

Hm, try this please:

```
void Ctrl::ICall(Callback cb)
{
    LLOG("Ctrl::Call " << IsMainThread() << ", nonmain: " << NonMain);
    if(IsMainThread())
        cb();
    else {
        GuiLock __;
        CallBox cbox;
        cbox.cb = cb;
        CtrlCall = callback1(PerformCall, &cbox);
        int level = LeaveGMutexAll();
        WakeUpGuiThread();
        LLOG("Waiting for semaphore");
        if(!Thread::IsShutdownThreads())
            cbox.sem.Wait();
        EnterGMutex(level);
    }
    LLOG("-- Ctrl::Call " << IsMainThread());
}
```

```
void Ctrl::Call(Callback cb)
{
    if(IsMainThread())
        cb();
    else {
        GuiLock __;
        CallBox cbox;
        cbox.cb = cb;
        UPP::PostCallback(callback1(PerformCall, &cbox));
        int n = NonMain;
        int nn = n;
        NonMain = 0;
        for(int i = 0; i < n; i++)
            NonMainLock.Leave();
        int level = LeaveGMutexAll();
        WakeUpGuiThread();
        if(!Thread::IsShutdownThreads())
            cbox.sem.Wait();
        for(int i = 0; i < n; i++)
            NonMainLock.Enter();
        EnterGMutex(level);
    }
}
```

```
NonMain = n;  
}  
}
```

(Rationale: It might be possible that Call gets invoked by different thread than the one that locked NonMainLock - in that case, setting NonMain to zero is indeed quite a bad idea).

All that convoluted mess just because M\$ long time ago decided that HWND is per thread (not process)...
