S60 UI Specification Guideline

Version 1.0
May 8, 2006

S60 platform

Legal Notice

Contents

Change History

May 8, 2006	Version 1.0	Initial document release

1. Introduction

This document is a practical guideline that explains how to create a good user interface (UI) specification for any product designed for the S60 platform. This document describes the purpose of the UI specification and the process needed for its creation. The *S60 UI Specification Template* published with this document is a simple form that can be used in any UI specification related to an S60 application.

1.1 What is a user interface specification?

In short, a user interface specification is a document that explains how the target product[1] works in practice. It is a complete description of what the end user sees, hears, and feels when he or she is interacting with the product, and it covers all actions that the end user may perform with the product. The main input needed to create a UI specification is the requirement specification, and the main customer for a finished UI specification is the product's implementation team. The UI specification is very often regarded as a great overview document that helps its readers to understand the purpose and design of the product. This makes it crucial for the whole product creation process. In addition to the implementation team, at least product testing personnel and marketing communications experts will use the UI specification as input for their work.

In a product development team, the processes of requirement specification, UI design, and UI implementation (that is, coding) are usually different tasks. Even if the same people would be participating in all these tasks, it is important to recognize the special characteristics and goals of each task.

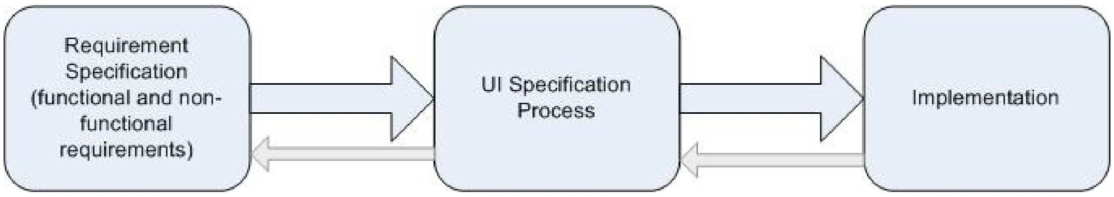

Requirement Specification (functional and non-functional requirements)
UI Specification Process
Implementation

Figure 1: UI specification process in a product development context

1.1.1 Requirement specification as input

The development of a product starts with the specification of the product requirements. In an optimal case, the requirements of a product reflect actual market and end-user needs and are the result of a thorough analysis. Since the main input for the UI specification is the software requirement specification (or several separate requirement specifications), the quality of those documents has a direct effect on the quality of the UI specification. Good requirements are clear, concise, and only state what the product should be able to do. Requirements should not restrict the UI design by explaining how the requirement should be fulfilled.

The first handover happens when the product requirements have been approved, and the frozen requirement specification is delivered to the UI designer. This is where the UI specification process begins.

[1] The term "product" is used for the sake of simplicity to refer to any of the following: a feature, a set of features, a service, an application, a set of applications, or a device.

	Tip: The requirements need to be ready and approved when the UI specification starts. It is, of course, possible to make changes to the requirements if the need arises, but as a rule, the requirements need to be frozen when the handover occurs.

1.1.2 UI specification in context

The UI specification can be regarded as the document that bridges the gap between the product management functions (requirements definition) and implementation. One of the main purposes of a UI specification is to process the product requirements into a more detailed format. In general, the goal of requirements is to describe WHAT the product is capable of, whereas the UI specification contains the details needed to describe HOW these requirements are implemented in practice.

However, the UI specification process includes more than the further development of the product requirements. It is also about design. The UI designer responsible for the UI specification is in charge of a creative task that requires him or her to be innovative and clever.

1.1.3 Implementation as customer

The second handover happens when the UI designer delivers the final version of the approved UI specification to the implementation team. The implementation team relies on the UI specification to contain all the details that they need when they start the implementation (usually coding) work.

The most important customer of the UI specification is the implementation team. When composing the UI specification, it is good to take into account the fact that it must serve the needs of the implementation.

1.2 Creating a user interface specification

It is important to keep in mind that a UI specification is only as good as the process by which it has been created. Therefore, there are a few things to keep in mind during the UI specification process.

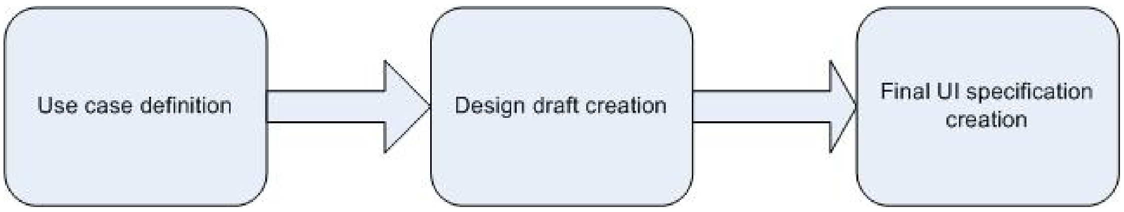


Figure 2: The UI specification process

The first task in the UI specification creation process is to ensure that all the (main) use cases are known. A great software requirements specification will provide a list of use cases that the product supports. However, in many cases, it is the UI designer who needs to create them.

1.2.1 Use case definition

Use cases are essentially short stories that explain how the end user starts and completes a specific task. Each product supports a number of main use cases that form the backbone for the product concept. For example, an application that allows the user to remotely lock and open doors could have the following use cases: (1) Lock a door (2) Open a door (3) Configure auto-lock. These use case titles represent the tasks that the user wants to do with the application.

The purpose of writing use cases is to enhance the UI designer's understanding of the features that the product must have and of the actions that take place when the user interacts with the product. With well-defined use cases, the task of creating a UI specification is much more efficient. Like requirements, use cases should not contain any details about HOW the user performs the task. Avoid using statements with design details. For example:

Avoid: "1. User selects the [Lock door] menu. 2. System displays the [Door locked] confirmation note"

Prefer: "1. User chooses to lock the door. 2. System informs the user that the door has been locked"

Tip: If it is not possible to write all the main use cases, keep a prioritized list of the use cases titles.

See Appendix A, "Use case example," for an example of a use case.

1.2.2 Design draft creation

The first version of a UI specification is called a UI design draft. The UI design draft is done on the basis of a use case analysis. The main purpose of the UI design draft is to show the design direction what the UI designer is proposing, and to explain how the user interface enables the user to complete the main use cases, without going into details. The design draft usually contains the following:

- Rough wireframes[2] that show the main UI states (a.k.a. views) in the device.
- The rough wireframes are usually displayed in a hierarchical UI state diagram. This is a block diagram that shows the hierarchical relationships between the main UI states.

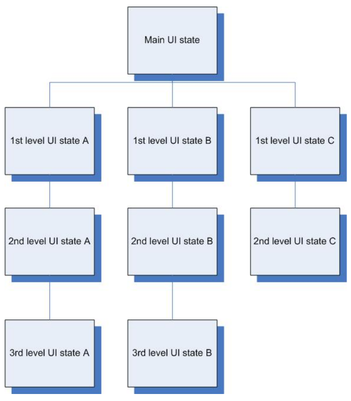


Figure 3: Example of a simple UI tree

- The task flow for the main use cases, usually displayed with wireframes.

[2] A wireframe is a crudely drawn picture of the user interface. It shows the UI components that are used, and their location on the display. It does not reflect the final graphic design. Wireframes are usually drawn with a vector graphics program.

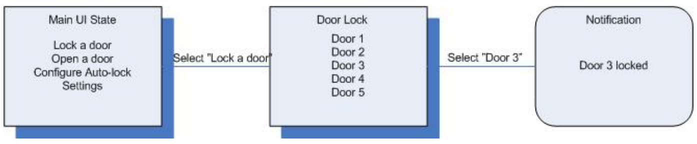


Figure 4: Example of a simple task flow diagram

Tip: Do not include any exceptional situations or minor use cases to the design draft.

The format of the design draft is not restricted; it can be anything from a PowerPoint presentation to an XML document. The most important thing to keep in mind is that it should be as visual as possible and all the material created must be in such a format that it can be used in the final UI specification.

The design draft should always be reviewed with the relevant product management and development staff: product manager, implementation team, localization and internationalization experts (if applicable), and product testing staff. The testing staff is usually interested in the UI specification because the test cases are often created on the basis of the UI specification. It is recommended that the UI design draft review is done properly, since at this point it is easy and fast to make changes to the document. The design draft reaches the APPROVED status when all review comments have been handled and all stakeholders agree that the document is now acceptable.

Tip: The design draft phase is the perfect time to conduct usability testing or expert evaluations. At this point in time, all improvement suggestions received from usability tests are easy to take into account.

To conclude, the approved UI design draft represents an important agreement between all the relevant players. A design draft that is well done makes the UI specification phase more efficient.

1.2.3 Writing the user interface specification

The user interface document extends and builds on the design draft. While the design draft is a light description of the most important design issues, the UI specification is a complete description that contains all details, exceptions, error cases, notifications, and so forth.

For a detailed description of the content of a UI specification, see Chapter 2, "The contents of a user interface specification."

The UI specification is a written document with images. The most usual formats are: Word document, PDF document, or HTML or XML document.

The UI specification should be reviewed with the same relevant stakeholders as the design draft. The main difference between these reviews is that while the design draft review represents an agreement on a design direction, the UI specification review is a formal acceptance of all the design details. The UI specification is considered to be approved when all the stakeholders have given their comments and the UI specification is edited accordingly. The approved UI specification is used as the basis for the implementation work.

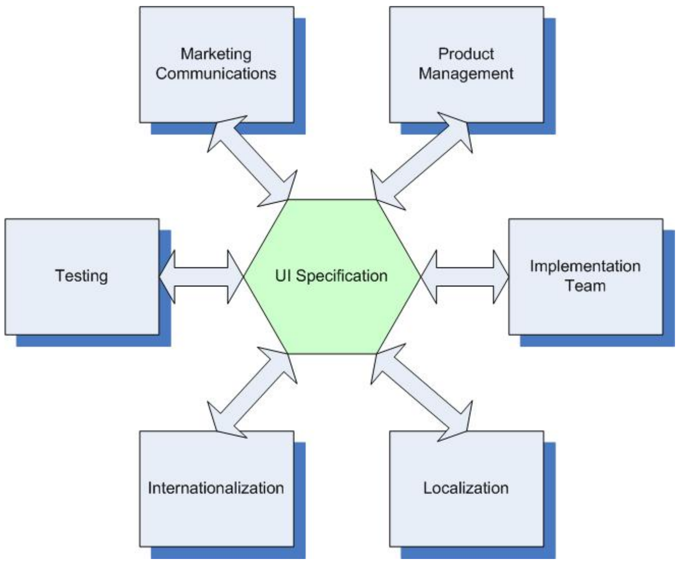
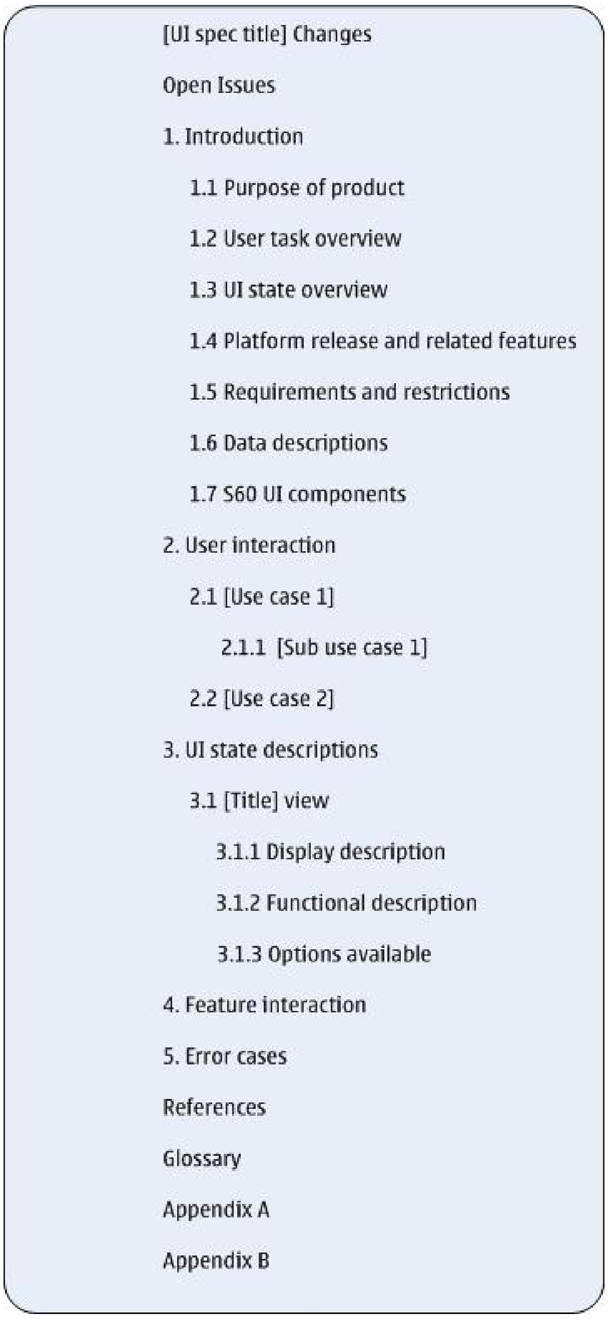


Figure 5: Different stakeholders in creating a UI specification

ℹ	**Note:** Marketing communications staff does not usually participate in the reviews, but they are interested in getting a draft of the UI specification as early as possible.

2. The contents of a user interface specification

The UI specification contains display descriptions of all the UI states and notifications used in the product. It explains what UI components are used and provides additional descriptions that are specific to the product in question. It also describes all the functions that are available in each of the UI states and notifications. In addition to providing a full depiction of the user interface, the UI specification also describes exceptions to the main use cases, and possible error situations that may occur. The recommended table of contents for a S60 UI specification looks like this:



[UI spec title] Changes

Open Issues

1. Introduction
    - 1.1 Purpose of product
    - 1.2 User task overview
    - 1.3 UI state overview
    - 1.4 Platform release and related features
    - 1.5 Requirements and restrictions
    - 1.6 Data descriptions
    - 1.7 S60 UI components
2. User interaction
    - 2.1 [Use case 1]
        - 2.1.1 [Sub use case 1]
    - 2.2 [Use case 2]
3. UI state descriptions
    - 3.1 [Title] view
        - 3.1.1 Display description
        - 3.1.2 Functional description
        - 3.1.3 Options available
4. Feature interaction
5. Error cases

References

Glossary

Appendix A

Appendix B

Figure 6: Table of contents

The following sections explain the proposed content in more detail.

ℹ	**Note:** Not all of the sections are mandatory. The detailed descriptions for the sections explain how and when they should be used.

2.1 Cover page

[Mandatory]

The cover page should display at least the following information:

- Name of the company
- Title of the UI specification
- Current version of the UI specification
- Status of the UI specification (DRAFT/PROPOSAL/APPROVED)

2.2 [UI specification title] changes

[Mandatory]

This unnumbered chapter in the beginning of the document is used to summarize the changes that have been made between the different versions of the document. The best way to show the changes is by using a table and devoting a new row to each batch of changes (that is, for each document version), starting with the most recent changes.

Each change entry should contain the following information:

- Version of the document
- Current status of the document (DRAFT, PROPOSAL or APPROVED)
- Date of the new version
- Name of the editor
- Reason for changes (for example, Inspection comments, or change request)

If the document version contains different kinds of changes, it is possible to use subtitles to differentiate between the types of changes.

For example:

Door Lock System Changes

Document version	**Document status**	**Date**	**Editor**	**Changes**
V1.1	APPROVED	27-Mar-2006	Edith Example	Editorial changes: • corrected a typo in Chapter 3 • improved graphic 1.
V1.0	APPROVED	15-	Edith	Inspection changes:

		Feb-2006	Example	• Door locked notification edited to include a large graphic • Instead of the “Locking wait note” a “Connecting to door progress note is displayed.”

2.3 Open issues

[Optional]

This section can be used to list all design or technology issues that are still open. If an approved UI specification contains open items, the document reviewers must agree that these items can be left pending. The structure of this chapter is free: lists, tables, or free text can be used as seen fit.

Note: The number of open items should be kept as small as possible to maintain the integrity of the specification.

2.4 Introduction

[Mandatory]

The purpose of this chapter is to give an overview of the product so that the reader can form a basic understanding of how it works. The different sections in this chapter provide different types of relevant information. When writing the introduction, avoid describing details, and use graphics whenever possible.

2.4.1 Purpose of product

[Mandatory]

This chapter should provide answers to the following questions:

- What can the product be used for? / What can the user do with this product?
- Who are the typical end-users of this product?
- What are the most important features of the product?
- What is the end user need that this product fulfills? / What are the main benefits of this product?
- What are the operator benefits (if any)?

Tip: Please note that this section provides general background information and highlights the most important characteristics of the product. A complete list of user tasks should be provided in Section 2.4.2, “User task overview.” Detailed, step-by-step descriptions of the user tasks are written in Chapter 2.5, “User interaction.”

2.4.2 User task overview

[Recommended]

This section is intended to contain a list of main use case headings. By listing the use case headings, this section gives a thorough overview of the available features. For each listed use case heading, there should be a reference to the appropriate subsection in Section 2.5 “User interaction.” Here is an example:

The Door Lock enables the user to complete the following tasks:

- Open a door remotely. See Section 2.5.1. “Opening a door remotely”
- Lock a door remotely. See Section 2.5.2. “Locking a door remotely”
- Configure the system to lock and open doors automatically. See Section 2.5.3. “Configuring the autolock system”

This section is recommended because it provides a well-rounded overview of the product features. By reading this section, the reader is able to form an understanding of what the product offers to the end user.

2.4.3 UI state overview

[Recommended]

The best way to give an overview of the different UI states is by using a UI diagram. The UI diagram is one item that should have been created for the UI design draft, so it should be readily available for the UI specification. If necessary, you can break the UI diagram into several smaller images.

This section is recommended, because it shows the UI structure in a highly visual way. It is easier to understand the UI view descriptions when it is possible to refer to a diagram that describes the relationships between the different views.

2.4.4 Platform release and related features

[Mandatory]

This section should list the S60 platform releases for which the product is intended.

If the product is closely related to another product or to a feature of the S60 platform, these relationships and dependencies should be described in this chapter. Similarly, if the product cannot coexist with another product or another feature of the S60 platform, this fact should be documented here. This section should only provide information about related S60 platform features, other requirements should be listed in the following section 2.4.5. “Requirements and restrictions”. Here is an example:

[Name of the product] requires the following features: [feature 1], [feature 2], and [feature 3].

2.4.5 Requirements and restrictions

[Optional]

This chapter explains the technical requirements of the product (that is, all the needed technical capabilities and accessories). It also describes the non-technical requirements, such as usability standards, market needs, and international standards that have to be met.

Examples:

1. [Name of product] requires that the S60 smartphone has either a built-in GPS chip or a compatible GPS accessory.
2. [Name of product] requires the API for messaging.
3. This product requires [X] amount of ROM and [X] amount of RAM.

If there are some restrictions related to the (implementation of) the product, they should be listed here. The implementation team is a good source of information if more details are needed about requirements or restrictions.

Tip: Check the following details: software and hardware limitations (for example, processor capabilities, memory consumption, and needed external APIs), and international standards that are relevant for the product.

2.4.6 Data descriptions

[Mandatory]

This chapter should provide the following information about the data handled by the product:

- What kind of data the product can handle (that is, store and render).
- What kind of data management functions are available for the user and what are the data types that the user can create, store, and/or modify.

Data can be numeric or alphanumeric in type, or it can be a file format, for example.

The most common way to present this information is by using the following table. If it is not possible to insert a specific piece of information, the N/A (not applicable) value can be used instead.

Data	Data type displayed to the user (Y/N)	Value range	Default value	Notes

If this section is not relevant, you can replace the usual content of this section with the following phrase: [Product name] does not handle any data that could be saved or edited.

2.4.7 S60 UI components

[Optional]

This section should list all the S60 UI components that have been used in the product. If the product does not use any of the UI components offered by the S60 platform, this section can be left out. When all the S60 UI components that have been used in the user interface are listed in one location, it is easy to find this information.

Here is an example of how to provide this information:

[Product name] utilizes the following S60 UI components:

- Selection grid
- Confirmation note
- List query

2.5 User interaction

[Recommended]

The purpose of this chapter is to provide task-oriented descriptions of the user interaction. Use cases are the method of choice for this type of information. As explained earlier, the UI specification process should start with the definition of the most critical use cases. This means that this part of the UI specification is most likely the first one to contain detailed information. It is recommended to include this chapter since use cases are a great way to make sense of the product's behavior. By writing use cases, the UI designer will gain a proper understanding of the user experience requirements of the product.

The use cases should be presented in a logical order, so that the reader can easily follow the descriptions.

	Tip: One way to order the use cases is by following the intended UI state structure and the content of the options lists. Please note that if you are using standard S60 UI components, it is not necessary to describe how they function in detail.

2.5.1 Use case 1

Each use case should follow the standard use case structure:

- Task title (and an optional introduction)
- Preconditions (optional)
  - The preconditions mean essentially the situation in the beginning of the use case.
- Trigger (optional)
  - Definition of the action that starts the use case.
- Task sequence described step-by-step
  - Most often the steps alternate between a user action and a system's response.
- Extensions
  - That is, optional ways for the use case to go forward.

- Exceptions
    - That is, unexpected errors or problems that alter the use case flow.

ℹ	**Note:** Please note that the exceptions should not include all possible exceptional situations. Include only those exceptions that are relevant for the use case at hand. For example: If the use case is about saving a file, a possible exception would be the situation in which the memory is full. However, a situation in which the memory has become corrupted is not an exception; it is an error case that should be covered elsewhere.

- Postconditions (optional)
    - Description of the situation after the use case has been completed.

See Appendix A, “Use case example,” for an example of a use case.

ℹ	**Note:** The use case should only describe WHAT happens in each step, but exclude all design details (that is, the HOW). The use case should contain both system and user actions.

2.5.1.1 Sub use case 1

Some use cases contain smaller use cases that are better described separately. Also, some exceptions can be separated into sub use cases.

2.6 UI state descriptions

[Mandatory]

This chapter is the placeholder for the detailed descriptions for all the UI states and views. This chapter complements the overview that is provided in Section 2.4.3, “UI state overview.”

In addition to the UI states, this chapter also describes the notifications used in the product. Notifications are described in the same way as the UI states.

Present the UI states and notifications in a logical order. For example, the UI states could be described in the order in which they appear in the UI diagram, and notifications could be presented in connection with the UI states to which they are connected. Use the same UI state names that have been used in the UI diagram, and ensure that the naming of the UI states is consistent. Each UI state should have its own section within this chapter. For example:

3 UI state descriptions

3.1 Lock Door Main View

3.2 Open Door View

3.3 Close Door View

3.4 Auto-lock View

3.5 Settings View

Each UI state description should have the following subsections: Display descriptions, Functional descriptions, and Options available.

Note: If the UI state uses default S60 UI components, it is not necessary to describe the components or their parts in the UI specification. With S60 UI components, it is enough to mention the name of the component clearly. However, if the component is used in an exceptional way, and some optional parts of the component are used in a different way, explain the differences in detail. Remember to explain the logical texts used in the UI component.

If the product supports several languages, the use of logical text strings is recommended. Logical text strings are an easy way to refer to a text string used in the UI in a unique way. It is possible to map several translations to one logical text string. Remember that there can be only one UI text string for each logical text. The final version of the UI specification should then use the logical texts when referring to a display text. For example: Automatic door locking off §auto.lock.off§

2.6.1 Display descriptions

[Mandatory]

This section should *always* contain images that represent the UI view in question. The images do not need to reflect the final graphical design of the view. It is possible to use hand-drawn images, screen shots from the emulator, PowerPoint sketches, or virtually any type of graphics. Use several images to show the entire screen content, if the screen is scrollable. If the UI state can have several states, include examples of the different states.

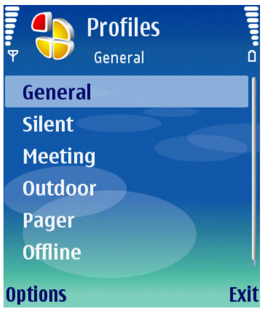


Figure 7: Example of a display description image

In addition to the image, this chapter should explain the layout used in the UI view. If the layout used is a default S60 UI component, use the correct name to refer to the UI component.

Use a table to describe the different areas in the display and the content shown in these display areas.

An example of the display description table:

Display area	Content displayed
Status pane	Displays the following data: View title [Main view] in the Title pane Clock [Analog or digital] in the Context

	pane …
Main pane	List of available features: Used component: Single-line item with graphics [Lock a door] [Open a door] [Configure auto-lock]
Context pane	Displays the following soft key labels: SK1 “Options” SK 2 “Exit”

Note: Keep in mind the use of logical texts!

2.6.2 Functional descriptions

This section defines all functions related to the UI state in question. The most efficient way to present this information is by using a table that lists all available functions and the related actions. Remember to describe key events of all available keys. If a key is inactive in the UI state, this fact needs to be specified.

Note: All key events should be included in the table, including, for example, all number keys in an ITU-T keyboard. Remember to define actions to the **Selection key** and to the **Clear key.**

User Action	Effect
Key 1	Detailed description of the effect, including references to other display descriptions. For example: Opens the Settings view, see Section x.x “Settings View”.
Key 2	See above.

Note: If the application supports S60 smartphones with a full QWERTY keyboard, another separate table is needed for the QWERTY keys.

2.6.3 Options available

If the UI state offers more functions via an **Options list**, all the **Options list** functions are listed and explained in this chapter. Please note that the **Options list** might also be a UI state of its own, depending on the design of the product. The order in which the options list items are presented should be the same in which they are intended to be listed in the final product.

Note: Remember to describe the contents of the context-sensitive menu. The context-sensitive menu is used in situations in which it is not sensible to assign a single action to the **Selection key**.

Again, a table can be used to present this information:

Options menu item	Effect
Item 1	Detailed description of the effect, including references to other display descriptions. For example: Opens the Settings view, see Section x.x “Settings View”.
Item 2	See above.

Note: Keep in mind the use of logical texts!

2.7 Feature interaction

This chapter complements the information provided in Section 2.4.4, “Platform release and related features.” While Section 2.4.4, “Platform release and related features” gives an overview of the feature interaction, this chapter is reserved for providing more in-depth information about the following issues:

- If the product interacts with other features or applications in the S60 platform, the nature of interaction should be explained here.
- Does the product have an impact on the behavior of other features or applications in the S60 platform or vice versa?
- Are there some S60 settings that affect the behavior of the product, or vice versa?

The use case format is the recommended way to present this information. See Appendix A for an example of a use case.

Tip: Consider also the effect of the following issues: memory-related behavior, possible effect of accessories, network-related questions, and the restoration of factory settings.

2.8 Error cases

This chapter is designed to explain the user experience of the product in error situations. The purpose of the error case descriptions is twofold:

1. To list all possible error cases.
2. To describe how the device behaves in — and recovers from — the error cases.

All error notifications that are used should be mentioned in this chapter. However, the description of the error notifications should be included in Section 2.6 “UI state descriptions," and they should follow the same logic by which all UI states are described (that is, an example image is needed, and both display and functional descriptions should be included).

If there are many different types of errors to be taken into account, it is possible to divide this chapter into several sections, for example:

1. Errors caused by the user.
2. Errors caused by the product.
3. Errors caused by the S60 smartphone.
4. Errors caused by the network.
5. Errors caused by the accessories.

The titles for error cases should reflect the situation as it is experienced by the user, or by the system. Here are some examples:

- No GPS signal received
- Unable to open a file
- Memory full

2.9 References

This chapter is essentially a list of documents that have been referred to in the UI specification. List the documents in alphabetical order. The usual references include: S60 UI Style Guide, S60 UI component descriptions, other UI specifications, and international standards.

2.10 Glossary

If the UI specification includes some terminology or abbreviations that may not be understood by all readers, it is possible to use this chapter to explain the terms. All terms should be listed in alphabetical order, and each term entry should contain an explanation. For example:

Term	Explanation
RDLS	Remote door locking system; the software that enables the user to remotely control the locking or opening of doors.

2.11 Appendices

Appendices can be used quite freely to provide additional reference material. For instance, Appendix A can be used to collect all the logical names mentioned in the UI specification. The benefit of this is that the localization work can be made more efficient when all the localizable text strings can be found in one location.

3. References

S60 UI Style Guide available at www.forum.nokia.com

3.1 Further reading

Introduction to S60 UI Components available at www.forum.nokia.com

Appendix A. Use case example

Title	Charging an Item To Phone Bill
Preconditions	User has entered Game Service and he has discovered an item he needs to buy. User's device, country, and operator are identified. Device supports COD/JAD. The item to be purchased can be downloaded to the user's own device or sent to somebody else's device.
Trigger	This use case is triggered when the user has selected an item from the list and the system shows details of the selected item: Item name, price, member offer price, SMS pull code, description (item type), and possible options: buy for yourself / send to a friend.
Step sequence	The user chooses the option to buy the item for himself. The system asks if the user wants to pay with credit card or charge the item to his phone bill. The user selects to charge the item to his phone bill. The system completes operator authorization. The system initiates a COD download. The system (device) asks for confirmation: item name, item price, description (item type), file size, vendor. The user accepts the purchase. The system (device) starts downloading the content. When the content has been successfully downloaded to the device, the user is charged. The user gets the content to his device and chooses to save it. The system shows a Thank You screen and notifies that the price has been charged to the user's phone bill.
Extensions	1a.The user chooses to send the item to somebody else. a1. The system asks if the user wants to pay with credit card or charge the item to his phone bill. a2. The user selects to charge the item to his phone bill. a3. The system completes operator authorization. a4. The system asks for the recipient's phone number. a5. The user types the recipient's phone number. a6. The system asks for confirmation: item name, item price, description (item type), file size, vendor, charging to phone bill. a7. The user accepts the purchase.

	a8. The system starts sending. a9. The system shows a Thank You screen and notifies that the item was sent and the price has been charged to the user’s phone bill.
Exceptions	4a. Operation authorization fails. a1. The system notifies the user that there has been an error. There is a Retry option. a2. The user chooses to retry. a3. Continues from step 2.
Postconditions	An item is bought and the user’s phone bill has been charged.

Evaluate this resource

Please spare a moment to help us improve documentation quality and recognize the resources you find most valuable, by rating this resource.