# Chapter 3- Requirements

This chapter provides a detailed description of the functional and non-functional requirements for WEAVE. Use case diagrams show how the system is expected to be used.

## 3.1 Background

This project builds on an existing system- IWE- for facilitating the use of worked examples in educational context. This system is described in Chapter 2. The evaluation on IWE clearly shows that such software would be a valuable asset contributing to the learning process of students. Due to the overly complicated procedure required to deploy IWE in schools while it is in the form of a Java standalone application, the need to turn it into a more easily deployable online version arose. Since it is crucial that students use this application in the schools and authors of such examples would rather use it on their personal machines, the focus of this project is the student interface. In addition, interviews with highly motivated teachers, who are part of PLAN C project (reference), have identified the need for one more interface to be used at schools. In order to improve their teaching practice, these teachers would benefit from feedback on how students in their classes use these worked examples. Information that would be valuable for them includes: identification of which students interacted with which examples; aggregated information on answers selected for multiple choice questions and the students that selected each answer; information about the average time spent at each step of an example as per the whole class; information about the time spent at each step of an example as per an individual student of a class; summary data of the total time spent at an example and the last step reached by each student in the class.

In order to build up on IWE, this project aims to achieve four goals:

G1- build a system that is interoperable with the author interface of IWE, i.e. ensure no changes to the way worked examples are created should be needed in order to add these examples to the new system

G2- provide an interface for teachers that would help them gain more information on how the worked examples are used

G3- replicate as close as possible the student interface of IWE

G4- add features to the student interface to store personalised usage data as opposed to anonymous one

## 3.2 Functional requirements

The functional requirements for G1, G2, G3 and G4 are described in sections 3.2.1, 3.2.2, 3.2.3 and 3.2.4 respectively. Each requirement is classified according to the *MoSCoW* classification method (reference). The categories considered are:

- **must-have**- requirements that are crucial for the achievement of the goal of this project and must be implemented

- **should-have**- requirements that are considered to be important but not necessarily crucial for achieving the goal of this project and should be implemented

- **could have**- requirements that have been identified as features that would add further value to the prototype but are thought of as stand-out ones rather than ones contributing to the correct functioning of the prototype and may not be implemented due to constraints

The **would-like** category coming from the **W** in MoSCoW is not part of the classification methods used for this project due to the fact that all the requirements fit comfortably in the other categories.

3.2.1 Interoperability with the existing author interface

The prototype:

* **must** be able to parse an XML file containing the fragmented problem specifications of the worked examples and their solutions.
* **must** be able to parse an XML document containing information about individual steps of the worked examples (e.g. which fragments of a document must be shown/hidden/highlighted, the explanation associated with a step and a question if one was provided).
* **must** be able to parse an XML document containing information about the layout of worked examples (e.g. number of panels for the example, their order and problem solutions associated with each panel).
* **must** be able to parse an XML document containing information about the styling associated with each example (e.g. font style, font size, etc.).
* **must** be able to support easy addition of new worked examples.
* **must** be able to support easy modification of existing worked examples.

3.2.2 Teacher interface requirements

The teacher:

* **must** be able to register with a username and password.
* **must** be able to login/logout of the system.
* **must** be able to create groups for their students
  + **must** be able to specify the name of the group.
  + **should** be able to specify the number of students for the group.
  + **could** be able to specify the academic year this group belongs to.
* **should** be able to update existing groups by adding more students to them.
* **should** be able to view a printable list showing the student ids for a group.
* **should** be able to view information on the average time spent by all students at each step.
* **should** be able to view information on the number of times an answer for a question has been chosen.
* **could** be able to view information on the average time spent by a particular student at each step.
* **could** be able to view information on the list of students that chose a particular answer to a question.
* **could** be able to view information on the total time a student spent on an example.
* **could** be able to view information on the last step a student reached on an example .
* **could** be able to delete existing group.

3.2.3 Replication of the IWE student interface

The prototype:

* **must** enable the student to select a worked example from a list of existing examples.
* **must** support multiple panels for the different parts of the problem solution.
* **must** contain a dedicated area for the explanation.
* **must** support showing/hiding/highlighting of fragments.
* **must** support the option to ask students questions.
* **must** enable the student to go back and forwards through steps.
* **should** record time spent at a step.
* **should** record answers to questions
* **should** enable the student to reset the example there are working on.
* **should** highlight the newly introduced fragments at each step.
* **could** provide a means for drawing the student’s attention to the newly introduced fragments.

3.2.4 Additional features needed for the student interface

The prototype:

* **must** allow the student to use the system without any identifying information.
* **should** be able to connect the usage information stored for a student to their teacher.
* **should** be able to connect the usage information stored for a student to their teacher and the current academic year.
* **should** be able to connect the usage information stored for a student to their teacher, the current academic year and a group they were allocated to.
* **should** connect the usage information stored for a student to their teacher, the current academic year, a group they were allocated to and a student id.

# 3.3 Non-functional requirements:

* The prototype **must** be easy to use.
* The worked examples **must** fit the entire screen.
* The size of the area showing the worked examples **must** not exceed the size of the screen.
* A modification to a worked example **must** not affect students doing the same example.
* Concurrent storage of information for multiple students connected to the same teacher **must** be supported.
* The student interface **should** include a tutorial on how to use the system.
* The teacher interface **should** provide information on how to use each feature.
* The panels showing the problem content **should** be resizable.
* The explanation area **should** be resizable.
* Shortcuts for easier transition between steps **could** be added.