Rationale for Tree structure

We choose our tree implementation based on a set of criteria that would allow for simple input analysis. At the same time the tree structure would allow for effective output generation. We took these conditions into account as well as the need for name analysis as we selected criteria that were important to our project. They are listed in order of importance below, 1 being most important. The points awarded are out of a total possible of 10.

1. Easy to determine scope  Score: 8
   The scope should be very intuitively organized. The tree uses block like structure to determine scope. In this way, everything that is need as far as scoping is concerned can be found easily in the tree. Sub-trees with a 'worksheet' element as root represent nested blocks and within these blocks, range scope applies.

2. Easy to do computations  Score: 7
   All the computations should be separated under different rules. This would make the implementation of every computation easier. Also all elements must be easily retrievable: the rules should refer directly, when possible, to the entities that will take part in a computation.

3. Minimal number of rules  Score: 6
   Only the needed rules should be present but enough to keep the meaning of the tree structure clear. Disambiguation should be made at another place.

4. Rule for every necessary computation  Score: 9
   Each major function that can be represented in the language should be specified by its own rule. This will make it easier to understand what the tree actually does and also to change components of tree independently of each other.

5. Readable  Score: 7
   The tree should be well organized. It should be easy to see how various elements relate to each other.

Our tree scored a 37 out of 50.