Dines Bjørner's MAP-i Lecture # 12

Discussion of Research Topics

Thursday, 28 May 2015: 16:45–17:30

9. Discussion of Research Topics

- There are a number of research topics:
  - some relate to domain analysis & description, cf. Chapter 1, and some of these are listed in Sect. 8.1,
  - other relate to requirements engineering, cf. Chapter 7, and some of these are listed in Sect. 8.2.

9.1. Domain Science & Engineering Topics

- The TripTych approach to software development,
  - based on an initial, serious phase of domain engineering,
  - a new phase of software engineering,
  - for which we claim to now have laid a solid foundation for domain engineering —
- opens up for a variety of issues that need further study.
- The entries in this section are not ordered according to any specific principle.

9.1.1. Analysis & Description Calculi for Other Domains

- The analysis and description calculus of this paper appears suitable for manifest domains.
- For other domains other calculi appears necessary.
  - There is the introvert, composite domain of systems software:
    - operating systems, compilers, database management systems, Internet-related software, etcetera.
    - The classical computer science and software engineering disciplines related to these components of systems software appears to have provided the necessary analysis and description "calculi."

- There is the domain of financial systems software
  - accounting & bookkeeping,
  - banking systems,
  - insurance,
  - financial instruments handling (stocks, etc.),
  - etcetera.

.

- Etcetera.
- For each domain characterisable by a distinct set of analysis & description calculus prompts such calculi must be identified.

- It seems straightforward:
  - to base a method for analysing & describing a category of domains
  - on the idea of prompts like those developed in this lecture.

9.1.2. On Domain Description Languages

- We have in this seminar expressed the domain descriptions in the RAISE [40] specification language RSL [39].
- With what is thought of as basically inessential, editorial changes, one can reformulate these domain description texts in either of
  - Alloy [45] or
  - The B-Method [1] or
  - VDM [30, 31, 37] or
  - Z [55].

- One could also express domain descriptions algebraically, for example in `CafeOBJ`.
  - The analysis and the description prompts remain the same.
  - The description prompts now lead to `CafeOBJ` texts.

- We did not go into much detail with respect to perdurants, let alone behaviours.
  - For all the very many domain descriptions, covered elsewhere, `RSL` (with its `CSP` sub-language) suffices.
  - But there are cases where we have conjoined our `RSL` domain descriptions with descriptions in
    - `Petri Nets` [52] or
    - `MSC` [44] or
    - `StateCharts` [42].

- Since this seminar only focused on endurants there was no need, it appears, to get involved in temporal issues.
- When that becomes necessary, in a study or description of perdurants, then we either deploy
  - `DC: The Duration Calculus` [56] or
  - `TLA+: Temporal Logic of Actions` [48].

9.1.3. Ontology Relations

- A more exact understanding of the relations between
  - the "classical" AI/information science/ontology view of domains [4, 5, 46], and
  - the algorithmic view of domains, as presented in the current paper,
  - seems required.
- The almost disparate jargon of the two "camps" seems, however, to be a hindrance.

9.1.4. Analysis of Perdurants

- A study of perdurants, as detailed as that of our study of endurants, ought be carried out.
- One difficulty, as we see it, is the choice of formalisms:
  - whereas the basic formalisms for the expression of endurants and their qualities was type theory and simple functions and predicates,
  - there is no such simple set of formal constructs that can "carry" the expression of behaviours.
    - Besides the textual CSP, [43], there is graphic notations of
    - Petri Nets, [52],
    - Message Sequence Charts, [44],
    - State-charts, [42], and others.

9.1.5. Commensurate Discrete and Continuous Models

- Section 5.3.7 Slides 268–270 hinted at
  - co-extensive descriptions of discrete and continuous behaviours,
  - the former in, for example, RSL,
  - the latter in, typically, the calculus mathematics of partial different equations (PDEs).
  - The problem that arises in this situation is the following:
    - there will be, say variable identifiers, e.g., x, y, ..., z
    - which in the RSL formalisation has one set of meanings, but
    - which in the PDE "formalisation" has another set of meanings.

- Current formal specification languages[33] do not cope with continuity.

- Some research is going on.
- But to substantially cover, for example, the proper description of laminar and turbulent flows in networks (e.g., pipelines, Example 61 on Slide 269) requires more substantial results.

[33] `Alloy` [45],`Event B` [1],`RSL` [39], `VDM-SL` [30, 31, 37], `Z` [55], etc.

9.1.6. Interplay between Parts, Materials and Components

- Examples 49 on Slide 215, 50 on Slide 219, 51 on Slide 222 and 61 on Slide 269 revealed but a small fraction of the problems that may arise in connection with modeling the interplay between parts and materials.
- Subject to proper formal specification language and, for example PDE specification, we may expect more interesting
  - laws, as for example those of Examples 50 on Slide 219, 51 on Slide 222,
  - and even proof of these as if they were theorems.
- Formal specifications have focused on verifying properties of requirements and software designs.
- With co-extensive (i.e., commensurate) formal specifications of both discrete and continuous behaviours we may expect formal specifications to also serve as bases for predictions.

9.1.7. Dynamics

- There is a serious limitation in what can be modeled with the present approach.
  - Although we can model the dynamic introduction of new atomic or removal of existing parts, when members of a composite set of such parts,
  - we cannot model the dynamic introduction or removal of the processes corresponding to such parts.
  - Also we have not shown how to model global time.
  - And, although we can model spatial positions,
  - we have not shown how to model spatial locations.

- These deliberate omissions are due to the facts
  - that the description language, `RSL`, cannot model continuity and
  - that it cannot provide for arbitrary models of time.
- Here is an area worth studying.

9.1.8. Precise Descriptions of Manifest Domains

- The focus on the principles, techniques and tools of domain analysis & description has been such domains in which humans play an active rôle.
  - ⟐ Formal descriptions of domains may serve to
    - ⊚ prove properties of domains,
    - ⊚ in other words, to understand better these domains, and to
    - ⊚ validate requirements derived from such domain descriptions, and
    - ⊚ thereby to ensure that software derived from such requirements
      - ∗ is not only correct,
      - ∗ but also meet users expectations.

- Improved understanding of man-made domains —
  - without necessarily leading to new software

  — may serve to
  - improve the "business processes" of these domains,
  - make them more palatable for the human actors,
  - make them more efficient wrt. resource-usage.
- Descriptions of domains are descriptions of the syntax and semantics of the technical languages used in speaking about and in the domain.

- The domain analysis required for the design of programming languages is based on computability: mathematical logic and recursive function theory.
- The domain analysis required for "real-world" domains is not based on computability: that "world" is not computable.
- Requirements engineering based on domain descriptions is based on deriving computable subsets of refined domain descriptions.
- The classical theory and practice of programming language semantics and compiler development [6] and [9, Part VII (Chapters 16–19)] can now be further developed into a theory and practice for deriving general software from formal domain descriptions [12].
- Descriptions of domains are descriptions of the syntax and semantics of the technical languages used in speaking about and in the domain.

- The domain analysis required for the design of programming languages is based on computability: mathematical logic and recursive function theory.
- The domain analysis required for "real-world" domains is not based on computability: that "world" is not computable.
- Requirements engineering based on domain descriptions is based on deriving computable subsets of refined domain descriptions.
- The classical theory and practice of programming language semantics and compiler development [6] and [9, Part VII (Chapters 16–19)] can now be further developed into a theory and practice for deriving general software from formal domain descriptions [12].

- Physicists study 'Mother Nature', the world without us.
- Domain scientists study man-made part and material based universes with which we interact — the world within and without us.
- Classical engineering builds on laws of physics to design and construct
  - buildings,
  - chemical compounds,
  - machines and
  - E&E products.

- So far software engineers have not expressed software requirements on any precise description of the basis domain.
- This seminar strongly suggests such a possibility.
- Regardless:
  - it is interesting to also formally describe domains;
  - and, as shown, it can be done.

9.1.9. Towards Mathematical Models of Domain Analysis & Description

- There are two aspects to a precise description of the **domain analysis prompts** and **domain description prompts**.
  - There is that of describing
    - the individual prompts
    - as if they were "machine instructions"
    - for an albeit strange machine;
  - and there is that of describing
    - the interplay between prompts:
      - the sequencing of **domain description prompts**
      - as determined by the outcome of the **domain analysis prompts**.

- We have
  - described and formalised the latter in [25, Processes];
  - and we are in the midst of describing and formalising the former in [19, Prompts].

9.1.10. Laws of Descriptions: A Calculus of Prompts

- Laws of descriptions deal with the order and results of applying the domain analysis and description prompts.
- Some laws are covered in [17].
- It is expected that establishing formal models of the prompts, for example as outlined in [19, 25], will help identify such laws.

- The various description prompts apply to parts (etc.) of specified sorts (etc.) and to a “hidden state”.
  - The “hidden state” has two major elements:
    - the domain and
    - the evolving description texts.
  - An “execution” of a prompt potentially changes that “hidden state”.

- Let P, PA and PB be composite part sorts where PA and PB are derived from P.
- Let $\Re_i$, $\Re_j$, etc., be suitable functions which rename sort, type and attribute names.
- In a proper prompt calculus
  - we would expect
  - `observe_part_sorts_PA;observe_part_sorts_PB`,
  - when "executed" by one and the same domain engineer,
  - to yield the same "hidden state" as
  - `observe_part_sorts_PB`;$\Re_i$;`observe_part_sorts_PA`;$\Re_j$.

- Also one would expect
  - `observe_part_sorts_PA`;$\Re_i$;`observe_part_sorts_PA`;$\Re_j$.
  - to yield the same state as just
  - `observe_part_sorts_PA`
  - given suitable renaming functions.
- Well ? or does one really ?

- There are some assumptions that are made here.
- One pair of assumptions is
  - that the domain is fixed
  - and to one observer.
  - yields the same analysis and description results
  - no matter in which order prompts are "executed".
- Another assumption is that the domain engineer
  - does not get wiser as analysis and description progresses.
- If, as one can very well expect, the domain engineer does get wiser,
  - then former results may be discarded and
  - either replaced by newer analysis and descriptions
  - or prompts repeated.
- In such cases these laws do not hold.

9.1.11. Domains and Galois Connections

- Section 1.1.8 very briefly mentioned that formal concepts form Galois Connections.
- In the seminal [38] a careful study is made of this fact and beautiful examples show the implications for domains.
- It seems that our examples have all been too simple.
- They do not easily lead on to the "discovery" of "new" domain concepts from appropriate concept lattices.
- We refer to [29, Section 9].
- Further study need be done.

9.1.12. Laws of Domain Description Prompts

- Typically `observe_part_sorts` applies to a composite part, p:P, and yield descriptions of one or more part sorts: p_1:P_1,p_2:P_2,...,p_m:P_m.
- Let p_i:P_i,p_j:P_j,...,p_k:P_k (of these) be composite.
- Now `observe_part_sorts`(p_i) and `observe_part_sorts`(p_j), etc., can be applied and yield texts ***text***$_i$, respectively ***text***$_j$.
- A law of domain description prompts now expresses that the order in which the two or more observers is applied is immaterial, that is, they commute.
- In [17] we made an early exploration of such laws of domain description prompts.
- More work, hear also next, need be done.

9.1.13. Domain Theories:

- An ultimate goal of domain science & engineering is to prove properties of domains.
  - Well, maybe not properties of domains, but then at least properties of domain descriptions.
- If one can be convinced that a posited domain description indeed is a faithful description of a domain,
  - then proofs of properties of the domain description
  - are proofs of properties of that domain.
- Ultimately domain science & engineering must embrace such studies of *laws of domains*.
- Here is a fertile ground for zillions of Master and PhD theses!

Example 110 . A Law of Train Traffic at Stations:

- Let a transport net, n:N, be that of a railroad system.
  - Hubs are train stations.
  - Links are rail lines between stations.
  - Let a train timetable record train arrivals and train departures from stations.
  - And let such a timetable be modulo some time interval, say typically 24 hours.

- Now let us (idealistically) assume
  - that actual trains arrive at and depart from train stations according the train timetable and
  - that the train traffic includes all and only such trains as are listed in the train timetable.

- Now a law of train traffic expresses
  - *"Over the modulo time interval of a train timetable it is the case that*
    - *the number of trains arriving at a station*
    - *minus the number of trains ending their journey at that station*
    - *plus the number of trains starting their journey at that station*
    - *equals number of trains departing from that station."*

9.1.14. External Attributes

- More study is needed in order to clarify
  - the relations between the various external attributes
  - and control theory.

9.2. Requirements Topics

9.2.1. Domain Requirements Methodology

- Further principles, techniques and tools
- for the projection, instantiation, determination, extension and fitting operations.

9.2.2. Domain Requirements Operator Theory

- A model of the domain to domain-to-requirements operators:
- projection, instantiation, determination, extension and fitting. (Sect. 4).

9.2.3. Methodology for Interface Requirements

- Sect. 7.3 did not go into sufficient detail as to method principles, techniques and tools.

9.3. Final Words

Have a Happy & Fruitful R&D Career !

Dines Bjørner's MAP-i Lecture # 12

End of MAP-i Lecture # 12:
Discussion of Research Topics

Thursday, 28 May 2015: 16:45–17:30