Modeling in Custom Instruments
Building smarter instruments

Chad Woolf, Performance Tools
Alejandro Lucena, Performance Tools
Instruments Architecture

Data Stream → Raw Events

Displayable Events → Instrument
Instruments Architecture

- Data Stream
- Raw Events
- Modeler
- Displayable Events
- Instrument
Creating Custom Instruments
Time Profiler
Transformation

Kernel → time-sample → time-profiler → time-profile → Time Profiler
Transformation

Kernel

time-sample

time-profiler

time-profile

Time Profiler
Expansion

Kernel → time-sample → time-profiler → time-profile → Time Profiler
Expansion

Kernel → time-sample → time-profiler → time-profile → Time Profiler
Expansion
Expansion

Kernel → time-sample → time-profiler → time-profile → Time Profiler
Expansion

- Kernel
- time-sample
- time-profiler
- time-profile
- Time Profiler
Themes
Themes

Absorbing complexity
Themes

Absorbing complexity

Simplicity elsewhere
Review modeling basics
Build a custom modeler
Rules execution
Speculation
Review modeling basics

Build a custom modeler

Rules execution

Speculation
Modeling Basics
Custom Instruments
Custom Instruments
Custom Instruments
Custom Instruments
Custom Instruments
When Do I Need a Custom Modeler?
When Do I Need a Custom Modeler?

Generated modelers are simplified
When Do I Need a Custom Modeler?

Generated modelers are simplified
Fuse data from multiple tables
When Do I Need a Custom Modeler?

Generated modelers are simplified
Fuse data from multiple tables
Maintain a working memory
When Do I Need a Custom Modeler?

Generated modelers are simplified
Fuse data from multiple tables
Maintain a working memory
Custom graphs
When Do I Need a Custom Modeler?

Generated modelers are simplified
Fuse data from multiple tables
Maintain a working memory
Custom graphs
Smarter instruments
Modelers

inputs → modeler → outputs

* indicates optional elements.
Modelers

Time ordered

inputs -> modeler -> outputs
Modelers

Time ordered

Working memory
Modelers

Time ordered

Working memory

CLIPS
Modelers

Time ordered
Working memory
CLIPS
Output

inputs * modeler * outputs
Getting Started
Getting Started

Decide what you want to model
Getting Started

Decide what you want to model
Getting Started

Decide what you want to model
Trace your code with os_signpost
Getting Started

Decide what you want to model

Trace your code with os_signpost

Write your rules
Building a Modeler
<table>
<thead>
<tr>
<th>Goats!</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleepy Goat</td>
</tr>
<tr>
<td>Hungry Goat</td>
</tr>
<tr>
<td>Happy Goat</td>
</tr>
<tr>
<td>Smart Goat</td>
</tr>
<tr>
<td>Billy Goat</td>
</tr>
<tr>
<td>Smart Goat</td>
</tr>
<tr>
<td>Silly Goat</td>
</tr>
<tr>
<td>Silly Goat</td>
</tr>
<tr>
<td>Sleepy Goat</td>
</tr>
<tr>
<td>Goatee</td>
</tr>
</tbody>
</table>
Goats!

- Sleepy Goat
- Hungry Goat
- Happy Goat
- Smart Goat
- Billy Goat
- Smart Goat
- Silly Goat
- Sleepy Goat
- Goatee
Goats!

- Sleepy Goat
- Hungry Goat
- Happy Goat
- Smart Goat
- Billy Goat
- Smart Goat
- Silly Goat
- Sleepy Goat
- Goatee
Dispatch Queue

Goats!

Sleepy Goat

Hungry Goat

Happy Goat

Smart Goat

Billy Goat

Smart Goat

Silly Goat

Sleepy Goat

Goatee
Dispatch Queue

Goats!

Sleepy Goat
Hungry Goat
Happy Goat
Smart Goat
Billy Goat
Smart Goat
Silly Goat
Sleepy Goat
Goatee
Dispatch Queue

Goats!

- Sleepy Goat
- Hungry Goat
- Happy Goat
- Smart Goat
- Billy Goat
- Smart Goat
- Silly Goat
- Sleepy Goat
- Goatee
Dispatch Queue

Goats!

Billy Goat

Billy Goat

Billy Goat

Billy Goat

Billy Goat

Billy Goat

Billy Goat

Billy Goat

Billy Goat

Billy Goat
Silly Goat
Mountain Goat
Happy Goat
Mountain Goat
Smart Goat
Sleepy Goat
Goatee
Silly Goat
Hungry Goat
Moving to "UI" Executing in "Get Goats"
Dispatch Queue

Moving to "UI"  Executing in "Get Goats"
Moving to “UI”

Executing in “Get Goats”

Moving to “Background”
Moving to "UI"
Executing in "Get Goats"
Moving to "Background"
Moving to "UI"
Executing in "Get Goats"
Moving to "Background"

Dispatch Queue

Executing in "Sorting"
Desired Instrument
<table>
<thead>
<tr>
<th>Time</th>
<th>Duration</th>
<th>Agent Kind</th>
<th>Stop Kind</th>
<th>State</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00.453</td>
<td>51.14 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Display Stop&quot; in mode &quot;Activating&quot;</td>
</tr>
<tr>
<td>00:00.504</td>
<td>200.15 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>executing</td>
<td>&quot;Activating&quot; at stop Display Stop</td>
</tr>
<tr>
<td>00:00.704</td>
<td>51.05 ms</td>
<td>Display Agent</td>
<td>Goat List Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Goat List Stop&quot; in mode &quot;Retrieve Goat List&quot;</td>
</tr>
<tr>
<td>00:00.756</td>
<td>201.15 ms</td>
<td>Display Agent</td>
<td>Goat List Stop</td>
<td>executing</td>
<td>&quot;Retrieve Goat List&quot; at stop Goat List Stop</td>
</tr>
<tr>
<td>00:00.957</td>
<td>50.57 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Display Stop&quot; in mode &quot;Display Goat List&quot;</td>
</tr>
<tr>
<td>00:01.077</td>
<td>205.01 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>executing</td>
<td>&quot;Display Goat List&quot; at stop Display Stop</td>
</tr>
<tr>
<td>00:01.212</td>
<td>51.07 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>Parking at</td>
<td>Parking at &quot;Display Stop&quot; in mode &quot;Finished&quot;</td>
</tr>
<tr>
<td>00:05.004</td>
<td>51.23 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Sort Stop&quot; in mode &quot;Activating&quot;</td>
</tr>
<tr>
<td>00:05.055</td>
<td>201.22 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>executing</td>
<td>&quot;Activating&quot; at stop Sort Stop</td>
</tr>
<tr>
<td>00:05.256</td>
<td>50.28 ms</td>
<td>Sorting Agent</td>
<td>Goat List Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Goat List Stop&quot; in mode &quot;Retrieve Goat List&quot;</td>
</tr>
<tr>
<td>00:05.306</td>
<td>200.40 ms</td>
<td>Sorting Agent</td>
<td>Goat List Stop</td>
<td>executing</td>
<td>&quot;Retrieve Goat List&quot; at stop Goat List Stop</td>
</tr>
<tr>
<td>00:05.507</td>
<td>50.49 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Sort Stop&quot; in mode &quot;Sort Goat List&quot;</td>
</tr>
<tr>
<td>00:05.557</td>
<td>201.44 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>executing</td>
<td>&quot;Sort Goat List&quot; at stop Sort Stop</td>
</tr>
<tr>
<td>00:05.759</td>
<td>50.13 ms</td>
<td>Sorting Agent</td>
<td>Goat List Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Goat List Stop&quot; in mode &quot;Commit Goat List&quot;</td>
</tr>
<tr>
<td>00:06.009</td>
<td>51.31 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>Parking at</td>
<td>Parking at &quot;Sort Stop&quot; in mode &quot;Finished&quot;</td>
</tr>
<tr>
<td>00:06.060</td>
<td>51.22 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Display Stop&quot; in mode &quot;Activating&quot;</td>
</tr>
<tr>
<td>00:06.112</td>
<td>201.28 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>executing</td>
<td>&quot;Activating&quot; at stop Display Stop</td>
</tr>
<tr>
<td>00:06.313</td>
<td>50.44 ms</td>
<td>Display Agent</td>
<td>Goat List Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Goat List Stop&quot; in mode &quot;Retrieve Goat List&quot;</td>
</tr>
<tr>
<td>00:06.363</td>
<td>200.75 ms</td>
<td>Display Agent</td>
<td>Goat List Stop</td>
<td>executing</td>
<td>&quot;Retrieve Goat List&quot; at stop Goat List Stop</td>
</tr>
<tr>
<td>Start</td>
<td>Duration</td>
<td>Agent Kind</td>
<td>Stop Kind</td>
<td>State</td>
<td>Activity</td>
</tr>
<tr>
<td>------------------</td>
<td>----------</td>
<td>--------------</td>
<td>-----------------</td>
<td>----------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>00:00.453.872</td>
<td>51.14 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Display Stop&quot; in mode &quot;Activating&quot;</td>
</tr>
<tr>
<td>00:00.504.810</td>
<td>200.15 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>executing</td>
<td>&quot;Activating&quot; at stop Display Stop</td>
</tr>
<tr>
<td>00:00.704.959</td>
<td>51.05 ms</td>
<td>Display Agent</td>
<td>Goat List Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Goat List Stop&quot; in mode &quot;Retrieve Goat List&quot;</td>
</tr>
<tr>
<td>00:00.756.011</td>
<td>201.15 ms</td>
<td>Display Agent</td>
<td>Goat List Stop</td>
<td>executing</td>
<td>&quot;Retrieve Goat List&quot; at stop Goat List Stop</td>
</tr>
<tr>
<td>00:00.957.162</td>
<td>50.57 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Display Stop&quot; in mode &quot;Display Goat List&quot;</td>
</tr>
<tr>
<td>00:01.007.737</td>
<td>205.01 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>executing</td>
<td>&quot;Display Goat List&quot; at stop Display Stop</td>
</tr>
<tr>
<td>00:01.212.745</td>
<td>51.07 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>Parking at</td>
<td>Parking at &quot;Display Stop&quot; in mode &quot;Finished&quot;</td>
</tr>
<tr>
<td>00:05.004.023</td>
<td>51.23 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Sort Stop&quot; in mode &quot;Activating&quot;</td>
</tr>
<tr>
<td>00:05.055.251</td>
<td>201.22 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>executing</td>
<td>&quot;Activating&quot; at stop Sort Stop</td>
</tr>
<tr>
<td>00:05.256.470</td>
<td>50.28 ms</td>
<td>Sorting Agent</td>
<td>Goat List Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Goat List Stop&quot; in mode &quot;Retrieve Goat List&quot;</td>
</tr>
<tr>
<td>00:05.306.753</td>
<td>200.40 ms</td>
<td>Sorting Agent</td>
<td>Goat List Stop</td>
<td>executing</td>
<td>&quot;Retrieve Goat List&quot; at stop Goat List Stop</td>
</tr>
<tr>
<td>00:05.507.150</td>
<td>50.49 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Sort Stop&quot; in mode &quot;Sort Goat List&quot;</td>
</tr>
<tr>
<td>00:05.557.641</td>
<td>201.44 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>executing</td>
<td>&quot;Sort Goat List&quot; at stop Sort Stop</td>
</tr>
<tr>
<td>00:05.759.078</td>
<td>50.13 ms</td>
<td>Sorting Agent</td>
<td>Goat List Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Goat List Stop&quot; in mode &quot;Commit Goat List&quot;</td>
</tr>
<tr>
<td>00:05.808.210</td>
<td>200.31 ms</td>
<td>Sorting Agent</td>
<td>Goat List Stop</td>
<td>executing</td>
<td>&quot;Commit Goat List&quot; at stop Goat List Stop</td>
</tr>
<tr>
<td>00:06.009.523</td>
<td>51.31 ms</td>
<td>Sorting Agent</td>
<td>Sort Stop</td>
<td>Parking at</td>
<td>Parking at &quot;Sort Stop&quot; in mode &quot;Finished&quot;</td>
</tr>
<tr>
<td>00:06.060.993</td>
<td>51.22 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Display Stop&quot; in mode &quot;Activating&quot;</td>
</tr>
<tr>
<td>00:06.112.213</td>
<td>201.28 ms</td>
<td>Display Agent</td>
<td>Display Stop</td>
<td>executing</td>
<td>&quot;Activating&quot; at stop Display Stop</td>
</tr>
<tr>
<td>00:06.313.492</td>
<td>50.44 ms</td>
<td>Display Agent</td>
<td>Goat List Stop</td>
<td>Moving to</td>
<td>Moving to &quot;Goat List Stop&quot; in mode &quot;Retrieve Goat List&quot;</td>
</tr>
<tr>
<td>00:06.363.934</td>
<td>200.75 ms</td>
<td>Display Agent</td>
<td>Goat List Stop</td>
<td>executing</td>
<td>&quot;Retrieve Goat List&quot; at stop Goat List Stop</td>
</tr>
</tbody>
</table>
Modeling Schematic

Application → Raw Events → Modeler → Output Rows → Output Table

Mobile Agents
Event Stream
Event Stream

- Moving
- Executing
- Moving
- Executing

- Moving to "Background"
- Executing in "Sorting"
- Moving to "UI"
- Executing in "Fill List"
// Mobile Agent Executing

func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "Agent of type %@ executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())

    execute()
}

// Mobile Agent Executing

func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "Agent of type %@ executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())

    execute()
}

// Mobile Agent Executing

func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "%Agent of type %@ executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())

    execute()
}

// Mobile Agent Moving

internal func visitNextStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Moved",
        signpostID: signpostID,
        "Agent of type %@ received by %@ for mode %@",
        agent.agentType(),
        destination.agentStopType(),
        agent.currentMode)

    // ...
}

//...
Mobile Agent Moving

```swift
internal func visitNextStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Moved",
        signpostID: signpostID,
        "Agent of type %@ received by %@ for mode %@",
        agent.agentType(),
        destination.agentStopType(),
        agent.currentMode)

    //...
}
```
// Mobile Agent Moving

internal func visitNextStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Moved",
        signpostID: signpostID,
        "Agent of type %@ received by %@ for mode %@",
        agent.agentType(),
        destination.agentStopType(),
        agent.currentMode)

    //...

}
Example — MobileAgent Activity
Example — MobileAgent Activity

os-signpost

Working Memory
Example — MobileAgent Activity

Detect MobileAgent

os-signpost

Working Memory
Example — MobileAgent Activity

Detect MobileAgent

os-signpost
Example — MobileAgent Activity

Detect MobileAgent

os-signpost
Example — MobileAgent Activity

Detect MobileAgent

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>instance</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>message</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time 0:00:42.000</td>
</tr>
<tr>
<td>instance 0x234543534</td>
</tr>
<tr>
<td>name &quot;Mobile Agent Moved&quot;</td>
</tr>
<tr>
<td>message &quot;Sort Agent to Background&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance 0x234543534</td>
</tr>
<tr>
<td>agent-kind &quot;Sort Agent&quot;</td>
</tr>
<tr>
<td>mode &quot;Moving to Background&quot;</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>instance</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>message</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

| os-signpost | time   | 0:00:42.000 |
|            | instance | 0x234543534 |
|            | name     | "Mobile Agent Moved" |
|            | message  | "Sort Agent to Background" |

| mobile-agent | instance | 0x234543534 |
|              | agent-kind | "Sort Agent" |
|              | mode       | "Moving to Background" |

| mobile-agent-movement | instance | 0x234543534 |
|                       | start    | 0:00:42.000 |
Example — MobileAgent Activity

Detect MobileAgent
Determine activity

os-signpost
instance 0x234543534
name "Mobile Agent Moved"
message "Sort Agent to Background"

mobile-agent
instance 0x234543534
agent-kind "Sort Agent"
mode "Moving to Background"

mobile-agent-movement
instance 0x234543534
start 0:00:42.000
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mobile-agent-movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>start</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mobile-agent-movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>start</td>
</tr>
</tbody>
</table>

os-signpost
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

Determine interval

os-signpost

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>0x234543534</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>&quot;Sort Agent&quot;</td>
</tr>
<tr>
<td>mode</td>
</tr>
<tr>
<td>&quot;Moving to Background&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mobile-agent-movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>0x234543534</td>
</tr>
<tr>
<td>start</td>
</tr>
<tr>
<td>0:00:42.000</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

Determine interval

mobile-agent

<table>
<thead>
<tr>
<th>instance</th>
<th>0x234543534</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent-kind</td>
<td>&quot;Sort Agent&quot;</td>
</tr>
<tr>
<td>mode</td>
<td>&quot;Moving to Background&quot;</td>
</tr>
</tbody>
</table>

mobile-agent-movement

<table>
<thead>
<tr>
<th>instance</th>
<th>0x234543534</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>0:00:42.000</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

Determine interval

```
<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mobile-agent-movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>start</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>instance</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>message</td>
</tr>
</tbody>
</table>
```
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

Determine interval

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance: 0x234543534</td>
</tr>
<tr>
<td>agent-kind: &quot;Sort Agent&quot;</td>
</tr>
<tr>
<td>mode: &quot;Moving to Background&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mobile-agent-movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance: 0x234543534</td>
</tr>
<tr>
<td>start: 00:00:42.000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time: 00:00:54.000</td>
</tr>
<tr>
<td>instance: 0x234543534</td>
</tr>
<tr>
<td>name: &quot;Mobile Agent Exec&quot;</td>
</tr>
<tr>
<td>message: &quot;Sort Agent Sorting&quot;</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

Determine interval

Output Table

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mobile-agent-movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>start</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>instance</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>message</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

Determine interval

Output Table

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>instance</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>message</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent
Determine activity
Determine interval

Output Table:

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>instance</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>message</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent
Determine activity
Determine interval

Output Table

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>0x234543534</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>&quot;Sort Agent&quot;</td>
</tr>
<tr>
<td>mode</td>
</tr>
<tr>
<td>&quot;Moving&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>0:00:54.000</td>
</tr>
<tr>
<td>instance</td>
</tr>
<tr>
<td>0x234543534</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>&quot;Mobile Agent Exec&quot;</td>
</tr>
<tr>
<td>message</td>
</tr>
<tr>
<td>&quot;Sort Agent Sorting&quot;</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent
Determine activity
Determine interval

Output Table

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
</tr>
<tr>
<td>agent-kind</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>instance</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>message</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

Determine interval

Output Table

<table>
<thead>
<tr>
<th>mobile-agent</th>
<th>instance</th>
<th>agent-kind</th>
<th>mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0x234543534</td>
<td>&quot;Sort Agent&quot;</td>
<td>&quot;Moving&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>os-signpost</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
</tr>
<tr>
<td>0:00:54.000</td>
</tr>
</tbody>
</table>
Example — MobileAgent Activity

Detect MobileAgent

Determine activity

Determine interval

Output Table

<table>
<thead>
<tr>
<th>mobile-agent</th>
<th>instance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0x234543543</td>
</tr>
</tbody>
</table>

agent-kind: "Sort Agent"

mode: "Moving"
// os-signpost

(os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,
    log: MobileAgent.signpostHandleID,
    name: "Mobile Agent Moved",
    signpostID: signpostID,
    "Agent of type %@ received by %@ for mode %@",
    agent.agentType(),
    destination.agentStopType(),
    agent.mode)
(os-signpost

  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,
   log: MobileAgent.signpostHandleID,
   name: "Mobile Agent Moved",
   signpostID: signpostID,
   "Agent of type %@ received by %@ for mode %@",
   agent.agentType(),
   destination.agentStopType(),
   agent.mode)
// os-signpost

(os-signpost
  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,
  log: MobileAgent.signpostHandleID,
  name: "Mobile Agent Moved",
  signpostID: signpostID,
  "Agent of type %@ received by %@ for mode %@",
  agent.agentType(),
  destination.agentStopType(),
  agent.mode)
// os-signpost

(os-signpost
  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,
  log: MobileAgent.signpostHandleID,
  name: "Mobile Agent Moved",
  signpostID: signpostID,
  "Agent of type %@ received by %@ for mode %@",
  agent.agentType(),
  destination.agentStopType(),
  agent.mode)
// os-signpost

(os-signpost
   (time ?t)
   (name "Mobile Agent Moved")
   (event-type "Event")
   (identifier ?instance)
   (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,
   log: MobileAgent.signpostHandleID,
   name: "Mobile Agent Moved",
   signpostID: signpostID,
   "Agent of type %@ received by %@ for mode %@",
   agent.agentType(),
   destination.agentStopType(),
   agent.mode)
(os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,
    log: MobileAgent.signpostHandleID,
    name: "Mobile Agent Moved",
    signpostID: signpostID,
    "Agent of type %@ received by %@ for mode %@",
    agent.agentType(),
    destination.agentStopType(),
    agent.mode)
// os-signpost

(os-signpost
  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,
  log: MobileAgent.signpostHandleID,
  name: "Mobile Agent Moved",
  signpostID: signpostID,
  "Agent of type %@ received by %@ for mode %@",
  agent.agentType(),
  destination.agentStopType(),
  agent.mode)
// os-signpost

(os-signpost

  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)

  (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,

  log: MobileAgent.signpostHandleID,
  name: "Mobile Agent Moved",
  signpostID: signpostID,

  "Agent of type %@ received by %@ for mode %@",
  agent.agentType(),
  destination.agentStopType(),
  agent.mode)
// os-signpost

(os-signpost
  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?)
)

os_signpost(.event,
    log: MobileAgent.signpostHandleID,
    name: "Mobile Agent Moved",
    signpostID: signpostID,
    "Agent of type %@ received by %@ for mode %@",
    agent.agentType(),
    destination.agentStopType(),
    agent.mode)
(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))
  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
)
(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))
  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
)

//MobileAgent Detection
(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))
  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
)
//MobileAgent Detection

(defrule MODELER::detect-new-mobile-agent

(os-signpost
  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?)
)

(not (mobile-agent (instance ?instance)))
=>

(assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
)
(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))

  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind))))
//MobileAgent Detection

(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))
  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
)
//MobileAgent Detection

(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?))

  (not (mobile-agent (instance ?instance)))

  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))

  )

// MobileAgent Detection

(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type \(\text{?agent-kind}\) received by \(\text{?}\)")
  )

  (not (mobile-agent (instance ?instance)))

  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))

)
//MobileAgent Detection

(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))
  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))

)
(defrule MODELER::detect-new-mobile-agent
  
  (os-signpost
     (time ?t)
     (name "Mobile Agent Moved")
     (event-type "Event")
     (identifier ?instance)
     (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))
  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
  )
(defrule MODELER::detect-new-mobile-agent

(os-signpost
  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?)
)

(not (mobile-agent (instance ?instance)))
=>

(assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
)
(defrule MODELER::detect-new-mobile-agent

(os-signpost
  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?))
)

(not (mobile-agent (instance ?instance)))
=>

(assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
)
(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))

  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))

)
(defrule MODELER::detect-new-mobile-agent

  (os-signpost
   (time ?t)
   (name "Mobile Agent Moved")
   (event-type "Event")
   (identifier ?instance)
   (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))

  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
)

//MobileAgent Detection
(defrule MODELER::detect-new-mobile-agent

    (os-signpost
      (time ?t)
      (name "Mobile Agent Moved")
      (event-type "Event")
      (identifier ?instance)
      (message$ "Agent of type " ?agent-kind " received by " $?)
    )

    (not (mobile-agent (instance ?instance)))
    =>

    (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))
  )
//MobileAgent Detection

(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?))

  (not (mobile-agent (instance ?instance)))
  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind))))
//MobileAgent Detection

(defrule MODELER::detect-new-mobile-agent

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )

  (not (mobile-agent (instance ?instance)))

  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))

)
//MobileAgent Detection

(defrule MODELER::detect-new-mobile-agent

  (os-signpost
   (time ?t)
   (name "Mobile Agent Moved")
   (event-type "Event")
   (identifier ?instance)
   (message$ "Agent of type " ?agent-kind " received by " $?)

  )

  (not (mobile-agent (instance ?instance)))

  =>

  (assert (mobile-agent (start ?t) (instance ?instance) (agent-kind ?agent-kind)))

)
(defrule MODELER::detect-mobile-agent-transition-begin
  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?))
  (mobile-agent (instance ?instance))

  =>
  (assert (mobile-agent-transition-started (start ?t) (instance ?instance)))

(defrule MODELER::detect-mobile-agent-transition-begin
  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )
  (mobile-agent (instance ?instance))

  =>
  (assert (mobile-agent-transition-started (start ?t) (instance ?instance))

)
(defrule MODELER::detect-mobile-agent-transition-begin
  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )
  (mobile-agent (instance ?instance))

  =>
  (assert (mobile-agent-transition-started (start ?t) (instance ?instance)))
  )
(defrule MODELER::detect-mobile-agent-transition-begin
  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )
  (mobile-agent (instance ?instance))

  =>
  (assert (mobile-agent-transition-started (start ?t) (instance ?instance)))
)
(defrule MODELER::detect-mobile-agent-transition-begin
  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )
  (mobile-agent (instance ?instance))

  =>
  (assert (mobile-agent-transition-started (start ?t) (instance ?instance)))
)

//MobileAgent Move
//MobileAgent Move

(defrule MODELER::detect-mobile-agent-transition-begin
    (os-signpost
        (time ?t)
        (name "Mobile Agent Moved")
        (event-type "Event")
        (identifier ?instance)
        (message$ "Agent of type " ?agent-kind " received by " $?)
    )
    (mobile-agent (instance ?instance))

    =>
    (assert (mobile-agent-transition-started (start ?t) (instance ?instance)))
)
(defrule MODELER::detect-mobile-agent-transition-begin

  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )
  (mobile-agent (instance ?instance))

  =>
  (assert (mobile-agent-transition-started (start ?t) (instance ?instance)))
)
(defrule MODELER::detect-mobile-agent-transition-begin

(os-signpost

  (time ?t)
  (name "Mobile Agent Moved")
  (event-type "Event")
  (identifier ?instance)
  (message$ "Agent of type " ?agent-kind " received by " $?)
)

(mobile-agent (instance ?instance))

=>

(assert (mobile-agent-transition-started (start ?t) (instance ?instance)))

)
(defrule MODELER::detect-mobile-agent-transition-begin
  (os-signpost
    (time ?t)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?agent-kind " received by " $?)
  )
  (mobile-agent (instance ?instance))

  =>
  (assert (mobile-agent-transition-started (start ?t) (instance ?instance)))
)

//MobileAgent Move
Rule Execution

Chad Woolf, Performance Tools
LHS => RHS
LHS $\Rightarrow$ RHS
LHS => RHS
LHS $\Rightarrow$ RHS
LHS $\Rightarrow$ RHS
LHS ⇔ RHS
LHS $\Rightarrow$ RHS
LHS $\Rightarrow$ RHS
Facts

Working Memory
(assert (mobile-agent (start ?start) (instance ?instance ...))))
### Facts

<table>
<thead>
<tr>
<th>mobile-agent</th>
</tr>
</thead>
</table>
| start       | 0:00:03.000  
| instance    | 0x234543534  
| kind        | "Sort Agent"  
| mode        | "Sorting"  
| state       | executing  

```lisp
(assert (mobile-agent (start ?start) (instance ?instance ...)))
```
### Facts

<table>
<thead>
<tr>
<th>mobile-agent</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>0:00:03.000</td>
</tr>
<tr>
<td>instance</td>
<td>0x234543534</td>
</tr>
<tr>
<td>kind</td>
<td>&quot;Sort Agent&quot;</td>
</tr>
<tr>
<td>mode</td>
<td>&quot;Sorting&quot;</td>
</tr>
<tr>
<td>state</td>
<td>executing</td>
</tr>
</tbody>
</table>
(modify ?fact-address (mode "Deliver List") (state in-transit))
(modify ?fact-address (mode "Deliver List") (state in-transit))
### Facts

<table>
<thead>
<tr>
<th>mobile-agent</th>
<th>start</th>
<th>instance</th>
<th>kind</th>
<th>mode</th>
<th>state</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0:00:03.000</td>
<td>0x234543534</td>
<td>&quot;Sort Agent&quot;</td>
<td>&quot;Deliver List&quot;</td>
<td>in-transit</td>
</tr>
</tbody>
</table>

(modify ?fact-address (mode "Deliver List") (state in-transit))
(defrule MODELER::count-mobile-agent-instances
    ?counter <- (mobile-agent-counter (count ?count))
    (mobile-agent (instance ?instance))
    =>
    (modify ?counter (count (+ 1 ?count)))
  )
(defrule MODELER::count-mobile-agent-instances
  
  ?counter <- (mobile-agent-counter (count ?count))
  (mobile-agent (instance ?instance))

  =>

  (modify ?counter (count (+ 1 ?count)))
)
(defrule MODELER::count-mobile-agent-instances
  
  ?counter <- (mobile-agent-counter (count ?count))

  (mobile-agent (instance ?instance))

  =>

  (modify ?counter (count (+ 1 ?count)))

)
(defrule MODELER::count-mobile-agent-instances
  ?counter <- (mobile-agent-counter (count ?count))
  (mobile-agent (instance ?instance))
  =>
  (modify ?counter (count (+ 1 ?count))))
(defrule MODELER::count-mobile-agent-instances
    ?counter <- (mobile-agent-counter (count ?count))
    (mobile-agent (instance ?instance))

    =>
    (modify ?counter (count (+ 1 ?count))))
(defrule MODELER::count-mobile-agent-instances
    ?counter <- (mobile-agent-counter (count ?count))  \[f-10\]
    (mobile-agent (instance ?instance))
    =>
    (modify ?counter (count (+ 1 ?count))))
(defrule MODELER::count-mobile-agent-instances
  ?counter <- (mobile-agent-counter (count ?count)) ← f-10
  (mobile-agent (instance ?instance)) ← f-100
  =>
  (modify ?counter (count (+ 1 ?count)))
)

<table>
<thead>
<tr>
<th>mobile-agent-counter</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
(defrule MODELER::count-mobile-agent-instances
  ?counter <- (mobile-agent-counter (count ?count)) ← f-10
  (mobile-agent (instance ?instance)) ← f-100
  =>
  (modify ?counter (count (+ 1 ?count))))
(defrule MODELER::count-mobile-agent-instances
  ?counter <- (mobile-agent-counter (count ?count))
  (mobile-agent (instance ?instance)) \[f-100\]
  =>
  (modify ?counter (count (+ 1 ?count))))
(defrule MODELER::count-mobile-agent-instances
  ?counter <- (mobile-agent-counter (count ?count))
  (mobile-agent (instance ?instance)) ← f-100
  =>
  (modify ?counter (count (+ 1 ?count))))

f-10

<table>
<thead>
<tr>
<th>mobile-agent-counter</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
(defrule MODELER::count-mobile-agent-instances

  ?counter <- (mobile-agent-counter (count ?count)) ← f-10
  (mobile-agent (instance ?instance)) ← f-100

  =>

  (modify ?counter (count (+ 1 ?count)))
)

f-10

<table>
<thead>
<tr>
<th>mobile-agent-counter</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
(defrule MODELER::count-mobile-agent-instances
  ?counter <- (mobile-agent-counter (count ?count))
  (mobile-agent (instance ?instance)) ← f-100
  =>
  (modify ?counter (count (+ 1 ?count))))
Stuck!

Mobile Agent Logic Loop

Timestamp      Message

00:01.632.475  Fatal error reported in run 1

Rules engine appears to be stuck
Modeler Console

--- Loading rules-0003.clp ---
--- Loading rules-0004.clp ---
MODELER> (bind ?*modeler-horizon* 618170976)

Rules engine appears to be stuck
MAIN> (watch all)
FIRE 1 count-mobile-agent-instances: f-18,f-48
<= f-18  (mobile-agent-counter (count 10995))
==> f-18  (mobile-agent-counter (count 10996))
==> Activation 0  count-mobile-agent-instances: f-18,f-48
FIRE 2 count-mobile-agent-instances: f-18,f-48
<= f-18  (mobile-agent-counter (count 10996))
==> f-18  (mobile-agent-counter (count 10997))
==> Activation 0  count-mobile-agent-instances: f-18,f-48
FIRE 3 count-mobile-agent-instances: f-18,f-48
<= f-18  (mobile-agent-counter (count 10997))
==> f-18  (mobile-agent-counter (count 10998))
(deftemplate MODELER::bump-agent-counter
  
  (slot instance (type INTEGER))
)

(defrule MODELER::count-mobile-agent-instances
  
  (mobile-agent (instance ?instance))
  =>
  (assert (bump-agent-counter (instance ?instance)))
)

(defrule MODELER::update-agent-counter
  
  ?counter <- (mobile-agent-counter (count ?count))
  ?goal <- (bump-agent-counter)
  =>
  (retract ?goal)
  (modify ?counter (count (+ ?count 1)))
)
(deftemplate MODELER::bump-agent-counter
  (slot instance (type INTEGER)))

(defrule MODELER::count-mobile-agent-instances
  (mobile-agent (instance ?instance))
  =>
  (assert (bump-agent-counter (instance ?instance)))
)

(defrule MODELER::update-agent-counter
  ?counter <- (mobile-agent-counter (count ?count))
  ?goal <- (bump-agent-counter)
  =>
  (retract ?goal)
  (modify ?counter (count (+ ?count 1)))
)
(deftemplate MODELER::bump-agent-counter
  (slot instance (type INTEGER))
)

(defrule MODELER::count-mobile-agent-instances
  (mobile-agent (instance ?instance))
  =>
  (assert (bump-agent-counter (instance ?instance)))
)

(defrule MODELER::update-agent-counter
  ?counter <- (mobile-agent-counter (count ?count))
  ?goal <- (bump-agent-counter)
  =>
  (retract ?goal)
  (modify ?counter (count (+ ?count 1)))
)
(deftemplate MODELER::bump-agent-counter
  (slot instance (type INTEGER))
)

(defrule MODELER::count-mobile-agent-instances
  (mobile-agent (instance ?instance))
  =>
  (assert (bump-agent-counter (instance ?instance)))
)

(defrule MODELER::update-agent-counter
  ?counter <- (mobile-agent-counter (count ?count))
  ?goal <- (bump-agent-counter)
  =>
  (retract ?goal)
  (modify ?counter (count (+ ?count 1)))
)
(deftemplate MODELER::bump-agent-counter
    (slot instance (type INTEGER))
)

(defrule MODELER::count-mobile-agent-instances
    (mobile-agent (instance ?instance))
    =>
    (assert (bump-agent-counter (instance ?instance)))
)

(defrule MODELER::update-agent-counter
    ?counter <- (mobile-agent-counter (count ?count))
    ?goal <- (bump-agent-counter)
    =>
    (retract ?goal)
    (modify ?counter (count (+ ?count 1)))
)
(deftemplate MODELER::bump-agent-counter
   (slot instance (type INTEGER))
)

(defrule MODELER::count-mobile-agent-instances
   (mobile-agent (instance ?instance))
   =>
   (assert (bump-agent-counter (instance ?instance)))
)

(defrule MODELER::update-agent-counter
   ?counter <- (mobile-agent-counter (count ?count))
   ?goal <- (bump-agent-counter)
   =>
   (retract ?goal)
   (modify ?counter (count (+ ?count 1)))
)
(deftemplate MODELER::bump-agent-counter
    (slot instance (type INTEGER)))

(defrule MODELER::count-mobile-agent-instances
    (mobile-agent (instance ?instance))
    =>
    (assert (bump-agent-counter (instance ?instance))))

(defrule MODELER::update-agent-counter
    ?counter <- (mobile-agent-counter (count ?count))
    ?goal <- (bump-agent-counter)
    =>
    (retract ?goal)
    (modify ?counter (count (+ ?count 1))))
Firing Order
func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "Agent of type %@ executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())
    execute()
}

func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "Agent of type %@ executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())

    execute()
}

func executeStop() {
    os_signpost(.event,
                log: MobileAgent.signpostHandleID,
                name: "Mobile Agent Exec",
                signpostID: signpostID,
                "Agent of type %@ executing mode %@. At stop %@",
                agentType(), currentMode, stop.agentStopType())

    execute()
}

"Sorting Agent" = 14 bytes
func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "Agent of type %d executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())

    execute()
}

func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "Agent of type %d executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())
    execute()
}
func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "Agent of type %d executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())

    execute()
}

0x000000001 = 4 bytes
func executeStop() {
    os_signpost(.event,
        log: MobileAgent.signpostHandleID,
        name: "Mobile Agent Exec",
        signpostID: signpostID,
        "Agent of type %d executing mode %@. At stop %@",
        agentType(), currentMode, stop.agentStopType())
    execute()
}
(deffacts MODELER::kind-code-mapping
    (agent-kind-code-to-name (kind-code 1) (kind "Sorting Agent"))
    (agent-kind-code-to-name (kind-code 2) (kind "Display Agent"))
    (agent-kind-code-to-name (kind-code 3) (kind "Editing Agent")))
(deffacts MODELER::kind-code-mapping

(agent-kind-code-to-name (kind-code 1) (kind "Sorting Agent"))
(agent-kind-code-to-name (kind-code 2) (kind "Display Agent"))
(agent-kind-code-to-name (kind-code 3) (kind "Editing Agent"))

)
(defrule MODELER::detect-new-mobile-agent
  (os-signpost
   (time ?t&~0)
   (name "Mobile Agent Moved")
   (event-type "Event")
   (identifier ?instance)
   (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
                     (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)
(defrule MODELER::detect-new-mobile-agent
  (os-signpost
    (time ?t&~0)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)
(defrule MODELER::detect-new-mobile-agent
  (os-signpost
    (time ?t&~0)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))=>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))=>
  (modify ?a (kind ?string))
)
(defrule MODELER::detect-new-mobile-agent
    (os-signpost
      (time ?t&~0)
      (name "Mobile Agent Moved")
      (event-type "Event")
      (identifier ?instance)
      (message$ "Agent of type " ?kind-code $?))
    (not (mobile-agent (instance ?instance)))
    =>
    (assert (mobile-agent (start ?t) (instance ?instance)
                          (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
    (mobile-agent (kind sentinel) (kind-code 0x1)
      (agent-kind-code-to-name (kind-code 0x1) (kind ?string))
    =>
    (modify ?a (kind ?string))
)
(defrule MODELER::detect-new-mobile-agent
  (os-signpost
    (time ?t&~0)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)
(defrule MODELER::detect-new-mobile-agent
  (os-signpost
    (time ?t&~0)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?kind-code $?
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string)))
(defrule MODELER::detect-new-mobile-agent
  (os-signpost
    (time ?t&~0)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?kind-code $?)
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)
(defrule MODELER::agent-parked
  (os-signpost (time ?end) (name "Mobile Agent Parked")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Parked in mode " ?mode)
  )
  ?a <- (mobile-agent (instance ?instance) (kind ?kind))

  =>
  ;; Actions
  )
(defrule MODELER::agent-parked
  (os-signpost (time ?end) (name "Mobile Agent Parked")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Parked in mode " ?mode)
  )
  ?a <- (mobile-agent (instance ?instance) (kind ?kind))
  ?transition-begin <- (mobile-agent-transition-started (start ?start) (instance ?instance)
    (stop-kind ?stop-kind&~sentinel) (state ?state))

  =>
  ;; Actions
  )
(defrule MODELER::agent-parked
    (os-signpost (time ?end) (name "Mobile Agent Parked")
      (event-type "Event")
      (identifier ?instance)
      (message$ "Parked in mode " ?mode)
    )
    ?a <- (mobile-agent (instance ?instance) (kind ?kind&~sentinel))
    ?transition-begin <- (mobile-agent-transition-started (start ?start) (instance ?instance)
      (stop-kind ?stop-kind&~sentinel) (state ?state))
    =>
    ;; Actions
    )
(defrule MODELER::detect-new-mobile-agent
  (os-signpost (time ?t&~0)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?kind-code $?)
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
  (declare (salience 100))
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)
(defrule MODELER::detect-new-mobile-agent
  (os-signpost (time ?t&~0)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code))))

(defrule MODELER::lookup-agent-kind
  (declare (salience 100))
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string)))
(defrule MODELER::detect-new-mobile-agent
  (os-signpost (time ?t&~0)
    (name "Mobile Agent Moved")
    (event-type "Event")
    (identifier ?instance)
    (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
)

(defrule MODELER::lookup-agent-kind
  (declare (salience 100))
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string)))
)
LHS => RHS
Activation

- f-124: open-interval
- f-17: mobile-agent
- f-345: os-signpost
- f-29: mobile-agent
- f-231: open-interval
- f-310: open-interval

Activation

Rule 1: f-345, f-29, f-124

Agenda
<table>
<thead>
<tr>
<th>Rule</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule 1</td>
<td>f-345, f-29, f-124</td>
</tr>
<tr>
<td>rule 99</td>
<td>f-345, f-17, f-231</td>
</tr>
<tr>
<td>rule 15</td>
<td>f-93, f-230</td>
</tr>
<tr>
<td>rule 9</td>
<td>f-17</td>
</tr>
<tr>
<td>rule 12</td>
<td>f-99, f-102, f-231</td>
</tr>
<tr>
<td>rule 14</td>
<td>f-99, f-17</td>
</tr>
</tbody>
</table>
Agenda

rule 1: f-345, f-29, f-124

rule 99: f-345, f-17, f-231

rule 15: f-93, f-230

rule 9: f-17

rule 12: f-99, f-102, f-231

rule 14: f-99, f-17
<table>
<thead>
<tr>
<th>Rule</th>
<th>Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>f-345, f-29, f-124</td>
</tr>
<tr>
<td>99</td>
<td>f-345, f-17, f-231</td>
</tr>
<tr>
<td>15</td>
<td>f-93, f-230</td>
</tr>
<tr>
<td>9</td>
<td>f-17</td>
</tr>
<tr>
<td>12</td>
<td>f-99, f-102, f-231</td>
</tr>
<tr>
<td>14</td>
<td>f-99, f-17</td>
</tr>
<tr>
<td>Rule</td>
<td>F-formula</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
<td>f-345, f-29, f-124</td>
</tr>
<tr>
<td>99</td>
<td>f-345, f-17, f-231</td>
</tr>
<tr>
<td>15</td>
<td>f-93, f-230</td>
</tr>
<tr>
<td>9</td>
<td>f-17</td>
</tr>
<tr>
<td>12</td>
<td>f-99, f-102, f-231</td>
</tr>
<tr>
<td>14</td>
<td>f-99, f-17</td>
</tr>
</tbody>
</table>

(retract f-17)
<table>
<thead>
<tr>
<th>Rule</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>f-345, f-29, f-124</td>
</tr>
<tr>
<td>99</td>
<td>f-345, f-17, f-231</td>
</tr>
<tr>
<td>15</td>
<td>f-93, f-230</td>
</tr>
<tr>
<td>9</td>
<td>f-17</td>
</tr>
<tr>
<td>12</td>
<td>f-99, f-102, f-231</td>
</tr>
<tr>
<td>14</td>
<td>f-99, f-17</td>
</tr>
</tbody>
</table>
Agenda

rule 1: f-345, f-29, f-124
rule 99: f-345, f-17, f-231
rule 15: f-93, f-230
rule 9: f-17
rule 12: f-99, f-102, f-231
rule 14: f-99, f-17
<table>
<thead>
<tr>
<th>Rule</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule 1</td>
<td>f-345, f-29, f-124</td>
</tr>
<tr>
<td>Rule 99</td>
<td>f-345, f-17, f-231</td>
</tr>
<tr>
<td>Rule 15</td>
<td>f-93, f-230</td>
</tr>
<tr>
<td>Rule 12</td>
<td>f-99, f-102, f-231</td>
</tr>
</tbody>
</table>
Separate Agendas
Separate Agendas

Salience
Separate Agendas

Salience

Modules*

*Modules in CLIPS are not related to modules in Swift.
Separate Agendas

Salience

Modules*

Standard modules

*Modules in CLIPS are not related to modules in Swift.
Separate Agendas

Salience

Modules*

Standard modules

*Modules in CLIPS are not related to modules in Swift.
Separate Agendas

Salience

Modules*

Standard modules

MODELER
RECODER

*Modules in CLIPS are not related to modules in Swift.
Separate Agendas

Salience

Modules*

Standard modules

<table>
<thead>
<tr>
<th>MODELER</th>
<th>RECORDER</th>
</tr>
</thead>
</table>

### Standard modules

- **MODELER**
  - **detect-agent:** f-345
  - **close-interval:** f-345, f-10
  - **open-interval:** f-345, f-234

- **RECORDER**
  - **record-interval:** f-986

*Modules in CLIPS are not related to modules in Swift.*
Separate Agendas

Salience

Modules*

Standard modules

MODELER

RECORDER

detect-agent: f-345

close-interval: f-345,f-10

open-interval: f-345, f-234

record-interval: f-986

*Modules in CLIPS are not related to modules in Swift.
Separate Agendas

Salience

Modules*

Standard modules

*Modules in CLIPS are not related to modules in Swift.
Separate Agendas

Salience

Modules*

Standard modules

*Modules in CLIPS are not related to modules in Swift.
Separate Agendas

Salience

Modules*

Standard modules

Custom modules

*Modules in CLIPS are not related to modules in Swift.
(defmodule LOOKUP)
  (import MODELER ?ALL)
)
(defrule LOOKUP::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)
(defrule MODELER::detect-new-mobile-agent
  (os-signpost (time ?t&~0) (name "Mobile Agent Moved") (event-type "Event")
    (identifier ?instance) (message$ "Agent of type " ?kind-code $?)
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
  (focus LOOKUP)
)
(defmodule LOOKUP
   (import MODELER ?ALL))

(defrule LOOKUP::lookup-agent-kind
   ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
   (agent-kind-code-to-name (kind-code ?code) (kind ?string))
=>
   (modify ?a (kind ?string))
)

(defrule MODELER::detect-new-mobile-agent
   (os-signpost (time ?t&~0) (name "Mobile Agent Moved") (event-type "Event")
      (identifier ?instance) (message$ "Agent of type " ?kind-code $?))
   (not (mobile-agent (instance ?instance)))
=>
   (assert (mobile-agent (start ?t) (instance ?instance)
      (kind sentinel) (kind-code ?kind-code)))
   (focus LOOKUP)
)
(defmodule LOOKUP
  (import MODELER ?ALL)
)

(defrule LOOKUP::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)

(defrule MODELER::detect-new-mobile-agent
  (os-signpost (time ?t&~0) (name "Mobile Agent Moved") (event-type "Event")
    (identifier ?instance) (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
  (focus LOOKUP)
)
(defmodule LOOKUP
  (import MODELER ?ALL)
)

(defunrule LOOKUP::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)

(defunrule MODELER::detect-new-mobile-agent
  (os-signpost (time ?t&~0) (name "Mobile Agent Moved") (event-type "Event")
    (identifier ?instance) (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
  (focus LOOKUP)
)
(defmodule LOOKUP
  (import MODELER ?ALL))

(defrule LOOKUP::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)

(defrule MODELER::detect-new-mobile-agent
  (os-signpost (time ?t&~θ) (name "Mobile Agent Moved") (event-type "Event")
    (identifier ?instance) (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))

(focus LOOKUP)
)
(defmodule LOOKUP
  (import MODELER ?ALL))
)
(defrule LOOKUP::lookup-agent-kind
  ?a <- (mobile-agent (kind sentinel) (kind-code ?code))
  (agent-kind-code-to-name (kind-code ?code) (kind ?string))
  =>
  (modify ?a (kind ?string))
)
(defrule MODELER::detect-new-mobile-agent
  (os-signpost (time ?t&~0) (name "Mobile Agent Moved") (event-type "Event")
    (identifier ?instance) (message$ "Agent of type " ?kind-code $?))
  (not (mobile-agent (instance ?instance)))
  =>
  (assert (mobile-agent (start ?t) (instance ?instance)
    (kind sentinel) (kind-code ?kind-code)))
  (focus LOOKUP))
)
Debugging and Profiling

Alejandro Lucena, Performance Tools
Logging
Logging

printf style tracing
Logging

printf style tracing

Dynamically enabled/disabled
(defrule MODELER::lookup-known-agent-kind
    ?agent <- (mobile-agent (instance ?instance) (kind-code ?kind-code))

    (agent-kind-code-to-name (kind-code ?kind-code) (kind ?kind)) =>
    (modify ?agent (kind ?kind))
    (log-narrative "Resolved agent kind code %uint64% to %string%" ?kind-code ?kind)
Profiling
Profiling

Rule activation counts
Profiling

Rule activation counts

Time distribution
Demo

Alejandro Lucena, Performance Tools
Speculation
What Happened
What Happened

Long intervals
What Happened

Long intervals

Only in working memory
What Happened

Long intervals

Only in working memory

UI only sees the output tables
What Happened

Long intervals
Only in working memory
UI only sees the output tables
Speculation mode
If this were your last chance to write, what would you write?
Speculation

Input

Output

Moving to "Background"
Speculation

Input: Moving
Output: Moving to "Background"

Event Horizon

Executing

Moving
Speculation Mode

| speculate          | event-horizon | 0:00:03.000 |

Working Memory
Speculation Mode

Speculate fact

<table>
<thead>
<tr>
<th>speculate</th>
</tr>
</thead>
<tbody>
<tr>
<td>event-horizon</td>
</tr>
</tbody>
</table>

Working Memory
Speculation Mode

Speculate fact

Added to rules

<table>
<thead>
<tr>
<th>speculate</th>
<th>event-horizon</th>
<th>0:00:03.000</th>
</tr>
</thead>
</table>

Working Memory
Speculation Mode

Speculate fact
Added to rules
Write open intervals

<table>
<thead>
<tr>
<th>speculate</th>
<th>event-horizon</th>
<th>0:00:03.000</th>
</tr>
</thead>
</table>

Working Memory
(defrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
    (instance ?instance) (stop-kind ?stop-kind&~sentinel))
=>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)

(defrule RECORDER::speculatively-record-execution
    (speculate (event-horizon ?end))
    (table (table-id ?output) (side append))
    (table-attribute (table-id ?output) (has schema mobile-agent-activity))
    (mobile-agent-execution-started (start ?start)
        (instance ?instance) (stop-kind ?stop-kind&~sentinel))
    =>
    (bind ?duration (- ?end ?start))
    (create-new-row ?output)
    (set-column start ?start)
    (set-column duration ?duration)
    (set-column instance ?instance)
    (set-column state "Executing")
    (set-column activity-type "Green")
    (set-column stop-kind ?stop-kind)
    (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)
(defrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
    (instance ?instance) (stop-kind ?stop-kind&~sentinel))
  =>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)
(defrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
    (instance ?instance) (stop-kind ?stop-kind&~sentinel))

  =>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)

(defrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
    (instance ?instance) (stop-kind ?stop-kind)&~sentinel))
=>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)

(deffrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
     (instance ?instance) (stop-kind ?stop-kind&~sentinel))
=>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)
(defrule RECORDER::speculatively-record-execution
    (speculate (event-horizon ?end))
    (table (table-id ?output) (side append))
    (table-attribute (table-id ?output) (has schema mobile-agent-activity))
    (mobile-agent-execution-started (start ?start)
        (instance ?instance) (stop-kind ?stop-kind&~sentinel))

    =>
    (bind ?duration (- ?end ?start))
    (create-new-row ?output)
    (set-column start ?start)
    (set-column duration ?duration)
    (set-column instance ?instance)
    (set-column state "Executing")
    (set-column activity-type "Green")
    (set-column stop-kind ?stop-kind)
    (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)
(defrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
    (instance ?instance) (stop-kind ?stop-kind&~sentinel))
  =>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
(defrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
    (instance ?instance) (stop-kind ?stop-kind&~sentinel))
  =>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
  )
(defrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
    (instance ?instance) (stop-kind ?stop-kind&~sentinel))

  =>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)
(defrule RECORDER::speculatively-record-execution
  (speculate (event-horizon ?end))
  (table (table-id ?output) (side append))
  (table-attribute (table-id ?output) (has schema mobile-agent-activity))
  (mobile-agent-execution-started (start ?start)
    (instance ?instance) (stop-kind ?stop-kind&~sentinel))
=>
  (bind ?duration (- ?end ?start))
  (create-new-row ?output)
  (set-column start ?start)
  (set-column duration ?duration)
  (set-column instance ?instance)
  (set-column state "Executing")
  (set-column activity-type "Green")
  (set-column stop-kind ?stop-kind)
  (set-column-narrative activity "Executing at stop %string%" ?stop-kind)
)
Immediate Mode
Immediate Mode

Purged
Immediate Mode

Purged

Become real when trace ends
Speculation in Action
Speculation in Action
Summary

Investment
Summary

Investment

Intelligence
Summary

Investment

Intelligence

Efficiency
More Information

developer.apple.com/wwdc19/421