Kernel Process Model Requirements (v0.1)

Glyn Normington

August 29, 2008

The Hebrides release of the SpringSource dm Server will be based on a separate *kernel* component. This document explores the requirements for how concurrent processes are modelled in the kernel.

Contents

1 Overview

There are three layers in the kernel process model: events, processes, and a completion service that provides callbacks when processes change state.

2 Events

Events have string names and associated value objects

$$[EventName, Value]$$

Each event has a name and an optional value object.

Event

$$name : EventName$$
$$value : Value$$

Events are correlated using correlator objects.

$$[Correlator]$$

EventWait

$$awaited : \mathbb{P}(EventName \times Correlator)$$
$$fired : EventName \times Correlator \pfun Event$$

$$awaited \cap \dom fired = \emptyset$$
$$\forall n : EventName; c : Correlator \mid (n, c) \in \dom fired \bullet$$
$$\quad fired(n, c).name = n$$

Events can be awaited. If the event has not already fired, the event is added to the awaited set of events (and the await operation blocks until the event fires).

AwaitNotFired

$$\Delta EventWait$$
$$c? : Correlator$$
$$n? : EventName$$

$$(n?, c?) \notin \dom fired$$
$$awaited' = awaited \cup \{(n?, c?)\}$$
$$fired' = fired$$

If the event has already fired, the state is not changed and the operation returns immediately.

AwaitFired

$$\Xi EventWait$$
$$c? : Correlator$$
$$n? : EventName$$

$$(n?, c?) \in \dom fired$$

Awaiting an event covers either case.

$$Await \mathrel{\widehat=} AwaitNotFire \lor AwaitFired$$

Correlators can be used to clean up old events. By definition, old events are not being waited upon.

Cleanup

$$\Delta EventWait$$
$$c? : Correlator$$

$$\forall a : awaited \bullet c? \neq second\ a$$
$$fired' = (EventName \times \{c?\}) \lhd fired$$

2.1 Serviceability Requirements

Event waits and event firings will appear in the trace.

The set of awaited events and the collection of fired events will appear in the dump.

3 Processes

Process states and transitions are defined by a process schema.

Process schemas may be checked statically for properties such as deadlock freedom.

Child processes can be related to their parent processes?

Progress of a process relative to its schema to be visible in dump and trace.

Overall process graph to be deducible from the dump.

4 Completion Service

TBD.