Lectures on Domain Engineering

Dines Bjørner

Faculté des Sciences, Bureau 266,
LORIA & Université Henri Poincaré Nancy 1,
BP 239, F-54506 Vandœuvre lès Nancy, France

E–Mail: bjorner@gmail.com , URL: www.imm.dtu.dk/~db. [1]

December 3, 2007

Contents

1 Summary

- **Course:** Domain Engineering

[1] www.imm.dtu.dk/ db

- **Prerequisite:** Basic knowledge of Abstraction using B / event B
  Thus the course is an advanced MSc or a beginners PhD course
- **Lecturer:** Dines Bjørner
  (Prof., Dr., Dr.h.c., MAE, MRANS, ACM Fellow, IEEE Fellow, ...)
- **Course Time:** Tuesdays 10–11:30 pm (90 mins.), Thursdays 2–3:30 pm (90 mins.)
- **Dates:** November: 6, 8, 13, 15, 20, 22, 27 and 29, and December 4 and 6.
- **Location:** Salle SC08, Batiment 2eme Cycle, Entreé 5A

2 Abstract

- **The Dogma:** We idealise major properties of software development:
  - Before software can be designed we must understand its requirements: that is, what the software should do, not how.
  - Before we can express the requirements we must understand the application domain, as it is, without any reference to the desired (required) computing.
- **Consequences:** We idealise sequence of software development phases:
  - First we domain engineer a proper description, informal and formal, of the domain, $\mathcal{D}$, as it is (not as we would want it to be).
  - Then we requirements engineer a proper prescription, informal and formal, of the requirements, $\mathcal{R}$, as we would want the software to
  - And finally we develop the software, $\mathcal{S}$, informally and formally.

  The basic idea is that we can eventually prove the software correct with respect to the requirements under the assumptions of the domain:

$$\mathcal{D}, \mathcal{S} \models \mathcal{R}$$

- **The Course:**
  - The course will primarily cover some, but not all aspects of a methodology of domain engineering.
  - The course will briefly indicate how to proceed from domain engine ring into requirements engineering.
  - Details are given below.

3 Domain Engineering

- Domain engineering is a relatively new part of software engineering.
- We motivated its role above.
- We justify its presence below.

3.1 What Is Domain Engineering ?

- In domain engineering the software developer constructs (i.e., 'engineers') a description of the domain
  - within which a software application is to be developed,
  - or for which we just wish to understand that domain.
- The domain engineer starts by identifying all the relevant domain stakeholders.
- The domain engineer then proceeds to study the domain: to acquire information about domain entities, functions, events and behaviours — usually based on some rough sketches of a number of domain "business" processes.
- The domain engineer analyses the acquired domain information, looks for inconsistencies and undesirable incompletenesses, removes these and forms abstract concepts.
- The domain engineer then carries out the major task of modelling the domain: informally and formally, while verifying implicitly expressed properties — thus building up a domain theory.
- The domain engineer finally validates the domain model "against" the stakeholders.

3.2 Why Domain Engineering ?

- The **dogma** above gave a main motivation.
- Further justification can be provided.
- The science of physics (electricity, plasma physics, mechanics, aerodynamics, etc.) is the basis for electrical and electronics engineering, for automotive engineering, for aeronautics, etc. Design engineers would not be hired into such engineering firms unless they have demonstrated clear mastery of their underlying science.
- Not so in software engineering. Many software houses — specialising in software for transportation systems, or for health care, or for the financial service industry, or for manufacturing, or for "the market" — employ programmers who have no prior knowledge of transportation, health care, the financial service industry, manufacturing, or "the market", respectively.

- So it seems that domain engineering can be justified. There are other justifications for practising domain engineering or for studying domains. Some of these will be mentioned during the course lectures.

4 The Course

4.1 Aims

- The course aims at covering the following aspects of domain engineering:
  - A rough outline of the stages and steps of domain engineering (1 lecture):
    - the notion of domain stakeholders,
    - the notion of domain acquisition,
    - the notion of domain verification and
    - the notion of domain validation.
  - A detailed review of domain modelling facets (7 lectures):
    - intrinsics,
    - support technologies,
    - management and organisation,
    - rules and regulations,
    - scripts and
    - human behaviour.
  - An overview of requirements engineering (4 lectures):
    - domain requirements,
    - interface requirements and
    - machine requirements.

4.2 Objectives

There are two loosely related setsa of objectives:

- To publish a book !
  - For the lecturer, tutors and students to develop the main text for a book **On Transport Domain Modelling** to be published
  - a book in which even numbered pages contain the domain description in **English** and using the RAISE Specification Language, **RSL**
  - and the odd numbered pages contain the domain description in **French** and using **B / event B** !

- The academic objective:
  - The "near" term course objectives are:

    - to acquaint the graduated course participants with the **triptych** model of software development so that the participants are better equipped to participate in or lead proper software development projects;
    - to enable the graduated course participants to themselves describe domains: informally and formally.
  - The longer term course objectives are:
    - to enable researchers to identify interesting and outstanding (open) research problems
    - (and thus, by research techniques obtained elsewhere, to contribute to their solution).
  - A more general objective is to indicate
    - that domain engineering is a required research and engineering discipline and
    - that domain engineering ought be part of every research and engineering curriculum.

4.3 Lectures and Tutoring Sessions

- There will be 10 lectures of 90 minutes each
- Most lectures have two parts
  - "Formal Part": Methodology
  - "Project Part": Applying the Methodology
- The lecturer will, additionally tutor each of 2–3 project groups 1–2 times weekly, 1–2 hours/time

4.4 Lecture Material

- Cover, Preface, Table of Contents and Chapters 8, 11, 17 and 19
  - 186 Click for Vol.3 Text Pages[2]
  - 682 Click for Nancy Lecture Slides[3]

  of Vol. 3: Software Engineering
  - Springer Home Page[4]
  - Software Engineering Home Page[5]

There are some supporting documents:

- A 42 page paper on Domain Engineering[6] (chapter in a book to be published by Springer).
  Summarises Domain Engineering.

[2] http://www.loria.fr/ bjoerned/nancy.pdf
[3] http://www.loria.fr/ bjoerned/nancy-1-9.pdf
[4] http://www.springer.com/east/home/computer?SGWID=5-146-6-467209-0
[5] http://www2.imm.dtu.dk/~db/Software_Engineering/
[6] http://www.loria.fr/ bjoerned/domain-paper-1Nov2007.pdf

- Domain Engineering: Practice and Theories[7]
  Reviews Domain Engineering and Discusses Research Challenges
- From Domains to Requirements[8] - a complete first draft of a paper for Prof. Ugo Montanari's 65 Festschrift May 2008.
  Summarises Domain Engineering and Requirements Engineering.
- Believable Software Management[9] - an incomplete draft of a paper for *Encyclopedia of Software Engineering* (ed. Philip Laplante, Taylor & Francis Publ.)
  Develops the theme of Software Management.
- Development of Transportation Systems[10] - an incomplete draft of a paper for *ISOLA 2007*[11]
  Shows how a generic model of a transportation domain can be "refined" (instantiated) into domain (or even requirements) models for railways, air traffic and (by reference) shipping (more precisely The Container Line Shipping Domain[12]).

4.5 Project Material

- The lecturer's **Transportation Domain** Project Report (English and RSL)
  - Click for Nancy Project Text Pages, PS File[13].
  - Click for Nancy Project Slides, PDF File[14] – four slides to one page.

Some of the material for the above comes from:

- From Domains to Requirements[15]

4.6 Project

- Class will – ideally – produce a 100–200 page report
- The report will be a domain description:
  - by you in French and B/event B
  - by me in English and RSL (Raise Spec. Lang.)
- Domain will be either of:

[7] http://www.loria.fr/ bjoerned/bjorner-ictac.pdf
[8] http://www.loria.fr/ bjoerned/ugo65.pdf
[9] http://www.loria.fr/ bjoerned/bsm.pdf
[10] http://www.loria.fr/ bjoerned/isola-pa.pdf
[11] http://www.isola2007.ensma.fr/
[12] http://www2.imm.dtu.dk/ db/container-paper.pdf
[13] http://www.loria.fr/ bjoerned/tp.ps
[14] http://www.loria.fr/ bjoerned/4ts.pdf
[15] http://www.loria.fr/ bjoerned/ugo65.pdf

  - Financial Services Industry
  - The Health Care Industry
  - The Market
  - **Transportation**

- Class will be "divided" into 2–3 groups
- Each group: 3–4 participants
- After course: Consolidation of Group Reports into an edited draft book — **French/B/event B/Rodin** + **English/RSL** !

4.7 Group Work

- You will form 3-4 person groups.
- Each group will get help from a PhD student.
- Each such group will meet to decide on which domain topics are to be described by which members of the group.
- Each such member will create and edit appropriate .tex documents for the above purpose.
- Group will exchange these draft documents.
- Lecturer will meet each group at least once a week for 60 to 90 minutes to discuss your reports, open problems, etc.

4.8 Evaluation

- Each group will contribute to the Project Report
- Each group report will cover possibly overlapping areas of the domain
- Each group report will will be separately evaluated
- There will be an exam (planned for two (2) hours)
- Your grade will be determined as follows:
  - 60% by your group reports evaluation
  - 40% by your examn evaluation

5 Lecture Plan

- (1) **Introduction**
  - Administrative Information
    - Why Domain Engineering ?
    - Lectures and Tutoring

    * Groups and Project Report
    * Examination

  + **Project:**

– "Top of an iceberg":
A Transportation Domain Description

Click for Lecture 1 Slides[16]

Domain Engineering

- (2) Overview of Domain Engineering, Chap. 8 + Domain Facilitators: Business Processes, Chap. 11 + **Project**

  Lecture is, exceptionally, at 10-11:30 am, 8 Nov.

  Click for Lecture 2 Slides[17]

  Domain Facets Chap. 11

- (3) Intrinsics + **Project**

  Click for Lecture 3 Slides[18]

- (4) Support Technologies + **Project**

  Click for Lecture 4 Slides[19]

- (5) Management and Organisation + Rules and Regulations + **Project**

  Click for Lecture 5 Slides[20]

- (6) Scripts + Human Behaviour + **Project**

  Click for Lecture 6 Slides[21]

  Click for Lecture 6B: Five Slides: Human Behaviour Example[22]

  Requirements Engineering

- (7) Overview of Requirements Engineering, Chap. 17

  Click for Lecture 7 Slides[23]

- (8) Domain Requirements, Chap. 19

  Click for Lecture 8 Slides[24]

- (9) Interface and Machine Requirements, Chap. 19

  Click for Lecture 9 Slides[25]

- (10) Pls. attend some lecture(s) at this event:
  To B or in any Event to B [26]

[16] http://www.loria.fr/ bjoerned/nancy-1.pdf
[17] http://www.loria.fr/ bjoerned/nancy-2.pdf
[18] http://www.loria.fr/ bjoerned/nancy-3.pdf
[19] http://www.loria.fr/ bjoerned/nancy-4.pdf
[20] http://www.loria.fr/ bjoerned/nancy-5.pdf
[21] http://www.loria.fr/ bjoerned/nancy-6.pdf
[22] http://www.loria.fr/ bjoerned/nancy-6b.pdf
[23] http://www.loria.fr/ bjoerned/nancy-7.pdf
[24] http://www.loria.fr/ bjoerned/nancy-8.pdf
[25] http://www.loria.fr/ bjoerned/nancy-9.pdf
[26] http://www.loria.fr/ mery/nicolas.htm

6 Lecture and Tutoring Dates and Hours

- Lectures will be twice a week, each time 90 minutes.
- Tuesday mornings and Thursday afternoons.
- Group tutoring hours will be negotiated with each group.
- Typically these tutoring hours will be on Tuesdays, Wednesdays and Thursdays.

```
        November                          December

 10-11:30,14-15:30                 10-11:30,14-15:30

Mon Tue Wed Thu Fri Sat Sun       Mon Tue Wed Thu Fri Sat Sun

      6       8(*)                      4       6
     13      15
     20      22
     27      29

(*) 10-11:30
```

7 Lecturer

7.1 Dines Bjørner

7.2 Home Pages

- My home page.[27]
- Domains: My Research Area[28]

7.3 Ultrashort CV

Dines Bjørner (DB) got an MSc.E.E. from The Technical University of Denmark (DTU) in January 1962, a PhD in CS also from DTU, and was at IBM R&D March 1962 – August 1975.

DB has been a professor of computing science between 1976 and 2007 at the Technical University of Denmark. DB tutored more than 120 MSc Theses and 22 PhD Theses.

DB founded and initially directed the UN University's International Institute for Software Technology, 1991–1997: http://www.iist.unu.edu.

For the period of October 15 to December 15, 2007, DB is visiting Faculté des Sciences, bureau 266, LORIA & Universiét Henri Poincaré Nancy 1, BP 239, F-54506 Vandoeuvre lès Nancy, France.

DB has published more than 120 papers, co-written two books, edited and co-edited 12 books and authored one three volume 2414 page book: http://www2.imm.dtu.dk/~db/Software_Engineering.

[27] http://www2.imm.dtu.dk/~db
[28] http://www2.imm.dtu.dk/~db/index.html#SECTION00040000000000000000

7.4 Detailed, "Chatty", CV

- Born in Odense, Denmark, 4 October 1937, DB, grew up in Århus where he attended the Århus Cathedral School (founded in 1142). DB graduated in January 1962 with an M.Sc. in Electronics Engineering and with a Ph.D. in Computer Science in January 1969 from the Technical University of Denmark (founded by Hans Christian Ørsted in 1828).

- DB joined IBM in March 1962 at their Nordic Laboratories (founded by Cai Kinberg) in Stockholm, Sweden (where DB also first met Jean Paul Jacob and Gunnar Wedell). DB was transferred to the IBM Systems Development Division (IBM SDD) at San Jose, California, USA, in December 1963. While doing his Ph.D. (September 1965–January 1969) DB was a lecturing consultant to IBM's European Systems Research Institute (ESRI) at Geneva, Switzerland (where DB acquainted with Carlo Santacroce and where DB's friendship with Gerald Weinberg started) (1967–1968). In 1969 DB worked at IBM's Advanced Computing Systems (IBM ACS) laboratory, Menlo Park, California, and, later that year until early 1973 at IBM Research, San Jose (again Jean Paul Jacob became a colleague). Transferred to the IBM Vienna Laboratory (directed then by Heinz Zemanek), Austria, DB resigned from IBM in August 1975 to return to Denmark after basically 13 years abroad.

- During his stay at IBM Research DB was a visiting lecturer, for several quarters, at University of California at Berkeley (1971–1972), instigated by Lotfi Zadeh whom DB considers his main mentor and for whom DB has the fondest regards. DB was a visiting guest professor at Copenhagen University in the academic year 1975–1976, before taking up his present chair in September 1976 at the Technical University of Denmark (DTU). During the summer semester of 1980 DB was the Danish Chair Professor at the Christian-Albrechts University of Kiel, Germany. DB was the founding and first UN Director of UNU/IIST, the United Nations University's International Institute for Software Technology, located in Macau. During the 1980s DB was chief scientist at Dansk Datamatik Center (DDC). (DB was a co-founder of DDC in 1979 and interim managing director in 1979.) In 1982–1984 DB was chairman of a Danish Government (Ministry of Education) Commission on Informatics.

- For a year: 2004-2005 DB was a visiting professor at National University of Singapore, and for the year 2006 DB was a research professor at at JAIST: Japan Advanced Institute for Science and Technology.

- DB has lectured and regularly lectures on six continents in almost 50 countries and territories and has graduated more than 80 MSc's and a dozen PhDs.

- At IBM DB first worked in the hardware (logic and systems) design of such equipment as the IBM 1070 (Sweden), the IBM 1800 and IBM 1130 computers (San Jose), and, finally, with Gene Amdahl and Ed Sussenguth, the IBM ACS/1 supercomputer (Menlo Park). At Research DB worked with John W. Backus and Ted Codd on Functional Languages, resp. Relational Data Base Systems. At Vienna, DB, together with such colleagues as Peter Lucas, the late Hans Bekič, Kurt Walk, and Cliff B. Jones, worked on a Denotational (–like) Semantics Description of PL/I while, with his colleagues conceiving, researching, developing and using VDM (the Vienna software Development Method). At DTU and at DDC, supported by the European Community, DB initiated several advanced research & development projects: (1) Formal Semantics Description of and (2) full language compiler for CHILL (the Intl. Telecommunications Unions Communications [C.C.I.T.T.] High Level Language), (4) Formal Semantics Description of and (3) the first European US DoD officially validated compiler for the US DoD Ada embedded systems programming language, (5) RAISE (Rigorous Approach to Industrial Software Engineering), (6) Formal Semantics Definition of VDM–SL (the VDM Specification Language), (7) ProCoS (Provably Correct Systems) with, amongst others Profs. Sir Tony Hoare (then Oxford, now Microsoft Research, Cambridge, UK), Hans Langmaack (Kiel) and Ernst-Rüdiger Olderog (Oldenburg), etc.

- At UNU/IIST DB had a rather free hand, and was able, with a small team of excellent colleagues (Prof. Zhou Chaochen (Academician, the Chinese Academy of Science), Søren Prehn (CRI, now Terma Electronics), Chris W. George, Richard Moore, Tomasz Janowski, Dang Van Hung, Xu Qi Wen and Kees Middelburg), to further explore the research issues currently (still) occupying DB's interest, and to apply them (ie. test them out) in a number of joint R&D projects with institutions in developing and newly industrialised countries [including newly independent states] (Argentina, Belarus, Brasil, Cameroun, China, Gabon, India, Indonesia, Mongolia, North Korea, Pakistan, Philippines, Poland, Romania, Russia, South Africa, South Korea, Thailand, Vietnam, Ukraine, Uruguay, etc.).

- DB was a co–founder of VDM-Europe in 1987 and moved VDM-Europe onto FME: Formal Methods Europe in 1991. DB co-chaired two of the VDM Symposia (1987, 1990), and the International Conference on Software Engineering (ICSE) in 1989 in Pittsburgh, Pennsylvania, USA. DB was chairman of the IFIP World Congress in Dublin, Ireland in 1986, and was the instigator and General Chairman of the first World Congress on Formal Methods, FM'99, in Toulouse, France, September 20–24, 1999.

- DB was knighted (The Order of Dannebrog[29], 1985); is a member of Academia Europaea[30] (MAE, and is chairman of its Informatics Section[31]), the Russian Academy of Natural Sciences (MRANS [AB]) and of IFIP Working Group 2.3 (1980–...), and is a member emeritus of IFIP Working Group 2.2. DB has received the IFIP Silver Core (1986), the IEEE Golden Core (1989), the John von Neumann Medal of the JvN Society of Hungary (1993), the Masaryk Gold Medal from the Masaryk University, Brno, The Czech Republic (1996) and the Danish Engineering Society (IDA) Information Technology Division's (IDA–IT) first BIT prize (1999).

- DB has authored more than 100 published papers and co-authored and co-edited some 15 books – and published, with Springer, in 2006 a three volume book on Software Engineering[32] (Vol.1[33], Vol.2[34], Vol.3[35]). A latest, co-edited book, Logics of Specification Languages[36], is due any day!

- DB's research interests, since his Vienna days, have centered on programming methodology: Methods as sets of principles for selecting and applying mathematics–based analysis and construction techniques and tools in order efficiently to construct efficient artifacts — software.

  DB sees his main contributions to be in the research, development and propagation of formal specification principles and techniques. DB has for some time focused on the triptych of domain engineering, requirements engineering and the software architecture and program organisation methods — emphasising such that relate these in mathematical as well as technical ways: (1) Intrinsic, support technology, management & organisation, rules & regulation, and human behaviour facets of domains; (2) projection, instantiation, extension and initialisation of domain requirements, etc.; (3) software architectures as refinements of domain requirements, and program organisation as refinements of machine requirements — with interface requirements (currently) being refinements of either and both! Currently, in his retirement, DB wishes to focus on domain theory: practice (container shipping, transport networks, railways, financial service industry, etc.) and theories.

- DB retired on March 31, 2007.

[29] http://kongehuset.dk/publish.php?dogtag=k_en_his_ord
[30] http://www.acadeuro.org/
[31] http://www.iis.ee.ic.ac.uk/~e.gelenbe/AEInformatics.html
[32] http://www.springer.com/east/home/computer?SGWID=5-146-6-467209-0
[33] http://www.springer.com/east/home/computer/foundations?SGWID=5-156-22-43949164-0
[34] http://www.springer.com/east/home/computer/foundations?SGWID=5-156-22-43949263-0
[35] http://www.springer.com/east/home/computer/foundations?SGWID=5-156-22-43949319-0
[36] http://www.springer.com/978-3-540-74106-0

- Among the very many people who have been kind to DB and helped DB in his professional career, he wishes to bear tribute, in approximate chronological order, to (the late) Cai Kinberg, Gunnar Wedell, Jean Paul Jacob, Gerald Weinberg, Carlo Santacroce, Gene Amdahl, Ed Sussenguth, Tien Chi (T.C.) Chen, Lotfi Zadeh, (the late) Ted Codd, (the late) John W. Backus, Peter Lucas, Cliff Jones, (the late) Hans Bekič, Kurt Walk, Ole N. Oest, Erich Neuhold, (the late) Søren Prehn, Sir Tony Hoare, Hans Langmaack, Zhou Chao Chen, and Chris George.

...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...
...