

Apex



Michael Freed (project lead)

Pete Bonasso
Michael Dalal
Ellen Drascher
Will Fitzgerald
Dawn Fitzpatrick
Robert Harris
Reagan Jew

Outline

- Project Overview
 - Applications Synopsis
 - Development Approach
 - System Components
- Execution Services and Agent Architecture
 - Basic procedure language and semantics
 - Complex Event Monitoring
 - Multitask Management
- Sherpa Develop/Debug/Demo Environment
- Applications in more Detail

Applications

The Apex project provides autonomy technology for a wide range of applications, each having:

- demanding AI functionality requirements
- low-medium barriers to acceptance of autonomy technology

Applications

**Real
Robot**

Autonomous Rotorcraft Project
intelligent surveillance and reconnaissance

**Simulated
Robot**

Mission Simulation Facility / REF
Riptide High-fidelity flight simulation
AuRA Wildfire detection, Earth Science
X-Plane Flight failure detection/recovery

**Real
Human**

Astronaut Procedure Guidance

**Simulated
Human**

CPM-GOMS HCI Analysis
VAMS Virtual Participants in HIL Simulations
MIDAS HCI Analysis
Dynamic Research Inc. Accident Analysis

Educational Outreach
Distributed on NASA website



Project Goals and Approach

Building/supporting many applications is a research and tool development strategy...

Goals

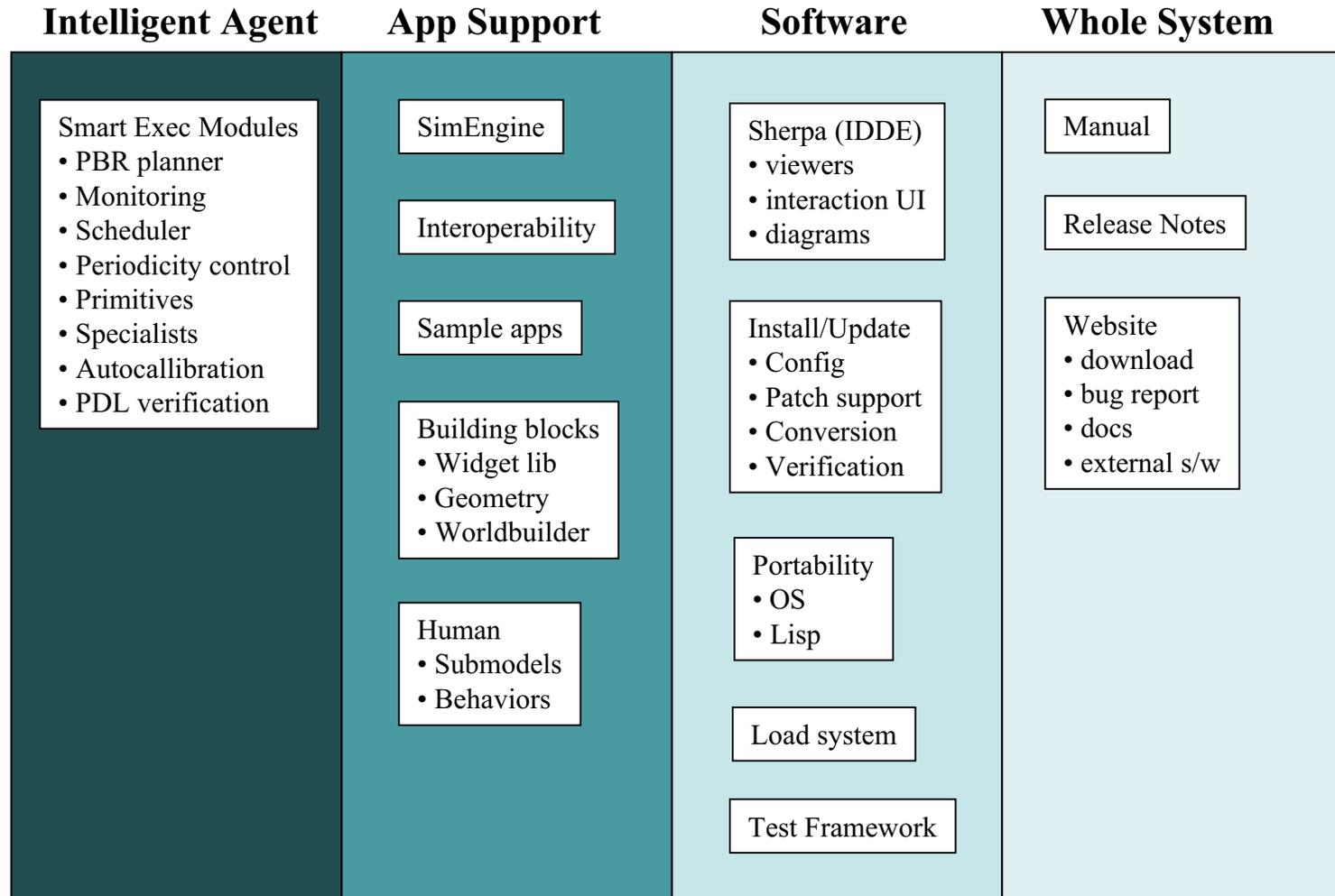
- *Versatility*: >apps drive development of diverse capabilities
- *Scalability*: >apps justifies >effort enhancing system qualities
- *Usefulness*: >apps focuses effort on improving leverage
- *Trust*: >apps and >time “prove” software reliability

Approach

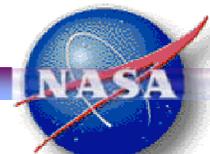
- Many applications; iterative refinement from lessons learned
- Prime directive: USABILITY
 - Reduce time, expertise, inventiveness required to build application
 - E.g. representation language, debugging support



Apex System Components



Apex



Basic procedure representation and execution semantics



Procedure Definition Language

Basic Functionality

```
(procedure
  (index (hold-altitude using mcp))
  (profile right-hand)
  (step s1 (clear right-hand))
  (step s2 (find-loc alt-hold-button => ?loc))
  (step s3 (press-button ?loc right-hand)
    (waitfor (empty right-hand)
      (location alt-hold-button ?loc)))
  (step end (terminate)
    (waitfor (illuminated alt-hold-button)))
  (step aux1 (restart ?self)
    (waitfor (resumed ?self))))
```

- concurrency
- reactivity
- abstraction/refinement
- contingency-handling
- multitask management



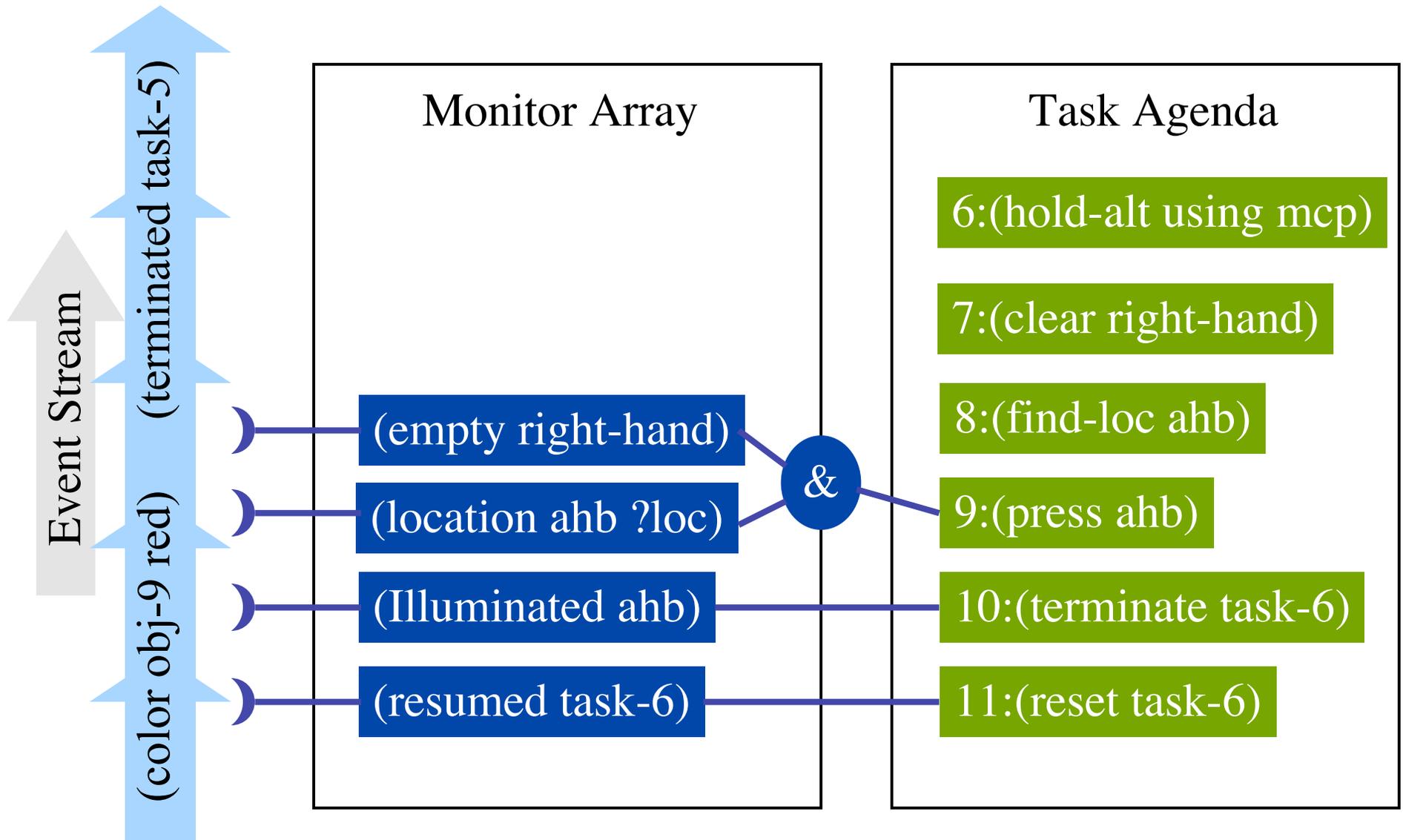
Procedure Definition Language

Usability Considerations

- readability
- compactness
- intuitiveness
- RT correspondence
- expressiveness

```
(procedure
  (index (hold-altitude using mcp))
  (profile right-hand)
  (step s1 (clear right-hand))
  (step s2 (find-loc alt-hold-button => ?loc))
  (step s3 (press-button ?loc right-hand)
    (waitfor (empty right-hand)
      (location alt-hold-button ?loc)))
  (step end (terminate)
    (waitfor (illuminated alt-hold-button)))
  (step aux1 (restart ?self)
    (waitfor (resumed ?self))))
```

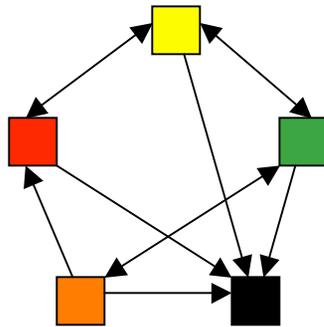
Basic execution semantics



Task control

A TASK represents decisions about (constraints on) how to transition to world states prescribed by a plan.

- Pending
- Enabled
- Ongoing
- Suspended
- Terminated



Task Agenda

STATE

6:(hold-alt using mcp) ■

7:(clear right-hand) ■

8:(find-loc ahb) ■

9:(press ahb) ■

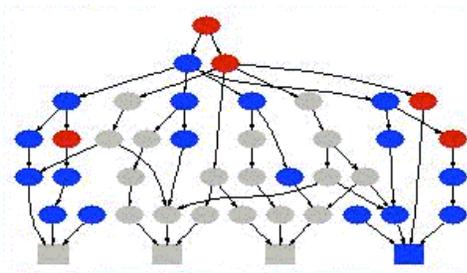
10:(terminate task-6) ■

11:(reset task-6) ■

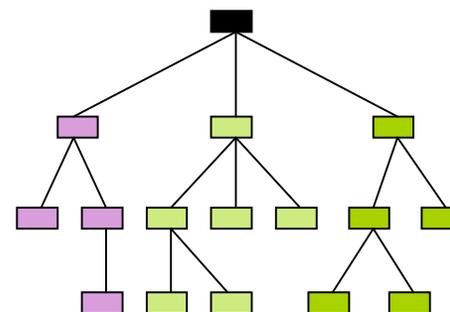


Understanding Apex Behavior

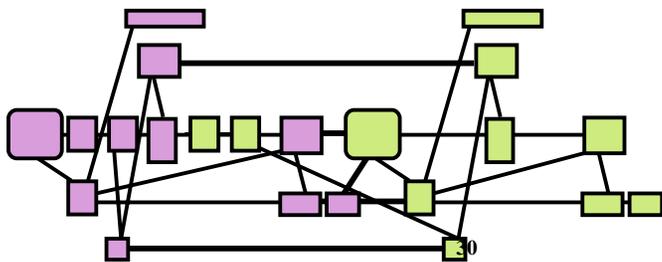
AI underpinnings



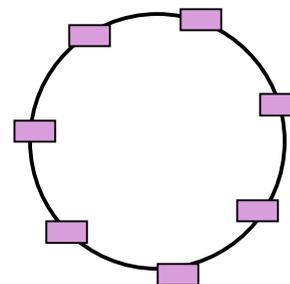
Event Monitoring



Procedure-based Reactive Planning



Dynamic Scheduling



Periodicity Control

Apex



Complex Event Monitoring

Monitoring in Apex

The WAITFOR clause generates **monitors** that look for specified **event patterns**, Detecting enabling conditions for both nominal and contingent tasks.

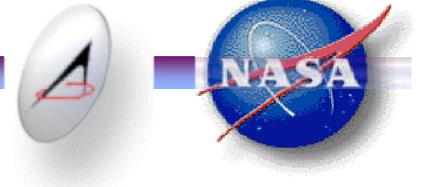
Basic event patterns

- Propositional forms (possibly with vars)
- Conjunctions
- Disjunctions

```
(procedure
  (index (hold-altitude using mcp))
  (profile right-hand)
  (step s1 (clear right-hand))
  (step s2 (find-loc alt-hold-button => ?loc))
  (step s3 (press-button ?loc right-hand)
    → (waitfor (empty right-hand)
              (location alt-hold-button ?loc)))
  (step end (terminate)
    (waitfor (illuminated alt-hold-button))
  (step aux1 (restart ?self)
    (waitfor (resumed ?self))))
```



Enhanced Event Monitoring



More sophisticated monitoring approach desirable in many Apex domains to support contingency detection, diagnosis and human intent inference.

History-dependent monitoring including time-series data analysis

- e.g. Altitude decreasing monotonically for time $> k$ AND no altitude decrease commanded
- e.g. Valve closure command sent followed within 2 seconds by valve-closed signal received
- e.g. High turbulence interval overlaps loss of communication interval

Unification of querying and monitoring

- e.g. The temperature of instrument A previously fell or later falls below T

Execution-time coordination of monitoring with commanding

- e.g. Distance to car in front of me $< d$ whose value depends on my target speed

Explicit constraints on data quality

- e.g. The temperature of A has held steady with measurements arriving at least 1/second

Uniform representation based on constraints, attributes, intervals

- * As needed to integrate with CAIP planner/schedulers such as Europa II



Enhanced Monitoring Ontology

Measurement	input signal : attribute, object, val, timestamp e.g. (altitude aircraft-1 = 34000)
Estimation	inferred measurement
<hr/>	
Simple Episode	abstraction of measurement history for att-obj pair e.g. (holding (alt aircraft-1) <time1> <time2>)
Complex Episode	logical/temporal relations over simple episodes e.g. (in-order (holding...) (descending...))
Atomic Episode	input signal denoting simple/complex episode but not specifying underlying measurement history e.g. (landed aircraft-1)



Enhanced Monitoring Example

Complain if the stove is above 200 degrees for at least 2 minutes

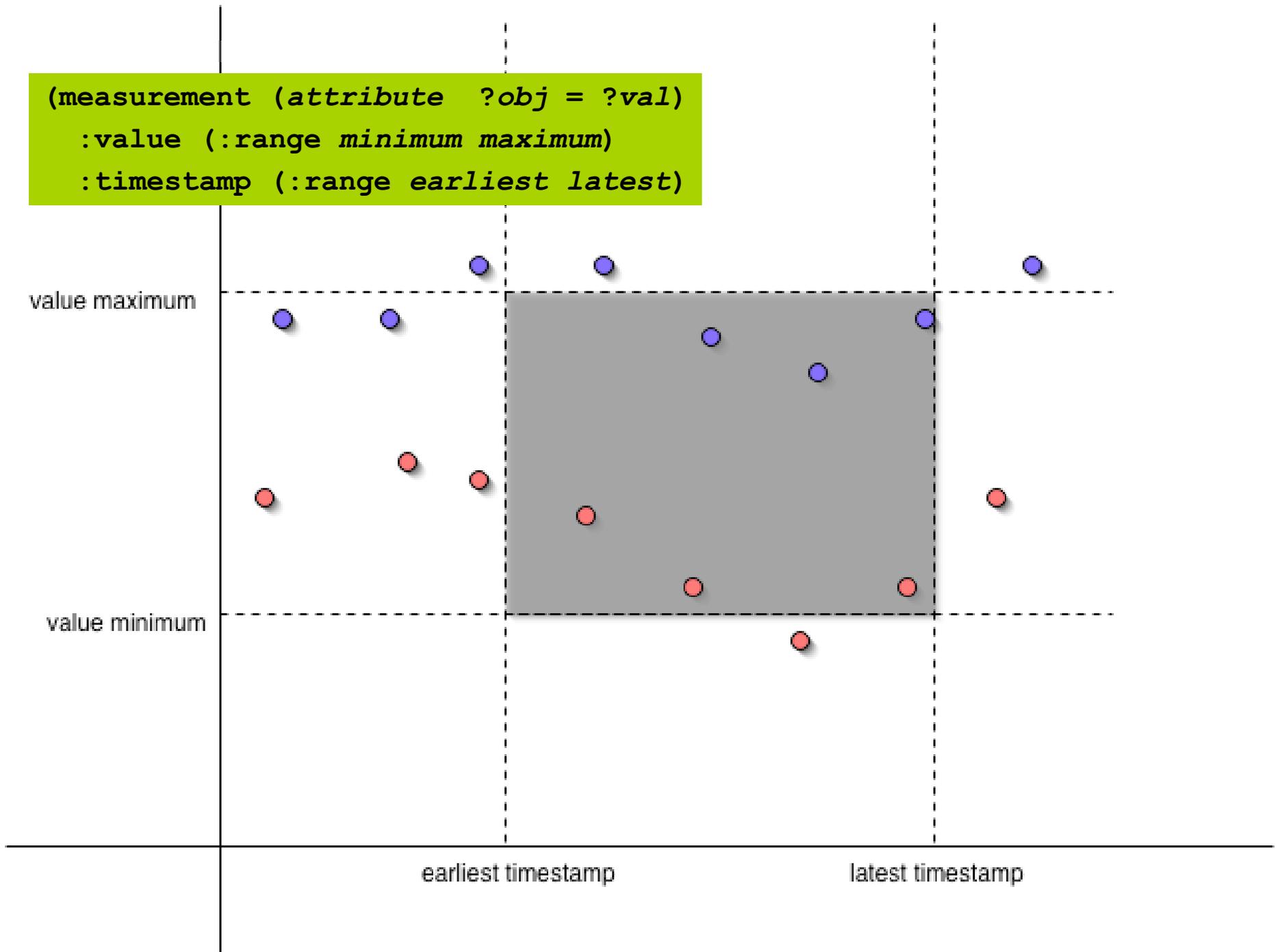
```
(procedure (warn-if-stove-too-hot-too-long ?stove)
  (log (temp <?stove>))
  (step s1 (say "Hey, the stove is too hot!")
    (waitfor (:episode (temp <?stove>
      :minimum-sample-interval P1S
      :stats (:mean (>= 200))
      :timing (:end (<= (+ (start-of +this-task+) P10M)))
      (:duration (:min P2M))))))
  (step term (terminate)
    (waitfor ?s1)))
```

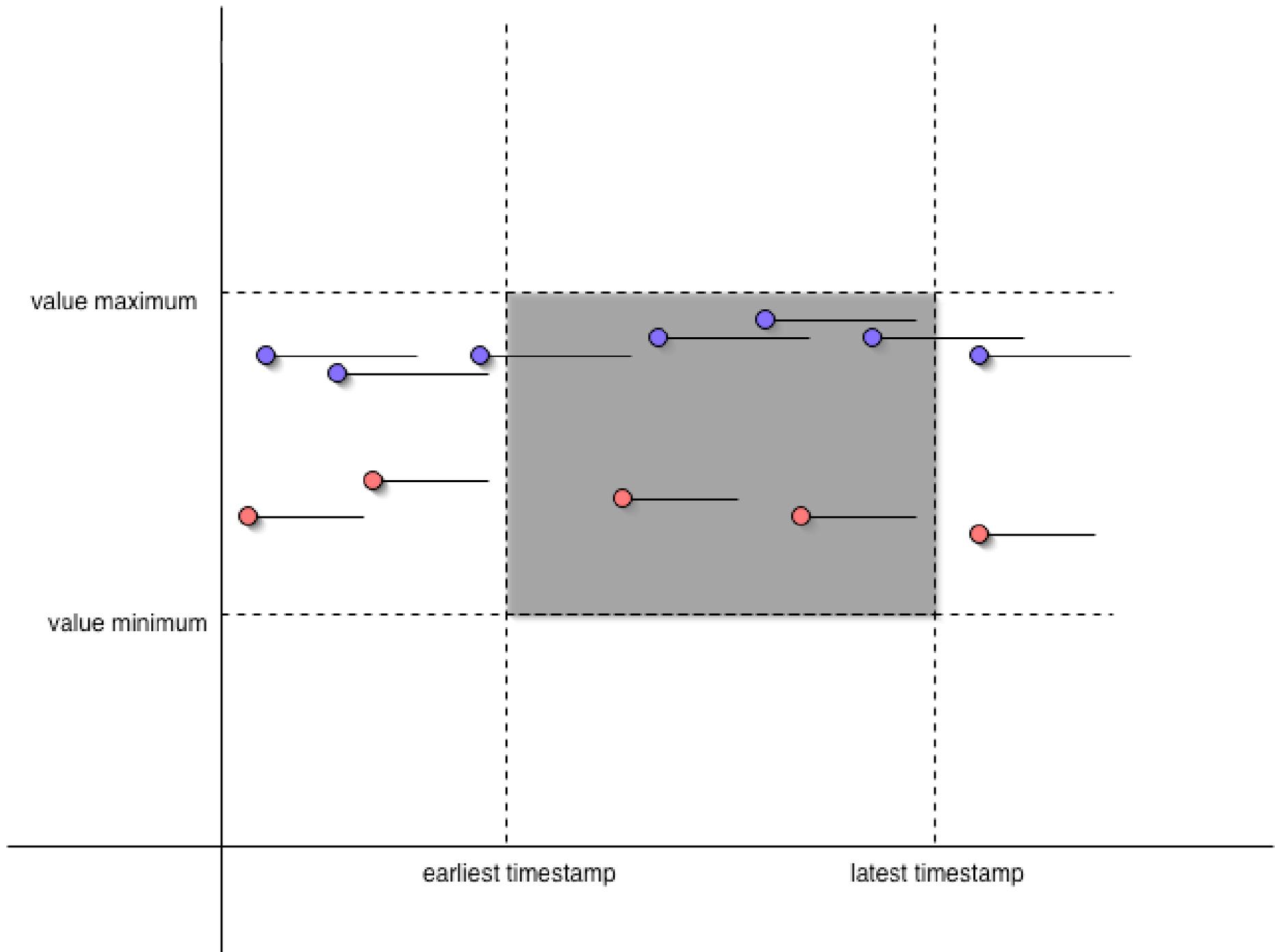
A second example

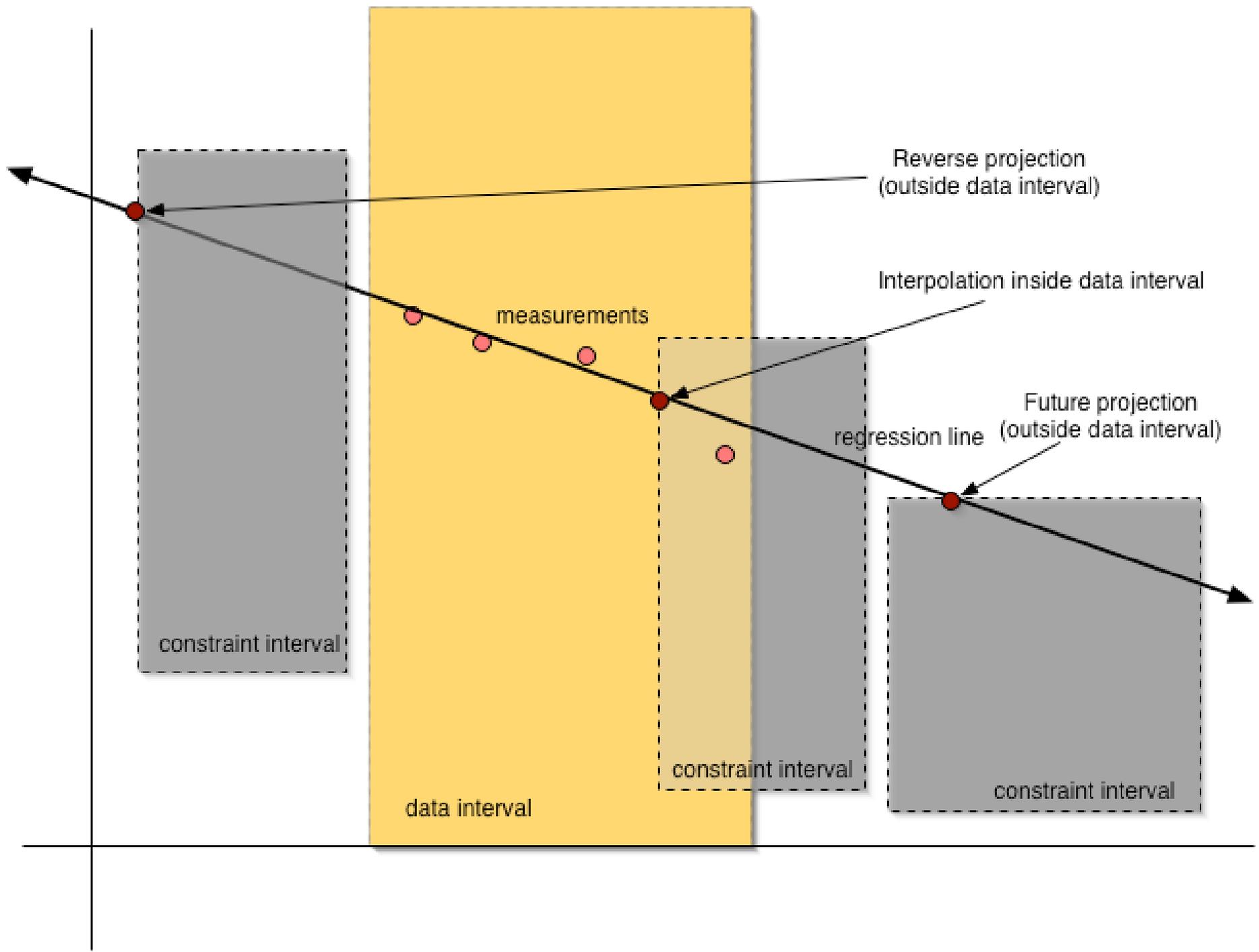
Wait for an interval in which a specified aircraft starts at cruise altitude, descends over a period starting after TASK-1 begins and lasting at least 5 minutes, ending at approach altitude and approach speed.

```
(waitfor
  (:in-order
    (:measurement m1 (altitude <?aircraft> = +cruise-altitude+))
    (:episode e1 (altitude <?aircraft>
      (:minimum-sample-interval P10s)
      (:timing (:start (> (start-of ?task-1))) (:duration (> P5m)))
      (:trend (:rate :increasing)))
    (:and
      (:measurement m2 (altitude <?aircraft> = +approach-altitude+))
      (:measurement m3 (airspeed <?aircraft> = +approach-speed+))))))
```

```
(measurement (attribute ?obj = ?val)  
  :value (:range minimum maximum)  
  :timestamp (:range earliest latest))
```





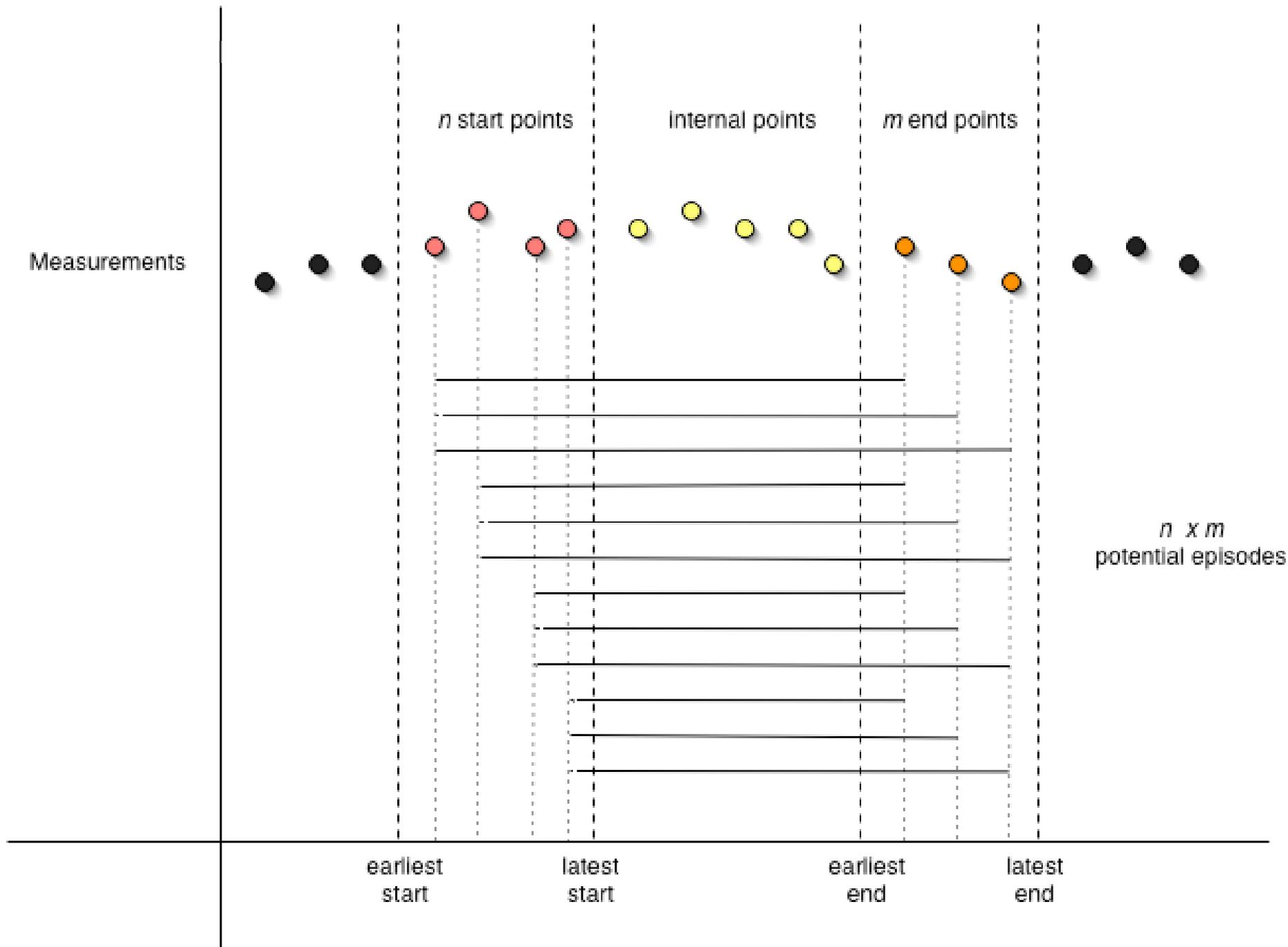


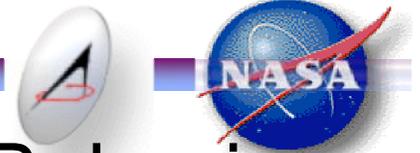
Episode syntax

1. (:episode [<tag>] (<attr> <obj/v>))
2. :minimum-sample-interval |
:msi <duration-spec>
3. [:timing <timing-constraint>]*]*
4. [:value <constraint>]*]*
5. [:first-value <constraint>]*]*
6. [:last-value <constraint>]*]*
7. [:object <constraint>]*]*
8. [:stats <stat-constraint>]*]*
9. [:trend <trend-constraint>]*]*)

Timing constraint

1. (:start <constraint>*) |
2. (:end <constraint>*) |
3. (:earliest-start <time-spec>) | (:es <time-spec>) |
4. (:latest-start <time-spec>) | (:ls <time-spec>) |
5. (:earliest-end <time-spec>) | (:ee <time-spec>) |
6. (:latest-end <time-spec>) | (:le <time-spec>) |
7. (:duration <constraint>*)





Understanding and Debugging Apex Behavior

A Big Challenge!

Thousands of lines of trace: not so helpful

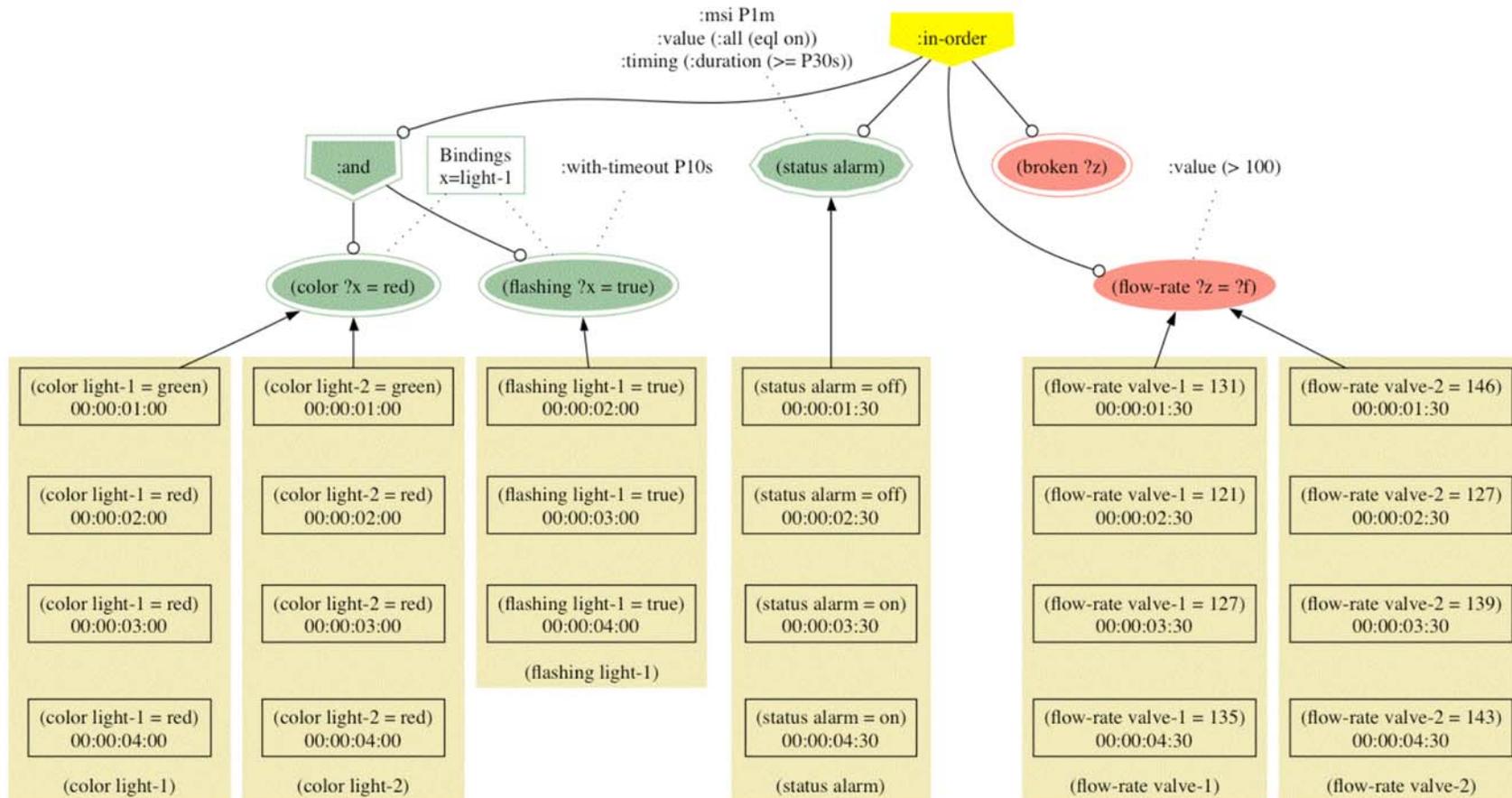
```
[5050-A] ..CREATING [TASK5735 (MAP EYE-TARGET [VOF:{DL603}] TO HAND-TARGET) {NIL}]
[5050-A] ..CREATING [TASK5736 (COMPUTE-POINTER-ICON-DISTANCE [VOF:{DL603}] ?POINTER) {NIL}]
[5050-A] ..CREATING [TASK5737 (SHIFT-GAZE-TO [VOF:{DL603}])] {NIL}]
[5050-A] ..CREATING [TASK5738 (FIND MOUSE POINTER) {NIL}]
[5050-A] ..CREATING [TASK5739 (GRASP LEFT MOUSE) {NIL}]
[5050-A] TESTING preconditions for
[TASK5739 (GRASP LEFT MOUSE) {PENDING}].... SATISFIED
[5050-A] SELECTING procedure for TASK5739... => (GRASP ?HAND ?OBJECT)
[5050-A] ENABLING [TASK5739 (GRASP LEFT MOUSE) {ENABLED}]
[5050-A] EXECUTING [TASK5739 (GRASP LEFT MOUSE) {ENABLED}]
[5050-A] ..CREATING [TASK5745 (TERMINATE ?SELF SUCCESS >> NIL) {NIL}]
[5050-A] ..CREATING [TASK5746 (RESET ?SELF) {NIL}]
[5050-A] ..CREATING [TASK5747 (SIGNAL-RESOURCE LEFT (GRASP MOUSE)) {NIL}]
[5050-A] ..CREATING [TASK5748 (CLEAR-HAND LEFT) {NIL}]
[5050-A] ..CREATING [TASK5749 (GRASP-STATUS LEFT MOUSE) {NIL}]
[5050-A] ENABLING [TASK5747 (SIGNAL-RESOURCE LEFT (GRASP MOUSE)) {ENABLED}]
[5050-A] EXECUTING [TASK5747 (SIGNAL-RESOURCE LEFT (GRASP MOUSE)) {ENABLED}]
[5050-S] --> LEFT ((GRASP MOUSE)) TASK5739
[5050-C] (TERMINATE [TASK5747 (SIGNAL-RESOURCE LEFT (GRASP MOUSE)) {ENABLED}] SUCCESS)
[5050-A] ENABLING [TASK5737 (SHIFT-GAZE-TO [VOF:{DL603}])] {ENABLED}]
[5050-A] EXECUTING [TASK5737 (SHIFT-GAZE-TO [VOF:{DL603}])] {ENABLED}]
```

Debugging Monitoring Behavior

- What tasks executed and why? Which did not?
- Why didn't the task start when I expected?
- Why did trigger when I didn't expect it to?
- Did the "usual" thing happen or something new?
- etc...

Understanding and Debugging Apex Behavior

Monitoring and querying



```
(:in-order
  (:and (color ?x = red) (:measurement (flashing ?x = true) :estimation (:persists :with-timeout P10s)))
  (:episode (status alarm) :msi P1m :value (:all (eql on)) :timing (:duration (>= P30S)))
  (broken ?z)
  (:measurement (flow-rate ?z = ?f) (:value (> 100))))
```

Apex



Multitask Management

Multitask Management

The term Multitask Management covers a range of capabilities for coordinating potentially interacting tasks, especially those arising from separate procedures.

Examples

- Postpone transition to interrupting task until good stopping point (finished typing sentence)
- Conditionally invoke transition behavior to reduce cost/risk of interruption (pull over car to read map)
- Insert compatible task into unexpected slack time in resource demanding task (put on coat at red light)

Aspects of multitask management addressed in Apex

- Concurrency control
- Rational interruption and resumption
- Robust interleaving
- Efficient use of resources

Concurrency Control

PDL Idioms

Converge

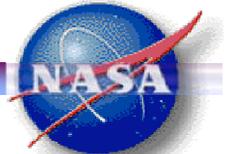
```
(procedure
  (index (do-it))
  (step s1 (do-A))
  (step s2 (do-B))
  (step s3 (do-C))
  (waitfor ?s1 ?s2))
(step s4 (terminate)
  (waitfor ?s3)))
```

Race

```
(procedure
  (index (do-it))
  (step s1 (do-A))
  (step s2 (do-B))
  (step s3 (do-C))
  (waitfor ?s1)
  (waitfor ?s2))
(step s4 (terminate)
  (waitfor ?s3)))
```

Synchronize

```
(procedure
  (index (do-it))
  (step s1 (do-A))
  (step s2 (do-B))
  (waitfor (started ?s1)))
(step s3 (terminate)
  (waitfor ?s1 ?s2)))
```



Rational Interruption and Resumption

Interruptions arise from resource conflicts

- resource requirements from PDL profile clause

```
(profile <resource-name> [tolerance] ...)
```

- interrupting tasks with duration below specified tolerance value not considered in conflict

Interruptions resolved based on priority heuristic

$$priority_b = IC + S\left(1 - \frac{1}{U + 1}\right)I_b + (S_{\max} - S)\left(1 - \frac{1}{I_b + 1}\right)U_b$$

Robust Interleaving

Successful execution-time task interleaving requires managing task transitions – e.g. safing a task for interruption, maintaining its viability while inactive and restoring preconditions to resume

```
(procedure
  (index (fly-cruise-leg using manual-control))
  (step s1 (maintain-altitude)
    (interrupt-cost 5))
    ...
  (step s12 (handoff-to-pilot-not-flying)
    (priority (importance 10) (urgency 10)))
    (waitfor (interrupted ?self)))
  (step s13 (monitor-pilot-not-flying)
    (waitfor (completed ?s12)))
  (step s14 (request-role-pilot-flying)
    (waitfor (resumed ?self)))
    ...)
```

Efficient Use Of Resources

- Combine redundant tasks

```
(merge <condition> [<task-pattern>])
```

- Exploit slack time in use of limited resources
 - Tolerance value specifies degree of “protection” over temporarily idle resources
 - Priority-based on-line scheduler attempts to maximize use of resources



Sherpa

Integrated Debugging and Demonstration Environment

Browser interaction model for viewing data

Switch between 6 ways to view a data object

forward/back buttons URL focal object All data objects as hypertext

Main Toolbar

Object Tree (navigation) Window

Application Status Bar

The screenshot shows the Sherpa software interface. At the top is a menu bar (File, View, Trace, Inspect, Window, Help) and a toolbar with buttons for Back, Forward, Forward, Reset, Run, Step, Trace, Diagram, Inspect, PERT, Agenda, and PDL. Below the toolbar is a 'Focal Object' field. On the left is an 'Object Tree' showing a hierarchical structure of the simulation. The main area is a 'Main View Window' displaying a table of events. At the bottom is an 'Application Status Bar' showing 'State PAUSED' and 'Time 1391'. A 'Communication Status Indicator' is located at the bottom right.

Timestamp	Object	Event Type	Parameters
490	agent	task-started	(task-287 (use resource (memory-1 unspecified) for (50 ms)))
540	agent	task-started	(task-143 (attend-cursor-at-target card-slot))
540	agent	task-started	(task-290 (use resource (memory-1 unspecified) for (50 ms)))
701	agent	task-started	(task-142 (world new-cursor-location card-slot))
701	agent	task-started	(task-293 (use resource (gaze-1 unspecified) for (0 ms)))
701	agent	task-started	(task-141 (perceive-cursor-at-target card-slot))
701	agent	task-started	(task-296 (use resource (vision-1 unspecified) for (100 ms)))
801	agent	task-started	(task-140 (verify-cursor-at-target card-slot))
801	agent	task-started	(task-299 (use resource (memory-1 unspecified) for (50 ms)))
851	agent	task-started	(task-139 (initiate-click card-slot))
851	agent	task-started	(task-302 (use resource (memory-1 unspecified) for (50 ms)))
901	agent	task-started	(task-138 (mouse-down card-slot))
901	agent	task-started	(task-305 (mouse-object))
901	agent	task-started	(task-304 (mouse-down on [card-slot-1 card-slot] with [right-hand-1] taking (100 ms)))
901	agent	task-started	(task-170 (attend-target key-4))
901	agent	task-started	(task-309 (use resource (memory-1 unspecified) for (50 ms)))
951	agent	task-started	(task-169 (initiate-eye-movement key-4))
951	agent	task-started	(task-312 (use resource (memory-1 unspecified) for (50 ms)))
1001	agent	task-started	(task-168 (eye-movement key-4))
1001	agent	task-started	(task-315 (use resource (gaze-1 unspecified) for (50 ms)))
1001	agent	task-started	(task-137 (mouse-up card-slot))
1001	agent	task-started	(task-319 (mouse-object))
1001	agent	task-started	(task-318 (mouse-up on [card-slot-1 card-slot] with [right-hand-1] taking (100 ms)))
1051	agent	task-started	(task-167 (perceive-target-complex key-4))
1051	agent	task-started	(task-322 (use resource (vision-1 unspecified) for (290 ms)))
1101	agent	task-started	(task-172 (initiate-move-cursor key-4))
1101	agent	task-started	(task-326 (use resource (memory-1 unspecified) for (50 ms)))
1151	agent	task-started	(task-171 (move-cursor key-4))
1151	agent	task-started	(task-329 (mouse-time key-4))
1151	agent	task-started	(task-328 (move-mouse to key-4 with [right-hand-1] taking (309 ms)))
1341	agent	task-started	(task-166 (verify-target-position key-4))
1341	agent	task-started	(task-332 (use resource (memory-1 unspecified) for (50 ms)))
1391	agent	task-started	(task-182 (attend-target key-9))
1391	agent	task-started	(task-336 (use resource (memory-1 unspecified) for (50 ms)))

Inspect
Trace
Diagram
PERT (schedule)
Agenda (tree)
PDL (template)

Main View Window

Communication Status Indicator



Sherpa Trace View

The screenshot shows the Sherpa Trace View interface for a simulation titled "ATM-CPM World". The interface includes a menu bar (File, View, Trace, Inspect, Window, Help), a toolbar with navigation and analysis tools (Back, Forward, Reload, Reset, Run, Step, Trace, Diagram, Inspect, PERT, Agenda, PDL), and a main display area. The main display area is divided into a "Focal Object" tree on the left and a large event log table on the right. The event log table displays 89 of 4337 events, with columns for Timestamp, Object, Event Type, and Parameters. The state at the bottom is "PAUSED" at time 1391.

Timestamp	Object	Event Type	Parameters
490	agent	task-started	{task-287 (use resource {memory-1 unspecified} for (50 ms))}
540	agent	task-started	{task-143 (attend-cursor-at-target card-slot)}
540	agent	task-started	{task-290 (use resource {memory-1 unspecified} for (50 ms))}
701	agent	task-started	{task-142 (world new-cursor-location card-slot)}
701	agent	task-started	{task-293 (use resource {gaze-1 unspecified} for (0 ms))}
701	agent	task-started	{task-141 (perceive-cursor-at-target card-slot)}
701	agent	task-started	{task-296 (use resource {vision-1 unspecified} for (100 ms))}
801	agent	task-started	{task-140 (verify-cursor-at-target card-slot)}
801	agent	task-started	{task-299 (use resource {memory-1 unspecified} for (50 ms))}
851	agent	task-started	{task-139 (initiate-click card-slot)}
851	agent	task-started	{task-302 (use resource {memory-1 unspecified} for (50 ms))}
901	agent	task-started	{task-138 (mouse-down card-slot)}
901	agent	task-started	{task-305 (mouse-object)}
901	agent	task-started	{task-304 (mouse-down on {card-slot-1 card-slot} with {right-hand-1} taking (100 ms))}
901	agent	task-started	{task-170 (attend-target key-4)}
901	agent	task-started	{task-309 (use resource {memory-1 unspecified} for (50 ms))}
951	agent	task-started	{task-169 (initiate-eye-movement key-4)}
951	agent	task-started	{task-312 (use resource {memory-1 unspecified} for (50 ms))}
1001	agent	task-started	{task-168 (eye-movement key-4)}
1001	agent	task-started	{task-315 (use resource {gaze-1 unspecified} for (50 ms))}
1001	agent	task-started	{task-137 (mouse-up card-slot)}
1001	agent	task-started	{task-319 (mouse-object)}
1001	agent	task-started	{task-318 (mouse-up on {card-slot-1 card-slot} with {right-hand-1} taking (100 ms))}
1051	agent	task-started	{task-167 (perceive-target-complex-r key-4)}
1051	agent	task-started	{task-322 (use resource {vision-1 unspecified} for (290 ms))}
1101	agent	task-started	{task-172 (initiate-move-cursor key-4)}
1101	agent	task-started	{task-326 (use resource {memory-1 unspecified} for (50 ms))}
1151	agent	task-started	{task-171 (move-cursor key-4)}
1151	agent	task-started	{task-329 (mouse-time key-4)}
1151	agent	task-started	{task-328 (move-mouse to key-4 with {right-hand-1} taking (309 ms))}
1341	agent	task-started	{task-166 (verify-target-position key-4)}
1341	agent	task-started	{task-332 (use resource {memory-1 unspecified} for (50 ms))}
1391	agent	task-started	{task-182 (attend-target key-9)}
1391	agent	task-started	{task-336 (use resource {memory-1 unspecified} for (50 ms))}

Important Features

- Interactive filtering by event type, time, object
- Preset show levels
- Runtime filter control
- Adv. pause/step control
- Find/highlight/filter controls
- Table controls such as column sorting
- Easy output to analysis tools such as Excel

Sherpa Agenda View

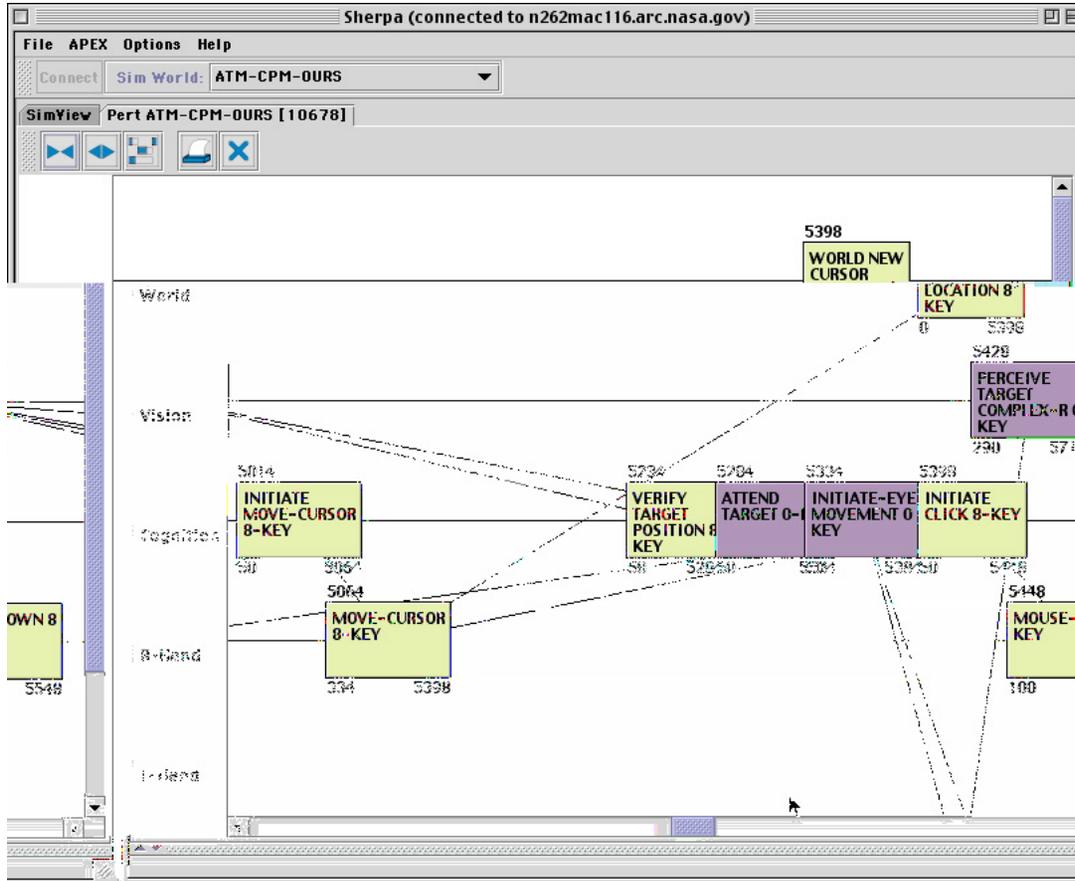
The screenshot shows the Sherpa Agenda View interface for an ATM-CPM World simulation. The main window displays a list of tasks and their states. The interface includes a menu bar (File, View, Trace, Inspect, Window, Help), a toolbar with navigation buttons (Back, Forward, Reload, Reset, Run, Step, Trace, Diagram, Inspect, PERT, Agenda, PDL), and a 'Focal Object' field. A tree view on the left shows the simulation hierarchy. The main table has the following columns: Task, Task State, Monitors, Resources, and Enabled Time.

Task	Task State	Monitors	Resources	Enabled Time
(do-domain)	ongoing	nil	nil	0
(do banking)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-2 (terminated ...	nil	0
(end session)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-5 (terminated ...	nil	0
(retrieve receipt)	ongoing	nil	nil	0
(slow-move-click money-slot)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-27 (terminate...)	nil	0
(enter-no)	ongoing	nil	nil	0
(slow-move-click no-key)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-29 (terminate...)	nil	0
(retrieve card)	ongoing	nil	nil	0
(slow-move-click card-slot)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-28 (terminate...)	nil	0
(initiate session)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-12 (terminate...)	nil	0
(insert card)	terminated	nil	nil	0
(enter password)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-14 (terminate...)	nil	0
(enter-ok)	ongoing	nil	nil	0
(enter-number key-1)	ongoing	nil	nil	0
(enter-number key-0)	ongoing	nil	nil	0
(enter-number key-9)	ongoing	nil	nil	0
(remember pin)	terminated	nil	nil	150
(enter-number key-4)	ongoing	nil	nil	0
(do transaction)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-8 (terminated ...)	nil	0
(retrieve money)	ongoing	nil	nil	0
(slow-move-click money-slot)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-21 (terminate...)	nil	0
(enter amount)	ongoing	nil	nil	0
(terminate ?self success >> nil)	pending	((atomic-episode-monitor-22 (terminate...)	nil	0
(enter-correct)	ongoing	nil	nil	0
(enter-number key-9)	ongoing	nil	nil	0
(enter-number key-0)	ongoing	nil	nil	0
(choose withdraw)	ongoing	nil	nil	0

Important Features

- Click access to other task views: PERT, inspect, procedure, monitors
- Searchable
- Time data provides points of reference in trace

Sherpa PERT view



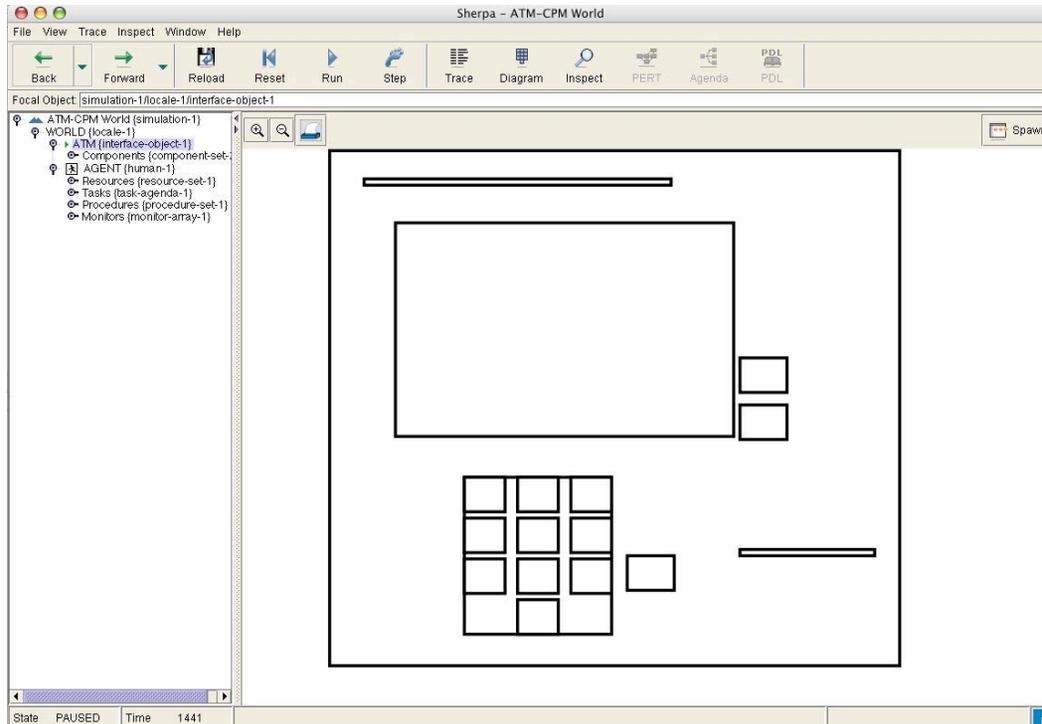
Shows runtime micro-scheduling

Important Features

- Switch to Gantt chart
- View controls incl. zoom
- Computes critical path
- Color code for common parent
- Shows inherited dependencies
- Effective printing to b/w printer
- Output to Powerpoint



Sherpa Diagram View (Vector)

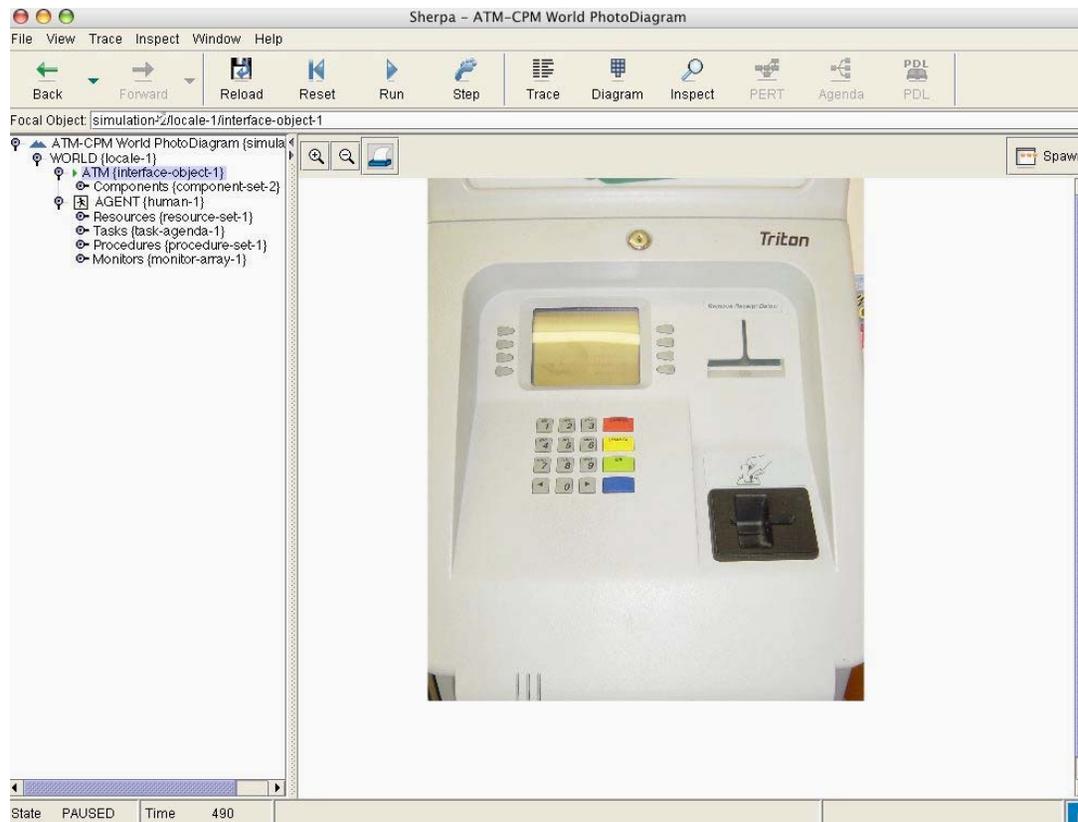


Allows user to view spatial data

- layout
- plotted movement data
(not yet implemented)

Important Features

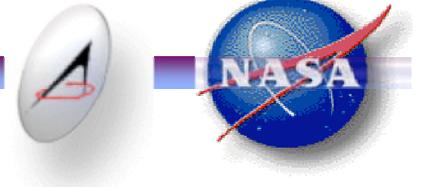
- Click access to other object views: inspect, trace, diagram
- Printer support



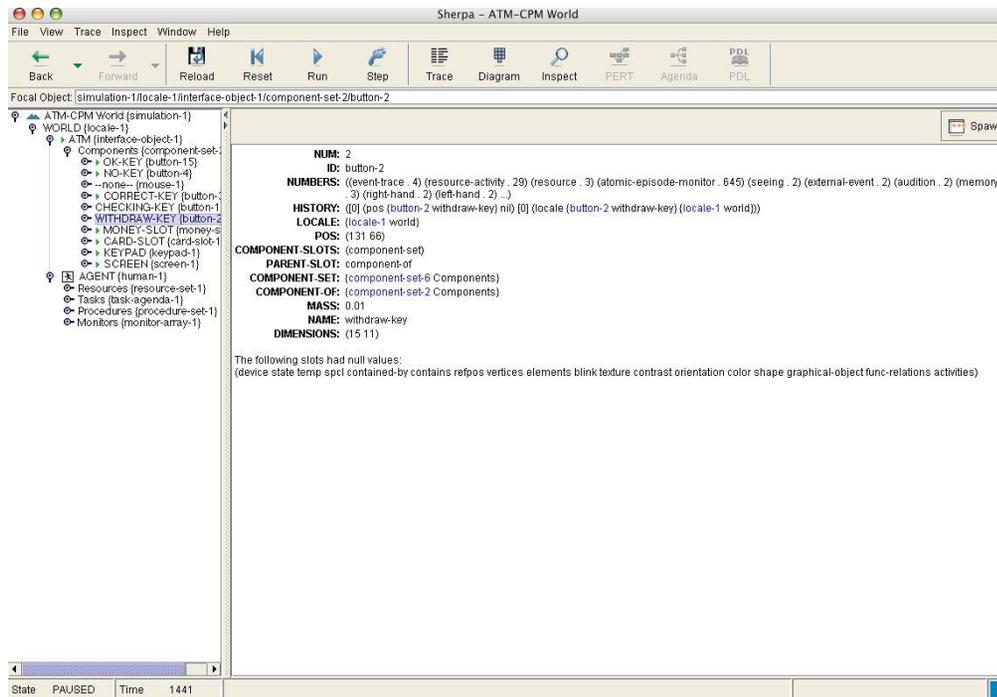
Facilitates demonstration

Facilitates exploration of application model

Provides way to identify part names, secondary navigation



Sherpa Inspect View



For debugging, easy access to internal representation of all objects

Important Features

- Click access to any other view of focal object
- Various controls over how much information is shown

Apex Applications

1. Army/NASA Autonomous Rotorcraft Project
2. Autonomous Robust Avionics (fixed-wing UAV)
3. Mission Simulation Facility (rover requirements elicitation)
4. Virtual Airspace Modeling and Simulation
5. MIDAS human crewstation analysis
6. HCI performance analysis using CPM-GOMS
7. Astronaut behavior modeling
8. Simulation testbed: Autonomous Underwater Robot
9. Simulation testbed: X-Plane
10. Other: educational outreach and external user support

Army/NASA Autonomous Rotorcraft Project

OBJECTIVE: versatile, practical and inexpensive airborne observation platform effective for a broad range of missions





The Surveillance Problem

Example Scenario



← Area of operations

Valuable Assets

- docks
- warehouses
- lighthouse
- orchard tract

Risk: any asset can start on fire at any time

UAV Goal: do a good job detecting fires and mitigating losses

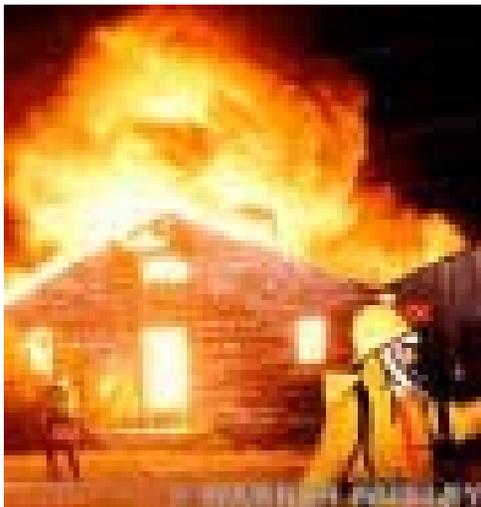
What does it mean to do a good job at surveillance in this kind of scenario?



The Surveillance Problem

Goal: minimize expected cost of ignorance over mission

Example



Probability of occurrence (pdf)

$$p(t) = ae^{-at}$$

← exponential

Cost if it occurs

$$\text{cost}(d) = c_0 + \left(\frac{2}{1 + e^{-k(d+l_1+l_2)}} - 1 \right) (m - c_0)$$

← sigmoid

Expected cost of ignorance $[t_1 \ t_2]$

$$\text{ECI}_-(t_1, t_2, a, k, m) = \int_{t_1}^{t_2} ae^{-at} m \left(\frac{2}{1 + e^{-k(t_2-t)}} - 1 \right) dt$$

minimize this

→

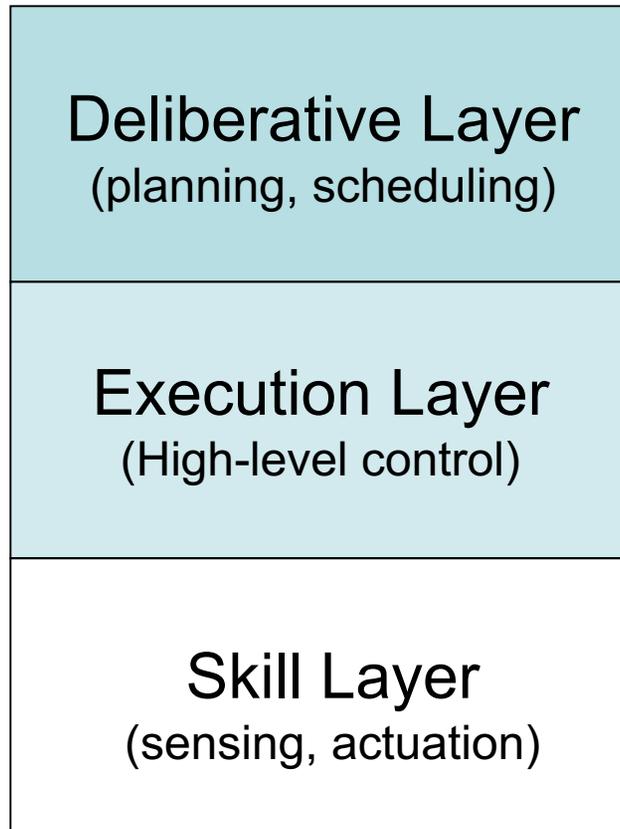
$$\text{Mission-ECI} = \sum_{\text{targets}} \sum_{\text{intervals}} \text{ECI}_-(t_{i-1}, t_i)$$



Autonomous Rotorcraft Project

Autonomy Software Architecture

Apex



3-Tier Agent Architecture

— Surveillance scheduling
(multitask management)

{ Tactical sensor positioning
Human interaction management
Monitoring and anomaly-handling
Obstacle avoidance path planning
Flight patterns

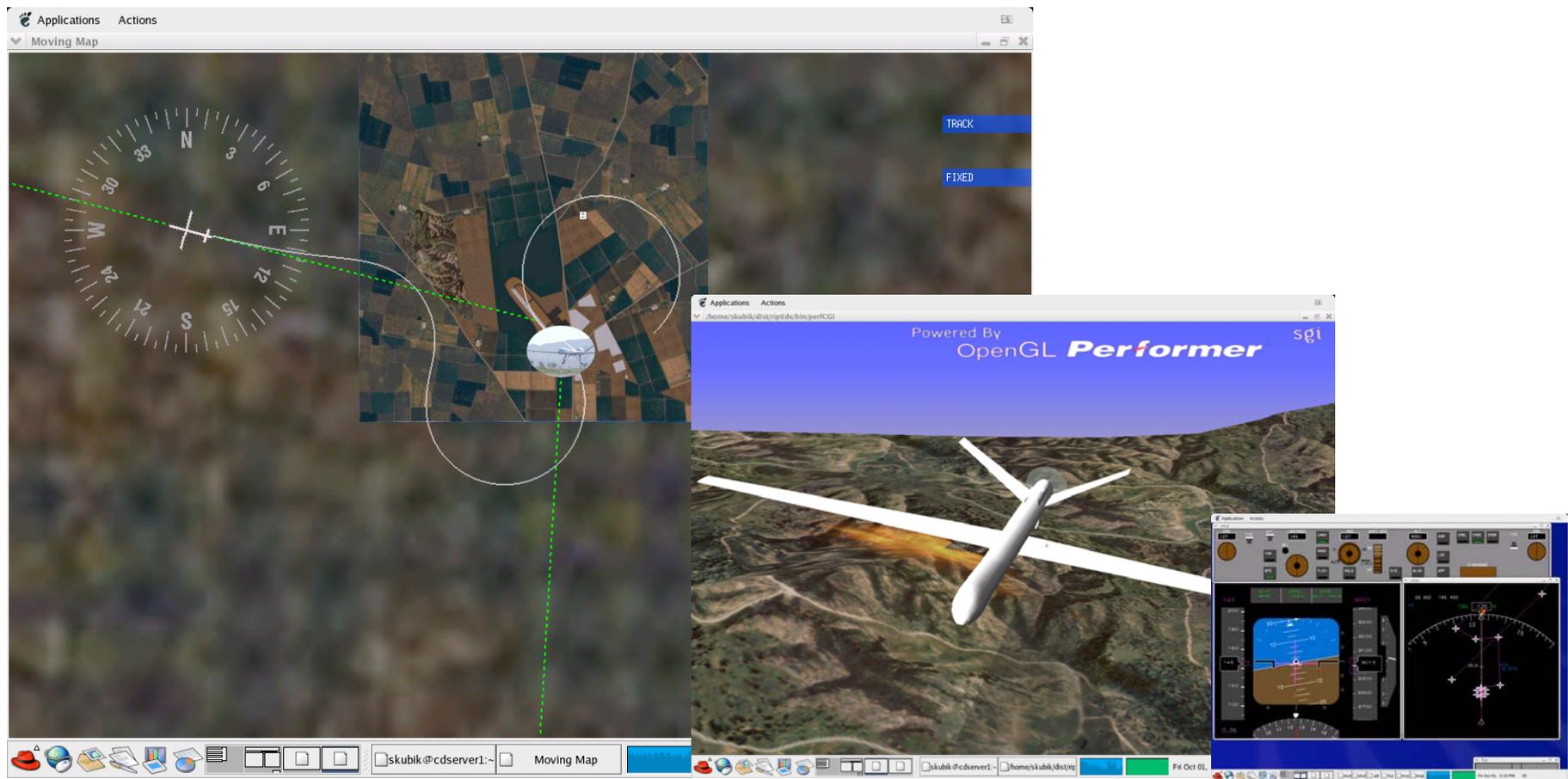
— Flight controls



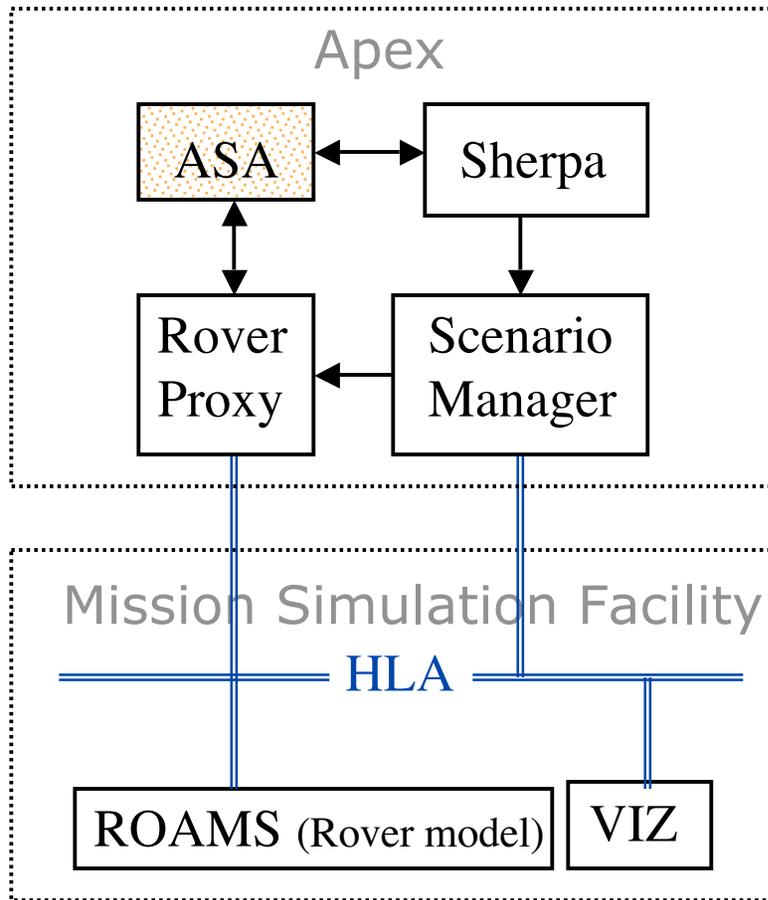
Autonomous Robust Avionics

Initial mission (complete): simulated wildfire detection and investigation

Future mission: fly real Predator B in planetary science analog mission



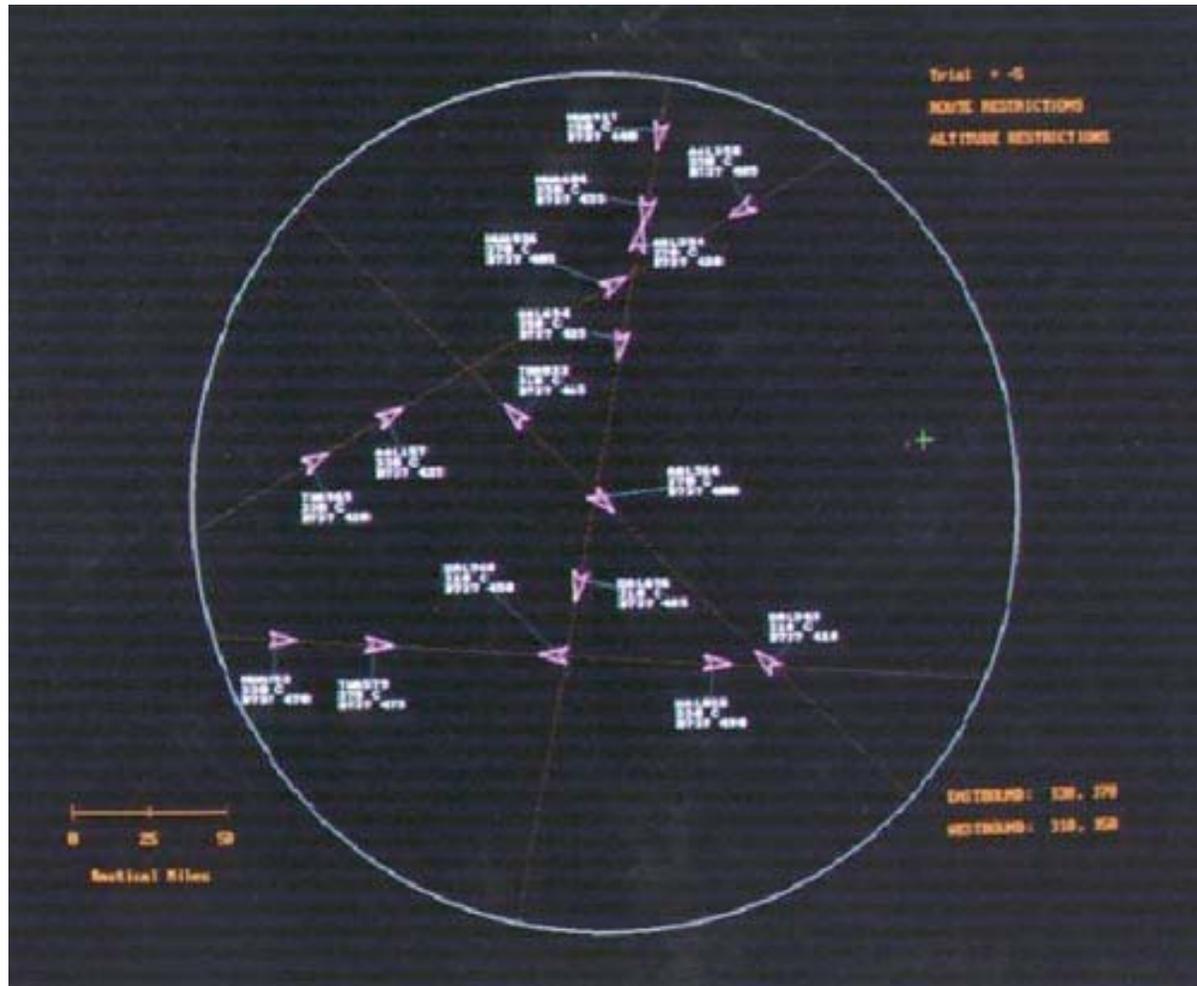
Requirements Elicitation Facility (REF)



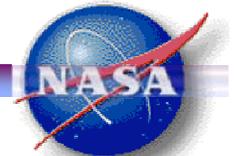
Michael Freed (PI)
 Mark Drummond
 David Stavens

Virtual Airspace Modeling and Simulation

Pseudo Air Traffic Controllers for large engineering sims



Roger Remington (PI)
 Seungman Lee
 Ujwala Ravinder

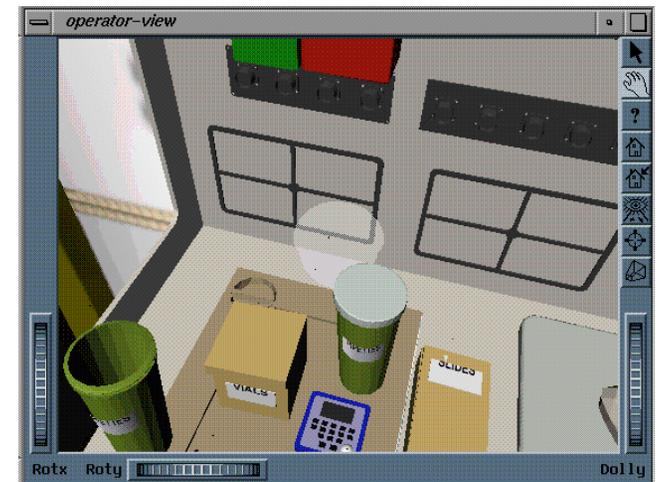
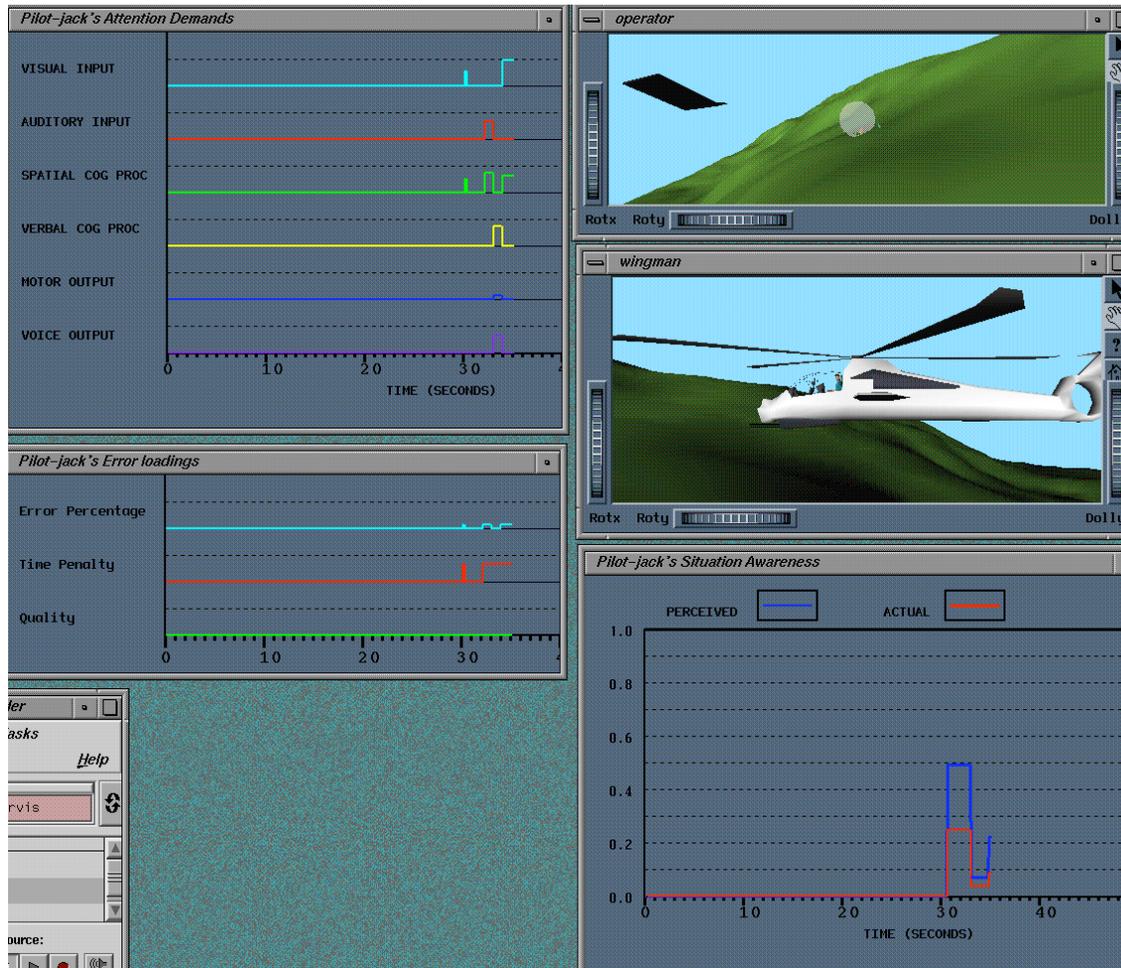


MIDAS



Simulation/Analysis of Human-Crewstation Performance

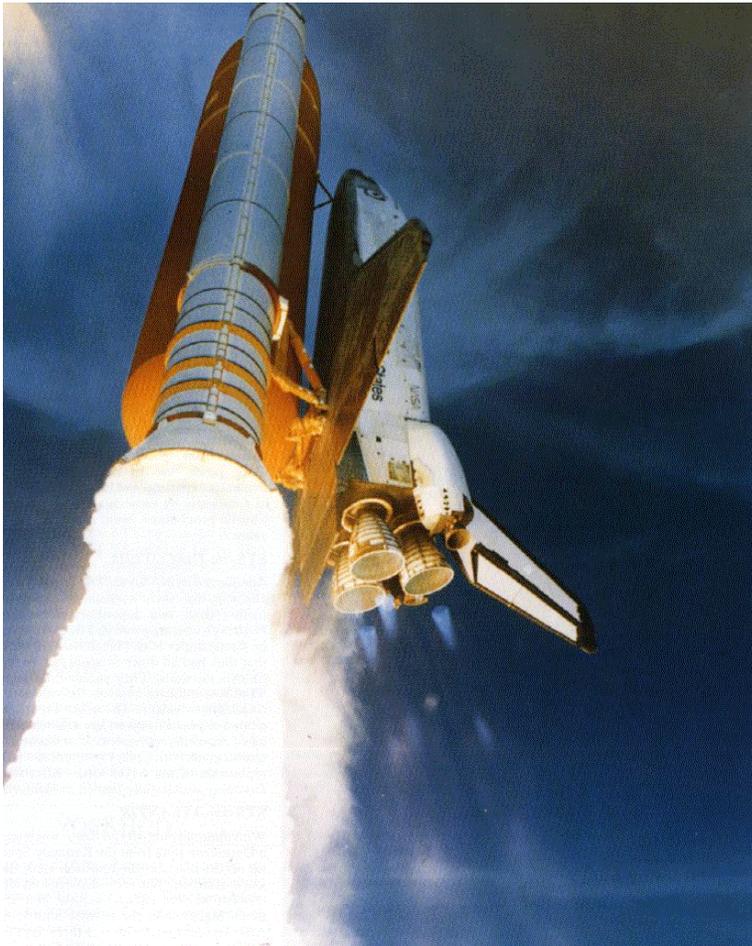
Sandy Hart (PI)
Peter Jarvis
Michael Dalal





Astronaut behavior modeling

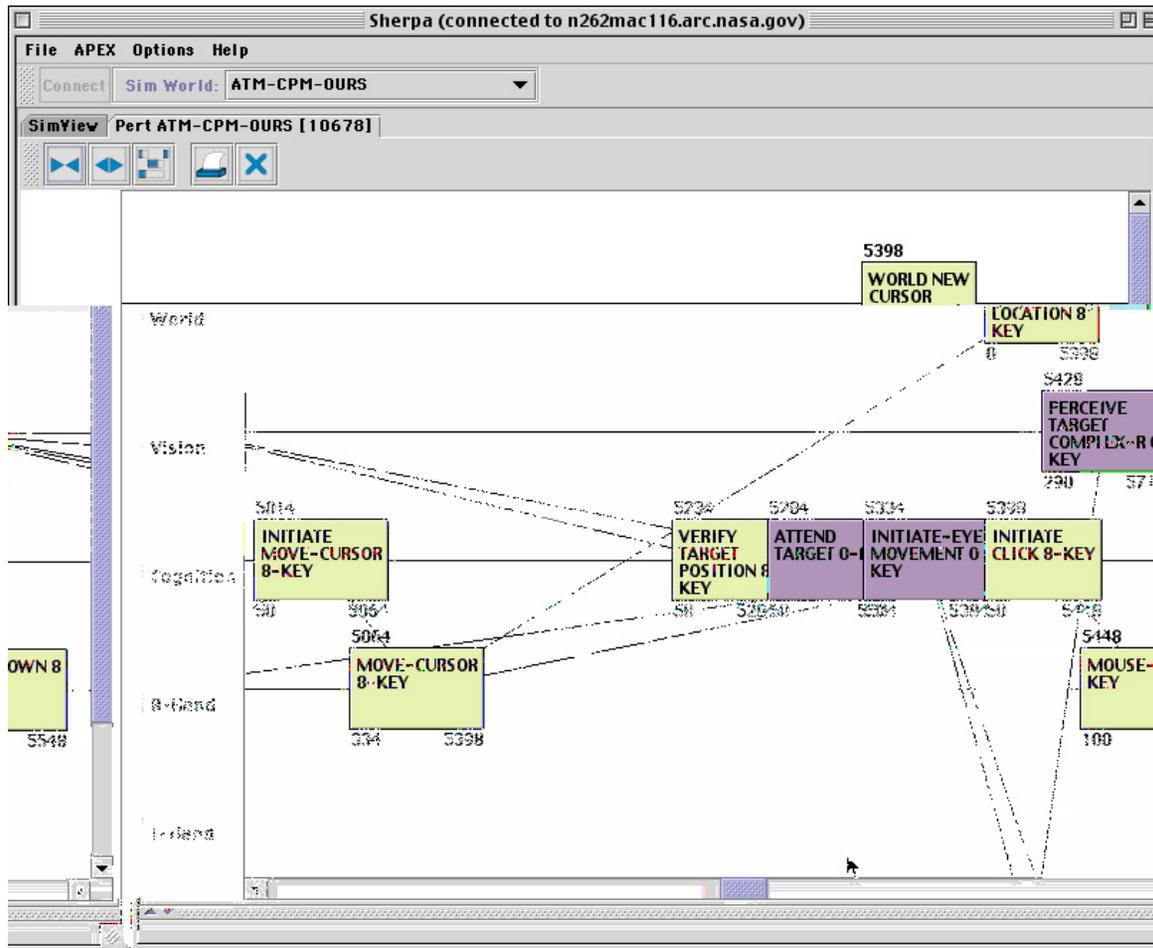
Space Shuttle Ascent Procedure



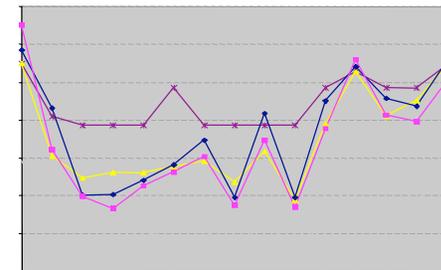
Rob McAnn (PI)
Michael Matessa

HCI Performance Analysis

predicting human task durations with CPM-GOMS



Michael Freed
 Bonnie John
 Michael Matessa
 Roger Remington
 Alonso Vera



Autonomous Underwater Robot

Simulation Testbed

Pete Bonasso (PI)



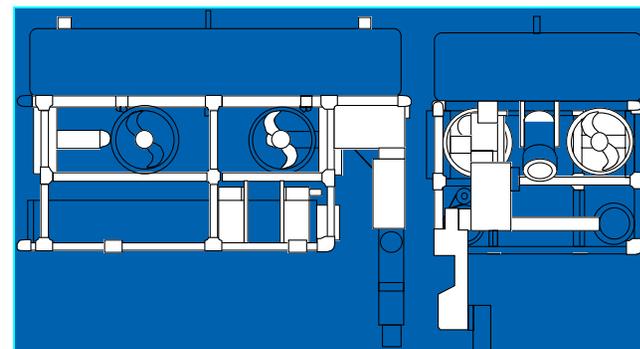
Mission: Long duration monitoring of deep ocean phenomena, i.e., hydrothermal vents.

Tasks

Vent detection via salinity/currents
Thermal, biological, effluent sampling
Data up-linking
IVHM
Power management

Equipment

- 6 DoF thrusters, gyro-stabilizers, IMUs
- Sonar-based obstacle detection system
- 6 DoF manipulators/w force-torque wrist
- 1 MHz laser sighted, pencil-beam sonar
- Stereo color cameras
- Thermal, salinity, current probes



X-Plane

Simulation Testbed

Robert Harris
Michael Freed



Mission: nominal flight in U.S. airspace in varied conditions, with diverse aircraft and flight goals.

Research emphasis: identifying, isolating and recovering from failure. X-Plane supports > 30 aircraft failure modes.

- 100s of aircraft models and airport
- rich geography and airspace model



Educational Outreach and Support for External Users

Apex used in classrooms to teach autonomy or HCI analysis (>100 students)

- University of Maryland (desJardins)
- Stanford (Freed)
- CMU (John)
- Hong Kong U (Vera)
- George Mason (Boehm-Davis)

Apex used for research or applied projects

- CMU (Sycara, John)
- Stanford (Peters)
- Dynamic Research Inc. (Sauer)

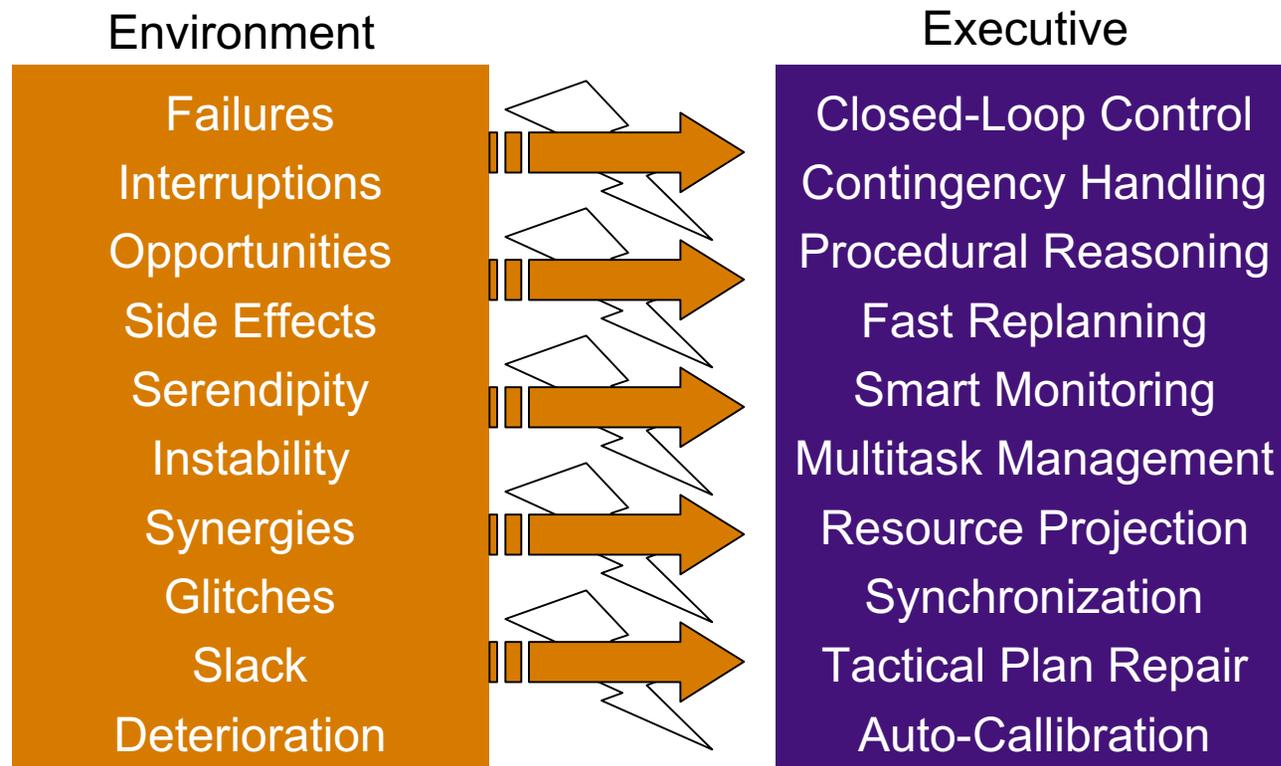
Apex



Extra

Smart Executive Functionality

Robust execution mechanisms should achieve goals and maintain safety despite changing conditions, unexpected outcomes, time-pressure and other factors that undermine planned and routine behavior.





Procedure-based Control

Procedure-based control: Procedural execution methods make use of a human-fabricated library of stored plans, refining, composing and adapting them on the fly to support coordinate pursuit of multiple goals. Important features of procedural approaches include scalability to complex, time-pressured control problems and the ease with which they are developed and explained, an important factor in generating stakeholder trust in autonomous systems.

Procedural methods are particularly appropriate for systems operating in proximity to humans where predictability and adherence to standard operating procedures is especially important. Key research emphases include integrated development environments and simulation-based validation.

Procedures are represented in a language that should be compatible with the output of broad class of planners, but should also include features not supported by any planner (to be used in hand-crafted plans, esp. for control)

Hardware - Yamaha RMAX

- 184 lb GW, 65 lb payload
- 3 m rotor diameter
- One hour endurance
- \$86,000

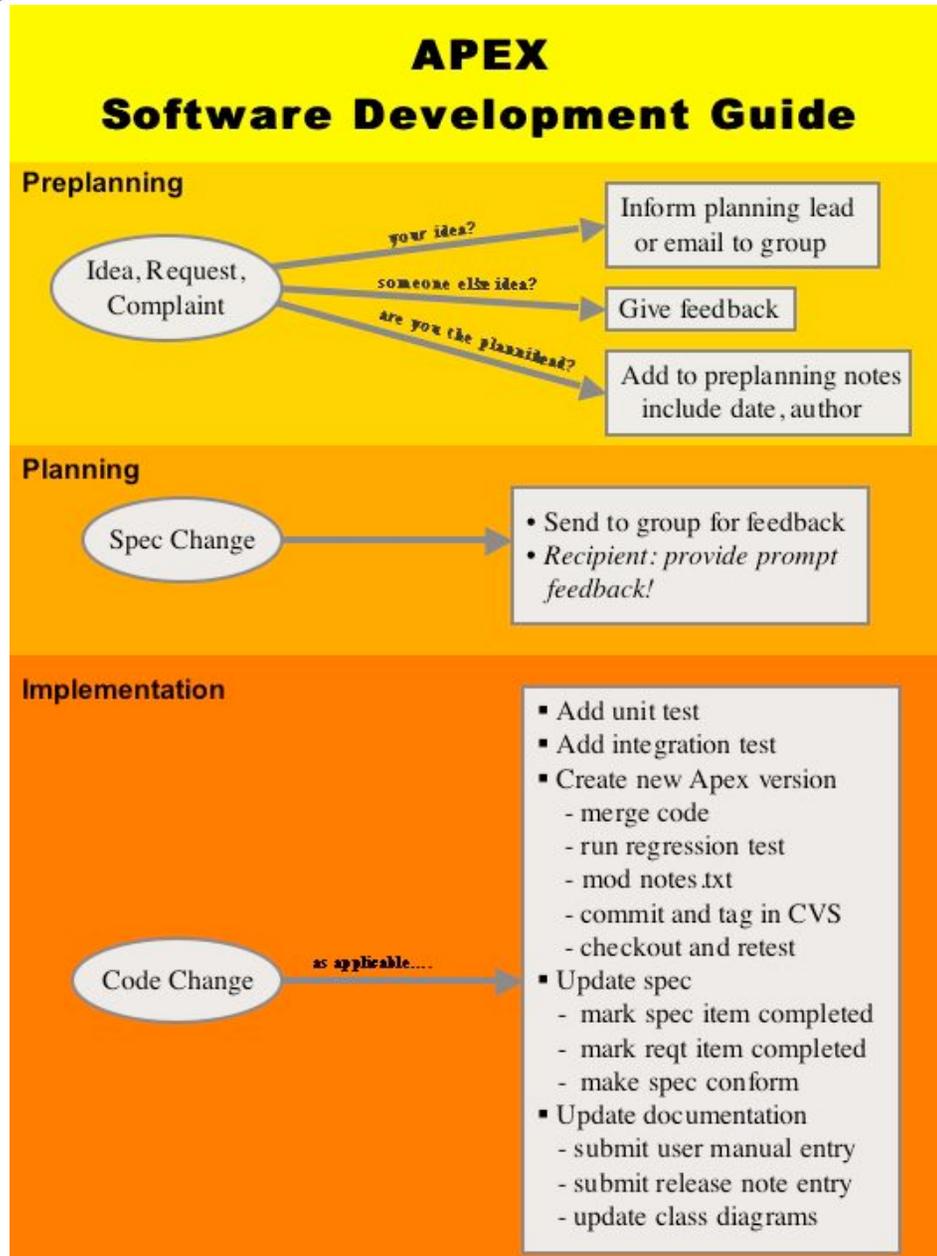


Avionics Payload

- Crossbow IMU
- Radio modem
- PC104+ flight computer
- PCI video computer
- Sonar
- Differential GPS
- Vibration Sensors
- Weight-on-wheels sensors

Vibration-isolated stub wing

- Stereo pair mono cams
- Actuated color camera
- Actuated video camera



Development Process

- Periodic new releases
 - current distributed version: 2.4
 - upcoming: 3.0
- Module leads handle preplanning
- PI leads planning
- Customer input central at all stages
- Testing/doc phase precedes each release

Apex Publications (through 2002)

Freed, M. and Dahlman, E. (2002) Requirements for inferring the intentions of multitasking agents. In *Working Notes of the AAAI Fall Symposium on Intent Inference*, Falmouth, Massachusetts.

John, B. E., Vera, A. H., Matessa, M., Freed, M., and Remington, R. (2002) Automating CPM-GOMS. In *Proceedings of CHI'02: Conference on Human Factors in Computing Systems*. ACM, New York, pp. 147-154.

Freed, M. and Vera, A. (2001) Simulating Human Agents. *AI Magazine*, 22(3), p. 115.

Freed, M. (2000) Reactive Prioritization. *Proceedings 2nd NASA International Workshop on Planning and Scheduling for Space*. San Francisco, CA.

Freed, M. and Remington, R. (2000) Making Human-Machine System Simulation a Practical Engineering Tool: An APEX Overview. In *Proceedings of the 2000 International Conference on Cognitive Modeling*. Groningen, Holland.

Freed, Michael and Remington, R. (2000) GOMS, GOMS+ and PDL. In *Working Notes of the AAAI Fall Symposium on Simulating Human Agents*. Falmouth, Massachusetts.

Freed, Michael. (2000) Heuristic management of execution-time task conflicts. In *Proceedings of the 2000 World Automation Congress*. Maui, HI.

Freed, M., Bear, T., Goldman, H., Hyatt, G., Reber, P., Sylvan, A. and Tauber, J. (2000) Towards More Human-Like Computer Opponents. In *Working Notes of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment*, 22-26

Freed, M. (1998) Managing multiple tasks in complex, dynamic environments. In *Proceedings of the 1998 National Conference on Artificial Intelligence*. Madison, Wisconsin.

Freed, M. and Remington, R. (1998) A conceptual framework for predicting errors in complex human-machine environments. In *Proceedings of the 1998 Meeting of The Cognitive Science Society*. Madison, Wisconsin.

Freed, M., Shafto, M., and Remington, R. (1998) Using simulation to evaluate designs: The APEX approach. In Chatty, S. and Dewan, P., editors, *Engineering for Human-Computer Interaction*, chapter 12. Kluwer Academic

Freed, M. and Remington, R. (1997) Managing decision resources in plan execution. *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*. Nagoya, Japan.

Freed, M. and Shafto, M. (1997) Human System Modeling: Some Principles and a Pragmatic Approach. *Proceedings of the 4th International Workshop on the Design, Specification, and Verification of Interactive Systems*. Granada, Spain.

Van Selst, M. and Freed, M. (1997) Using APEX to model anticipated human error: analysis of a GPS navigational aid. *Proceedings of the Seventh International Conference on Human-Computer Interaction*. San Francisco, California.

Freed, M. (1996) Using the RAP system to simulate human error. *Proceedings of the 1996 AAAI Fall Symposium on Plan Execution: Problems and Issues*, pp.52-58, Cambridge, MA.

Freed, M. and Johnston, J. (1995) Simulating Cognition in the Domain of Air Traffic Control. *Proceedings of 1995 Spring Symposium on Representing Mental States and Mechanisms* (eds. Cox, M. and Freed, M.), Palo Alto, CA.