ELF flattening tentative

Presentation to: Workshop “Use of INSPIRE data”

Author: Dominique Laurent

Date: 27th November 2018

ISN 18.139

the Competitiveness and Innovation framework Programme (CIP)
ICT Policy Support Programme (PSP) Call 6 (Grant 325140)

Plan

- Context, Introduction
- The envisaged production process
- Specification documents
- Portrayal choices

Context, introduction

ELF (European Location Framework)

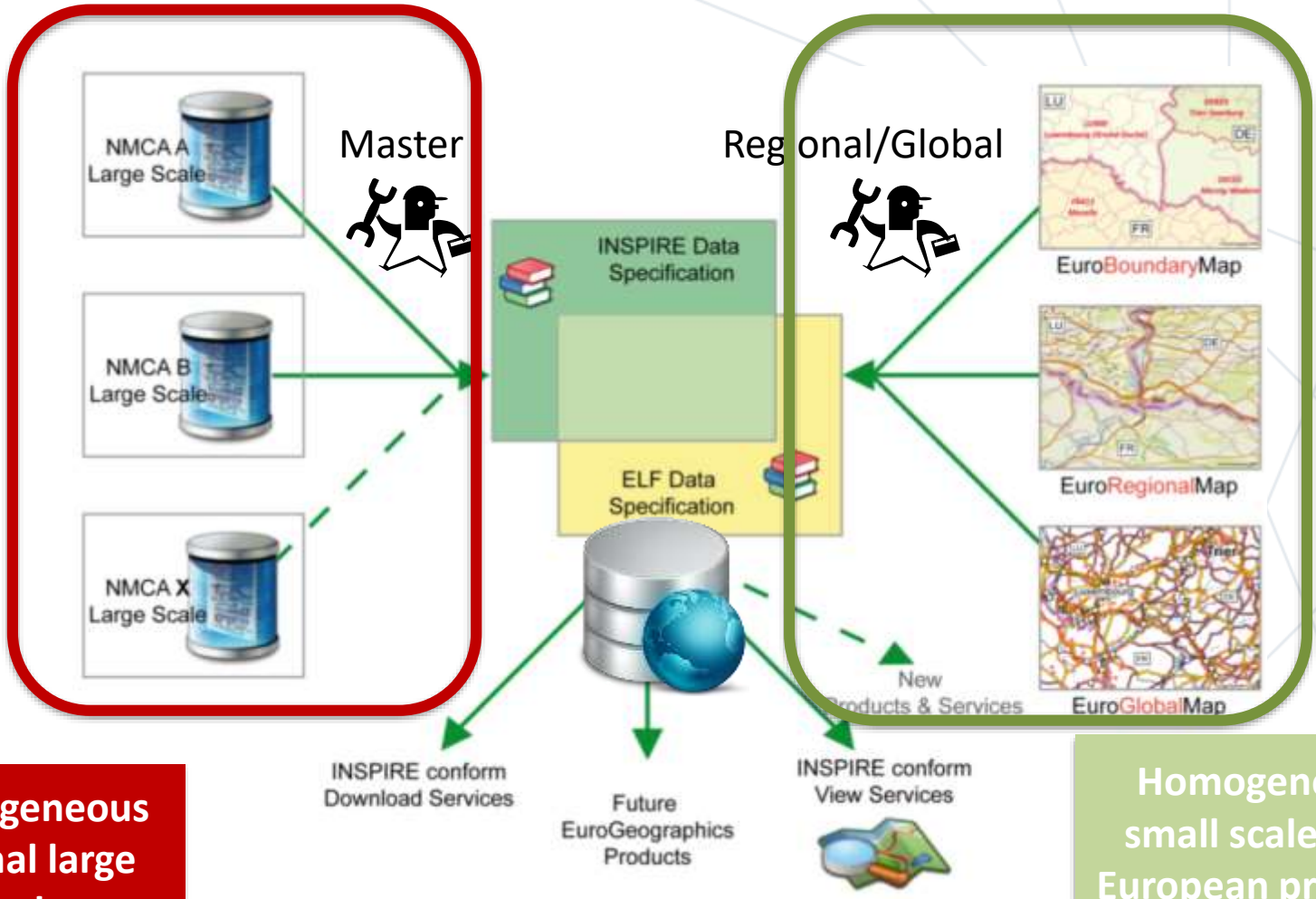
★ ELF is a European project

- ★ From March 2013 to October 2016
- ★ Around 30 partners
  - NMCAs and EuroGeographics
  - Technology providers, Universities, Application developers

ELF partners
(data providers)

- ★ Co-funded by European Commission and the consortium partners
- ★ Main objectives:
  - Implement **INSPIRE +** (interoperability – download services)
  - Offer single access point to these harmonised data and services from NMCAs
  - Make use of INSPIRE data and services
    - Basic services View service (BaseMap), Geocoding Services (GeoLocator)
    - Business applications

Source data (in flat formats)

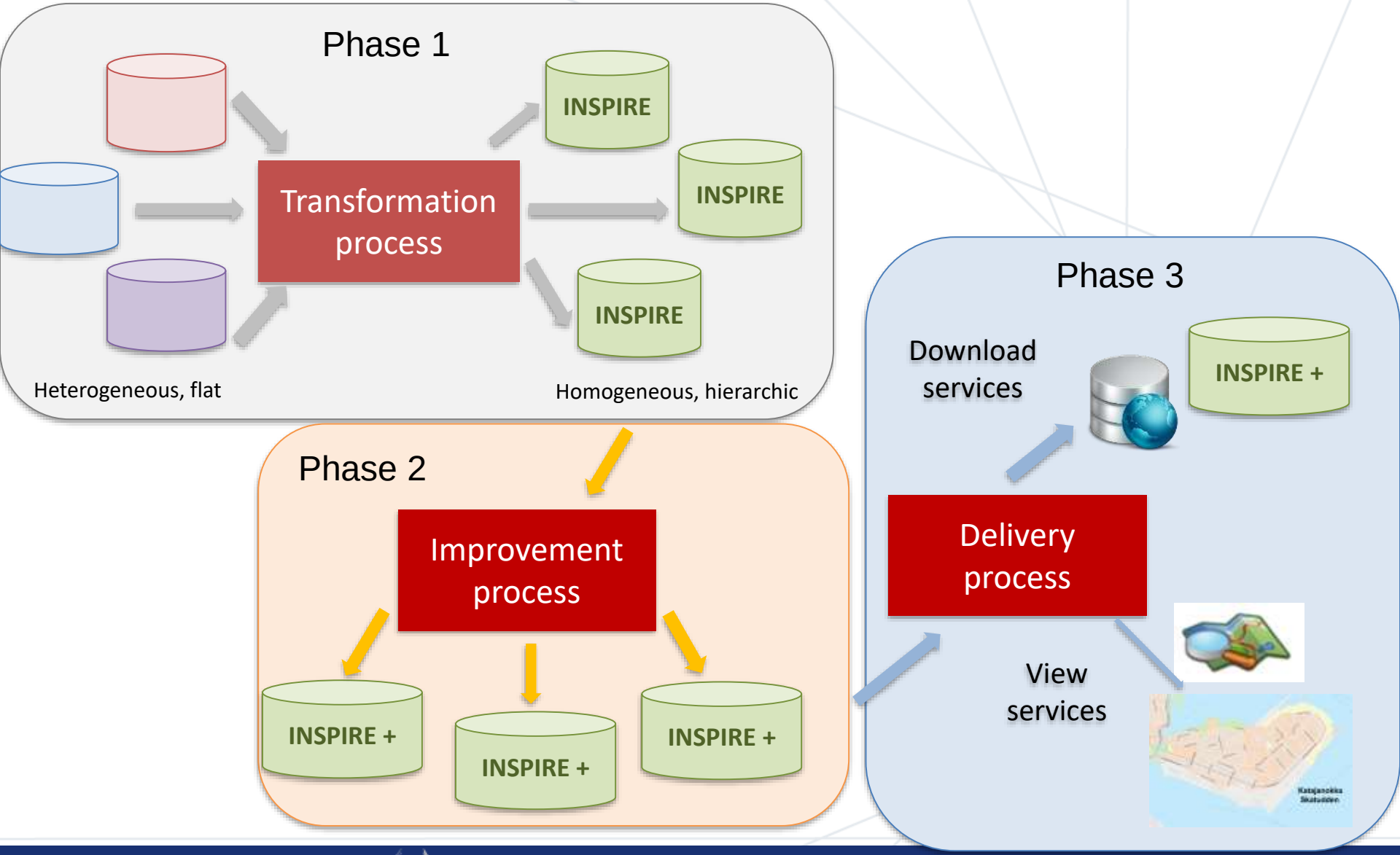


Heterogeneous national large scale data

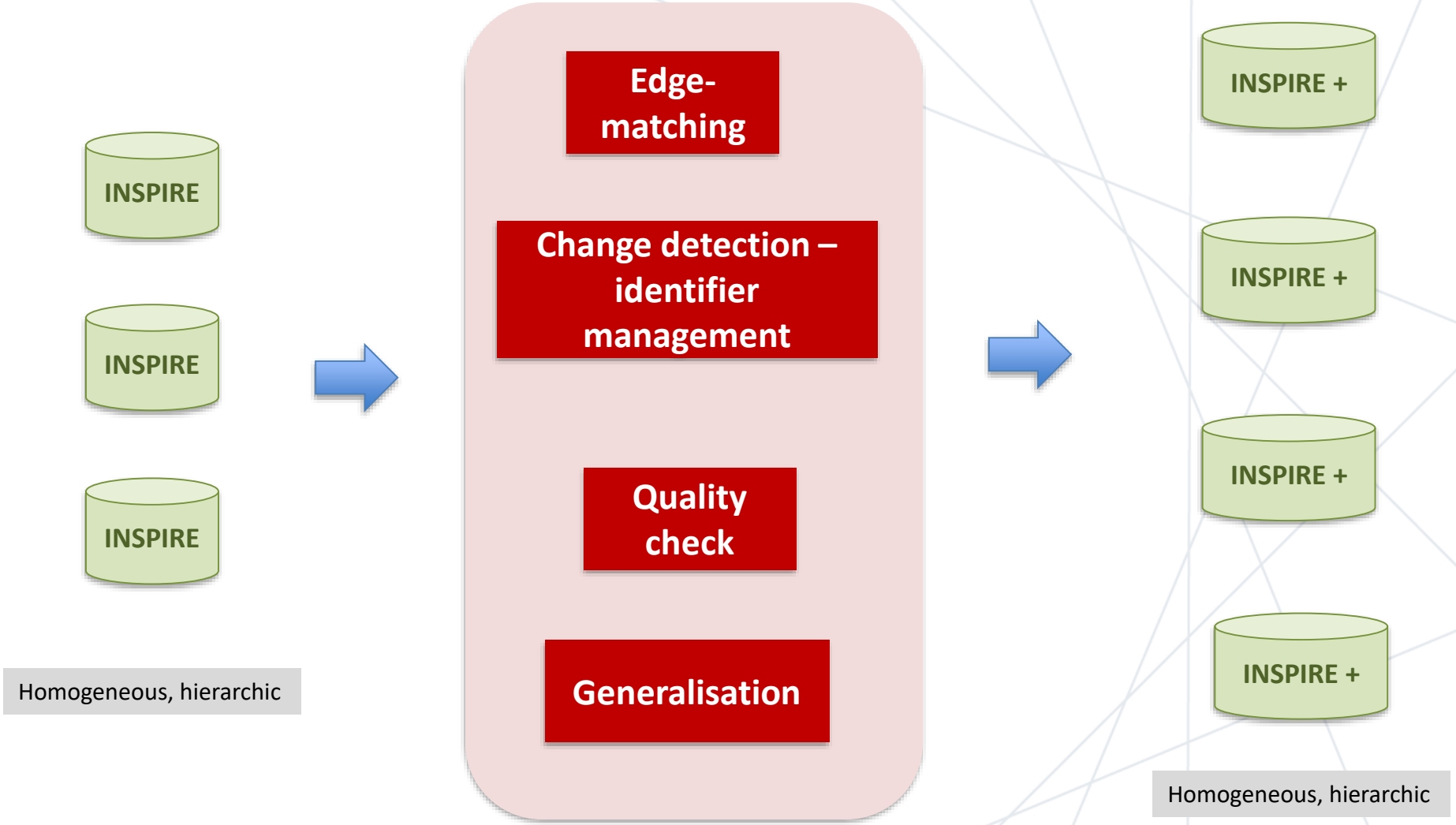
Homogeneous small scale pan-European products

the Competitiveness and Innovation framework Programme (CIP)
ICT Policy Support Programme (PSP) Call 6 (Grant 325140)

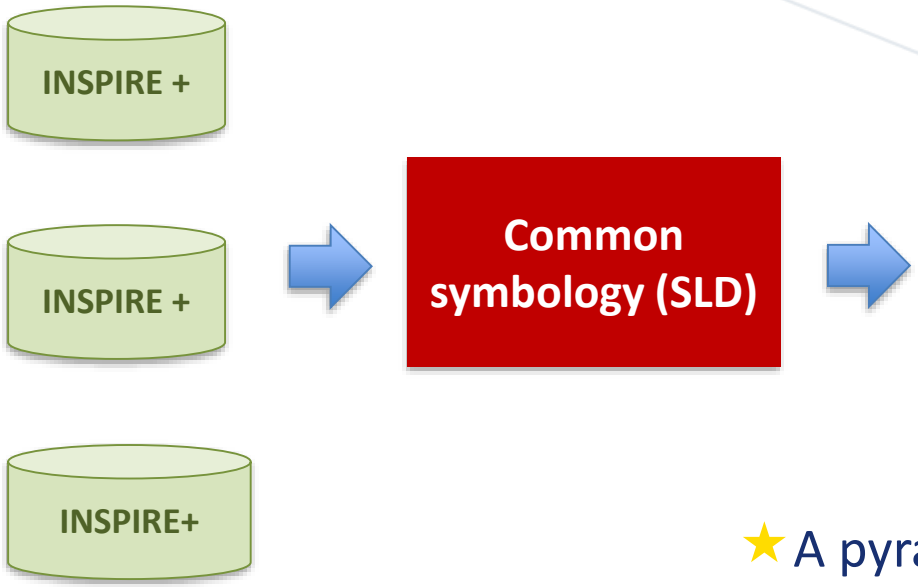
General Data Flow



Data flow (improvement process)



Data flow (Topographic BaseMap)



Homogeneous, hierarchic

- A pyramid of digital images at various levels of zoom
  - From 2K to 40M
- From INSPIRE based data
  - themes AU, TN, HY, LC, BU, GN, EL
- Service WMTS harvesting national WMS services

Common issue

- INSPIRE hierarchic data models
- Initial GeoTools" and SLD generally working only on flat data models

Decision to consider the possibility of common INSPIRE flattened models

Flattening tentative process

Flattening options

- Initial work was to identify the flattening options
- Common issues
  - Attributes with multiplicity > 1
  - Data types
- Specific issues: features without geometry
  - Compound features (e.g. Road, WatercourseLinkSequence)
  - TN properties (or AD components)

Flattening options

★ Example 1 : multiplicity > 1

Flattening options	Example
A) Copies of the property are added to the class so that the class has as many properties as the maximum cardinality	property[1..2] → property_1 and property_2
B) Concatenation of the values into one property	property[1..4] → property='value_1:value_2:value_3:value_4'
C) Link to additional table	property[1..*] → additional table with attributes: UID and property

Flattening options

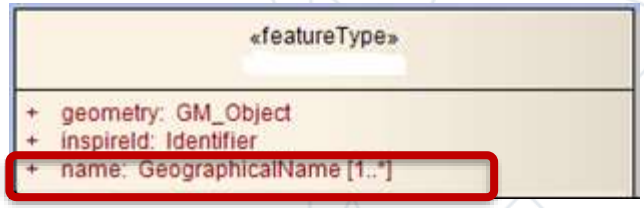
★ Example 1 : data type

Flattening options	Example
A) Copy all properties of the data type over to the original type	Property : dataType / dataType has properties prop1 and prop2 → property.prop1 and property.prop2
B) Concatenation of the values of the data type into one property of the original type	property:dataType / dataType has properties prop1 (val_1) and prop2 (val_2) → property='val_1:val_2'
C) Link to additional table	property:dataType / dataType has properties prop1 and prop2 → additional table with attributes: UID, prop1 and prop2

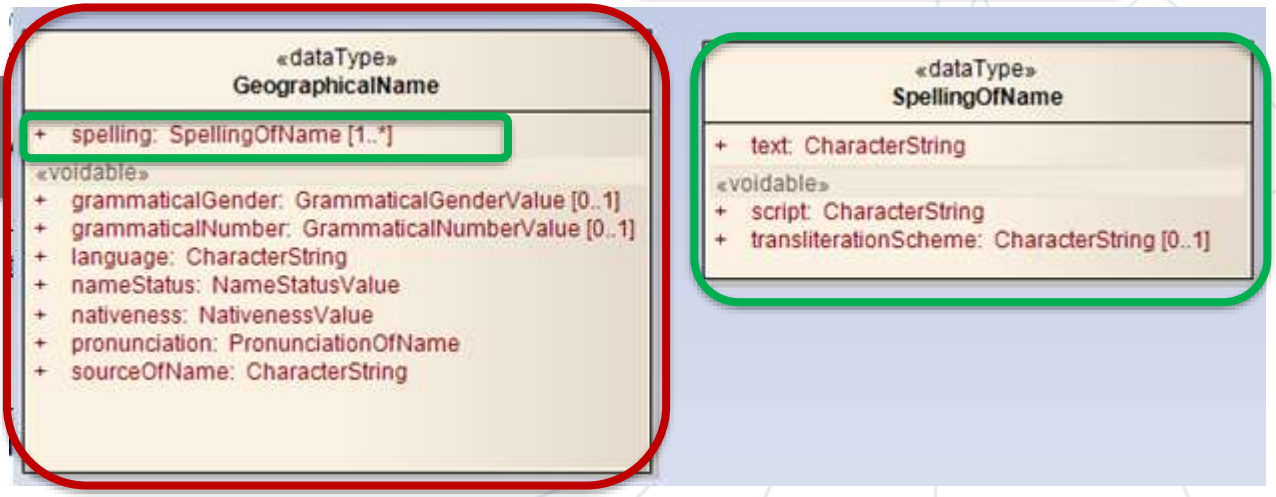
Flattening exercise

- ★ Done for Geographical Name
  - ★ widely used in INSPIRE data models
  - ★ combining :
    - Multiplicity of values
    - Two levels hierarchy (2 data types)

Most feature types may have several names



A name may have several spellings



Flattening exercise

- Flattening option chosen: copies of properties

- Decide on a maximum number of occurrences
  - Name: 3
    - Name1, name2, name3
  - Spelling : 3
    - Name1.spelling1, name1.spelling2, name1.spelling3
    - Name2.spelling1, name2.spelling2, name2.spelling3
    - Name3.spelling3, name3.spelling3, name1.spelling3

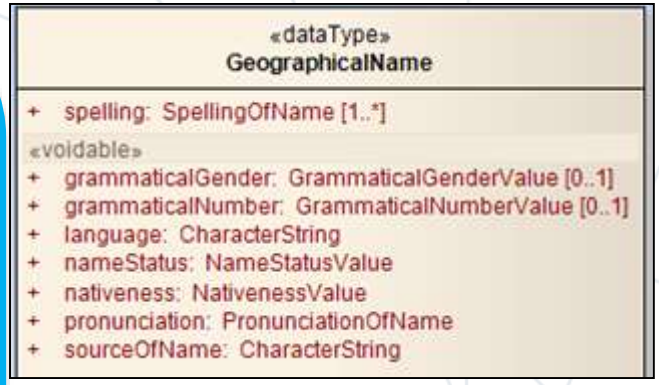
Flattening exercise

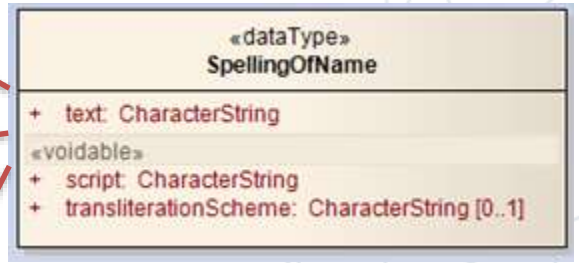
★ Name1

- Name1.grammaticalGender
- Name1.grammaticalNumber
- Name1.language
- Name1.nameStatus
- Name1.nativeness
- Name1.pronunciation
- Name1.sourceOfName
- Name1.spelling1.text
- Name1.spelling1.script
- Name1.spelling1.transliterationScheme
- Name1.spelling2.text
- Name1.spelling2.script
- Name1.spelling2.transliterationScheme
- Name1.spelling3.text
- Name1.spelling3.script
- Name1.spelling3.transliterationScheme

7+3x3 = 16 attributes for each name

3x 16 = 48 attributes





Comments on flattening

★ General comments

Flattening options	Example
A) Copies of the property are added to the class so that the class has as many properties as the maximum cardinality	property[1..2] → property_1 and property_2
B) Concatenation of the values into one property	property[1..4] → property='value_1:value_2:value_3:value_4'
C) Link to additional table	property[1..*] → additional table with attributes: UID and property

This not a flatten model but just a relational implementation

Comments on flattening

- General comments
  - Suitability for various software?
    - objective: conceptual flat model
  - Missing topics
    - Voidable attributes and VoidValueReason
    - Feature types with heterogeneous geometries, e.g. mixing points and polygons
    - Multi-lingual text
    - ….

Comments on flattening

- Specific comments by IGN
  - IGN responsible
    - Topographic BaseMap
    - Generalisation tool (from Regional to Global)
    - Change detection tool

Geotools

Comments on flattening

★ **Keep whole content of INSPIRE data models**?

★ GeoTools : **yes**, INSPIRE as input and output data

★ Topographic BaseMap: **no**

- Need for geometry and a few useful semantic attributes
  - » Selection
  - » Legend

Comments on flattening

★ **Theme TN: Keep the INSPIRE structure with Transport properties as feature types** ?

★ Change Detection tool : **yes,** purpose is to manage **feature identifiers**

★ Topographic BaseMap and generalisation tool: **no**

– Transport properties have to be directly attached to the geometry (as attributes)

Flattening tentative results

Flattening decision

★ We gave up: **ELF didn't propose a common flatten solution for INSPIRE data models**

★ Reasons:

- ★ Initiative came too late in the project
- ★ Industrial software providers (ESRI, 1Spatial, …) were already quite advanced in their developments
- ★ Heterogeneous requirements
  - IGN example: 3 different types of flattening requirements

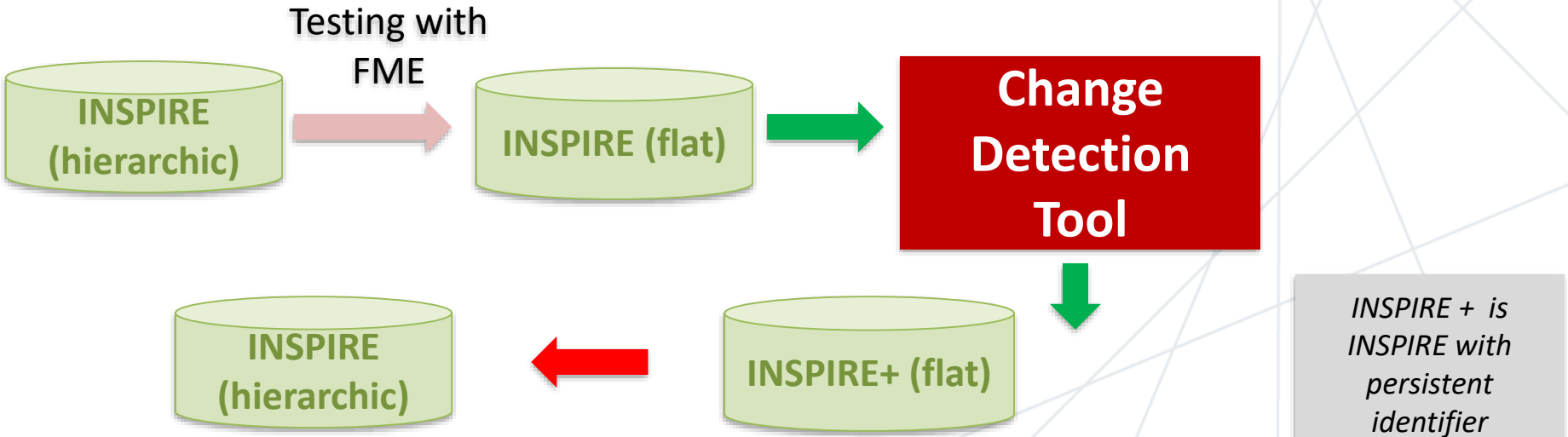
Alternative solutions

- Generalisation tool (IGN)
  - Main purpose was to derive EGM from ERM
  - Both products exist with flat structure
  - No strong need to use the INSPIRE data models

Generalisation tool has been developed based on the flat models of ERM and EGM.

Alternative solutions

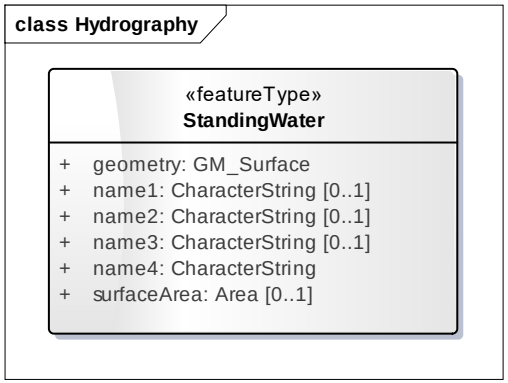
- Change Detection tool (IGN)
  - It can work on any (flat) data model
  - It has been tested on source data (ERM + national – Finland)
  - It has been used for ERM
  - It might work on INSPIRE data with some extra transformation



Alternative solutions

★ Topographic BaseMap design

Topographic BaseMap has been designed based on simplified and flatten models of INSPIRE data .



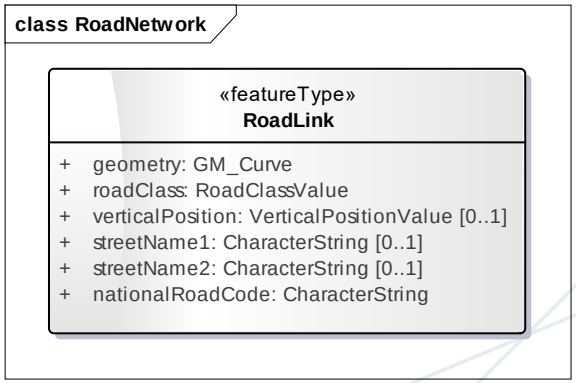
Only necessary attributes have been kept
=> To be used in the SLD

```
<se:Name>HY.PhysicalWaters.StandingWater</se:Name>
<UserStyle>
    <se:FeatureTypeStyle>
        <se:Rule>
            <se:Name>StandingWater</se:Name>
            <se:MaxScaleDenominator>72200.000000</se:MaxScaleDenominator>
            <se:PolygonSymbolizer>
                <se:Fill>
                    <se:SvgParameter name="fill">#c2e9fc</se:SvgParameter>
                </se:Fill>
            </se:PolygonSymbolizer>
        </se:Rule>
    </se:FeatureTypeStyle>
</UserStyle>
```

Alternative solutions

- Topographic BaseMap implementation (IGN)
  - For most themes, we have used INSPIRE data (if available)
  - For theme TN, we have transformed our source data (flat structure) to the BaseMap (flat) structure.

Some adaptation (roadClass) =>matching guidelines



Conclusion

- Strong interest for flatten INSPIRE data models
- Flattening INSPIRE for a given end user application is easy
  - Target model driven by the application requirements
  - Keep only useful content
  - It depends on the application
- **Flattening INSPIRE while keeping its whole content is quite difficult**
  - Different requirements
  - No simple solution