Is the FATE of Assignment Manager FATE?

What's FATE?

Fault Tolerant Executor which is introduced in Apache Accumulo(http://www.slideshare.net/acordova00/accumulo-14-features-and-roadmap) for table operations such as table creation and deletion.

Motivations

We can view FATE as a design model rather than an implementation in Accumulo(quoted from Enis) because its implementation details can be changed for HBase use cases. Imaging if current assignment manager doesn't need to handle failure recovery, it would be much cleaner and same symptom applies to table operations. As of today we leave partial state behind when them fails in the middle.

To me, FATE can

1) Provide a systematic way to simplify designs on executions of multiple operations with fault tolerance
2) Enforce developer to split a complex operation into multiple small operations with resume capability (Adampotent) ***Adampotent means: f(f'(x)) = f(x) where f'(x) denotes partial execution of f(x)***
3) Be a general framework for table operations, region assignment and others like. The framework is easy to understand so features built upon it are also easy to follow.
4) Easy to add test capabilities because FATE is a general framework which drives operation executions
5) Provide a reasoning path for debugging&resume because it keeps the state of each sub operation.

FATE In High Level

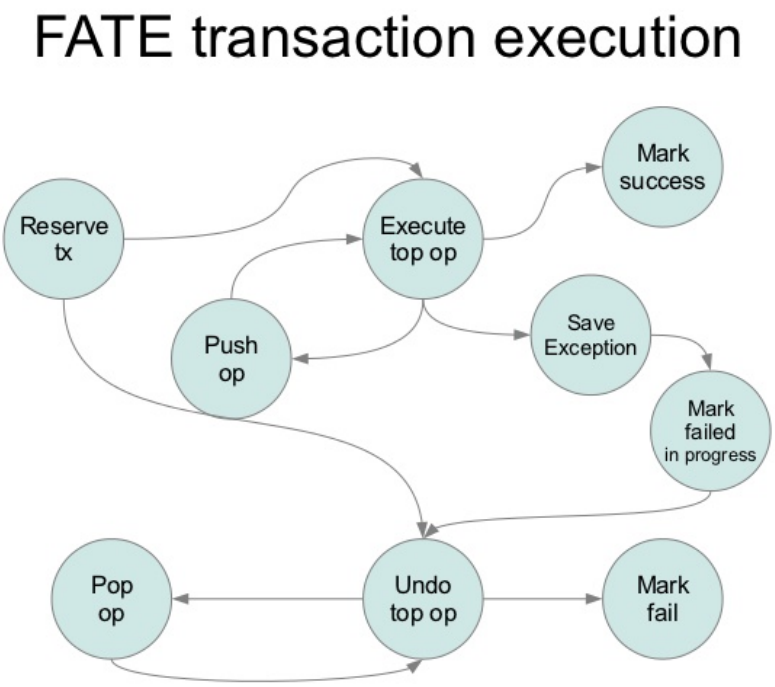
(Details @http://www.slideshare.net/acordova00/accumulo-14-features-and-roadmap)

1) If a process dies, previously submitted operations can continue to execute on restart/retry
2) Serializes operation in permanent store before execution
3) Can be coded in library for distribute FATE operations

Discussions on state permanent store

We could store transition states in Zookeeper, WAL(Write Ahead Log) or System table. Below are differences I see:

1) Zookeeper enable us to put FATE library in multiple nodes(distributed).
2) WAL file make it hard to move FATE library out of Master node
3) System table makes recovery hard because Master recovery depends on AM and if using system table then AM would depend on system table is assigned firstly, a mutual dependency/deadlock situation.



(A slide from http://www.slideshare.net/acordova00/accumulo-14-features-and-roadmap)

Envision FATE in HBase

Example: FATE Region Assignment in High Level

1) Log who is asking for a region move and reason
2) Is region in transition already(Try lock)
3) Lock region
4) Send RPC to current RS to close region
5) Wait for region is fully closed
6) Get a new hosting Region Server
7) Send RPC to the new RS to open region
8) Wait for region is opened
9) Update location(in Meta)
10) Release Lock
11) Log the region move completed

In FATE model, each operation is serialized to permanent store so we can resume from where previous operation fails. In case a total failure(after enough retries), FATE undos all operations conducted before the failure. In cases we can't undo one step for whatever reason, we can store the step where the failure comes from so that next execution can try to resume previous failed operation OR try to undo again depends on the state of running system at that time.
As you can see, FATE can help us simplify complex operation design and code in such way with fault tolerance in mind(providing Adampotent) and leave the general framework to execute.