Subject: Re: Thoughts about alternative approach to multithreading
Posted by mirek on Sun, 05 Jul 2009 20:08:48 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Sun, 05 July 2009 15:43luzr wrote on Fri, 03 July 2009 20:49I would like to, but right now I seem to be unable to describe it right. The problem was that it was user driven application and there are problems basically with "queue lag".

Maybe that the heart of problem is (was) the fact that it worked in "post" mode (not "execute") - messages (callbacks) being posted and not waiting for completition. Too often I ended with wrong events in the queue...

BTW, without posting, your method is equivalent to one mutex per instance and locking for any method call...

Mirek

1. What do you mean by "queue lag"? In my case case calling asynchronouse callback (this means: copy arguments, awake thread, post arguments and start execution), is almost as quick as calling U++ callback (I`ve posted these results above).

It is not about performance, but about race conditions.

Quote:
Personally I denied ANY types of events because I consider them absolutely artifical. The only thing which is really needed is executing some callback. So I had no problems identifying any types of events. I will appreciate any example where this approach fails (of course avoiding boundaries mentioned).

Queued callbacks and events are equivalent here.

Well, I will try to describe the example. The code was image viewer. GUI thread to manage gui and other threads to do background loading.

Background threads do loading of whole directories in advance. Now the problem was that they got events to load directory, started performing it and meanwhile user switches to another directory.

With shared access (with mutex), this is quite easily manageable (I mean, stopping loading and doing something else). With queues, not so much.

Also, the fact that I need to pass all info using events is not very handy. With new U++ GUI threading, it is much easier to read actual GUI status in the background thread and adjust GUI as needed.