Subject: Thoughts about alternative approach to multithreading
Posted by Mindtraveller on Mon, 13 Oct 2008 23:10:14 GMT
View Forum Message <> Reply to Message

Recently I`ve met description of Erlang language approache to make understandable and debuggable multithreading (MT) in applications. In short, they avoid many difficulties with MT by proposing single and unified multithreading technique.

Each thread has it`s queue and variables. Threads see each other but they don`t see each others` variables. The only way for interaction is exchanging with events. Each event is placed into thread local queue and processed consequently.

No (public) critical sections, no deadlocks, no synchronization issues. No headache. No tricky debug.

IMO this makes sense. Recently developing MT-based classes I`ve come to the same approach. And it proved to be very stable. And almost as effective as "classic" synchronization objects.

What do you think about that? Maybe we should change Thread class development vector a little? Should we deny any Mutex/Semaphore support in favor of thread local queue member functions?