Scenario Analysis

*Why, What & How*

---

**Scenario Analysis: Why?**

- System Specification
  - Elicitation
  - Real Needs

- Software Req. Specification
  - Elicitation
  - Validation

- Software architectural design

- Implementation
thru scenarios
clarify the relevant properties of the application domain
uncover missing (functional) reqs.
evaluate design alternatives
validate design alternatives
ensure testing adequacy
(integration, validation, ...)

Lawrence Chung
"Scenarios describe how users of the system interact with the system to achieve their particular tasks."

"Scenarios usually refer to representative instances of user-system interactions."

Three Types of Scenarios

- **Use cases:**
  short, informal descriptions of situations possibly followed by explanatory phrases.

- **Episodes:**
  phases of activity; an episode is a cluster of inter-related event occurrences.

- **Scripts:**
  sequences of (detailed) user-system interaction in a tabular or diagrammatic form. (action table/diagram)
Use cases - Jacobson’s:

"A use case is a specific flow of events through the system (seen as a black box)"

Example

A. The user case begins when the actor Guest enters the restaurant.
B. The actor Guest has the possibility of leaving her/his coat in the cloakroom, after which s/he is shown to a table and given a menu.
C. When the actor Guest has had sufficient time to make up her/his mind, s/he is asked to state her/his order. Alternatively, Guest can attract the waiter’s attention so that the order can be placed.
D. When the Guest has ordered, the kitchen is informed what food and beverages the order contains.
E. In the kitchen, certain basic ingredients, such as sauces, rice, and potatoes, have already been prepared. Cooking therefore involves collecting together these basic ingredients, adding spices and sorting out what needs to be done just before the dish is served.
F. When the dish is ready, it is served to the actor Guest. When it has been eaten, the actor is expected to attract the waiter’s attention in order to pay.
G. Once payment has been made, Guest can fetch her/his coat from the cloakroom and leave the restaurant. Then use case is then complete.

Use cases - in Usability Engineering & HCI:

"A task scenario is a sequence of system-user interactions and activities." (-> a main basis for defect detection!)

Example

<table>
<thead>
<tr>
<th>Label</th>
<th>Task Scenario #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Goal</td>
<td>To see if context switching is flexible, in moving between data and process</td>
</tr>
<tr>
<td>Starting Point</td>
<td>The system is set-up, the architect is about to use the tool.</td>
</tr>
<tr>
<td>Intermediary Situation:</td>
<td>The system load goes high.</td>
</tr>
</tbody>
</table>
A high-level script

Initiator determines participants
Initiator prescribes date range
Initiator asks for needed information
Participants respond
Scheduler sets meeting & inform everybody
**Episode 1: Initiation**
Initiator determines important, active & other participants
Initiator prescribes date range
Initiator asks for preference set from potential participants
Initiator asks for exclusion set from potential participants
Initiator asks active participants for equipment reqs
Initiator asks active participants for location preferences
Initiator asks important participants for location prefs.

**Episode 2: Responding**
Participants respond to requests for pref. & excl. sets
Active participants respond for equipment reqs.
Important participants respond for preferred loc.

**Episode 3: Scheduling**
Scheduler chooses meeting time
Scheduler chooses location

**Episode 4: Reserving & Notification**
Scheduler reserves room & equipment
Scheduler notifies participants & initiator of meeting

**Sample use cases**
- conflicts
- reminders to late participants, resource manager
- confirmation
- changes in preference, exclusion) sets
- dropout
- cancel meeting
- reschedule meeting
- changes of active participants and/or equipments
- substitute active and/or important participants
- multiple booking
## Use cases

**Sample use cases**

- conflicts
- reminders to late participants, resource manager
- confirmation
- changes in preference, exclusion sets
- dropout
- cancel meeting
- reschedule meeting
- changes of active participants and/or equipments
- substitute active and/or important participants
- multiple booking
- meeting bumped by more important meeting

## Episodes

### Episode 1: Initiation
- Initiator determines important, active & other participants
- Initiator prescribes date range
- Initiator asks for needed information
- Initiator asks for exclusion set from potential participants
- Initiator asks for preference set from potential participants
- Initiator asks active participants for equipment requires
- Initiator asks active participants for location preferences
- Initiator asks important participants for location preferences
- Initiator prescribes date range

### Episode 2: Responding
- Participants respond to requests for pref. & excl. sets
- Active participants respond for equipment requires
- Important participants responds for preferred loc.

### Episode 3: Scheduling
- Scheduler sets meeting and informs everybody
- Scheduler reserves room and equipment
- Scheduler reserves location
- Scheduler reserves equipment

### Episode 4: Reserving & Notification
- Initiator asks important participants for location prefs.
- Initiator asks active participants for location preferences
- Initiator asks for exclusion set from potential participants
- Initiator asks for preference set from potential participants
- Initiator prescribes date range

## Scripts

### A high-level script

- Initiator determines participants
- Initiator prescribes date range
- Initiator asks for needed information
- Participants respond
- Scheduler sets meeting and informs everybody

### A high-level script b

- Initiator determines participants
- Initiator prescribes date range
- **Scheduler** asks for needed information
- Participants respond
- Scheduler sets meeting and informs everybody
Episode 1: Initiation
Initiator determines important, active & other participants
Initiator prescribes date range

Episode 1b: Initiation by Scheduler
Scheduler asks for preference set from potential participants
exclusion set from potential participants
active participants for equipment reqs
active participants for location preferences
important participants for location prefs.

Episode 2: Responding
Participants respond to requests for pref. & excl. sets
Active participants respond for equipment reqs.
Important participants respond for preferred loc.

Episode 3: Scheduling
Scheduler chooses meeting time
Scheduler chooses location

Episode 4: Reserving & Notification
Scheduler reserves room & equipment
Scheduler notifies participants & initiator of meeting

Use cases
Sample use cases:
- conflicts
- reminders to late participants, resource manager
- confirmation
- changes in preference, exclusion sets
- dropout
- cancel meeting
- reschedule meeting
- changes of active participants and/or equipments
- substitute active and/or important participants
- multiple booking
- meeting bumped by more important meeting

Scripts
- A high-level script b
- Initiator determines participants
- Initiator prescribes date range
- Scheduler asks for needed information
- Participants respond
- Scheduler sets meeting & inform everybody

Episode 1: Initiation
Initiator determines important, active & other participants
Initiator prescribes date range

Episode 1b: Initiation by Scheduler
Scheduler asks for preference set from potential participants
exclusion set from potential participants
active participants for equipment reqs
active participants for location preferences
important participants for location prefs.

Episode 2: Responding
Participants respond to requests for pref. & excl. sets
Active participants respond for equipment reqs.
Important participants respond for preferred loc.

Episode 3: Scheduling
Scheduler chooses meeting time
Scheduler chooses location

Episode 4: Reserving & Notification
Scheduler reserves room & equipment
Scheduler notifies participants & initiator of meeting
Scenario Instantiation - Script for: *Alice Goes To Wonderland*

Initiator determines important, active & other participants
- Ian
- Martha
- Alice
- Olga

Initiator prescribes date range
- Mon-Fri next week (it's Monday p.m. now)

Scheduler asks for preference set from potential participants
- Martha
- Alice
- Olga

Scheduler asks for exclusion set from potential participants
- Alice

Scheduler asks active participants for equipment reqs
- Alice

Scheduler asks active participants for location preferences
- Martha

Scheduler asks important participants for location prefs.
- Martha

Participants respond to requests for pref. & excl. sets
- Martha: Tue-Fri afternoon, Mon morning
- Alice: Mon-Fri next week
- Olga: Mon-Thu, Fri afternoon

Active participants respond for equipment reqs.
- Ian responds for equipment reqs.

Important participants responds for preferred loc.
- Martha responds for preferred loc.

Scheduler chooses meeting time
- Mon morning

Scheduler chooses location
- Mon-Fri next week

Scheduler reserves room & equipment

Scheduler notifies participants & initiator of meeting

Conflict -> new episodes, use cases, scripts

### Detailed Script (Action Diagram)

Initiator determines important, active & other participants
- Initiator sets date range

Scheduler asks for exclusion and preference sets from potential participants
- Scheduler asks active participants for equipment requirements
- Scheduler asks active and important participants for location preferences

Participants respond

Detection of conflict

Scheduler sets meeting
- and informs everybody
changes in absolute exclusion dates

individual negotiations

broadcast conflicts

inform initiator of conflicts
Detailed Script (Action Diagram)

1. Initiator determines important, active & other participants
2. Initiator sets date range
3. Scheduler asks for exclusion and preference sets from potential participants
4. Scheduler asks active participants for equipment requirements
5. Scheduler asks active and important participants for location preferences
6. Changes in absolute exclusion dates
7. Reminders to late participants
8. Participants respond
9. Detection of conflict
10. Notify conflicts
11. Notify failure
12. Unsuccessful meeting?
Go through an iterative process of refinement:

- Challenge requirements (specifications)
- Add use cases, episodal structures & conjoin them to construct scripts
- Challenge scenarios

Success of Scenarios

- largely depends on how well questions are posed

  but, generation of questions can be hard

  - categorize question types
  - select and pose questions systematically, while also using bottom-up approach

- Answers to questions:

  - might be given by the analyst and users through analysis of scenarios
  - refinements of scenarios
  - refinements of existing requirements
  - discovery of missing requirements
## Question Types

- **"What-if":**
  - pursue hypothetical "what could go wrong?" lines of reasoning
  - "What could go wrong with participants’ response to the date set?"
    - (i) Participants submit consistent preferred date set
    - (ii) Participants submit inconsistent preferred & exclusion **(Inconsistent sets)**
    - (iii) Participants submit preferred date set late **(Slow Responder)**
    - (iv) Participants do not submit preferences **(No Response)**

- **"Who":**
  - "Who initiates a meeting?"
    - (i) An initiator (Person)
    - (ii) An initiator (Person) & The Meeting Scheduler
    - (i) The Meeting Scheduler

## Question Types

- **"What-kinds-of":**
  - "What kinds of meeting should be supported?"
    - (i) One-shot meeting
    - (ii) Periodic meeting

- **"When":**
  - "If a potential meeting attendee does not respond, at what time should the scheduler go ahead & schedule the meeting?"
    - **drop-dead date**

- **"How-to":**
  - "How can participants resolve conflicts?"
    - (ii) An initiator (Person) & The Meeting Scheduler
    - (i) The Meeting Scheduler
Success of Scenarios: Revisited

"Potentially endless chains of questions & answers"
"Potentially an infinite number of scenarios"
"Any scenario set is necessarily incomplete"

( **Prioritize scenarios** (/requirements) )

- high frequency scenarios
- critical scenarios (benefits, costs, risks)

( **Pruning** )

divide the scenario space into 3 mutually exclusive sets:
- discarded set: of scenarios which won’t be considered further
- selected set:
- undecided set:

( **Explore scenarios based on priorities** )

use breadth-first or depth-first accordingly
What More Is Needed?

- **Multiple scenarios**
  
  - Multiple classes of users  
    
    (Who: important, active, ordinary)
  
  - Multiple scenarios for each class of users
    
    (What-kinds-of, How-to, When, ...:
    conflict detection & resolution, responding, constraints)

- **Interacting scenarios**
  
  - Concurrency
  
  - Communications
    
    (initiator, important, active, ordinary, scheduler)
  
  - Synchronization
    
    (all responses in? -> schedule)
  
  - Events-driven branching/decisions
    
    (reminders every kth day)

Need for representational medium