Case Study: Multi-storey car park

You are acting as a requirements engineering consultant to a client who wants to automate his existing multi-storey car park with time-stamped ticket-issuing machines, payment machines, closed-circuit television cameras to deter both theft and non-payment, and automatic barriers operated by validated (paid-up) tickets.
In concrete terms

• The automation consists in managing inputs and outputs; information useful for car park users shall be continuously displayed.

• The automation consists also to guarantee a high level of security. We will focus on two specific points: any stay shall be guaranteed to be fully paid and theft of cars shall be prevented as far as possible.
Case study statement (details)

Regulation prescribes the fulfilment of security constraints officially identified in several documents. For instance, according to a fire security regulation,

- the total authorized weight of vehicles admitted in covered car parks shall not exceed 3.5 tons
- In case of fire, all barriers shall be open (no opening mode is prescribed).

In the following, we suppose that the car park to be automated complies with the facility structure regulation and that the application to be developed shall be responsible for a part of the regulatory security.
First Goal Identification

You are acting as requirements engineering consultant to a client who wants to automate his existing multi-storey car park with time-stamped ticket-issuing machines, payment machines, closed-circuit television cameras to deter both theft and non-payment, and automatic barriers operated by validated (paid-up) tickets.

- Candidate goals: the problem to solve
- Technical elements: constraints on the solution
- Typical: lots of elts related to a solution
- RE objective: get a deeper insight into the PROBLEM
First questions

• Deter thefts... of what?
  – vehicles?
  – in vehicles?
  – at payment machines?
  – mugging?

• Automated barrier: strange position in the statement

• Automate car park: does not tell the why; project enveloppe → discarded for a while
First Goal Identification

In concrete terms

- The automation consists in managing/controlling inputs and outputs; information useful for car park users shall also be continuously displayed.
- The automation consists also to guarantee a high level of security. We will focus on two specific points: any stay shall be guaranteed to be fully paid and theft of cars shall be prevented as far as possible.

- Candidate goals:
  - Managing/controlling inputs & outputs: a means to deter thefts & non-payment
  - Information displayed for users: which kind of information is relevant? → user goals?
  - Stay fully paid. Provides a definition for « validated tickets » (paid-up tickets)
  - Theft of cars. Makes the initial statement more precise.
First Goal identification

Regulation prescribes the fulfillment of security constraints officially identified in several documents. For instance, according to a fire security regulation,

- the total authorized weight of any vehicle admitted in covered car parks shall not exceed 3.5 tons
- In case of fire, all barriers shall be open (no opening mode is prescribed).

In the following, we suppose that the car park to be automated complies with the facility structure regulation and that the application to be developed shall be responsible for a part of the regulatory security.

- Security constraints = a specific category of mandatory goals; introduces constraints on ways behavioural goals shall be achieved
  \( \Rightarrow \) take them into consideration when the possibly impacted goals are studied
First diagram

- First Non payment deterred, then Thefts deterred
- Non payment deterred: HOW?
- New goal: Car exit only after payment of a fee

Tactic: Milestone decomposition
Tactic: milestone decomposition

Not yet addressing solution techniques

Just analysing the pb based on what’s known about the needs
An alternative...

Identifying goals helps the analyst ask questions...

Do we need both or only one of them?

Subscription, …
AND vs. OR-refinement

Both subgoals are needed to satisfy the parent goal

AND-refinement

Only one of the subgoals is needed to satisfy the parent goal

OR-refinement

NFR often used to motivate a choice
Decomposing goals: going on...

...until to identify who does what...

Unability of the control system to realise that: other agents are needed...
Tactic: making up for unabilities

• HOW to compute the stay duration?
  – System with tickets
  – System with cameras reading plate numbers

• Goal analysis: allows one to...
  – envision several solutions and position them correctly
  – confirm that the expressed needs are real
  – think about the system
Leaf level

- Information on tickets: ID?, entry time? blank?
- WHO manages time? → CPC

CPC: Car Park Controller

TIM: Ticket Issuing Machine

The right questions at the right time!
Behavioural specification

UML sequence or activity diagrams

Easy translation from leaf goal diagrams

Makes behaviours more/fully deterministic wrt goals (no order on AND-refinement)
Leaf level

Stay duration computed at exit time

Ticket at exit time

No entry if no ticket taken

Δ (entrance, current) time computed

Ticket inserted into payment machine

Ticket kept during stay

No ticket taken ⇒ barrier maintained closed

Barrier maintained closed ⇒ no entry

Definition of the system scope!

Hypothesis!
Behavioural specification

Driver

- push button

Tim

- pop ticket from tank
- read ticket-id
- send ticket-id
- read (ticket-id, current time)
- entry authorized
- release ticket

< display mse

- take ticket

< open barrier

No ticket taken → barrier maintained closed!
Thefts deterred

First idea: case decomposition

Diamond-like structure:
Need for the intermediate layer???
Thefts deterred (cont’d)

- Directed goal graph, not a tree

Diagram:
- Non-payment deterred
- Thefts deterred
  - Can exist only after payment of fee
  - Constant watch kept over all sensitive area
Take all viewpoints into account

- Viewpoints:
  - Manager
  - Owner
  - User
  - Authorities

- Source for conflicts
Example of user goals

Other goals: cheap price, security, user-friendliness, information goals, ...
Conflicts

- between goals of a same agent
- between goals of several agents
Conflict resolution

Obstacle resolution: new requirements!
Nor top-down, nor bottom-up...

Piecing a jigsaw puzzle together instead...

NFR taxonomy:
- high potential for reuse

- Can park control successful
- Law and regulations respected
- User-friendly interface
- Private life respected
- Secure infrastructure
- Safe infrastructure
- No waste of time
- Cheap price
- Non-payment deferred
Regulation

Hypothesis
Cfr statement

- Laws and regulations respected
- Facility structure regulations respected
- Private life respected
- Maximum weight of vehicles respected
- Weight limit between good signs at entrance gates
- Scales at entrance gate
- Barriers opened in case of fire

The list is updated
Obstacles

• De-idealise the model: analyse all what could go wrong

• Process:
  1. Negate the requirement or the expectation
  2. Find the obstacle origin and motivation
  3. Evaluate the obstacle
  4. Solve the obstacle
  5. Integrate in the goal graph
Example

Goal-driven risk analysis

Obstacle pertinence needs validation from domain experts
Example (step 5)

Refinements complemented with obstacle resolution
The best defense, ...

- Playing the bad guy
- Define the anti-goals and refine them into anti-requirements
- Identify the vulnerabilities
- Solve the anti-requirements and the vulnerabilities
Think negatively...

- Pay less than due fare
- Cheat on the full time interval
- Ticket declared lost, stolen, destroyed and short duration claimed
- No way to park duration without ticket
- Anticipate payment
- Use false ticket to exit

Vulnerability

Systematic analysis of delinquent behaviours
Adopt counter-measures

- Use obsolete ticket to exit
  - No exit in obsolete ticket
  - No exit allowed if max duration exceeded
  - Exit time compared with payment time
  - Last payment time recorded
  - Multi-session payment supported

Can park & message

Max duration allowed btw payment and exit time
Behavioural Specification

Goals capture the rationales behind the prescribed behaviour.

Avoid [cheating on the entry time]

Important to inform developers about rationales initially & for releases.

Sequence diagrams: flat, unmotivated

© 2010 by Respect-IT. All rights reserved.
Object Modelling

You are acting as requirements engineering consultant to a client who wants to automate his existing multi-storey car park with time-stamped ticket-issuing machines, payment machines, closed-circuit television cameras to deter both theft and non-payment, and automatic barriers operated by validated (paid-up) tickets.

- **Candidate objects**: domain specific concepts
- **Object attributes**: object qualification
# First object inventory

<table>
<thead>
<tr>
<th>Statement</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ticket-issuing machine</td>
<td>Ticket Issuing Machine (TIM)</td>
</tr>
<tr>
<td>Ticket</td>
<td>Ticket</td>
</tr>
<tr>
<td>Payment machine</td>
<td>Payment machine</td>
</tr>
<tr>
<td>Closed-circuit television camera</td>
<td>Surveillance system</td>
</tr>
<tr>
<td>Automatic barrier operated by validated tickets</td>
<td>Exit station, Automatic barrier</td>
</tr>
<tr>
<td>Car park</td>
<td>Car Park System</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ticket</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stamped</td>
<td>entry time</td>
</tr>
<tr>
<td>Validated (payed up)</td>
<td>payed</td>
</tr>
</tbody>
</table>
Object definition

- Agent Stereotype: if direct operational behaviour (not by means of an aggregation relationship)
- Attribute type: not needed at this stage
- Fill the model in → operationalise the requirements
Object model (« final »)

→ architecture

Minimalist approach: Introduce concepts to cover the requirements
Operationalisation I

- Concerned agents: CPC and TIM
Operationalisation : TIM

No ticket taken => barrier maintained closed

pre: state = closed
post: state = open
Operationalisation: TIM

Operates exit barrier in case of fire

Pre: state = closed
Post: state = open

Shift trigger: Fire Detected

Automatic Barrier
- state {open, closed}
Operation specification (I)

**Operation** Open entry barrier

**Input** \( ab: \text{AutomaticBarrier} \)

**Output** \( ab: \text{AutomaticBarrier} \)

**Pre** \( ab.\text{state} = \text{closed} \)

**Post** \( ab.\text{state} = \text{open} \)

**PerformedBy** TIM

**ReqPre for** [No ticket taken => barrier maintained closed]

  Ticket taken
Operationalisation: CPC

No multi-session yet!
Operation specification (II)

Operation Compute fee

Input fin: Fee, clk: Clock, t: Ticket

Output fout: Fee

Pre t ∈ Ticket_DB ∧ t.ticket_id = fin.ticket_id

Post
- fout.ticket_id = fin.ticket_id
- fout.dueFee = tarif (t.entryTime, clk.currentTime)

PerformedBy CPC

Operationalizes [Diff (entrance, current) time computed]
Operationalisation II

- Use obsolete ticket to exit

- Can park ticket to exit

- Max duration allowed between payment and exit time

- Multi-session payment supported

- Last payment time recorded

- Exit time compared with payment time

- No exit allowed if max duration exceeded

- CPC
Multi-session payment
Operation specification (III)

Operation Compute fee

**Input** fin: Fee, clk: Clock, t: Ticket

**Output** fout: Fee

**Pre** \( t \in \text{Ticket}_\text{DB} \land t.\text{ticket}\_id = \text{fin.}\text{ticket}\_id \)

**Post**

- fout.\text{ticket}\_id = \text{fin.}\text{ticket}\_id
- fout.\text{dueFee} = \text{tarif} (t.\text{entryTime}, \text{clk.}\text{currentTime}) - t.\text{paidFee}

**PerformedBy** CPC

**Operationalizes** [Diff (entrance, current) time computed]

**Operationalizes** [Multi-session supported]
Operation specification (IV)

**Operation** RegisterTicket

**Input** tid: Ticket_Id, clk: Clock

**Output** t: Ticket

**Pre** ¬ (∃ tx ∈ Ticket_DB ∧ tx.ticket_id = tid)

**Post** t.ticket_id = tid

**PerformedBy** CPC

**ReqPost for** [Multi-session supported]

\[ t.payedFee = 0 \]
Operation specification (V)

Operation RecordPayment

Input tid: Ticket_Id, t: Ticket, p: Payment

Output t: Ticket

Pre \( t.\text{ticket}_\text{id} = \text{tid} \land t \in \text{Ticket_DB} \)

Post \( t.\text{payedFee}' = t.\text{payedFee} + p.\text{amount} \)

PerformedBy CPC

operationalizes [Multi-session supported]
State-transition of a ticket

- Payment request / read ticket
  - out [amount paid < due fee]
  - [amount paid > due fee] / release ticket

- Registered
  - exit request

- Timeout (15')

- Fully paid
  - exit request / swallow ticket
  - open exit barrier

NB: incomplete

No exit allowed if max duration exceeded

Deduced from the operation model
Conclusion

Goal model:
- Identification of essential properties to fulfill
- Prescriptive and declarative

Operation model:
- Can be easily translated in UML
- Requirements progressively and systematically taken into account → method

KAOS/Objectiver: a method
to build good UML models
to drive developers (incl. outsourcing)