Conceptual modelling for the management of environmental data

Thérèse LIBOUREL ROUGE
libourel@lirmm.fr

Professor
Montpellier Laboratory of Computer Science, Robotics and Microelectronics
CNRS – University of Montpellier 2
Overview

- Introduction
- Object concepts and UML formalism
  - Functional model
  - Structural model
  - Dynamic model
- EIS specificities
- Conclusion and perspectives
Overview

- **Introduction**
- **Object concepts and UML formalism**
  - Functional model
  - Structural model
  - Dynamic model
- **EIS specificities**
- **Conclusion and perspectives**
Environmental information systems
Conceptual modelling for the management of environmental data

Introduction

The real world

Conceptual model
- Independent of the data model
- Independent of the DBMS

Logical model
- Dependent on the data model
- Independent of the DBMS

Physical model
- Dependent on the data model
- Dependent on the DBMS

Physical organization of data
- Data storage structures
- Accelerating structures (index)
The approach and the modelling are essential
.... for future extensions
.... for exchange

A model is a simplification/abstraction of the reality

Constructing models allows us to better understand the systems we are developing ...
Why methods?

A necessity: reduce the gaps

The real

- Evolutionary
- Ambiguity

The computer-based world

- Codified languages
- Unique semantics
The methods = structuring guides

- Decomposition of work
- Organization into stages
- Founding concepts
- Semi-formal representations

Ensure a reproducible approach to obtain reliable results
To summarize ...

An analysis and design method proposes an approach that distinguishes the development stages of a software’s life cycle bases itself on a representation formalism that facilitates communication, organization and verification.

The modelling language produces documents (models) that facilitate feedback on the design and evolution of applications.
The trend towards EIS

- Systemic methods vs. object methods
- E/R vs. OO data modelling
- From closed formalisms to extensible formalisms (ex. UML)
- From aspatial and atemporal formalisms to formalism with spatial and temporal extension
Introduction

- CONGOO [Pantazis and Donnay, 1996]
- Geo-ER [Hadzilacos and Tryfona, 1997]
- Geo-OM [Tryfona et al., 1997]
- GeoOOA with its modelling software [Kösters et al., 1997]
- MADS with its modelling software [Parent et al., 1997]
- Modul-R with its modelling software and its automatic code generator [Bédard and Paquette, 1989; Pageau and Bédard, 1992; Caron et al., 1993; Bédard et al., 1996] ...

PERCEPTORY

- OMEGA and AIGLE [El Bath, 1997]
- POLLEN [Gayte et al., 1997]
Object-oriented approach for projects

Keywords
The 4 P’s

Process

Tools

automatization

* participants

Artefact, result

People

Project

Product
UML: a formal description language

Accepted by the OMG – 1997
A unique common language:
  A meta-model
  A less ambiguous language
  A simple graphical notation,
  Comprehensible by non-computer-specialists
  Allows communication between actors

Has become THE reference for object modelling
Overview

- **Introduction**
- **Object concepts and UML formalism**
  - Functional model
  - Structural model
  - Dynamic model
- **EIS specificities**
- **Conclusion and perspectives**
Object concepts and ULM formalism

The perception of models
The graphical views (diagrams)

Class diagrams
Object diagrams

Sequence diagrams
Collaboration diagrams

State-transition diagrams
Activity diagrams

Use case diagrams
Component diagrams
Deployment diagrams
System functions from the user's point of view.

Objects and basic relationships between these objects.

Object concepts and ULM formalism

Static structure of classes and relationships between these classes.

Physical components of an application.

Representation of behaviour in terms of states.

Schemas for installation of components on the hardware mechanisms.

Representation of objects, mutual links and potential interactions.

Representation of operational behaviour in terms of actions.
Object concepts and ULM formalism

Construction structure

Logic
classes/dynamics

Implementation
components

Use cases
use case + scenarios

Processes
scenarios on the components
concurrency
distribution
tolerance to breakdowns

Deployment
Components ‘projected’
on the hardware

Model orientation by use cases
The ‘USE CASEs’

- Descriptive models from the users’ points of view
- Functional scenarios

The manner of using the system
Two concepts

Actor

Use case

Conceptual modelling for the management of environmental data

Functional model

Actor (Decider)

Actor (Producer)

Manage data

Display indicators

<<communicate>>
Use cases can be linked by relationships

- ‘use’ usage (decomposition)
- ‘extend’ refinements (exceptions processing)
Conceptual modelling for the management of environmental data

Diagram of the ‘static context’

```
Actor (role 1) -- 0..1 -- system

Actor (role 2) -- 0..* -- system

'actor' role
```

association
In UML, the structural or static model is described using two types of diagrams

**Class diagrams**
description of all or part of a system in an abstract way, in terms of classes, structure and associations.

**Object diagrams**
description of configuration examples of all or part of the system, in terms of objects, values and links.
Conceptual modelling for the management of environmental data

**Structural model**

**The objects**

Objects in the real world

Computer-based objects
Conceptual modelling for the management of environmental data

**Object**

State: evolves over time

Behaviour: actions and reactions

Identity: essence

Behaviour influences state
State reflects past behaviours
Conceptual modelling for the management of environmental data

**Structural model**

- **Discipline**
  - **Label**
  - **Label** nbhCourse
  - **Label** nbhPracticals
  - **Describe ()**
  - **Inscribe (e: Student)**

- **name of the class**
- **attributes**
- **operations**

**Implementations**

- **Operations and methods**
- **Methods**
Conceptual modelling for the management of environmental data

**Class**

**Attributes (properties)**

- Discipline
  - Label
  - nbhCourse
  - nbhPracticals

**Instance**

**Attribute values (State)**

- : Discipline
  - Label = ACO
  - nbhCourse = 12
  - nbhPracticals = 15
Association/Link
(analogy Class/Instance)

- **Discipline**
  - label

- **Teacher**
  - name

- **: Discipline**
  - label=ACO

- **: Teacher**
  - name = Louis
Association in general is binary (degree = 2) but...
Multiplicity and an association’s roles

**Company**
- Name
- Address

**Individual**
- Name
- Birth date
- SS no.
- Address

**Multiplicities and Roles**
- `Company` `employs` `Individual` (0..1)
- `Individual` `works-for` `Company` (1..*):
  - `Company` `employs` `Individual` (1..*)
  - `Individual` `works-for` `Company` (1..*)

**Additional Information**
- `Individual` `name`
- `Individual` `Birth date`
**Conceptual modelling for the management of environmental data**

**Structural model**

**Association class**

<table>
<thead>
<tr>
<th>Enterprise</th>
<th>Possesses-shares</th>
<th>Individual</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>*</td>
<td>name</td>
</tr>
<tr>
<td>address</td>
<td></td>
<td>date of birth</td>
</tr>
</tbody>
</table>

Possession

- Line of portfolio
- quantity
Other ‘abstractions’

- **Specific associations**
  (composition/aggregation)

- **Specialization/generalization**
Composition

Specific association All/part

- Window
  - Title bar
    - Title
  - B’ground
  - Border
  - Scroll bar
    - Arrow
    - Indicator
  - 0..2
  - 2

Structural model
**Aggregation**

Semantic Collection/Item

![Diagram showing the structural model with Aggregation](image-url)
Structural model

**Composition/Aggregation**

Constraints

- Exclusivity/Sharing
- Dependence/Independence

Propagation/Distribution


**Generalization/Specialization**

Mechanism for intellectual inference of characteristics
Either we refine (specialization)
Or we abstract (generalization)

Semantic
Set-theoretic point of view
Logical point of view
Generalization/Specialization

<table>
<thead>
<tr>
<th>Individual</th>
</tr>
</thead>
<tbody>
<tr>
<td>name address</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID no. address</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Teacher</th>
</tr>
</thead>
<tbody>
<tr>
<td>grade address</td>
</tr>
<tr>
<td>teach</td>
</tr>
</tbody>
</table>

{disjoint}
Generalization/Specialization

![Diagram of conceptual modelling for environmental data]

- Equipment
  - Pump
    - Centrifugal pump
    - Submersible pump
  - Heat exchanger
  - Tank
    - Pressure tank

Type of equipment
Type of pump
Type of tank
Multiple generalization/Specialization
Composition/Aggregation or Generalization/Specialization?

- Aggregation
  - link between instances
  - an aggregation tree is composed of objects that are part of a composite object

- Generalization
  - link between classes
The constraints

- The *constraints* are predicates, that can apply to several items of the static model, which have to be verified at all times.

- The constraints allow details to be taken into account at a very fine granularity in a class diagram. They can express conditions or restrictions.

- In UML, the constraints are expressed in textual form, between brackets and, preferably, in OCL (Object Constraint Language).

- Constraints are inherited.
The constraints

- **Route**
  - *{ordered}
  - 1..*

- **Individual**
  - *memberOf
  - {subset}
  - 1

- **Committee**
  - *presides
  - *

- **Edge**
  - *{ordered}
  - 1..*

- **Constraint on association extremity**

- **Constraint between two associations**
The constraints
Describes interactions between objects and changes that take place over time

- Collaboration diagrams
- Sequence diagrams
- State-transition diagrams
- Activity diagrams (not covered)
Collaboration diagram

Dynamic model
**Sequence diagram**

Dynamic model
Life line

Creation by ‘create’ message

Activation of object that executes an operation op

Destruction by another object

`create` message

`destroy` message

Dynamic model
Dynamic model
Dynamic model

**Event and State**

- State of an object
  - values of its attributes and of its links
  - over time an object can change state

- Event
  - stimulus from an object towards another object
**Notation of states**

- **Initial**
- **Final**
- **Simple**
- **Complex**

**State name**
- entry/op1
- exit/ op2
- on evt1/ op3
- on evt2/ op4
- do/ activity

**Internal activities**
- At the beginning
- At the end
- During the event
- All the time

**Creditor**
Notation of arcs

- event(parameters)
- [condition]
- /action
**State diagrams**

**States of a bank account**

- **Open**
  - CreateRequest()
- **Close**
  - Close()

**Balance > 0**
- **Debtor**
  - We withdraw/premiums
  - We deposit/increase balance
- **Creditor**
  - We withdraw/debit balance
  - We deposit/increase balance

**Balance < 0**
- **open**
- Close()
Overview

• **Introduction**
• **Object concepts and UML formalism**
  • Functional model
  • Structural model
  • Dynamic model
• **EIS specificities**
• **Conclusion and perspectives**
**EIS specificities**

Spatiality

Temporality

Which representations?...
Taking the geometry into account
Representations of space

*Euclidean geometry*

point (dim 0) line (dim 1)
area (dim 2) volume (dim 3)
**Euclidean geometry**

- point (dim 0)
- line (dim 1)
- area (dim 2)
- volume (dim 3)
Euclidean geometry

point (dim 0) line (dim 1)
area (dim 2) volume (dim 3)
**Euclidean geometry**

point (dim 0) line (dim 1)
area (dim 2) volume (dim 3)
Representations of space

**Graphs**

Node, Edge

Node

Edge

2..2

1..n
delimits

Is-linked-to

0..n
Representations of space

Topology...
Representations of space

Topology...
Representation of time ...

Instant

1 start

1 end

Interval

0..n

duration

0..n

operators
And the dynamics ...

Objects’ life line

- Creation
- Fission
- Fusion
- Displacement
- Growth
- Deformation

Life

Movement
Objects’ life line

Conceptual modelling for the management of environmental data

EIS specificities
Overview

• **Introduction**

• **Object concepts and UML formalism**
  • Functional model
  • Structural model
  • Dynamic model

• **EIS specificities**

• **Conclusion and perspectives**
**Conclusion**

Approach having ‘proven’ itself

Example: Aggregative perceptions of the landscape

[Lardon & al., 2000]
Conclusion

OpenGIS standard
**Perspectives**

- Taking the ‘cognitive’ into account

**ST pictograms**

**PVL (Perceptory)**

0D  1D  2D  ...

**Mads**
Perspectives

Complex projects

- towards the GIS’s (geodatabase)
- towards object-oriented DBMS extensions thematic and spatial integration

Instrumentation: AGL
Conceptual modelling for the management of environmental data

Thérèse LIBOUREL ROUGE
libourel@lirmm.fr

Professor
Montpellier Laboratory of Computer Science, Robotics and Microelectronics
CNRS – University of Montpellier 2
Overview

- Introduction
- Object concepts and UML formalism
  - Functional model
  - Structural model
  - Dynamic model
- EIS specificities
- Conclusion and perspectives
Overview

• **Introduction**

• **Object concepts and UML formalism**
  - Functional model
  - Structural model
  - Dynamic model

• **EIS specificities**

• **Conclusion and perspectives**
**Environmental information systems**

Diagram showing the interaction between social, environmental, and legislative aspects.
Conceptual modelling for the management of environmental data

**Introduction**

The real

Conceptual model

- Independent of the data model
- Independent of the DBMS

Logical model

- Dependent on the data model
- Independent of the DBMS

Physical model

- Dependent on the data model
- Dependent on the DBMS

Physical organization of data
Data storage structures
Accelerating structures (index)
The approach and the modelling are essential
.... for future extensions
.... for exchange

A model is a simplification/abstraction of the reality

Constructing models allows us to better understand the systems we are developing ...
Why methods?

A necessity: reduce the gaps

- Evolutionary
- Ambiguity

The real

- Codified languages
- Unique semantics

The computer-based world
Introduction

The methods = structuring guides

- Decomposition of work
- Organization into stages
- Founding concepts
- Semi-formal representations

Ensure a reproducible approach to obtain reliable results
To summarize ...

An analysis and design method proposes an approach that distinguishes the development stages of a software’s life cycle bases itself on a representation formalism that facilitates communication, organization and verification.

The modelling language produces documents (models) that facilitate feedback on the design and evolution of applications.
The trend towards EIS

• Systemic methods vs. object methods
• E/R vs. OO data modelling
• From closed formalisms to extensible formalisms (ex. UML)
• From aspatial and atemporal formalisms to formalism with spatial and temporal extension
CONGOO [Pantazis and Donnay, 1996]
Geo-ER [Hadzikac and Tryfona, 1997]
Geo-OM [Tryfona et al., 1997]
GeoOOA with its modelling software [Kösters et al., 1997]
MADS with its modelling software [Parent et al., 1997]
Modul-R with its modelling software and its automatic code generator [Bédard and Paquette, 1989; Pageau and Bédard, 1992; Caron et al., 1993; Bédard et al., 1996] ...
PERCEPTORY
OMEGA and AIGLE [El Bath, 1997]
POLLEN [Gayte et al., 1997]
Object-oriented approach for projects

Keywords

The 4 P’s

- People
- Project
- Process
- Tools

Relationships:
- People to Project (participants)
- Project to Product (Artefact, result)
- Process to Tools (automatization)
UML: a formal description language

Accepted by the OMG – 1997
A unique common language:
  A meta-model
  A less ambiguous language
  A simple graphical notation,
  Comprehensible by non-computer-specialists
  Allows communication between actors

Has become THE reference for object modelling
Overview

• Introduction
• Object concepts and UML formalism
  • Functional model
  • Structural model
  • Dynamic model
• EIS specificities
• Conclusion and perspectives
Object concepts and ULM formalism

The perception of models
The graphical views (diagrams)

Class diagrams
Object diagrams

Sequence diagrams
Collaboration diagrams

State-transition diagrams
Activity diagrams

Use case diagrams
Component diagrams
Deployment diagrams
Conceptual modelling for the management of environmental data

Object concepts and ULM formalism

System functions from the user’s point of view.

Objects and basic relationships between these objects.

Physical components of an application.

Representation of behaviour in terms of states.

Schemas for installation of components on the hardware mechanisms.

Static structure of classes and relationships between these classes.

Representation of operational behaviour in terms of actions.
Conceptual modelling for the management of environmental data

Object concepts and ULM formalism

**Construction structure**

<table>
<thead>
<tr>
<th>Logic</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>classes/dynamics</td>
<td>components</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Processes</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>scenarios on the components</td>
<td>Components ‘projected’ on the hardware</td>
</tr>
<tr>
<td>concurrence</td>
<td></td>
</tr>
<tr>
<td>distribution</td>
<td></td>
</tr>
<tr>
<td>tolerance to breakdowns</td>
<td></td>
</tr>
</tbody>
</table>

**Use cases**

- use case + scenarios

**Model orientation by use cases**
The 'USE CASEs'

- Descriptive models from the users’ points of view
- Functional scenarios

The manner of using the system
Two concepts

Actor

Use case

Conceptual modelling for the management of environmental data
Use cases can be linked by relationships

- ‘use’ usage (decomposition)
- ‘extend’ refinements (exceptions processing)
Diagram of the ‘static context’

- **Actor (role 1)**
  - 0..1 relationship with **system**
  - 0..1 relationship with **‘actor’ role**

- **Actor (role 2)**
  - 0..* relationship with **system**

- **System**
  - Association with **Actor (role 2)**
  - 0..1 relationship with **‘actor’ role**
**Structural model**

In UML, *the structural or static model is described using two types of diagrams*.

**Class diagrams**
description of all or part of a system in an abstract way, in terms of classes, structure and associations.

**Object diagrams**
description of configuration examples of all or part of the system, in terms of objects, values and links.
**The objects**

Objects in the real world

Computer-based objects

*Structural model*

Visible behaviour

Hidden internal state
Object

State evolves over time

Behaviour actions and reactions

Identity essence

Behaviour influences state

State reflects past behaviours
Conceptual modelling for the management of environmental data

Discipline

System

DB

Professor

Sophie

Luc

Alain

Two objects or instances
Conceptual modelling for the management of environmental data

**Structural model**

**Discipline**
- **Label**: nbhCourse, nbhPracticals
- **Describe ()**: Inscribe (e: Student)

**attributes**
- name of the class

**operations**

**Implementations**
**Methods**

**Operations and methods**
Conceptual modelling for the management of environmental data

**Class**

**Attributes (properties)**

- Discipline
  - Label
  - nbhCourse
  - nbhPracticals

**Instance**

**Attribute values (State)**

- : Discipline
  - Label = ACO
  - nbhCourse = 12
  - nbhPracticals = 15
Conceptual modelling for the management of environmental data

**Structural model**

Association/Link
*(analogy Class/Instance)*

- **Discipline**
  - label

- **Is-taught**

- **Teacher**
  - name

**: Discipline**
- label=ACO

**: Teacher**
- name = Louis
Association in general is binary (degree = 2) but...
**Multiplicity and an association’s roles**

![Diagram showing conceptual modelling with relationships between Company and Individual entities.]

- **Company**
  - Name
  - Address

- **Individual**
  - Name
  - Birth date
  - SS no.
  - Address
  - [Staff] 0..1
  - [Head] 0..1

- **Relationships**
  - **Employs**
    - Employer: * (Company)
    - Employee: 1..* (Individual)
  - **Works-for**
    - Employee: * (Company)
    - Employer: 1..* (Individual)
Association class

Enterprise
  name
  address

Possesses-shares
  *
  capital
  1..*
  shareholder

Individual
  name
  date of birth
  address

Possession
  Line of portfolio
  quantity

Structural model
**Other ‘abstractions’**

- **Specific associations**
  (composition/aggregation)

- **Specialization/generalization**
Composition

Specific association All/part

Structural model

- Window
  - Title bar
    - Title
  - B’ground
  - Border
  - Scroll bar
    - Arrow
    - Indicator
    - 2
    - 0..2
Aggregation

Semantic Collection/Item

- Forest
  - Tree
    - 1..n
    - 1

- Country
  - Region
    - 1
    - 1..n
  - State
    - 1
    - 1..n
Conceptual modelling for the management of environmental data

*Structural model*

**Composition/Aggregation**

Constraints

- Exclusivity/Sharing
- Dependence/Independence

Propagation/Distribution
Generalization/Specialization

Mechanism for intellectual inference of characteristics
Either we refine (specialization)
Or we abstract (generalization)

Semantic
Set-theoretic point of view
Logical point of view
Generalization/Specialization

Individual
- name
- address

Student
- ID no.
- address

Teacher
- grade
- address
- teach

{disjoint}
Generalization/Specialization

- Equipment
- Pump
  - Centrifugal pump
  - Submersible pump
- Heat exchanger
- Tank
  - Pressure tank
Multiple generalization/Specialization

- Vehicle
  - Terrestrial vehicle
    - Car
  - Aquatic vehicle
    - Amphibious vehicle
    - Ship
**Composition/Aggregation or Generalization/Specialization?**

- **Aggregation**
  - link between instances
  - an aggregation tree is composed of objects that are part of a composite object

- **Generalization**
  - link between classes
The constraints

• The constraints are predicates, that can apply to several items of the static model, which have to be verified at all times.

• The constraints allow details to be taken into account at a very fine granularity in a class diagram. They can express conditions or restrictions.

• In UML, the constraints are expressed in textual form, between brackets and, preferably, in OCL (Object Constraint Language).

• Constraints are inherited.
The constraints
The constraints

- **Constraint on class**
  - \{ active = passive \}

- **Constraint on attribute**
  - \{ Individual.employer = Individual.head.employer \}

- **Constraints on 2 associations**
Describes interactions between objects and changes that take place over time

- Collaboration diagrams
- Sequence diagrams
- State-transition diagrams
- Activity diagrams (not covered)
**Conceptual modelling**

*Dynamic model*

**Collaboration diagram**

- **Communication**
  - **Country**
  - **Region**
  - **State**

*message*
Sequence diagram
**Conceptual modelling** for the management of environmental data

**Dynamic model**

**Life line**

- Creation by 'create' message
- Activation of object that executes an operation op
- Destruction by another object

- 'create'
- 'destroy'

- :C1

- op
Dynamic model

Conceptual modelling for the management of environmental data
Event and State

- State of an object
  - values of its attributes and of its links
  - over time an object can change state

- Event
  - stimulus from an object towards another object
**Notation of states**

- **Initial**
- **Final**
- **Simple**
- **Complex**

**State name**
- entry/op1
- exit/ op2
- on evt1/ op3
- on evt2/ op4
- do/ activity

**Internal activities**
- At the beginning
- At the end
- During the event
- All the time

**Dynamic model**

**Creditor**
**Notation of arcs**

- event(parameters)
- [condition]
- /action
State diagrams

States of a bank account

CreateRequest()  \rightarrow  Open

Open  \rightarrow  Close()

Close()  \rightarrow  \bullet

open

[Balance \geq 0]

Creditor
We withdraw/debit balance
We deposit/increase balance

[Balance < 0]

Debtor
We withdraw/premiums
We deposit/increase balance

Close()
Overview

- **Introduction**
- **Object concepts and UML formalism**
  - Functional model
  - Structural model
  - Dynamic model
- **EIS specificities**
- **Conclusion and perspectives**
**EIS specificities**

Spatiality

Temporality

Which representations?...
Taking the geometry into account

- **Road**
  - name: string
  - speed: integer
  - Create()
  - Display()
  - ChangeSpeed(v)

- **Highway**

- **Line**
  - 1

- **Point**
  - x: real
  - y: real
  - 0..*
  - 2..*
Representations of space

Euclidean geometry

- point (dim 0)
- line (dim 1)
- area (dim 2)
- volume (dim 3)
Euclidean geometry

point (dim 0) line (dim 1)
area (dim 2) volume (dim 3)
Euclidean geometry

point (dim 0) line (dim 1)
area (dim 2) volume (dim 3)
Euclidean geometry

point (dim 0) line (dim 1)
area (dim 2) volume (dim 3)
Representations of space

Graphs

Node, Edge

Node

Edge

delimits

Is-linked-to

0..n

2..2

1..n
Representations of space

Topology...
Representations of space

Topology...
Representation of time ...

Diagram:

- Instant
  - 1 start
  - 1 end

- Interval
  - 0..n duration
  - 0..n operators
And the dynamics ...

Objects’ life line

- Creation
- Fission
- Fusion
- Displacement
- Growth
- Deformation

Life

Movement
Objects’ life line

- 'create'
- evolution
- fission

Time
Overview

- Introduction
- **Object concepts and UML formalism**
  - Functional model
  - Structural model
  - Dynamic model
- **EIS specificities**
- **Conclusion and perspectives**
**Conclusion**

Approach having ‘proven’ itself

Example: Aggregative perceptions of the landscape

[1..*]

[Lardon & al., 2000]
Conclusion

OpenGIS standard
**Perspectives**

- Taking the ‘cognitive’ into account

**ST pictograms**

**PVL (Perceptory)**

- 0D
- 1D
- 2D
- ...

**Mads**
Perspectives

Complex projects

• towards the GIS’s (geodatabase)
• towards object-oriented DBMS extensions
  thematic and spatial integration

Instrumentation: AGL