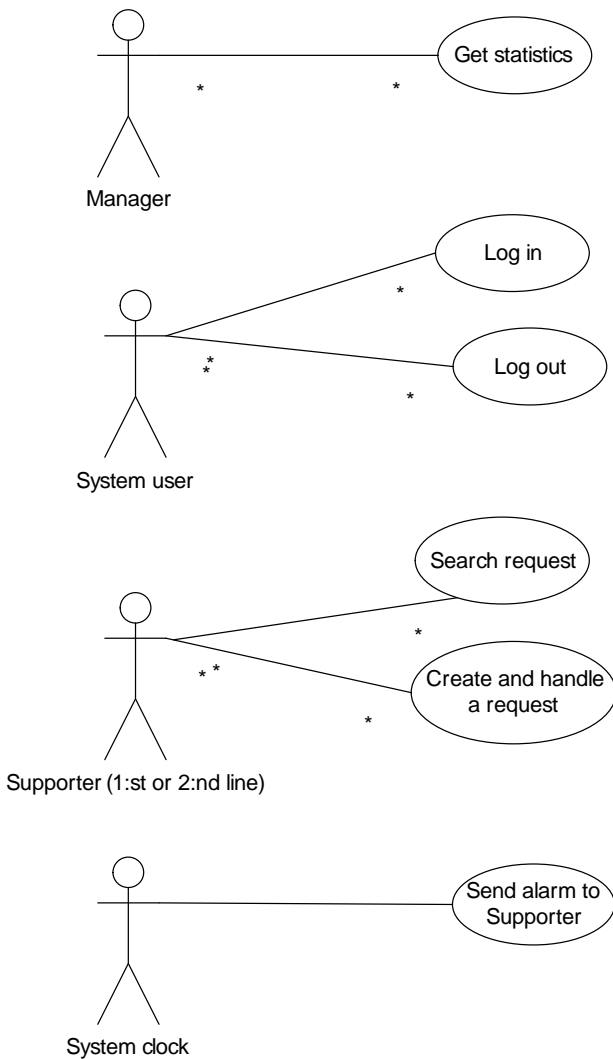


Hotline support system - Ulrika Nordstrom

Use Case: Use Case Model

1. Use Case Model

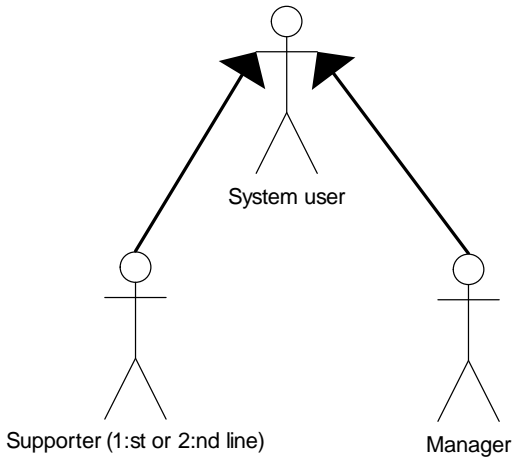
1.1 Use Case Diagram



1.2 Actors

1.2.1 System user

Abstract actor. Supporter and Manager inherit from System user.



1.2.2 Supporter

The Supporter can handle a request (e.g. register a solution to a request) and register a new request. The Supporter acts as 1:st or 2:nd line support.

1.2.3 Manager

The Manager uses the System to see statistics.

1.2.4 System clock

Triggers the System at specific times.

1.3 Short description of Use Cases

1.3.1 Send alarm to Supporter

Events listed in the flows triggers the System to send notifications to Supporters.

1.3.2 Create and handle a request

...

1.3.3 Search request

...

1.3.4 Log out

...

1.3.5 Log in

...

1.3.6 Get statistics

...

Hotline support system

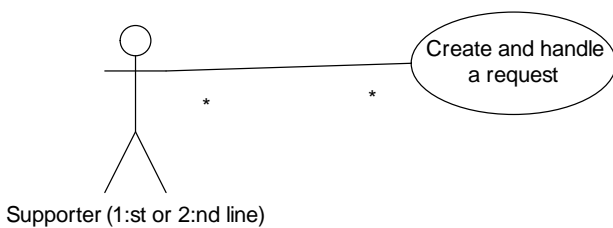
Use Case: Create and handle a request

2. Create and handle a request

2.1 Short description

The Use case describes functionality to handle Requests in the System. The Supporter can for example register a Request in the System, take ownership of a Request and take action to an already registered Request.

2.2 Local Use Case Diagram



2.3 Actors

2.3.1 Supporter

The Supporter can handle a request (e.g. register a solution to a request) and register a new request. The Supporter acts as 1:st or 2:nd line support.

3. Precondition

3.1 Main flow 1

- The Supporter is logged in.

3.2 Main flow 2

- The Supporter is logged in.
- There is at least one Request registered in the System. The Request has Status = Open <same support level as the Supporter>.

3.3 Alternative flows

- The Supporter is logged in.
- There is at least one Request registered in the System.

4. Flow of events

4.1 Main flow 1 – Register a Request

Main flow 1 describes the flow when a Supporter registers a Request in the System.

1. The use case starts when the Supporter chooses to register a Request.
2. The System asks for Request information; Request description, Priority, Expected completion time, Current owner, ..and more.
Out: Priority {selection of Priority where default Priority is Open <same support level as the

Supporter}, Current owner {default current Supporter}.

3. The Supporter gives Request information and confirms the registration.
In: Request {Request description, Priority, Expected completion time, Current owner.
4. The System creates and saves the Request and displays a confirmation.
5. Use Case ends.

4.2 Main flow 2 – Handle a Request

Main flow 2 describes the flow when a Supporter takes action to Request that is registered in the System.

1. The use case starts when the Supporter selects a Request to handle.
In: Request
2. The System displays the Request.
Out: Request {Request ID, Request description, Status, Current owner, Priority, Expected completion time, Incoming date and time}
3. **If** Status is not already Taken by the Supporter, the Supporter sets the Request Status to Taken.
4. *<Reference point: Take action>*
The Supporter types a Description and ...more and confirms the changes.
5. The System saves the Request and sets Status to Done.
6. *<Reference point: End flow>*
Use Case ends.

4.3 Alternative flows

4.3.1 Send notification

The alternative flow starts in *<Reference point: End flow>* if the Reported by – contact info is set.

1. The System asks the Supporter if a notification shall be sent
Here, the details must come when the project is running and development has started.
2. The Supporter adds (optional) additional text to the message and confirms.
 - 2.1. **If** the Supporter does not want to send a notification, the Use case ends.
3. The System sends the e-mail to Reported-by.
Out: Message sent to Reported-by.
This can be done from the system or by integrate to a mailserver but that is too early to say at this point.
4. Use Case ends.

4.3.2 Take ownership

The alternative flow starts in Main flow 2 <**Reference point: Take action**> if Request Status is maybe rules here? And the Supporter takes the Ownership from another Supporter.

1. The System set the current Supporter to be Request Owner and save the Request.
2. Alternative flow ends.

4.3.3 ...and more requirments

The alternative flow starts in <Reference point: ...> ...

3. The System ...
4. ...
5. Alternative flow ends.

5. Postcondition

5.1 Main flow

A new Request with a unique identifier has been created in the System. Also, date-time when the Request was created.

5.2 Alternative flows

All updates that have been made by the Supporter has been stored in the System. Also, date-time when the Request was updated.

6. Local Supplementary Requirements

7. Glossary

7.1 Status

A request must have one (and only one) status. The following Status types are valid:

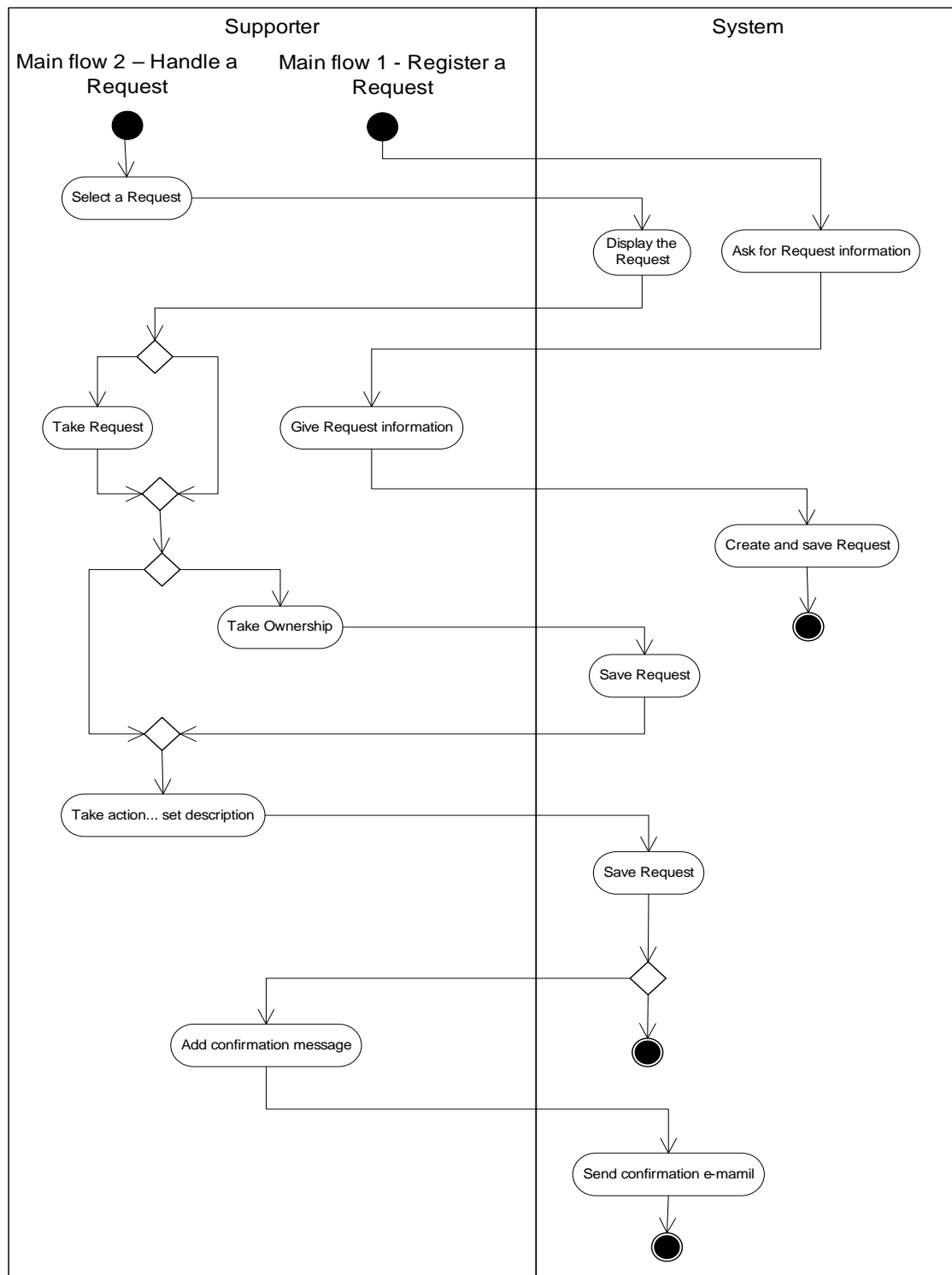
First line	<description>
Second line	<description>
Taken	<description>
Parked	<description>
Remainder	<description>
Closed	<description>
Open	<description>

7.2 Request

Request ID	A unique identifier
Status	
Request description	<description>
Current owner	Initials of Supporter
Expected completion time	Yymmdd
Incoming date and time	Yymmdd hh:mm:ss

Solution	
Reported by	<i>Name</i>
Reported by – contact info	<i>Email address</i>

8. Activity diagram



Hotline support system

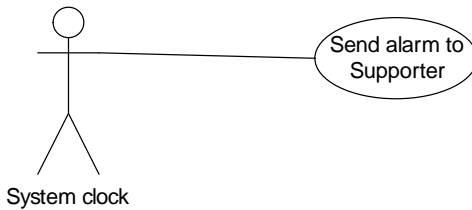
Use Case: Send alarm to Supporter

9. Send alarm to Supporter

9.1 Short description

Events listed in the flows triggers the System to send notifications to Supporters.

9.2 Local Use Case Diagram



9.3 Actors

9.3.1 System clock

-

10. Precondition

- There is a Request with Status not equal to Closed for which the Expected completion date has expired.

11. Flow of events

11.1 Main flow

6. The System clock sends an email to the Supporter that is the Owner of the Request.
Out: Email message {Request ID, Expected completion date, ...more}
7. Use Case ends.

11.2 Alternative flow

11.2.1 Alternative flow 1 ...

5. Other “automatic” happenings listed as alternative flows.

12. Postcondition

-