Methodological Extensions to Object-Oriented Approaches

- A means for agent technologies to gain traction within industrial settings may be by being introduced through well-established technologies.
- The Unified Modeling Language (UML) is gaining wide acceptance for the representation of engineering artifacts using the object-oriented paradigm.
- There are several attempts to extend UML so as to encompass agent concepts.
- In general, building methods and tools for agent-oriented software development on top of their object-oriented counterparts seems appropriate:
  - It lends itself to smoother migration between these different technology generations.
  - It improves accessibility of agent-based methods and tools to the object-oriented developer community which, as of today, prevails in industry.
The Prometheus Methodology

• Phases
• Tools
• From Prometheus to ROADMAP

Prometheus

- Prometheus, is an iterative methodology covering the complete software engineering process
  - Analysis, Design, Detailed design, Implementation

- Aims at the development of intelligent agents (in particular BDI agents)
  - Uses goals, beliefs, plans, and events.

- The resulting specification can be implemented in any agent implementation software that covers such abstractions
  - Specially aimed for implementation with JACK

- It is evolved out of practical experiences

- It is aimed at industrial software development, not researchers
Prometheus Overview

- Methodology developed over 7-8 years in collaboration with industry partner (Agent Software). Feedback from many students and industry partner clients.
- Focus on detailed guidance and structure to facilitate tool support.
- Mixture of
  - graphical notation for overview
  - (structured) text notation for detail.
- Hierarchical and modular.
- Prototype tool available and used externally

Prometheus Phases

- The Prometheus methodology covers three phases
  - The system specification focuses on identifying the basic functions of the system, along with inputs (percepts), outputs (actions) and their processing (for example, how percepts are to be handled and any important shared data sources to model the system’s interaction with respect to its changing and dynamic environment)
  - The architectural design phase subsequent to system specification determines which agents the system will contain and how they will interact
  - The detailed design phase describes the internals of each agent and the way in which it will achieve its tasks within the overall system. The focus is on defining capabilities (modules within the agent), internal events, plans and detailed data structures.
Prometheus

Process Overview

Prometheus

System Specification Phase
Prometheus
System Specification phase

- System defined by
  - Goals: goal diagram
  - Scenarios: user case scenarios
  - Functionalities: functionality descriptors

- System interface with environment described in terms of
  - actions,
  - percepts
  - external data

---

Prometheus
System Specification phase: Steps (non-sequential!)

- Start with high-level description of the system (textual)
- Identify actors
- Identify top-level scenarios for each actor
- Identify inputs/outputs (actions/percepts)

The online bookstore’s percepts and events include customers visiting the website, selecting items, placing orders using forms, and receiving email from customers, delivery services and book suppliers. Actions include bank transactions, sending email, and placing delivery orders.
Prometheus
System Specification phase: Steps (non-sequential!)

- Add a corresponding system goal for each use-case

**System Design**

4. Multiagent Systems Design

jvazquez@lsi.upc.edu

---

Prometheus
System Specification phase: Steps (non-sequential!)

- Apply Goal Abstraction to system goals
- Refine Goal (OR/AND refinement)
- Link goals to (sub)scenarios

**Scenario**

- Maintain large range of books
  - **how?**
  - **why?**

- Order books
  - **AND**
  - **how?**

- Find cheapest price
- Organise delivery
- Log Order

**Actions**

- Borrow books from other libraries

jvazquez@lsi.upc.edu
Prometheus
System Specification phase: Steps (non-sequential!)

- Identify the functionalities of the system
  - Idea: identify roles and activities

**NAME:** Welcoming  
**Description:** Welcomes a new visitor to the world wide web site (with personalised information if possible).  
**Percepts/events/messages:** CustomerArrived (message), CustomerInformation (message)  
**Messages sent:** CustomerInformationRequest (message), CustomisedWWWPage (message).  
**Actions:** DisplayCustomisedWWWPage  
**Data used:** CustomerDB, CustomerOrders  
**Interactions:** CustomerManager (via CustomerInformationRequest, CustomerInformation) OnlineInteraction (via CustomisedWWWPage, CustomerArrived)

Prometheus
System Specification phase: Steps (non-sequential!)

- Develop and refine the Scenarios and sub-scenarios
  - Steps inside a scenario consist of:
    - Incoming event/percept (→ receiving functionality)  
    - Message (sender → receiver)  
    - Activity or actions (→ functionalities)

**Scenario:** Book Order  
**Overview:** The user orders a book. Delivery options are explored and then confirmed (with an OrderRequest). The books are shipped, stock updated, and the user notified.  
**Context:** Assumes this book is in stock.  
**Steps:**
1. **Event** BookOrder (→ Online Interaction)  
2. DeliveryOptionQuery (Online Interaction → Transport Information)  
3. DeliveryOptions (Transport Information → Online Interaction) Data saved: Transport FIR  
4. Obtain preferred delivery option (Online Interaction)  
5. MakePayment (Online Interaction → Sales Transaction)  
6. **Action** FulfillTransaction (Sales Transaction)  
7. PlaceOrder (Sales Transaction → Order Handling)  
8. Register order (Order Handling) Writes data: CustomerOrders  
9. **Action** TransactCustomerCompany (Order Handling)  
10. DecreaseStock (Order Handling → Stock Managers)  
**Variations:** steps 6 (actual payment) and 10 (decrease stock) replaced with notification of delay (Order Handling to Customer Contact) and then placing an order for more stock (Order Handling to Stock Manager).
Prometheus
Architectural Design Phase

System specification artifacts

- Scenarios
- Actions, Percepts
- System goals
- Functionality descriptors

Architectural Design

- Interaction diagrams
- Conversation protocols
- System overview
- Agent descriptors

Detailed design

4. Multiagent Systems Design

Prometheus
Architectural Design Phase: Agent types

- Identify the agent types in the system
  - Group functionalities to agent types based on cohesion and coupling
  - Group functionalities that are
    - related based on common sense
    - group functionalities that require a lot of the same information:
      - Data Coupling Diagram
  - Do not group functionalities that are
    - clearly unrelated
    - exist on different hardware platform
    - security and privacy
    - Modifiable by different people
- Evaluate grouping:
  - Simple descriptive names (heuristic)
  - Generate agent acquaintance diagram

jvazquez@lsi.upc.edu
4. Multiagent Systems Design

Prometheus
Architectural Design Phase: Agent Descriptors

- Generate Agent Descriptors based on the agent types
  - How many agents of a each agent type (one, many, dynamic)?
  - What is the life time of the agent?
  - What is the initial state of the agent?
  - What should be done when agent is killed?
  - What is the data used/produced by the agent?
  - To which event the agent should react?
Prometheus
Architectural Design Phase: Agent Descriptors

Name: Sales Assistant agent
Description: greets customer, follows through site, assists with finding books
Cardinality: one/customer
Lifetime: Instantiated on customer arrival at site; Demise when customer logs out or after inactivity period
Initialisation: Obtains cookie, Reads Customer DB
Demise: Closes open DB connections
Functionalities included: Online Interaction, Sales Transaction, Welcomer, Book Finder
Uses data: Customer DB, Customer Orders, Stock DB
Produces data: Customer preferences, orders, queries
Goals: Welcome customer, Update customer details, Respond to queries, Facilitate purchases
Events responded to: new arrival, customer query, customer purchase, credit check response customer response
Actions: Display information to customer (greetings, book info, info requests, Display customised WWW page, RequestCreditCheck messages
Interacts with: Warehouse Manager (book request protocol), Delivery Manager (order protocol, order query protocol), Customer Manager (customer information query protocol, customer information update protocol)

Prometheus
Architectural Design Phase: System Overview Diagram
Prometheus Detailed Design Phase

- The details of the agent internals are developed
  - Defined in terms of capabilities, data, events and plans
  - Process diagrams are used as stepping stone between interaction protocols and plans
- Steps (I)
  - Develop the internal structure of individual agents
  - Identify the capability of each agent (start with functionalities)
  - Generate capability descriptors

Name: Delivery Problem Handling
External interface to the capability: events used/produced
Natural language description: Respond if books are not in stock
Interaction with other capabilities: Transport capability
Data used/produced by the capability: Note problem to transport capability
Inclusion of other capabilities: None

- Generate agent overview diagrams
Prometheus
Detailed Design Phase: Agent Overview Diagrams

Prometheus
Detailed Design Phase: Agent Overview Diagrams
Prometheus
Detailed Design Phase: Event, Data & Plan Descriptions

- Steps (II)
  - Plan descriptions

  **Name**: Delivery Problem Handling
  **Natural language description**: Respond if books are not in stock
  **Triggering event type**: Delivery problem, Delayed delivery
  **Plan steps**: Delivery Query, Register problems
  **Context of performing the plan**: The delivery is delayed
  **Data used/produced**: Produce note problem

- Event descriptions
  - Identify the purpose of events and the data carried by it

- Data descriptions
  - Identify the data structure and operations on the data
Prometheus
Tools: the Prometheus Design Tool (PDT)

- System Specification
- Architectural Design
- Detailed Design
- Implementation
- Debugging
- Testing

From Prometheus to ROADMAP

- Main strengths:
  - Structured processes to refine design.
  - Automated consistency checking between (some of) the design artefacts.
  - Hierarchical and modular views.

- Actively continuing development…

- Next step: the ROADMAP methodology
  - More abstract and high level than Prometheus.
  - Concerned with high level view of models needed.
  - Focusses particularly on requirements analysis.
  - Notation for new elements still being defined.

jvazquez@lsi.upc.edu
ROADMAP
Overview of Models

Domain specific
- Environment Model
- Knowledge Model

Application specific
- Goal Model
- Role Model
- Agent Model
- Interaction Model

Reusable service models
- Social Model
- Service Model

Prometheus provides details in these models - and a little in the environment model.

4. Multiagent Systems Design

ROADMAP
Models

- **Goal hierarchy** (Requirements, propagates down)
- **Roles** associated with goals (Requirements)
- **Interaction model:**
  - Scenarios (Requirements).
  - Protocols (Architectural design)

Requiring more work:

- Knowledge Model
- Environment Model
- Possibly need a Task Model
- Social Model
- Services Model
ROADMAP
Integration with Prometheus

- Prometheus actors/stakeholders and functionalities become external/internal roles
- Can identify goals or scenarios at top level
- Add soft goals as annotations on all entities
- Percepts and actions possibly wait till architectural design
- Still need to decide common notation

ROADMAP
Example of new models: Goal Model
References


These slides are based mainly in [2], [4], [5] and material from M. Winikoff, L. Padgham, M. Luck and M. Dastani