Start of Lecture 5: REQUIREMENTS – up to and incl. Determination

4. An Ontology of Requirements Constructions

Definition: Requirements. *A condition or capability needed by a user to solve a problem or achieve an objective* [IEEEStd.610.12].

Definition: Machine. *By the machine we understand the hardware plus software that implements some requirements, i.e., a computing system.*

Definition: Requirements Prescription. *By a requirements prescription we mean just that: the prescription of some requirements. Sometimes, by requirements prescription, we mean a relatively complete and consistent specification of all requirements, and sometimes just a requirements prescription unit.*

Definition: Requirements Engineering. *The engineering of the development of a requirements prescription,*

- *from identification of requirements stakeholders,*
- *via requirements acquisition,*
- *requirements analysis, and*
- *requirements prescription to*
- *requirements validation and*
- *requirements verification.*

- We shall just focus on *requirements prescription*,
- that is, the modelling of *requirements*.

4.1. Business Process Reengineering

Definition: Business Process. *By a business process we shall understand a behaviour of an enterprise, a business, an institution, a factory. Thus a business process reflects the ways in which a business conducts its affairs, and is a facet of the domain. Other facets of an enterprise are those of its intrinsics, management and organisation (a facet closely related, of course, to business processes), support technology, rules and regulations, and human behaviour.*

Definition: Business Process Engineering. *By business process engineering we shall understand the design, the determination, of business processes. In doing business process engineering one is basically designing, i.e., prescribing entirely new business processes.*

Definition: Business Process Reengineering. *By business process reengineering we shall understand the redesign, the change, of business processes. In doing business process reengineering one is basically carrying out change management.*

4.1.1. Goals Versus Requirements

- Whereas
  - a domain description presents a domain **as it is**,
  - a requirements prescription presents a domain **as it would be** if some required **machine** was implemented (from these requirements).
- The **machine** is the **hardware** plus **software** to be designed from the requirements.
- That is, the *machine* is what the requirements are about.

- We distinguish between three kinds of requirements:
  - the **domain requirements** are those requirements which can be expressed solely using terms of the domain;
  - the **machine requirements** are those requirements which can be expressed solely using terms of the machine, and
  - the **interface requirements** are those requirements which must use terms from both the domain and the machine in order to be expressed.

- We make a distinction between **goals** and **requirements.**
- Goals are what we
  - expect satisfied by the software
  - implemented from the requirements.
- But goals could also be
  - of the system
  - for which the software is required.
- First we exemplify the latter, then the former.

4.1.1.1. Goals of a Toll Road System

- A goal for a toll road system may be
  - to decrease the travel time between certain hubs and
  - to lower the number of traffic accidents between certain hubs,

4.1.1.2. Goals of Toll Road System Software

- The goal of the toll road system software is to help automate
  - the recording of vehicles entering, passing and leaving the toll road system
  - and collecting the fees for doing so.
- Goals are usually expressed in terms of properties.
- Requirements can then be proved to satisfy the $\mathcal{G}$oals: $\mathcal{D}, \mathcal{R} \models \mathcal{G}$.

4.1.1.3. Arguing Goal-satisfaction of a Toll Road System

- By endowing links and hubs with average traversal times for both ordinary road and for toll road links and hubs
  - one can calculate traversal times between hubs
  - and thus argue that the toll road system satisfies "quicker" traversal times.
- By endowing links and hubs with traffic accident statistics (real, respectively estimated)
  - for both ordinary road and for toll road links and hubs
  - one can calculate estimated traffic accident statistics between all hubs
  - and thus argue that the combined ordinary road plus toll road system satisfies lower traffic fatalities.

4.1.1.4. Arguing Goal-satisfaction of Toll Road System Software

- By recording
  - tickets issued and collected at toll booths and
  - toll road hubs and links entered and left
  - as per the requirements specification brought in forthcoming examples (Sects. –),
- we can eventually argue that
  - the requirements of the forthcoming examples (Sects. –)
  - help satisfy the goal of the example **??** on page ??.

- We shall assume that the (goal and) requirements engineer elicit both $\mathcal{G}$oals and $\mathcal{R}$equirements from requirements stake-holders.
- $\mathcal{D}, \mathcal{R} \models \mathcal{G}$
  - The $\mathcal{G}$oals can be argued to hold
  - by reasoning over the $\mathcal{R}$equirements
  - and the $\mathcal{D}$omain.
- But we shall focus only on
  - domain and
  - interface

  requirements such as "derived" from domain descriptions.

4.1.2. Reengineered Nets

- The nets defined in Lecture 3 could be of any topology.
  - They could consist of two or more nets that were not linked to one another;
  - they could consist of connected nets or nets that were acyclic; etc.;
  - and the nets were not specifically road, rail, sea lane or air lane nets.

- We shall now consider a special kind of road nets: basically the road nets we have in mind
  - are linear sequences of pairs of links of opposite direction link "states",
  - where these links, let us call them toll road links, are connected to toll road hubs;
  - where, in addition, these toll road hubs are linked, via toll plazas (i.e., "special" hubs) to toll road hubs
  - by means of on/off links.

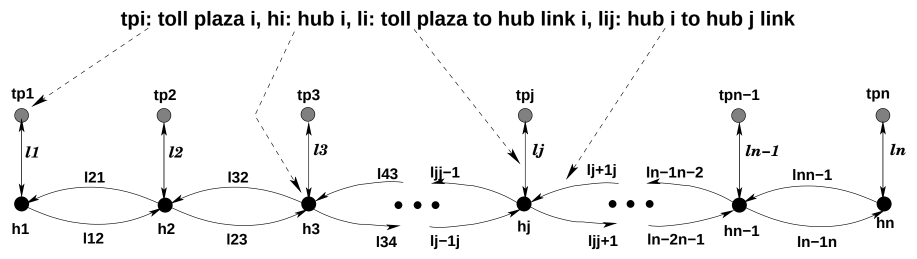

Figure 3: A Toll Road System

- We do not consider the general nets that are (possibly) connected to the toll plazas.
- The pragmatics behind these nets is the following:
  - Drivers enter and leave the toll road nets at toll road plazas;
  - collect tickets from toll road plaza ticket-issuing booths when entering the toll road net and
  - present these at toll road plaza ticket-collection booths and pay according to some function of the time and length (from entry to exit plaza) driven on the toll road net when leaving the net;
  - drivers are otherwise free to “circle” the toll road net as they see fit:
    - ∗ multiple times “up and down” the net,
    - ∗ circling toll road hubs,
    - ∗ etc.

- Our sketch centers around a toll road net with toll booth plazas.
- The BPR focuses
  - first on entities, actions, events and behaviours,
  - then on the six domain facets.

125. **Re-engineered Entities:**
- We shall focus on a linear sequence of toll road intersections (i.e., hubs) connected by pairs of one-way (opposite direction) toll roads (i.e., links).
- Each toll road intersection is connected by a two way road to a toll plaza.
- Each toll plaza contains a pair of sets of entry and exit toll booths.
- (Sect. brings more details.)

126. **Re-engineered Actions:**

- Cars enter and leave the toll road net through one of the toll plazas.
- Upon entering, car drivers receive, from the entry booth, a plastic/paper/electronic ticket which they place in a special holder in the front window.
- Cars arriving at intermediate toll road intersections choose, on their own, to turn either "up" the toll road or "down" the toll road — with that choice being registered by the electronic ticket.
- Cars arriving at a toll road intersection may choose to "circle" around that intersection one or more times — with that choice being registered by the electronic ticket.
- Upon leaving, car drivers "return" their electronic ticket to the exit booth and pay the amount "asked" for.

127. **Re-engineered Events:**

- A car entering the toll road net at a toll both plaza entry booth constitutes an event.
- A car leaving the toll road net at a toll both plaza entry booth constitutes an event.
- A car entering a toll road hub constitutes an event.
- A car entering a toll road link constitutes an event.

128. **Re-engineered Behaviours:**

- The journey of a car,
  - from entering the toll road net at a toll booth plaza,
  - via repeated visits to toll road intersections
  - interleaved with repeated visits to toll road links
  - to leaving the toll road net at a toll booth plaza,

  constitutes a behaviour — with
  - receipt of tickets,
  - return of tickets and
  - payment of fees

  being part of these behaviours.
- Notice that a toll road visitor is allowed to cruise "up" and "down" the linear toll road net – while (probably) paying for that pleasure (through the recordings of "repeated" hub and link entries).

129. **Re-engineered Intrinsics:**

- Toll plazas and abstracted booths are added to domain intrinsics.

130. **Re-engineered Support Technologies:**

- There is a definite need for domain-describing the failure-prone toll plaza entry and exit booths.

131. **Re-engineered Rules and Regulations:**

- Rules for entering and leaving toll booth entry and exit booths must be described as must related regulations.
- Rules and regulations for driving around the toll road net must be likewise be described.

132. **Re-engineered Scripts:**
   - No need.
133. **Re-engineered Management and Organisation:**
   - There is a definite need for domain describing
   - the management and possibly distributed organisation
   - of toll booth plazas.
134. **Re-engineered Human Behaviour:**
   - Humans, in this case car drivers, may not change their behaviour in the spectrum from diligent and accurate via sloppy and delinquent to outright traffic-law breaking – so we see no need for any “re-engineering”.

4.2. Domain Requirements

Definition: Domain Requirements. *By domain requirements we understand such requirements — save those of business process reengineering — which can be expressed sôlely by using professional terms of the domain.*

Definition: Domain Requirements Facet. *By domain requirements facets we understand such domain requirements that basically arise from either of the following operations on domain descriptions (cum requirements prescriptions): domain projection, domain determination, domain extension, domain instantiation and domain fitting.*

4.2.1. Projection

Definition: Projection. *By projection we shall here, in a somewhat narrow sense, mean a technique that applies to domain descriptions and yields requirements prescriptions. Basically projection “reduces” a domain description by “removing” (or, but rarely, hiding) entities, functions, events and behaviours from the domain description. If the domain description is an informal one, say in English, it may have expressed that certain entities, functions, events and behaviours might be in (some instantiations of) the domain. If not “projected away” the similar, i.e., informal requirements prescription will express that these entities, functions, events and behaviours shall be in the domain and hence will be in the environment of the machine being requirements prescribed.*

Keep

- N, H, L,
- obs_Hs,
- obs_Ls,
- HI, LI,
- obs_HI,
- obs_LI,
- obs_LIs,
- obs_HIs,
- PLAN, LHIM,
- wf_PLAN,
- ND, wf_ND,
- $L\Sigma$, $L\Omega$,
- obs_$L\Sigma$, obs_$L\Sigma$,
- $H\Sigma$, $H\Omega$,
- obs_$H\Sigma$, obs_$H\Sigma$,
- V, VI, VP,
- obs_VI, obs_VP,
- TF, T and
- wf_TF.

4.2.2. Instantiation

Definition: Instantiation. *'To represent (an abstraction) by a concrete instance'* [mw2004]. Domain instantiation is a *domain requirements facet*. It is an operation performed on a *domain description* (cum *requirements prescription*). Where, in a domain description certain *entities* and *function*s are left undefined, domain instantiation means that these entities or functions are now instantiated into constant *value*s.

- The following instantiation prescription only covers the static aspects of the toll road net, i.e., simple entities.
- That is, the states of hubs and links will first be dealt with in Sect. .

135. A toll road net (a subnet of a larger previously described net) consists of a pair: toll road links and toll road to plaza hubs and links.
   (a) The toll road links component is a linear sequence of one or more pairs of toll road links.
   (b) The toll road to plaza hubs and links component is a linear sequence of two or more triples of a plaza, a (plaza to toll road hub) link and a toll road hub.

   (c) The wellformedness of toll road nets are expressed next.
      i. The length of the toll road links sequence is one less than the length of the toll road to plaza hubs and links sequence. The idea is that the toll road links at position i connect the toll road hubs at positions i and $i+1$ of the toll road to plaza hubs and links sequence — i being the indexes of the toll road links sequence.
      ii. All links have distinct link identifiers.
      iii. All hubs and plazas have distinct hub identifiers.
      iv. From the links in the pairs of links, (l_i, l'_i), of position i in the toll road links component one observes exactly the same two element set of hub identifiers,
      v. and these are the identifiers of the hubs at positions i and $i+1$ of the toll road to plaza hubs and links sequence.
      vi. The plaza to toll road hub links are indeed connected to these plazas and hubs; and
      vii. the plaza and toll road hubs are connected only to the links as mentioned above.

   (d) A toll road plaza is like a hub, with an observable hub identifier (and equipped with ticket-issuing tool booths and ticket-collection and payment toll booths).

type
135. $\mathrm{TRN'} = \mathrm{TRLs} \times \mathrm{PHLs}$
135. $\mathrm{TRN} = \{| \mathrm{trn:TRN'} \cdot \mathrm{wf\_TRN(trn)} |\}$
135(a). $\mathrm{TRLs} = (\mathrm{L} \times \mathrm{L})^*$
135(b). $\mathrm{PHLs} = (\mathrm{PZ} \times \mathrm{L} \times \mathrm{H})^*$

```
value
135(c).        wf_TRN: TRN′ → Bool
135(c).        wf_TRN(trn:(trls,phls)) ≡
135((c))i.       len trls +1 = len phls ∧
135((c))ii.      card xtr_Hs(trn) = card xtr_HIs(trn) ∧
135((c))iii.     card xtr_Ls(trn) = card xtr_LIs(trn)
135((c))iv.      ∀ i:Nat•i ∈ inds trls ⇒
135((c))iv.        let (l,l′)=trsl(i),(p,l″,hi)=phls(i),(_,l‴,hj)=phls(i+1) in
135((c))iv.        obs_HIs(l) = obs_HIs(l′) =
135((c))v.         {obs_HI(hi),obs_HI(hj)}  ∧
135((c))vii.       case i of
135((c))vii.         1 → obs_LIs(hi) = xtr_LIs({l,l′,l″}),
135((c))vii.         len trsl − 1 → obs_LIs(hj) = xtr_LIs({l,l′,l‴}),
135((c))vii.         _ → let (l⁗,l⁗′)=trsl(i) in obs_LIs(hi)=xtr_LIs({l,l′,l″,l⁗,l⁗′}) end
135((c))vii.       end end ∧
135((c))vii.     ∀ i:Nat•i ∈ inds phls ⇒
135((c))vii.       let (p,l,h)=phls(i) in obs_HIs(l)=xtr_HIs({p,h}) ∧
135((c))vii.       obs_LIs(p) = {obs_LI(l)} end
```



```
type
135(d).   PZ
value
135(d).   obs_HI: PZ → HI

    xtr_Hs: TRN → H-set
    xtr_Hs(_,phls) ≡ {pz,h|(pz,l,h):(PZ×L×H)•(pz,l,h)∈ elems phls}
    xtr_Ls: TRN → L-set
    xtr_Ls(trls,phls) ≡
       {l,l′|l,l′:L•(l,l′)∈ elems trls} ∪ {l|(pz,l,h):(PZ×L×H)•(pz,l,h)∈ elems phls}

    xtr_HIs: TRN → HI-set,   xtr_HIs(trn) ≡ {obs_HI(h)|h:(H|PZ)•h ∈ xtr_Hs(trn)}
    xtr_LIs: TRN → LI-set,   xtr_LIs(trn) ≡ {obs_LI(l)|l:L•l ∈ xtr_Ls(trn)}
    xtr_HIs: H-set → HI-set,  xtr_HIs(hs) = {obs_LI(h)|h:H•h ∈ hs}
    xtr_LIs: L-set → LI-set,  xtr_LIs(ls) = {obs_LI(l)|l:L•l ∈ ls}
```


4.2.2.1. Abstraction: From Concrete Toll Road Nets to Abstract Nets

136. From concrete toll road nets, trn:TRN, one can abstract the nets, n:N, of Items 1–9.
    (a) the abstract net contains the hubs of the concrete net,
    (b) and the links likewise.

```
value
136.     abs_N: TRN → N
136.     abs_N(trn) as n
136(a).     obs_Hs(n) = xtr_Hs(trn) ∧
136(b).     obs_Ls(n) = xtr_Ls(trn)
```

137. One can prove the following theorem: If trn satisfies wf_TRN(trn) then abs_N(trn) satisfies Axioms 2–3 and 5–8 (Page 12).

```
137.  ∀ trn:TRN · wf_TRN(trn) ⊨ abs_N(trn) satisfies axioms 2.–3. ∧ axi
```


4.2.3. Determination

Definition: Determination. *Domain determination is a domain requirements facet. It is an operation performed on a domain description cum requirements prescription. Any nondeterminism expressed by either of these specifications which is not desirable for some required software design must be made deterministic (by this requirements engineer performed operation).*

We shall focus on making more specific the rather generically defined nets, hubs and links.

There are no traffic signals within the toll road net and pairs of toll road links are “one way, opposite direction” links.

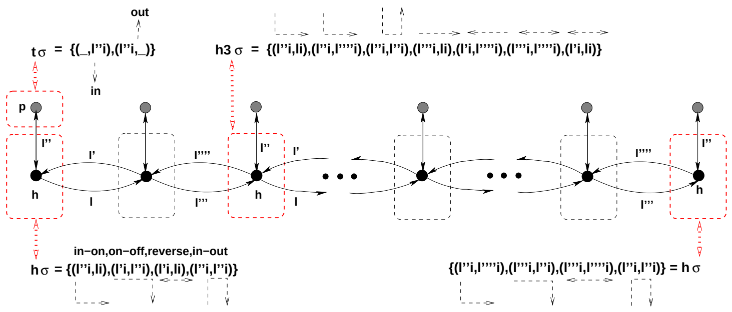

Figure 4: Four example hub states: plaza, end hubs, "middle" hub

138. Pairs of toll road links, l, l', connecting adjacent hubs hj, hk, of identifiers hj_i, hk_i, respectively, always and only allow traffic in opposite directions, that is, are always in respective states $\{(hj_i, hk_i)\}$ and $\{(hk_i, hj_i)\}$.
139. Hub, h, states, $h\sigma$, are constant and allow traffic onto connected links not closed for traffic in directions from hub h.
140. Plazas allow traffic only onto connected plaza to hub links of the toll road net. (Whatever other links, "outside" the toll road net, the plazas may be connected to is covered in the last line of the axiom below.)


```
axiom
  ∀ (trls,phls):TRN •
    ∀ i:Nat • i ∈ inds trls
      let (l,l′) = trls(i), (p,l″,h) = phls(i) in
      case i of
        1 → obs_HΣ(h) = {(obs_LI(l″),obs_LI(l)),
                         (obs_LI(l′),obs_LI(l″)),(obs_LI(l′),obs_LI(l)),
                         (obs_LI(l″),obs_LI(l″))},
        _ → let (l‴,l⁗) = trls(i−1) in
            obs_HΣ(h) = {(obs_LI(l″),obs_LI(l)),
                         (obs_LI(l″),obs_LI(l⁗)),(obs_LI(l″),obs_LI(l″)),
                         (obs_LI(l″),obs_LI(l″)),(obs_LI(l‴),obs_LI(l)),
                         (obs_LI(l′),obs_LI(l⁗)),(obs_LI(l‴),obs_LI(l⁗)),
                         (obs_LI(l′),obs_LI(l))} end end end ∧
    let (l‴,l⁗) = trls(len trsl), (p,l″,h) = phls(1 + len trsl) in
    obs_HΣ(h) = {(obs_LI(l″),obs_LI(l⁗)),
                 (obs_LI(l‴),obs_LI(l″)),(obs_LI(l‴),obs_LI(l⁗)),
                 (obs_LI(l″),obs_LI(l″))} end ∧
    ∀ (p,l″,_):(PZ×L×H)•(p,l″,_) ∈ elems phls ⇒
      let lis = obs_LIs(p) assert: obs_LI(l″) ∈ lis in
      obs_HΣ(p) = {(li,obs_LI(l″)),(obs_LI(l″),li)|li:LI•li ∈ lis} end
```


- In the last line of the wellformedness axiom above we express that the plaza maybe connected to many links not in the toll road net and that the plaza is open for all traffic from these into the net (via l″), from l″ to these and that traffic may even reverse at the plazas, that is, decide to not enter the toll road net after having just visited the plaza.

End of Lecture 5: REQUIREMENTS – up to and incl. Determination