CS4ZP6 Proof of Concept demo Plan

Yuriy Toporovskyy, Yash Sapra, Jaeden Guo

January 19, 2016

Ampersand Introduction Ampersand is a software and an approach for the use of business rules to define business processes. Though business rules aim to enhance the quality and efficiency of requirement elicitation in software projects there is still a huge gap between business rules are they are presented and the way they are implemented in software systems, and that is what Ampersand hopes to bridge. The main advantage being that Ampersand produces provably correct requirements' consistency and traceability.

Ampersand begins by describing business rules in a formal language (Abstract Data Language), and turns that into a functional specification, documentation and a working software prototype.

Ampersand As it is Ampersand contains high-level architecture and is theoretically based off of mathematical concepts such as relational algebra and Tarski's axioms to name a few. The Ampersand process begins by taking the end-user writing business rules in a domain specific language (ADL), it is then parsed into a parse tree (referred to as the P-structure) which feeds into the type checker and converts it into relational algebra format (referred to as the A-structure). The semantics of ADL is expressed in terms of the A-structure then passed to the Calc component which generates the functional structure (referred to as the F-structure or F-spec). The F spec contains all necessary specification and generate the output, and the inner working of F-spec is the focus of our Capstone Project.

Current Advancements made to Ampersand Software As mentioned previously one of the artifacts that Ampersand generates is a Database with all tables. ECA rules (or Event-Condition-Action rules) are used to maintain consistency of data. It is our task to translate these ECA rules to SQL statements which can be used to resolve any violation or inconsistency within the Ampersand System. In the previous system the ”EXEC ENGINE” was a bandaid generally used for this purpose, but was very limited and missing many essential functionalities such as the ability to update.

The ECA rules are composed of a reference, a delta, a data type referred to as a PAClause. The PAClause indicates what is to be done and the delta can be thought of as a parameter for the ECArule that indicates which action needs to be taken.

The PAClauses contain various actions that can be executed, and each is based on a guard that may trigger none, a few or all possible actions. These PAClauses maintain system invariants which must be consistent, we are bridging the SQL commands to bridge the actions to the meaning.

In order to capture the semantics and syntax of SQL, unique data types and methods had to be constructed to capture relationships between entities. Each component was deconstructed to fit the types represented in Haskell, include associated SQL types with their associated domain of interpretation.

Yuriy – freestyle, I can't explain future steps

- mockdata base
- magic?
- undefined evaluations?

original The focus of the Proof Of Concept will be to demonstrate the current advancements made to the Ampersand software that allow us to translate the Event - Condition - Action (ECA) rules to SQL statements which can be used be Ampersand to resolve any violations or inconsistencies in the Ampersand Database.

We will first give a brief introduction of the Ampersand software to the team attending our Proof of concept demo. This will allow team 2 to familiarize themselves with Ampersand and what it is capable of producing (i.e. Information Systems). We will first compile a simple ADL script in Ampersand to generate a working information system. At this point, once we have a working information system generated from Ampersand we we'll inform the audience about what other artifacts can be generated from the Ampersand software.

Once everyone in the audience is on board with Ampersand and its capabilities, we will go ahead and demonstrate our project. For this purpose, we'll use the same example that was used in the previous demonstration, which is a Project Administration Information System. The aim is to demonstrate the SQL that is generated from the ECA rules that Ampersand already generates. A manual translation of ECA to SQL will demonstrate the minimal correctness of the generated SQL. Once that is clear, the we'll inform the audience about what challenges lie ahead and what's the next milestone we will be working on.