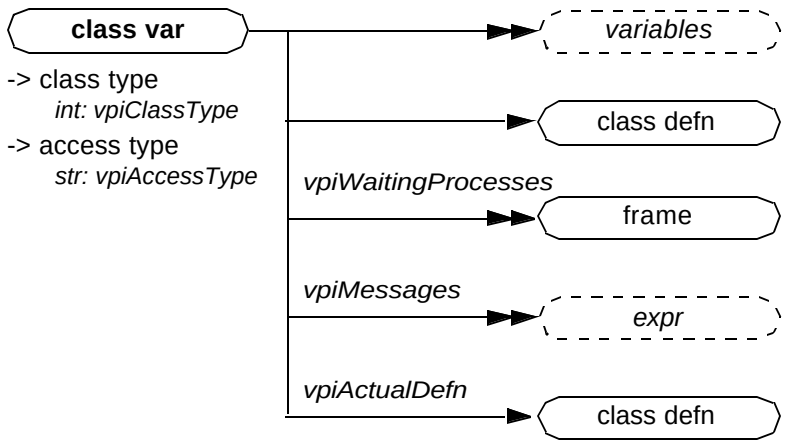
REPLACE

32.22 Class variables



NOTES:

1) The **vpiWaitingProcesses** iterator on a mailbox or semaphore shall show the processes waiting on the object.

   — Waiting process means either frame or task/function handle.

2) **vpiMessage** iterator shall return all the messages in a mailbox.

3) **vpiClassDefn** returns the ClassDefn that was used to create the handle. **vpiActualDefn** returns the ClassDefn that handle object points to when the query is made. The difference can be seen in the example below:

```
class Packet
  ...
endclass : Packet

class LinkedPacket extends Packet
  ...
endclass : LinkedPacket

LinkedPacket l = new;
Packet p = l;
```

   In this example, the **vpiClassDefn** of variable `p` is Packet, but the **vpiActualDefn** is “LinkedPacket”.

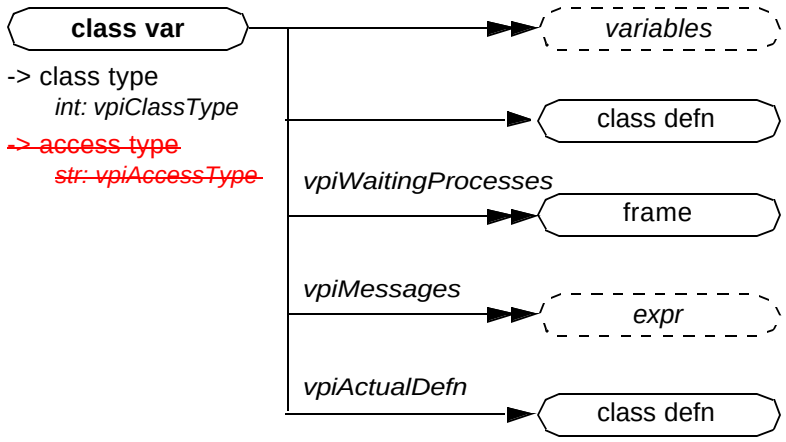
4) **vpiClassDefn/vpiActualDefn** both shall return `NULL` for built-in classes.

5) **vpiClassType** can be one of **vpiUserDefinedClass**, **vpiMailboxClass**, **vpiSemaphoreClass**.

6) **vpiAccessType** can be one of **vpiPublicAcc**, **vpiProtectedAcc**, **vpiLocalAcc**.

WITH

32.22 Class variables



NOTES:

1) The **vpiWaitingProcesses** iterator on a mailbox or semaphore shall show the processes waiting on the object.

   — Waiting process means either frame or task/function handle.

2) **vpiMessage** iterator shall return all the messages in a mailbox.

3) **vpiClassDefn** returns the ClassDefn that was used to create the handle. **vpiActualDefn** returns the ClassDefn that handle object points to when the query is made. The difference can be seen in the example below:

```
class Packet
  ...
endclass : Packet

class LinkedPacket extends Packet
  ...
endclass : LinkedPacket

LinkedPacket l = new;
Packet p = l;
```

   In this example, the **vpiClassDefn** of variable `p` is Packet, but the **vpiActualDefn** is “LinkedPacket”.

4) **vpiClassDefn/vpiActualDefn** both shall return `NULL` for built-in classes.

5) **vpiClassType** can be one of **vpiUserDefinedClass**, **vpiMailboxClass**, **vpiSemaphoreClass**.

~~6) **vpiAccessType** can be one of **vpiPublicAcc, vpiProtectedAcc, vpiLocalAcc.**~~

REPLACE

32.14 Variables (supersedes P1364 26.6.7 and 26.6.8)

NOTES

1)....

2)...

23)....

WITH

1)...

2)..

23) ...

24) **vpiVisibility** denotes the visibility (**local**, **protected**, or default) of a variable that is a class member. **vpiVisibility** shall return **vpiPublicVis** for a class member that is not **local** or **protected**, or for a variable that is not a class member.

Section 32.26 Task and function declaration (supersedes p1364 26.6.18)

REPLACE

4) **vpiReturn** shall always return a var object, even for simple returns.

WITH

4) **vpiReturn** shall always return a var object, even for simple returns.

5) **vpiVisibility** denotes the visibility (**local**, **protected**, or default) of a task or function that is a class member (a method). **vpiVisibility** shall return **vpiPublicVis** for a class member that is not **local** or **protected**, or for a task or function that is not a class member.