P1450.4 syntax subgroup meeting minutes - 09/19/05

Attendees: Jim O'Reilly, Tony Taylor, Greg Maston, Dave Dowding

Not present: Doug Sprague

Agenda:

- Process discussion (some confusion among the syntax subgroup regarding
  - Who owns/maintains the document?
    - Tony will continue to maintain the official draft document. The working document is a FrameMaker document; revisions will (continue to) be published to the IEEE STIL website as PDF files. The main reason for this is that we want to maintain the document in one editing program. Since IEEE uses FrameMaker, it's much better to do all our editing in that program, and since I (Jim) don't have access to FrameMaker, it makes most sense for Tony to have control of editing the official draft document.
  - How do we get proposed changes in front of syntax subgroup for discussion without making it look as if someone else is taking ownership of the document?
    - Any proposed changes go only to syntax subgroup and document should indicate that it's proposed only and not accepted.
    - The proposed changes should include the changed clauses only.
    - Both the file name containing a set of proposed changes, and the document headers or footers should indicated that this is a proposal only – i.e. use a header of the form "Proposal for changes to IEEE P1450.4/D15– <date of changes)".
    - In addition, if possible, include the watermark "Proposed Changes Only" in any documents.

Summary (areas of discussion):

- Discussion of Jim's proposal to merge the TestObjectsDefs and TestMethodDefs blocks, and eliminate the "pure interface specification" functionality of the current TestMethodDefs block.
  - Comments: Not a good idea. Provides no clear delineation of external vs. internal test methods. Names get misspelled all the time – don't want a misspelled name to imply that the entity (test flow or user-defined method) is actually a method from an external library.
- Much discussion back and forth about the above issue.
  - The consensus was that we DO want to keep a block type which provides a linkage to an external library, in addition to a block type which specifies a template for a particular type of test - the template including:
    - pre- and post-actions and exit port statement
    - the specification of parameters to be passed to and/or from a specific instance based on the template type
    - Possibly including STIL setup specification statements such as Timing, Levels, etc.
  - The current proposal is that:
    - The block which specifies linkage to an external library of executable test routines will be called TestMethodDefs.
    - The block which specifies the test template will be called TestNode (no "Defs" at the end of the keyword). This was previously called TestObjectDefs. Note that this is distinct from FlowNode, which can only appear within the context of a TestFlow. Even though there's no "Defs" at the end of the keyword, this is a template only, not an instantiation of the template. Only the TestInstances or TestExec keyword, or an On* entry point can actually instantiate such a template.
    - The "Defs" portion of the TestFlowDefs will also be dropped. Nonetheless, the TestFlow block defines a template only, not an actual instance of that flow. The

instance of a flow based on a TestFlow template will not occur until it's instantiated by a TestInstances orTestExec statement, or by an On* (no, not OnStar!) entry point.

- The "Inherit" statement will be moved from its current location in TestMethodDefs to the (newly-renamed) TestNode block. It will be highlighted in red and struck through, to indicate that it has not yet been accepted. While there are probably valid reasons for allowing such a statement, its behavior has not yet been defined to any level of detail. Further it's acknowledged that the use of the Inherit statement in other STIL block types (such as WaveformTables) has been less than expected, and can lead to difficulties in understanding. We have made no decision about whether to drop or keep it – more discussion to follow.
- TestInstances:
    - It was felt that this was a useful feature to keep – so that users could create specific instances of TestNodes or TestFlows which persist both before and after any specific call to a TestNode or TestFlow. The issue of WHAT attributes of a TestNode or TestFlow need to persist beyond the point of call has not yet been resolved. Certainly, for a TestNode, the result of the TestMethod needs to be saved – but does this need to stored as part of the state of a persistent object, or can we simply require that it be saved in a user-provided Out variable? At this point, we'll be persistent, and keep the notion of persistent objects, with some additional work to be done on defining WHAT attributes of a TestNode or TestFlow need to persist.
- Inclusion of an OnUnload entry (exit?) point. Not much discussion here – it was felt that it could be useful, but the precise behavior (when is it called? How are the lifetimes of persistent objects affected by the OnUnload execution? We need to discuss this more. Nonetheless, I believe that Tony agreed to add the OnUnload to the list of EntryPoints.
- Finally, we talked about the issue of including Timing, DCLevels, DCSets, and PatternBurst keywords in addition to Category and Selector wherever we had currently specified Category and Selector. Currently, that includes the TestProgram level, the TestFlow level, and the TestNode (nee TestObjectDefs level). We had very little time to discuss this, so I just put the idea on the table to get people thinking about it.

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