Inf1B: Data and Analysis Use Cases

Contents

What are Use Cases?

Use Cases capture **who (*actor*)** does **what (*interaction*)** with the system, for **what purpose (*goal*),** without dealing with system internals

A **complete set of Use Cases**:
- specifies ***all ways to use the system***
- defines ***all behaviours required*** of the system

A **Scenario** is an ***instance of a use case***, and represents a ***single path through a Use Case***
- one scenario for the main flow through the use case
- other scenarios for each possible variation of flow through the use case
(e.g. triggered by options, error conditions, etc.)

Use Cases….

- are a ***means of representing the intended functionality*** of the system or interface
- are taken from a ***user perspective***
- ***different cases*** are used ***for different system uses***
- ***show where to start*** interaction with the system
- go through the basic system flow ***one step at a time***
- ***document alternative flows***
- ***vary in detail*** from only show intent, to detail of interface
- may ***help generate further functional requirements*** not previously identified
- are the ***first step in*** moving from requirements to ***implementation*** (but **do not specify implementation**)
- are ***useful in communicating with clients*** who require system
- are ***useful in generating test cases*** for the system

Writing a Use Case

1. Brainstorm functional requirements of system
2. Identify the main actors (roles that people or objects play)
3. Exhaustively list the user goals for the system
4. Select one use case to expand
5. Write the main success scenario (MSS)
6. Document most basic flow of events, ignoring problems
7. Brainstorm and exhaustively list the alternative flows/extension conditions
8. Write steps to handle extensions: will point back to MSS, to success (exit) or failure (sub case or alternative flow)
9. Add, subtract, merge as needed

See http://alistair.cockburn.us/usecases/usecases.html

Joke Generation Tool: Functional and Usability Requirements

Be able to generate jokes:
1. Based on a **topic** **Food > Vegetables > Onion**
   What kind of vegetable can jump?
2. From **keyword**(s) Using **car** and **sandwich**
   What do you get when you cross cars and sandwiches?
3. From **templates** **bazaar:** How does a ___ ___?
   How does a whale cry?
4. From **Favourite Jokes** list
   How is a car like an elephant?

- not too many key presses
- easy to go back if make unintended selection
- different levels of access to manage varying language skills

Joke Generation Tool: Functional and Usability Requirements

Be able to generate jokes:

1. Based on a **topic** **Food > Vegetables > Onion**
   What kind of vegetable can jump? ***A spring onion.***
2. From **keyword**(s) Using **car** and **sandwich**
   What do you get when you cross cars and sandwiches? ***Traffic Jam***
3. From **templates** **bazaar:** How does a ___ ___?
   How does a whale cry? ***Blubber blubber.***
4. From **Favourite Jokes** list
   How is a car like an elephant? ***They both have trunks.***

- not too many key presses
- easy to go back if make unintended selection
- different levels of access to manage varying language skills

Use Cases: Main Toolbar

Options presented through the main toolbar:
"USE CASE 0" Main toolbar

Alternative flows for main toolbar options
A1: User selects **'BACK'** option
A2: User selects **'HOME'** option
A3: User selects **'EXIT'** option
A4: User selects **'HELP'** option
A5: User selects **'FORWARD'** option

Main Toolbar - Steps and Alternative Flows

A2: User selects 'HOME' option
A2.1. System: ***displays a request for confirmation*** and offers the options:
- *Confirm go to main menu*
- *Cancel go to main menu request*

A2.2. User: ***confirms to go to main menu*** [A2-1]
A2.3. System: ***<<uses>> USE CASE 1***

A2-1: User decides to cancel go to main menu request
A2-1.1. System: ***returns to System step*** in Use Case where User selected 'HOME' option,
e.g. if User selects 'HOME' option at User step *i*,
System returns to step *i*-1.

Use Cases 1: Main Menu

USE CASE 1. Main menu
1. System: displays options:
   - Get a joke ***by topic***
   - Get a joke ***by keyword***
   - Get a joke ***by type***
   - Browse "***My Favourite Jokes***" list

   It also displays the **main toolbar**.
2. User: selects to get a joke ***by topic*** [A1] [A2] [A3] [A4] [A5] [A6] [A7] [A8]
3. System: ***<<uses>> USE CASE 2*** where ***topic T*** is the 'root' topic
4. System: (upon end of USE CASE 2), jump to Step 1 of USE CASE 1.

END OF USE CASE 1

Use Cases 1: Alternative Flows

Alternative flows for USE CASE 1

A6: User chooses to get a joke by keyword
A6.1. System: ***<<uses>> USE CASE 3***
A6.2. System: (upon end of USE CASE 3), jump to Step 1 of USE CASE 1.

A7: User chooses to get a joke by type
A7.1. System: ***<<uses>> USE CASE 4***
A7.2. System: (upon end of USE CASE 4), jump to Step 1 of USE CASE 1.

A8: User chooses to browse "My Favourite Jokes" list
A8.1. System: ***<<uses>> USE CASE 5*** where *C* = "My Favourite Jokes" list
A8.2. System: (upon end of USE CASE 5), jump to Step 1 of USE CASE 1.

USE CASE 2: To get a joke by topic

USE CASE 2. **To get a joke using topic hierarchy *T***
1. System: ***displays N subtopics*** and/or keywords under topic *T*.
   If there are more than *N* subtopics available under topic *T*,
   it presents an option to see some more subtopics under *T*.
   It also presents an option to get a joke on topic *T* .
   Finally, it also displays the **main toolbar**.
2. User: ***opts to get a joke on topic T*** [A1] [A2] [A3] [A4] [A5] [A6] [A8]
3. System: ***presents choice to User***:
   - *Get previously generated/stored jokes* under selected subtopics
   - *Try to generate a new joke* under selected subtopics

   It also displays the **main toolbar**.
4. User: ***opts to see a newly-generated joke*** [A1] [A2] [A3] [A4] [A5] [A9]

USE CASE 2: continued….

5. System: ***generates a new joke on selected topic T*** and ***adds it to the generated joke log***.
6. System: ***displays the generated joke*** onscreen for User to read. Offers options:
   - *Get another new joke* on topic *T*
   - *Get previously generated/stored jokes* on topic *T*
   - *Add currently displayed joke* to "My Favourite Jokes" list
   - *Use speech synthesis to speak joke*

   It also displays the **main toolbar**.
7. User: ***selects to have system speak joke*** [A1] [A2] [A3] [A4] [A5] [A7] [A9]
8. System: **<<uses>> USE CASE 6** where *J* = joke generated at Step 7.
9. System: (upon end of USE CASE 6), jump to Step 8.

END OF USE CASE 2

Some Alternative flows for USE CASE 2

A6: User chooses to see subtopics and/or keywords under subtopic *S*

A6.1. System: ***Jumps to USE CASE 2 Step 1***, ***replacing T with S***.

A8: User chooses to see more subtopics (at current level, where >*N* choices)

A8.1. System: ***Jumps to USE CASE 2 Step 1***, but this time ***showing the next N subtopics*** under topic *T*.
If there are fewer than *N* remaining subtopics, presents all of them, and if there are no more remaining subtopics, it cycles back to the first *N* subtopics.

A9: User decides to see previously generated/stored jokes

A9.1. System: ***<<uses>> USE CASE 5*** where *C* = set of previously generated/stored jokes under topic *T*

Use Cases and UML

Unified Modelling Language (UML) is standard way ***to specify, construct and document systems that use object-oriented code***

[Derived from Booch (Object-Oriented Design, Rumbaugh (Object Modelling Technique) and Jacobson (Object-Oriented Software Engineering)]

UML Diagrams provide ***different perspectives*** for viewing software systems ***in varying degrees of abstraction***:

- **Use Case Diagrams**
- Class Diagrams
- State Diagrams
- Sequence Diagrams
- Collaboration Diagrams
- Activity Diagrams
- Component Diagrams
- Deployment Diagrams

More more information see http://www.uml.org/

References

Cockburn, A. Structuring Use Cases with Goals. *Journal of Object-Oriented Programming*, Sep-Oct, 1997 and Nov-Dec, 1997. Also available on: http://alistair.cockburn.us/crystal/articles/sucwg/structuringucswithgoals.htm

Cockburn, A. Resources for Writing Effective Use Cases, http://alistair.cockburn.us/usecases/usecases.html

Jacobson, I. et al. *Object-Oriented Software Engineering: A Use-Case Driven Approach*, Addison-Wesley, Reading, MA, 1992