Field-orientation for Continuous Spatio-temporal Phenomena

Silvia Gordillo, Universidad de La Plata
Catalina Mostaccio, Universidad de La Plata
Robert Laurini, Université Lyon I

Contents

1 - Introduction

- Discrete versus continuous objects
- Examples
  - mountains
  - temperature
  - pressure
  - winds
  - rain
  - some sociological / demographic data

Helen Couclelis (1/2)

"*Is the world ultimately made up of discrete, indivisible elementary particles, or is it a continuum with different properties at different locations? This question, already debated by the ancients Greeks, remains one of the major unanswered problems in the philosophy of physics.*"

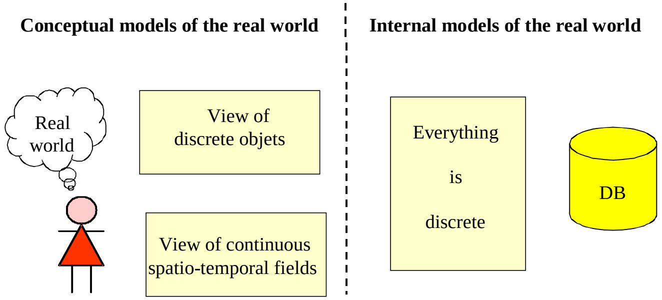
Helen Couclelis (2/2)

"*somehow boundaries are intrinsic to the notion of atom, whereas in the case of extensive entities they are contingent. In other words, the notion of boundary a priori sits better with the atom view of things (and vector GIS) than the plenum view (and raster GIS), whereas the real geographic world forces us to consider both discrete and extensive entities*".

Important Distinction

- Closed boundaries / undetermined boundaries
- Distinguish
  - discrete objects with fuzzy boundaries
  - continuous objects with no boundaries

Conceptual and internal models



Objective of a FO system

- **Conceptual model allowing the user to view the world as consisting of both continuous fields and discretized objects,**

  whereas everything is discretized at implementation level.

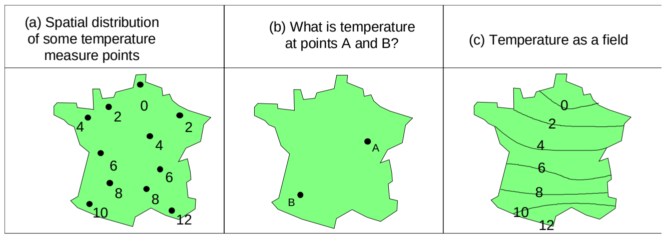
Issues

- Declaration
- Acquisition
- Estimation by interpolation / extrapolation
- Computation of values, gradients, integrals, average, etc.)
- Queries
- Mixing objects and fields

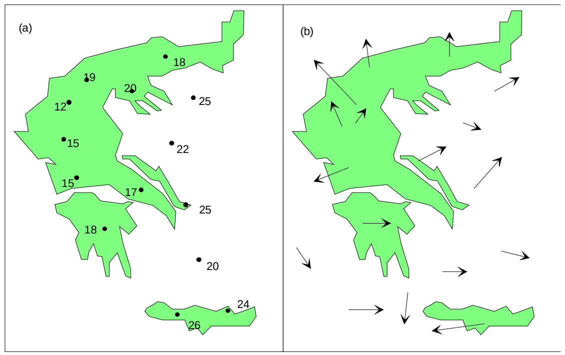
2 - Concepts

- Properties - Continuity
- Scalar and vector fields
- Cross-sections
- Dimensions
- Samples
- Constraints
- Estimation

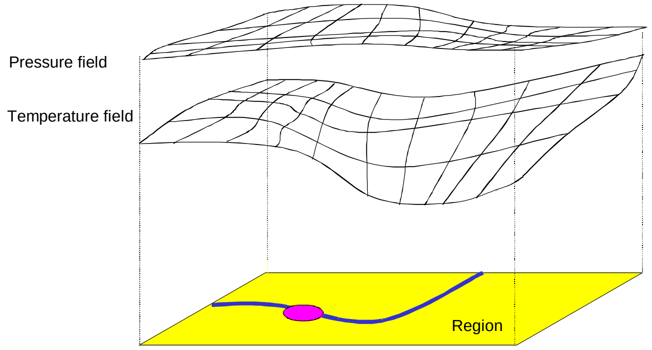
Temperature as a field



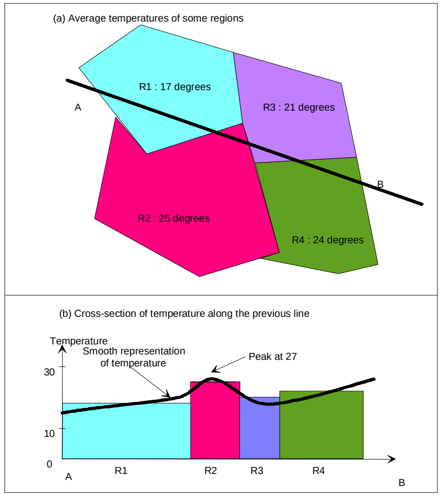
Scalar and Vector fields



Several fields in the same region



Cross-section of a field



Different kinds of fields

Dimension / type	**Scalar**	**Vector**
2D	$h = f(x,y)$	$hx = f(x,y)$ $hy = f(x,y)$
3D	$h = f(x,y,z)$	$h_x = f(x,y,z)$ $hy = f(x,y,z)$ $hz = f(x,y,z)$
3D + time	$h = f(x,y,z,t)$	$h_x = f(x,y,z,t)$ $hy = f(x,y,z,t)$ $hz = f(x,y,z,t)$

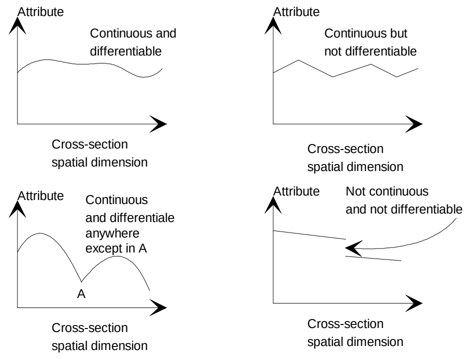
Laplace equation

$$\frac{\partial^2 F}{\partial x^2} + \frac{\partial^2 F}{\partial y^2} + \frac{\partial^2 F}{\partial z^2} + \frac{\partial^2 F}{\partial t^2} = 0$$

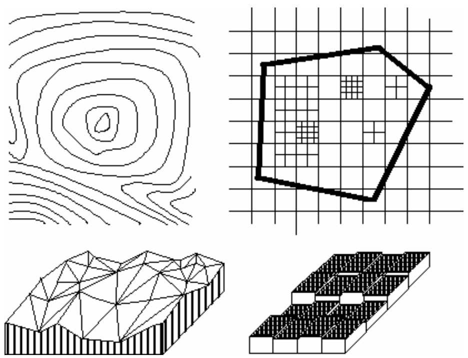
A Finite Differences Approximation

- In 2D:
- $Z_{ij} = (Z_{i-1,j} + Z_{i+1,j} + Z_{i,j-1} + Z_{i,j+1})/4$,
- where $Z_{i}j$ is the value in cell *ij*

Continuity versus differentiability



Different models of scalar fields



Samples

- Impossible to know completely a field
- Measures in some points
  - along a grid
  - along iso-curves
  - random → Triangulated Irregular Networks

Constraints

- Statistical constraints
  - Sometimes integrals or averages are known
- Morphological constraints
  - huge discontinuity (cliffs, etc.)

Estimation

- Estimation everywhere at any time
- Based on the values at the vicinity
  - interpolation procedures
  - extrapolation procedures
- Functions
  - linear
  - splines
  - etc.

3 - Designing GIS Applications

- Involves dealing with different kind of data
- Spatial data must be combined with non-spatial data
- Even, differents kinds of spatial data must be combined (discrete and continuous)
- Multiple algorithms and representations are needed

How can we keep the design clear and evolvable?

Designing GIS Applications

- We should apply good and proven design practices
- We should try to record our design decisions
- We should maximize decoupling of components
- We should focus on architecture prior the implementation

Patterns to the rescue

Design Patterns

- express proven techniques that can be used in new systems
- help to choose design alternatives that make the system easy to evolve
- improve documentation and maintenance

Elements of a pattern

The *pattern name*

The *problem* describes when the pattern can be applied

The *solution* describes the elements that make up the design, their relationships, responsibilities and collaborations

The *consequences* are the results and trade-offs of applying the pattern

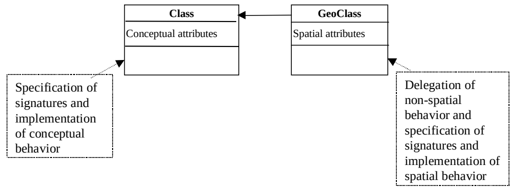
Using DP in GIS

Decoupling geographic from conceptual features

Conceptual model
- Identification of conceptual features and relationships
- Definition of conceptual classes

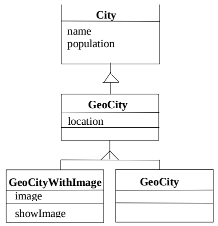
Geographic model
- Identification of which conceptual classes have spatial characteristics
- Definition of spatial classes by using the “Decorator” design pattern

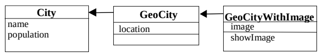
Defining spatial characteristics of discrete objects



The “Decorator” pattern allows us “to add responsibilities to individual objects dynamically and transparently”. It is a more flexible alternative than subclassing for extending functionality

Subclassing vs. Decorating





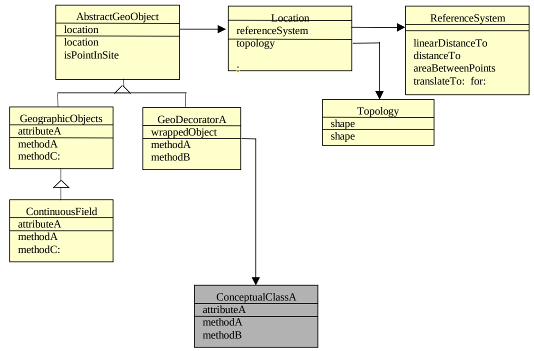
Static Solution

If we add an image to a created object, we have to delete it and create it again

Dynamic solution

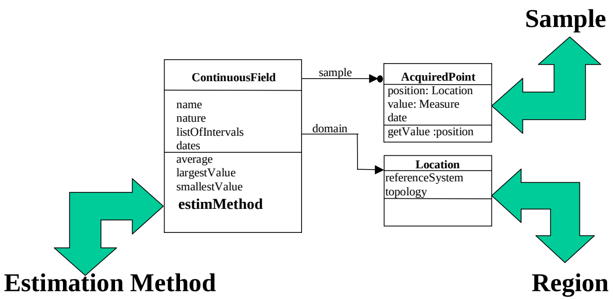
Different characteristics can be added to individual objects

A model to design GIS applications

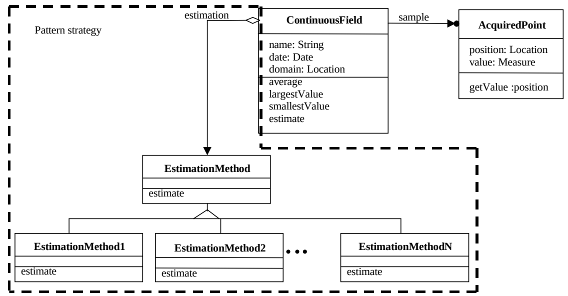


Continuous fields

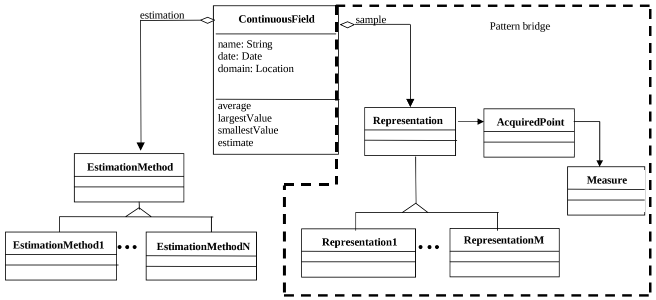
To obtain modularity we have to improve the manipulation of:



Continuous fields: a better design solution



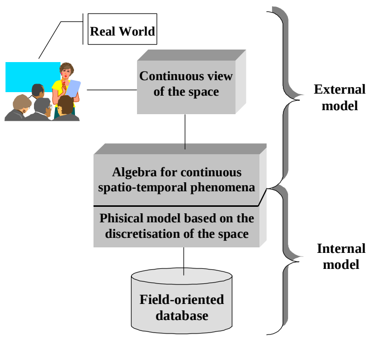
Continuous fields: a better design solution



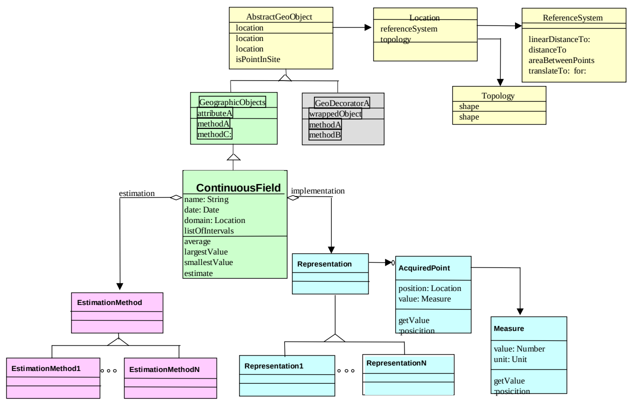
Goals of the architecture

- An external model which is continuous
- A discrete internal model

External and Internal Models

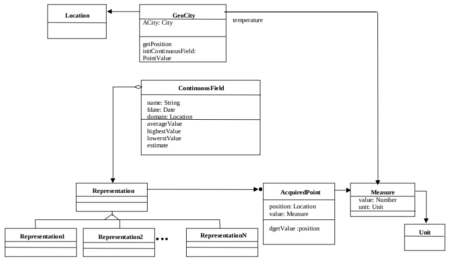


Final Architecture



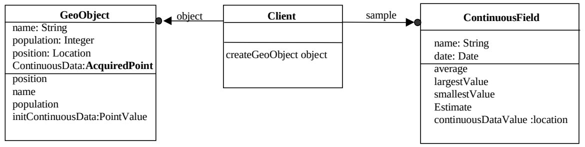
4 - Continuous Fields and Objects

A vector object can take values from a continuous field



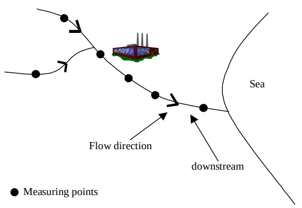
Continuous Fields

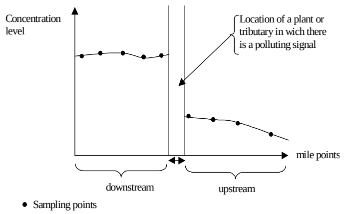
When an instance is created, the client must know the continuous field and the city



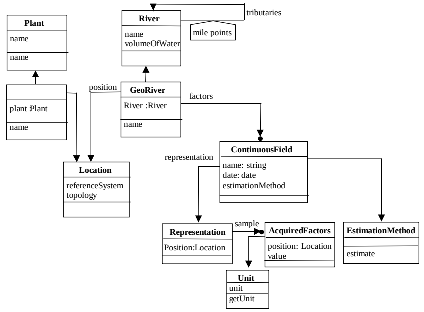
Continuous fields as attributes of objects

Region ➲ Zone ➲ field or subfield as an attribute





Continuous fields as attributes of objects



5 - Operations

Unary operations: They works over the values of a continuous field. The average of the values of the field, the selection of a subfield are some examples.

They are implemented as methods of the class ContinuousField

Binary operations: They combine information from two fields. Operations are based on the union intersection and the difference of fields and a function may be applied to values to obtain the result.

They are implemented in a separate hierarchy

Binary operations

We have to take into account to define the output field:

- The sample of the output field { union, intersection, difference }
- The domain of the output field
- The values of the phenomenon in the new field { max/min, average, addition,etc. }
- The estimation method of the new field
- The compatibilization of the input representations
- The representation of the new field

The sample of the output field

Union:

all acquired points of both fields
estimation of not acquired points
application of an operation to obtain the final value

Intersection:

all acquired points that belongs to both fields
application of an operation to obtain the final value

difference:

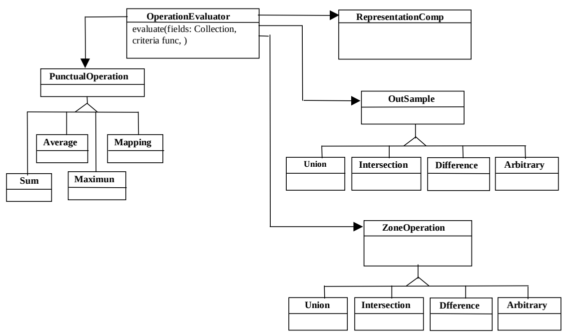
only those points that were acquired in the first field and do not exist in the second fields

The domain of the output field

We have defined two criteria to define the output domain

- Finding the convex hull by taking the output sample
- Performing the union, intersection or difference of the convex hull of the input fields.

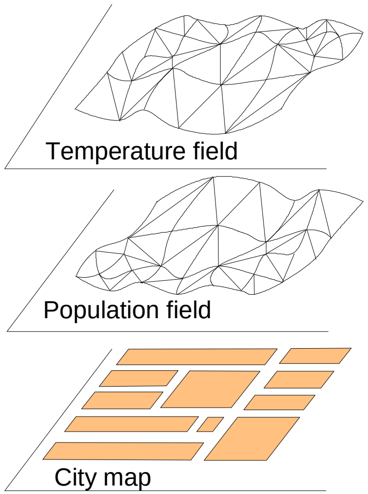
The hierarchy of operations



Example

- Several fields for the same place
- Description of the fields
- Some operators
- Extensions of SQL

Cities and fields



Definition of the previous field

```
Field TEMPERATURE ("temperature","temperature", 3,
 ([0.0,450.0], [-51.5, 378.384] ), [9.10,11.12],
 /* four sample points */
 ( #sample1, #sample2, #sample3, #sample4 ),
 /* two statistical constraints */
 ( #id_stat_mean_1, #id_stat_std_deviation_2 ))

…...

estimate ( TEMPERATURE )
```

Description of a city

```
City Elasty-City
(
 town_area : Location
 temperature:Temp
 TEMPERATURE (town_hall_loc),
 town_population : POPULATION (town_area)
 Method assign_temperature(temp,town_area)
 temperature = temp.getValue(town_area);
)
```

Other operators

```
add_sample ( F, #id_sample )
add_mean ( F, #id_stat_mean )
add_discontinuity ( F, #id_discontinuity )

rem_sample ( F, #id_sample )
rem_mean ( F, #id_stat_mean )
rem_discontinuity ( F, #id_discontinuity )
```

Field-Oriented SQL: FO-SQL (Ouatik, 1999)

- What is the value and the gradient of temperature for a point $p_0 (x_0, y_0, z_0)$ at t_0 ?

```
select X.attribute_value, X.gradient
from X in TEMP
where ( #id_point=p0 ) and ( t=t0 )
```

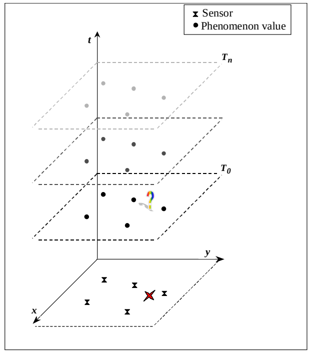
Another query in FO-SQL

- What is the integral of a field RAIN in a zone at a precise date t_0 ?

```
select X.integral
from X in RAIN
where ( #id_Geographic_area=area ) and ( t=t0 )
```

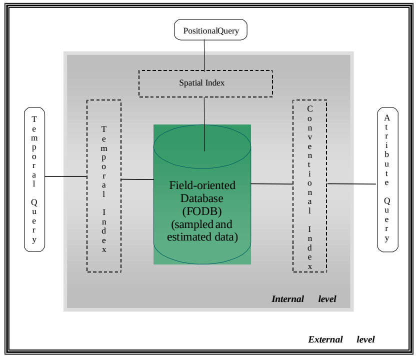
6 - Indexing:

2D Field information to store

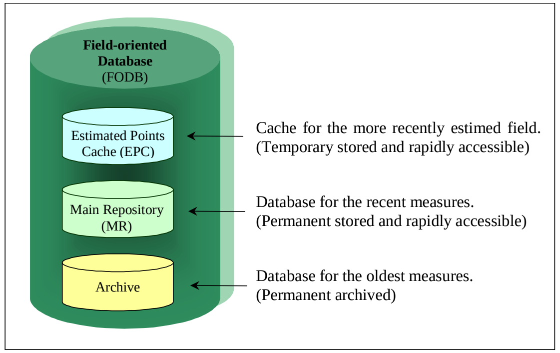


- Sensor information (***Sensors***)
- Sampled data information (***Measures***)
- Estimated data (***EstimatedValues***)

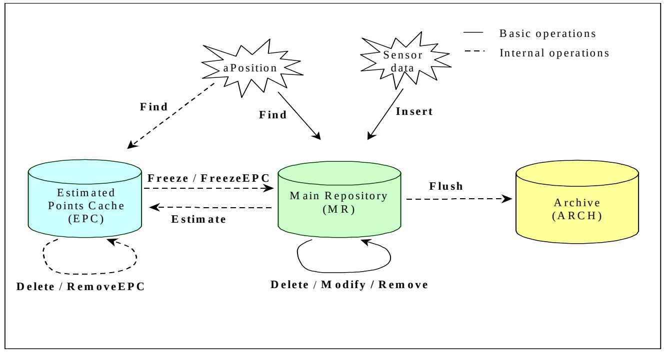
Relation between the external and internal levels



Structuring continuous field data



Operating with field-oriented databases



7 - Conclusions

Our object-oriented architecture allows:

To define a framework in order to reuse basic structures in geographic applications

To reach a good level of design and documentation
(as applications follow closely the architecture style)

To define continuous information in a structured and homogeneus way and combine it with discrete information.

Further works

- Manipulation of other representations
- Implementation of new reference systems
- Implementation of new operations
- 3D fields
- Definition of the query language

Thanks for your attention!

http://lisi.insa-lyon.fr/~laurini

http://www-lifia.info.unlp.edu.ar/~gordillo

http://www-lifia.info.unlp.edu.ar/~catty