Module Software Systems (201300071)
Design Test, 3 december 2014, 8:45–11:45

- Different questions will be graded by different persons. Therefore we ask you **to use a separate sheet for each question** (not the back side of another question).
- The text of this test is in English, but you may answer in Dutch if you prefer (e.g. names of diagram elements or any additional explication you want to give).
- You are allowed to consult the book, the manual and a printed version of the slides. It is not allowed to consult other materials – including your own notes.
- Diagrams can be drawn with pen or pencil as you like.
- When you are ready, please hand in only the answers to the questions. You can take the test with you.

Repair service of an internet shop

RedHot Ltd. Is an internet shop for consumer electronics, such as tablets and smartphones. It claims to distinguish itself from similar shops by its superb customer service. If you have a problem with any product purchased from RedHot you can call them for advice, and you'll be connected to a knowledgeable person within a minute or two. E-mails are answered within a few hours. You can also visit them physically in their Rotterdam shop, where service staff can help you select the best choice before you buy something.

In this case study we disregard most of their services and focus only on the repair service.

Any article that you bought at RedHot can be sent back for repair when it breaks down.

If a customer wants to return an article for repair, they call the RedHot Customer Service. The Customer Service employee inquires what the problem is, and creates a service record for this article in the system, containing a brief description of the problem. The customer than (automatically) gets an e-mail with instructions. This e-mail has two attachments:

(1) A return form to be filled in and enclosed with the article. This includes the printed summary of Customer Service. The customer can add further details as appropriate.

(2) A label for the mail parcel. It contains a free delivery code, so that the customer does not have to pay postal charges. Also, it carries a barcode that RedHot can use, when the parcel arrives, to route it to the right destination (the Technical Service department).

When an article arrives at RedHot, its arrival is registered by the Technical Service department. If an article is still under warranty (*garantie*), the repair is free of charge. If the warranty has expired, the repair costs will be charged to the customer. However, the customer needs to know what the cost will be before they can decide whether or not to repair it.
The procedure for this is as follows. If there is no warranty, Technical Service will first make an estimate of the repair costs. Making this estimate is free of charge. A Technical Service employee enters the cost estimate into the system, possibly with a brief report to explain what the estimate is based on (and it is also stored who reported it and when). The customer automatically gets an e-mail with the estimate and the report, asking the customer to give permission for the repair (by clicking a link in the e-mail) or to cancel the repair (by another link).

If the costumer does not respond within a week, the system sends another email. This time, a clause is added that failure to respond within three days is interpreted by RedHot as cancelling the repair. *(Note: you can model failure to respond as a cancelling message from the customer.)* This happens very rarely, however, most customers respond within a day.

If the repair is cancelled, Technical Services returns the broken article to the customer. Giving permission for the repair includes giving permission to RedHot to debit the repair costs to the customer's bank account or credit card.

Depending on the type of defect and the warranty status, an article will be repaired by RedHot's Technical Service department or returned to the supplier for repair. *(We assume, for*

simplicity, that any article can be repaired. In some cases it will be replaced, rather than repaired, but that does not change the process in any way.)

When repairs are carried out by RedHot's Technical Service, this is always done fast. It goes without saying, in line with their service-oriented values, that the customer does not need to wait longer than necessary. However, when an article is sent to the supplier for repair, this is outside RedHot's control. It happens that suppliers take a lot more time…

When an article is sent to a supplier for repair, a Technical Service employee enters into the system that the article has been sent. When the article returns from the supplier, this is also entered by a Technical Service employee

When an article has been sent to the supplier for repair, and it has not been returned within 14 days, Redhot (automatically) sends an e-mail to the customer, apologizing for the delay, explaining that this runs against RedHot policies, but they cannot force the supplier, etc. In the unfortunate case that the repair takes many weeks, this message is repeated every 14 days.

When the repair is finished, a Technical Service employee finalizes the repair by filing a repair report. The employee also indicates the cost of the repair, if applicable (and it is also stored who reported it and when). The cost that is charged is never higher than the given estimate (even though the real costs could be, at the expense of RedHot) – but it could be lower if the repair was easier than anticipated.

When the article has been repaired by the supplier, finalizing the repair is essentially the same. In this case also a Technical Service employee finalizes the repair, upon arrival of the article, by filing a repair report and the repair costs (*which have to be refunded to the supplier sometime, but that is outside the scope of our model*).

The last step in the repair process is returning the repaired article to the customer, which is also done by a Technical Service employee. If there is no warranty, the Technical Service employee debits the repair costs to the customer's bank account or credit card.

It could happen that a repaired article breaks down again and needs to be repaired a second time. In the system this is treated as a separate repair, which will follow the same repair process again.

RedHot is always keen to optimize its processes. For that purpose, for every action by a RedHot employee in the repair process (e.g. creating a service record; giving a cost estimate; sending the article to the supplier) the following information is stored in the system: what the action is (for each action there is a specific code); the time (which includes date) at which the action took place; the employee who performed the action.

In addition to the information already mentioned above, the following data are stored in the system:

- For each customer: surname; Christian name; e-mail; telephone number; address; customer number.
- For each RedHot employee: surname; Christian name; e-mail; telephone number; address; *Burgerservicenummer* ("BSN", social security number); employee number; department; function; e-mail (private); e-mail (work).
- Articles have a brand name (e.g. "Apple"); a product name (e.g. "iPhone"); often a product description (e.g. "5S 16 GB"); sometimes a colour (e.g. "Black"); a sales price; a supplier (*see below*); a warranty period (e.g. one year). For every sold article it is known: the customer whom it was sold to; date of purchase; order number; price paid (which can be different from the current sales price).
- The supplier can be the manufacturer or another organization representing the manufacturer. We make no difference, and store the following data about the supplier: company name; address; e-mail; telephone number; contact person (if known); contact hours (i.e. when the supplier is reachable by telephone).

The various e-mails mentioned above need <u>not</u> be included in the class diagram. (*They are stored somewhere, but that is outside the scope of our model.*)

Source: The described process is based on the repair service of coolblue.nl. However, in order to make the case study doable, some aspects have been modified or simplified.

Question 1 [2 points]

Make an activity diagram for the repair service at RedHot. Make the AD for a single repair. (If an article needs to be repaired again, the same process can be applied a second time).

Additional information:

For this activity diagram we use a new construct, not included in the slides and lab sessions. A *composite activity* can be used to refer to another activity diagram for part of the model (like the **ref** construct in sequence diagrams). Figure 1 shows the collapsed form. The symbol "⊓" indicates that it is not an atomic activity, but there is another diagram in which this activity is spelled out in more detail. You can simply copy Figure 1 into your activity diagram, as a placeholder for the subprocess that has been modelled in Figure 2.

Figure 2 takes care of repair by RedHot and repair by the supplier, as well as sending delay messages to the customer.

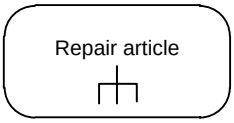

Figure 1: Composite activity "Repair article" (collapsed)

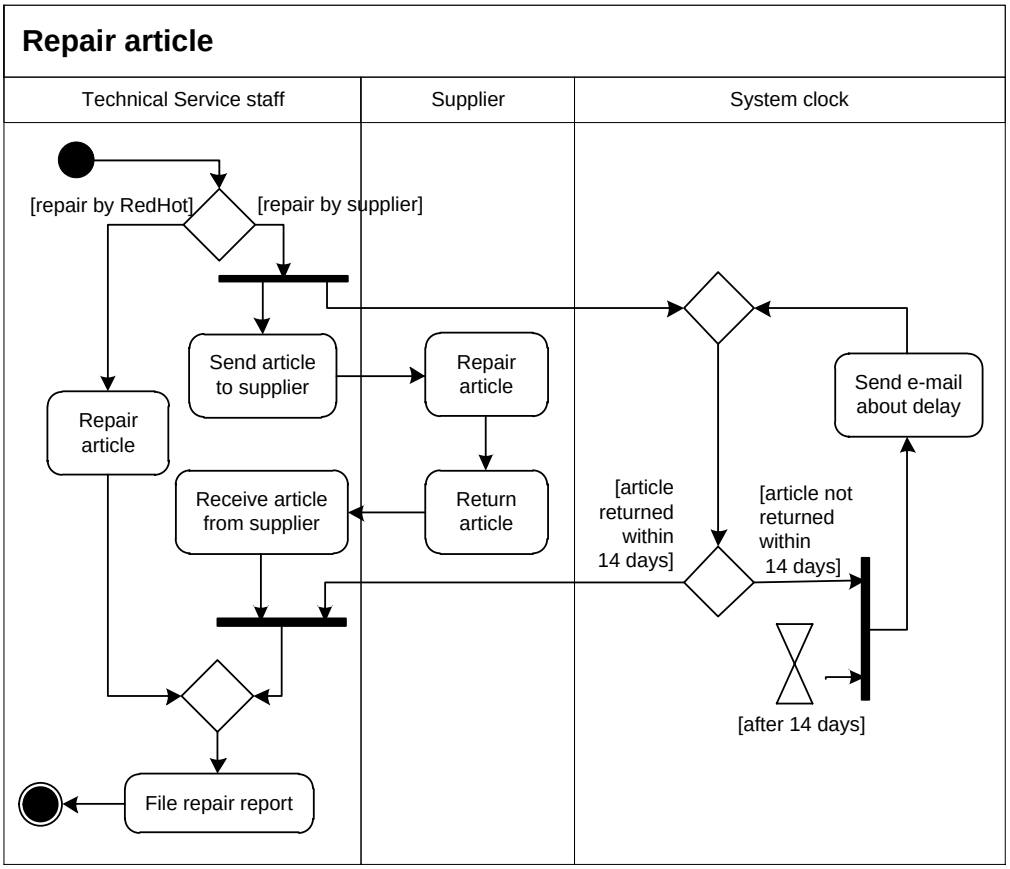

Figure 2: Activity diagram for "Repair article"

Some more remarks about Figure 2:

- The tricky part in this model concerns the repeated e-mail messages to the customer. This has been modelled by two parallel threads, for the repair and for the messages. Note that the messages thread can be finished at any moment when the article is returned.
- Note the swimlane labelled "System clock". Each activity must have an actor. "Send e-mail" is automated, no human actor is involved, so the clock is the actor.

Question 2 [2 points]

2a. Make an actor list for the repair service. Include all actors that directly interact with the system and only those.

2b. Make a use case diagram for the repair service of RedHot.

Hint: When a repaired article is received back from the supplier, the return needs to be stored, but otherwise it is the same as when the article would have been repaired by the Technical Service. This can be modelled as an «include» as in Figure 3. Note, furthermore, that "Find service record" would be an obvious inclusion in "Receive article from supplier"; it is included now through inclusion of "Finalize repair".
You can copy Figure 3 in your answer and extend it as appropriate.

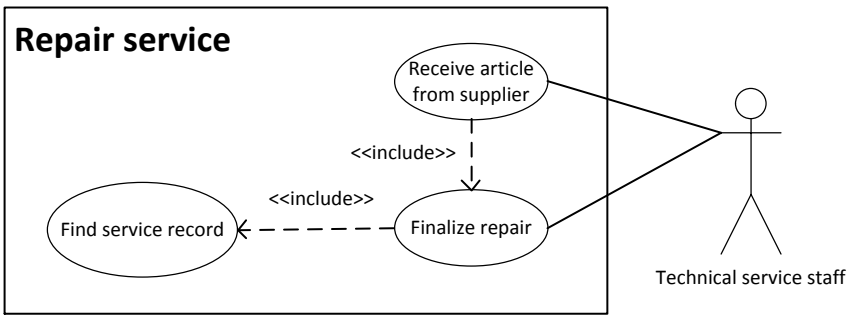

Figure 3: "Finalize repair" included by "Receive article from supplier"

Question 3 [3,5 points]

Make a class diagram for the system, based on the description above.

Hint: Use "Service record" – see Figure 4 – as the central class in your class diagram. The attribute "permission known" denotes whether the customer has given (or denied) permission to repair the article after getting a cost estimate. If "warranty" is false, then "to be repaired" only gives useful information after "permission known" has been set to true.

Do not add further attributes to the class "Service record". Additional information that is relevant for the service record can be modelled by associating it with other classes.

Service record
description: String warranty: Boolean permission_known: Boolean to_be_repaired: Boolean

Figure 4: the class "Service record"

Question 4 [2,5 points]

Make a state machine for an article, including the various stages of repair. The state machine should allow for multiple repairs for the same article.

Hints:
(1) you can start in a state "article at customer's, no defects known" (no need to include states of the article before it was purchased).
(2) Note that during the lifetime of the article, the warranty may expire. This is independent from the repair procedure (when the service record is created it is determined whether warranty applies, it doesn't matter if it would expire during the repair process).