PRACTIONIST: implementing PRACTIcal ReasONIng SySTems

Vito Morreale, Susanna Bonura, Fabio Centineo, Alessandro Rossi

R&D Lab
ENGINEERING Ingegneria Informatica S.p.A.

Massimo Cossentino
ICAR – Italian National Research Council

Salvatore Gaglio
University of Palermo - DINFO
Overview

- The BDI model of agency
- Motivation
- **PRACTIONIST**: PRACTical reasONIing sySTem
  - Relevant abstractions
  - Execution model
  - PRACTIONIST Agent Introspection Tool
- Conclusions and future work
The BDI model recognizes the importance of beliefs, desires, and intentions in rational actions.

- **Beliefs**: information attitudes (they may be incomplete or incorrect)
- **Desires**: motivational attitudes, the states of affairs the agent “wish” to bring about
- **Intentions**: deliberative attitudes, the states of affairs the agent is committed to achieving

Based on theory of rational action in humans
The BDI theory of rational action was originally developed by Michael Bratman. It is a theory of practical reasoning: the process of reasoning (in everyday life), deciding moment by moment which action to perform next.

Central role of intention
Human practical reasoning consists of two activities:

- **deliberation**
  deciding **what** state of affairs to achieve

- **means-ends reasoning**
  deciding **how** to achieve these states of affairs

- The outputs of **deliberation** are **intentions**
- The outputs of **means-ends reasoning** are **intended means** (e.g. plans)
Intentions in Practical Reasoning

- Intentions
  - drive means-ends reasoning by figuring out the means to achieve them
  - persist: I will not drop my intentions without a good reason (e.g. I believe I achieved them, I believe I cannot achieve them, I believe there is no longer reason to achieve them, …)
  - constrain future deliberation: I will not consider options that are inconsistent with my current intentions
  - are related to beliefs about the future: I can plan taking into account that I will achieve them
While true
    observe the world;
    update internal world model;
    deliberate about what intention to achieve next;
    use means-ends reasoning to get a plan for the intention;
    execute the plan;
end while
Several simplifications have been made to trade off expressive power for a practical system.

Only explicit beliefs and goals have been considered. Often beliefs and goals are ground literals.

Intentions are represented as a structured set of plans.

...
BDI Implementations

- PRS system
- DMARS system
- J AM
- J ACK (commercial)
- J ADEX
- 3APL
- PRACTITIONIST
PRACTIcal reasONIng sySTem

PRACTIONIST is a Java-based (and currently JADE-based) framework with a Prolog Belief Base.
The PRACTIONIST Framework

PRACTIONIST Agent: components

- **Agent-specific components**
  - Belief Base
  - Goals
  - Actions
  - Plan Library

- **Agent-Environment Interaction**
  - Perceptors
  - Specific Agent
  - Effectors

- **PRACTIONIST Agent**
  - Perception Handler
  - BDI Event
  - BDI Action
  - Action Dispatcher
  - Planner
  - Deliberation
  - Means-ends Reasoning
  - Plan execution

**PRACTIONIST Framework**
It is possible to express the states of any formula $\varphi$ by using two **modal connectives**: 

\[ Bel(\alpha, \varphi): \text{the agent } \alpha \text{ believes that } \varphi \text{ is true}; \]

\[ Bel(\alpha, \neg \varphi): \text{the agent } \alpha \text{ believes that } \varphi \text{ is false}; \]

\[ Ubi f(\alpha, \varphi) \equiv \neg Bel(\alpha, \varphi) \land \neg Bel(\alpha, \neg \varphi): \]

the agent $\alpha$ does not have any belief about $\varphi$.

The formula $\varphi$ can be a modal logic formula such as:

\[ \varphi, \lozenge \theta, \Box \lozenge \tau, \forall x \psi(x) \]

**Form of predicates:**  *predicate-name*(r1: v1, ..., rn: vn)

\[ \text{on(under: block3, over: block4)} \]
The PRACTITIONIST Framework

Nested beliefs

\[ Bel(\alpha, bel(\beta, \varphi)) \]

\( \alpha \) believes that it is true that \( \beta \) believes that \( \varphi \) is true

\[ Bel(\alpha, \neg bel(\beta, \varphi)) \]

\( \alpha \) believes that it is not true that \( \beta \) believes that \( \varphi \) is true
The \textbf{PRACTIONIST} Framework

\textbf{Entailed beliefs (KD45)}

\begin{align*}
\text{Bel}(\varphi \Rightarrow \psi) & \Rightarrow (\text{Bel } \varphi \Rightarrow \text{Bel } \psi) \quad \text{(K)} \\
\text{If you believe that } \varphi \text{ implies } \psi \text{ then if you believe } \varphi \text{ then you believe } \psi
\end{align*}

\begin{align*}
\text{Bel } \varphi & \Rightarrow \neg \text{Bel } \neg \varphi \\
\text{(D)} \\
\text{This is the consistency axiom, stating that if you believe } \varphi \text{ then you do not believe that } \varphi \text{ is false}
\end{align*}

\begin{align*}
\text{Bel } \varphi & \Rightarrow \text{Bel } \text{Bel } \varphi \\
\text{(4)} \\
\text{If you believe } \varphi \text{ then you believe that you believe } \varphi
\end{align*}

\begin{align*}
\neg \text{Bel } \varphi & \Rightarrow \text{Bel } \neg \text{Bel } \varphi \\
\text{(5)} \\
\text{If you do not believe } \varphi \text{ then you believe that you do not believe that } \varphi \text{ is true}
\end{align*}

\textbf{if } \varphi \text{ is a theorem of KD45 then so is Bel } \varphi \quad \text{(Nec)}

You believe all theorems implied by the logic
The PRACTITIONIST Framework

Belief base example

\[ \text{bel}(self, table(name : table1)) \]
\[ \text{bel}(self, table(name : table2)) \]
\[ \text{bel}(self, table(name : table3)) \]

\[ \text{bel}(self, on(over : block4, under : block3)) \]
\[ \text{bel}(self, on(over : block5, under : block4)) \]
\[ \text{bel}(self, on(over : block8, under : block5)) \]

\[ \ldots \]
\[ \text{bel}(self, clear(obj : block10)) \]
\[ \text{bel}(self, clear(obj : block7)) \]
\[ \text{bel}(self, clear(obj : block8)) \]

\[ \text{bel}(Whoever, over(up : B1, down : B2) \iff \text{on}(over : B1, under : B2)) \]
\[ \text{bel}(Whoever, over(up : B1, down : B2) \iff \text{on}(over : B1, under : B3) \land \text{over}(up : B3, down : B2)) \]
### The PRACTITIONIST Framework

#### Plans

**Recipes** to achieve intentions, to handle external events, or to handle belief updates.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Unambiguous (within each agent) identifier of plans</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Trigger event</strong></td>
<td>If this event matches the selected event, this plan can be activated. In this case the plan is defined as <em>practical</em>.</td>
</tr>
<tr>
<td><strong>Context</strong></td>
<td>A modal logic formula that, when believed true by the agent, makes <em>applicable</em> a practical plan, so that the agent can select it to pursue its objectives.</td>
</tr>
<tr>
<td><strong>Success condition</strong></td>
<td>When the agent believes that this condition holds, the plan ends with success, regardless its execution state.</td>
</tr>
<tr>
<td><strong>Cancel condition</strong></td>
<td>When the agent believes that this condition holds, the plan ends with failure, regardless its execution state.</td>
</tr>
<tr>
<td><strong>Body</strong></td>
<td>Set of acts that are performed during the execution of the plan. The body defines the actual behavior of the plan.</td>
</tr>
<tr>
<td><strong>Invariant</strong></td>
<td>Condition that must remain true during the execution of the plan. As soon as it becomes false (at least according to the agent’s point of view), it will try to restore it.</td>
</tr>
<tr>
<td><strong>Belief updates in case of success</strong></td>
<td>Effects of this plan, in terms of belief updates in case the plan ends with success.</td>
</tr>
<tr>
<td><strong>Belief updates in case of failure</strong></td>
<td>Effects of this plan, in terms of belief updates in case the plan ends with failure.</td>
</tr>
</tbody>
</table>
The PRACTITIONIST Framework

Available plans

Plan library:
set of built-in plans + dynamically generated plans

- **Backward state-space search - regression planner**
  which, from a state of affairs the agent wants to bring about, builds a plan optimized with respect to the number of selected actions.

- The planner may be invoked **when**:
  - a state goal must be satisfied;
  - desired plans are missing.

```plaintext
action(move(block: Block, to: To),
  inputs: [Block, To],
  outputs: [],
  preconditions: [ on(over: Block, under: From),
    clear(obj: To), clear(obj: Block) ],
  success: [ -clear(obj: To), -on(over: Block, under: From),
    +clear(obj: From), +on(over: Block, under: To) ],
  failure: []
)```

The PRACTITIONIST Framework

PRACTITIONIST Agent execution model

PRACTITIONIST Agent

Perceptors → Event Queue → Build intended means → Event selection → Options → Intention commitment → Execute intended means → Effectors

Belief Base

Actions

Planning

Entailed plans

Entailed event

Perception

External event
The PRACTITIONIST Framework

PRACTITIONIST Agent execution model
The PRACTITIONIST Framework

Intended means

- Intended Means 3
  - Executing plan 3

Main plan
The PRACTIONIST Framework

PRACTIONIST Agent execution model

PRACTIONIST Agent

- Perceptors
- Belief Base
- Event Queue
- Event selection
- Build intended means
- Options
- Plan Library
- Intention commitment
- Execute intended means
- Reasoning about goals
- Goals
- Effectors

perception
external event
belief event
belief updates
The PRACTIONIST Framework

PRACTIONIST Agent execution model
The PRACTIONIST Framework

Intended means stacks

- **Synch goal events**
  - Intended Means 3
    - Executing plan 3
  - Intended Means 2
    - Executing plan 2
  - Intended Means 1
    - Executing plan 1
- **External events**
- **Belief update events**
- **Asynch goal event**
The PRACTITIONIST Framework

PRACTITIONIST Agent execution model

PRACTITIONIST Agent

- Perception
  - External event
  - Belief event

Event Queue

- Event selection
  - Selected event

Build intended means

- Intended means (new stack)
  - Intended means (existing stack)

Options

- Plans
  - Practical plans

Planning

- Enlisted plans

Intention commitment

- Execute intended means

Reasoning about goals

- Goals
  - Desire to pursue a goal

Effectors

- Action

Concurrent execution
The PRACTITIONIST Framework

PRACTITIONIST Agent execution model

PRACTITIONIST Agent

- Perceptors
- Belief Base
- Event Queue
- Event selection
- Build intended means
  - Intended means (new stack)
  - Intended means (existing stack)
- Options
  - Practical plans
  - Plans
- Plan Library
- Actions
  - Entailed plans
- Planning
- Effectors
- Action

Perception
- External event
- Internal event

Belief updates

Intended means stacks

Goal event

Desire to pursue a goal
The PRACTITIONIST Framework

Goals

\[ G = \{ \Sigma_G, \Pi_G, \Gamma_G, \Xi_G \} \]

where:

- \( \Sigma_G \) is the success condition of the goal \( G \), which is true if and only if the goal succeeds;

- \( \Pi_G \) is the possibility condition of the goal \( G \), that is \( G \) is achievable if and only if this proposition is true;

- \( \Gamma_G \) is the set of goals inconsistent with the goal \( G \), that is when each goal in this set succeeds the goal \( G \) does not succeed;

- \( \Xi_G \) is the set of goals whose success implies the success of the goal \( G \).
The PRACTITIONIST Framework - Goals

Families of goal

- **State goals** describe some states of affairs the agent wish and probably will intend to bring about, or cease, or preserve, or avoid.
  - **Achieve**, which represents what kind of world state an agent wants to bring about;
  - **Cease**, which represents a world state an agent wants to stop;
  - **Maintain**, which has the purpose to observe some desired world state and continuously re-establish this state when it does not hold;
  - **Avoid**, which has the purpose to observe some desired world state and continuously re-cease this state when it holds;
  - **Query**, which aims at having some information about a state of affairs.

- **Perform goals** are not related to some desired world states but to some activities the agent wants to perform.
The **PRACTITIONIST Framework - Goals**

**Goal vs. Desire vs. Intention**

\[
G = \{ \Sigma_G, \Pi_G, \Gamma_G, \Xi_G \}
\]

e.g. `achieve(on(over: block1, under: table3))`

- Check if the **Goal** is inconsistent with active goals
- Check if the **Goal** is impossible
- Check if the **Goal** already succeeds
- Check if the **Goal** is implied by some active goal

**Desire**

**Intention**

**Means-ends**

**Intended means execution**

*In case of failure, alternative goals might be achieved*
The PRACTIONIST Framework

PRACTIONIST Agent execution model

PRACTIONIST Agent

Perceptors
Belief Base

Event Queue

Event selection

Build intended means

Options

Plan Library

Actions

Planning

Entailed plans

Effectors

Goals

Intended means stacks

Desire to pursue a goal

New plan

Intended means (new stack)

Intended means (existing stack)

Entail event

External event

perception
**The PRACTIONIST Framework**

**Intention commitment strategies**

**Blind Commitment**
The agent will continue to maintain an intention until it believes that the intention has actually been achieved.

**Single-Minded Commitment**
The agent will continue to maintain an intention until it believes that either the intention has been achieved or it is no longer possible to achieve the intention.

**Open-Minded Commitment**
The agent will continue to maintain an intention as long as it is still believed possible.
The PRACTIONIST Framework
Commitment Strategies

**Blind Commitment**
The agent will continue to maintain an intention until it believes that the intention has actually been achieved.

**Open-Minded Commitment**
The agent will continue to maintain an intention as long as it is still believed possible.
PRACTIONIST Agent Introspection Tool
**PRACTIONIST Agent Introspection Tool**

**Belief Base**

Belief:
- Bel(self, or
  - (clear :obj block10),
  - (clear :obj block1))

Bel(self, Bel
  - BuilderAgent(on
    :under block4
    :over block5))

Bel(self, not
  - BelBuilderAgent(on
    :under table1
    :over block3)))

Bel(self, Bel
  - ArchitectAgent(on
    :under block2
    :over block4))

Bel(self, not Bel
  - ArchitectAgent(on
    :under table1
    :over block3)))
PRACTIONIST Agent Introspection Tool
Belief Base updating
PRACTIONIST Agent Introspection Tool
Event Queue
PRACTIONIST Agent Introspection Tool

Intended Means

<table>
<thead>
<tr>
<th>Severity</th>
<th>Level</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFO</td>
<td>TopLevelPlan FixBlockOnAnotherPlan.Cla...</td>
<td>1.8.1.1 ClearBlockPlan completed ..........</td>
</tr>
<tr>
<td>WARNING</td>
<td>TopLevelPlan FixBlockOnAnotherPlan.Cla...</td>
<td>1.8.1.1 this is an example...</td>
</tr>
<tr>
<td>INFO</td>
<td>TopLevelPlan FixBlockOnAnotherPlan.Cla...</td>
<td>1.8.1 ClearBlockPlan completed ..........</td>
</tr>
<tr>
<td>WARNING</td>
<td>TopLevelPlan FixBlockOnAnotherPlan.Cla...</td>
<td>1.8.1 this is an example...</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.8 this is an example...</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan</td>
<td>1.8 Achieving (fix :under block8 :over block9)</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.9 FixBlockOnAnotherPlan body started ...</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.9 Achieving (clear :obj block8)</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.9 Achieving (clear :obj block9)</td>
</tr>
<tr>
<td>INFO</td>
<td>TopLevelPlan FixBlockOnAnotherPlan.Cla...</td>
<td>1.9.1 ClearBlockPlan body started ...</td>
</tr>
<tr>
<td>INFO</td>
<td>TopLevelPlan FixBlockOnAnotherPlan.Cla...</td>
<td>1.9.1 Achieving clear(obj: block10)</td>
</tr>
<tr>
<td>INFO</td>
<td>TopLevelPlan FixBlockOnAnotherPlan.Cla...</td>
<td>1.9.1 ClearBlockPlan completed ..........</td>
</tr>
<tr>
<td>WARNING</td>
<td>TopLevelPlan FixBlockOnAnotherPlan.Cla...</td>
<td>1.9.1 this is an example...</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.9 FixBlockOnAnotherPlan body ended ...</td>
</tr>
<tr>
<td>ERROR</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.9 this is an example...</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan</td>
<td>1.8 Achieving (fix :under block9 :over block10)</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.10 FixBlockOnAnotherPlan body started ...</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.10 Achieving (clear :obj block5)</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.10 Achieving (clear :obj block10)</td>
</tr>
<tr>
<td>DEBUG</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.10 FixBlockOnAnotherPlan body ended ...</td>
</tr>
<tr>
<td>ERROR</td>
<td>TopLevelPlan FixBlockOnAnotherPlan</td>
<td>1.10 this is an example...</td>
</tr>
</tbody>
</table>
PRACTIONIST Agent Introspection Tool

Messages

Direction: Outgoing
Convo.: buyer@AcerNegozio04:1099/JADE
Receivers: (agent-identifier name buyer@AcerNegozio04:1099/JADE)
Reply-to: 
Communicative act: request
Content:
sell(item: book|title: Il codice da vincere)

Language: Language
Encoding: 
Ontology: Ontology
Protocol: Protocol
Conversation id: 123456
In-reply-to: buyer@AcerNegozio04:1099/JADE 1205421534
Reply-with: buyer@AcerNegozio04:1099/JADE 1205421534
Reply-by: 
User Properties: 

Direction: Outgoing
Sender: seller@AcerNegozio04:1099/JADE
Receivers: (agent-identifier name seller@AcerNegozio04:1099/JADE)
Reply-to: 
Communicative act: inform
Content:
has_been_ordered(item: book|title: Il codice da vincere)

Language: Language
Encoding: 
Ontology: Ontology
Protocol: Protocol
Conversation id: 123456
In-reply-to: seller@AcerNegozio04:1099/JADE 1205421534
Reply-with: seller@AcerNegozio04:1099/JADE 1205421534
Reply-by: 
User Properties: 

PRACTITIONIST Agent Introspection Tool

Plan Library

Plan description

- Trigger event:
  MsgEDIEvent[Message: (REQUEST; ontology :)
  
- Success belief adds:
  
- Context:
  (ready :who self)

- Success belief deletes:

- Cancel:
  (not :what (ordering :item blocks))

- Failure belief adds:

- Invariant:
  (ableToOrder :who self)

- Failure belief deletes:

- Args:

- Success:
  (ordered :blocks (set #0 tables #1 block)

- PlanID:

- user click:

OK
Conclusions

PRACTITIONIST wrt BDI features

- Deliberation process
- Means-ends reasoning
- Re-planning: alternative plans
- Commitment strategy: Single-Minded Commitment
- Intentions constrain future deliberation (options that are inconsistent with current intentions are not considered)
- Intention reconsideration
- Dynamic planning
- Belief logic
- Desire logic
- Intention logic
Conclusions

Other features

- Flexibility in the trade-off between reactive and proactive abilities
- Very important the concept of Goal
  - Theory on reasoning about goals
- It is possible to declare several Perceptors for corresponding Perceptions
- It is possible to declare several Effectors for corresponding Actions
- Application developed: travel planner
Conclusions

Future work

- Increasing the **expressivity** of **Beliefs**
- **Declarative language** for Plans, Goals, Actions, Belief Set (beliefs e belief formulas), …
- Formal specification of PRACTIONIST agents
- Implement **applications** in several domains (e.g. agro-food e-marketplace, financial risk management, production capacity planning, …)
- **Development tool** (Eclipse plug-in)
- **Methodology**
Thanks !!!

- **Contact info:**
  Vito Morreale  vito.morreale@eng.it
  Research & Development Lab (Palermo)
  ENGINEERING Ingegneria Informatica S.p.A.

- **Web site:**
  www.practionist.org