Software Requirements Specification for “Visual Family Tree”

Kaya Oğuz

1 Introduction

“Visual Family Tree” is going to be a desktop application that will be used to create family trees. The users will be able to add, edit, and remove family members, set relationships between them, and find out the extended relationships by querying two members. It is planned as a standalone application that does not depend on any other application or service.

2 User Requirements

The user requirements can be detailed as follows.

Functional Requirement 1: The user shall be able to add a family member.

Rationale: Since this is a family tree application, this is the most basic functionality. The user shall be able to enter the name, surname, gender and birth date of the family member. The gender will be used when the relationships are queried. Birth date will be used to calculate the age of the family member. The name, surname and birth date are expected to uniquely define the family member; that is, we do not expect two people in the same family to be born on the same day having the same names.

Functional Requirement 2: The user shall be able to define relationships between two family members that have already been added.

Rationale: This relationship is about two family members who are either married, or had children outside marriage. It is possible for a member to have more than one relationship, and have children from each of these relationships.

Functional Requirement 3: The user shall be able to add children to the defined relationships between two family members.

Rationale: This relationship association between two family members should also keep track of the children that are born from that relationship. The parent-child relationship is based on this association.

Functional Requirement 4: The user shall be able to search for a family member.

Rationale: On large trees, it would be useful to be able to search for a family member. The search should be based on all properties of a family member.

Functional Requirement 5: The user shall be able to query the relationship between two family members.

Rationale: This functionality is expected to return the extended relationship between two family members. For example, when two family members are given (or selected on a graphical user interface) this function should return the relationship, such as uncle-nephew, on these family members.

Non-functional Requirement 1: The user shall be able to use the system after reading the manual.

Rationale: Instead of trial and error, the user shall be provided with a comprehensive documentation to use the software.

3 System Requirements

The system requirements can be detailed as follows.

Functional Requirement 6: The system shall be able to save the current family tree to a file on the file system.

Rationale: This is a basic functionality that is required to provide persistent data storage for the created family trees.

Functional Requirement 7: The system shall be able to load a family tree from a previously saved file on the file system.

Rationale: This is a basic functionality that is required to load and keep working on a previously created family tree.

Functional Requirement 8: The system shall be able to merge two family trees.

Rationale: This requirement enables the user to merge two family trees by joining them on family members that exist on both trees. The merge will replace the common family members from the second tree so that there is only one of the same family members in the final tree. When the trees do not have a common family member, the trees will still be merged but they will be without a connection. This connection can be added by adding a new family member to the new tree.

Non-functional Requirement 2: The system shall be able to run on a Windows system.

Rationale: The target platform is Windows, however, the selected programming language could provide portability to other platforms, too. The application does not depend on any operating system specific functionality.

Non-functional Requirement 3: The system shall be in Turkish.

Rationale: The system shall be in Turkish so that a wider range of extended relationships could be used, such as hala/teyze instead of just aunt. This will require the definition of these relationships to depend on the genders of parents, too.