In this chapter we will discuss the requirements for our BioBrick modeling application. What does the application have to do? What doesnt belong in the basic functionality? What kind of programming language will we use to develop this program and when is it due? These are the type of questions we will answer. First we will go into detail on functional requirements, so what does the program do? Secondly, we will discuss non-functional requirements. These are questions like: What programming language will we use, but also how we will increase the usability of the application. Finally we will have a look at constraints.

1 Functional requirements

The following are use cases the application must be able of completing in whatever way.

1. **Circuit abstraction** The application must be able to abstract a designed circuit into a BioBrick, and then treat this as a gate, so it can be used in building of subsequent circuits/BioBricks.

2. **Protein specification** The user must be able to manually specify which protein is used to represent which signal in a circuit.

3. **Available proteins** The application must be able to present the user with an overview of available proteins to assign to signals. The available proteins must be predefined by a user.

4. **Export XML** The application must be able to export a built circuit as a BioBrick in XML (in the SBML schema).

5. **Import XML** The application must be able to import XML (in the SBML schema) into the application as a BioBrick.

6. **Interfering/invalid signals** The application must be able to notify the user that the current circuit design has protein(s) assigned to multiple signals, or still has signals which do not have a protein assigned.

2 Pseudo-requirements

Non-functional requirements are requirements about how a the application behaves and operates (usability, accessibility, availability, performance). In this section we will also cover constraints on the source-code and development (programming language, documentation, tools/frameworks, compatibility, extensibility, maintainability, testability, deadlines).

1. **Documentation** For documentation we will use LaTeXto finalize the reports and will be written in English. GIT will be used for hosting and collaborating on documentation.

2. **Tools/frameworks** The use of other software development tools has not been decided upon yet.

3. **Testability** We want to be able to test every detail. We were discussion test drive development, but we have not fully decided upon that either. In any case, there should be a lot of tests, in the form of checks and sort of unit-tests, but also for user friendliness by real people.

4. **Usability** Because the program is not made for information technology students so we will spend some time increasing the usability. However, the target group consists of scientists, and to be more specific, mainly bio-informatics experts. Because of this we will not create step-by-step tutorials. Our aim is to make the application basic enough so that a regular scientist can work with the application.

5. **Extensibility** We want to create our application in such a way that BioBricks can be exported and re-used later as part of even greater BioBricks. As for the application, it is always best practice to make the application extendable, and so we shall aim for that.

6. **Maintainability** Maintainability is actually not one of our focuses. Because the application works on its own, and is only required to be developed until the end of this semester, this is not a major concern.

7. **Accessibility** Because our application frontend will be written for the web, we will host it somewhere and it should be accessible from anywhere, if there is internet. From the time of launch of our first version until the end of the project, there should always be a testable version hosted somewhere. It's also likely that we will use modern frameworks such as Bootstrap CSS to provide a high start-level of usability, although this has not been decided yet. The use of such well developed frameworks guarantees a high accessibility of the website.

8. **Deadline** The first version is up for evaluation on the fourth of May (04-05-212). The final version of the application is due on the fifteenth of June (15-06-2012).

9. **Biological plausibility** The application does *not* have to give an biologically accurate or plausible simulation.

10. **Frontend** The GUI frontend of the application will be a website written in HTML+CSS+JavaScript. This means our application frontend will be compatible with all the big operating systems that support web browsers.

11. **Backend** The backend will be written in Java Server Pages.