1 User requirements

Each user requirement is uniquely identified by an ID, which follows the format *UR-YY-XX*, where *YY* identifies the type of the requirement, either a capacity (*CA*) or a restriction (*RE*); and *XX* identifies the sequential number of the requirement within that type, starting at *01*.

Table 1 provides the template used for the specification of the requirements, including the description of each attribute.

Table 1: User requirement template

UR-YY-XX	
Description	Detailed description of the requirement.
Necessity	Priority of the requirement for the user (*Essential*, *Convenient* or *Optional*).
Priority	Priority of the requirement for the developer (*High*, *Medium* or *Low*).
Stability	Indicates the requirement's variability trough the development process (*Constant*, *Inconstant* or *Very unstable*).
Verifiability	Ability to test the validity of the requirement (*High*, *Medium* or *Low*).

Table 2: Requirement UR-CA-01

UR-CA-01	
Description	Mi carro me lo robaron estando de romería.
Necessity	Essential
Priority	Low
Stability	Constant
Verifiability	Medium

Table 3: Requirement UR-CA-02

UR-CA-02	
Description	Mi carro me lo robaron anoche cuando dormía.
Necessity	Convenient
Priority	High
Stability	Constant
Verifiability	High

Table 4: Requirement UR-RE-01

UR-RE-01	
Description	Mi carro me lo robaron estando de romería.
Necessity	Optional
Priority	Medium
Stability	Inconstant
Verifiability	Medium

Table 5: Requirement UR-RE-02

UR-RE-02	
Description	Mi carro me lo robaron anoche cuando dormía.
Necessity	Convenient
Priority	Low
Stability	Very unstable
Verifiability	Low

Requirement UR-CA-01 is very cool.

2 System requirements

Each software requirement is uniquely identified by an ID, which follows the format *SR-YY-XX*, where *YY* identifies the type of the requirement, either functional (*FN*) or non-functional (*NF*); and *XX* identifies the sequential number of the requirement within that type, starting at *01*.

Table 6 provides the template used for the specification of the requirements, including the description of each attribute.

Table 6: Software requirement template

SR-YY-XX	
Description	Detailed description of the requirement.
Necessity	Priority of the requirement for the user (*Essential*, *Convenient* or *Optional*).
Priority	Priority of the requirement for the developer (*High*, *Medium* or *Low*).
Stability	Indicates the requirement's variability trough the development process (*Constant*, *Inconstant* or *Very unstable*).
Verifiability	Ability to test the validity of the requirement (*High*, *Medium* or *Low*).
Origin	User requirements that derived this requirement.

Table 7: Requirement SR-FN-01

SR-FN-01	
Description	¿Dónde estará mi carro?
Necessity	Convenient
Priority	Low
Stability	Constant
Verifiability	High
Origin	UR-CA-01, UR-CA-02

Table 8: Requirement SR-FN-02

SR-FN-02	
Description	¿Dónde estará mi carro?
Necessity	Convenient
Priority	Low
Stability	Constant
Verifiability	High
Origin	UR-CA-01, UR-CA-02

Table 9: Requirement SR-NF-01

SR-NF-01	
Description	¿Dónde estará mi carro?
Necessity	Optional
Priority	High
Stability	Inconstant
Verifiability	Medium
Origin	UR-RE-01

Table 10: Requirement SR-NF-02

SR-NF-02	
Description	¿Dónde estará mi carro?
Necessity	Essential
Priority	High
Stability	Very unstable
Verifiability	Low
Origin	UR-RE-02

Requirement SR-NF-02 is trully atomic.

3 Use cases

Each use case is uniquely identified by an ID, which follows the format *UC-XX*, where *XX* identifies the sequential number of the use case, starting at *01*.

Table 11 provides the template used for the specification of the use case, including the description of each attribute.

Table 11: Use case template

UC-XX	
Name	Brief description of the use case.
Actors	External agent that executes the use case.
Objetive	The use case's purpose.
Description	Steps that the external agent must take to execute the use case.
Pre-condition	Conditions that must be fulfilled *before* executing the use case.
Post-condition	Conditions that must be fulfilled *after* executing the use case.

Table 12: Use case UC-01

UC-01	
Name	Recuperar el carro.
Actors	Manolo
Objetive	Quiere recuperar su carro.
Description	1. Lo busca 2. Lo encuentra 3. Profit
Pre-condition	Se lo han robao.
Post-condition	Encarcelan al que lo ha hecho.

I think Use case UC-01 really shows what's going on in the user's head.

4 Components

Table 13 provides the template used for the specification of the components, including the description of each attribute.

Table 13: Component template

Name	
Role	Component's function in the system.
Dependencies	Components that depend on this component.
Description	Explanation of the functioning of the component.
Data	Input (*in*) and output (*out*) data of the component.
Origin	Software requirements that derived the component.

Table 14: Component 'Carro'

Carro	
Role	Ser robado
Dependencies	N/A
Description	Es mu bonito, pero propenso a ser robado.
Data	• **in:** gente • **out:** más gente
Origin	SR-FN-01

I really did a good job on Component 'Carro'.

5 Tests

Each test is uniquely identified by an ID. The ID follows the format *YYY-XX*, where *XX* identifies the sequential number of the test case, starting at *01*, and *YYY* represents the type, either *VET* (verification) or *VAT* (validation).

Table 15 provides the template used for the specification of the test case, including the description of each attribute.

Table 15: Test template

YYY-XX	
Description	Test description.
Preconditions	Conditions that must be fulfilled in order to perform the test.
Procedure	Description of the steps to take in order to perform the test.
Postconditions	Conditions that must be fulfilled after performing the test in order to pass.
Origin	Requirements that originated this test.
Evaluation	Result of the test (*OK* or *Error*).

Table 16: Test VET-01

VET-01	
Description	Verificar que mi carro verdaderamente ha desaparecío.
Preconditions	El carro era mío.
Procedure	1. Me dicen que le quitaron 2. los clavos que relucían 3. creyendo que eran de oro 4. de limpios que los tenía
Postconditions	Desaparece.
Origin	SR-FN-02
Evaluation	OK

Table 17: Test VAT-01

VAT-01	
Description	Verificar que el carro merece la pena recuperarse.
Preconditions	Mi carro era muy querido.
Procedure	1. En mi carro gasté 2. una fortuna 3. y en mis noches de amor 4. llevé la luna
Postconditions	Lo he usado.
Origin	UR-RE-02
Evaluation	OK

Qué bonito es el Test VET-01.

6 Traceability matrixes

	UR-CA-01	UR-CA-02
SR-FN-01	•	•
SR-FN-02	•	•

Figure 1: Traceability between functional requirements and capabilities

	UR-RE-01	UR-RE-02
SR-NF-01	•	
SR-NF-02		•

Figure 2: Traceability between non-functional requirements and restrictions

	SR-FN-01	SR-FN-02
Carro	•	

Figure 3: Traceability between components and functional requirements

	SR-FN-01	SR-FN-02	SR-NF-01	SR-NF-02
VET-01		●		

Figure 4: Traceability between verification tests and software requirements

	UR-CA-01	UR-CA-02	UR-RE-01	UR-RE-02
VAT-01				●

Figure 5: Traceability between validation tests and user requirements