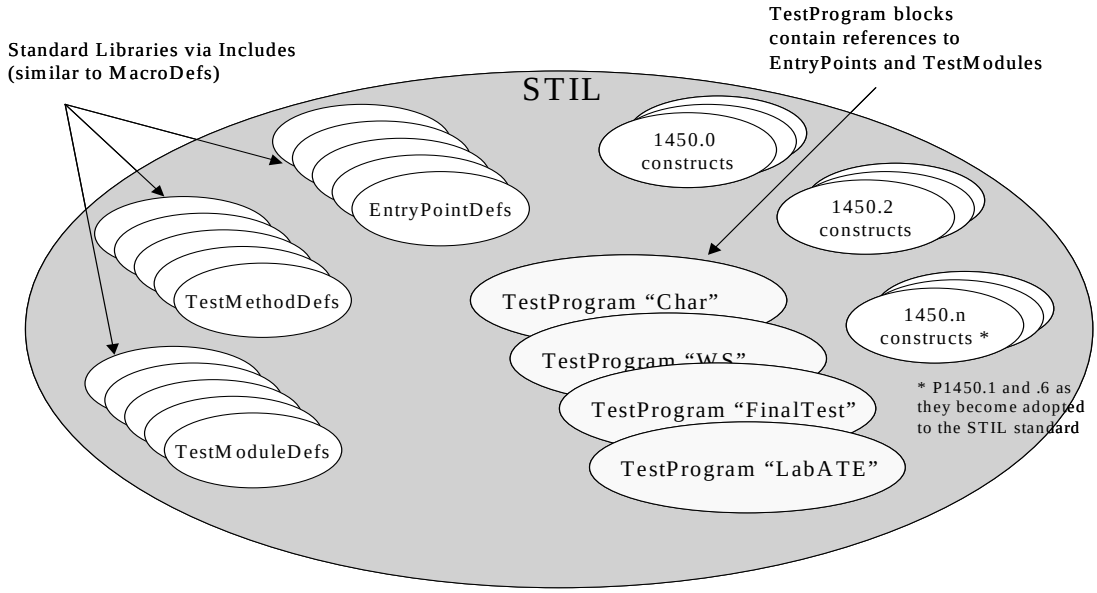
1.0 Top-Down Conceptual View

The P1450.4 constructs will reside within a IEEE 1450-1999 STIL Std file. Its top level constructs will add to the STIL top level constructs/keywords. Figure1 shows the TestProgram block, TestModuleDefs, TestMethodDef and EntryPointDefs blocks. The “Defs” blocks contain all definitions of their respective definition blocks and parallel the MacroDef block in the 1450.0 STIL standard construct.

FIGURE 1. Top Level Test Program Flow Constructs Diagram



2.0 Test Program Flow Extension Terms

2.1 TestProgram Block:

The top level test program construct. There can be one or more. One may be global (unnamed). There may be one or more named TestProgram blocks in a STIL file.

2.2 Flow Block (or TestFlow):

This is the top level program flow construct. There can be one unnamed Flow block. There can be one or more named Flow blocks. This block contains definition of flow and bin entities that make up a given test program flow.

2.3 EntryPoint:

This is a reference to a special program level task activated by the tester (tester operating system, system interrupts, etc.) This entry point references a TestModule. There is a general set of EntryPoint entities defined by this extension (i.e. OnStart, OnReset, etc.). These can be named and one instance of each can be unnamed and treated as global to any Flow

block that does not declare a named one of each type. When a flow is active and a tester event requires an EntryPoint response, the associated EntryPoint TestModule/FlowModule that was declared, or the default if not declared is run.

2.4 FlowNode:

(See Figure2) A node in the program flow that contains a ModuleRef (Body) that references a TestModule or FlowModule. This node has PreActions that defines the entry point into the node and may contain actions, declarations such as Spec/Category selection, etc. Absence of actions may in the Pre section may cause default actions (tbd). The Post section contains PostActions, Arbitrator, and ExitActions. The ExitActions give directives as to the follow-on flow path taken out of the FlowNode.

2.5 TestModule:

(See Figures 3, 4 and 5)

2.6 BinNode and BinMap:

Not yet defined/discussed

(Need two natures: terminal and flow-through)

2.7 TaskNode and DecisionNode:

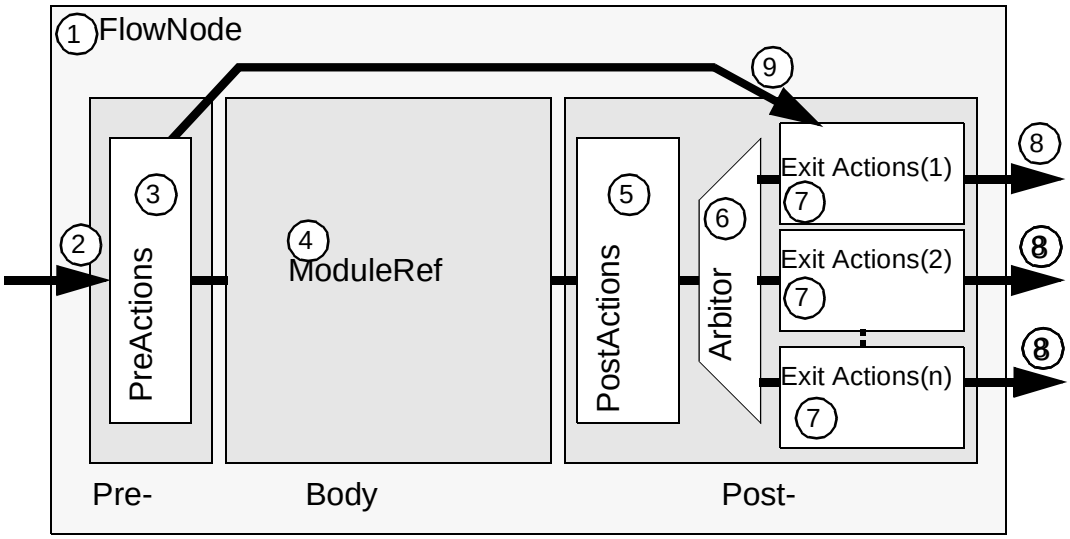
These are non-test type nodes used for non-test activities and flow decision content.

2.8 “Harness”

This is an informative term that is not intended to have a keyword in the extension language. It is a common denominator descriptor for all TestModule instantiations. Its use refers to a data type (or object type) from which the various types of TestModules (shown in figures 3 and 5) are derived.

3.0 FlowNode Conceptual Model

FIGURE 2. The FlowNode Conceptual Model Diagram (with its named components)



3.1 FlowNode Components Descriptions

1. FlowNode

2. EntryPath

3. PreActions Block

4. ModuleRef (Module Reference)

5. PostActions Block

6. Arbitor Block

7. ExitActions Block

8. ExitPath

9. SkipPath (can goto to any ExitAction Block)

3.2 FlowNode Informative Term Descriptions

1. “Pre-” portion

2. “Body” portion

3. “Post-” portion

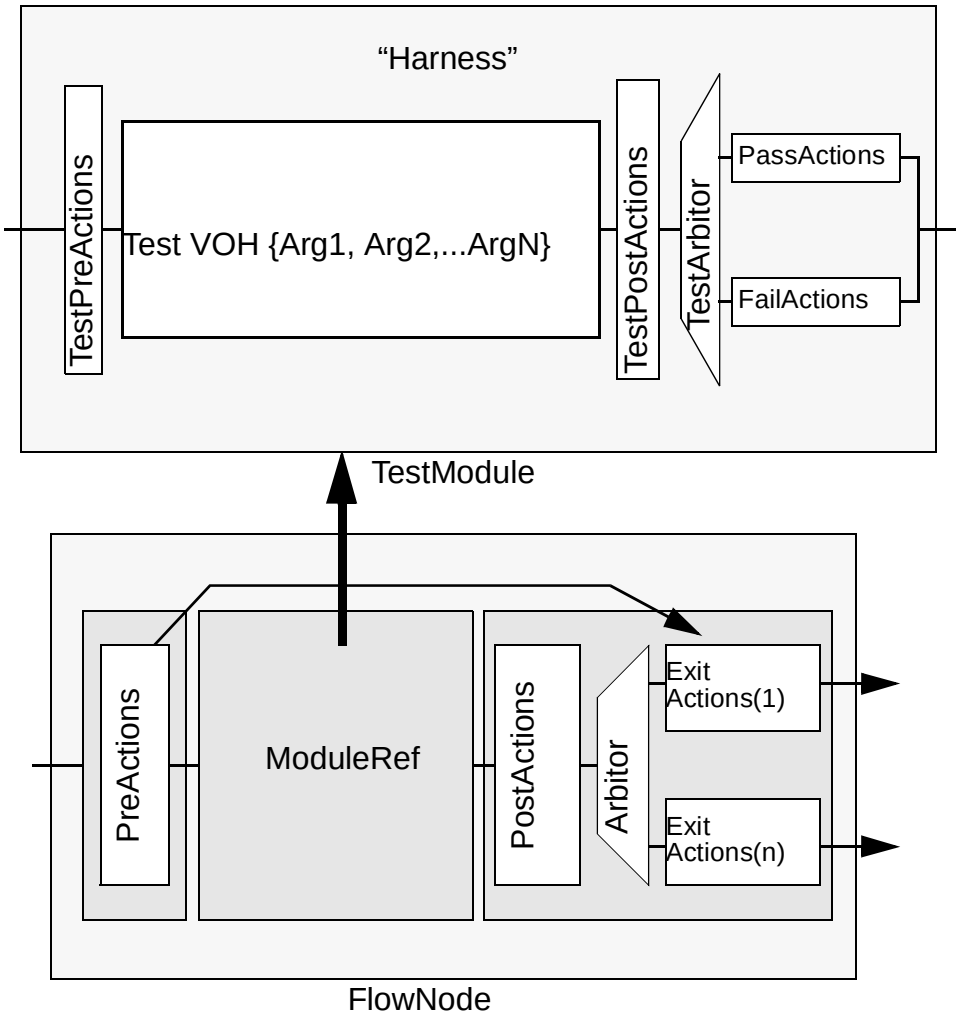
4.0 Relationship of FlowNodes to TestModules

The instantiation of the TestMethod (i.e. VOH) can be “in-line” or “defined-before-use”. “Define-before-use” is the mechanism by which two or more FlowNodes can refer to the same “Test Object” (i.e. Module). Test Module instantiation involves placing the instantiated TestMethod inside the harness as shown to the left (the harness is the grey portion of the box labeled TestModule.)

TestMethod “VOH” {Arg1, Arg2,...ArgN} is the type definition, and

Test VOH {Arg1, Arg2,...ArgN}is the instantiation of the VOH TestMethod in the test module.

FIGURE 3. Block Diagram View of FlowNode Where ModuleRef References a TestModule



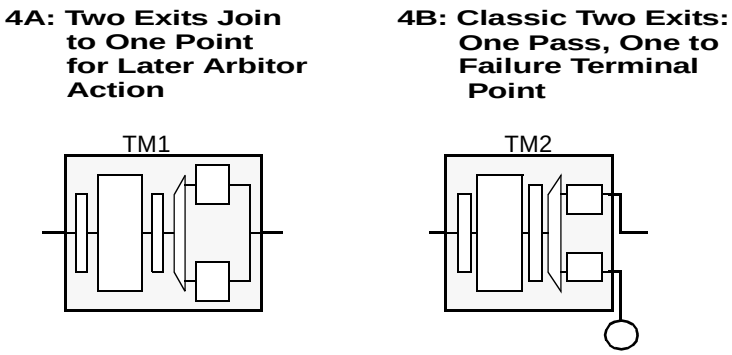
5.0 TestModule Characteristics

5.1 The "Harness" of the TestModule (a better title will emerge)

Commonalities. What the harness provideds. The FlowNode has dependencies on the the harness. Describing the data interaction model. Perhaps an upper level view of the interface between the FlowNode and the TestModule. Examples. Cummunication with input and output arguments flowing into and out of the TestModules.

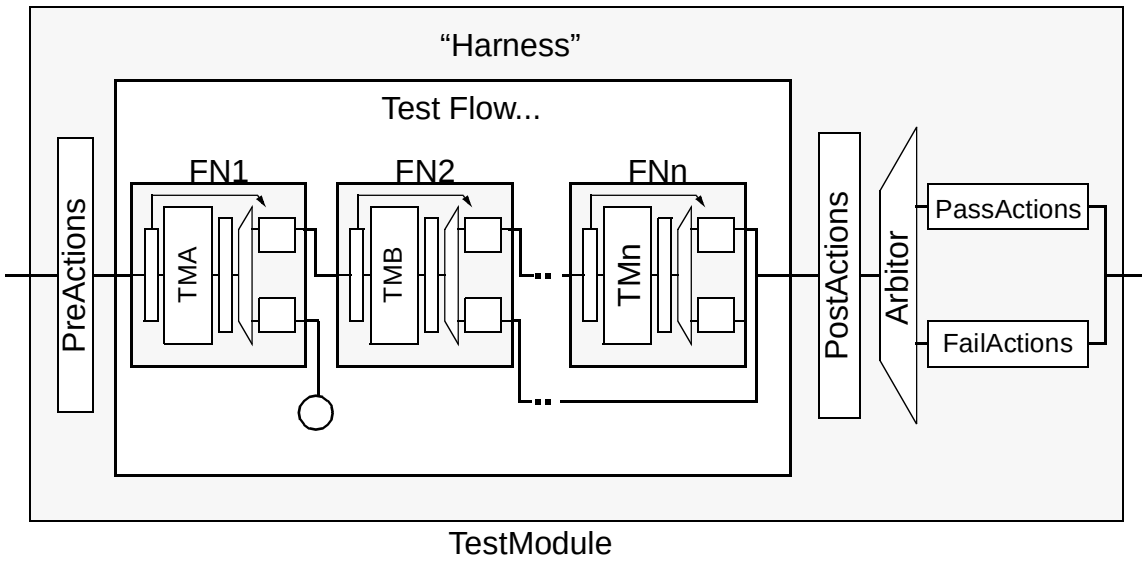
5.2 Two Types of "Outflow" Configurations for TestModules

FIGURE 4. Conceptual Block Diagrams of the Two Outflow Types



5.3 TestFlow as the TestModule Body

FIGURE 5. TestModule Referencing a TestFlow



6.0 Upper Level Example of Program Flow Constructs

This section of the document is only to give the conceptual models described above an idea how they would be seen in the STIL file. This section is not providing construct syntax. This working document is providing the conceptual view of the P1450.4 language only.

FIGURE 6. STIL File Example - Initial STIL.0 Constructs

```
STIL  1.0;
Header {
    Title "ProgramFlow Mockup to Show Construct Organization";
    Date "Wed Mar 04 14:30:00 2004";
    Source "ATE_TestProgGenerator_v0.0.0";
    History {
      Ann {* Hand edited by Dave Dowding                          *}
      Ann {* ---------------------------------------------------- *}
      Ann {* File name:     chipProgram.stil                       *}
      Ann {* ---------------------------------------------------- *}
   }
}
Signals {
...
}
SignalGroups  {
...
}

SignalGroups "WaferSort" {
...
}

SignalGroups "FinalTestPKG_A" {
...
}
...
...
...

Timing  {
    WaveformTable "WFT1" {
        Period '5.234ns';
        Waveforms {
...
        }
    }
...
...
}

```

FIGURE 7. TestModuleDefs and EntryPointDefs Segments

```
42 TestModuleDefs "StandDevice" {
43     "StdContinuity" {
44 ...
45     }
46     "EZFunctionals" {
47 ...
48     }
49     "HighFreqVDD_minFuncs" {
50 ...
51     }
52 ...
53 ...
54 }                                 // end of TestModuleDefs block
55
56 EntryPointoDefs "StandardEPs" {
57     OnLoad "StdLoadWS" {
58 ...
59     }
60     OnStart "StdStartWS" {
61 ...
62     }
63     OnReset "ResetPwrOff" {
64 ...
65     }
66 ...
67 ...
68 }                                 // end of EntryPointsDefs
69
70 EntryPointoDefs {
71     OnPowerDown {
72 ...
73     }
74     OnInitFlow {
75 ...
76     }
77 }                                 // end of EntryPointsDefs
78
79
```

FIGURE 8. TestProgram Blocks Segment

```
 80 TestProgram "WS" {              // Wafer Sort TestProgram "WS" Start
 81
 82     EntryPoints {
 83         OnStart StdStartWS; )
 84         OnReset ResetPwrOff; )
 85         OnLoad  StdLoadWS; )
 86 // Note: No OnPowerDown EntryPoint referenced, use global
 87 // Note: No OnInitFlow EntryPoint referenced, use global
 88 ...
 89 ...
 90     }           // end of EntryPoint declaration for TestProgram WS
 91
 92     DomainReferences {
 93     SignalGroups WaferSort;
 94 ...
 95 ...
 96         }       // end of DomainReferences for TestProgram WS
 97
 98     Flow "WaferSort" {
 99 // WS Program flow hierarchy body (references items in TestModuleDefs...)
100 ...
101 ...
102     }                          // end of Flow WaferSort
103 }                              // End of TestProgram WS
104
105
106     TestProgram "FT"  {        // Final Test TestProgram "FT" Start
107         EntryPoints {
108                 "StandardEPs"
109         }       // end of EntryPoints declarations for TestProgram FT
110
111         DomainReferences {
112         SignalGroups FinalTestPKG_A;
113 ...
114 ...
115     }           // end of DomainReferences for TestProgram WS
116
117     Flow "FinalTest" {
118 ...
119 ...
120     }                          // end of Flow FinalTest
121 }                              // end of TestProgram FT
122
123 ...
124 ...
125 //                             Rest of the STIL file...
```