Integrating Linear Temporal Logic and a Parameterized Action Representation
&
Creating 3D Animated Human Behaviors for Virtual Worlds

Jan M. Allbeck
George Mason University

Norman I. Badler
Center for Human Modeling and Simulation
University of Pennsylvania

Penn Engineering

GEORGE MASON UNIVERSITY

PAR Review

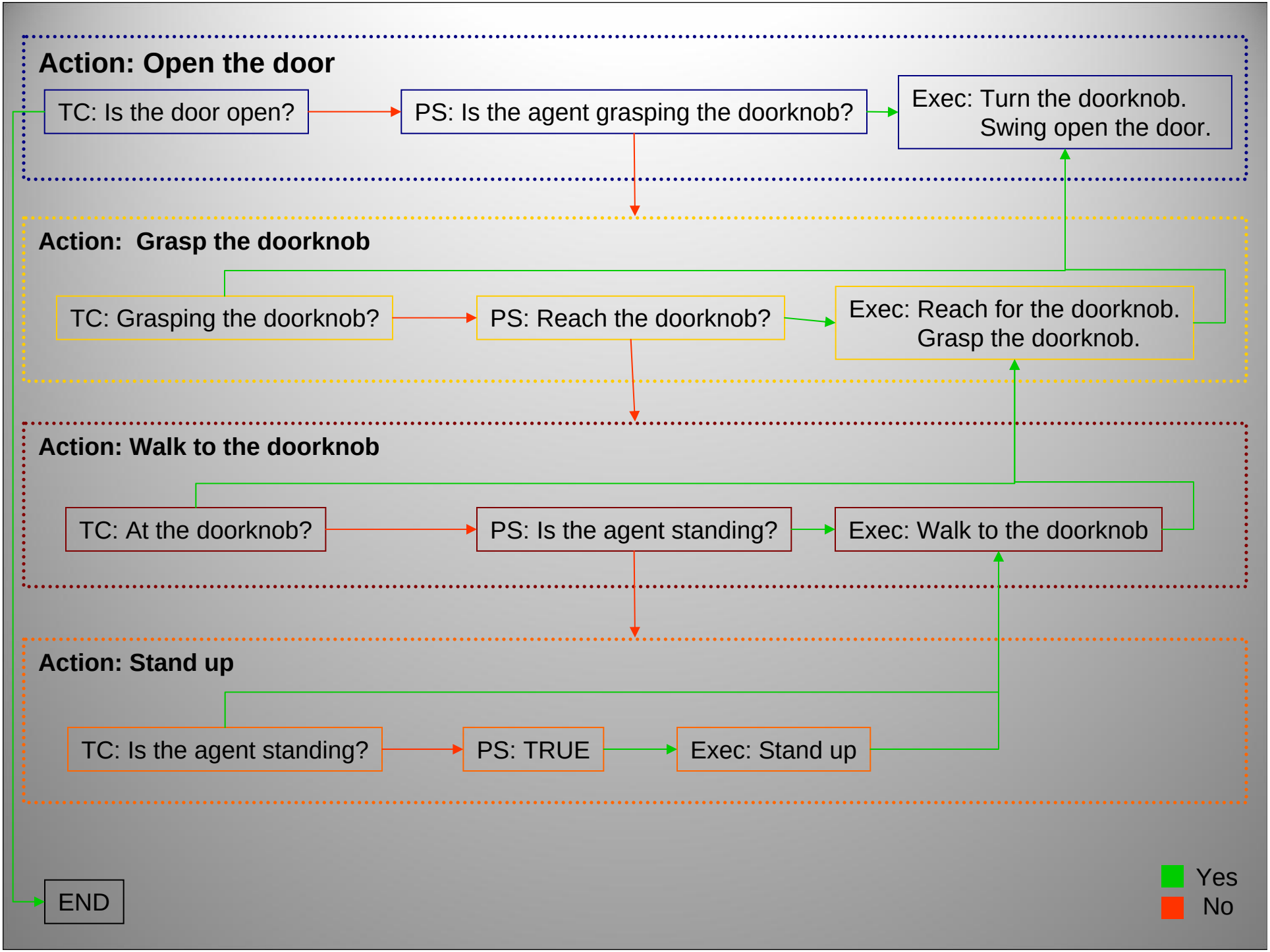
Parameterized Action Representation

- Action and Object representations
- Ontology for simple and complex physical behaviors.
- Natural language and animation intermediary
- Applications: VET, ATOV
- Stored in Hierarchies
- Uninstantiated and instantiated

PAR Actions

- *core semantics:* motion, force, state-change, paths
- *participants:* agent, objects
- *purpose:* state to achieve, action to generate, etc.
- *manner:* how to perform action (e.g. “carefully”)
- *type:* aleatoric, reactive, opportunistic

- *duration:* timing, iteration, or extent; e.g., “for 6 seconds”, “between 5 and 6 times”
- *sub-steps:* actions to perform to accomplish action (includes parallel constructs)
- *next-step:* next action to be performed
- *super-step:* parent action
- *conditions: prior, post*


Action: Open the door
TC: Is the door open?
PS: Is the agent grasping the doorknob?
Exec: Turn the doorknob.
Swing open the door.
Action: Grasp the doorknob
TC: Grasping the doorknob?
PS: Reach the doorknob?
Exec: Reach for the doorknob.
Grasp the doorknob.
Action: Walk to the doorknob
TC: At the doorknob?
PS: Is the agent standing?
Exec: Walk to the doorknob
Action: Stand up
TC: Is the agent standing?
PS: TRUE
Exec: Stand up
END
Yes
No

Object Representation

type object representation =

(name:	STRING;
is agent:	BOOLEAN;
properties:	sequence property-specification;
status:	status-specification;
posture:	posture-specification;
location:	object representation;
contents:	*sequence* object representation;
capabilities:	*sequence* parameterized action;
relative directions:	*sequence* relative-direction-specification;
special directions:	*sequence* special-direction-specification;
sites:	*sequence* site-type-specification;
bounding volume	bounding-volume-specification;
coordinate system	site;
position:	vector;
velocity:	vector;
acceleration:	vector;
orientation:	vector;
data:	ANY-TYPE).

World Model

Information in Effective Instructions

- Core action semantics (e.g. “remove”)
- Action/sub-action structure
- Participants (agent, objects)
- Path, manner, purpose information (“context”)
- Initiation conditions (applicability | preconditions)
- Termination conditions (success or failure cases)

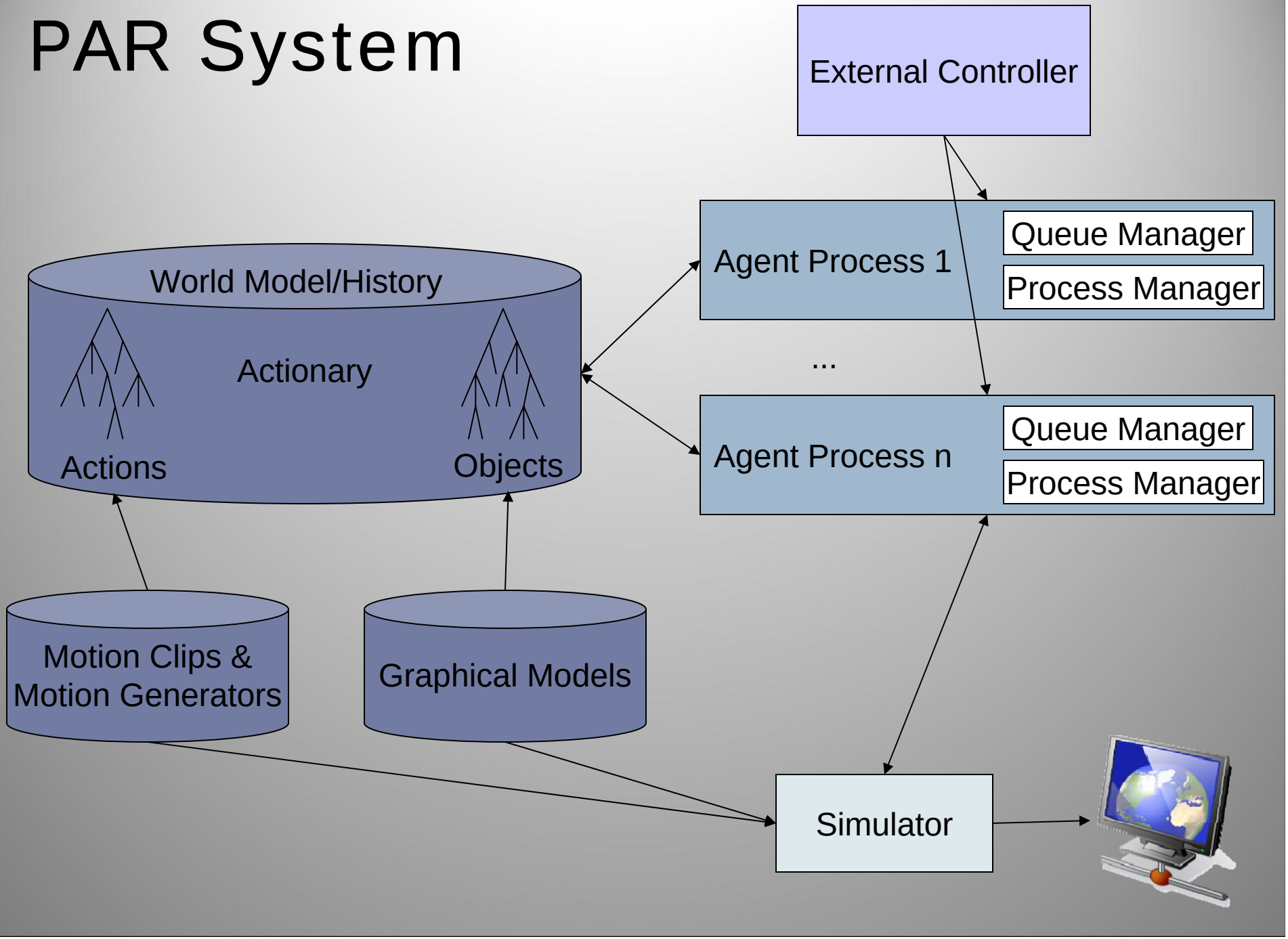
NL: Murray, pickup bomb quickly

PAR: Agent: *Murray* Action: *PickUp*

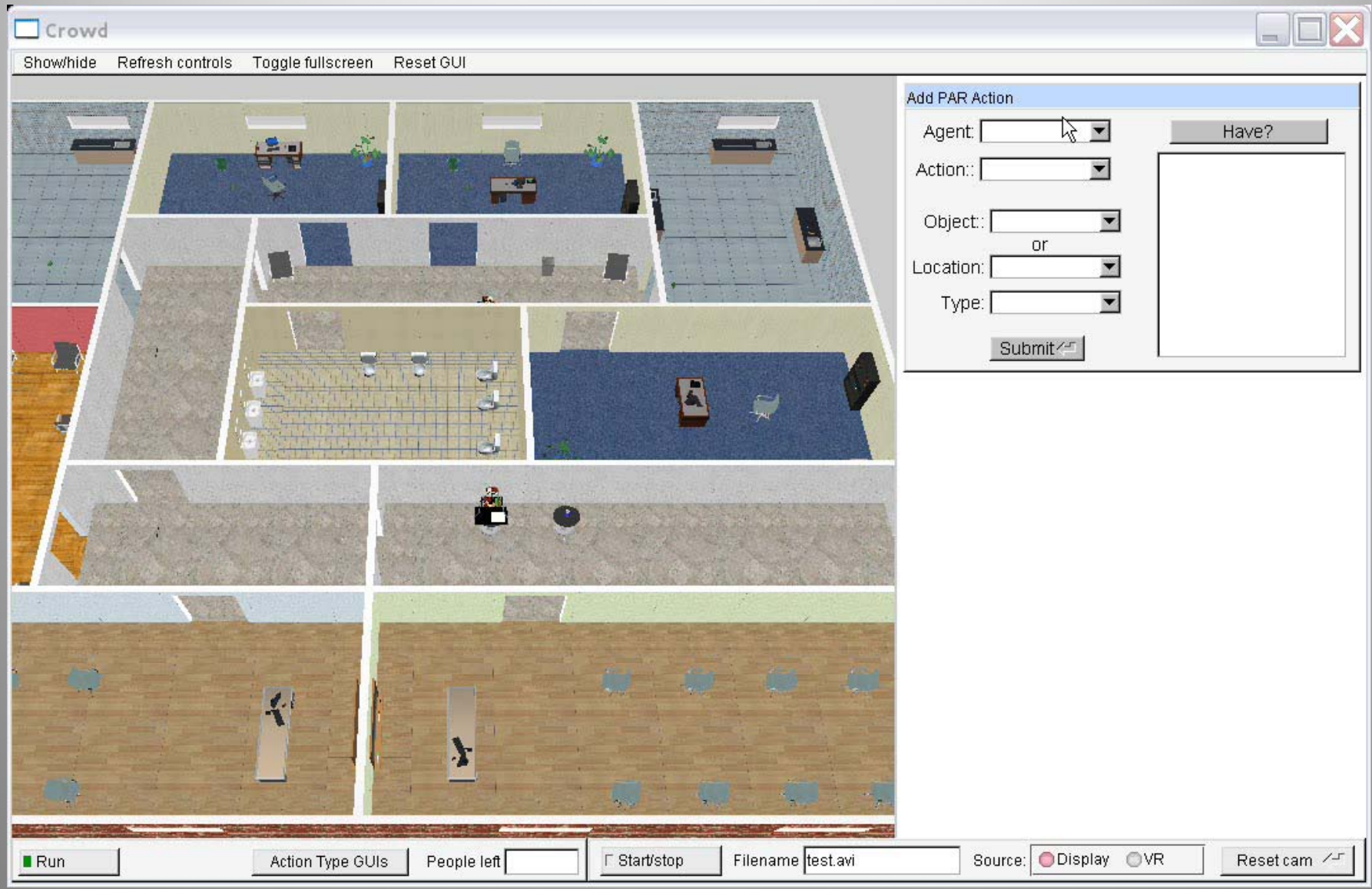
Object: *Bomb* Manner: *quickly*

Animation:

PAR System

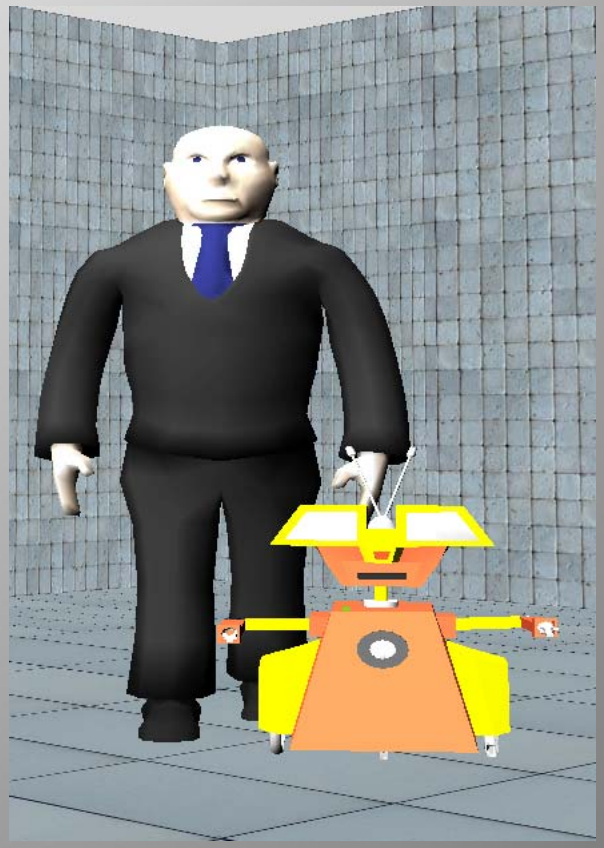

External Controller
Agent Process 1
Queue Manager
Process Manager
...
Agent Process n
Queue Manager
Process Manager
World Model/History
Actionary
Actions
Objects
Motion Clips &
Motion Generators
Graphical Models
Simulator

Murray Interactive Demo



PAR Summary

- Data driven
- Includes a world model
- Provides context
- Captures semantics
- Links to other software systems
- Levels of detail
- Reusable
- Compositional
- Memory or history in the DB

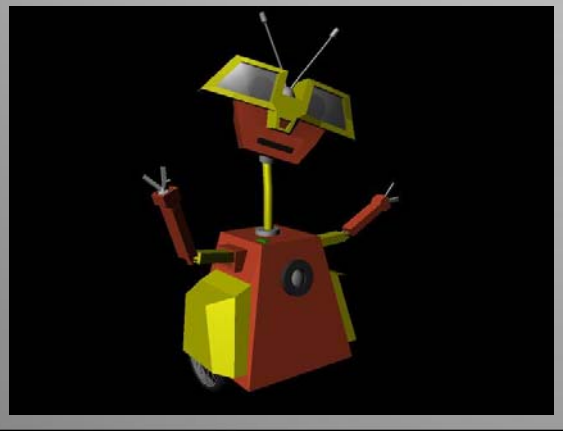


LTL and PAR Integration

- Semantics & Pragmatics->**LTL & PAR**
- Commands in the form of LTL expressions and PAR predicates will be instantiated from pragmatics.
- LTL automatically and verifiably composes controllers that satisfy high level task specifications.
- PAR can be used to fill in parameters of the actions and for simulation.
- Additionally, PAR provides LTL with precepts of the environment that produce state transitions in the LTL automaton and grounds terms.

Example Mission

- Murray starts in room 11.
- “Search rooms 1,2,3 and 4. If you see a dead body, abandon the search and go to room 11. If you see a bomb, pick it up and take it to room 13 and then resume the search.”



Integration

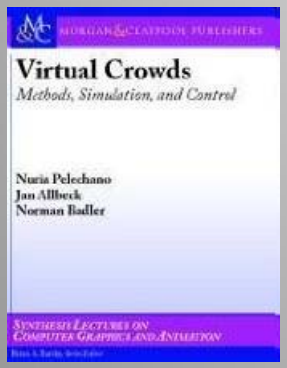
- LTL predicates are linked to PAR objects, actions, parameters, and predicates (*e.g.* spatial predicates).
  - •Bombs •Bodies •Rooms
  - •Pickup •Drop •Walk
- The PAR system
  - Loads the LTL automata
  - Perceives the world and steps through the automaton accordingly
  - Simulates behaviors

Predicates

- Types: weapon, chair, robot, person
- Spatial relations: at the end of the hall, on the desk, in the room (QUD)
- Properties: color, size
- Postures: open, standing
- States: on, idle, broken, armed
- Time and history?

Creating 3D Animated Human Behaviors for Virtual Worlds

(a.k.a. Places Everyone)



Jan M. Allbeck

Advisor: Norman I. Badler

Goal

- *Functional teams* that are easier to create and modify.
- Using roles to specify behaviors.
- Using PAR to add semantics of actions, objects, and agents.
- Implementing four types of actions: scheduled, reactive, opportunistic, and aleatoric.

CAROSA Framework

- Crowds with
- Aleatoric
- Reactive
- Opportunistic and
- Scheduled
- Actions

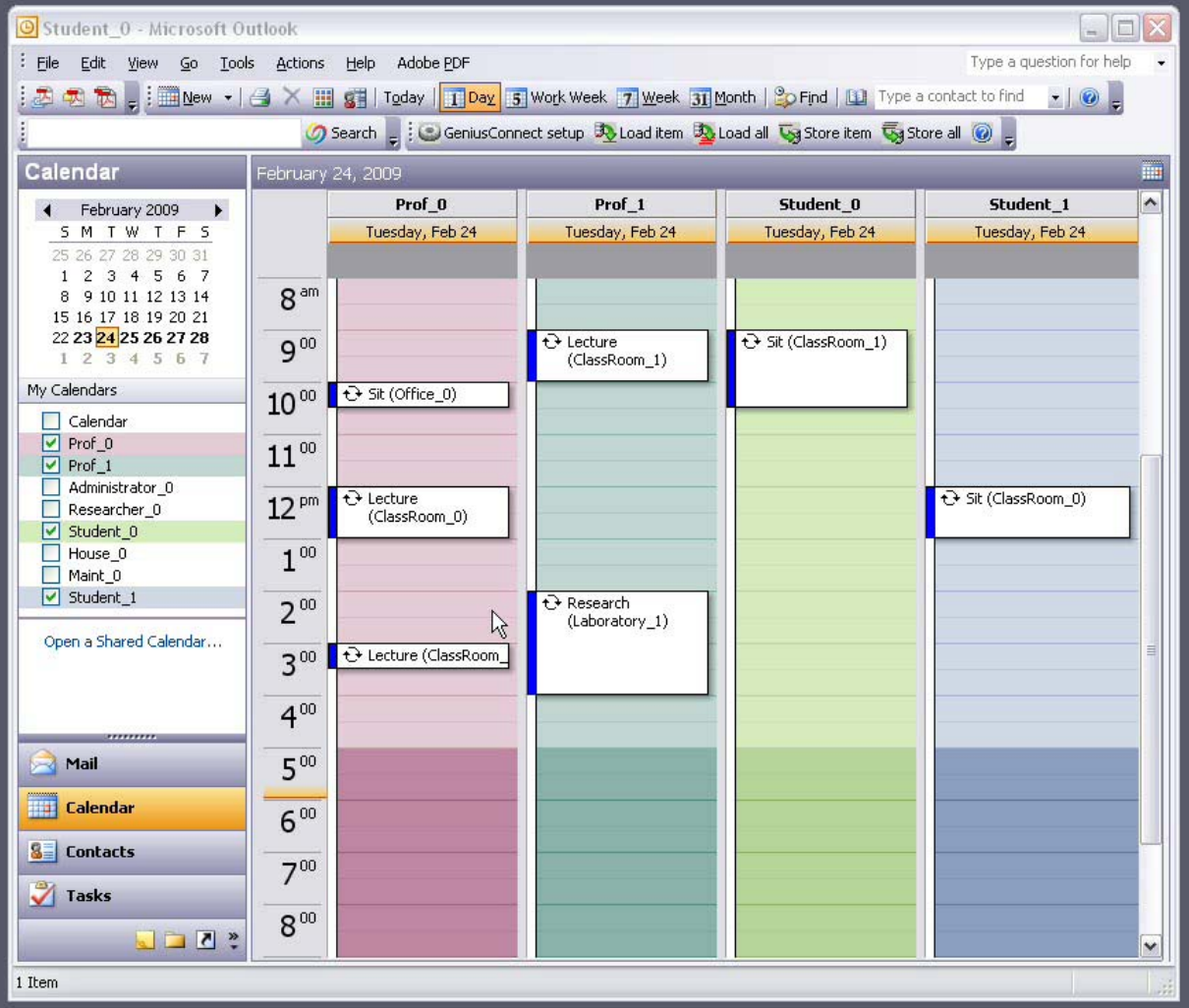
Action Types

- **Scheduled**: arise from specified roles for individuals or teams (*e.g.* Patrol)
- **Reactive**: are triggered by contextual events or environmental constraints (*e.g.* Encounter a hostile)
- **Opportunistic**: arise from explicit goals and priorities (*e.g.* Recharge battery)
- **Aleatoric**: are random but structured by choices, distributions, or parametric variations (*e.g.* vary behavior so not predicted)

Schedule Actions

- Adds *structure* to the simulations
- Assigned to individuals or teams
- Can be primitive or complex actions
- Who? What? When?
  - Participants don't need to be fully specified
  - *Robot_1 patrol the building*

A Quick Look at Outlook



Reactive Actions

- Adds realism to the simulations
- Emergent behaviors
- Created for individuals, teams, or all
- React to individuals, groups, all, object instances, object types, properties
- Triggered by perceptions
- Suspends or preempts current actions
- *Report location of all hostiles*

Opportunistic Actions

- Need based
- Needs are defined with decay rates
- Fulfillments are defined with growth indicates
- Opportunistic action priorities increase over time
- Attempt to schedule them in based on distance from path
- Will suspend other actions if needed
- *Low battery -> recharge*

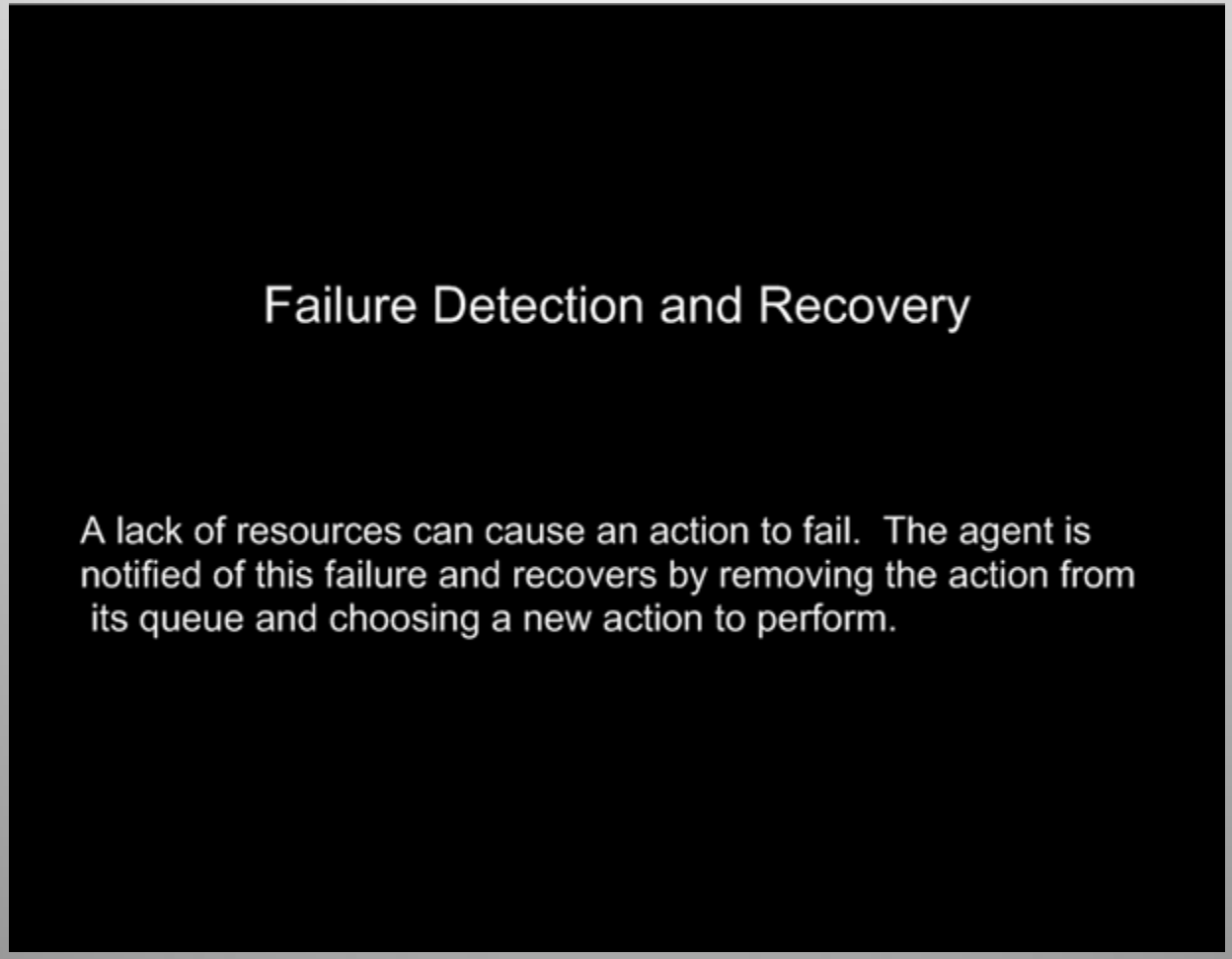
Aleatoric Actions

- Stochastic
- Adds variability
- Simple probability for each sub-action
- Sub-actions chosen when action is added to the queue
- Not hard coded
- Composed of other PARs
- *Change route*

Resource Manager

- Do not need to specify every participant for every action for every agent
- Based on types from Object Hierarchy
- Automatically created from environment file (and Actionary)
- Location based and global free list
- Allocate specific objects or find an object of the needed type in the needed location

Failure Detection and Recovery



Roles, Groups, and Teams

- Linked by naming convention
- Roles provide default locations, possessions, and actions (*i.e.* specialties)
- Groups allow actions to be specified for larger numbers
- Teams are composed of different roles
- Population is specified as number of agents in each group and therefore role.

Engineering School

Contributions

- Functional, heterogeneous crowds appropriate to time and place
- Semantically meaningful interactions with environment and other agents.
- High level, data-driven approach
- Readily extensible into new simulation domain (not hand scripted)
  - Actionary
- Emergent behaviors
- Reconfigurable environment

The Beginning

- Teams: cooperation, competition, coordination
- Refining up and down
- Social, cultural, psych