CSC3003S Capstone Project — Stage One

Goals (Scope) [21 Marks]

|  |  |
| --- | --- |
| Project Abbreviation and Name | ViKER Interface |
| Client/Supervisor + email | Maria Keet  [mkeet@cs.uct.ac.za](mailto:mkeet@cs.uct.ac.za) |
| Tutor + email | Ryan Lazar  [lzrrya001@myuct.ac.za](mailto:lzrrya001@myuct.ac.za) |
| Date | 26 Jul 2019 |
| Team Members | DPLJER001  Jeremy Du Plessis  [DPLJER001@myuct.ac.za](mailto:DPLJER001@myuct.ac.za) |
|  | STNGAB004  Gabriel Stein  [STNGAB004@myuct.ac.za](mailto:STNGAB004@myuct.ac.za) |
|  | GRMSTJ001  St John Grimbly  [GRMSTJ001@myuct.ac.za](mailto:GRMSTJ001@myuct.ac.za) |
| Overall purpose and stakeholders [5] | Apart from the team, the main stakeholder is the client.  The purpose of this software is to implement the theoretical rules of how to transform an AR model into and EER (conceptual) model and how to transform an EER (conceptual) model into an AR model.  The KnowID paper, provided by the client, has provided and developed the transformation logic of moving between EER and AR models. It is the requirement of the team to implement the rules in code and develop an easy interface for the user to interact with. The software should provide valuable and insightful feedback to the end-user to ensure that the software is user friendly – especially for non-expert users of the underlying querying system.  The intention of the software must be to show the user that there is an easier way to query a database other than writing long, complex SQL queries, therefore this software must be easy to use and require almost no exposure to SQL or other complex querying technologies. |
| SMART Goals [5] | **Project Features**   * Graphically construct database queries through a ‘point and click mechanism’ in a GUI, transforming ER (conceptual model representation) to ARM and ARM to ER representations of data stored in a database * View report of success / failure of the transformation with details about the errors encountered during the attempted transformation   **Goals (SMART)**  The following goals are listed according to the S.M.A.R.T. technique - Specific, Measurable, Agreed Upon, Realistic and Trackable.  **Goal 1**: Implement transformations from EER to ARM and from ARM to EER as described in the KnowID paper provided by the client.   * **Specific**: The KnowID paper provided by the client details ‘rules’ for performing computational transformations from EER to ARM and from ARM to EER models. These should be correctly implemented in the program. * **Measurable**: We can be assured our goal has been reached once our test cases for the transformations are correctly achieved. * **Achievable**: The team members have acknowledged the goal as reasonable and achievable. There is the danger of unforeseen issues as this has never been done before. The client is aware of this issue. The client has provided resources to ensure that the task is reasonable and achievable. * **Relevant**: Implementing the correct transformations for this project is critical to the functionality of the ViKER interface. Incorrect performance of transformations would result in incorrect querying, broken interfaces etc. * **Time-bound**: To have the ViKER interface functional in time for unit testing, we need to implement the rules correctly at the prototype stage.   **Goal 2**: Report on success/failure of a transformation   * **Specific**: The ViKER interface must provide non-expert users with a user-friendly interface for querying and working with the underlying database. Quality, accurate and specific feedback of the success or failure of transformations is very important to the client. * **Measurable**: We can be assured that our goal has been accomplished if users are happy with the feedback of the interface and the client is satisfied by the type of feedback provided by the GUI. * **Achievable**: The team members have acknowledged the goal as reasonable and achievable. This goal will be a specific task assigned to the Chief Architect of the project who will continuously check on the user friendly nature of the program. * **Relevant**: Proving good feedback is very important to the client. It is not a critical goal to the actual functioning of the transformations, but it is a very important aspect of the usability of the program. Since the goal of this project is to allow non-expert users the ability to work with a database without expert knowledge of the underlying architecture of the database, the team has decided to make this a critical goal of the project. * **Time-bound**: To get good feedback for this goal, the team and the client have the expectation that high quality feedback will be part of the project at the prototype stage.   **Goal 3**: Report on those things that could not be transformed (error reporting)   * **Specific**: Even though specific “rules” are outlined in the KnowID paper for transforming between a conceptual model and an ARM, it is inevitable that there will be test cases where not everything selected by the client may be transformed into its equivalent representation in the ARM. When such cases arise, the information about which things could not be transformed by the program should be reported to the user. * **Measurable**: We will know we have accomplished our goal if all errors thrown by the program when processing the test cases are caught and declared in the GUI. Although there may be edge cases we do not think of in testing, we will do our best to cover as many scenarios as possible. * **Achievable**: The goal of reporting exceptional cases in which certain aspects of a transformation cannot be completed has been deemed an achievable task by the team. Naturally the program is based on rules, and if the rules cannot be executed the program will throw an error, which can be translated and relayed back to the client which makes this goal realistic and attainable. * **Relevant**: Communicating to the user what has occurred in the process of generating the ARM (which will ultimately become a query which will be used to retrieve needed information) is essential to the program, especially if elements of the conceptual model (EER in our case) cannot be transformed. It forms part of the core of the program. * **Time-bound**: The team and the client have the expectation that the “error reporting" aspect of the program will be part of the project at the prototype stage.   **Goal 4**: Report on what happened with each element.   * **Specific**: Each element of the conceptual model (EER diagram in our case) will be acted upon by the rules, we need to communicate to the user how each element of the model has been transformed, if it has indeed been transformed. * **Measurable**: We can be assured that our goal has been achieved since we will create each test and will hence know what the correct transformation of each element will be and can ensure that the reporting has been implemented correctly. * **Achievable**: The team members have acknowledged the goal as reasonable and achievable. This goal is entailed in (it is in fact a by-product of) the correct implementation of the “rules” in the KnowID paper which forms the core of the program. * **Relevant**: In accordance with the principle of user experience and design, a good product / program should provide feedback to the user. Thus this feature is relevant to the implementation of the program. * **Time-bound**: The team and the client have the expectation that the “reporting" aspect of the program will be part of the project at the prototype stage.   **Goal 5**: Build a “point-and-click" GUI to allow users to construct the query visually.   * **Specific**: the ViKER interface must provide non-expert users with a user friendly interface for querying and working with the underlying database. An easy to use interface is essential to the client. * **Measurable**: We can be assured that our goal has been accomplished if users find the interface easy to use and the client is satisfied that non-expert users will be able to interact with the interface. * **Achievable**: The team members have acknowledged the goal as reasonable and achievable. This goal will be a specific task assigned to the Chief Architect of the project who will continuously check on the user friendly nature of the program, and implement the “point-and-click" functionality. * **Relevant**: Providing an easy to use interface is very important to the client. It is not a critical goal to the actual functioning of the of the program, but it is a very important aspect of improving the ease of use of the program. Since the goal of this project is to allow non-expert users the ability to work with the software, the team has decided to make this a critical goal of the project. * **Time-bound**: The team and the client have the expectation that the “point-and-click" interface will be part of the project at the prototype stage.   **Goal 6**: Implement save/load functionality for models.   * **Specific**: The ViKER program must provide users with the option of saving the models to come back to at a later stage. The ability to store models is very important to the client as it aligns with saving the user as much time as possible when querying data. * **Measurable**: We can be assured that our goal has been accomplished if users are able to save their data and the client is satisfied by the implementation. * **Achievable**: The team members have acknowledged the goal as reasonable and achievable. This goal will be a specific task assigned to the Chief Architect of the project who will implement the saving functionality. * **Relevant**: Saving users time in querying data is very important to the client. It is not a critical goal to the actual functioning of the of the software, but it is a very important aspect of the usability of the program. Since the goal of this project is to allow save users time in querying data, storing models will allow them to come back to their models at any time, without the users having to recreate them, the team has decided to make this a critical goal of the project. * **Time-bound**: The team and the client have the expectation that the ability to save models will be part of the project at the prototype stage.   **Goal 7:** Construct at least 10 test cases   * **Specific**: Test cases are critical to this project. The team has decided that all the team members as well as the client should be involved in this process, as how the project will be used and ‘tested’ will provide much of the direction for how the project will be developed. Test cases will be designed such that critical flaws and failures of the program to properly transform between models is exposed. * **Measurable**: The test cases will be hard to measure per se, however we have set the minimum at 10 test cases in total - 5 test cases for EER to ARM transform and 5 test cases for ARM to EER. Ideally the project should be bug-free. The usability of the program itself – i.e. client happiness will be a defining ‘measure’ in this scenario. * **Achievable**: This goal is deemed very achievable by the team. Further assessment will be done with client and detailed in progress reports. * **Relevant**: The client and the team believes the use of test cases is essential to the project. Since this transformation scheme has never been done before, the development of test cases is a major part of the project itself. * **Time-bound**: Test cases should be developed by stage 3 of the project. Test cases are required for development of the prototype – which is itself critical for the project. Development of these test cases is thus critical itself and time-sensitive. |
| Inputs, outputs and performance [5] | The user will construct an ER model and use that as input for the backend to construct and output the AR model. The user will also be able to construct an AR model and use that as input for the backend to construct and output the ER model.  This functionality allows the user to go both ways which is essential for testing the software and the implementation of the transformation rules.  Regarding the performance of the program; the transformations should be efficient and scalable, adding more components to the ER or AR diagrams when executing a transformation should ideally only cause the computational time to increase linearly in proportion.  However, the main purpose of the program is to making it easier for non-expert users to create complicated queries in an easier, graphical way, so this should be the focus of the development. The responsiveness of the software should be easy and efficient for non-expert users to use. Ideally, performance/time needed to query should also be as good, if not better, than using a normal SQL query. This would give the technology use for both expert and non-expert users alike. |
| Resources and Constraints [3] | Apart from the project team, we need a frontend framework that allows us to display the relational models and allow the elements within the model to be interactive (i.e. can be dragged around and connected to different elements), we will also need to find an open-source library that integrates with our chosen framework in order to add the needed functionality.  If we cannot find the needed software, we will be limited to a text-based interface that will take more time to create (above the time already wasted) and severely impact the project timeline. |
| Feasibility [3] | The implementation of a frontend framework to graphically display the relational models appears comfortably feasible as there are already existing libraries that implement this functionality, therefore saving us a lot of time.  The functionality of being able to implement all the transformation rules is not certainly fully feasible as it has never been done before. The client has accepted the fact that not all transformation rules will work and has asked us to create an error log that outputs a useful message detailing every time there is a transformation error; the program should provide as much detail as possible in this regard.  The biggest constraint we have at this stage is the lack of available testing resources for our program, as this is newly developed software and the first of its kind. This makes it difficult to test and predict issues that may arise. |