Roadmap – Course Overview

- Introduction
- Cooperation Models and Mechanisms
- Communication

- Micro-Level Architectures
- Macro-Level Structures
- Development Methods
Roadmap – Chapter Details

Micro-Level Architectures

- Introduction
  - What is an Agent?
  - What is an Agent Architecture?
  - Typification of Agent Architectures

- Belief-Desire-Intention (BDI) Architectures

- Layered Architectures

- Constraint-oriented Architectures
Emerging “standard definition”:

An agent is a (computational) entity that is situated in some environment and that is able to pursue its design objectives flexibly, autonomously, and in cooperation with other (human or computational) entities.

Remarks on this “standard” – key characteristics:

- **situatedness**: agent is embedded in (integrated into) its surroundings
- **flexibility**: agent (inter-)acts reactively + proactively
- **autonomy**: under self-control, able to act independently (up to some degree)
- **objectives**: the agent’s purpose and goals, prespecified by a designer/user
Other characteristics of agency sometimes claimed to be essential:

- rationality
- mobility
- adaptivity
- introspection
- benevolence

Often mental attitudes are attached to agency, e.g.

- belief, knowledge, ... (information)
- intention, plan, commitment, ... (control)
- desire, preference, ... (motivation)
Comparing agents and objects

- Both encapsulate identity ("who"), state ("what"), and passive behavior ("how, if invoked").
- Agents additionally encapsulate active behavior ("when", "why", "with whom", "whether at all")
- The agent and object concepts
  - allow for qualitatively different system perspectives
  - are concerned with different levels of abstraction
  - thus are complementary rather than mutually exclusive
- Think of a gradual transition from agents to objects, rather than a sharp borderline (active object concept, constructs such as preconditioning in Eiffel)
Agents and the evolution of programming concepts (Odell, Jennings):

<table>
<thead>
<tr>
<th></th>
<th>MONOLITHIC</th>
<th>MODULAR</th>
<th>OO</th>
<th>AO</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIT BEHAVIOR</td>
<td>nonmodular</td>
<td>modular</td>
<td>modular</td>
<td>modular</td>
</tr>
<tr>
<td>UNIT STATE</td>
<td>external</td>
<td>external</td>
<td>internal</td>
<td>internal</td>
</tr>
<tr>
<td>UNIT INVOCATION</td>
<td>external</td>
<td>external</td>
<td>external</td>
<td>internal</td>
</tr>
</tbody>
</table>

- move from machine- to problem-oriented abstractions
- units show increasing localization and encapsulation
- thus: agents can be viewed as a natural next step
- evolution, not revolution
What is an Agent Architecture?

Architectures determine behavioral space:

- Interactive vs. isolated
- Levels of sociability
- Reactive vs. pro-active
- Levels of cognition
- Time-evolving behavioral path

Architecture =

arrangement of data and algorithms

+ flow of data and control
Typification of Agent Architectures

- Belief-Desire-Intention (BDI) architectures
- Layered architectures
- Constraint-oriented architectures
- Other characterizations:
  - reactive versus deliberative architectures
  - isolated versus social architectures
Roadmap – Chapter Details

Micro-Level Architectures

- Introduction
- BDI Architectures
  - General Principle
  - PRS
  - IRMA
  - GRATE*
  - COSY

Layered Architectures

Constraint-oriented Architectures
General Principle

General structure:

- **BELIEFS** ("database")
- **DESIREs** ("goals")
- **INTENTIONS** ("current goals")

Input: BELIEFS → DESIREs → INTENTIONS → reasoning

Output: reasoning → INTENTIONS → DESIREs → BELIEFS

The reasoning process involves inferencing and learning.
General flow of internal data and control (Wooldridge 1999):
PRS = "Procedural Reasoning System"
IRMA = “Intelligent Resource-bounded Machine Architecture”
top-level view:

- **interagent communication**
  - cooperation module
  - acquaintance model
  - self model
  - information store

- **cooperation & control layer**
  - situation assessment module

- **interface**
  - control module
  - task 1
  - task 2
  - task n

---

*GRATE* (Jennings 1993)
details:
top-level view:
RDRC in detail:

- Agenda
- Intention structure

Filter

- Deliberation component
- Reaction component

Scripts for execution & intra-protocol decisions
Micro-Level Architectures

- Introduction
- BDI Architectures
- Layered Architectures
  - General Principle
  - INTERRAP
  - TouringMachines
- Constraint-oriented Architectures
Structure and flow of data/control (from Wooldridge 1999)

(a) Horizontal layering  (b) Vertical layering (One pass control)  (c) Vertical layering (Two pass control)
**top-level view:**

- **situative context**
- **activity patterns**
- **mental context**

**filter + decision**

**behavior based** + **plan based**

**methods**

**filter + decision**

**execution**
details:

- cooperation component
- plan-based component
- behavior-based component
- social model (cooperation knowledge)
- mental model (planning knowledge)
- world model (behavior patterns)
- world interface (acting - communication - perception)
top-level view:

- perception subsystem
- modeling layer
- planning layer
- reactive layer
- action subsystem
- control subsystem
- CLOCK
Touring Machines (Cont’d)

Micro-Level Architectures • Layered Architectures

details on planning layer:

- Goal Stack
- Schema Library
- Topological World Map

sensors, resource monitor, other layers

Partial Planner

action effectors
details on modeling layer:
Micro-Level Architectures

- Introduction
- BDI Architectures
- Layered Architectures
- Constraint-oriented Architectures
  - General Principle
  - CCAF
  - Waffler
constraint = condition under which activity is to be carried out, thus behavior-influencing

“constraints everywhere”

- standard constraints: time, cost, quality
- others: individual preferences, collective preferences, psychological and social commitments, resource limitations, roles an agent has to play, conventions, ...

Key assumption: ability to act flexibly has much to do with flexible handling of constraints

usual distinction: soft versus hard constraints

particularly challenging: handling constraints in applications that are distributed, dynamic, and/or real-time
CCAF = "Constraint-Centered Architectural Framework"

Underlying assumptions:

- constraints and all agent-internal activities must be tightly intertwined
- an agent must be able to carry out activities in cooperation with others (shared/delegated), when required by constraints
- communication must be sensitive to constraints
- agents must be able to reason about constraints (quantification of strength, importance, risk of violation)
- constraint handling within an agent to be realized as a centralized process (efficiency)
top-level view:
Waffler: after a colloquialism for improvisation ("waffling")

**top-level view:**

```
| concepts |
| intentions |
| settings |
| reactions |
| WORKING MEMORY |
| pool of constraints |
| possible actions |
| deliberation |
| action |

**perception**

**desires**

**LONG-TERM MEMORY**

**control constraints**

---

*Waffler* (Anderson 1995)
Micro-Level Architectures • Constraint-oriented Architectures

the role of constraints in more detail:

perception → recalls → concepts intentions

pool of constraints

propose inhibit reinforce

possible actions

limit, focus

expectations

focus

limit recollection

limit exploration

action → deliberation

proposes