Subject: Re: what about WorkQueue : public CoWork
Posted by kohait00 on Wed, 03 Feb 2010 12:31:01 GMT
View Forum Message <> Reply to Message

hi mirek,

you're almost right but it's not quite the same i think.

this way (your post) you actually have some kind of "synchronisation" barriers (scope close) till when you expect the offload work to be done. -> which is an excelent way of building some "inline parallellism" until a certain sync point, actually a great idea the more i think about..this should be posted somewhere in a CoWork docu..

while in the WorkQueue issue you post the work and even more work, *without* the need to have sync points, you dont wait for the work completion any more, all if it is independant of your further execution from the point of posting. it works most in the Amarican war manner: "Fire and Forget" it wiil execute somewhere, not here, but in fired sequential order.

the use case is maybe this one:

you have a GUI (with its main thread) doing all painting and controling, another (WorkQueue)Thread is doing the work. but in most cases you still need to do some kind of response to GUI while working, and the gui needs to remain responsive. one could imagine to start a Progress, using its ref in the posted Work to do its redaw/refresh or at least a PostCallback for that.

i dont know if it's clear or possible.. but is a quite nessesary case..when apps start to grow bigger, you cant rely on main thread alone anymore..the typical GUI app design pattern is needed.. control/woker threads, where one starts to use complecated and scary constructs with message queues and the like to inform the threads of state of each others "work".

this is basicly the idea behind that all.