**Progress Report**

**Date of Report:** 7th August 2019

**Report Period:** Phase 3 (2nd August 2019 – 8th August 2019)

**Project Name:** ViKER

**Client:** A/Professor Maria Keet

**Team Members:**

Jeremy du Plessis (DPLJER001)

St John Grimbly (GRMSTJ001)

Gabriel Stein (STNGAB004)

1.1 Task Activities During Report Period

* Reworked and coded basic class structures for back end:
  + Attributes
  + Inheritance
  + Method declarations
  + Excluding method body (next period)
  + We have updated our domain-class model diagram to reflect the new class
* Created JSON structure for ARM and EER
  + These will be used as input and output for test cases
  + Written test cases in the JSON structure
* Created non-functional GUI prototype for front end:
  + EER Display
  + ARM Display
  + Created window area in GUI for error log

Our objective during this period was to build the class framework for the VIKER software program. Particularly important to the success of the project is the JSON representations of the EER and ARM objects in memory, especially since such an ARM representation does not exist in any of the current literature. The OOP structures that are used in the transformation process exist conceptually in between the two JSON representation and enable the transformation from one to the other.

1.2 Planned Activities For Next Report Period

* Code up body of transformation functions
* Meet with client to ensure transformations are executing correctly
* Create drag and drop functionality in front end
* Broad back-end testing
* Integration of Front and Back end

The focus of the next period (one week long) will be to successfully complete the back end source code and test the transformation functions and figuring out how to read and write JSON to and from the GUI. During the period (week) following that, we will focus on integration of front and back end.

1.3 Problems / Challenges Experienced

* One of our team members was ill during the period and was not able to do as much work as they otherwise would have, so we are a little behind on some aspects of programming
* The algorithm in the KnowID paper is quite abstract and there are elements in the algorithm that we do not know how to map to real code in python which is delaying our progress in implementation:
  + How to implement the path functional dependency (*pathfd(A1, A2, …) -> self*) aspect of the ARM is still not clear to us – hopefully this can be clarified at the demonstration
  + The ‘covering’ and ‘disjointness’ constraints are not clear to us – we assume they mean the same as they do in mathematics, but we are not sure.

The abstract ideas underlying the KnowID paper (specifically Ontology) are not ideas we have been familiar with, so we have had to figure things out as we are developing and that has slowed the process down somewhat. It will be important to keep a channel of communication open with the client to ensure our understanding, and eventually our results, are correct.