Before **software** can be **designed & coded**
- we must have a reasonable grasp of what is **expected & required** from that software,
- and before we can **prescribe** those **expectations & requirements**
- we must have a reasonable grasp of the **domain**, i.e., be able to **describe** it.

As a consequence we can claim that

- **Software Systems Development** can be "divided" into three phases:
  - **Domain Science & Engineering**
  - **Requirements Engineering**
  - **Software Design**

In this course we shall only consider domain analysis & description.

2.4 Informatics & IT

2.4.1 Informatics

- By **informatics** we shall understand a confluence of
  - mathematics: "pure" as well as "applied",
  - computer & computing science, and
  - software.

To us informatics is a universe of quality: correct, fit-for-purpose and pleasing

2.4.2 IT: Information Technology

- By **information technology** we shall understand a confluence of
  - hardware
  - the natural science-based technologies that *"go into making"* hardware:

∗ electronics, ∗ mechanics, ∗ chemistry, ∗ et cetera.

To us IT is a universe of quantity: faster, larger, cheaper, etc.

3 Schedule

3.1 Day/Week/Month Overview – A Suggestion

NOVEMBER 2018

Monday	Tuesday	Wednesday	Thursday	Friday
		*7	8	*9
12	13	*14+	15	*16
19	20	*21	22	*23

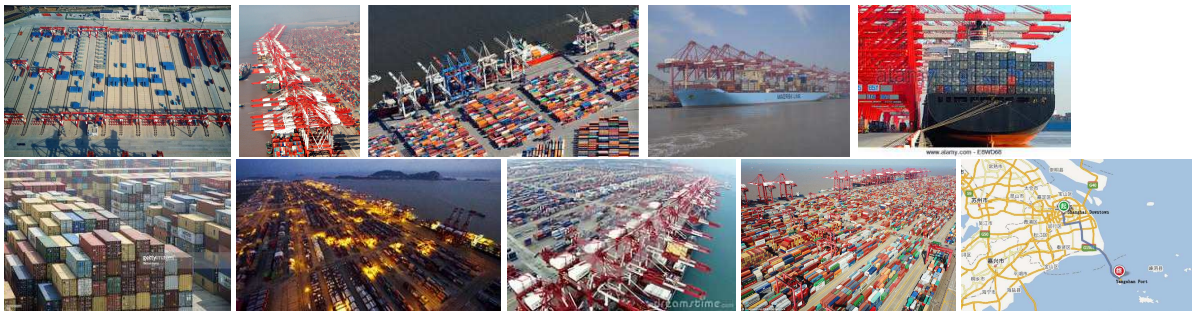
* Lecture and Project dates
\+ Excursion to SECT, 12:00-16:30
Wednesdays: Math.Bldg., Room 201
Fridays: SanGuan Bldg., Room 231

3.2 Office Hourse

- The lecturer, **Dines Bjørner**, is available all weekday mornings:
  - On lecture / project session days in connection with lectures and project sessions
  - On other days, in the office, 9:00–12:00

4 Course Project

- It is proposed that the class analyse & describe
  a *generic container terminal port*, a la Shanghai's YangShan port.



- Topic is decided upon and participants on the first course day, 6.11
- Two hour project sessions every course day, right after lecture
- First project session is on first course day, 6.11
- At that section lecturer explains problem and refers to subject literature,
  starts sketching a domain model and directs participants as their home work
- In preparation for each project session participants work out paper descriptions

- Each, but the first, project session starts with lecturer and participants discussing partipant home work based on distributed copies of these
- The lecturer – in-between and subsequently – proceeds, jointly with participants, with further "white board" modelling and directs next homework assignments
- Lecturer will, as a result of and in step with project session progress "formalise" current work and post this on the net.

A sketch description of an essence of container terminal ports should result from this course.

5 Literature

- [1, 3, Domains Analysis & Description]
- [4, 5, Domain Facets: Analysis & Description]
- [6, 7, Formal Models of Processes and Prompts]
- [8, 9, To Every Manifest Domain Mereology a CSP Expression]
- [10, 11, From Domain Descriptions to Requirements Prescriptions]
- [12, 13, Domains: Their Simulation, Monitoring and Control]
- [14, Domain Analysis & Description: Some Issues of Philosophy]
- [2, Domain Science & Engineering: A Compendium], a collection of [1, 4, 6, 8, 10, 12, 14]

Bibliography

[1] Dines Bjørner. A Domain Analysis & Description Method – Principles, Techniques and Modeling Languages. Research Note based on [3], Technical University of Denmark, Fredsvej 11, DK-2840 Holte, Denmark, February 20 2018. http://www.imm.dtu.dk/ dibj/2018/tosem/Bjorner-TOSEM.pdf.

[2] Dines Bjørner. Domain Science & Engineering – A Compendium. Technical report, Technical University of Denmark, Fredsvej 11, DK-2840 Holte, Denmark, June 2018. A collection of [1, 4, 6, 8, 10, 12, 14].

[3] Dines Bjørner. Manifest Domains: Analysis & Description. *Formal Aspects of Computing*, 29(2):175–225, Online: July 2016.

[4] Dines Bjørner. Domain Facets: Analysis & Description. May 2018. Extensive revision of [5]. http://www.imm.dtu.dk/~dibj/2016/-facets/faoc-facets.pdf.

[5] Dines Bjørner. Domain Engineering. In Paul Boca and Jonathan Bowen, editors, *Formal Methods: State of the Art and New Directions*, Eds. Paul Boca and Jonathan Bowen, pages 1–42, London, UK, 2010. Springer.

[6] Dines Bjørner. Domain Analysis and Description – Formal Models of Processes and Prompts. Technical report, Technical University of Denmark, Fredsvej 11, DK-2840 Holte, Denmark, 2016. Extensive revision of [7]. http://www.imm.dtu.dk/~dibj/2016/process/-process-p.pdf.

[7] Dines Bjørner. Domain Analysis: Endurants – An Analysis & Description Process Model. In Shusaku Iida, José Meseguer, and Kazuhiro Ogata, editors, *Specification, Algebra, and Software: A Festschrift Symposium in Honor of Kokichi Futatsugi*. Springer, May 2014.

[8] Dines Bjørner. To Every Manifest Domain a CSP Expression — A Rôle for Mereology in Computer Science. *Journal of Logical and Algebraic Methods in Programming*, (94):91–108, January 2018.

[9] Dines Bjørner. On Mereologies in Computing Science. In *Festschrift: Reflections on the Work of C.A.R. Hoare*, History of Computing (eds. Cliff B. Jones, A.W. Roscoe and Kenneth R. Wood), pages 47–70, London, UK, 2009. Springer.

[10] Dines Bjørner. From Domain Descriptions to Requirements Prescriptions – A Different Approach to Requirements Engineering. Technical report, Technical University of Denmark, Fredsvej 11, DK-2840 Holte, Denmark, 2016. Extensive revision of [11].

[11] Dines Bjørner. From Domains to Requirements. In *Montanari Festschrift*, volume 5065 of *Lecture Notes in Computer Science (eds. Pierpaolo Degano, Rocco De Nicola and José Meseguer)*, pages 1–30, Heidelberg, May 2008. Springer.

[12] Dines Bjørner. Domains: Their Simulation, Monitoring and Control – A Divertimento of Ideas and Suggestions. Technical report, Technical University of Denmark, Fredsvej 11, DK-2840 Holte, Denmark, 2016. Extensive revision of [13]. http://www.imm.dtu.dk-/~dibj/2016/demos/faoc-demo.pdf.

[13] Dines Bjørner. Domains: Their Simulation, Monitoring and Control – A Divertimento of Ideas and Suggestions. In *Rainbow of Computer Science, Festschrift for Hermann Maurer on the Occasion of His 70th Anniversary.*, Festschrift (eds. C. Calude, G. Rozenberg and A. Saloma), pages 167–183. Springer, Heidelberg, Germany, January 2011.

[14] Dines Bjørner. A Philosophy of Domain Science & Engineering – An Interpretation of Kai Sørlander's Philosophy. Research Note, Technical University of Denmark, Fredsvej 11, DK-2840 Holte, Denmark, Spring 2018. http://www.imm.dtu.dk/~dibj/2018/-philosophy/filo.pdf.