VPI calls for Temporal Logic Operators

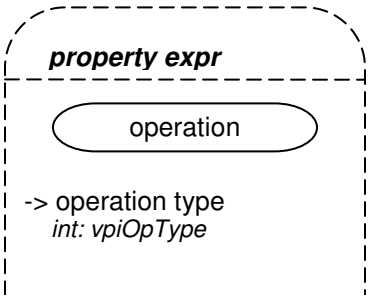
Aligned with p1800-2008-draft 4

Objectives:

Elaborate VPI for the new operators introduced in 1932. This proposal also contains a bug fix in the expression VPI (see 36.50).

36.45 Property specification

REPLACE (note to editor: only affected parts are shown)

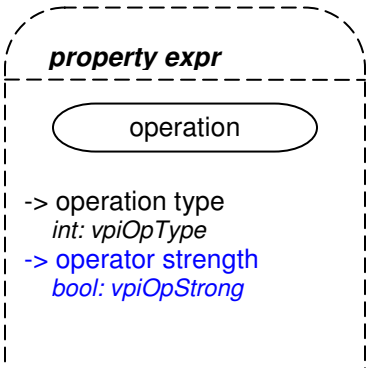


Details:

1) Variables are declarations of property variables. The value of these variables cannot be accessed.

2) Within the context of a property expr, **vpiOpType** can be any one of **vpiNotOp**, **vpiOverlapImplyOp**, **vpiNonOverlapImplyOp**, **vpiCompAndOp**, **vpiCompOrOp**, **vpiIfOp** or **vpiIfElseOp**. Operands to these operations shall be provided in the same order as shown in the BNF.

WITH



Details:

1) Variables are declarations of property variables. The value of these variables cannot be accessed.

2) Within the context of a property expr, **vpiOpType** can be any one of **vpiNotOp**, **vpiOverlapImplyOp**, **vpiNonOverlapImplyOp**, **vpiCompAndOp**, **vpiCompOrOp**, ~~or~~ **vpiImpliesOp**, **vpiIffOp**, **vpiIfOp**, **vpiIfElseOp**, **vpiOverlapFollowedByOp**, **vpiNonOverlapFollowedByOp**, **vpiNexttimeOp**, **vpiAlwaysOp**, **vpiEventuallyOp**, **vpiUntilOp**, and **vpiUntilWithOp**. Operands to these operations shall be provided in the same order as shown in the BNF~~.~~ with the following exceptions:

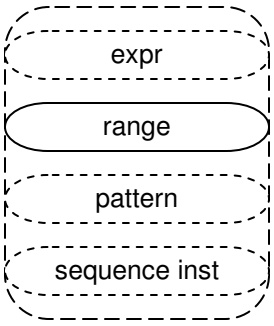
   — **vpiNexttimeOp**: Arguments shall be: property, constant. constant shall only be given if different from 1.

   — **vpiAlwaysOp** and **vpiEventuallyOp**: Arguments shall be: property, left range, right range.

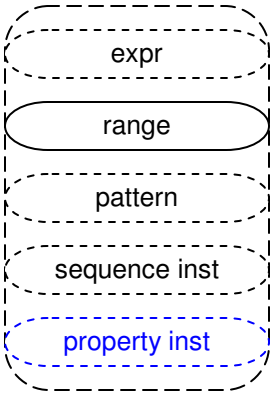
3) **vpiOpStrong** is valid only for operations **vpiNexttimeOp**, **vpiAlwaysOp**, **vpiEventuallyOp**, **vpiUntilOp**, **vpiUntilWithOp** and for sequence expression **vpiOpStrong** shall return TRUE to indicate the strong version of the corresponding operator.

36.50 Expressions

REPLACE (note to editor: only affected parts are shown)



WITH



M.2 Source code

REPLACE

```
#define vpiImplyOp 50 /* implication operator */
#define vpiNonOverlapImplyOp 51 /* |=> nonoverlapped implication */
#define vpiOverlapImplyOp 52 /* |-> overlapped implication operator */
```

WITH

```
#define vpiImplyOp 50 /* implication operator */
#define vpiNonOverlapImplyOp 51 /* |=> nonoverlapped implication */
#define vpiOverlapImplyOp 52 /* |-> overlapped implication  n operator */
#define vpiOverlapFollowedByOp editor to fill /* overlapped followed_by
operator
*/
#define vpiNonOverlapFollowedByOp editor to fill /* nonoverlapped
followed_by operator */
#define vpiNexttimeOp editor to fill /* nexttime operator */
#define vpiAlwaysOp editor to fill /* always operator */
#define vpiEventuallyOp editor to fill /* eventually operator */
#define vpiUntilOp editor to fill /* until operator */
#define vpiUntilWithOp editor to fill /* until_with operator */
```

REPLACE

```
#define vpiCompAndOp 79 /* Composite and operator */
#define vpiCompOrOp 80 /* Composite or operator */
```

WITH

```
#define vpiCompAndOp 79 /* Composite and operator */
#define vpiCompOrOp 80 /* Composite or operator */
#define vpiImpliesOp editor to fill /* implies operator */
```

REPLACE

```
#define vpiLocalVarDecls 609
```

WITH

```
#define vpiLocalVarDecls 609
#define vpiOpStrong editor to fill /* strength of temporal operator */
```