

Requirements specification for Hotline Support System

Customer

...

Supplier

...

Author: Soren Lauesen

Contents

A. Background and reading guide.....	2	C6. Change role	5
A1. Background	2	Work area 3: Management.....	5
A2. How to use the requirements	2	C7. Study performance and statistics	5
B. High-level demands.....	2	C8. Update basic data	5
C. Tasks to support.....	3	D. Data to record	6
Work area 1: The IT user	3	D1. Request	6
C1. Report a problem	3	D2. History	7
C2. Follow up on a problem.....	3	D3. Cause.....	7
Work area 2: Hotline.....	3	D4. Employee	7
C3. Handle a request in first line.....	3	D5. Supporter.....	7
C4. Handle a request in second line.....	4	E. Virtual Windows.....	8
C5. Handle message from an external supplier.....	5		

A. Background and reading guide

A1. Background

This is Søren Lauesen's solution to the use case experiment announced June 2009. The solution is based on the template Requirements SL-07. The goal of the experiment is to compare use cases and tasks when applied to the same problem.

A2. How to use the requirements

The requirements are written in tables, e.g. a table with requirements relating to a specific user task. The tables have two columns. The left column describes the customer's demand or a problem to be mitigated (the requirement). The right column may show a solution example to help the reader understand the demand. The solution is not a requirement. Often the solution column is blank.

In the hotline case, the customer isn't sure whether they want to buy an existing system or expand the one they have. They should use the tables in this way:

1. Together with a potential supplier, they walk through all the demands and make notes about how well the supplier's solution satisfies the demands. As an example they carry out the user tasks by means of the supplier's solution. The notes are written in the solution column.
2. The customer transforms these notes into ratings of the supplier. They may repeat the procedure for a few more suppliers and in this way select the best.
3. They also walk through the demands and notice how well their present system satisfies the demands and what would be needed to improve the support.

Experience shows that this is a good basis for deciding what to do. It is of course necessary to look at all the other requirements too, for instance response time, usability, and security. The SL-07 template covers all of this, but it is not part of the exercise.

In a formal tender process, the suppliers submit a proposal where they explain their solution in the right-hand column. In a later contract, the right-hand side has become the agreed solution.

The requirements are enumerated. Alternative ways of performing a subtask are *variants*, marked with letters a, b, etc. Problems relating to how things are done today are marked with the letters p, q, etc.

B. High-level demands

This chapter explains how the customer's business goals are met through the requirements and how to mitigate high-risk requirements. Not relevant in the experiment.

C. Tasks to support

The system should support all user tasks in this chapter, including all subtasks and variants. It should also as far as possible, remedy the problems. The left-hand side of the tables shows what user and computer must do together. The right-hand side may show an example of a solution. The example is not a requirement.

The subtasks are numbered for reference purposes. They don't have to be carried out in this sequence, and many of them are optional. A subtask may also be repeated during the task.

1a: Means a variant of subtask 1 - another way to do subtask 1.

1p: Means a problem relating to subtask 1. The problem is something the customer experiences today.

Work area 1: The IT user

C1. Report a problem

Start: The user has an IT problem, often urgent.

Subtasks and variants:	Example solutions:
1 Call hotline or meet in person.	
1a Send mail to hotline.	
2 The problem may be solved on the spot.	
2p Problem: When not solved on the spot, when can I expect a reply?	The system reminds the supporter to set <i>reminderTime</i> and helps him send a mail about it to the user.

C2. Follow up on a problem

Start: The user is impatient or gets a note from hotline.

Subtasks and variants:	Example solutions:
1 Check whether the problem has been solved.	
1p Problem: Hotline forgets to tell when the request is completed.	The system helps the supporter send a mail when hotline closes the request.
2 See how far the request has come.	Lookup in the request base
3 Provide more information.	
4 Maybe close the request.	
5 Maybe resume the request (open it again).	

Work area 2: Hotline

We might describe 1st and 2nd line with the same task because many subtasks are the same. But the two lines have different triggers and other details differ too. In general it is a bad idea to combine tasks too early, because important details may be lost.

C3. Handle a request in first line

Start: This task can be started in many ways, for instance: A user calls or sends a mail; a reminder arrives because a request is overdue; or the supporter has done something else and now looks for the next request to handle. This is described below as variants of subtask 1.

End: The supporter cannot do more about the request right now.

Subtasks and variants:	Example solutions:
1 Record the request, particularly the user's phone or email. (See data details in section D1.)	The system warns if user or phone isn't specified when the request is created.
1p Problem: Cumbersome, particularly when it is an on-the-spot solution.	Fields allowed being blank as far as possible.

Subtasks and variants:		Example solutions:
1a	The request arrives by email through the support system.	The system records sender details, subject, etc. automatically.
1b	The user provides new information about an open request. Record it.	
1c	The user wants to reopen a closed request.	
1d	The user records the new information or the reopen request on-line.	
1e	A reminder arrives because a request has not been dealt with in due time.	The system sends a reminder.
1f	The supporter has finished doing something else and now looks for a request to handle.	
1r	Problem: In busy periods it is hard to spot the important and urgent requests.	Can restrict the list to relevant requests. Can sort according to reminder time, priority, etc.
2	Maybe solve the problem on the spot and close the request, possibly with an explanatory email.	The system warns if the cause hasn't been set.
2p	Problem: To gather statistics, a cause should be specified, but this is difficult and cumbersome today.	A list of frequent causes to choose from.
3	Maybe leave the request in the "in-tray" for later treatment.	The system warns if a reminder time hasn't been set.
4	Maybe transfer the request to second line or a supporter with special expertise. Give it a priority and a reminder time.	
4p	Problem: Hard to see which supporters are present now.	

C4. Handle a request in second line

Start: The supporter gets an email about a request or looks for pending requests.

End: The supporter cannot do more about the request right now.

Subtasks and variants:		Example solutions:
1	Look at open second-line requests from time to time, or receive an email.	
1p	Problem: In busy periods it is hard to spot the important and urgent requests.	Can restrict the list to relevant requests. Can sort according to reminder time, priority, etc.
2	If necessary, contact the user or receiver to obtain more information.	
3	Solve the problem by moving to the problem location.	
3p	Problem: Would be nice if the supporter could handle several problems on his trip.	Mark some of them and print them.
4	Work for some time on the problem. Inform others that they don't have to look at it.	Put the request in state <i>taken</i> .
5	Order something from an external supplier and park the request.	The system warns if no reminder time has been set.
6	In case of a reminder, contact the supplier and set a new reminder time.	
6p	Problem: The user doesn't know about the delay.	The system sends a mail when the reminder time is changed.
7	Close the case.	The system warns if the cause hasn't been set.
7p	Problem: To gather statistics, a cause should be specified, but this is difficult and cumbersome today.	A proven list of causes to choose from.

Subtasks and variants:		Example solutions:
7q	Problem: The user isn't informed when the request is closed.	The system sends a mail when the request is closed. The supporter has the possibility to write an explanation in the mail.
8	Maybe leave the request in the "in-basket" or transfer it to someone else.	

C5. Handle message from an external supplier

Start: A message or a delivery is received.

End: The supporter cannot do more about the request right now.

Subtasks and variants:		Example solutions:
1	Find the request.	
2	(Continue as in C4)	

C6. Change role

Start: A supporter has to change line, leave the hotline, etc.

Subtasks and variants:		Example solutions:
1	Change own settings for line, etc. (see section D5).	
2	Change other supporter's settings, for instance if they are ill.	
3	Transfer any taken requests to 2nd line.	
3p	Problem: The supporter forgets to transfer them.	The system warns and suggests changing status for all of them.

Work area 3: Management

C7. Study performance and statistics

Start: (More elicitation needed.)

Subtasks and variants:		Example solutions:
1	Get access to history data.	Transfer all history records to a data-mining tool, which is not part of this delivery.
2	Look at data in various ways. (More elicitation is needed.)	

C8. Update basic data

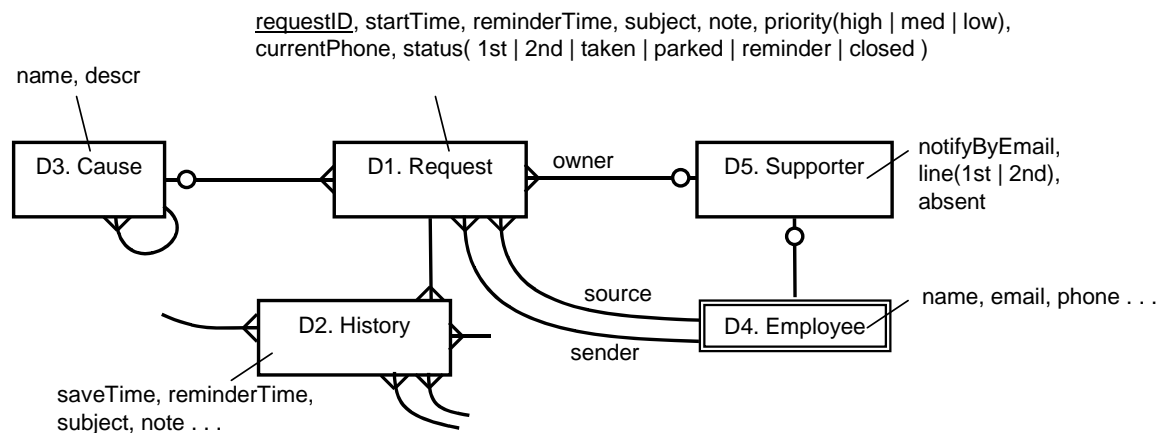
Start: (More elicitation needed.)

Subtasks and variants:		Example solutions:
1	Update employee data.	Do it in the personnel system and let hotline get the data from there.
2	Update supporter list, list of problem causes, etc.	

D. Data to record

The system should record the data described in this chapter. It is possible to create, view, and change the data through the appropriate tasks. In some cases data is retrieved from an external system, for instance the personnel system (shown as a double-border box). Data may also be exported, for instance history data to a data mining package. (In the full template this is described in section F.)

The figure is an Entity/Relationship diagram (E/R) that gives an overview of the data. Each box holds all entities (objects) of a certain kind. A crow's foot shows that an entity in one box is related to many entities in the other box. Data need not be structured in this way in the system, but it should be handled in some way.



D1. Request

A request is a problem report sent by the IT user or recorded by a supporter.

Fields and relationships:	Example solutions:
1 requestID: The identification of the request.	Generated by the system.
2 startTime: The point in time where the user sent the request or the supporter recorded it.	Generated by the system.
3 reminderTime: The point in time where the system will set the request in reminder state.	The system offers some default times.
4 subject: The title given by the IT user.	
5 note: An explanatory text written by the person who set the request in its current state.	
6 priority: The supporter's estimate of how urgent the request is.	
7 currentPhone: If the sender works away from his usual place, we may record his current phone.	
8 status: The current state of the request.	See solution notes below.
9 cause: A relationship to what the supporter currently believes is the cause.	
10 sender: A relationship to the IT user who reported the problem.	Set automatically if the user sent the request by email.
11 source: A relationship to the person who set the request in the current state. When a supporter records the request, he is the source. When an IT user records it, he is the source as well as the sender.	Generated by the system.
12 owner: A relationship to the supporter who works on the request. Null if nobody works on it.	Generated by the system.

Solution notes

The system might support these states:

First line	A first-line supporter must take on the request, for instance because it just arrived.
Second line	A second-line supporter must take on the request.
Taken	The request is handled by a supporter (the owner). The owner may change from one line to another while he is handling the request.
Parked	The request awaits something, for instance an external delivery, and hotline need not do anything meanwhile.
Reminder	The request hasn't been closed in due time, or the external delivery wasn't received in due time.
Closed	The request has been handled. However, it may be opened again, for instance because the user doesn't think the problem has been solved.

D2. History

A history record is a time-stamped copy of a request. The history records make up a log of what has happened in the hotline.

Fields and relationships:	Example solutions:
1 saveTime: The history record was saved.	Generated by the system.
2 requestID, reminderTime, etc.: Copies of the request data.	Generated by the system.

D3. Cause

The cause records make up a hierarchical list of possible request causes. It is a maintenance task to update this list according to changing needs.

Fields and relationships:	Example solutions:
1 name: A short name of the cause, e.g. "Password lost".	
2 descr: A longer explanation of the cause, maybe including ways to deal with the problem.	
2p It is difficult to create a suitable list.	The system provides a proven list as a starting point
3 belongsTo: A relationship to the higher-level cause in the hierarchy.	

D4. Employee

Data about all employees.

Fields and relationships:	Example solutions:
1 name, email, phone: As usual.	The system retrieves data from the existing personnel file or copies them on a regular basis.

D5. Supporter

Describes the current role of a supporter.

Fields and relationships:	Example solutions:
1 notifyByEmail: Whether the system sends an email to the supporter when there is a request for him.	
2 line(1st 2nd): Whether the supporter currently works in 1st or 2nd line.	
3 absent: Whether the supporter is away, for instance for lunch or on a course.	

E. Virtual Windows

This chapter might be part of the requirements. It visualizes the data requirements. However, the virtual windows shown below were made as the first step of designing the user interface. The full design, the program, the test data, and lessons learned are available at request.

Request list

Include: 1st: ☒ Parked: ☒ All taken: ☒ From: 23-09-05 12:30
 2nd: ☒ Closed: ☐ To:

Req	Status	Owner	Prior	Age	Left	Cause	Sender	Subject
6134	1st	knt	high	0:01			carsten	Please help . . .
6048	taken			0:16	0:14	Printer	hilde	Toner missing
6047	2nd		high	0:31	1:29	Security	nhn	Password wrong
5978	parked	esben		2d 02:12	21:48		carstensen	Own printer
5930	remind	esben		3d 00:03	0:00	Printer	nete	Color toner missing

Warning color for reminders

Request

ReqID: 6047 Start: 23-09-05 12:53 Sender: nhn Nicole Hanson Phone: 5512
 Status: closed Remind: 23-09-05 14:53 Subject: Password wrong
 Owner: Priority: high Latest note: Guided on phone
 Source: kevin
 Cause: Security

Time	Status	Owner	Source	Cause	Note
23-09-05 13:52	closed		kevin	Security	Guided on phone.
23-09-05 13:40		kevin	kevin	Password	Needed logon to the personnel system
23-09-05 13:05	2nd	peter	peter	Password	Lost access rights?
23-09-05 12:56	closed	peter	peter	Password	Guided on phone.

Two alternative display modes

Time	Status	Owner	Note
23-09-05 13:52	kevin		Guided on phone. User couldn't find the settings menu.
23-09-05 13:40	kevin		Need logon to the personnel system
23-09-05 13:05	peter		Lost access rights? Or some mistake when we updated the personnel list?
23-09-05 12:56	peter		Guided on phone. Got new password and instruction.

Cause

Password
 Printer
 Security
 . . .

Black
 Color
 Ink jet
 . . .

Reminder

None
 Now + 10 min
 Now + 30 min
 Now + 2 hours
 Now + 6 hours
 Now + 1 day
 Now + 3 days

Supporters

Email	Absent	Line	Email notify
brh	<input checked="" type="checkbox"/>	1st <input type="text"/>	<input checked="" type="checkbox"/>
esben	<input type="checkbox"/>	2nd <input type="text"/>	<input checked="" type="checkbox"/>
knt	<input type="checkbox"/>	1st <input type="text"/>	<input type="checkbox"/>
. . .			

At most 10-15 supporters