# Chapter 3- Requirements

This chapter provides a detailed description of the functional and non-functional requirements for WEAVE.

## 3.1 Background

As described in Chapter 2, this Level 4 project builds on an existing system for facilitating the use of worked examples in educational context. 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. 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 and to be able to provide high quality feedback to their pupils, these teachers would benefit from knowing 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 upon IWE, this project aims to achieve four goals:

G1- build a web-based viewing system that is interoperable with the author interface of IWE, i.e. ensure that worked examples created using the old system can be viewed in the new system.

G2- provide an interface for teachers that will help them gain more information on how the worked examples are used by their own pupils

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

G4- ensure that worked example authors can view usage data in an anonymous manner, such that individual pupils, classes or schools are not identifiable

## 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 existing system uses XML files to store the worked examples. The structure of these files is shown on Figure 2.2 in the Background chapter. The web-based system will need to read in worked examples from these data files. Furthermore, feedback from students and teachers will inevitably lead to changes being required in some of the worked examples. The existing authoring tool supports editing of worked examples, and it is expected that it will still be used to make such changes. WEAVE will need to be able to support these changes. Due to the fact that the update model of IWE is destructive- no versioning of the examples is supported- and that WEAVE does not provide means for modifying examples, the update model will follow the one of the old system.

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 into the web-based system of new worked examples created using the old authoring tool.
* **must** be able to incorporate new versions of worked examples already installed in the web-based system.

3.2.2 Teacher interface requirements

Need to discuss issues around security. Why is a password required? Why named groups, students etc? Why printable lists of students? What did you get from Peter that helped guide your design here? You need to give the reader a clear sense as to WHY you have all these MUSTs. Not enough simply to state them. You have groups as MUST and number of students as SHOULD. Why? What does just groups give? What do both give? The number of students alone is not enough for him/her to identify individual students. You need to explain the whole mode of use that you agreed in consultation with Peter. The system creates anonymous very short ids for pupils. Why anonymous? Why very short? Why have the printed paper?

All of this discussion, on how you arrived at this solution, is essential precursor to stating the requirements. I don't think you need use cases and all that, but you must write down the process that led to these 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.
  + **must** be able to uniquely identify the students in each 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

One of the goals for this Level 4 project (G3) is to replicate the student interface of IWE as closely as possible. While the main features of this interface should remain unchanged, the student interface of WEAVE will need to be suobrazen with the fact that it is intended to be used in schools where the size of the screen might be relatively small.

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

This section describes the requirements for satisfying goals G2 and G4 – supporting identifiable usage information for the teacher and anonymous usage information for worked examples authors and for Computing Science education researchers. It is important that each teacher is able to link usage data to their groups/students while authors of such examples must not be able to identify by any means what the group or who the student is due to the privacy issues discussed above..

The prototype:

* **must** be able to show personalised usage data to the teacher.
* **must**  be able to show anonymous usage data to authors of worked examples and Computing Science education researchers.
* **must** allow the student to use the system without any identifying information.
* **must** be able to connect the usage information stored for a student to their teacher.
* **must** be able to connect the usage information stored for a student to their teacher and the current academic year.
* **must** 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.
* **must** 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 non-functional requirements for WEAVE are guided mostly by the web-based nature of the system and by the context it is intended for. In order for pupils to be able to study the worked examples effectively, and also due to the small workstation screen sizes found in schools, the area showing the worked example should be maximised. Furthermore, due to the step-by-step nature of worked examples, some steps may put more emphasis on the explanation while others might be more intensive in the problem specification areas so the system must be able to deal with such situations accordingly.

The web-based nature of WEAVE poses a possible problem when uploading modifications to existing examples because students might be working on these examples at the same time. Consistency must be ensured in such cases, meaning that the pupil should be able to see either the old or the new version of the example, rather than a mixture of both.

Since pupils may have not worked with such a system before, they may benefit from a brief guide on how to use WEAVE in an optimal way.

* 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. Why? Personally, I don't like to scroll a lot. Can you find some research to back this up?
* A modification to a worked example **must** not affect students doing the same example.
* 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. Why?