CS4ZP6 Problem Statement
Ampersand Tarski Event-Condition-Action Rules

Yuriy Toporovskyy, Yash Sapra, Jaeden Guo

September 25, 2015

[**YT:** *What is EFA? What is ECA? These should be defined before they are used; so, they shouldn't appear in the title. In fact we should only mention it in passing; overuse of acronyms can make a text undecipherable. I know what ECA is, but I don't know what EFA is and it is never defined in this document, at all.*] [**JG:** *EFA turned out to be the acronym for the project which was the title E.F.A (Event-Condition Action) rules for Ampersand on the Project Registration. It was on there you signed it and when I asked you two for a project title, neither of you said anything and it was due*] [**YS:** *I thought its was ECA, my bad. I don't think we can change what's done*]

[**Note:** *You don't need to put /indent before every paragraph. Latex inserts that automatically. You can change the amount by changing the value of /parindent. Also instead of manually adding a line break after every paragraph, just use parskip (see above).*]

The Ampersand Project translates the non-functional requirements of a business into the functional specifications required by software engineers. It also provides engineers with a variety of aids which help them to design products that fulfill all of the needs of their clients and the end-users. [**YT:** *What does "such as data models and service catalogues" mean? If I don't understand it it, the prof won't*] [**JG:** *from what I can tell ADL generates data models and service catalogues according to functional specifications, and ADL is part of Ampersand; I got that from Joosten's article – we could just take it out.*] Business requirements are not universal and often vary from one country or culture to the next. Additional difficulty arises from the fact that business requirements are not easily translatable into design requirements.

The Ampersand Project has proven reliable at translating natural language into technical specifications so that they could be incorporated into the overall design. Business transactions input by the user are translated into "process rules"; and re-

quirements are translated into "Event-Condition Action" (ECA) rules. Both of these rules are combined into a single SQL program. Currently, a programmer using Ampersand must manually ensure that each requirement is satisfied after each transaction, by telling Ampersand how each violation must be fixed. Not only is this process error prone, it is unmaintainable; when new requirements are added, the already-existing system may violate the new restrictions. In many scenarios, the requirements may be safety and security critical. To this end, Ampersand must also generate relational algebra proofs to show that the SQL generated from ECA rules is sound.

This project will focus primarily on modifying Ampersand to include the translation from requirements to ECA rules to SQL commands (whereas the pipeline for process rules is already in place). This will ameliorate Ampersand in two ways; by automating part of the programmers [Sp. Should be "programmer's" or "programmers' " —DS] job, namely writing code to preserve requirements throughout a transaction; and by generating SQL code which is proven to not violate any system requirements.

[I understand what you'll be doing for this project, but what is the problem you will be solving? —DS]

[Who exactly are the stakeholders? Programmers? Software Engineers? Why should they care about Ampersand (or your modifications to Ampersand)? —DS]

[Your problem statement is not abstract enough. It is very focused on the implementation (which, while important in this case, still clouds the problem being solved) —DS]