1450.4 meeting minutes – 04/20/11

Attendees: Ernie Wahl, Paul Reuter, Ajay Khoche
Not present: Oleg Erlich, Jim O'Reilly, Markus Seuring

Agenda:

- IEEE Meeting Preamble (No discussion of proprietary information).
- Discuss changing day of week for WG meeting (teleconference)
- Continue work on section 7.6 "Variables and Expressions".

Summary:

- Reached consensus on paragraph near line 1671: A variable defined in a TestType Variables block shall be accessible by name to that test-type only, i.e., a derived test-type shall not access base-type variables. Each instance of a particular test type shall have its own copy of parameters and variables. Parameters however, shall be accessible from the outside via, e.g., test.param, where test is the name of an instance of a particular test-type and param is the name of a parameter of that test-type. A test parameter or variable identifier shall hide a global variable of the same name (see keyword Global).
- Concluded that phase II will not be painted into a corner and defer actual solution to phase II with regard to:
    - Planned output, e.g., cout << "spec": if spec is defined, will the output operator choose to write out type Spec or type String ? By what mechanism ?
    - Suggestions: Use escape character ? Define operator String("spec") ?
- Recommended acceptance of Footnote 2, pg 24:
- A TPG may use Permissions to mark variables on whose presence and/or value a test program depends so that a GUI for the TPG or other tool that reads TPG STIL.4 output, may warn a user requesting deletion of a required variable or a change of a required value. This differs from Const which affects variable behavior only at runtime.
- Recommended change to Footnote 3, pg 25 to: One global variable block may optionally be unnamed. The unnamed global variable block and the global named variable blocks specified in the TestProgram block shall be visible to STIL.4.
- Variable blocks inside test-type definitions shall be unnamed.
- Discussed philosophical parameters and STIL.4 goals to help focus our efforts:
    - working on phase I, requirements for automatic test program generation (phase II is for language executable on a target tester)
    - STIL.4 only describes high level test interface, i.e., no low level, e.g., hardware register loading code
    - minimalist approach, maximum simplicity: only include types and features for which there is known utilty, i.e., it's easier to add features than to take them away

Actions:

Reference documents (If logged into your google account, can edit. If not, can only view.)

- http://spreadsheets.google.com/ccc?key=0AoKiPr1I9LY9dF95dkhSTVVqOU5GbWJyWFNhY0JPX0E&hl=en
- Namespace resolution examples document: http://docs.google.com/Doc?docid=0AYKiPr1I9LY9ZGY4dmNjNTNfMGZkOGJ2bmZy&hl=en
- Scratchpad spreadsheet: https://spreadsheets0.google.com/ccc?key=tQ93VDnAZ-Cl9RFKpPrPDzw&authkey=COzyro8K&hl=en&authkey=COzyro8K#gid=0
- Scratchpad "Word" doc: https://docs1.google.com/document/d/1zVu2M8nTJsrm0nFbBhiuM8-YRt4ErYqdy_uSa3x3_T4/edit?authkey=CLrgwrsG#

Next meeting: 04/27/11

For reference STIL .4 information can be found at the IEEE STIL website: http://grouper.ieee.org/groups/1450/ (select the P1450.4 link from the table) or use the direct link http://grouper.ieee.org/groups/1450/dot4/index.html