UNIVERSITY OF TARTU
1632

Institute of Computer Science

MTAT.03.094 – Software Engineering (Tarkvaratehnika)

Final Exam – 13 January 2012

Notes:

- The exam is open-book and open-laptop. Web browsing is allowed, but you are not allowed to use e-mail clients nor Instant Messaging clients.

Context: Selver Supermarkets and Curbside Pickup

Selver (www.selver.eu) is an Estonian-based supermarket chain consisting of 34 stores, employing 2500 people, with annual sales of 300 million euros (2010 number) and a net profit of 10 million euros.

Selver is considering the possibility of offering a new service that has recently picked up in other countries like the US and France. The service is called "Grocery Curbside Pickup" and allows customers to order their shopping cart online and pick it up from a dedicated loading bay at their store of choice. Supermarket staff assembles the shopping cart during the hour preceding the pick-up time chosen by the customer.

Curbside pickup is already offered by international retailers such as Harris Teeter in the US (http://www.theinsightaisle.com/economics-grocery-curbside-pickup/). These previous experiences have put into evidence two of the advantages of offering this service, namely:

- In the short term it allows a first mover (innovative company) to differentiate itself in a highly competitive market like grocery retail. This differentiation leads to improved brand image and additional customers.
- In the middle and long term, this system has the potential to bring down costs for retail chains by allowing them to further automate their check-out process and to adopt better inventory management tactics, especially for fresh products (e.g. dairy products, meat).

Selver hired consultancy firm *StrategicPartners* to do a market analysis and to assess the potential benefits of offering curbside pickup. As a result of this analysis, StrategicPartners recommended that Selver introduces the curbside pickup service in 10 of their stores with a minimum order size of 30 euros. In the first year, Selver could expect 50 purchases per store per day (350 days per year) and the cost of the average shopping cart would be 50 euro. Their analysis also showed that 90% of the sales would come from existing customers switching from normal shopping to curbside pickup. In other words, these purchases would occur anyway regardless of whether curbside pickup is introduced or not. The remaining 10% of online purchases would not occur without curbside pickup, meaning that these purchases would bring additional revenue. StrategicPartners also expects that the sales through curbside pickup will double in Year 2, and increase by another 50% in Year 3 (with respect to Year 2). In Years 4 and 5 sales are expected to be the same as in Year 3.

Disclaimer: The above scenario is fictitious and does not reflect the current situation or plans of the companies in question.

Part I: REQUIREMENTS SPECIFICATION

[40 points]

1) Why does this problem need to be solved?

Using language for the **K**nowledge **A**cquisition for aut**O**mated **S**pecification (KAOS)

a) ***Define a system goal model***. The initial goal model is provided in Figure 1. Ask questions "*why*", "*how*" and "*how else*" to complete this goal model. Your model should result in at least 6 requirements and 3 expectations (potentially more), and illustrate at least 2 alternative goal refinements.

[10 points]

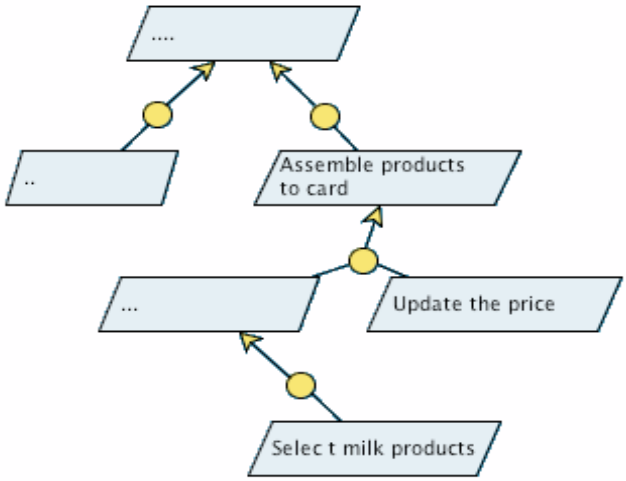

Figure 1. Initial goal model

b) ***Create an agent responsibility model***. Identify major environment and software agents, who are capable to fulfil requirements and expectations identified in your goal model (you do not need to re-draw the goal model, the agents can be assigned on the same goal model).

[5 points]

c) ***Prepare an operation model*** for (at least) one requirement/expectation from your goal model.

[5 points]

d) ***Discuss, which system alternative*** you would recommend to choose for the actual software system design, and ***why***?

[5 points]

Write your answer here:

Your answers will be evaluated (*i*) for the solution correspondence to the given problem, (*ii*) for the solution completeness regarding the requirements given for each task, and (*iii*) for the correctness of the language application (constructs and relationships).

2) What does the software system need to do?

[5 points]

Following the analysis performed in the previous task create a use case diagram. The use case diagram should include at least 5 relevant use cases, illustrate application of *includes* relationship. The *extends* relationship is optional.

You will be evaluated for *syntactic correctness* of the use case diagram, and the *consistency* between your use case diagrams and the KAOS models, as well as whether or not your use case diagram captures the *key functionality* of the envisioned system. You can start with the diagram given in Figure 2.

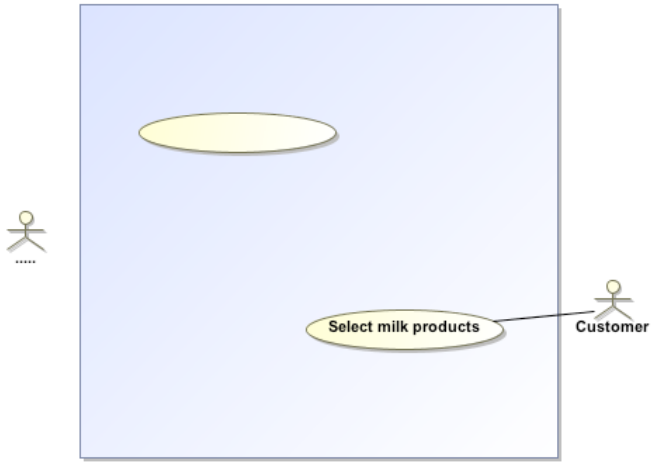

Figure 2. Initial use case model

3) How does the problem manifest itself?

[10 points]

Define 2 use cases for the envisioned system. Each of two use cases will be evaluated for 5 points. The score will be given for the *consistency* between the use case graphical and textual descriptions, for the *level of the details* filled in the template entries, and for the *correspondence* to the problem description.

Please use the "Use Case" template attached to the exam.

Part II: MODELLING

[30 points]

4) Identify the major classes and perform the static problem analysis by creating a class diagram for the given problem.

You will be evaluated for:

- Semantic relevance to the given problem:
  - Identify key classes;
  - Identify relevant attributes and operations;
  - Identify relevant relationships.
- Syntactic correctness and completeness of your diagram. The diagram should include:
  - Association relationships (with reading directions, and multiplicities);
  - At least two association classes;
  - At least one generalisation relationship.

You can start with diagram provided in Figure 3.

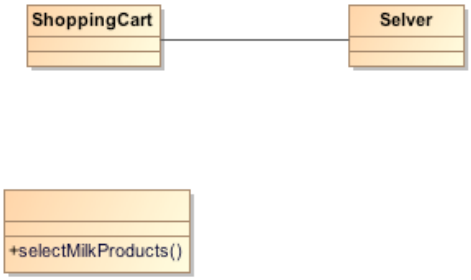

Figure 3. Initial use case model

Part III: CODE DESIGN

[30 points]

Analyse the code snippet given to you below. What are the code-smells or its design issues as to how the code can be improved?

```
01:  public HashMap<Integer, Client> go(boolean rd) {
02:    HashMap<Integer, Client> kliendid = new HashMap<Integer, Client>();
03:    Connection c = null;
04:    try {
05:      c = this.database.getConnection();
06:      ResultSet resultSet = c.prepareStatement("SELECT * FROM
customer").executeQuery();
07:      while (resultSet.next()) {
08:        int customerID = resultSet.getInt("num"); // primaarv√µti
09:        if (rd && resultSet.getString("status").lastIndexOf("M") > 3 || customerID
< 0) {
10:          continue; // skip these clients
11:        }
12:        String nm = resultSet.getString("name");
13:        Date birthday;
14:        try {
15:          birthday = new
SimpleDateFormat("ddMMYY").parse(resultSet.getString("birthday"));
16:        }
17:        catch (ParseException e) {
18:          birthday = null; // who cares?
19:        }
20:        Client cl = new Client(nm, birthday);
21:        if (cl.isKorralik(true)) {
22:          kliendid.put(customerID, cl);
23:        }
24:      }
25:    }
26:    catch (SQLException e) {
27:      System.err.println("sqlexception");
28:    }
29:    finally {
30:      if (c != null)
31:        try {
32:          c.close();
33:        }
34:        catch (SQLException e) {
35:          // vahet pole
36:        }
37:    }
38:    return kliendid;
39:  }
```

You will be evaluated for number of problems/improvements (both general ones and line by line cases) identified.

Use case ID	**ID**		
Use case name	Select milk products		
Created by		**Last updated by**	
Date created	06.01.2012	**Date last updated**	
Actor			
Description	When selecting…		
Trigger			
Pre-condition			
Post-condition			
Normal flow	s.1. …		
Alternative flows	In case …		
Exceptions			
Includes			
Priority			
Frequency of use			
Business rules			
Special requirements			
Assumptions			

Use case ID	**ID**		
Use case name			
Created by		**Last updated by**	
Date created	06.01.2012	**Date last updated**	
Actor			
Description	…		
Trigger			
Pre-condition			
Post-condition			
Normal flow	s.1. …		
Alternative flows	In case …		
Exceptions			
Includes			
Priority			
Frequency of use			
Business rules			
Special requirements			
Assumptions			