Engineering Characteristics of Autonomous Agent Architectures

Presentation: Nancy Reed*

Paper: Paul Scerri and Nancy Reed

Journal of Experimental and Theoretical Artificial Intelligence, vol. 12, pp. 191 - 212, April 2000.

*Department of Information and Computer Sciences
University of Hawai'i at Manoa, Honolulu, HI
nreed@hawaii.edu

Outline

- What is an agent architecture?
- How are agent systems developed?
- Are there differences between industry's needs and research needs?
- Candidate agent architecture characteristics for industry applications
- Summary

What is an Agent Architecture?

- A framework for developing agents
- Includes
  - philosophy - how agents make decisions
  - methodology - how to construct systems
  - code library - re-usable primitives
  - user interfaces - tools for agent construction and/or for run-time support
  - documentation
- Similar to a 'shell' for an expert system

Agent Architectures

- Wide range of fundamentally different architectures available
  - BDI - (belief, desire, intention), Jade,
  - Open Agent Architecture, JAT-lite
  - BeeGent, Aglets, Jack,
  - .....
- Agents built using many different architectures are capable of performing the required tasks

How to Choose an Architecture for an Agent Application?

- Currently - ad hoc criteria
- Should be similar to the choice of programming language/environment for the implementation of other software systems
- Current agent architecture descriptions do not include practical engineering characteristics (experience with projects?)

Developing Agent Systems

- Research
- Science

- Emphasis on finding new solutions and determining what can be done with different approaches

- Industry
- Engineering

- Emphasis on construction and delivery of quality products - on time and within budget

Current Agent Architecture Descriptions

- Scientific literature focuses on the evaluation of capabilities of agents
- Developers will *use* architectures rather than just *build* agents for them
- Equally important for the developer to know is how difficult it is to build agents with the required behavior in the architecture

7

Example Application - Simulation Environments

- Highly complex, real-time training and testing simulation environments
- Agents used in place of humans or other entities
- Examples
  - Air-combat simulation
  - Disaster management
  - RoboCup simulated soccer

8

Experience from Two Systems and Architectures

- Chicken factory
  - agents for soccer simulation
  - Simulator: RoboCup
- TACSI rule-system
  - agents for the control of fighter aircraft
  - Simulator: TACSI (tactical simulation)
  - system pre-dates EASE architecture and TACSI

9

What Characteristics could be More Important to Industry?

Potential Characteristics:

- Accessibility
- Representational Power
- Reuse
- Generality
- Project Methodology
- Correctness Measures
- Computational Requirements

10

Accessibility

- How easy is it for non-agent experts to create agents with the architecture?
- How much does a designer need to understand about the architecture in order to create useful agents?

Agent "gurus" only — Any Chris, Rick or Mary

11

Accessibility Experience

- Chicken Factory
  - relatively accessible
  - GUI interfaces
- TACSI rule-system
  - relatively accessible
  - GUI interfaces

12

Representational Power

- How powerful is the behavior specification method?
- Number of distinct reactions to specification elements or the amount of specification required for certain behavior

Verbose specifications — Succinct

13

Representational Power Experience

- Chicken Factory
  - relatively low - close to 1
  - specification element is a behavior
- TACSI rule-system
  - relatively low
  - number of states plus number of rules that specify behavior
  - however, built-in behaviors and pre-conditions mean that complex agents can be built with few specification elements

14

Reuse

- How easy is it to reuse parts of one specification in another specification?
- Factors: application areas, granularity of specification elements, quality of the design, and abstraction level

None/difficult — All/easy

15

Reuse Experience

- Chicken Factory
  - high within applications
  - low across
- e.g.
  - agent specifications
  - behaviors
  - states
  - architecture tools
- TACSI rule-system
  - low
  - copy/paste individual rules
- e.g.
  - state machines
  - specific states

16

Generality

- For what range of applications is the architecture useful?

One specific application only — General problem solving

17

Generality Experience

- Chicken Factory
  - low
  - specific to dynamic environments where sensing is low-cost and planning is not required
- TACSI rule-system
  - low
  - single goal at a time
  - independent resources
  - tailored to industry setting

18

Agent Development Process

- Which methodologies does the architecture support?
- Is it scalable as needed?
- Development time/cost?

Ad hoc hacking — Built-in development methodology

19

Agent Development Process Experience

- Chicken Factory
  - iterative refinement most used
  - difficult top-down?
  - environment necessary for testing/development (bottom-up difficult)
  - not scalable
- TACSI rule-system
  - no clear process, incremental development used
  - good modularity
  - should scale up

20

Computational Correctness

- Does the architecture support verification and validation of behavior?
- How easy is it to test and debug agents?

No verification/validation — Full verification and validation

21

Computational Correctness Experience

- Chicken Factory
  - difficult if not impossible to verify behavior
  - complex interaction of environment and behaviors is difficult to predict
  - cumbersome to test
  - easy to debug
- TACSI rule-system
  - no support for validation or verification
  - testing and debugging support is extensive

22

Computational Requirements

- Benchmark base
  - time between an unexpected event's detection and the first response action
  - time between the adoption of a new high-level goal to the first action pursuing that goal

Requires mainframe for a few agents — 1000s of agents on one PC

23

Computational Requirements Experience

- Chicken Factory
  - moderate
  - increase in computation is linear with the increase goal complexity
  - full team runs in real time on workstation/ PC
- TACSI rule-system
  - moderate
  - increase in computation with the number of rules is linear
  - rule system runs on same workstation as simulator

24

Trade-offs

- The ability of the system to produce the required functionality is not the only consideration for industry
- Cost of development, $ and time
- Ease of maintainability
- Fit to application environment - speed, resources, ...

25

Discussion

- Criteria suggested are guidelines, other characteristics likely exist
- Use good software engineering techniques in all development
- Hopefully more groups will describe these characteristics for their architectures

26

Summary

- Most capabilities required of agents can be achieved with available architectures
- Little emphasis is placed on how difficult it is for a programmer, or user, to develop agents to perform certain tasks
- We have proposed criteria by which we believe agent architectures should be evaluated in order for non-agent experts to find the architecture they need

27

Questions?

- What is an agent architecture?
- Are agents useful for solving all problems?
- What is most important to examine in agent architectures when selecting one for developing a system for a particular application?

28