

## APPENDIX D: DEFECT INDICATORS, DETECTION METHODS AND FIX RECOMMENDATIONS

In this section, we describe the verification heuristics implemented in the C&L tool. For each *quality property* and its *verification heuristics*, we define the *defect indicators*, detection methods and fix recommendations.

The *input* of the following heuristics are the *sentences* to be analyzed and the NLP *annotated information* (*tokens*, *subjects*, *direct-objects*, *indirect-objects*, *action-verbs*, *complement-action-verbs*, *modifier-action-verbs*, *modifier-subjects* and *complement-subjects*) extracted from them. *subjects*, *objects* and *action-verbs* are set of tokens.  $token_i = \{index, word, POS, lemma\}$

### Vagueness

**Verification Heuristic:** Check that a sentence does not contain vague terms (e.g. *adaptability*, *additionally*).

**Indicator:** The *sentence* (Title, Goal, Episode Sentence or Alternative Solution Step) contains *vague* words or phrases (e.g. *adequate*, *also*, *unless*, *unnecessary*, *useful*, *varying*, ...). We use the list of *vague-terms* provided by the SREE tool [70].

- **Detection Method:** Check whether a *token* (or two consecutive tokens) in an *episode sentence* (or alternative solution step or title or goal) is included in *weak-dictionary* ( $token_i.word \in \text{vague-dictionary} \mid token_i.word + token_{i+1}.word \in \text{vague-dictionary} \mid token_i.word + token_{i+1}.word + token_{i+2}.word \in \text{vague-dictionary} \mid token_i.word + token_{i+1}.word + token_{i+2}.word + token_{i+3}.word \in \text{vague-dictionary}$ ). Extraction of tokens is done by the *Stanford Parser*.
- **Fix Recommendation:** Re-describe the sentence by removing vague terms.
- **Example:** "System contact with dLibra server to obtain all **necessary** data."

### Subjectiveness

**Verification Heuristic:** Check that a sentence does not contain *compara-tive/superlative adverbs/adjectives* (e.g. *similar*, *better*).

**Indicator:** The *sentence* (Title, Goal, Episode Sentence or Alternative Solution Step) contain words like *comparative/superlative adverbs/adjectives* (e.g. *similar*, *better*, *similarly*, *best*, *as possible*).

- **Detection Method:** Check whether a *token* in an *episode sentence* (or alternative solution step or title or goal) is a *comparative/superlative adverb or adjectives* ( $token.POS == JJR \mid JJS \mid RBR \mid RBS$ ). Extraction of tokens and POS tags from the sentence is done with the help of the *Stanford Parser*.
- **Fix Recommendation:** Re-describe the sentence by removing subjective terms
- **Example:** "Allow customers to find the **best** supplier for a given order"

### Optionality

**Verification Heuristic:** Check that a sentence does not contain optional words (e.g. *as desired*, *at last*).

**Indicator:** The *sentence* (Title, Goal, Episode Sentence or Alternative Solution Step) contain words that express *optionality* (e.g. *as desired*, *at last*, *probably*, *whether*, ...). We use the list of optional-terms provided by the SREE tool [70].

- **Detection Method:** Check whether a *token* (or two consecutive tokens) in an *episode sentence* (or alternative solution step or title or goal) is included in *optional-dictionary* ( $token_i.word \in \text{optional-dictionary} \mid token_i.word + token_{i+1}.word \in \text{optional-dictionary}$ ). Extraction of tokens is done by the *Stanford Parser*.
- **Fix Recommendation:** Re-describe the sentence by removing optional terms.
- **Example:** "The MCSS shall be capable of operating on **either** one or both of its independent power supplies at any one time"

### Weakness

**Verification Heuristic:** Check that a sentence does not contain weak terms (e.g. *can*, *preferred*).

**Indicator:** The *sentence* (Title, Goal, Episode Sentence and Alternative Solution Step) contains clauses that are apt to cause *uncertainty* (e.g. *can*, *could*, *may*, *might*, ...). We use the list of weak-terms provided by the SREE tool [70].

- **Detection Method:** Check whether a *token* (or two consecutive tokens) in an *episode sentence* (or alternative solution step or title or goal) is included in *weak-dictionary* ( $token_i.word \in \text{weak-dictionary} \mid token_i.word + token_{i+1}.word \in \text{weak-dictionary}$ ). Extraction of tokens is done by the *Stanford Parser*.
- **Fix Recommendation:** Re-describe the sentence by removing weak terms.
- **Example:** "User select a client for whom new contract **will** be added"

## Multiplicity

**Verification Heuristic:** Check that a sentence does not contain conjunction or disjunction of verbs or subjects (e.g. and, or, and/or).

**Indicator:** The title contains *conjunction* or *disjunction* of verbs or subjects (e.g. and, or, and/or).

- **Detection Method:** Check whether a *token* in the title is included in *multiple-dictionary* ( $\text{token.word} \in \text{multiple-dictionary}$ ). Extraction of tokens is done by the *Stanford Parser*.
- **Fix Recommendation:** Split the sentence into multiple sentences
- **Example:** “Scenario ends when users logs out **or** select a different option”

**Indicator:** The *Episode Sentence* or *Alternative Solution Step* has more than one *subject*.

- **Detection Method:** Check whether *episode sentence* (or alternative solution step) has more than one *subject*. Extraction of subjects from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the sentence into multiple sentences
- **Example:** “**Guest** and **administrator** upload files”

**Indicator:** The *Episode Sentence* or *Alternative Solution Step* has more than one *action-verb*.

- **Detection Method:** Check whether *episode sentence* (or alternative solution step) has more than one *action-verb*. Extraction of action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the sentence into multiple sentences
- **Example:** “The customer **enters** her login information and **selects** the password reminder option”

## Implicitly

**Verification Heuristic:** Check that a sentence does not contain implicit words (e.g. anyone, he, her).

**Indicator:** The *sentence* (Title, Goal, Episode Sentence or Alternative Solution Step) do not specify the subject or object by means of its specific name but uses *pronoun* or indirect reference (e.g. anyone, anybody, anything, everyone, he, her, hers, herself).

- **Detection Method:** Check whether a *subject* or *direct-object* or *indirect-object* of an *episode sentence* (or alternative solution step or title or goal) is included in *implicit-dictionary* (e.g.  $\text{subject.word} \in \text{implicit-dictionary}$ ). Extraction of subjects and objects is done by the *Stanford parser*.
- **Fix Recommendation:** Re-describe the sentence by specifying subjects/objects by means of its specific name.
- **Example:** “Administrator types the message and posts **it**”.

## Quantifiability

**Verification Heuristic:** Check that a sentence uses quantification words in a clear way (e.g. all, any, few).

**Indicator:** The *sentence* (Title, Goal) contain words that express *quantification* (e.g. all, any, few, little, many, much, several, some). We use the list of quantity-terms provided by the SREE tool [70].

- **Detection Method:** Check whether a *token* in the title (or goal) is included in *quantity-dictionary* ( $\text{token.word} \in \text{quantity-dictionary}$ ). Extraction of tokens is done by the *Stanford parser*.
- **Fix Recommendation:** Re-describe the sentence by removing quantifiable terms.
- **Example:** “User informs **some** product”

**Indicator:** The *sentence* (Episode Sentence or Alternative Solution Step) contain words that express *quantification* (e.g. all, any, few, little, many, much, several, some) followed by *vague* words. We use the list of quantity-terms and vague-terms provided by the SREE tool [70].

- **Detection Method:** Check whether a *token* in an *episode sentence* (or alternative solution step) is included in *quantity-dictionary* ( $\text{token}_i.\text{word} \in \text{quantity-dictionary} + \text{token}_{i+1}.\text{word} \in \text{vague-dictionary}$ ). Extraction of tokens and POS tags is done by the *Stanford parser*.
- **Fix Recommendation:** Re-describe the sentence by removing quantifiable terms.
- **Example:** “User provides **all required** data”

## Minimality

**Verification Heuristic:** Check that a sentence does not contain additional information (Text after a dot, hyphen,

*semicolon or other punctuation mark*).

**Indicator:** The *sentence* (Title, Goal, Episode Sentence or Alternative Solution Step) contain a *text after* a dot, hyphen, semicolon or other punctuation mark (e.g. : ; ! ?).

- **Detection Method:** Check whether a *token* in an *episode sentence* (or alternative solution step or title or goal) is included in *non-minimal-dictionary* ( $\text{token.word} \in \text{non-minimal-dictionary}$ ). Extraction of tokens is done by the *Stanford parser*.

**Fix Recommendation:** Split the sentence into multiple sentence

**Example:** “Administrator adds more channels. Proceed to step 7”

## Atomicity

**Verification Heuristic:** Check that Title defines exactly one situation [24]

**Indicator:** The *title* contains *conjunction* or *disjunction* of *verbs* or *subjects* (e.g. and, or, and/or).

- **Detection Method:** Check whether a *token* in the *title* is included in *multiple-dictionary* ( $\text{token.word} \in \text{multiple-dictionary}$ ). Extraction of tokens is done by the *Stanford parser*.
- **Fix Recommendation:** Split the scenario into multiple scenari-os or remove one action-verbs or objects
- **Example:** “Submit and print order”

**Verification Heuristic:** Check that Goal satisfies exactly one purpose [24]

**Indicator:** The *goal* contains more than one action-verb.

- **Detection Method:** Check whether *goal* has more than one *action-verb*. Extraction of action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix recommendation:** Split the scenario into multiple scenari-os or remove one action-verb
- **Example:** “The customer enters her login information and selects the password reminder option”

**Verification Heuristic:** Check that Title contains a verb in infinitive form and an object [9][24]

**Indicator:** Unnecessary *subjects* in the *Title*

- **Detection Method:** Check whether *title* has at least one *subject*. Extraction of subjects from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** It is not necessary subjects in Title
- **Example:** “User submit order”

**Indicator:** Missing *object* in the *Title*

- **Detection Method:** Check whether *title* is described without a *direct-object* (or indirect-object). Extraction of objects from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform an Object after the Action-Verb
- **Example:** “Search”

**Indicator:** Missing *action-verb* in the *Title*

- **Detection Method:** Check whether *title* is described without an *action-verb*. Extraction of verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform an Action-Verb in infinitive form
- **Example:** “Order”

**Indicator:** The *Title* contains more than one *action-verb*

- **Detection Method:** Check whether *title* has more than one *action-verb*. Extraction of action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the scenario into multiple scenarios or remove one action-verb
- **Example:** “Submit and register order”

**Indicator:** *Action-Verb* in the *Title* is not in INFINITIVE (base) FORM

- **Detection Method:** Check whether the *action-verb* of the *title* is not in infinitive form ( $\text{token.POS} \neq \text{VB} \mid \text{VBP}$ ). Extraction of action-verbs and POS tags from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform an Action-Verb in infinitive form
- **Example:** “User submits order”

## Simplicity

**Verification Heuristic:** Check that Episode-Sentence is described from user point of view (Subject + present simple tense and active form of verb + Object), or by another scenario (infinitive verb – base form + Object) [8][9][24][42]

**Indicator:** Missing subject in the Episode Sentence

- **Detection Method:** Check whether *episode sentence* is described without a *subject*. Extraction of subjects from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** IF sentence do not reference another scenario THEN Inform who (Subject) performs the Action-Verb
- **Example:** “**register** order”

**Indicator:** Missing object in the Episode Sentence

- **Detection Method:** Check whether *episode sentence* is described without a *direct-object* (or indirect-object). Extraction of objects from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform who (Object) is impacted by the Action-Verb
- **Example:** “The system **prints**”

**Indicator:** Missing action-verb in the Episode Sentence

- **Detection Method:** Check whether *episode sentence* is described without an *action-verb*. Extraction of verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform an Action-Verb an action-verb in the present simple tense and active form
- **Example:** “The system **is** online”

**Indicator:** The Episode Sentence contains more than one subject

- **Detection Method:** Check whether *episode sentence* has more than one *subject*. Extraction of subjects from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the sentence into multiple sentences
- **Example:** “**Guest** and **administrator** upload files”

**Indicator:** The Episode Sentence contains more than one sentence

- **Detection Method:** Check whether *episode sentence* is described by more than one *sentence*. Extraction of sentences from the episode sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the sentence into multiple sentences
- **Example:** “User sends the order. The system ends”

**Indicator:** The Episode Sentence contains more than one action-verb

- **Detection Method:** Check whether *episode sentence* has more than one *action-verb*. Extraction of action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the sentence into multiple sentences
- **Example:** “The customer **enters** her login information and **selects** the password reminder option”

**Indicator:** The Episode Sentence contains an action-verb not in the third form

- **Detection Method:** Check whether *action-verb* of a *episode sentence* is not in third form (token.POS != VBZ). Extraction of action-verbs and POS tags from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** IF sentence do not reference another scenario THEN Use an action-verb in the present simple tense and active form
- **Example:** “The system **broadcast** the order to the suppliers”

**Indicator:** The Episode Sentence contains more than one complement-action-verb

- **Detection Method:** Check whether *episode sentence* is described by complement verbs and, the number of complement-action-verbs is more than one. Extraction of complement-action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the sentence into multiple sentences
- **Example:** “User wants to **change** and **save** his pin”

**Indicator:** The Episode Sentence contains more than one modifier-action-verb

- **Detection Method:** Check whether *episode sentence* is described by modifier verbs and, the number of modifier-action-verbs is more than one. Extraction of modifier-action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the sentence into multiple sentences
- **Example:** “User signals the system to **proceed** and **save** the transaction”

**Verification Heuristic:** Check that *Alternative-Solution-Step-Sentence* is described from user point of view (present simple tense and active form of verb + Object), or by another scenario (infinitive base form of verb + Object). Optionally, it contains a Subject [8][9][24][42]

**Indicator:** Missing object in the *Alternative Solution Step*

- **Detection Method:** Check whether *alternative solution step sentence* is described without a *direct-object* (or indirect-object). Exception occurs when the *subject* is the “system” or “scenario” or “use case” (e.g. use case ends). Extraction of subjects, objects from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform who (Object) is impacted by the Action-Verb
- **Example:** “User **informs**”

**Indicator:** Missing *action-verb* in the *Alternative Solution Step*

- **Detection Method:** Check whether *alternative solution step sentence* is described without an *action-verb*. Extraction of verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform an Action-Verb an action-verb in the present simple tense and active form
- **Example:** “System **is** offline”

**Indicator:** The *Alternative Solution Step* contains more than one *subject*

- **Detection Method:** Check whether *alternative solution step sentence* has more than one *subject*. Extraction of subjects from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the solution into multiple solution steps
- **Example:** “**User** or **System** restart the sensor”

**Indicator:** The *Alternative Solution Step* contains more than one *sentence*

- **Detection Method:** Check whether *alternative solution step* is described by more than one *sentence*. Extraction of sentences from the episode sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the solution into multiple solution steps
- **Example:** “User re-describe the order. **System** saves the order”

**Indicator:** The *Alternative Solution Step* contains more than one *action-verb*

- **Detection Method:** Check whether *alternative solution step sentence* has more than one *action-verb*. Extraction of action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the solution into multiple solution steps
- **Example:** “User **describes** and **saves** the order”

**Indicator:** The *Alternative Solution Step* contains an *action-verb* not in the third or infinitive form

- **Detection Method:** Check whether *action-verb* of an *alternative solution step sentence* is not in third or infinitive form (*token.POS != VB | VBP | VBZ*). Extraction of action-verbs and POS tags from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** IF sentence do not reference another scenario THEN Use an action-verb in the present simple tense and active form (or infinitive form)
- **Example:** “System is **returning** to step 1”

**Indicator:** The *Alternative Solution Step* contains more than one *complement-action-verb*

- **Detection Method:** Check whether *alternative solution step sentence* is described by complement verbs and, the number of *complement-action-verbs* is more than one. Extraction of complement-action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the solution into multiple solution steps
- **Example:** “User wants to **change** his pin and **register** a new order”

**Indicator:** The *Alternative Solution Step* contains more than one *modifier-action-verb*

- **Detection Method:** Check whether *alternative solution step sentence* is described by modifier verbs and, the number of *modifier-action-verbs* is more than one. Extraction of modifier-action-verbs from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Split the solution into multiple solution steps
- **Example:** “User signals the system to **restart** the sensor and **rollback** the transaction”

**Verification Heuristic:** Check that *Title* does not contain extra unnecessary information [29]

**Indicator:** The *Title* contains unnecessary information

- **Detection Method:** Check whether *title* contains *text between parentheses* or *text representing an URI*. Extraction of parentheses and URIs is done by *Regular Expressions*.
- **Fix Recommendation:** Remove unnecessary information



- **Example:** “Create order [\(see http://.....\)](#)”

**Verification Heuristic:** Check that Episode coincidence only takes place in different situations [24]

**Indicator:** Duplicated Episode Sentence

- **Detection Method:** Check whether several *episodes* have similar *sentences* (subject + predicate). Extraction of sentence is done by *Regular Expressions*. Comparison between any two sentences is done by measuring the *Levenshtein’s distance*.
- **Fix Recommendation:** Remove or re-write one episode
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. Supplier submits a Bid
3. Supplier submit a Bid
ALTERNATES/EXCEPTIONS:

```

**Indicator:** Duplicated Episode Id/Step

- **Detection Method:** Check whether several *episodes* have the same *step* (Id). Extraction of step/id is done by *Regular Expressions*.
- **Fix Recommendation:** Remove or re-write one episode
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. Supplier submits a Bid
2. Supplier submit a Bid
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Check that episodes involving validation are described using the verbs *verify/validate/ensure/establish* and followed by *that*; i.e., avoid verbs like *check/see* followed by *If/Whether*. Complicated validation steps can confuse the user and be difficult to understand [8][9][24]

**Indicator:** The *Episode Sentence* involves a validation action and it is hard to understand and follow (contain structures like checks if / see whether)

- **Detection Method:** Search for specific keywords in an *episode sentence*, such as “*check*” | “*see*” followed by “*if*” | “*whether*”. Extraction of keywords is done by *Regular Expressions*.
- **Fix Recommendation:** Instead, re-write using the optimistic scenario, use one of the other validation verbs (*verify* / *validate* / *ensure* / *establish* followed by *that*) or relocate conditions and their actions to alternate/exception flow section
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. System checks whether the movement is valid
3. Supplier submit a Bid
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Check that nested IF statement is not used in a Conditional Episode, i.e., it can confuse the user and be difficult to read [8][11]

**Indicator:** More than one *Episode-Sentence* inside a nested IF structure

- **Detection Method:** Check whether two consecutive episodes have similar steps/Ids, i.e., second episode step = first episode step + “.” + (*digit*)<sup>+</sup>. Extraction of step/Id is done by *Regular Expressions*.
- **Fix Recommendation:** Create a new scenario and extract the sequence to it, or It should be in a separate Alternate/Exception flow section

- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. If user is valid:
2.1. System process the Bid
2.2. System broadcasts the Bid
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Check that alternative is handled by a simple action [24], i.e, if the interruption is treated by a sequence of steps (>3), this sequence should be extracted to a separate scenario [8]

**Indicator:** The Alternative Solution has too many steps (> 3)

- **Detection Method:** Check whether the number of steps in an alternative solution is more than 3 (alternative.solution.length > 3). Extraction of steps is done by *String Searching* and *Regular Expressions*.
- **Fix Recommendation:** Extract the sequence to a separated scenario
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
ALTERNATES/EXCEPTIONS:
1.a. Order is not valid:
1.a.1. System process the Bid
1.a.2. System cancels the Bid
1.a.3. ...
1.a.4. System exits

```

**Verification Heuristic:** Check that every alternative flow returns to a specific episode of the main flow or finishes the scenario [40]

**Indicator:** The Alternative does not return to the main flow in the last solution step

- **Detection Method:** Search for specific keywords in an alternative solution step, such as “(GO | BACK | RETURN | RESUME) + TO + (STEP | EPISODE)? + <Step>”, where <Step> is an episode step/Id and the alternative solution step is not the last element in the alternative solution collection. Extraction of keywords is done by *Regular Expressions*.
- **Fix Recommendation:** Move the solution step with GO TO to the last position
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. ...
3. ...
ALTERNATES/EXCEPTIONS:
2.a. Order is not valid:
2.a.1. System process the Bid
2.a.2. System goto step 1
2.a.3. System cancels the Bid

```

**Indicator:** The Alternative returns to the main flow using an invalid episode Id/Step

- **Detection Method:** Search for specific keywords in an alternative solution step, such as “(GO | BACK | RETURN | RESUME) + TO + (STEP | EPISODE)? + <Step>”, where <Step> is an invalid (not exist) episode step/Id. Extraction of keywords is done by *Regular Expressions*.
- **Fix Recommendation:** Inform a valid episode Id/Step
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. Supplier receives the Order and examines it
  2. ...
  3. ...
ALTERNATES/EXCEPTIONS:
  2.a. Order is not valid:
    2.a.1. System process the Bid
    2.a.2. System goto step 4

```

## **Uniformity**

**Verification Heuristic:** *Ensure that Title is present [24]*

**Indicator:** Missing Title

- **Detection Method:** Check whether the sentence describing a *title* is empty. Extraction of sentence is done by *String Searching*.
- **Fix Recommendation:** Inform the Title
- **Example:**

**Verification Heuristic:** *Ensure that Goal is present [24]*

**Indicator:** Missing Goal

- **Detection Method:** Check whether the sentence describing a *goal* is empty. Extraction of sentence is done by *String Searching*.
- **Fix Recommendation:** Inform the Goal
- **Example:**

**Verification Heuristic:** *Check the existence of more than one Actor per Scenario [24]*

**Indicator:** Missing Actors

- **Detection Method:** Check whether *actors* is empty (`actors.length == 0`). Extraction of actors is done by *String Searching* and *Regular Expressions*.
- **Fix Recommendation:** Inform at least one Actor
- **Example:**

**Verification Heuristic:** *Ensure that Context contains its relevant sub-components [24]*

**Indicator:** Context does not contain its relevant subcomponents

- **Detection Method:** Check whether scenario pre-condition is empty (and *post-condition* is empty and *temporal-location* is empty and *geographical-location* is empty). Extraction of context subcomponents is done by *String Searching* and *Regular Expressions*.
- **Fix Recommendation:** Inform at least one Pre-condition, Postcondition, Temporal Location or Geographical Location
- **Example:**

```

TITLE: Submit order
CONTEXT:
  Pre-condition: 
  Post-condition: 
ACTOR: System
RESOURCE:
EPISODES:
  1. Supplier receives the Order and examines it
  2. System ...
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** *Check the existence of more than one Episode per Scenario [24];*

**Indicator:** Missing Episodes

- **Detection Method:** Check whether *episodes* is empty (`episodes.length == 0`). Extraction of episodes is done by *String Searching* and *Regular Expressions*.
- **Fix Recommendation:** Inform at least one Episode



- **Example:**

**Verification Heuristic:** *Ensure that Episode contains its relevant parts [24]*

**Indicator:** The *Episode* does not contain an Id/Step

- **Detection Method:** Check whether the *step/Id* of an *episode* is empty. Extraction of step/id is done by *Regular Expressions*.
- **Fix Recommendation:** Inform at least: Id/Step and Sentence
- **Example:**

```
TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  Supplier receives the Order and examines it
  System ...
ALTERNATES/EXCEPTIONS:
```

**Indicator:** The *Episode* does not contain a Sentence

- **Detection Method:** Check whether the *sentence* of an *episode* is empty. Extraction of sentence is done by *Regular Expressions*.
- **Fix Recommendation:** Inform at least: Id/Step and Sentence
- **Example:**

```
TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. Supplier receives the Order and examines it
  2. IF user is valid
ALTERNATES/EXCEPTIONS:
```

**Indicator:** The *Conditional* or *Loop Episode* does not contain its Conditions

- **Detection Method:** Check whether an *episode* is “conditional” or “loop”, and its *condition* is empty. Extraction of episode type and condition is done by *String Searching* and *Regular Expressions*.
- **Fix Recommendation:** IF episode is Conditional or Loop THEN inform at least: Id, Condition and Sentence
- **Example:**

```
TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. Supplier receives the Order and examines it
  2. System searches the database, when
ALTERNATES/EXCEPTIONS:
```

**Verification Heuristic:** *Ensure that non-sequential episodes construct have a begin and an end keywords (e.g. #)*

**Indicator:** Missing *end* instruction in *Non-sequential Construct* (episodes delimited by # ... #)

- **Detection Method:** Given an *episode*  $e_i$  that begins (starts with the keyword “#”) a non-sequential group, check there exist an *episode*  $e_j$  that ends (ends with the keyword “#”) the group ( $i < j$ ). If there not exist  $e_j$ , the non-sequential group of episodes is incomplete. Identification of keywords (“#”) is done by *String Searching*.
- **Fix Recommendation:** Complete the non-sequential construct: begin and end keywords # ... #
- **Example:**

```
TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. Supplier receives the Order and #it
  2. ...
  3. #Local supplier ...
  4. International supplier #
ALTERNATES/EXCEPTIONS:
```

**Verification Heuristic:** *Ensure that Alternative contains its relevant parts [24]*

**Indicator:** The *Alternate/Exception* does not contain an Id/StepRef

- **Detection Method:** Check whether the *stepRef/Id* of an *alternative* is empty. Extraction of *stepRef/id* is done by *Regular Expressions*.
- **Fix Recommendation:** Inform at least: Id/StepRef, cause and Sentence
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. ...
3. ...
ALTERNATES/EXCEPTIONS:
Order is not valid:
System process the Bid

```

**Indicator:** The *Alternate/Exception* does not contain a Solution

- **Detection Method:** Check whether the *solution* of an *alternative* is empty. Extraction of solutions steps is done by *Regular Expressions*.
- **Fix Recommendation:** Inform at least: Id/StepRef, cause and Sentence
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. ...
3. ...
ALTERNATES/EXCEPTIONS:
2.a. Order is not valid

```

**Indicator:** The *Alternate/Exception* does not contain its *Causes*

- **Detection Method:** Check whether the *cause* of an *alternative* is empty. Extraction of cause is done by *String Searching* and *Regular Expressions*.
- **Fix Recommendation:** Inform at least: Id/StepRef, cause and Sentence
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. ...
3. ...
ALTERNATES/EXCEPTIONS:
2.a. System cancels the order

```

## Usefulness

**Verification Heuristic:** *Check that every Actor participates in at least one episode [24]*

**Indicator:** *Actor* does not participate in the *situation* – episodes

- **Detection Method:** Check that every *actor* in *actors* is mentioned in at least one *episode sentence* (subject or direct-object or indirect-object of the sentence). Extraction of subject, direct-object and indirect-object from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Mention the actor in at least one episode
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: User
RESOURCE:
EPISODES:
  1. System receives the Order
  2. System prints the order
  3. System submit a Bid
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Check that every Resource is used in at least one episode [24]

**Indicator:** Resource does not participate in the situation - episodes

- **Detection Method:** Check that every resource in resources is mentioned in at least one episode sentence (subject or direct-object or indirect-object of the sentence). Extraction of subject, direct-object and indirect-object from the sentence is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Mention the resource in at least one episode
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE: GPS
EPISODES:
  1. System receives the Order
  2. System prints the order
  3. System submit a Bid
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Check that every Actor mentioned in episodes is included in the Actor section [24] or is the System [42] or, it is included in Resources

**Indicator:** The Episode Sentence contains an undeclared Actor

- **Detection Method:** Check whether the subject of an episode sentence is defined as an actor (or resource or is the “system”). Extraction of subject is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Include the Subject in Actors or use the “System” word
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: User
RESOURCE:
EPISODES:
  1. System receives the Order
  2. System prints the order
  3. Supplier submits a Bid
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Check that every Resource mentioned in episodes is included in the Resource [24] section or, it is included in Actors;

**Indicator:** The Episode Sentence contains undeclared Resource

- **Detection Method:** Resources are preferentially Objects. Check whether an indirect-object of an episode sentence is defined as a resource (or actor or is the “system”). Extraction of object is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Include the Indirect-Object in Resources or Actors
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: User
RESOURCE:
EPISODES:
  1. System receives the Order
  2. The Broker System sends the Bid to the Customer
  3. System prints on the screen
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Check that every Actor mentioned in alternatives is included in the Actor section [24] or is

*the System [42] or, it is included in Resources*

**Indicator:** The *Alternative Solution Step* contains undeclared Actor

- **Detection Method:** Check whether the *subject* of an *alternative solution step sentence* is defined as an **actor** (or resource or is the “system”). Extraction of subject is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Include the Subject in Actors or use the “System” word
- **Example:**

```
TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. System receives the Order
  2. System prints the order
  3. System submit a Bid
ALTERNATES/EXCEPTIONS:
  2.1 Supplier is offline
```

**Verification Heuristic:** *Ensure that step numbering between the main flow and alternative flow are consistent [25]*

**Indicator:** *Branching Episode* of an *Alternative* is missing

- **Detection Method:** Check that the *step* (<step> from <step> + <Ref>) part of an *alternative* is equal to an *episode step* in episodes Extraction of step is done by *Regular Expressions*.
- **Fix Recommendation:** Update the alternative Id/StepRef to appoint the correct episode
- **Example:**

```
TITLE: Submit order
GOAL: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. System receives the Order
  2. System prints the order
  3. System submit a Bid
ALTERNATES/EXCEPTIONS:
  5.1 Supplier is offline
```

**Verification Heuristic:** *Check the existence of more than 2 and less to 10 episodes per scenario [8][9][23]*

**Indicator:** Number of *episodes* in current scenario is less than 3 or more than 9

- **Detection Method:** Check whether the number of *episodes* is between 3 and 9. Extraction of episodes is done by *String Searching* and *Regular Expressions*.
- **Fix Recommendation:** Re-write the scenario to keep between 3 and 9 episodes
- **Example:**

```
TITLE: Submit order
GOAL: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. System receives the Order
  2. System prints the order
ALTERNATES/EXCEPTIONS:
```

## Conceptually Soundness

**Verification Heuristic:** *Check that the Title describes the Goal*

**Indicator:** The *Title* and the *Goal* does not *share action-verbs and direct-objects*

- **Detection Method:** Check whether *title* and *goal* have *common action-verbs* and *direct-objects*. When *direct-objects* is empty, use *indirect-objects*; when *action-verbs* is empty, use *complement-action-verbs* or *modifier-action-verbs*. Extraction of verbs and objects from the sentences is done with the help of the *Stanford parser*. Comparison of sentences is done by *Syntactic Similarity*.
- **Fix Recommendation:** Re-write the Title to satisfy the Goal

- **Example:**

```

TITLE: Submit order
GOAL: Print order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. System receives the Order
  2. System prints the order
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Ensure that Episodes contain only actions to be performed [24]

**Indicator:** Missing Action-Verb in the Episode Sentence

- **Detection Method:** Check whether *episode sentence* has at least one *action-verb*. Extraction of verbs from the sentences is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform an action-verb in the present simple tense and active form
- **Example:**

```

TITLE: Submit order
GOAL: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. System receives the Order
  2. System is online
ALTERNATES/EXCEPTIONS:

```

**Verification Heuristic:** Ensure that solutions in Alternatives contain only actions to be performed [24]

**Indicator:** Missing Action-Verb in the Alternative Solution Step

- **Detection Method:** Check whether *alternative solution step* has at least one *action-verb*. Extraction of verbs from the sentences is done with the help of the *Stanford parser*.
- **Fix Recommendation:** Inform an Action-Verb an action-verb in the present simple tense and active form
- **Example:**

```

TITLE: Submit order
GOAL: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
  1. System receives the Order
  2. System prints the order
  3. System submit a Bid
ALTERNATES/EXCEPTIONS:
  2.1 Supplier is offline
    2.1.1 System is online

```

## **Integrity**

**Verification Heuristic:** Check that every included scenario (Pre-condition, Post-condition, Episode sentence, Alternative solution) exists within the set of scenarios [24]

**Indicator:** Pre-condition references to a scenario that does not exist within the set of scenarios

- **Detection Method:** Check whether a *sentence* describing a *pre-condition* is *uppercase* (capital letters) and, that this *sentence* does not exist in the set of *scenarios* (compare to titles). Extraction of sentences in uppercase from a pre-condition is done by *Regular Expressions*.
- **Fix Recommendation:** Include the related scenario to the set of scenarios
- **Example:**



<b>TITLE:</b> Register Customer <b>GOAL:</b> <b>CONTEXT:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. ... <b>ALTERNATES/EXCEPTIONS:</b>	<b>TITLE:</b> Process Bid <b>GOAL:</b> <b>CONTEXT:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. ... <b>ALTERNATES/EXCEPTIONS:</b>
<b>TITLE:</b> Submit order <b>GOAL:</b> Submit order <b>CONTEXT:</b> PRE-CONDITION: <b>FILL ORDER</b> <b>ACTOR:</b> System <b>RESOURCE:</b> <b>EPISODES:</b> 1. System receives the Order 2. System prints the order 3. System submit a Bid <b>ALTERNATES/EXCEPTIONS:</b> 2.1 Supplier is offline 2.1.1 System exits	

**Indicator:** *Post-condition* references to a scenario that does not exist within the set of scenarios

- **Detection Method:** Check whether a *sentence* describing a *post-condition* is *uppercase* (capital letters) and, that this *sentence* does not exist in the set of *scenarios* (compare to titles). Extraction of sentences in uppercase from a post-condition is done by *Regular Expressions*.
- **Fix Recommendation:** Include the related scenario to the set of scenarios
- **Example:**

<b>TITLE:</b> Register Customer <b>GOAL:</b> <b>CONTEXT:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. ... <b>ALTERNATES/EXCEPTIONS:</b>	<b>TITLE:</b> Process Bid <b>GOAL:</b> <b>CONTEXT:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. ... <b>ALTERNATES/EXCEPTIONS:</b>
<b>TITLE:</b> Submit order <b>GOAL:</b> Submit order <b>CONTEXT:</b> POST-CONDITION: <b>PAY ORDER</b> <b>ACTOR:</b> System <b>RESOURCE:</b> <b>EPISODES:</b> 1. System receives the Order 2. System prints the order 3. System submit a Bid <b>ALTERNATES/EXCEPTIONS:</b> 2.1 Supplier is offline 2.1.1 System exits	

**Indicator:** *Episode sentence* references to a scenario that does not exist within the set of scenarios

- **Detection Method:** Check whether a *sentence* describing an *episode* is *uppercase* (capital letters) and, that this *sentence* does not exist in the set of *scenarios* (compare to titles). Extraction of sentences in uppercase from an episode is done by *Regular Expressions*.
- **Fix Recommendation:** Include the related scenario to the set of scenarios
- **Example:**

<b>TITLE:</b> Register Customer <b>GOAL:</b> <b>CONTEXT:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. ... <b>ALTERNATES/EXCEPTIONS:</b>	<b>TITLE:</b> Process Bid <b>GOAL:</b> <b>CONTEXT:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. ... <b>ALTERNATES/EXCEPTIONS:</b>
--	--

TITLE: Submit order
GOAL: Submit order
CONTEXT:
PRE-CONDITION:
ACTOR: System
RESOURCE:
EPISODES:
1. System receives the Order
2. System prints the order
3. <b>SUBMIT BID</b>
ALTERNATES/EXCEPTIONS:
2.1 Supplier is offline
2.1.1 System exits

**Indicator:** *Alternative solution step* references to a scenario that does not exist within the set of scenarios

- **Detection Method:** Check whether a *sentence* describing an *alternative solution step* is *uppercase* (capital letters) and, that this *sentence* does not exist in the set of *scenarios* (compare to titles). Extraction of sentences in uppercase from an alternative solution is done by *Regular Expressions*.
- **Fix Recommendation:** Include the related scenario to the set of scenarios
- **Example:**

TITLE: Register Customer GOAL: CONTEXT: ACTOR: RESOURCE: EPISODES: 1. ... ALTERNATES/EXCEPTIONS:	TITLE: Process Bid GOAL: CONTEXT: ACTOR: RESOURCE: EPISODES: 1. ... ALTERNATES/EXCEPTIONS:
TITLE: Submit order GOAL: Submit order CONTEXT: PRE-CONDITION: ACTOR: System RESOURCE: EPISODES: 1. System receives the Order 2. System prints the order 3. System submit a Bid ALTERNATES/EXCEPTIONS: 2.1 Supplier is offline 2.1.1 <b>FILL ORDER</b>	

**Verification Heuristic:** *Ensure that actions present in the Pre-conditions are already performed [24]*

**Indicator:** Missing scenario *Post-condition* (of another scenario) that satisfies the current *Pre-condition*

- **Detection Method:** Check whether a scenario *pre-condition* is not described as *post-condition* in another scenario. Comparison of sentences is done by *String Matching*.
- **Fix Recommendation:** IF the pre-condition is not an uncontrollable fact THEN describe it as post-condition of another scenario
- **Example:**

**Verification Heuristic:** *Check that Episode coincidence only takes place in different scenarios [24]*

**Indicator:** Duplicated *Episode Sentence*

- **Detection Method:** Check whether several *episodes* have similar *sentences* (subject + predicate). Extraction of sentence is done by *Regular Expressions*. Comparison between any two sentences is done by measuring the *Levenshtein's distance*.
- **Fix Recommendation:** Remove or re-write one episode
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. Supplier submits a Bid
3. Supplier submit a Bid
ALTERNATES/EXCEPTIONS:

```

**Indicator:** Duplicated Episode Id/Step

- **Detection Method:** Check whether several episodes have the same step (Id). Extraction of step/id is done by *Regular Expressions*.
- **Fix Recommendation:** Remove or re-write one episode
- **Example:**

```

TITLE: Submit order
CONTEXT:
ACTOR: System
RESOURCE:
EPISODES:
1. Supplier receives the Order and examines it
2. Supplier submits a Bid
2. Supplier submit a Bid
ALTERNATES/EXCEPTIONS:

```

## Coherency

**Verification Heuristic:** Check coherence between Pre-conditions in related scenarios [24]

**Indicator:** Pre-conditions of a related scenario are not coherent with the Pre-conditions of the main scenario

- **Detection Method:** *Difficult to be automated.*
- **Fix Recommendation:** Re-write the pre-conditions of related or main scenario
- **Example:**

**Verification Heuristic:** Check that Geographical and Temporal location of the related scenarios are equal or more restricted than those of the main scenario [24]

**Indicator:** Geographical location of a related scenario is not in the set of Geographical locations of the main scenario

- **Detection Method:** Check whether the *geographical location* of a sequentially related scenario is not described as *geographical location* in the main scenario. Comparison of sentences is done by *Levenshtein's distance*.
- **Fix Recommendation:** Re-write the Geographical locations of related scenario to be more restrict to the main scenario
- **Example:**

```

TITLE: Submit order
GOAL: Submit order
CONTEXT:
  GEOGRAPHICAL LOCATION: Location 1
ACTOR: System
RESOURCE:
EPISODES:
1. System receives the Order
2. System prints the order
3. PROCESS BID
ALTERNATES/EXCEPTIONS:
2.1 Supplier is offline
2.1.1 System exits

```

```

TITLE: Process Bid
GOAL:
CONTEXT:
  GEOGRAPHICAL LOCATION: Location 2
ACTOR:
RESOURCE:
EPISODES:
1. Customer examines the bid
2. Customer signals the system to proceed with bid
3. HANDLE PAYMENT
ALTERNATES/EXCEPTIONS:

```

**Indicator:** Temporal location of a related scenario is not in the set of Temporal locations of the main scenario

- **Detection Method:** Check whether the *temporal location* of a sequentially related scenario is not described as *temporal location* in the main scenario. Comparison of sentences is done by *Levenshtein's distance*.
- **Fix Recommendation:** Re-write the Temporal locations of related scenario to be more restrict to the main scenario
- **Example:**

<b>TITLE:</b> Submit order <b>GOAL:</b> Submit order <b>CONTEXT:</b> TEMPORAL LOCATION: Location 1 <b>ACTOR:</b> System <b>RESOURCE:</b> <b>EPISODES:</b> 1. System receives the Order 2. System prints the order 3. <b>PROCESS BID</b> <b>ALTERNATES/EXCEPTIONS:</b> 2.1 Supplier is offline 2.1.1 System exits	<b>TITLE:</b> Process Bid <b>GOAL:</b> <b>CONTEXT:</b> TEMPORAL LOCATION: <b>Location 2</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. Customer examines the bid 2. Customer signals the system to proceed with bid 3. <b>HANDLE PAYMENT</b> <b>ALTERNATES/EXCEPTIONS:</b>
--	--

**Verification Heuristic:** Check that referenced scenarios do not reference the main scenario [31] (adapted from [25])

**Indicator:** Circular inclusion (The related scenario reference in its description to the main scenario)

- **Detection Method:** Check whether a *related scenario* references in its description (episodes, alternatives, context pre-condition or context post-condition) the *title* of the main scenario. Comparison of sentences is done by *String Matching*.
- **Fix Recommendation:** Remove the reference to the main scenario (in referenced scenario)
- **Example:**

<b>TITLE:</b> Submit order <b>GOAL:</b> Submit order <b>CONTEXT:</b> PRE-CONDITION: <b>ACTOR:</b> System <b>RESOURCE:</b> <b>EPISODES:</b> 1. System receives the Order 2. System prints the order 3. <b>PROCESS BID</b> <b>ALTERNATES/EXCEPTIONS:</b> 2.1 Supplier is offline 2.1.1 System exits	<b>TITLE:</b> Process Bid <b>GOAL:</b> <b>CONTEXT:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. ... 2. <b>SUBMIT ORDER</b> <b>ALTERNATES/EXCEPTIONS:</b>
---	--

## Uniqueness

**Verification Heuristic:** Check that the *Title* of a scenario is not already included in another scenario

**Indicator:** Two scenarios have similar *Titles*

- **Detection Method:** Check whether the *title* of a scenario is equal to the *title* of another scenario. Comparison of sentences is done by *Levenshtein's distance*.
- **Fix Recommendation:** IF the sets of episodes are the same THEN remove one scenario; IF the sets of episodes are not the same THEN rename the Title of one scenario.
- **Example:**

<b>TITLE:</b> <b>Submit order</b> <b>GOAL:</b> Submit order <b>CONTEXT:</b> PRE-CONDITION: <b>ACTOR:</b> System <b>RESOURCE:</b> <b>EPISODES:</b> 1. System receives the Order 2. System prints the order 3. ... <b>ALTERNATES/EXCEPTIONS:</b> 2.1 Supplier is offline 2.1.1 System exits	<b>TITLE:</b> <b>Submit Orders</b> <b>GOAL:</b> <b>CONTEXT:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> 1. ... 2. ... <b>ALTERNATES/EXCEPTIONS:</b>
---	---

**Verification Heuristic:** Check that the *Goal* of a scenario is not already included in another scenario

**Indicator:** Two scenarios have similar *Goals*

- **Detection Method:** Check whether the *goal* of a scenario is equal to the *goal* of another scenario. Comparison of sentences is done by *Levenshtein's distance*.
- **Fix Recommendation:** IF the sets of episodes are the same THEN remove one scenario; IF the sets of episodes are

not the same THEN rename the Goal of one scenario.

- **Example:**

<p>TITLE: Submit order          GOAL: The broker Submit order to suppliers          CONTEXT:          PRE-CONDITION:          ACTOR: System          RESOURCE:          EPISODES:          1. System receives the Order          2. System prints the order          3. ...          ALTERNATES/EXCEPTIONS:          2.1 Supplier is offline          2.1.1 System exits</p>	<p>TITLE: Submit Orders          GOAL: The broker Submits order to suppliers          CONTEXT:          ACTOR:          RESOURCE:          EPISODES:          1. ...          2. ...          ALTERNATES/EXCEPTIONS:</p>
--	--

**Verification Heuristic:** Check that the Pre-condition of a scenario is not already included in another scenario

**Indicator:** Two scenarios have similar Pre-conditions

- **Detection Method:** Check whether the *pre-conditions* of a scenario are equal to a subset of the *pre-conditions* of another scenario. Comparison of sentences is done by *Levenshtein's distance*.
- **Fix Recommendation:** IF the sets of episodes are the same THEN remove one scenario
- **Example:**

<p>TITLE: Submit order          GOAL: Submit order          CONTEXT:          PRE-CONDITION: Local Supplier has submitted a bid          ACTOR: System          RESOURCE:          EPISODES:          1. System receives the Order          2. System prints the order          3. PROCESS BID          ALTERNATES/EXCEPTIONS:          2.1 Supplier is offline          2.1.1 System exits</p>	<p>TITLE: Process Bid          GOAL:          CONTEXT:          PRE-CONDITION: Local Supplier has submitted a bid          ACTOR:          RESOURCE:          EPISODES:          1. Customer examines the bid          2. Customer signals the system to proceed with bid          3. HANDLE PAYMENT          ALTERNATES/EXCEPTIONS:</p>
---	--

**Verification Heuristic:** Check that the set of Episodes of a scenario is not already included in another scenario

**Indicator:** Two scenarios have similar Episodes

- **Detection Method:** Check whether the *episodes* of a scenario are equal to a subset of the *episodes* of another scenario. Comparison of sentences is done by *Levenshtein's distance*.
- **Fix Recommendation:** IF the set of episodes of scenario\_2 is a subset of scenario\_1 THEN remove the duplicated episodes in scenario\_1 and reference to scenario\_2; IF the sets of episodes are the same THEN remove one scenario
- **Example:**

<p>TITLE: Submit order          GOAL: Submit order          CONTEXT:          PRE-CONDITION: System is online          ACTOR: System          RESOURCE:          EPISODES:          1. System receives the Order          2. System prints the order          3. Customer examines the bid          4. Customer signals the system to proceed with bid          5. HANDLE PAYMENT          ALTERNATES/EXCEPTIONS:          2.1 Supplier is offline          2.1.1 System exits</p>	<p>TITLE: Process Bid          GOAL:          CONTEXT:          PRE-CONDITION: Local Supplier has submitted a bid          ACTOR:          RESOURCE:          EPISODES:          1. Customer examines the bid          2. Customer signals the system to proceed with bid          3. HANDLE PAYMENT          ALTERNATES/EXCEPTIONS:</p>
--	--

**Verification Heuristic:** Check that two scenarios do not have similar Titles

**Indicator:** Two scenarios share action-verbs and direct-objects in their Titles

- **Detection Method:** Check whether two titles have *common action-verbs* and *direct-objects*. When direct-objects is empty, use indirect-objects. Extraction of verbs and objects from the sentences is done with the help of the *Stanford parser*. Comparison of titles is done by *Syntactic Similarity*.
- **Fix Recommendation:** IF the sets of episodes are the same THEN remove one scenario; IF the sets of episodes are



not the same THEN rename the Title of one scenario

- **Example:**

<b>TITLE:</b> Submit order <b>GOAL:</b> Submit order <b>CONTEXT:</b> <b>PRE-CONDITION:</b> <b>ACTOR:</b> System <b>RESOURCE:</b> <b>EPISODES:</b> <ol style="list-style-type: none"> <li>1. System receives the Order</li> <li>2. System prints the order</li> <li>3. PROCESS BID</li> </ol> <b>ALTERNATES/EXCEPTIONS:</b> <ol style="list-style-type: none"> <li>2.1 Supplier is offline               <ol style="list-style-type: none"> <li>2.1.1 System exits</li> </ol> </li> </ol>	<b>TITLE:</b> Customer Submits Order <b>GOAL:</b> <b>CONTEXT:</b> <b>PRE-CONDITION:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> <ol style="list-style-type: none"> <li>1. Customer examines the bid</li> <li>2. Customer signals the system to proceed with bid</li> <li>3. HANDLE PAYMENT</li> </ol> <b>ALTERNATES/EXCEPTIONS:</b>
--	---

**Indicator:** Two scenarios share action-verbs and the direct-objects (in synonymous forms) in their *Titles*

- **Detection Method:** Check whether two *titles* have *common action-verbs* and a *direct-objects*. When direct-objects is empty, use indirect-objects. When *action-verbs* or *objects* are not equal, use their *synonymous* forms. Extraction of verbs and objects from the sentences is done with the help of the *Stanford parser*. We can get the synonymous forms of the objects with the help of WordNet [72] database. Comparison of sentences is done by *Semantic Similarity*.
- **Fix Recommendation:** IF the sets of episodes are the same THEN remove one scenario; IF the sets of episodes are not the same THEN rename the Title of one scenario

- **Example:**

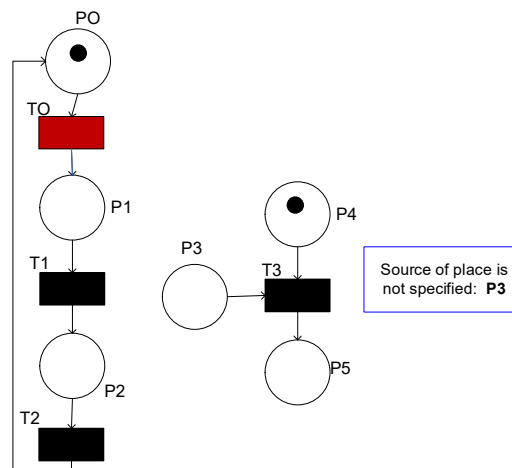
<b>TITLE:</b> Withdraw Cash <b>GOAL:</b> Submit order <b>CONTEXT:</b> <b>PRE-CONDITION:</b> <b>ACTOR:</b> System <b>RESOURCE:</b> <b>EPISODES:</b> <ol style="list-style-type: none"> <li>1. ...</li> <li>2. ...</li> <li>3. ...</li> </ol> <b>ALTERNATES/EXCEPTIONS:</b> <ol style="list-style-type: none"> <li>2.1 Supplier is offline               <ol style="list-style-type: none"> <li>2.1.1 System exits</li> </ol> </li> </ol>	<b>TITLE:</b> Withdraw money <b>GOAL:</b> <b>CONTEXT:</b> <b>PRE-CONDITION:</b> <b>ACTOR:</b> <b>RESOURCE:</b> <b>EPISODES:</b> <ol style="list-style-type: none"> <li>1. ...</li> <li>2. ...</li> <li>3. ...</li> </ol> <b>ALTERNATES/EXCEPTIONS:</b>
---	--

## Feasibility

**Verification Heuristic:** Check that is possible to derive an initial system design from related scenarios [13]

**Indicator:** Source or destination of events is not specified

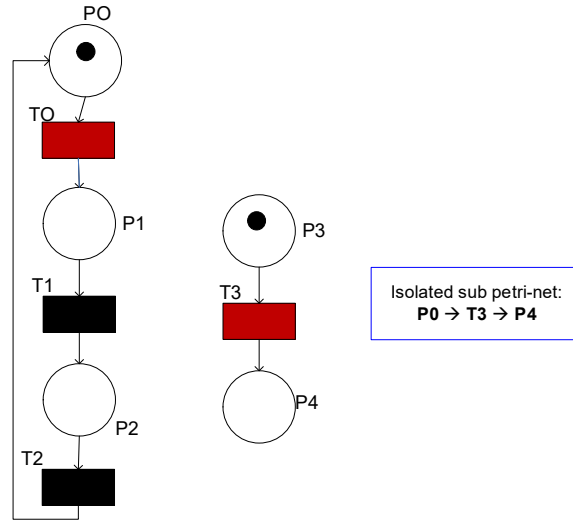
- **Detection Method:** Every *place* (transformed from episodes and alternatives) in the *Petri-Net* must have at least one input arc (that is not a pre-condition) and an output arc (that is not a post-condition). If they are missing, the tokens in the Petri-Net cannot *pass correctly*.
- **Fix Recommendation:** Inform the relevant parts of Episodes and Alternatives
- **Example:**



**Verification Heuristic:** Check that initial system design does not contain isolated sub-systems

**Indicator:** *Isolated* events – unreachable operations

- **Detection Method:** The *transitions* (transformed from episodes and alternatives) in the *Petri-Net* should interact with each other to exchange information (tokens). If there are transitions that do not interact with others, it will cause *isolated sub Petri-Nets*.
- **Fix Recommendation:** Inform the relevant parts of Episodes and Alternatives
- **Example:**

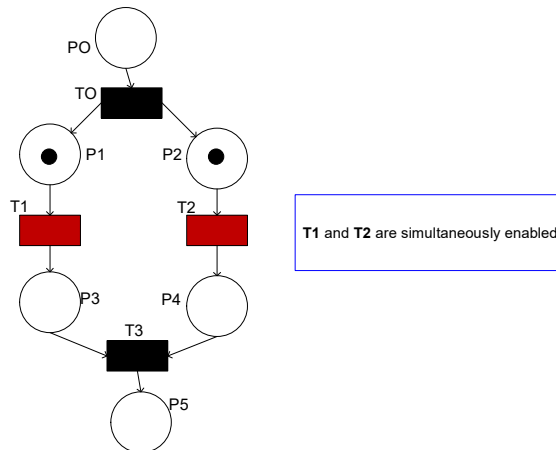


### Non-interferential

**Verification Heuristic:** Check the absence of non-determinism, i.e., a set of operations are simultaneously enabled by common pre-conditions [22]

**Indicator:** *Simultaneously* enabled operations

- **Detection Method:** Check whether the Petri-net contains *non-deterministic execution paths*, i.e., a set of *transitions* that are simultaneously enabled due to presence of tokens in their input places. *Reachability analysis* can reveal simultaneously enabled *transitions*.
- **Fix Recommendation:** Check that all pre-conditions or constraints associated to the episode/alternative corresponding to the transition are fulfilled; Notify to the next software development activities
- **Example:**

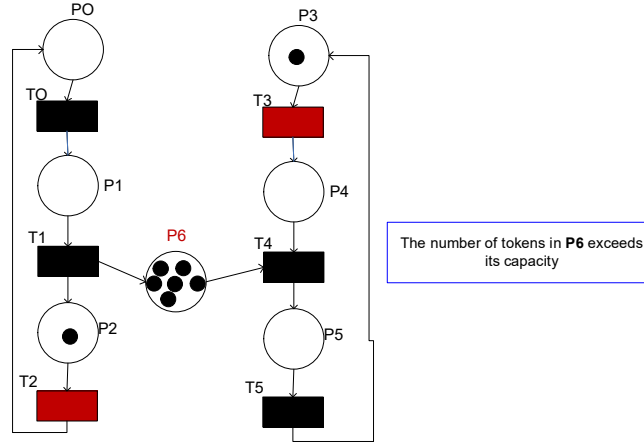


### Boundedness

**Verification Heuristic:** Check the absence of overflow, i.e., the number of elements in some communication channel or resource exceeds a finite capacity [22]

**Indicator:** *Overflowed* resources

- **Detection Method:** An *overflow* exists in a Petri-Net when the number of *tokens* in some place exceeds a finite number  $k$  for any marking reachable from initial marking  $M_0$ . If the Petri-Net is not *bounded*, overflow exists in some place [6][28]. *Reachability analysis* can reveal unbounded *places*.
- **Fix Recommendation:** Check that the overflowed resource is a critical shared resource modified by several operations or scenarios; Check that the overflowed resource capacity; Notify to the next software development activities
- **Example:**

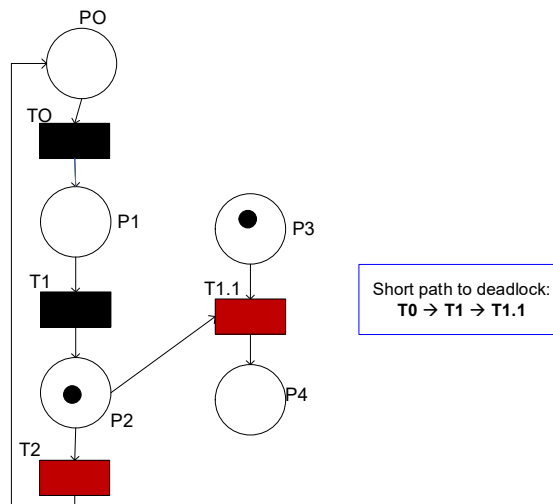


## Liveness

**Verification Heuristic:** Check the absence of paths to deadlocks [22], e.g., it could occur when an alternative flow does not return to the main flow or finish the scenario

**Indicator:** Path to *deadlock*

- **Detection Method:** Check whether the Petri-net contains a *short path* (consecutive transitions) that blocks the execution of the Petri-Net, i.e., the Petri-Net is not *deadlock free*. *Reachability analysis* can reveal short *paths* to *deadlock*.
- **Fix Recommendation:** Check whether there are shared resources modified by the scenarios and their relationships; Check that every alternative flow returns to a specific episode of the main flow or finishes the scenario; Notify to the next software development activities
- **Example:**

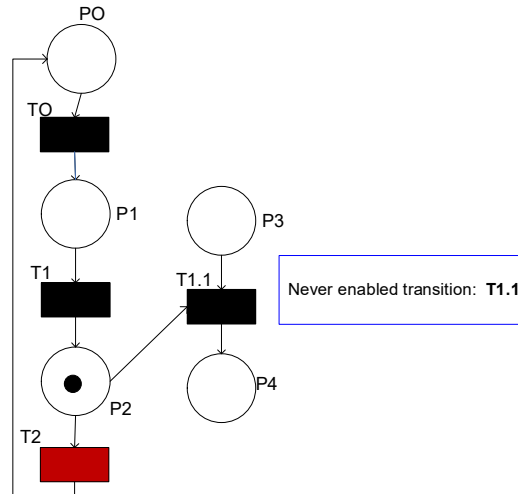


**Verification Heuristic:** Check the absence of never enabled operations, e.g., when the pre-conditions of an operation are never fulfilled

**Indicator:** Never enabled operations

- **Detection Method:** Check whether the Petri-net contains a set of *transitions* that are never enabled (unreachable code in programs). *Reachability analysis* can reveal short *never enabled transitions*.
- **Fix Recommendation:** Check that all pre-conditions, constraints, conditions or causes of the episode/alternative corresponding to the transition are fulfilled; Notify to the next software development activities

- **Example:**



### **Reversibility**

**Verification Heuristic:** Check that automatic error recovery is possible [6]

**Indicator:** Automatic error recovery is not possible.

- **Detection Method:** If the reachability analysis reveals that the *Petri-Net* is not *bounded*, not *safe* (1-bounded) and not *live*, then, the *Petri-Net* is not reversible [6][28].
- **Fix Recommendation:** Check that the performed scenarios are releasing resources, pre-conditions and constraints after completion; Check that every alternative flow returns to a specific episode of the main flow or finishes the scenario; Check the absence of deadlocks or never enabled operations
- **Example:**