Dines Bjørner's MAP-i Lecture # 7

Requirements – An Overview and Projection

Tuesday, 26 May 2015: 15:30–16:15

7. Requirements

- In Chapter 1. we introduced a method for analysing and describing manifest domains.
- In the next lectures of this PhD course
  - we show how to systematically,
  - but of course, not automatically,
  - "derive" requirements prescriptions from
  - domain descriptions.
- There are, as we see it, three kinds of requirements:
  - **domain requirements**,
  - **interface requirements** and
  - **machine requirements**.
- The **machine** is the hardware and software to be developed from the requirements ■

- **Domain requirements** are those requirements which can be expressed sôlely using technical terms of the domain ■
- **Interface requirements** are those requirements which can be expressed only using technical terms of both the domain and the machine ■
- **Machine requirements** are those requirements which can be expressed sôlely using technical terms of the machine ■

- We show principles, techniques and tools for "deriving"
  - domain requirements and
  - interface requirements.
- The domain requirements development focus on
  - projection,
  - instantiation,
  - determination,
  - extension and
  - fitting.

- These domain-to-requirements operators can be described briefly:
  - **projection** removes such descriptions which are to be omitted for consideration in the requirements,
  - **instantiation** mandates specific mereologies,
  - **determination** specifies less non–determinism,
  - **extension** extends the evolving requirements prescription with further domain description aspects and
  - **fitting** resolves "loose ends" as they may have emerged during the domain-to-requirements operations.

7.1. Introduction

Definition 18 . Requirements (I): *By a **requirements** we understand (cf. IEEE Standard 610.12):*

- *"A condition or capability needed by a user to solve a problem or achieve an objective"* ■

7.1.1. General Considerations

- The objective of requirements engineering is to create a **requirements prescription**:
  - A **requirements prescription** specifies externally observable properties of endurants and perdurants: functions, events and behaviours of **the machine** such as the requirements stake-holders wish them to be ■
  - The **machine** is what is required: that is, the **hardware** and **software** that is to be designed and which are to satisfy the requirements ■

- A *requirements prescription* thus (**putatively**) expresses what there should be.
- A requirements prescription expresses nothing about the design of the possibly desired (required) software.
- We shall show how a major part of a requirements prescription can be “derived” from “its” prerequisite domain description.

Rule 1 The "Golden Rule" of Requirements Engineering: *Prescribe only those requirements that can be objectively shown to hold for the designed software* ■

- "Objectively shown" means that the designed software can
  - either be tested,
  - or be model checked,
  - or be proved (verified),
- to satisfy the requirements.

Rule 2 An "Ideal Rule" of Requirements Engineering: *When prescribing (including formalising) requirements, also formulate tests and properties for model checking and theorems whose actualisation should show adherence to the requirements* ■

- The rule is labelled "ideal" since such precautions will not be shown in this seminar.
- The rule is clear.
- It is a question for proper management to see that it is adhered to.

Rule 3 Requirements Adequacy: *Make sure that requirements cover what users expect* ■

- That is,
  - do not express a requirement for which you have no users,
  - but make sure that all users' requirements are represented or somehow accommodated.
- In other words:
  - the requirements gathering process needs to be like an extremely "fine-meshed net":
  - One must make sure that all possible stake-holders have been involved in the requirements acquisition process,
  - and that possible conflicts and other inconsistencies have been obviated.

Rule 4 Requirements Implementability: *Make sure that requirements are implementable* ■

- That is, do not express a requirement for which you have no assurance that it can be implemented.
- In other words,
  - although the requirements phase is not a design phase,
  - one must tacitly assume, perhaps even indicate, somehow, that an implementation is possible.
- But the requirements in and by themselves, stay short of expressing such designs.

Rule 5 Requirements Verifiability and Validability: *Make sure that requirements are verifiable and can be validated* ■

- That is, do not express a requirement for which you have no assurance that it can be verified and validated.
- In other words,
  - once a first-level software design has been proposed,
  - one must show that it satisfies the requirements.
- Thus specific parts of even abstract software designs are usually provided with references to specific parts of the requirements that they are (thus) claimed to implement.

Definition 19 . Requirements (II): *By* **requirements** *we shall understand a document which prescribes desired properties of a machine:*

- *(i) what endurants the machine shall "maintain", and*
- *what the machine shall (must; not should) offer of*
  - *(ii) functions and of*
  - *(iii) behaviours*
- *(iv) while also expressing which events the machine shall "handle"* ■

- By a machine that "maintains" endurants we shall mean:
  - a machine which, "between" users' use of that machine,
  - "keeps" the data that represents these entities.
- From earlier we repeat:

Definition 20 . Machine: *By machine we shall understand a, or the, combination of hardware and software that is the target for, or result of the required computing systems development* ■

- So this, then, is a main objective of requirements development:
- to start towards the design of the hardware + software for the computing system.

Definition 21 . Requirements (III): *To specify the machine* ■

- When we express requirements and wish to "convert" such requirements to a realisation, i.e., an implementation, then we find
  - that some requirements (parts) imply certain properties to hold of the hardware on which the software to be developed is to "run",
  - and, obviously, that remaining — probably the larger parts of the — requirements imply certain properties to hold of that software.
- So we find
  - that although we may believe that our job is software engineering,
- important parts of our job are to also "design the machine"!

7.1.2. Four Stages of Requirements Development

- We shall unravel requirements in four stages — the first three stages are sketchy (and thus informal) while the last stage
  - is systematic,
  - mandates both strict narrative,
  - and formal descriptions, and
  - is "derivable" from the domain description.
- The four stages are:
  - the *problem/objective* sketch,
  - the narrative system requirements sketch,
  - the narrative user requirements sketch, and
  - the systematic narrative and formal functional requirements prescription.

7.1.2.1. Problem and/or Objective Sketch

Definition 22 . Problem/Objective Sketch: *By a* **problem/objective sketch** *we understand*

- *a narrative which emphasises*
- *what the problem or objectie is*
- *and thereby names its main concepts* ■

Example 66 . The Problem/Objective Requirements: A Sketch:

- The objective is to create a **road-pricing product.**
  - By a road-pricing product
    - we shall understand an information technology-based system
    - containing computers and communications equipment and software
    - that enables the recording of *vehicle* movements
    - within a well-delineated *road net*
    - and thus enables
      - the *owner* of the road net
      - to charge
      - the *owner* of the vehciles
      - *fees* for the usage of that road net

7.1.2.2. Systems Requirements

Definition 23 . System Requirements: *By a* **system requirements narrative** *we understand*

- *a narrative which emphasises*
- *the overall hardware and software*
- *system components* ■

Example 67 . The Road-pricing System Requirements: A Narrative:

- The requirements are based on the following a-priori given constellation of system components:
  - there is assumed a GNSS: a `Global Navigation Satellite System`;
  - there are specially equipped *vehicles*;
  - there is a well-delineated road net called a *toll-road* net with specially equipped toll-gates with *barrier*s which afford (only the specially equipped) vehicles to enter into and exit from the toll-road net; and
  - there is a [road-pricing] *calculator*.

- These four system components are required to behave and interact as follows:
  - The `GNSS` is assumed to continuously offer vehicles timed information about their global positions;
  - *vehicles* shall contain a `GNSS receiver` which based on the global position information shall regularly calculate their timed local position and offer this to the *calculator* — while otherwise cruising the general road net as well as the toll-road net, the latter while carefully moving through toll-gate barriers;
  - *toll-road gates* shall register the identity of vehicles entering and exiting the toll-road and offer this information to the calculator; and
  - the *calculator* shall accept all messages from vehicles and gates and use this information to record the movements of vehicles and bill these whenever they exit the toll-road.

- The requirements are therefore to include requirements to
  - the `GNSS` radio telecommunications equipment,
  - the vehicle `GNSS receiver` equipment,
  - the vehicle software,
  - the toll-gate in and out sensor equipment,
  - the electro-mechanical toll-gate barrier equipment,
  - the toll-gate barrier actuator equipment,
  - the toll-gate software,
  - the actuator software, and
  - the communications

- It is in this sense that the requirements are for an information technology-based system
  - of both software and
  - hardware —
    - not just hard computer and communications equipment,
    - but also movement sensors
    - and electro-mechanical "gear"

7.1.2.3. User and External Equipment Requirements

Definition 24 . User and External Equipment Requirements: *By a* **user and external equipment requirements narrative** *we understand*

- *a narrative which emphasises*
  - *the human user and*
  - *external equipment*

  interfaces
- *to the system components* ■

Example 68 . The Road-pricing User and External Equipment Requirements: Narrative:

- The human users of the road-pricing system are
  - vehicle drivers,
  - toll-gate sensor, actuator and barrier service staff, and
  - the road-pricing service calculator staff.
- The external equipment are
  - the GNSS satellites
  - and the telecommunications equipment
    - which enables communication between
    - the GNSS satellite sand vehicles ,
    - vehicles and the road-pricing calculator,
    - toll-gates and the road-pricing calculator and
    - the road-pricing calculator and vehicles (for billing),
  - We defer expression of
    - human user and
    - external equipment requirements

    till our treatment of relevant functional requirements

7.1.2.4. Functional Requirements

Definition 25 . Functional Requirements: *By* **functional requirements** *we understand precise prescriptions of*

- *the endurants*
- *and perdurants*

of the system components ■

- There are, as we see it, three kinds of requirements:
  - domain requirements,
  - interface requirements and
  - machine requirements

- **Domain requirements** are those requirements which can be expressed sôlely using technical terms of the domain ■
- **Interface requirements** are those requirements which can be expressed only using technical terms of both the domain and the machine ■
- **Machine requirements** are those requirements which can be expressed sôlely using technical terms of the machine ■

7.2. Domain Requirements

Definition 26 . Domain Requirements Prescription: *A* **domain requirements prescription**

- *is that subset of the requirements prescription*
- *which can be expressed sôlely using terms from the domain description* ■
- To determine a relevant subset all we need is collaboration with requirements stake-holders.

- Experimental evidence,
  - in the form of example developments
    - of requirements prescriptions
    - from domain descriptions,

    appears to show
  - that one can formulate techniques for such developments
  - around a few domain description to requirements prescription operations.
  - We suggest these:
    - projection,
    - instantiation,
    - determination,
    - extension,
    - fitting

    and, perhaps, other domain description to requirements prescription operations.

7.2.1. Domain Projection

Definition 27 . Domain Projection: *By a* **domain projection** *we mean*

- *a subset of the domain description,*
- *one which leaves out all those*
  - *endurants:*
    - *parts,*
    - *materials and*
    - *components,*

    as well as
  - *perdurants:*
    - *functions,*
    - *events and*
    - *behaviours*

  that the stake-holders do not wish represented by the machine.
- *The resulting document is a* ***partial domain requirements prescription*** ■

- In determining an appropriate subset
  - the requirements engineer must secure
  - that the final prescription
  - is complete and consistent — that is,
    - that there are no “dangling references”,
    - i.e., that all entities that are referred to
    - are all properly defined.

7.2.1.1. Domain Projection — Narrative

- We now start on a series of examples
- that illustrate domain requirements development.

Example 69 . Domain Requirements. Projection A Narrative Sketch:

- We require that the Road-pricing IT, computing & communications system shall embody the following domain entities, in one form or another:
  - the net,
    - its links and hubs,
    - and their properties
      (unique identifiers, mereologies and attributes),
  - the vehicles, as endurants,
    - as endurants,
    - and the general vehicle behaviour, i.e., the vehicle signature.

- To formalise this we copy the domain description, $\Delta_{\triangle}$,
- From that domain description we remove all mention of
  - the link insertion and removal functions,
  - the link disappearance event,
  - the vehicle behaviour, and
  - the monitor
- to obtain the $\Delta_{\mathcal{P}}$ version of the domain requirements prescription.[25]

[25] Restrictions of the net to the toll road nets, hinted at earlier, will follow in the next domain requirements steps.

7.2.1.2. Domain Projection — Formalisation

- The requirements prescription hinges, crucially,
  - not only on a systematic narrative of all the
    - projected,
    - instantiated,
    - determinated,
    - extended and
    - fitted

    specifications,
  - but also on their formalisation.
- In the series of domain projection examples following below we, regretfully, omit the narrative texts.
  - In bringing the formal texts
    we keep the item numbering from Sect. 2.,
  - where you can find the associated narrative texts.

Example 70 . Domain Requirements. Projection Root Sorts:

type

112. $\Delta_{\mathcal{P}}$

112a.. $\mathsf{N}_{\mathcal{P}}$

112b.. $\mathsf{F}_{\mathcal{P}}$

value

112a.. **obs_part**_$\mathsf{N}_{\mathcal{P}}$: $\Delta_{\mathcal{P}} \rightarrow \mathsf{N}_{\mathcal{P}}$

112b.. **obs_part**_$\mathsf{F}_{\mathcal{P}}$: $\Delta_{\mathcal{P}} \rightarrow \mathsf{F}_{\mathcal{P}}$

type

113a.. $\mathsf{HA}_{\mathcal{P}}$

113b.. $\mathsf{LA}_{\mathcal{P}}$

value

113a.. **obs_part**_HA: $\mathsf{N}_{\mathcal{P}} \rightarrow \mathsf{HA}$

113b.. **obs_part**_LA: $\mathsf{N}_{\mathcal{P}} \rightarrow \mathsf{LA}$

Example 71 . Domain Requirements. Projection Sub-domain Sorts and Types:

type

114. $\mathsf{H}_{\mathcal{P}}$, $\mathsf{HS}_{\mathcal{P}} = \mathsf{H}_{\mathcal{P}}$**-set**

115. $\mathsf{L}_{\mathcal{P}}$, $\mathsf{LS}_{\mathcal{P}} = \mathsf{L}_{\mathcal{P}}$**-set**

116. $\mathsf{V}_{\mathcal{P}}$, $\mathsf{VS}_{\mathcal{P}} = \mathsf{V}_{\mathcal{P}}$**-set**

value

114. **obs_part**_$\mathsf{HS}_{\mathcal{P}}$: $\mathsf{HA}_{\mathcal{P}} \rightarrow \mathsf{HS}_{\mathcal{P}}$

115. **obs_part**_$\mathsf{LS}_{\mathcal{P}}$: $\mathsf{LA}_{\mathcal{P}} \rightarrow \mathsf{LS}_{\mathcal{P}}$

116. **obs_part**_$\mathsf{VS}_{\mathcal{P}}$: $\mathsf{F}_{\mathcal{P}} \rightarrow \mathsf{VS}_{\mathcal{P}}$

117a.. links: $\Delta_{\mathcal{P}} \rightarrow$ L-**set**

117a.. links($\delta_{\mathcal{P}}$) $\equiv$ **obs_part**_$\mathsf{LS}_{\mathcal{R}}$(**obs_part**_$\mathsf{LA}_{\mathcal{R}}$($\delta_{\mathcal{R}}$))

117b.. hubs: $\Delta_{\mathcal{P}} \rightarrow$ H-**set**

117b.. hubs($\delta_{\mathcal{P}}$) $\equiv$ **obs_part**_$\mathsf{HS}_{\mathcal{P}}$(**obs_part**_$\mathsf{HA}_{\mathcal{P}}$($\delta_{\mathcal{P}}$))

Example 72 . Domain Requirements. Projection Unique Identifications:

type
118a.. HI, LI, VI, MI
value
118c.. **uid**_HI: $H_{\mathcal{P}} \rightarrow$ HI
118c.. **uid**_LI: $L_{\mathcal{P}} \rightarrow$ LI
118c.. **uid**_VI: $V_{\mathcal{P}} \rightarrow$ VI
118c.. **uid**_MI: $M_{\mathcal{P}} \rightarrow$ MI
axiom
118b.. $HI \bigcap LI = \emptyset$, $HI \bigcap VI = \emptyset$, $HI \bigcap MI = \emptyset$,
118b.. $LI \bigcap VI = \emptyset$, $LI \bigcap MI = \emptyset$, $VI \bigcap MI = \emptyset$

Example 73 . Domain Requirements. Projection Road Net Mereology:

value
120. **obs_mereo**_$H_{\mathcal{P}}$: $H_{\mathcal{P}} \rightarrow$ LI-**set**
121. **obs_mereo**_$L_{\mathcal{P}}$: $L_{\mathcal{P}} \rightarrow$ HI-**set**
121. **axiom** $\forall$ l:$L_{\mathcal{P}}$ • **card** **obs_mereo**_$L_{\mathcal{P}}$(l)=2
122. **obs_mereo**_$V_{\mathcal{P}}$: $V_{\mathcal{P}} \rightarrow$ MI
123. **obs_mereo**_$M_{\mathcal{P}}$: $M_{\mathcal{P}} \rightarrow$ VI-**set**

axiom
124. $\forall$ $\delta_{\mathcal{P}}$:$\Delta_{\mathcal{P}}$, hs:HS•hs=hubs(δ), ls:LS•ls=links($\delta_{\mathcal{P}}$) $\Rightarrow$
124. $\forall$ h:$H_{\mathcal{P}}$•h $\in$ hs $\Rightarrow$
124. **obs_mereo**_$H_{\mathcal{P}}$(h)$\subseteq$xtr_his($\delta_{\mathcal{P}}$) $\wedge$
125. $\forall$ l:$L_{\mathcal{P}}$•l $\in$ ls •
124. **obs_mereo**_$L_{\mathcal{P}}$(l)$\subseteq$xtr_lis($\delta_{\mathcal{P}}$) $\wedge$
126a.. **let** f:$F_{\mathcal{P}}$•f=**obs_part**_$F_{\mathcal{P}}$($\delta_{\mathcal{P}}$) $\Rightarrow$
126a.. vs:$VS_{\mathcal{P}}$•vs=**obs_part**_$VS_{\mathcal{P}}$(f) **in**
126a.. $\forall$ v:$V_{\mathcal{P}}$•v $\in$ vs $\Rightarrow$
126a.. **uid**_$V_{\mathcal{P}}$(v) $\in$ **obs_mereo**_$M_{\mathcal{P}}$(m) $\wedge$
126b.. **obs_mereo**_$M_{\mathcal{P}}$(m)
126b.. = {**uid**_$V_{\mathcal{P}}$(v)|v:V•v $\in$ vs}
126b.. **end**

Example 74 . Domain Requirements. Projection Attributes of Hubs:

type
127a.. $\mathsf{H}\Sigma_{\mathcal{P}} = (\mathsf{LI}\times\mathsf{LI})$**-set**t
127b.. $\mathsf{H}\Omega_{\mathcal{P}} = \mathsf{H}\Sigma_{\mathcal{P}}$**-set**
value
127a.. **attr**_$\mathsf{H}\Sigma_{\mathcal{P}}$: $\mathsf{H}_{\mathcal{P}} \rightarrow \mathsf{H}\Sigma_{\mathcal{P}}$
127b.. **attr**_$\mathsf{H}\Omega_{\mathcal{P}}$: $\mathsf{H}_{\mathcal{P}} \rightarrow \mathsf{H}\Omega_{\mathcal{P}}$
type
129. HGCL
value
129. **attr**_HGCL: H $\rightarrow$ HGCL
axiom
128. $\forall\ \delta_{\mathcal{P}}:\Delta_{\mathcal{P}}$,
128. **let** hs = hubs($\delta_{\mathcal{P}}$) **in**
128. $\forall$ h:$\mathsf{H}_{\mathcal{P}}$ $\cdot$ h $\in$ hs $\cdot$
128a.. xtr_lis(h)$\subseteq$xtr_lis($\delta_{\mathcal{P}}$)
128b.. $\wedge$ **attr**_$\Sigma_{\mathcal{P}}$(h) $\in$ **attr**_$\Omega_{\mathcal{P}}$(h)
128. **end**

Example 75 . Domain Requirements. Projection Attributes of Links:

type

131. LEN

132. LGCL

133a.. $L\Sigma_{\mathcal{P}}$ = (HI×HI)-**set**

133b.. $L\Omega_{\mathcal{P}}$ = $L\Sigma_{\mathcal{P}}$-**set**

value

131. **attr**_LEN: $L_{\mathcal{P}} \rightarrow$ LEN

132. **attr**_LGCL: $L_{\mathcal{P}} \rightarrow$ LGCL

133a.. **attr**_$L\Sigma_{\mathcal{P}}$: $L_{\mathcal{P}} \rightarrow L\Sigma_{\mathcal{P}}$

133b.. **attr**_$L\Omega_{\mathcal{P}}$: $L_{\mathcal{P}} \rightarrow L\Omega_{\mathcal{P}}$

axiom

133a..– 133b. on Slide 324.

Example 76 . Domain Requirements. Projection Behaviour:

Global Values

value

146. $\delta_{\mathcal{P}}$:$\Delta_{\mathcal{P}}$,

147. n:$N_{\mathcal{P}}$ = **obs_part**_$N_{\mathcal{P}}(\delta_{\mathcal{P}})$,

147. ls:$L_{\mathcal{P}}$**-set** = links($\delta_{\mathcal{P}}$),

147. hs:$H_{\mathcal{P}}$**-set** = hubs($\delta_{\mathcal{P}}$),

147. lis:LI**-set** = xtr_lis($\delta_{\mathcal{P}}$),

147. his:HI**-set** = xtr_his($\delta_{\mathcal{P}}$)

Behaviour Signatures

value

153. $trs_{\mathcal{P}}$: **Unit** → **Unit**

154. $veh_{\mathcal{P}}$: VI×MI×ATTR → ... **Unit**

The System Behaviour

value

156a.. $trs_{\mathcal{P}}$()=‖{$veh_{\mathcal{P}}$(**uid**_VI(v),**obs_mereo**_V(v),**attr**_ATTRS(v)) | v:$V_{\mathcal{P}}$•v ∈ vs}

7.2.1.3. A Projection Operator

- Domain projection thus take a domain description, $\mathcal{D}$, and yields a projected requirements prescription, ,$\mathcal{R}_{\mathcal{P}}$.
- ⟐ **type** projection: $\mathcal{D} \rightarrow \mathcal{R}_{\mathcal{P}}$.
- Semantically
  - ⟐ $\mathcal{D}$ denotes a possibly infinite set of meanings, say $\mathbb{D}$ and
  - ⟐ $\mathcal{R}_{\mathcal{P}}$ denotes a possibly infinite set of meanings, say $\mathbb{R}_{\mathbb{P}}$,
  - ⟐ such that some relation $\mathbb{R}_{\mathbb{P}} \sqsubseteq \mathbb{D}$ is satisfied.

Dines Bjørner's MAP-i Lecture # 7

End of MAP-i Lecture # 7:
Requirements – An Overview and Projection

Tuesday, 26 May 2015: 15:30–16:15