ACAI'2001 & EASSS'2001

Prague
6 July 2001

Multi-Agent Oriented Methodology And Programming

(provisional set of slides)

Yves Demazeau
Yves.Demazeau@imag.fr

LECTURE CONTENTS

INTRODUCTION

What is an Agent ?

External Definition : a **real** or **virtual** entity that evolves in an **environment**, that is able to **perceive** this environment, that is able to **act** in this environment, that is able to **communicate** with other agents, and that **exhibits** an **autonomous** behaviour

---> autonomous agents

---> the autonomy principle

What is an Agent ?

External Definition : a **real** or **virtual** entity that evolves in an **environment**, that is able to **perceive** this environment, that is able to **act** in this environment, that is able to **communicate** with other agents, and that **exhibits** an **autonomous** behaviour

---> the autonomy principle

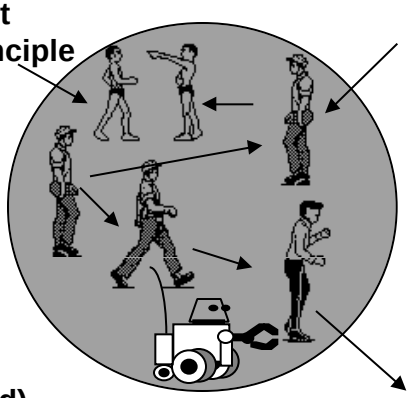
Internal Definition : a **real** or **virtual** entity that **encompasses** some **local control** in some of its **perception** , **communication** , **knowledge acquisition** , **reasoning** , **decision** , **execution**, **action** processes.

---> the delegation principle
---> personal assistants, mobile objects, AI systems

What is a Multi-Agent System ?

A set of possibly organized agents which interact in a common environment

---> the distribution principle



MAS main interests :

---> To extend classical mono-agent AI models and tools (A-centered)

---> To study specific multi-agent models and tools (MAS-centered)

Agents and Multi-Agent Systems

External Definition : a **real** or **virtual** entity that evolves in an **environment**, that is able to **perceive** this environment, that is able to **act** in this environment, that is able to **communicate** with other agents, and that **exhibits** an **autonomous** behaviour

---> the autonomy principle

Internal Definition : a **real** or **virtual** entity that **encompasses** some **local control** in some of its **perception** , **communication** , **knowledge acquisition** , **reasoning** , **decision** , **execution**, **action** processes.

---> the delegation principle

But there is no agent without multi-agent systems !

Agents Environments Interactions Organisations

Agents
- **internal architectures of the processing entities**

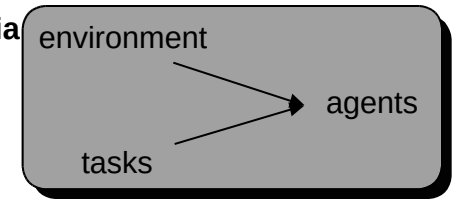
Environment
- **domain-dependent elements for structuring external interactions between entities**

Interactions
- **elements for structuring internal interactions between entities**

Organisations
- **elements for structuring sets of entities within the MAS**

Distributed Problem Solving

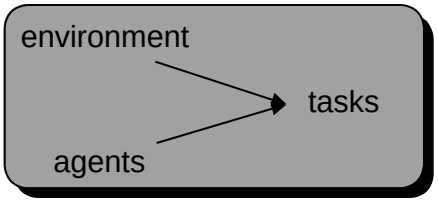
global conceptual model
global problem
global success criteria
division of :
- knowledge
- resources
- control
- authority



focus on the collaborative resolution of global problems by a set of distributive entities
- society goals directed
- input : tasks, environment
- output : model of the distributed entities
- schema to solve the tasks

Decentralized System Simulation

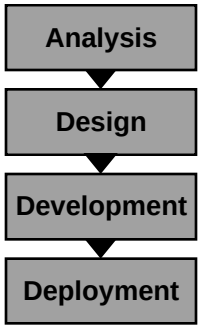
local conceptual models
local problems
local success criteria
division of :
- knowledge
- resources
- control
- authority



focus on the coordinated activities of a set of agents evolving in a multi-agent world
- agent goals directed
- input : agents, environment
- output : tasks which can be solved
- schema to solve the tasks

MAS Methodology

Methodology
= Approach + Model + Tools + Problem + Domain
= Analysis + Design + Development + Deployment



Identify the problem and the domain

Get rid of the domain / Define the solution

Implement the solution / Plug the domain

Apply the solution to the problem/domain

MAS Applications

Computer-Aided Design
Computer Vision
Decision Support
Electronic Commerce
Entreprise Modelling
Manufacturing Systems
Natural Language Processing
Network Monitoring
Office and Home Automation
Robotics Control
Societies Simulation
Spatial Data Handling
Telecommunication Routing
Traffic Management

ANALYSIS AND DESIGN

SATURNE : Origin of the studies

Building, maintaining, using a world description from data issued by several sensors
Building an open, domain-independent system

- Decomposing the knowledge representation problem into level-agents (cf. abstraction, decentration)
- Decomposing the knowledge processing problem into focus-agents (cf. focalisation / characteristics)
- Intersecting the level-agents and the focus-agents into basic agents
- Two behaviours to be exhibited by the society :

---> modelling : scene understanding
---> interpreting : recognition and localisation

SATURNE : Horizontal levels of representation

abstraction

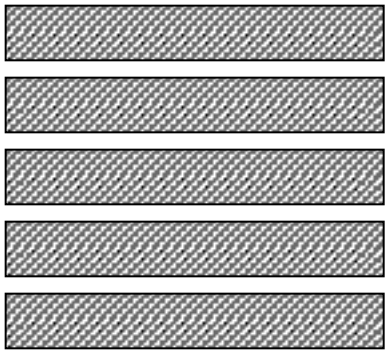
cf.
representation

+

decentration

cf.
referential

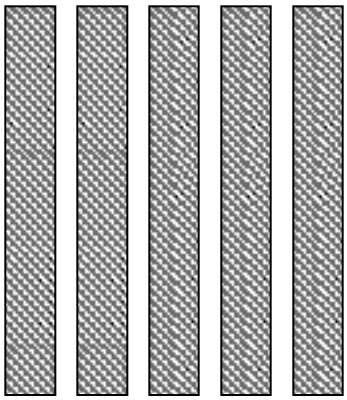
- scene
- object
- scene features
- image features
- images



SATURNE : Vertical foci of attention

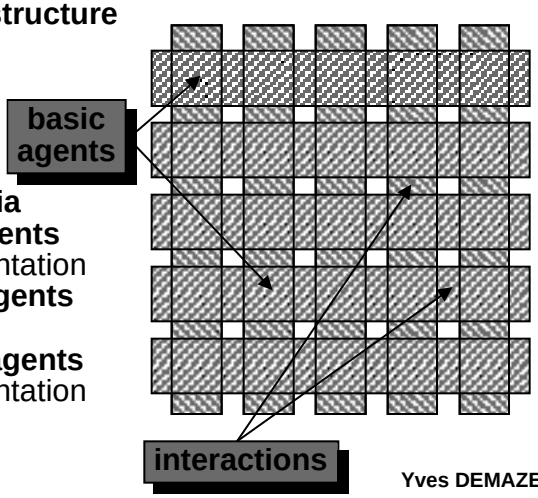
explicitely designed
cf. characteristics

- contours
- highlights
- range data
- stereo vision
- regions
- ...



SATURNE : Agents and Society of Agents

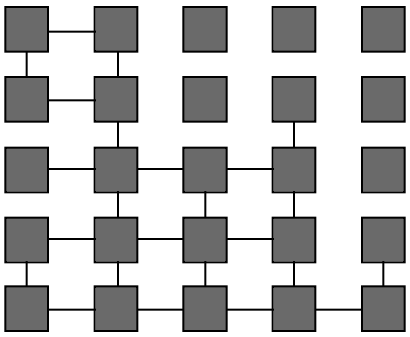
organisational structure
horizontal links
vertical links



interaction media
between foci agents
levels of representation
between level agents
foci of attention
between basic agents
levels of representation
x foci of attention

SATURNE Behaviour : Scene Understanding

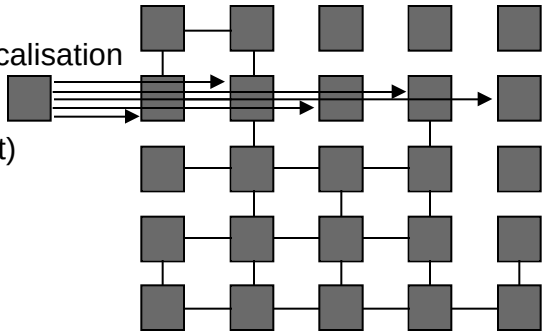
input
image
(environment)
basic agents
output
scene understanding
(global goals)



data driven
no explicit goal
no centralised representation
information exchange towards local coherence
decentralized system simulation

SATURNE Behaviour : Recognition Localisation

input
recognition, localisation
(global goals)
image
(environment)
output
basic agents



goal directed
explicit goal
purposive, centralised representation
information exchange towards global coherence
distributed problem solving

The COHIA (or KR x KP) Approach

Structuring the knowledge representation
- criteria : abstraction and decentration
- horizontal decoupling levels of representation
- vertical first-hand **interactions** : perception

Structuring the knowledge processing
- criteria : foci on space, time, features, models, tasks
- vertical decoupling into foci of attention
- horizontal second-hand **interactions** : communication

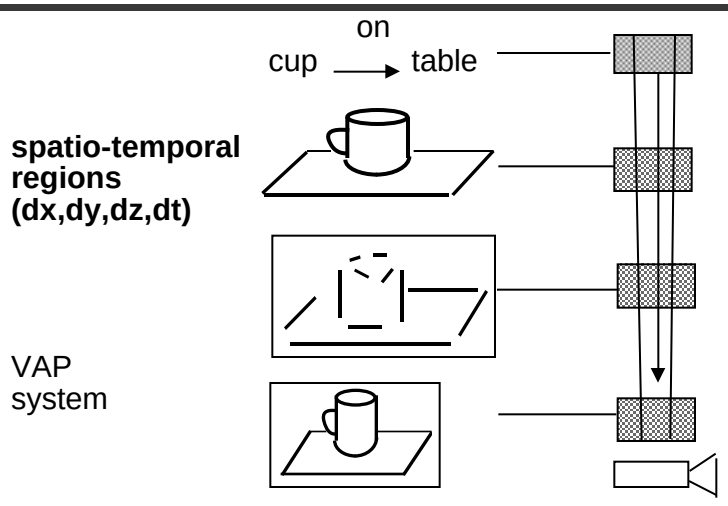
Identifying the basic entities of the system
- definition : intersection of level-agents & focus-agents
- choices : **agents**, **organisation**, **environment** models

Identifying the behaviour of the system
- System simulation : driven by the nature of the agents
- Problem solving : guided by the goals of the society

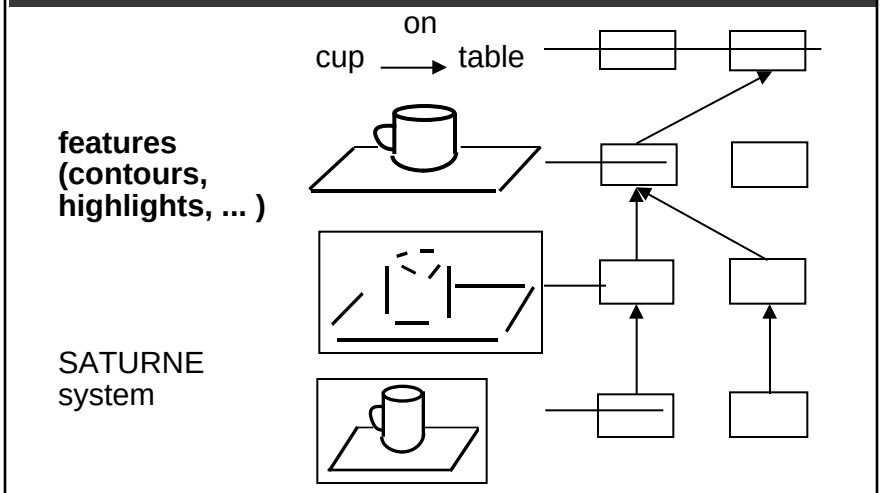
Decomposition by Abstraction [Demazeau 86]

	scene	scene	interpretation
3D model	object	feature grouping	recognized objects
2,5D model	scene features	scene description	scene elements
primal sketch	image features	image description	image elements
images	images	images	raw data
[Marr]	[Demazeau]	[Crowley]	[Neuman]

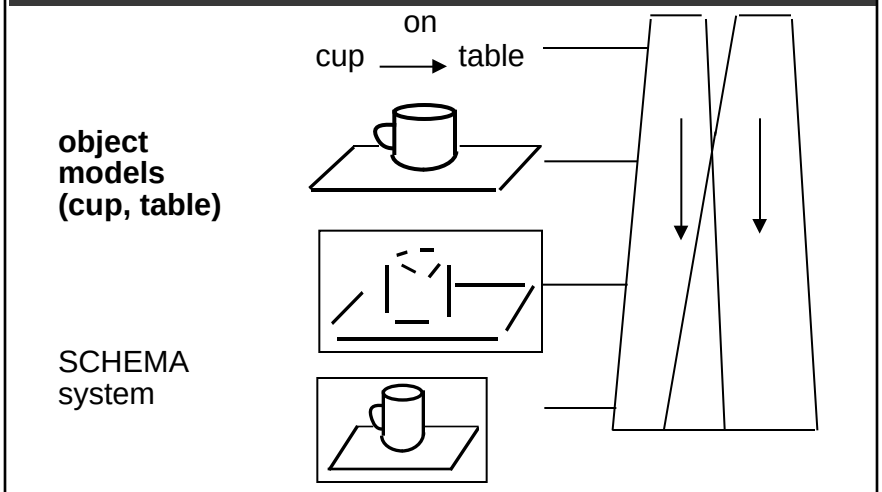
Decomposition by Location [Crowley 89]



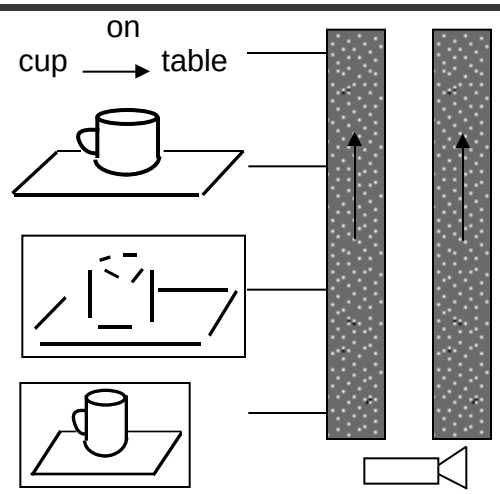
Decomposition by Input [Demazeau 86]



Decomposition by Output [Draper 83]



Decomposition by Tasks [Aloimonos 90]



Extrinsic Decomposition [Alvares 96]

Characteristics

- each agent is able to solve the whole problem
- the use of many agents in parallel speeds up the problem solving
- it is a purely physical (spatial or temporal) decomposition of the work between the agents

Examples

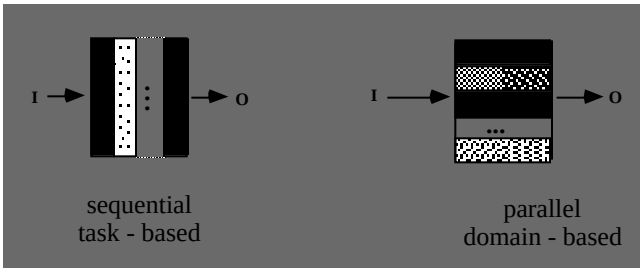
- there is an examination to be prepared by several professors. Each one wil be responsible to prepare a given number of questions (spatial)
- each professor will work for a given time (temporal)

Intrinsic Decomposition [Alvares 96]

The decomposition is based on a specialization

Two possible ways

- to solve the problem partially for any case
- to solve the problem entirely for some cases



Using many criteria (1) [Alvares 96]

The criteria are not mutually exclusive, we can combine them
At every level, the decomposition criteria are exclusive

Example: to prepare an examination subject

- Determine the number of questions and the respective value by topic (sequential)
- There will be people to prepare questions about topic t1 and people to prepare questions about topic t2 (parallel)
- In topic t1, there will be discursive and simple choice questions (parallel).
- There will be people to revise all questions (sequential)
- Each question will be revised for technical aspects and for linguistic aspects (parallel)

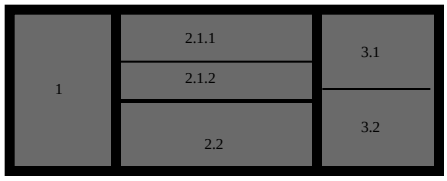
Using many criteria (2) [Alvares 96]

The problem is decomposed into :
- 1 determine topics 2 prepare questions 3 revise questions

The subproblem 2 is decomposed into
- 2.1 topic t1 2.2 topic t2.

The subproblem 2.1 is decomposed into
- 2.1.1 discursive questions 2.1.2 simple choice questions.

The subproblem 3 is decomposed into
- 3.1 technical review; 3.2 linguistic review.



Comparative Properties [Alvares 96]

	extrinsic	sequential task-bsd	parallel domain-bsd
ag's competence and behaviour	same	different	different
allowance of parallelism	yes	no	yes
allowance of ag's simplification	no	yes	yes
type of decomposition	quantitative	qualitative	qualitative
communication between agents	minimal	maximal	minimal

How MAS Methodology is specific ?

= Approach + Model + Tools + Problem + Domain
= Analysis + Design + Development + Deployment

It provides a new analysis and design approach

...

MAS Approach : Decomposing into Entities

A new approach to analyze and design SS

1. MAS are situated, and the real environment differs from the perceived environment
2. The methods are mainly process-centered, but non-only task-based
3. The methods involve both declarative and computational specifications
4. The control is mainly decentralized, highly modular, it is distributed among entities and partly in an emergence engine
5. The entry point of the design is not unique nor imposed, even usually focused on Agents first
6. VOWELS decomposes the MAS into A, E, I, O
7. ...

MODELS

MAGMA Models : Pluridisciplinarity

Mathematics
- Maths : Logics, Graphs and Trees
- Maths : Geometry, Topology
- Maths : Analysis
- Maths : Algebra

Physics
- Physics : Mechanics, Statistical Mechanics
- Physics : Automata, Control

Other Sciences
- H&S Sciences : Social Psychology, Sociology
- H&S Sciences : Philosophy
- H&S Sciences : Economy
- N&L Sciences : Ecosystems

Models : Agents and Environments

Agents

- Maths : Logics : COHIA, ASIC
- Maths : Graphs and Trees : SMAM
- Physics : Mechanics : PACO, PACO+
- Physics : Automata : SMARRPS
- Physics : Control : ASTRO
- H&S Sciences : Social Psychology

Environnements

- Maths : Geometry : SMARRPS, SIGMA, AGENT
- Maths : Topology : SMAM
- Physics : Mechanics + Maths : Geometry : PACO
- H&S Sciences : Social Psychology
- Natural Sciences : Ecosystems

Models : Interactions and Organisations

Interactions

- Maths : Logics + H&S Sciences : Philosophy : IL, IL2
- Maths : Graphs and Trees : IL Interaction Protocols
- Maths : Graphs and Trees : Dynamic Interaction Models
- Physics : Mechanics : PACO, SMARRPS
- H&S Sciences : Social Psych. + Philosophy : Dialogism

Organisations

- Maths : Logics : PACORG
- Maths : Analysis + H&S Sciences : Economy : Markets
- Maths : Graphs and Trees : SMAM
- Physics : Mechanics : SIGMA
- H&S Sciences : Social Psychology : Social Power

Models : Recursion and Emergence

Recursion
- Maths : Graphs and Trees

Emergence
- Physics : Mechanics : PACO, SMARRPS
- Physics : Statistical Mechanics : PHAMUS, SMAM
- Maths : Algebra + H&S Sc. : Sociology : ((A + I) +O) + E)
- H&S Sciences : Social Psychology : Social Power
- N&L Sciences + H&S Sciences

How MAS Methodology is specific ?

= Approach + Model + Tools + Problem + Domain
= Analysis + Design + Development + Deployment

It provides a new analysis and design approach

It is supported by existing formalisms,

...

MAS Models : Modelling these Entities

New models supported by existing formalisms

1. At higher abstraction level than other existing methods, closer to natural human way of thinking and reasoning about systems, not only devoted to computer scientists
2. It does not supply any new formalism currently, but entities are formalized using existing formalisms like traditional logics, Petri nets, algebraic languages, design patterns,...
3. VOWELS As range from reactive to cognitive
4. VOWELS Es range from spatial to topological
5. VOWELS Is range from forces to speech acts
6. VOWELS Os range from groups to markets
7. ...

DEVELOPMENT

Evolution of Programming Paradigms

1950's
- Machine and assembly language

1960's
- Procedural programming

1970's
- Structured programming

1980's
- Object-Based programming, Declarative programming

1990's
- Frameworks, design patterns, scenarios, and protocols

2000's
- Agents… Multi-Agent Systems…

…

Features of Languages and Paradigms

Concept	Proc. L.	Object L.	Agent L.
abstraction	type	class	society
building block	data	object	agent
computational model	procedure call	method message	perceive reason / act
design paradigm	tree of procedures	interaction patterns	cooperative interaction
architecture	functional decompos.	inheritance polymorph.	managers assistants,peers
modes of behavior	coding	designing and using	enabling and enacting
terminology	implement	engineer	activate

Agent Oriented Programming [Shoham 93]

A complete AOP system will inculde three primary components

- a restricted formal language with clear syntax and semantics for describing mental state: the mental state will be defined uniquely by several modalities, such as belief and commitment
- an interpreted programming language in which to define and program agents, with primitive commands such as REQUEST and INFORM: the semantics of the language will be required to be faithful to the semantics of the mental state
- an "agentifier", converting neutral devices into programmable agents.

AOP vs. OOP : introduction

- The use of mental states is to design the computational system
- From an engineering point of view, AOP can be viewed as a specialization of OOP, in the orginial sense of Hewitt's Actors model
- OOP proposes viewing a computational system as made up of modules that are able to communicate with one another and that have individual ways of handling incoming messages
- AOP specializes the framework by fixing the state (mental state) of the modules (agents) to consist of components such as beliefs (about the world, about themselves, about one another), capabilities, and decisions, each of which enjoys a precisely defined syntax.

Components of mental state

Components of mental state

- the future is determined by two factors : past history and current actions of agents.

The actions of an agent are determined by its decisions, or choices

Decisions are logically constrained, though not determined, by the agent's beliefs

- These beliefs refer to the state of the world, to the mental state of other agents, and to the capabilities of this and other agents

Basics : beliefs, capabilities, obligations (decision is simply an obligation to oneself)

A language for belief, obligation, and capabilities

time / action	$holding(robot,cup)^t$
belief	$B^t_a\varphi$ φ **a (recursive) sentence**
obligation	$OBL^t_{a,b}\varphi$
decision	$\equiv_{def} OBL^t_{a,a}\varphi$
capability	$CAN^t_a\varphi$
	$ABLE_a\varphi =_{def} CAN^{time(\varphi)}_a\varphi$

time(φ) the outermost time occurring on

Properties of the various components

internal consistency	$\forall a,t \{\varphi : B^t_a \varphi\}$ **is consistent** $\forall a,t \{\varphi : OBL^t_{a,b} \varphi\}$ **is consistent**
good faith	$\forall t,a,b,\varphi\ OBL^t_{a,b}\varphi \supset B^t_a((ABLE_a\varphi) \wedge \varphi)$
introspection	$\forall t,a,b,\varphi\ OBL^t_{a,b}\varphi \equiv B^t_a OBL^t_{a,b}\varphi$ $\forall t,a,b,\varphi \neg OBL^t_{a,b}\varphi \equiv B^t_a \neg OBL^t_{a,b}\varphi$
persistence	**so are mental state, obligations**
capability	**capabilities do not fluctuate widely**

Agent0 : basic loop

The role of agent programs is to control the evolution of an agent's mental state

- actions occur as a side-effect of the agent's being committed to an action whose time has come

Each agent iterates the following two steps at regular intervals

- read current messages, and update your mental state, including your beliefs and commitments (the agent program is crucial for this update)
- execute the commitments for the current time, possibly resulting in further belief change (this phase is independent of the agent's program)

Interaction Oriented Programming [Huhns 96]

Motivations

- errors will always be in complex systems;
- Error-free code can be a disadvantage;
- Where systems interact with the real world, there is a power that can be exploited

Example : children forming a circle

- conventional approach: create a C++ class for each type of object, write a control program that uses trigonometry to compute the location of each object
- interaction-oriented approach: children approach is robust due to local intelligence and autonomy, write the program based on objects having attitudes, goals, agent models

IOP : Active modules, declarative specification, modules that volunteer, modules holdbelief about the world, especially about themselves and others

Organisation Oriented Programming [Lemaitre 98]

Designing, Maintaining, Using MAS utilize different integrative frameworks that include features to deal with agents, interactions, environments, ... MAS programming itself follows history of programming.

The most well-known effort towards MAOP is AOP [Shoham 93] ... IOP [Huhns 97] is an alternative...

OOP is another one [Lemaitre 98] ... EOP does not actually exist as a trend but looks like Artificial Life.

These approach respectively focus on Agents, on Interactions, on Organisations, on Environments, as being the respective basic bricks at the disposal of the designer / MAS / user...

(((A + I) + O) + E) O.Programming [Demazeau 96]

The Population structure is the set of agents, the set of possible behaviors of the agents, and the set of all interaction processes between agents

Pop = (Ag, Bh, Ip; bc, ic)

Ag : set of agents
Bh : set of behaviors agents are able to perform
Ip : set of interaction processes
bc : Ag ---> P(Bh), behavioral capability,
bc(a), set of behavior a is able to perform
ic : Ag x Ag ---> P(Ip), interaction capability,
ic(a1,a2), set of interaction processes agents a1 and a2 may perform together

(((A + I) + O) + E) OP : Organisations

The Organization structure is composed of organizational roles and organizational links

Org = (Ro; Li)

Ro is defined in a relational way

- e.g. Ro Lp x Gp : global processes (Gp) and local processes (Lp), the role is the part of agent's behavior that is integrated in the global process.
- e.g. Ro Fo x Lv : foci of interest (Fo), representation levels (Lv), the role is the agent's behavior for a given focus at a given level.

Li $\subseteq$ Ro x Ro

(((A + I) + O) + E) OP : Pop ℜ Org = imp

The suitable relation between the organization and the population of a system is called the system's organization implementation
It is any relation imp = Pop ℜ Org, on (Ro x Ag) ∪ (Li x Ip), Pop = (Ag, Bh, Ip; bc, ic), Org = (Ro; Li).

- if (r,a) imp, r is said to implemented by a
- if (l,p) imp, l is said to implemented by p

imp is said "proper" iff ℜ is an homomorphism.

- r Ro, a Ag / (r,a) imp, and r is properly implemented by some behavior b bc(a)
- l = (l1,l2) Li, ip Ip / { (l,ip) imp (a1,a2) Ag x Ag / ip ic(a1,a2), (r1,a1) imp, (r2,a2) imp, and r1, r2 are properly implemented by the behaviors of a1 and a2, respectively }

(((A + I) + O) + E) OP : PopOrgs

The Interior (= Population + Organisation) of a time-invariant multi-agent system is captured by a population-organization structure PopOrg = (Pop, Org; imp), where

- Pop = (Ag, Bh, Ip; bc, ic) is a population structure
- Org = (Ro; Li) is a organization structure
- imp (Ro x Ag) (Li x Ip) is an organization implementation relation as defined previously

Multi-Agent Oriented Programming

Not Object-Oriented Programming
- S = Objects + Message passing

Not Logic nor Expert Systems Programming
- S = Knowledge + Inference Mechanism

Not Ontology-Oriented Programming
- S = Knowledge + Problem Solving Methods

But Agent-Oriented Programming
- S = BDI Agents + KQML (Interactions)

But (((A + I) + O) + E)-Oriented Programming
- S = ((A + I) + O) + E)

But VOWELS Programming
- S = [A*; E*; I*; O*] + (Recursion & Emergence) Mechanism

How MAS Methodology is specific ?

= Approach + Model + Tools + Problem + Domain
= Analysis + Design + Development + Deployment

It provides a new analysis and design approach

It is supported by existing formalisms,

It integrates existing programming paradigms,

...

MAS Tools : Developing these Entities

New tools integrating existing paradigms

1. MAS is not (yet?) an implementation model and MAS oriented tools are usually not specific
2. Agents themselves just begin to have their own languages
3. MAS Development relies on existing languages and programming paradigms
4. The trend of the work is towards Multi-Agent Oriented Programming, meaning programming MAS with MAS tools
5. The closest related tools for VOWELS seems be frameworks but are still under investigation from the computational point of view
6. ...

DEPLOYMENT

Object-Based (Concurrent) Languages

Actors-like Languages

- Massive Parallelism
- Every interaction is made by passing of asynchronous buffered messages
- Use of the local continuity
- Object-oriented design
- Full distributed control

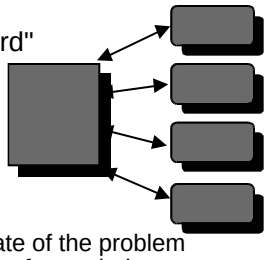
Actors [Hewitt]

- An actor is composed of acquaintances, scripts, ...
- A message is an actor containing method, continuation, and complaint actors
- A method is a set of actions : create a new actor, send a message, modify actor's behaviour

(Distributed) Blackboard Systems

BB-Like frameworks

- Interactions through the "blackboard"
- Opportunistic activation of the knowledge sources
- Initially Centralized Control



DBB

- Blackboard
  - ✓ A global database representing state of the problem
  - ✓ Hierarchically organised into levels of granularity
- Knowledge Sources
  - ✓ Produce changes (hypotheses) onto given levels of BB
  - ✓ Contain particular subset of domain knowledge
- Centralized Control
  - ✓ Evaluates the status of the problem
  - ✓ Controls the knowledge sources

Integrative Environments

Language for constructing agents
- Different agent architectures
- Langage for knowledge representation
- Mechanisms for reasoning, deciding, controlling

Representing and dealing with the environment
- Representation of the environment and its evolution
- Implementing the perception of the environment
- Implementing the actions of the agents in the environment

Representing and dealing with other agents
- Primitives, protocols, message processing
- Implementing communication between agents

Development Interface
- Visualisation, trace, inputs-outputs

Application Interface

Interface with Target (Distributed or not) System

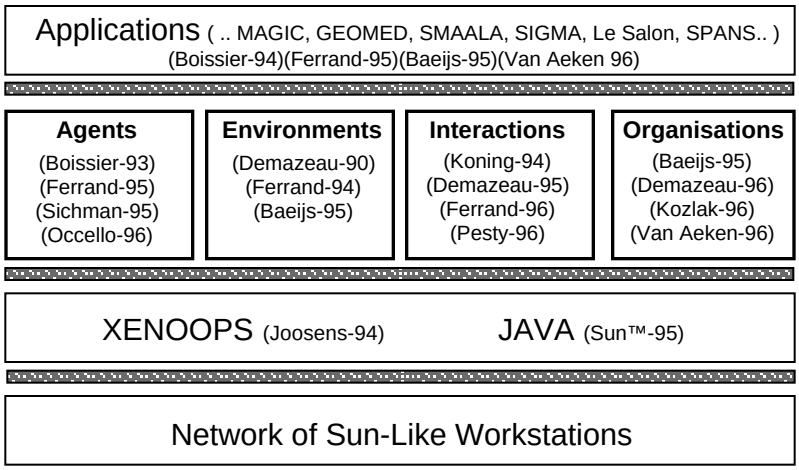
PACT [Cutkosky]

Palo Alto Collaborative Testbed

Computer-Aided Concurrent Engineering
- Stanford, Lockheed, Hewlett-Packard, Enterprise Integration Technologies
- Have integrated four pre-existing CE systems into common distributed framework supporting aspects of small robot

Uses quasi-standards
- Knowledge Interchange Format (KIF) [DARPA-92]
- Knowledge Query & Manipulation Language (KQML) [Finin-92]
- Product Data Exchange Specification (PDES)

MASK [Occello]



How MAS Methodology is specific ?

= Approach + Model + Tools + Problem + Domain
= Analysis + Design + Development + Deployment

It provides a new analysis and design approach

It is supported by existing formalisms,

It integrates existing programming paradigms,

It is striving towards industrial quality,

…

Todays offer

Academics

- Firefly (MIT before Microsoft) (no more accesible)
- MadKit (LIRMM Montpellier - Ferber's group)
- Simula (II Porto Alegre - Alvares's group)
- dMARS (-> Jack, by Agent Oriented Software)
- *From 2000 :* Volcano...

Industrials

- Voyager (ObjectSpace) - freeware (linked with OMG)
- JINI (Sun) - freeware
- Aglets (IBM) - freeware
- Javabeans (Sun) - freeware (based on components)
- Agentbuilder (Reticular) - freeware + product (AOP based)
- ZEUS (BT) - freeware product (FIPA compliant)
- *From 2000 :* JADE, Jack...

Qualification criteria

Four *qualities* for each stages:

- Completeness: quantity & quality
- Applicability: scope, restrictions
- Complexity: competence required, workload
- Reusability: reuse of previous work

16 criteria + availability & support

	Analysis	Design	Development	Deployment
Completeness				
Applicability				
Complexity				
Reusability				

Selected platforms

Platforms requirements :

- based on a strong academic model
- high quality software, well maintained
- cover as many aspects as possible of MAS
- cover the four methodological stages

AgentBuilder, Jack, Madkit, Zeus

- As of first semester 2000

Jack™

Developed by Agent Oriented Software Pty.

Including the dMARS BDI model
Great versatility
Focus on the development stage

	Analysis	Design	Development	Deployment
Completeness	none	ident. of classes	extended Java	manual
Applicability	n / a	Jack BDI A	Any MAS	n / a
Complexity	n / a	Jack BDI A	Java & Logic P.	n / a
Reusability	n / a	difficult	classes	n / a

MadKit

Developed by O. Gutknecht & J. Ferber, LIRMM

Based on the AALAADIN organisational model
Graphical multi-agent runtime engine
Good versatility
Light methodology, no BDI

	Analysis	Design	Development	Deployment
Completeness	none	Aalaadin	Pure Java	G-Box
Applicability	n / a	broad range	simple A	small large MAS
Complexity	n / a	intuitive	few code base	GUI
Reusability	n / a	design patterns	classes	dynamic reconf.

AgentBuilder®

Developed by Reticular Systems Inc.

Grounded on Agent0/Placa BDI architecture
Almost all stages covered
Complete graphical tools
Limited to a single agent model

	Analysis	Design	Development	Deployment
Completeness	ontology	A definition	behavoural rules	RT A engine
Applicability	universal	cognitive A	AgentBuild. BDI	small societies
Complexity	OO GUI	MAS design GUI	Logic P. GUI	GUI
Reusability	ontology	protocols	A	none

Zeus

Developed by British Telecom

All stages covered, from analysis to deployment
Methodological and Software tools
Limited to a single agent model

	Analysis	Design	Development	Deployment
Completeness	role modelling	finding solutions	5 activities	tools docs
Applicability	role o. MAS	task o. A	Zeus A model	debug visualis.
Complexity	UML	design skills	GUI tools	GUI
Reusability	role models	reusable formal.	partial	A reconf.

Pitfalls of current MAS offers

Completeness
- Much on development… nothing about analysis/design
- Much focus on approach… but poor technical aspects
- Nothing about deployment

→ Every stage must be developed in the platform !

Applicability
- An agent platform…but not a multi-agent platform
- A generalisation of a specific multi-agent system …multi-domain, but single-problem platform
- Fixed models, and no way to escape

→ The platform must be as versatile as possible !

Complexity
- The documentation is sparse
- You have to code a lot
- The user interface is unfriendly

→ Understanding, (re)using the platform must be facilitated !

Volcano

Developed by PM. Ricordel & Y. Demazeau, LEIBNIZ

A multi-agent platform to fulfil all these criteria

- Based on the AEIO MAS decomposition [Demazeau]
- Full analysis-to-deployment chain
  - ✓ Problem/domain decomposition
  - ✓ AEIO modelling
  - ✓ Open library of models (simplicity, versatility, reusability)
  - ✓ Intelligent deployment tools

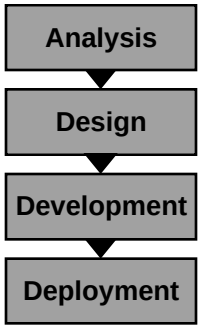
But

- Still under development…
- Looking for partners
- To be fully evaluated...

METHODOLOGY

MAS Methodology

Methodology
= Approach + Model + Tools + Problem + Domain
= Analysis + Design + Development + Deployment



Identify the problem and the domain

Get rid of the domain / Define the solution

Implement the solution / Plug the domain

Apply the solution to the problem/domain

MAS methods vs. Systemic methods

Systemic Methods meaning...
- Information Systems

Characteristics of the Systemic Methodology
- data-centered
- centralized
- almost not modular

Characteristics of the MAS Methodology
- mainly process-centered
- decentralized
- highly modular

MAS methods vs. Formal methods

Formal (Specification) Methods meaning...

- Logics, Algebraic languages like Z, Automatas, Petri Nets, ...

Characteristics of the FS Methodology

- mainly used for validation
- include automatic generation

Characteristics of the MAS Methodology

- very low supported by a dedicated formal framework, but...
- ... possible use of existing formalisms to specify MAS components
  - ✓ logics-based approach [Fischer 94], [Huntbach 95], ...
  - ✓ Z, algebraic language approach [Luck 95], ...
  - ✓ Petri Nets approach [Elfallah 96], ...

MAS methods vs. Knowledge methods

Knowledge (Representation) Methods meaning...

- KADS, CML, KSM [Molina 95]...

Characteristics of the KR Methodology

- mainly declarative specifications
- control lays in the system inference engine

Characteristics of the MAS Methodology

- both declarative and computational specifications [Glaser 96], ...
- control lays in processing units and an emergence engine
  - ✓ (agent) control lays in the processing units [Occello 97], ...
  - ✓ (MAS) control lays in the system emergence engine, this engine involves the processing units with a recursion principle, whichever they are agents, environments, interactions, organisations [Demazeau 95], ...

MAS methods vs. Functional methods

Functional Methods meaning...
- SART, ...

Characteristics of the Functional Methodology
- task-based
- hierarchical
- decision as automata
- global context

Characteristics of the MAS Methodology
- non-only task-based [Alvares 97], ...
- hierarchical and possibly recursive [Occello 97], ...
- reactive and cognitive decision [Brazier 95], [Jonker 98], ...
- global and local contexts [Drogoul 98], ...

MAS methods vs. Object methods (start)

Object Methods meaning...
- OO analysis and design, modelling, implementation

Characteristics of the Object Methodology
- continuity Approach / Modelling / Implementation
- ...

Characteristics of the MAS Methodology
- no full continuity Approach / Modelling / Implementation
  - ✓ MAS is not (yet?) an implementation model
  - ✓ Agents just begin to have their own languages [Shoham 93], [Thomas 95], ... but the programming is not always based on Agents [Demazeau 97]
  - ✓ MAS design is based on existing languages and programming paradigms [Poggi 94], ...
  - ✓ towards multi-agent oriented programming [Demazeau 97], ...
- ...

MAS methods vs. Object methods (cont'd)

Characteristics of the Object Methodology

- object classes
- inheritance mechanism
- no organisation nor group primitives
- objects are built first, and then their dynamics
- ...

Characteristics of the MAS Methodology

- Agents, Environments, Interactions, Organizations [Demazeau 95], ...
- component groups, recursive mechanism [Fisher 94], [Kinny 96], [Occello 97], ...
- organisation and group primitives [Occello 97], ...
- entry point of the design is not unique nor imposed [Demazeau 97], ... even it often corresponds to agents
- ...

MAS methods vs. Object methods (end)

Characteristics of the Object Methodology

- environnement of an object does not exist, even if the environment of an object system does
- fixed Data Interaction Model
- global control, RPC mechanism,

Characteristics of the MAS Methodology

- MAS are situated, the real environment differs from the perceived environment [Moulin 95], [Kendall 95], ...
- free Data interaction Model [Demazeau 95], ...
- global (protocols) [Demazeau 95], [Koning 98], ... and local control (agent's decision) [Shoham 93], [Kendall 95], ...

MAS methods vs. Components methods (start)

Components Methods meaning...

- Components meaning JavaBeans, MS-COM, ...

Characteristics of the Components Methodology

- continuity Approach / Modelling / Implementation
- fixed Data Interaction Model between components
- no organisation nor group primitives
- components are built first, and then their dynamics

Characteristics of the MAS Methodology

- no full continuity Approach / Modelling / Implementation
- free Data interaction Model [Demazeau 95], ...
- organisation and group primitives [Occello 97], ...
- entry point of the design is not unique nor imposed [Demazeau 97], ... even it often corresponds to agents

MAS methods vs. Components methods (end)

Some common features between the methods

- introspection, persistence, mobility of basic entities
- event-driven communication between entities
- entities design and integration into applications

Characteristics of the Components Methodology

- customisation of entities at design time only
- existing de facto standards towards interoperability
- application independent reusable interoperable entities

Characteristics of the MAS Methodology

- possible dynamic allocation of roles during run time
- efforts to standardisation through the FIPA foundation
- still frequently application dependent entities

How MAS Methodology is different ? (start)

An enriched process-centered, decentralized, highly modular information system methodology

A currently poorly formalized formal specification methodology, reusing existing formalisms

An enriched knowledge representation methodology with computational specifications, a decentralized control and an emergence engine

An enriched functional methodology, not-only task-based, with possible recursion, cognitive decision, and local contexts

...

How MAS Methodology is different ? (end)

An enriched but incomplete object methodology

- **with extended classes (A, E, I, O), groups, organizations, recursive mechanism, and where the design is not always based on agents,**
- **with situated agents, free interactions, local control,**
- **where the programming is not always based on agents, but where no full continuity Analysis / Design / Implementation is not yet acheived**

An close component methodology, more flexible but still to be standardized

An enriched UML methodology which is not restricted to the design of systems

Existing MAS methods and current work

Australia
- Kendall (Melbourne), Kinny (Melbourne), …

France
- Barthes (Compiegne), Cazalens (Nantes), Demazeau (Grenoble), Ferber (Montpellier), Glize (Toulouse), ...

Netherlands
- Treur (Amsterdam), ...

Spain
- Garcia (UPM), ...

UK
- Jennings (Southampton), Wooldridge (Liverpool), …

USA
- Shoham (Stanford), ...

...

CASSIOPEE : General Issues

From the Analysis of natural organisations to the Design of artificial organisations

Based on several applications and experiments

Three Abstraction Levels
- individual agents, interactions, organizations

Agents is defined as a set of Roles
- individual roles, interactional roles, organizational roles

Lacks of Models and Tools

CASSIOPEE : Abstraction Levels

Agents
- Which architecture to choose to implement the agents ?
- Which scope of knowledge and how to best use it ?
- Which competences and how are they distributed ?

Interactions
- How do agents communicate ?
- Which content ?
- Can agents influence / alterate other's behaviour ?

Organisations
- How do the agents cooperate ?
- Is there a global goal, how to build a plan to reach it ?
- Which structure to organize, which evolution of the structure ?

CASSIOPEE : Composing Roles

Domain & Problem Dependent Typology of Roles
Individual Roles
Getting abstracted from the Domain by Resource / Functional Dependence (conflicts, permits, facilitates, needs, ...)
Problem based Typology of Relational Roles
Interactional roles (influencing, influenced)
Getting abstracted from the Problem by Identification of Potential Groups (SIGs, ...)
Typology of Organizational Roles
Organizational roles (initiator, participant)

DESIRE : General issues

Design and Specification
- Complex reasoning systems in general
- Proposes a powerfull design tool
- A design approach more than an analysis approach

A Formal Framework
- Formal specifications to automatically generate a prototype

Interacting Components based
- Input/output components

Reflective
- reasoning
- architecture

DESIRE : A Specification Framework

Components Decomposition
- Components Hierarchy
- Primitive and composed components

Information Exchange between Components
- Information links for information flows (channels)
- different levels of dynamic interaction models

Sequencing of tasks
- a local control process in each component
  - ✓ rules set (facts)
  - ✓ required data (required interactions)

Hierarchical knowledge structures
- adapted to components granularity

DESIRE : Modeling Agents

Models

- Agents as composed components
- Modeling of specific types of Information Exchange
  - ✓ more communication than interaction
  - ✓ MAS interaction = components interaction
  - ✓ interaction is embedded in components

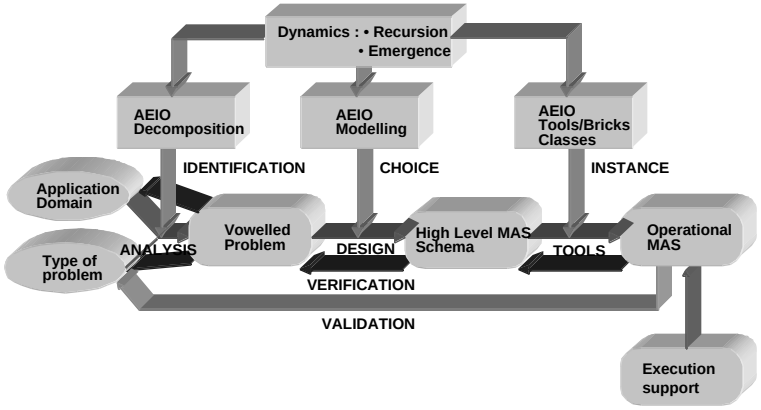
Approach

- A task based approach (functional)
  - ✓ no explicit AEIO models

Design

- An agent centered approach
  - ✓ no external expression of interaction
  - ✓ no external expression of organisation

VOWELS : Methodology



VOWELS : Analysis

(((A + I) + O) + E)	**Social Science**
(((A + E) + I) + O)	**Robotics Science**
(((E + A) + I) + O)	**Life Science**
(((I + O) + A) + E)	**Military Science**
(((O + I) + E) + A)	**Economic Science**

...

Historical ? Biased ? Grounded ?

VOWELS : Design

Agents
- internal architectures of the processing entities

Environment
- domain-dependent elements for structuring external interactions between entities

Interactions
- elements for structuring internal interactions between agents

Organizations
- elements for structuring sets of agents within the multi-agent system

These definitions are agent-centered but alternatives exist, firstly based on environments, on interactions, or on organisations.

VOWELS : Development

The Declarative Principle
MAS = A + E + I + O

The Functional Principle
Function(MAS) = $\sum$ Function(A)
+ Emergence Function

The Recursive Principle
A = basic A | MAS

These principles are agent-centered but alternatives exist : any recursion dealing with A, E, I, O or any combination of the four basic elements

VOWELS : Deployment

MASK - Platform
- MASK Design, Agent (API, ASTRO,SIGMA)
- Interaction (API, IL, Protocols), Organisation (RESO)
- Integration and Illustrations : Conveyors, Factorial, Appointment service

VOLCANO - Platform
- VOLCANO Design
- A E I O components and interfaces
- Integration
- Illustrations : Robocup, Appointment service, SMAMS

VOWELS : Usage

MAOP subsumes AOP, IOP, OOP-like OP...

We defend an instance of MAOP, the VOWELS framework in which :

- 1/ to express the problem to solve independently of the domain
- 2/ to "vowellify" the problem in terms of A E I O, ...
- 3/ to choose understood frames of A, E, I, O, dynamics, and recursion
- 4/ to leave VOWELS "emergence engine" complete the missing bricks by itself and build the appropriate MAS...
- 5/ ... to be deployed as self on a distributed settling...
- 6/ ... to be settled and used interactively

APPLICATIONS

SIGMA (academic project)

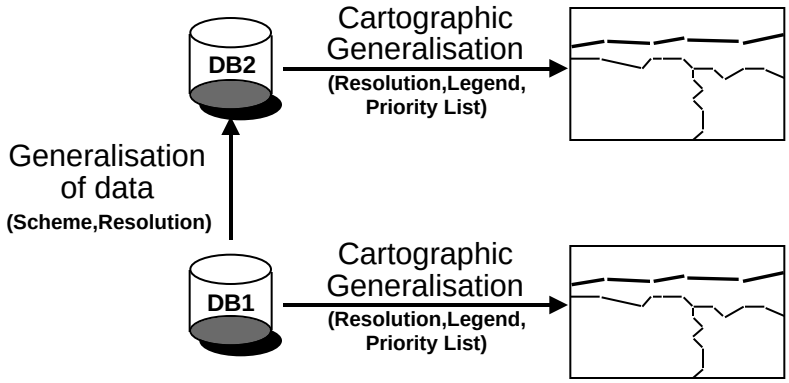
A reactive multi-agent approach to cartographic generalization LIFIA-INPG (F), IGN (F)

Interaction and organisation modelling to study their reciprocal interdependencies

Approach

- following the PACO approach (multiple types + organizational knowledge)
- reaching the relative importance of data types according to a desired global goal
- operators to transforms the representations of the data and the possible changes of scale
- interactive validation
- Implementation on C/C++ on Sun WS - LAN/XENOOPS

SIGMA : Types of Generalisation



SIGMA : Principles

Partial automatizing of cartographic generalization

- Creation of a readable and useful cartographic map from a geographical database given the aim of the map (pre-order) and using a non-holostic approach
- Modelling agents, interactions and organizational structures, and studying the convergence effects

Extension of the PACO paradigm

- Geographical objects are represented by a collection of "geographical entities" which "may" become agents
- Introduction of organizational knowledge to study their impact on a local level (behaviour of the agents) as well as on a global level (convergence of the system)

SIGMA : Model : E and A

Environment

- Geographical entities placed on a 2D grid, initially corresponding to the raw data (World of Reference)
- Active work on a copy (Active World) of the initial world to offer the opportunity to later geographical verification mechanisms

Agents

- A geographical entity becomes an agent as soon as its position in the organization (its mass) is important enough with respect to the aim of the map
- Each agent possesses several self-controled scopes:
  - ✓ Perception (local environment)
  - ✓ Communication (class, object, proximity, groups)
  - ✓ Action (class, object, proximity, groups)

SIGMA : Model : I and O

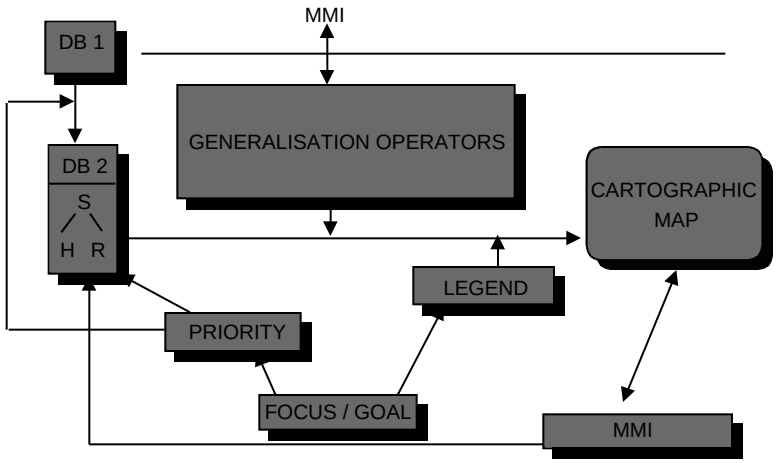
Interactions

- Between artificial agents (or objective groups)
  - ✓ Repulsion Force
  - ✓ Proportional Following (against local deformation of objects)
  - ✓ Unconditional Following (agents "sticking" together)
  - ✓ Change of symbolization
- Between the user and the agents (or subjective groups)
  - ✓ Change of symbolization
  - ✓ Formation or breaking of topological structures
  - ✓ Displacement of agents

Organizations

- Pre-orders, figuring "power"- relationship between geographical classes
- Groups, consisting of agents sharing the same local environment to realize a common task

SIGMA : The Architecture of the System



SIGMA : Implementation and Results

Implementation

- Full implementation in C++ under Unix with acceptable results

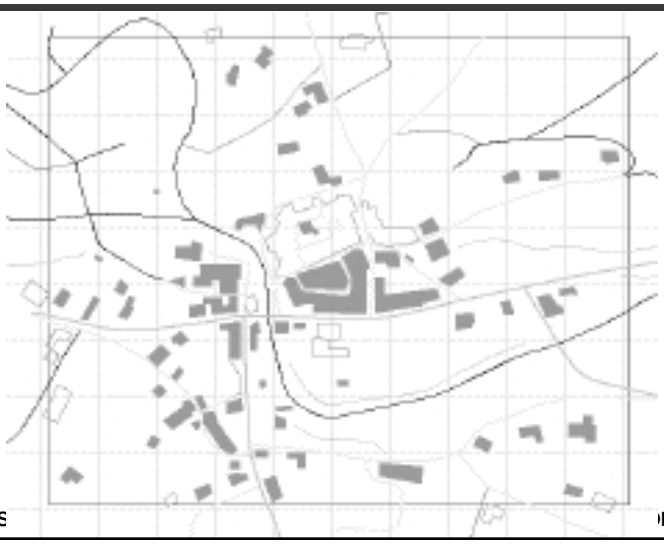
Results

- "Les Matelles": • 300 objects -> 1800 geo. entities acceptable results in quality and computation
- "Neighbourhood of Les Matelles": • 2000 objects -> 15000 geo. entities • acceptable results in quality

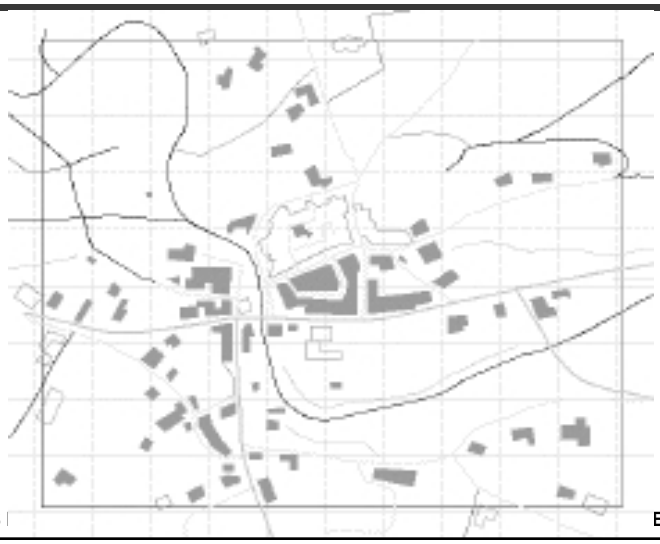
Perspectives

- Full explicitation of the organizational issues in the system
- Distribution/Parallellization of the system using XENOOPS
- Followup within the CEC-IT-LTR AGENT project

SIGMA : Les Matelles : initial map



SIGMA : Les Matelles : final map



CNRS EMAZEAU

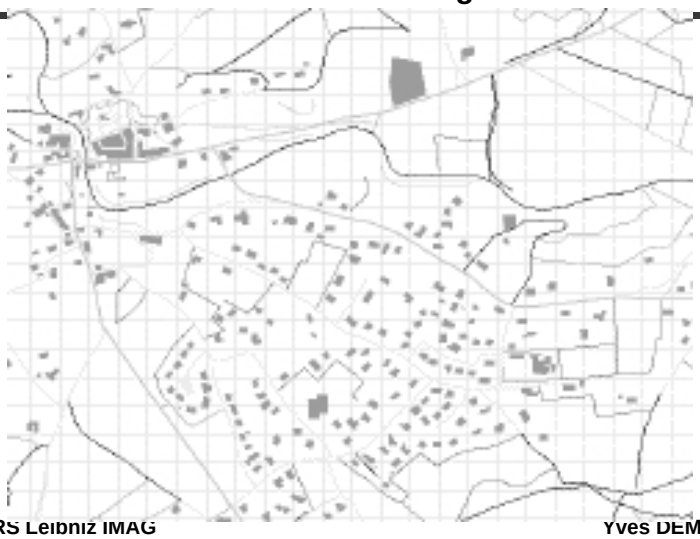
SIGMA : Les Matelles : Details

Before

After

CNRS Leibniz IMAG Yves DEMAZEAU

SIGMA : Les Matelles Surroundings : initial

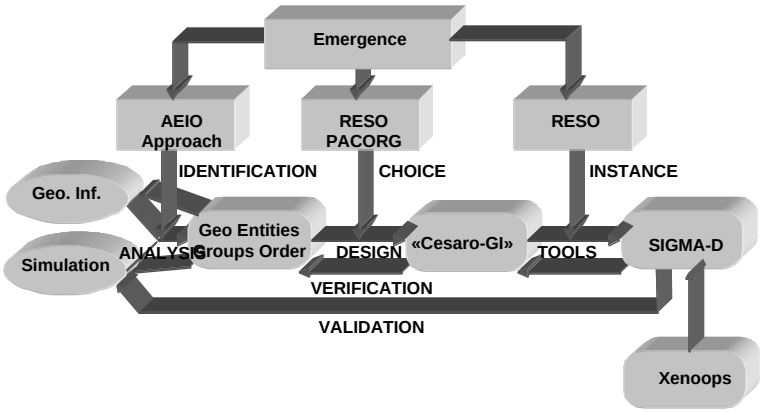


CNRS Leibniz IMAG Yves DEMAZEAU

SIGMA: Les Matelles Surroundings : final

CNR MAZEAU

VOWELS : SIGMA-D



AGENT (EU project - IT-LTR 24939)

Automatic GEneralisation New Technology IGN (F), LaserScan Ltd. (UK), LEIBNIZ-INPG (F), U. Zürich (CH), U. Edinburgh (UK)

Automated generalisation to provide maps from cartographic databases

Approach

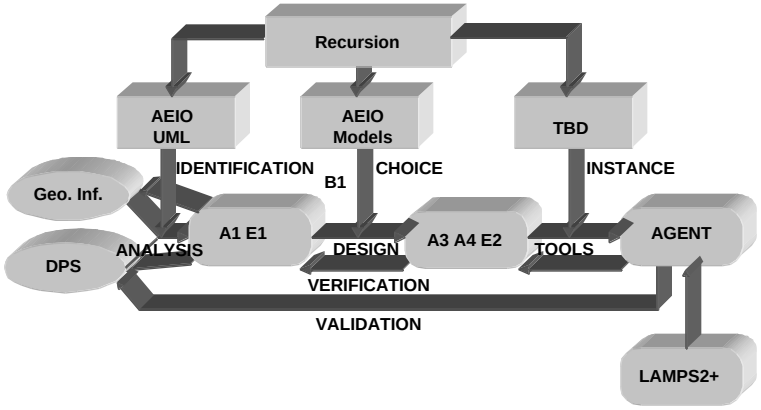
- COHIA agent architecture, micro agents (independent generalisation), meso agents (contextual generalisation), macro agents : an heterogeneous system
- Simple IL interaction mechanisms but sophisticated generalisation operators
- Recursive organisations between agents
- Full implementation on GOTHIC/LAMPS2 - Sun WS and PC - LAN & WWW - Commercialized

AGENT : Results

AVANT

APRES

VOWELS : AGENT



TOOLS

MASK - Platform
- MASK Design, Agent (API, ASTRO,SIGMA)
- Interaction (API, IL, Protocols), Organisation (RESO)
- Integration and Illustrations : Conveyors, Factorial, Appointment service

MAOP - Programming
- Parallelisation
- (((A + I) + O) + E) Programming
- VOWELS

VOLCANO - Platform
- VOLCANO Design
- A E I O components and interfaces
- Integration
- Illustrations : Robocup, Appointment service, SMAMS

VOWELS Past Applications

Learning to Walk : Freddy Walker (aca. VUB)
Image Feature Tracking : PACOVISION (ind. EAP)
Scene Understanding : MAGIC (aca. PRC-CHM)
Understanding Written French : TALISMAN (EU IT)
Und. Written Portuguese : NALAMAS (aca. CNPq)
Linear Planning : SMAALA & SANPA (ind. CERREP)
Negotiation : Le Salon & GEOMED (EU Telematics)
Cartographic Generalisation : SIGMA (ind. IGN)
Socializing the WWW : Friends (ind. FT)
Resource Management : Fishbanks (aca.CIRAD)
Cartographic Generalisation : AGENT (EU ESPRIT)
Autonomous Robots : Robocup (aca. project)

Which applications are better handled by MAS ?

MAS methods cater for distributed intelligence applications : Network based, Human involved, Physically distributed, Decentralized controlled, ...

It suits when only local computational models are available whilst global ones are unknown

- Telecommunications, Internet Applications, Vision, NLP, ...

It is adequate for application domains and kinds of problem as soon as non-provability is acceptable

- Vision, Robotics, NLP, GIS, Societies Simulation, ...

It suits when the human is involved in the life cycle of a distributed system

- Internet Applications, Groupware, CSCW, GIS, ...

How MAS Methodology is specific ?

= Approach + Model + Tools + Problem + Domain
= Analysis + Design + Development + Deployment

It provides a new analysis and design approach

It is supported by existing formalisms,

It integrates existing programming paradigms,

It is striving towards industrial quality,

It caters for distributed intelligence applications,

CONCLUSION

Issues driving to the use of MAS

Availability of supporting technologies
- network Capacity 78%
- processor performance 48%
- software Language and Tool Power 11%

Inherent distribution
- physical
- organizational

System openess
- changing system structure
- uncertain environment

Competitive collaboration
- multiple knowledge domains
- multiple solution methods

Natural or social systems modelling

...

Issues refraining to the use of MAS

Social acceptability of MAS applications
- degree of delegation (trust by users)
- degree of autonomy (responsability of owners)

Important properties cannot be guaranteed
- deadlock avoidance
- convergent negotiation

Impossibility to prove system behaviour
- prediction of system behaviour
- validation of system behaviour

Transfer from research to industry
- first to market has greater impact than best technology
- theoretical / formal MAS research have little impact

...

How MAS Methodology is specific ?

= Approach + Model + Tools + Problem + Domain
= Analysis + Design + Development + Deployment

It provides a new analysis and design approach

It is supported by existing formalisms,

It integrates existing programming paradigms,

It is striving towards industrial quality,

It caters for distributed intelligence applications,

It will always imply difficulties in provability.

MAS Principles

MAS generates revenue
- New applications viable
- Reduces risk of building the applications
- Encourages reusability

Widespread adoption of MAS requires
- Methodology
- Industrial strength toolkits
- Standards

Deployment of lead applications requires
- Use of simple, well understood techniques
- Focus on application value NOT technology
- Industrial partnership

MAS Research issues

The problem lays in the relations between mental issues and coordination theories, between micro and macro issues.
- Mutual representations
- Coordination models
- Organisations
- Methodologies

Multi-agent systems are in the near future what object oriented systems are today: a set of well defined techniques
- Multi-Agent Oriented Programming
- Testbeds and Benchmarks
- Standards
- Available industrial platforms

Evolution of Agents and Multi-Agent Systems

Robotics Agents
Mobile Agents
Software Agents
Interface Agents
WWW Agents
…

Artificial Intelligence
Telecommunications
Software Engineering
HC Interfaces
Internet Computing

MAS assuming Closed Environments
MAS integrating Open Environments
MAS including Human Agents (CSCW, ITS)
…

THE MAGMA GROUP

MAGMA : Project in 00-01

Main Axes (Stream 98-02)
- **Development of the MAGMA Method**
- **Interactions (Approach - Models - Tools)**

Approach
- **A E I O, Emergence**, Recursion
- Domains, **Problems**

Models
- **Agents,** Environments**, Interactions, Organisations**
- **Emergence**, Recursion

Tools
- Programming, **Platform**

Applications
- Information Systems
- **Mediation Systems**
- **Autonomous Systems**

MAGMA : Applications in 00-01

Information Systems
- **Cartographic Generalisation**
  - ✓ AGENT (EU ESPRIT)
- Spatio-Temporal Reasoning

Mediation Systems
- **Distant Learning**
  - ✓ Baghera (academic IMAG)
- **Virtual Reality**
  - ✓ Deuxième Monde (industrial Canal+)

Autonomous Systems
- **Autonomous Robots**
  - ✓ Robocup (academic project)
- Citizen Agents

MAGMA : People in 00-01

Staff	**Y. Demazeau**	**CNRS**	**approach, organisations**
	H. Fiorino	**UJF**	**agents, interactions**
	JL. Koning	**INPG**	**interactions, models**
	M. Occello	**UPMF**	**recursion, approach**
	S. Pesty	**UPMF**	**interactions, agents**
Post Doctoral	**C. Baeijs**	**INPG**	**organisations, approach**
PhD students	***E. Fianyo - 4***	***UPD***	***dynamics, application***
	G. Chicoisne - 3	**INPG**	**interactions, agents**
	K. Fernandes - 3	**UJF**	**approach, recursion**
	M.-P. Huget - 3	**UPD**	**interactions, models**
	P.-M. Ricordel - 3	**INPG**	**tools, programming**
	J.L. Tavares - 2	**UJF**	**interactions, applications**
	C. Webber - 2	**INPG**	**interactions, applications**
	R. Frozza - 2	**UFRGS**	**interactions, applications**
	I. Romero - 1	**UJF**	**interactions, models**
	A. Zunino - 1	***UJF***	***interactions, applications***
MSc students	**Ph. Couspeyre**	**INPG**	**approach, applications**

Future MAGMA research

Multi-Agent Oriented Programming

- The Volcano platform
- Domains and Problems
- Applications : The Citizen Agents and AgentCities projects

Emotional and Conversational Agents

- Dialogism
- Emotional agents
- Applications : The PARI and Baghera projects

Real-Time Coordination and Planning

- MAS-centered Planning
- MAS-centered Learning
- Applications : The AgentCities and PARI projects