**Workflow Foundation Internals (III)**

**Andrew Au**

Continue with our last article, we will work on allowing parallelism and create the workflow/host communication pattern. This pattern is actually very common, such as waiting for multiple approvers to approve a document. Last time we notice the problem is that ScheduleActivity can only be called once, so the most obvious solution is to allow multiple delegates to be put in the states. Looking at states.Frames, it is a stack and we access the top of the stack anytime. With parallelism, we want to have a stack like data structure with multiple tops. Think about it, this is really just a rooted tree. In our data structure, we will keep all the leaves and have parent pointers for the parent frames. With multiple tops, activity execution with a pointer to the set of all states will not work, because the activity doesn’t know which frame it is in. We changed the delegate from Action<State> to Action<Frame> and the rest will become obvious.



*Code Sample: Version 8*

We are using the old activities because we want to keep safe. No parallelism feature is used in the last sample, we will now introduce parallelism. Parallel is the simplest activity out of the box and we will use parallel to schedule two Reads.



*Code Sample: Version 9*

Notice the sequential execution of the work despite of the use of Parallel activity. Parallel activity essentially allows multiple outstanding works. When these work items are waiting for external signals, then these external signals are all waiting together. We used frame as dependencies for delegates, but sometimes delegates are not ready for execution not because of control dependencies, but data dependencies. These data eventually come from the host. From states, we can signal the host to wait for these data, and from the host, those dependencies could be broken when the data is ready. WF3 provided this mechanism though WorkflowQueue, and WF4 provided this through Bookmarks. Queue is sometimes overkill when the program really just request one item.



*Code Sample: Version 10*

This will be the end of our home made WF series, certainly we still have a long way to go until we get what will have today for workflow foundation. But I guess up to now you should able to appreciate what WF brings to you and how to best leverage WF for your application.