Compan
y Logo

Robocup Wrapper Requirements Document

Prepared by

Reference
Issue/Revision
Date of Issue
Status

Company Logo

APPROVAL

Title	
Issue Number	**Revision Number**
Author	**Date**
Approved By	**Date of Approval**

CHANGE LOG

Reason for change	Issue Nr.	Revision Number	Date

CHANGE RECORD

Issue Number	Revision Number		
Reason for change	**Date**	**Pages**	**Paragraph(s)**

DISTRIBUTION

Name/Organisational Unit

Company Logo

Table of contents:

1 INTRODUCTION

The Requirements Document (RD) is dedicated to providing a comprehensive set of requirements applicable to the project together with the relevant justifications.

In case a waterfall approach to the requirements engineering is retained, the Requirements Document (RD) will be discussed at the BDR. The set of Requirements herewith described will be the baseline for the following design and development activities and associated verification.

To allow formal traceability of the different requirements, the designer shall associate to each requirement a unique identifier using a suitable methodology. Such methodology shall use a suitable set of acronyms (e.g. UN for User Needs, UR for User Requirements, SR for System Requirements) to facilitate traceability.

The approach could be different depending on the type of users involved in the project. It could be necessary to identify before User Needs, formalise them in measurable Users Requirements and map them into system Requirements; it could also happen Users Needs could be directly mapped into System Requirements or even that a skilled user already has a set of Users Requirements that can be used by the designer for the System Requirements specification.

The approach to be followed is left to the designer; justification shall be anyhow provided.

Reference Documents

Ref.	Document ID.	Title	Rev.
[RD1]			

Acronyms

Tag	Description
RD	Requirements Document
UN	User Needs
UR	User Requirements

2 USER NEEDS AND USER REQUIREMENTS

User Needs

RoboCup is an annual international robotics competition proposed and founded in 1996. The aim of the competition is to promote robotics and AI research by offering a publicly appealing – but formidable – challenge. Currently, Wits has an interest group dedicated to this competition, who use a modified version of the UTAustinVIlla Code Base which is written in C++. The user would like us(to write a Python wrapper for this code base firstly to make it easier to get started with trying different strategies and secondly because Python is a popular language for robotics research. This project will also provide insight into how other popular Python libraries which are wrappers of C code, like NumPy, work.

User Needs Description

Any user needs shall be defined on the basis of the following rules. In particular, it will be listed in a row of table as presented in Table 1.

ID	User Need	Source
UN-0100	A python coder needs a math library so that they will be able to perform the matrix operations in the python language	Python Coder
UN-0200	A python coder needs a python module that corresponds to a C++ library that enables them to calculate the Fibonacci series in python	Python Coder
UN-0300	Users have the need for a Robocup Python wrapper so that they can participate in Robocup using Python instead of C++ (because Python is easier for first time users)	Python Coder
UN-0400	Users have the need for an instance of the world model in a python class, so that they can access the world model data in coding their strategies	Python Coder
UN-0500	Users have the need for Robocup C++ objects to be Python objects so that they can access C++ data from Robocup in Python.	Python Coder
UN-0600	Users have the need for the Robocup wrapper to return chosen skill in form of int from python to C++ so that their python code can interact with the Robocup simulator	Python Coder
UN-0700	Users have the need for a C++ interface to facilitate for the transfer of data between python side and C++ side	C++ Coder
UN-0800	Users have the need to an optimised Robocup wrapper for the C++ code to interact with python code to avoid delays between the user selecting the skill in python and it executing in C++	Python Coder

Table 1 – User Needs

Where:

- **ID**: unique identification composed of a prefix, for instance "**UN"**, followed by a serial number, which could be composed of four digits (e.g., **UN-0100, UN-0200** etc.). In the initial phase the serial number could start from 0100 and will proceed in steps of 100 per requirement, in order to allow the possibility of adding new or more detailed needs during the course of the requirements definition.
- **User Need**: Describes the need in qualitative terms.
- **Source**: The user that expressed this need.

User Requirements

This part is describing in a structured form the set of statements originated by the users describing the functions, performance and capabilities that the system will bring to them during its utilisation. A mapping between User Requirements and User Needs is part of this section.

User Requirements Description

Any User requirement shall be defined on the basis of the following rules. In particular, it will be listed in a row of table as presented in Table 2.

Req. ID	User Requirement Name	Description	Justification and/or comment	User Need Ref.
UR-PERF-0100	BB Robocup Python Wrapper	**A** working Python wrapper so that enables a user to use Python language for RoboCup (instead of the more difficult language c++)	For the user to be able to interact with the simulator.	UN-0300
UR-PERF-0200	BB method access perfomance	A minimum runtime of below milliseconds must be maintained for accessing C++ methods in python.	So that there is not a delay when users use the Python wrapper	UN-0400
UR-PERF-0300	BB C++ object access minimum runtime	A minimum runtime of below milliseconds must be maintained for accessing C++ objects in python.	So that there is not a delay when users use the Python wrapper	UN-0500
UR-PERF-0400	BB Python select skill	A user must be able to select a skill in python and for the corresponding skill to be executed in C++	No Comment	UN-0600
UR-PERF-0500	Math Library	Math library coded in C++ must be created, wrapped and made callable in python	No Comment	UN-0100
UR-PERF-0600	Fibonacci Series	A user must be able to call a C++ library from python so that they can calculate the Fibonacci Series	No Comment	UN-0200

Table 2 – User requirements

Where:

- **ID**: unique identification composed of the **UR** prefix, followed by a serial number composed of four digits (e.g., **UR-0100, UR-0200** etc.). It is suggested to use for the numbering scheme the same criteria used for the User Needs (e.g. start from 0100 and proceed in steps of 100 per requirement).
- **User Requirement**: Define the requirement through a concise name
- **Description**: Describes the requirement in more details and with regard to the impact on the system definition
- **Justification**: Provides the rationale for the requirement and what are the benefits.
- **User Need Ref.**: Define the cross reference with respect to the satisfied need.

3 SYSTEM REQUIREMENTS

A system requirement is a statement typically originated by the designer about what the system shall do and/or shall be to fulfil the User Requirements (e.g. associated to constraints, environment, operational and performance features).

This section is identifying, allocating and specifying the System Requirements defined by the designer. A mapping between System Requirements and User Requirements (if applicable) or User needs is part of this section.

System Requirements

Any requirement shall be defined on the basis of the following rules. In particular, it will be listed in a row of table as presented in Table 3.

ID	System Requirement	Priority	Description	Verification Method	User Req.
SR-PRF-0100	World Model Data	M	The python wrapper must have access to world model data passed to it by the server at every world model instance while adhering to an access runtime that is below milliseconds	T	UR-PERF-0100, UR-PERF-0200, UR-PERF-0300
SR-PRF-0200	Select Skill	M	The python wrapper must be able to select skills with runtime being low the runtime of milliseconds	T	UR-PERF-0100, UR-PERF-020
SR-PRF-0300	Math Library	C	The python wrapper should be able to compute matrix operations and compute the Fibonacci series in python with runtime below milliseconds	T	UR-PERF-0500, UR-PERF-0600

Table 3 – Requirement description

Where:

- **ID**: unique identification composed of a prefix, followed by a serial number composed of four digits (eg. **SR-0100**, **SR-0200** etc.).
- **Priority**: define whether the requirement is:

  - Must have (**M**) – must be implemented in the system.
  - Should have (**S**) – must be implemented but may wait until a second increment.
  - Could have (**C**) – could be implemented but it is not central to the project objectives.
  - Wish to have (**W**) – will not be implemented but it will be considered for a future phase.
- **Description**: describes the requirement.
- **Verification Method**:
  - Inspection (**I**) – Verification by inspection shall consist of visual determination of physical characteristics. Visual inspection of either graphical interface, textual results, user manual, or equipment manufacturer specifications. It will require an analysis of the documentation and/or visual inspection, providing evidence of the correct implementation that satisfy the requirement by means of screenshot, extraction of sections from operational manuals, etc. Therefore no specific test procedure with detailed operations is envisaged.
  - Analysis (**A**) – Verification by analysis is done when other methods are not appropriate or too cumbersome to perform a verification by test. It is usually done by collecting data like test results related to some part of the system, and then, knowing the system design, an engieneering based judgement is perfomed to infer whether the verification was successful or not.
  - Demonstration (**D**) – Verification by demonstration is done verifying the behaviour of the system, either once or more than once, without special test equipment or intrumentation. Demonstration can be documented in different ways, such as with pictures or screen captures.
  - Test (**T**) - Verification tests consist of measuring product performance and functions under representative environments.