P1450.4 meeting minutes - 02/18/04

Attendees: Dave Dowding, Tony Taylor, Jim O'Reilly, Ernie Wahl, Jose Santiago, Jim Mosley
Not present: Don Organ, Tom Micek, Yuhai Ma, Doug Sprague, Eric Nguyen

Agenda:
- STC/STIL collaboration update
- Review latest flow diagram documents
- Review dates for face-to-face meeting
- Progress report on STC/STIL .4 collaboration

Progress on STC/STIL.4 collaboration:
Update from Jose Santiago: Discussions within STC are ongoing - some issues being discussed center around which documents will be released to the STIL .4 WG. STC doesn't necessarily want to release HW documents. Jose has delivered the message to STC that initially, the STIL .4 WG would be interested only in the OTPL language documents. The next STC meeting is in the Bay Area March 25/26, 2004. We plan to communicate to STC that we'd like to have the language syntax documents available for our WG face-to-face meeting to be held in March (see below).

Face-to-face meeting:
Tentative date for a face-to-face is March 16-17, 2004. Location to be determined (will be in Bay Area). If there's a compelling reason to change it, let Dave know within the next week.

Review latest flow diagram documents:
Further review of Dave's latest flow document (TestProgramFlowWorkingDiagramsB.pdf, dated 2/11/2004) - emailed to WG members on that date (not yet posted on the web site).

- Some discussion about the TestProgram block, and its (currently-shown) encapsulation in an Environment block. Jim O. pointed out that, per the P1450.1 standard, an Environment block is not intended to be referenced by other STIL blocks, but only by external tools (i.e., STIL readers). Of course, the Environment block(s) CAN (and do) reference other STIL blocks. Since, in the original concept of STIL.4 (from 1999), the TestProgram block served a similar top-level function (multiple test program blocks can exist in a STIL file, and external tools - in this case, either STIL readers or tester operating system executives - would select which of the TestProgram blocks would be used). Thus, the function of an Environment block is conceptually quite similar to a TestProgram block in terms encapsulating (or referencing) other STIL data. As such, there is some question as to whether the Environment block may be needed (the current diagrams show the Enviroment block enclosing TestProgram block(s)) - or if the TestProgram block can stand alone at the top level. After a bit of discussion about this, we agreed that it could probably go either way - but before we need to make that decision, we need to finish fleshing out some of the lower-level concepts (see third bullet point below).

- Some discussion about the terms "TestModule" and "FlowModule" (p3) and Figs. 2-4. Jose said he wasn't completely clear about the differences between a TestModule (Fig. 2) and a FlowModule (Fig. 4). The main difference between the two is in the type of TestMethod that is called - a TestModule calls a simple TestMethod (i.e., a single function/procedure call or execution of a TestMethod object), while a FlowModule calls a TestMethod which happens to be of type "Flow" or "SubFlow". Both TestModule and FlowModule are derived from (?) a "Harness" (Ernie: I don't quite remember exactly how you described what a "Harness" is - can you clarify?).

  Since the two are very similar, Jose questioned why we need two different names - if they both had the same name, the concepts conveyed by the diagrams would be clearer. Further, he suggested that since this WG is dealing with flow issues (as opposed to TestMethods, which is the for-now-suspended .5 effort), perhaps we should use FlowModule. On the other hand, since the module is calling a type of test (be it a simple test method call, or a subflow call), TestMethod may be more logical. I don't think there was strong preference for one term over the other, only that we use the same term for both.

Again, this is not something that requires an immediate decision, other than that we start using the same term for both, and change the captions of Fig. 2 and Fig. 4 (i.e., the caption of Fig. 2 would now indicate that this TestModule is an "Example of a TestModule containing a TestMethod call", while the caption of Fig. 4 would now read " Example of a TestModule containing a SubFlow call" - or something to that effect.

- Among the lower-level elements and concepts we need to flesh out are:
  - Binning, variables (scalar or multi-dimensional)

  - Distributing functions of PatternExec among the new constructs we're defining. Currently, PatternExec specifies the following:

    PatternExec (PAT_EXEC_NAME) **{**
    (**Category** CATEGORY_NAME **;**)*
    (**Selector** SELECTOR_NAME **;**)*
    (**Timing** TIMING_NAME **;**)
    (**DCLevels** (DC_LEVELS_NAME);)
    (**DCSets** (DC_SETS_NAME);)
    (**PatternBurst** PAT_BURST_NAME **;**)
    }

    The specification of each of these will probably be redistributed to either the flow-node level, the TestModule level, or the TestMethod level (or some combination of the above). It might also be possible to allow (for the very simple programs) specification of the above constructs at the test program level.

  - Use of and updating of symbolic variables (spec/category/selector variables).

  - How to create new (user-defined) types from the basic elements provided by STIL.

  - Definition and usage of test program variables, such as
    - Global variables (global in scope to the TestProgram)
    - Other external variables (i.e. TSEM variables)

- Tony has agreed (at an appropriate time) to take on the task of formatting our documents into IEEE "standardese". We assured him that we will remember that offer!

And that's about all I can remember (without listening to the call, which I don't have time to do - I have ITC paper reviews to set up, and I can hear Tony's whip cracking to complete my ballot on P1450.1 <grin>)

Jim