An Adaptive Middleware Core for a Multi-Agent Coordination Language

Andreas Witsch and Kurt Geihs

Distributed Systems Group
Kassel University
Table of Contents

Motivation

PROViDE Middleware Core

Results

Conclusion
Ways to Coordinate Agents
Ways to Coordinate Agents

Task 1
1..1

Task 2
1..1

Eat Hay Propositional

- $Z_0$ to $Z_3$
  - isEmpty
  - EatLeft
  - EatRight

- $Z_1$ to $Z_4$
  - isEmpty
  - EatLeft
  - EatRight
Ways to Coordinate Agents

Task 1
1..1

Task 2
1..1

Eat Hay Propositional

\[ Z_0 \xrightarrow{\text{isEmpty}} Z_3 \]
\[ Z_1 \xrightarrow{\text{isEmpty}} Z_4 \]

- EatLeft
- EatRight
- isEmpty

Z0
Z3
Z1
Z4
S
Ways to Coordinate Agents

Eat Hay Propositional

Eat First
0..2
EatLeft
Z\textsubscript{0}

Eat Second
0..2
EatRight
Z\textsubscript{1}
Ways to Coordinate Agents

Eat Hay Propositional
- Eat First
  - $Z_0$
  - EatLeft
- Eat Second
  - $Z_1$
  - EatRight

Eat Hay Dynamic
- $Z_0$
- Eat($x$)
Robotic Soccer Example

Gameplay

**Attack**

1..1

- **Z₀**: Find Ball
- **Z₁**: Get Ball
- **Z₂**: PassTo $\{x, y\}$
- **Z₃**: Score Goal

**Defend**

0..∞

- **Z₆**: Interfere
- **Z₇**: InterceptAt $\{x, y\}$

**Goalkeeping**

1..1

- **Z₄**: Position
- **Z₅**: Save Ball

Variables: $\{x, y\}$
Robotic Soccer Example

**Motivation**

**PROViDE**

**Results**

**Conclusion**

---

**Gameplay**

**Attack**  
1..1

- $Z_0$: Find Ball  
- $Z_1$: Get Ball  
- $Z_3$: Score Goal  
- $Z_2$: PassTo  
- Variables: $\{x, y\}$

**Defend**  
0..\infty

- $Z_6$: Interfere  
- $Z_7$: InterceptAt  
- Variables: $\{x, y\}$

**Goalkeeping**  
1..1

- $Z_4$: Position  
- $Z_5$: Save Ball

Variables: $\{x, y\}$

---

Andreas Witsch and Kurt Geihs  
An Adaptive Middleware Core for a Multi-Agent Coordination Language  
4 of 19
Coherent Decisions for Multi-Agent Systems

Research Goals:

1. Negotiation of decision proposals
2. High probability for success
   - Safety: Must agree on a single value (do nothing evil)
   - Liveliness: Must agree on proposed values only (do something good)
3. Required time for decisions should be low
Proposal Replication for Value Determination
PROViDE Middleware
PROViDE Overview

- PROViDE is a distributed decision middleware
- Requirements:
  - High availability with transient and unreliable communication
  - Adaptable to the problem
  - Persistent replication of decision proposals
  - Eventually agreement on one proposal
- Middleware features:
  - Data replication (distributed shared memory)
  - Publish-Subscribe
  - Clock synchronization
  - Tracking the current team
Steps to Achieve Agreement

- Consistency Strategy
  - How to distribute value proposals
- Acceptance Strategy
  - When to accept a proposal as own proposal
- Value Decision Strategy
  - Determine the current value from the known proposals
Steps to Achieve Agreement

◦ Consistency Strategy
    ⇒ How to distribute value proposals
Steps to Achieve Agreement

◊ Consistency Strategy
   ⇒ How to distribute value proposals

◊ Acceptance Strategy
   ⇒ When to accept a proposal as own proposal
Steps to Achieve Agreement

- Consistency Strategy
  ⇒ How to distribute value proposals

- Acceptance Strategy
  ⇒ When to accept a proposal as own proposal

- Value Decision Strategy
  ⇒ Determine the current value from the known proposals
PROViDE Consistency Levels

PROViDE:Robot a

![Command(VPa)](Command(VPa))

PROViDE:Robot b

accept()

$L = 1$
PROViDE Consistency Levels

PROViDE:Robot a

Command(VP_a)

L = 1

PROViDE:Robot b

Ack_a(VP_b)

L = 2

accept()
PROViDE Consistency Levels

PROViDE:Robot a

PROViDE:Robot b

Command(VPa)

accept()

L = 1

L = 2

L = 3

Acka(VPb)

ShortAckb
PROViDE Consistency Levels

⇒ When should we resend lost packets?
Resend Time

- Network delay jitter of single hop UDP packets follows a Laplace distribution (Zheng et al.)
  \[ P(D = t|\mu, b) = \frac{1}{2b} \exp \left( -\frac{|t-\mu|}{b} \right) \]
- By measuring multiple packet delays, we can use a maximum likelihood estimator for \( \mu \) and \( b \)
- Assuming symmetric delays, the probability distribution of an echo is \( D_e = D_s + D_a + C_p \) with \( D_a = D_s \)
Resend Time

The probability of acknowledgement after time $t_r$ is:

$$P(D_e \leq t_r|\mu, b)$$

$$= \int_{-\infty}^{t_r} P(D_a|\mu, b) \times P(D_a|\mu, b) \times P(D_p)$$

$$= \begin{cases} 
\frac{1}{4} e^{\frac{t_r-\Delta}{b}} (2 - \frac{t_r-\Delta}{b}), & t_r \leq 2\mu \\
\frac{1}{4} e^{-\frac{t_r-\Delta}{b}} (-2 - \frac{t_r-\Delta}{b}) + 1, & \text{otherwise}
\end{cases}$$

and solving for $t_r$:

$$t_r = \begin{cases} 
\Delta + bW_{-1} \left( \frac{-4P(D_e \leq t_r)}{e^2} \right) + 2b, & P(D_e \leq t_r) \leq 0.5 \\
\Delta - bW_{-1} \left( \frac{4(P(D_e \leq t_r) - 1)}{e^2} \right) + 2b, & \text{otherwise}
\end{cases}$$

where $W_{-1}(x)$ is the inverse function of $f(w) = we^w$.
Acceptance Strategy: Believe Ordering

A conflict resolution strategy is a function $C$, which decides for the highest ranked input belief for a Variable $x$:

$$VP_a(x) = C(VP_a(x), VP_b(x), ..., VP_n(x))$$

$$= \max(VP(x) \in <_o)$$

$\Rightarrow$ $C$ chooses the maximum w.r.t. a strict ordering relation $<_o$:

$$<_o \subseteq VP(x) \times VP(x)$$
Acceptance Strategy: Believe Ordering

A conflict resolution strategy is a function $C$, which decides for the highest ranked input belief for a Variable $x$:

$$VP_a(x) = C(VP_a(x), VP_b(x), ..., VP_n(x))$$

$$= \max(\{VP(x) \in <o\})$$

$\Rightarrow$ $C$ chooses the maximum w.r.t. a strict ordering relation $<o$:

$$<o \subseteq VP(x) \times VP(x)$$

**Example**

The default conflict resolution order beliefs by the Lamport time $<Lt$ = \{(VP_1(x), VP_2(x)) \mid Lt(VP_1(x)) < Lt(VP_2(x))\}
Value Decision

A simple rule to choose a value among the proposals:

- Most recent proposal
- Majority voting
- Collection
Results
Evaluation Setup
Transmission Time

The graph illustrates the transmission time in milliseconds (ms) as a function of packet loss percentage. The x-axis represents the packet loss percentage, ranging from 0% to 90%, and the y-axis represents the time in ms, ranging from 0 to 500 ms.

Three different protocols are compared:
- **PROViDE** (blue line) shows a steady increase in time as packet loss increases.
- **OpenSplice DDS** (green dashed line) exhibits a more pronounced increase in time with packet loss.
- **TCP** (magenta dotted line) displays a steep increase in time at higher packet loss percentages.

The graph highlights the impact of packet loss on transmission time for each protocol, demonstrating the efficiency and robustness of each communication method under varying network conditions.
Agreement vs Packet Loss

Time to reach agreement for a commanded value at slaves

Probability of achieving consistency

Packet Loss [%]

Time [ms]

Probability for Consensus [%]
Conclusion

PROViDE is ...

- an adaptable middleware for distributed decision making
- explicitly supporting negotiation of conflicting value proposals
- enabling to choose between a high success probability and a short decision time
- able to adapt to network errors
An Adaptive Middleware Core for a Multi-Agent Coordination Language

Andreas Witsch and Kurt Geihs

Distributed Systems Group
Kassel University