Methodology and Tool Supporting Cooperative Composition of Semantic Domain Models for Experts and Developers

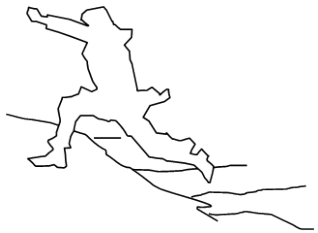
Master of Science Thesis

Maciej Rząsa

AGH University of Science and Technology,
Kraków

22.09.2011

Supervisor: Marian Bubak, PhD
Reviewer: Maciej Zygmnut, PhD, ABB Kraków
Consultancy: Tomasz Gubała, ACC Cyfronet Kraków



Background

- Motivation: cooperation participants characteristics
- Objectives: requirements for a methodology and a tool

Methodology

- Metamodel: framework of domain description
- Cooperation method: iterative model building

Tool

- Requirements and functionality
- Architecture and implementation

Evaluation: experimental modelling sessions

- Introduction: outcome types
- Flood forecasting: controlled experiment
- Road designing: full experiment

Conclusions

An expert

- posses extensive domain knowledge
- describes the domain using specialized language
- e.g. scientist, businessman

Chaos and Deterministic *versus* Stochastic Non-linear Modelling

By MARTIN CASDAGLI†

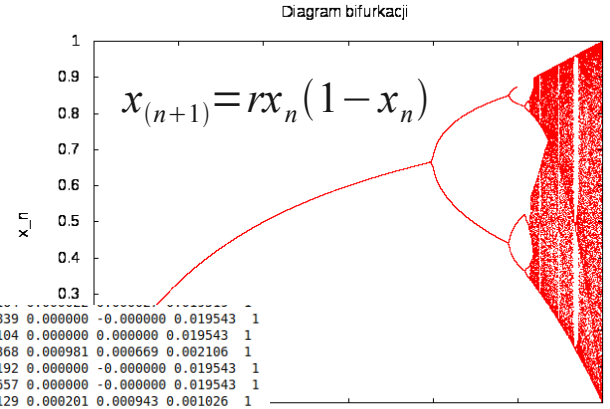
Santa Fe Institute, USA

[*Read before* The Royal Statistical Society *at a meeting on Chaos organized by the* Research Section *on Wednesday, October 16th, 1991*]

SUMMARY

An exploratory technique is introduced for investigating how much of the irregularity in an aperiodic time series is due to low dimensional chaotic dynamics, as opposed to stochastic or high dimensional dynamics. Non-linear models are constructed with a variable smoothing parameter which at c
extreme defines a line
as a function of the
generating the time se
occurring time series

Keywords: CHAOS; DIMENS



6.4902	0.1006	13.0839	0.000000	-0.000000	0.019543	1
14.7595	1.1225	5.1104	0.000000	0.000000	0.019543	1
3.0317	0.6546	15.1368	0.000981	0.000669	0.002106	1
12.0500	1.2905	3.3192	0.000000	-0.000000	0.019543	1
3.5165	0.6401	7.2657	0.000000	-0.000000	0.019543	1
6.3139	0.8445	15.0129	0.000201	0.000943	0.001026	1
13.4882	0.9691	7.2071	0.000000	0.000000	0.019543	1
7.8366	0.7876	9.7438	0.000000	0.000000	0.019543	1
9.6512	0.8143	12.4799	0.000000	0.000000	0.019543	1
5.3225	0.7701	14.3578	0.000030	0.000042	0.019549	1
10.1032	0.0052	9.1848	0.000000	-0.000000	0.019543	1
14.7004	0.0931	13.8717	0.000000	-0.000000	0.019543	1
5.1978	0.0277	15.0150	0.001167	0.001477	0.006499	1

A developer

- uses programming languages, formal modelling tools and technical specifications
- needs a domain description that could be easily mapped onto programming language

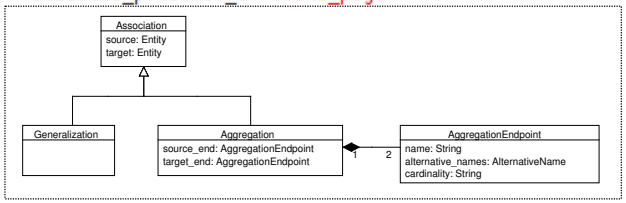
```
class Entity < ActiveRecord::Base
  unloadable
  include Aliasable

  attr_accessor :position_x, :position_y, :row, :col

  belongs_to :wiki_page
  has_many  :at
  has_many :sou
    :class_name
    :foreign_ke
    :dependent
  has_many :tar
    :class_name
    :foreign_ke
    :dependent

  has_many :chi
    :class_name
    :foreign_ke

  after_destroy

  validates_presence_of :wiki_page
```

Internet Engineering Task Force (IETF) P. Saint-Andre
Request for Comments: 6120 Cisco
Obsoletes: 3920 March 2011
Category: Standards Track
ISSN: 2070-1721

Extensible Messaging and Presence Protocol (XMPP): Core

Abstract

The Extensible Messaging and Presence Protocol (XMPP) is an application profile of the Extensible Markup Language (XML) that enables the near-real-time exchange of structured yet extensible data between any two or more network entities. This document defines XMPP's core protocol methods: setup and teardown of XML streams, channel encryption, authentication, error handling, and communication primitives for messaging, network availability ("presence"), and request-response interactions. This document obsoletes RFC 3920.



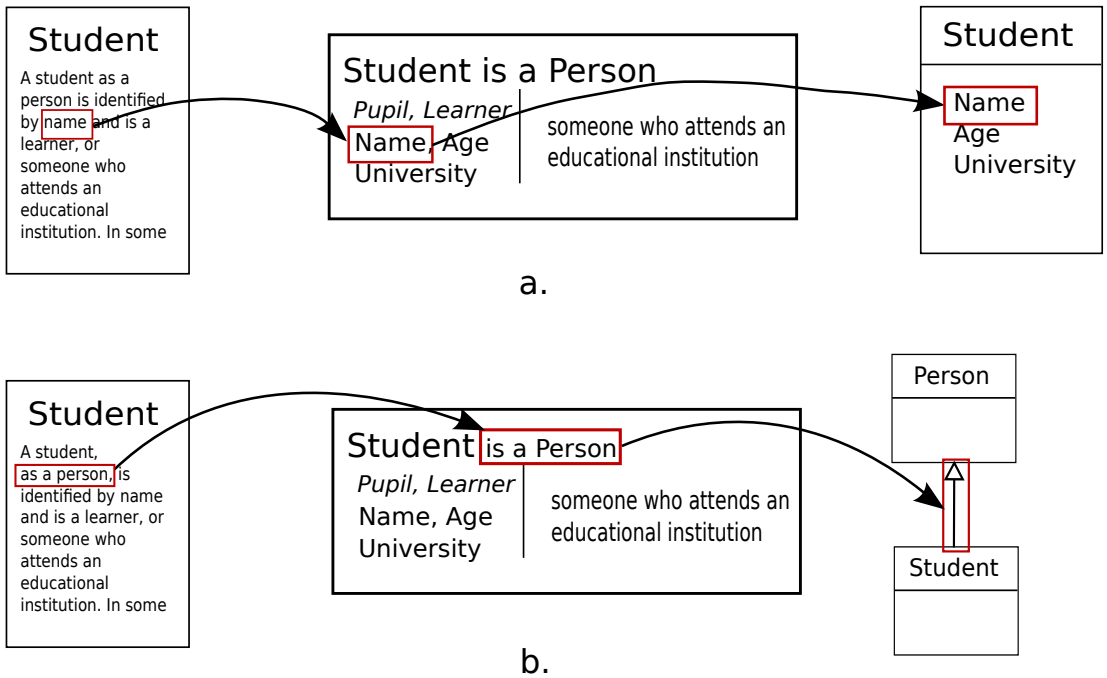
Objectives

1. The aims of this work:
   - transmission knowledge from an expert to a developer in convenient way
   - enable effective knowledge validation
2. Approach: semantic domain model – common (*ubiquitous*) language for collaborators:
   - consists in domain concepts (expert language)
   - posses structure convenient for formal modelling (developer language)
   - understandable both by humans and computers
3. Solution:
   - a methodology of the cooperation
     - the aim: knowledge passing
     - the outcome: semantic domain model
     - users: developers and experts
   - a tool
     - support for the methodology
     - validation of the methodology
     - easy to use for non-computer science users

Metamodel elements: structuring knowledge

Metamodel is a framework used for mapping domain concepts onto programming structures.

- concept definition $\mapsto$ *Entity* $\mapsto$ `Class`
- simple feature $\mapsto$ *Attribute* $\mapsto$ `Field`
- interconnection $\mapsto$ *Association*
  - *a kind of* association $\mapsto$ *Generalization* $\mapsto$ `Inheritance`
  - complex features $\mapsto$ *Relation* $\mapsto$ `Association`



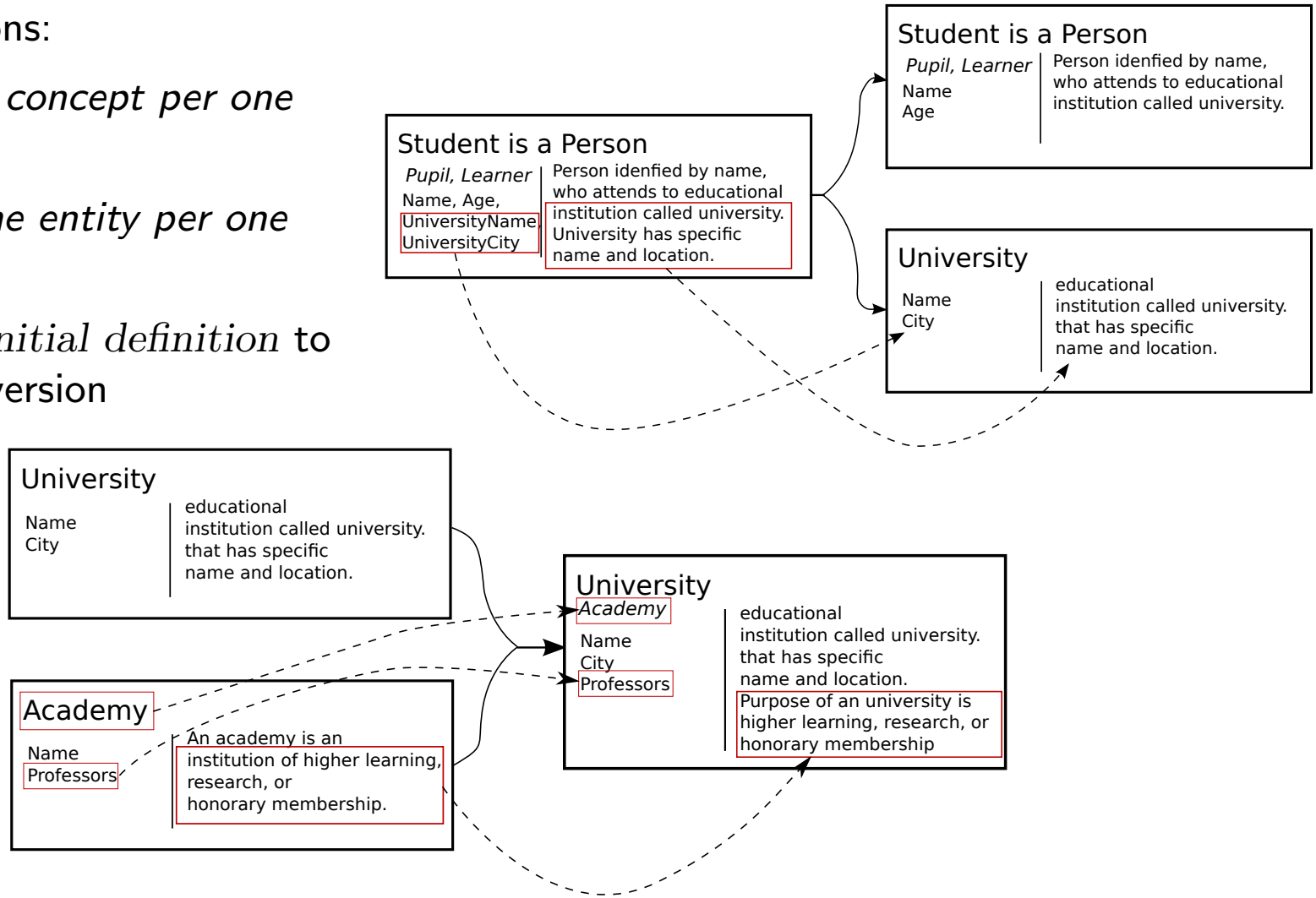
Mapping examples: (a) *attribute*; (b) *generalization*.

Metamodel transitions: evolving the structure

Model transitions (*split*, *merge* and *extract*) enable evolution of domain description to obtain desired level of simplicity and expressiveness.

Aims of transitions:

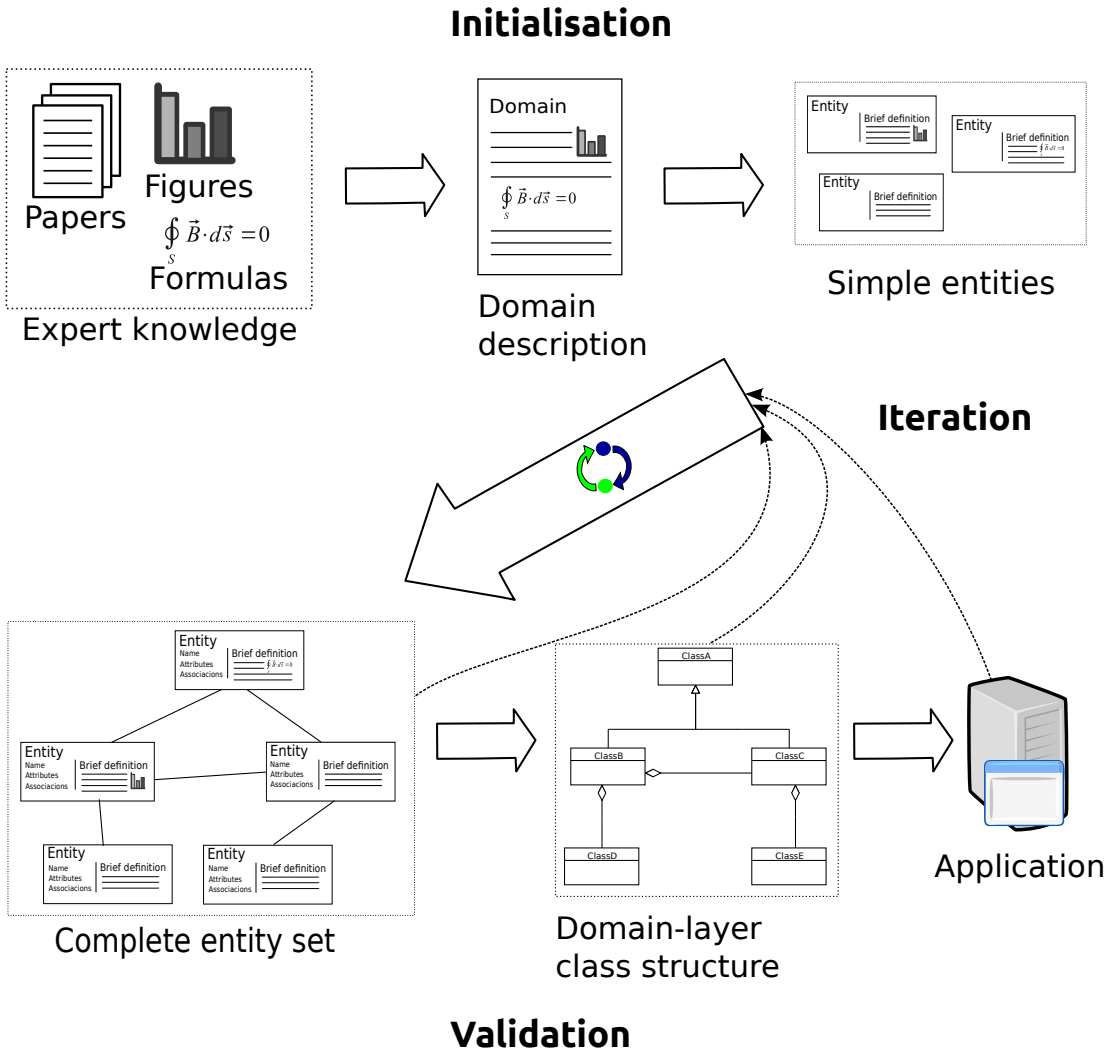
- *split* – *one concept per one entity*
- *merge* – *one entity per one concept*
- *extract* – *initial definition* to *entity* conversion



Methodology: iterative model elaboration

After defining of domain basics (*initial definition*), collaborators iteratively improve model to obtain sufficient domain description.

- initialisation: domain basics establishment
  - (E) main concepts definition
  - (D) simple entities extraction
- iteration: correcting and adding details
  - (E) review of model elements
  - (D) model transformation
- stop condition: consistent model
  - (E) validation of domain correctness
  - (D) validation of model structure



Methodology overview

Methodology summary: roles and goals

1. Participants' roles
   - expert
     - defining domain concepts
     - correcting developer work
   - developer
     - extracting and transforming
     - consistency checking
2. Model state after the cooperation process:

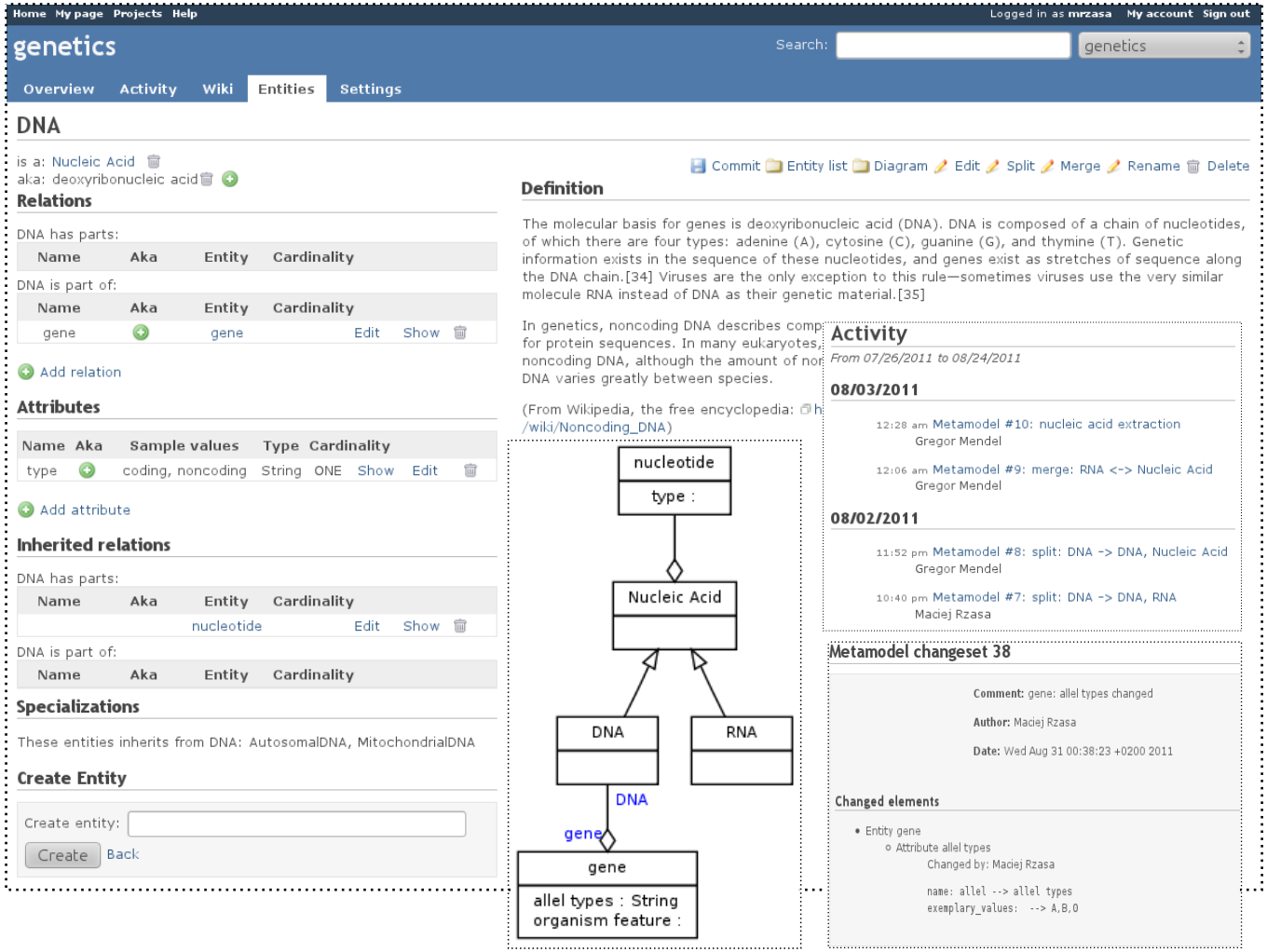
Cohesion	state of a single entity: *one entity per a concept and one concept per entity*
Completeness	state of a whole model: the model describes a strictly limited part of the domain
Consistency	state of model: elements of the model do not contradict each other

Whereas cohesion and completeness are desirable, consistency is necessary condition of model correctness.

Domain Model Builder: methodology implementation

Main functions of the DMB:

- metamodel implementation (elements and transitions)
- methodology support (functions for defining and evolving knowledge)
- model visualisation
- cooperation process logging
- changes tracking

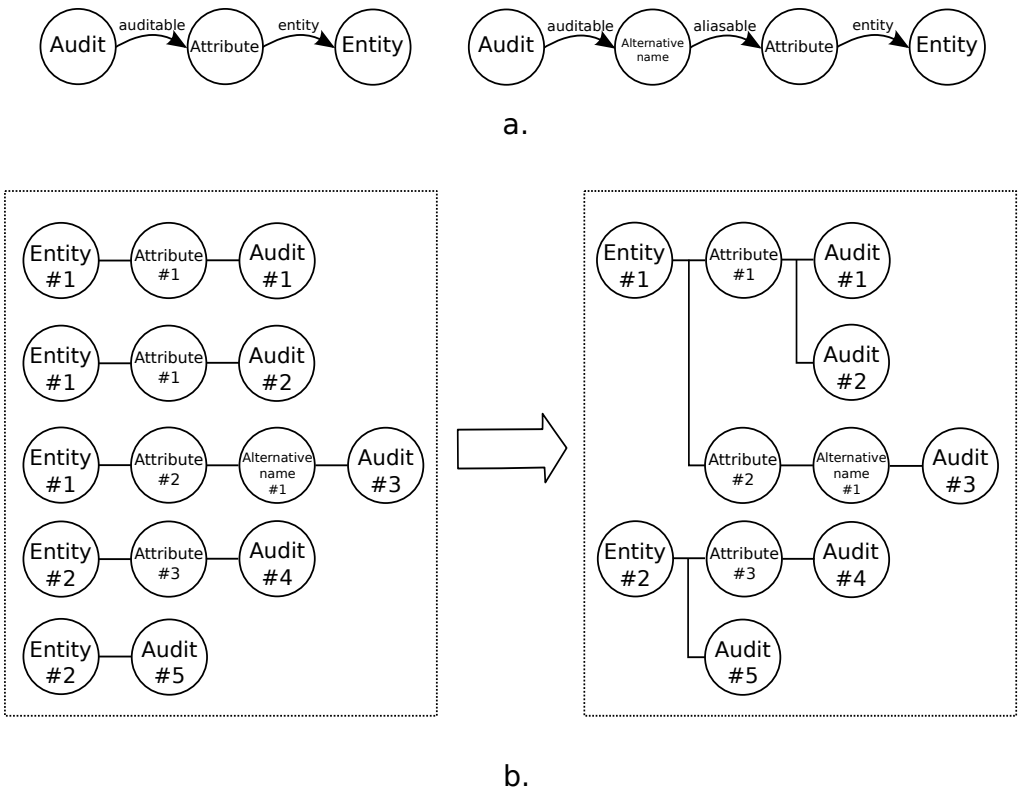


Functionality: entity page, diagram, activity log, changes details.

Domain Model Builder internals

The DMB is a web application implemented using Ruby on Rails as a Redmine plugin.

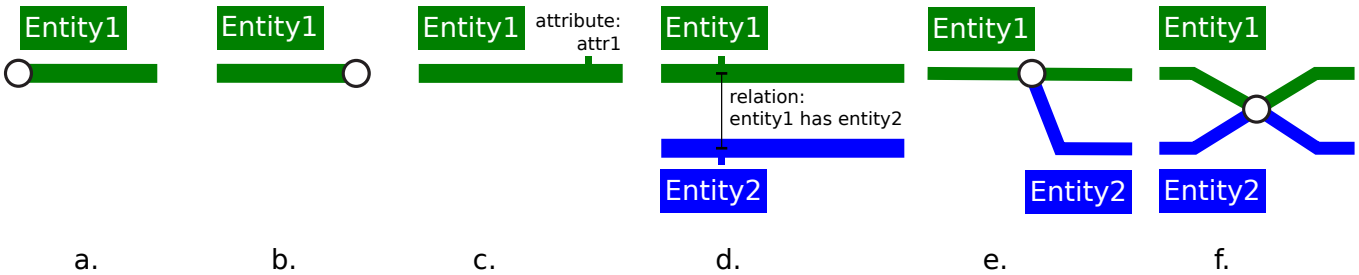
- DMB – Redmine plugin:
  - Redmine – web application for project management;
  - DMB uses Redmine functions (wiki, activity log, user management)
  - DMB adapts Redmine classes (patching them with *mixins*)
- metamodel: Rails classes
- diagram: generated with Graphviz
- changes tracking (cf. Figure): `acts_as_audited`, `acts_as_paranoid`, `acts_as_revisionable` (Rails plugins)



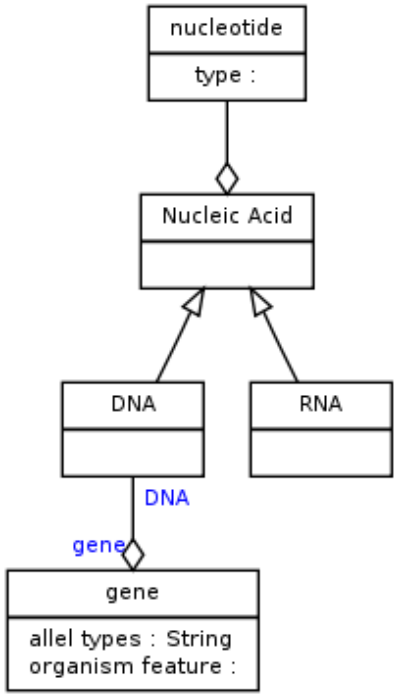
Building of changes forest: (a) linking to entity; (b) merging branches.

Evaluation: cooperation with experts

1. Experiments: modelling sessions involving experts of two different domains.
2. Outcome: a domain model and a cooperation course:
   - model represented on a diagram
   - cooperation course visualised on a metro map infographics



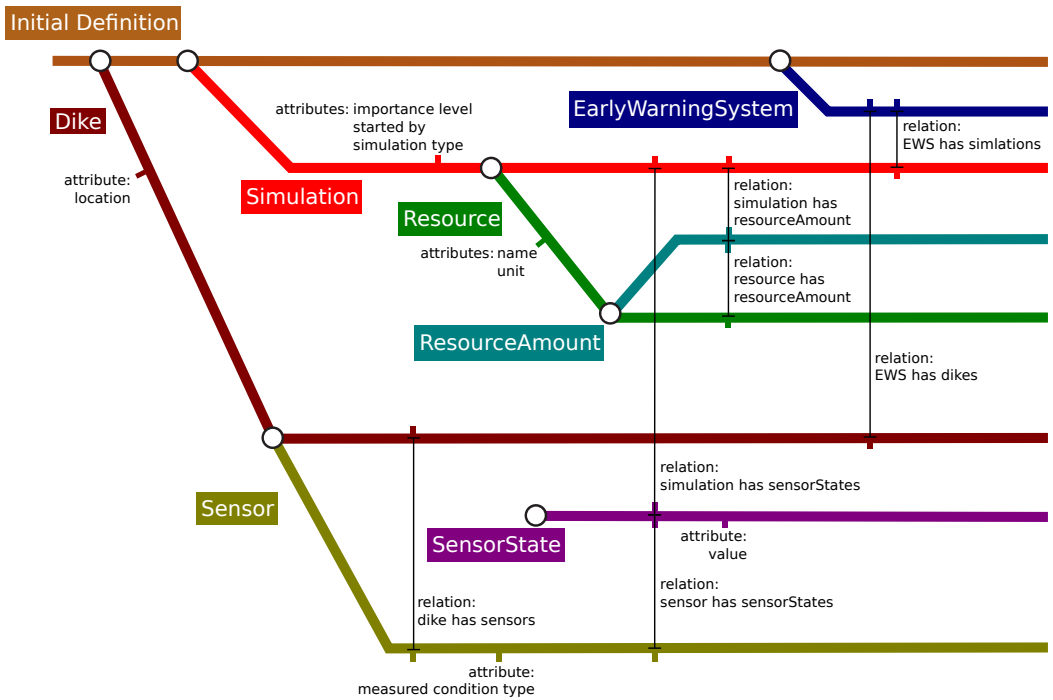
Cooperation diagram: (a) *entity* created; (b) *entity* deleted; (c) *attribute* added; (d) *relation* added; (e) *split*; (f) *merge*.



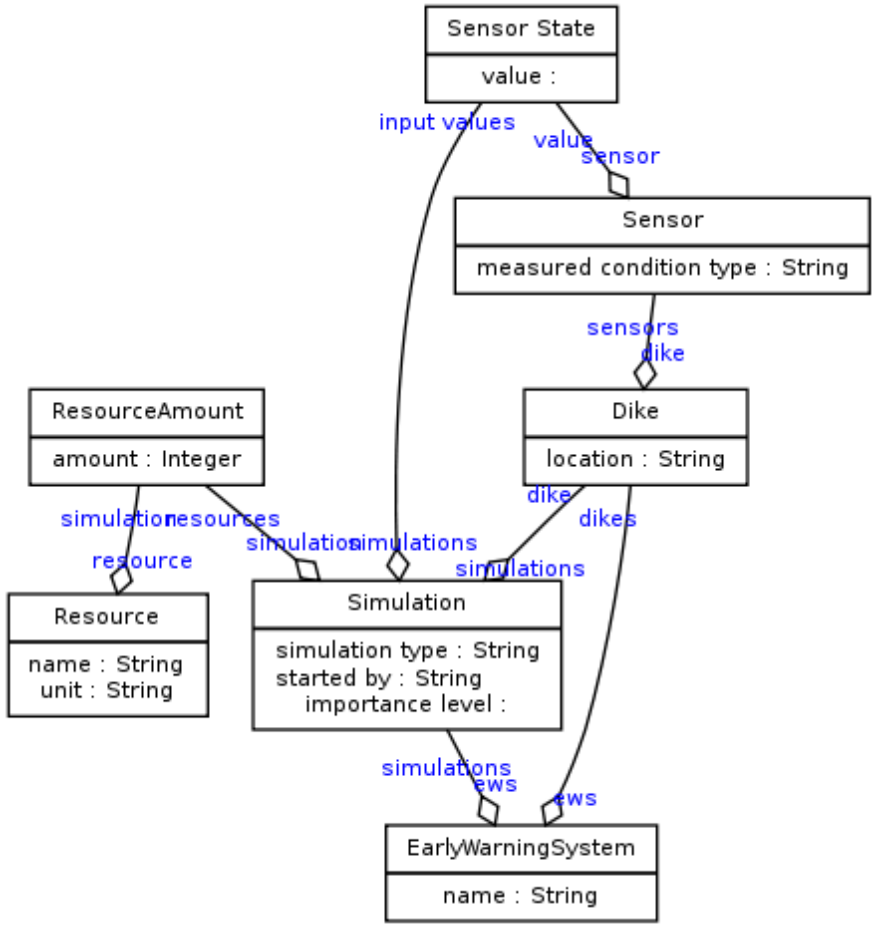
A domain model diagram

Controlled experiment: flood forecasting

- modelling domain of existing system (UrbanFlood)
- expert: UrbanFlood developer
- cooperation: personal + DMB + email
- expert's opinion
  - \+ easy way of knowledge extraction
  - \+ readable *entities*
  - − presentation, usability



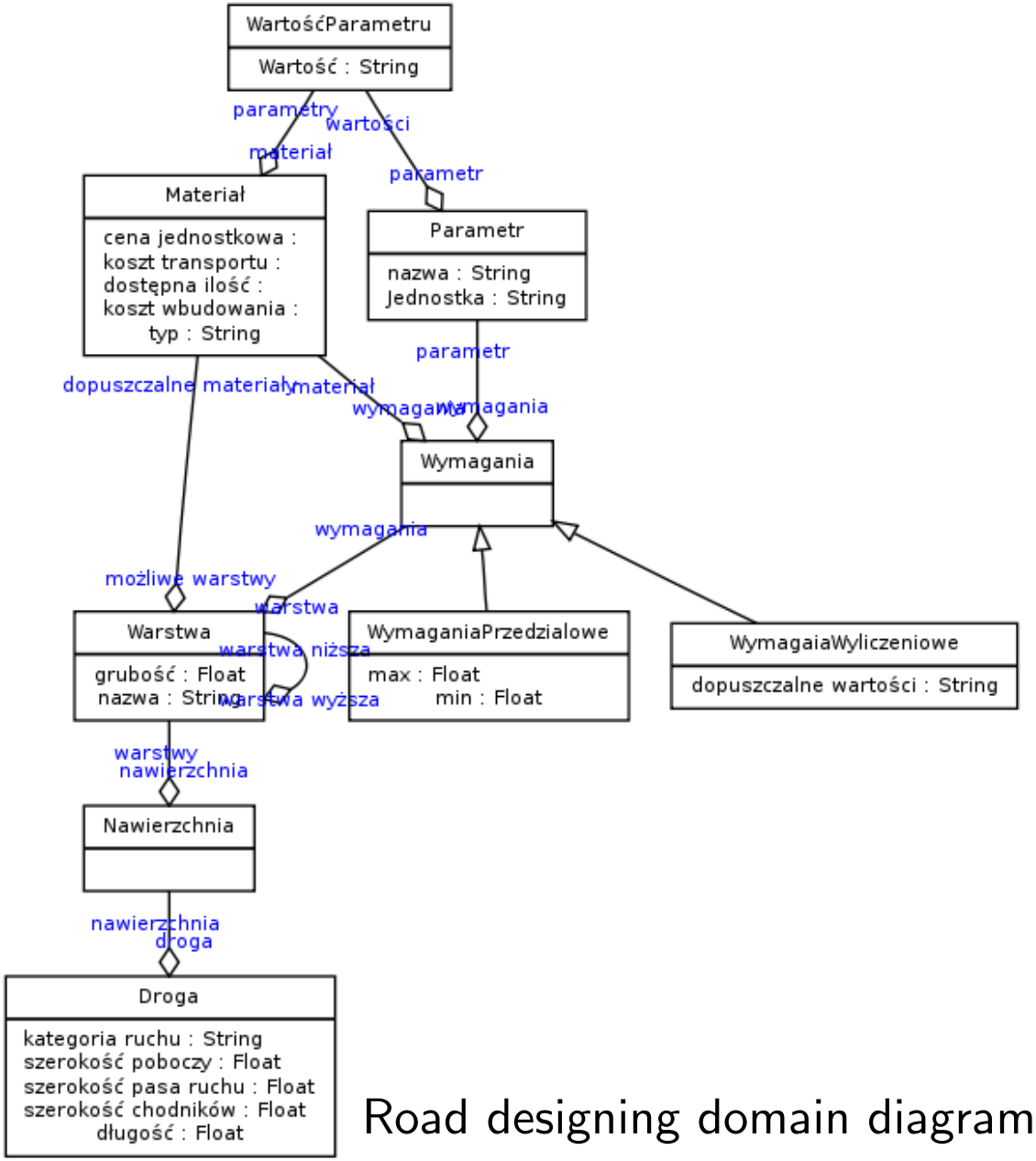
UrbanFlood modelling course



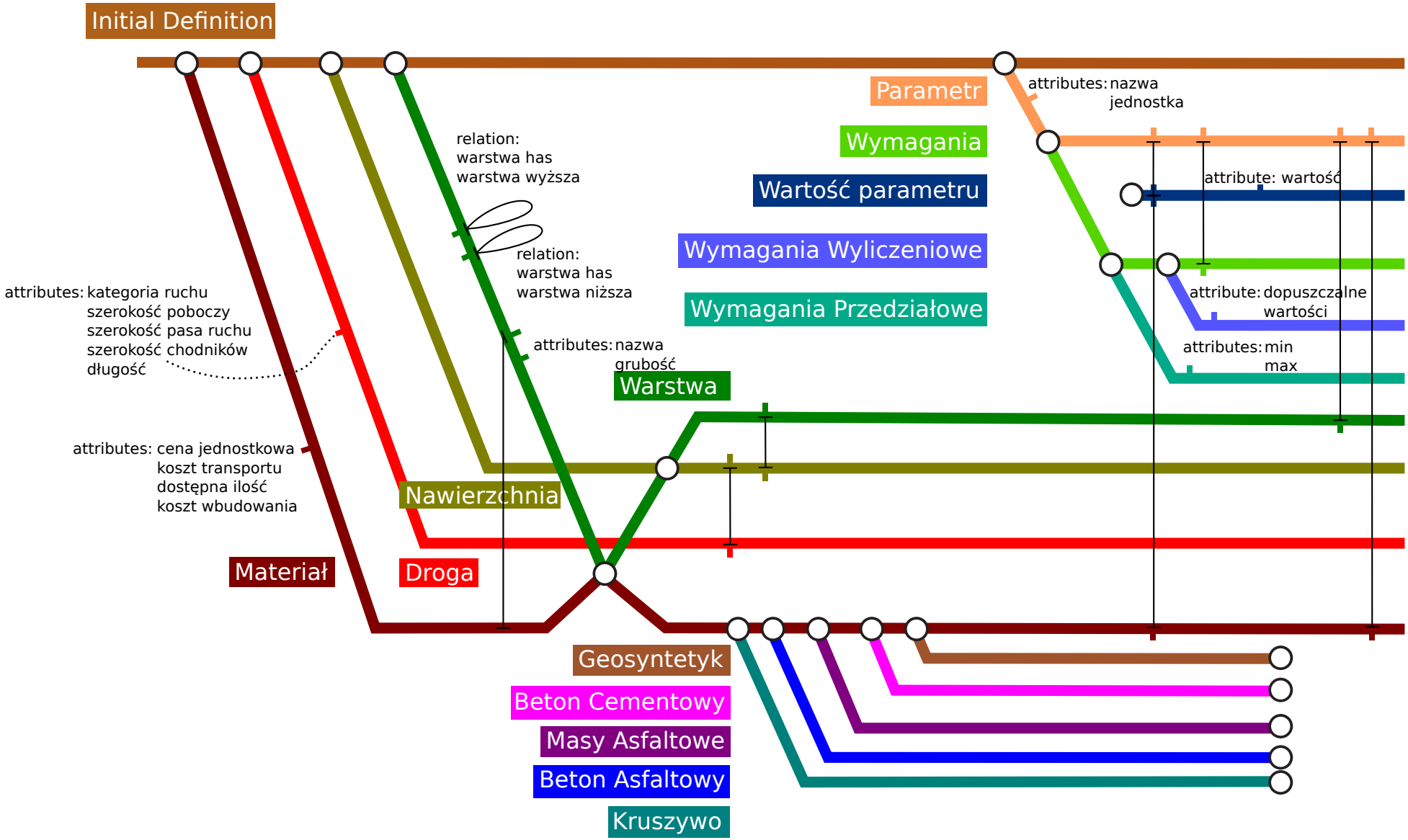
UrbanFlood domain diagram

Full experiment: road designing

- modelling domain of road surface design
- expert: civil engineer
- cooperation: DMB (mainly *initial page*) + email
- expert's opinion
  - \+ method well suited for cooperation involving engineers
  - \+ precise transmission and verification of knowledge
  - \+ *entities* and diagram understandable
  - – lack of feedback questions
  - – need of place for discussion



Road designing domain diagram



Road design modelling course

Summary

Results

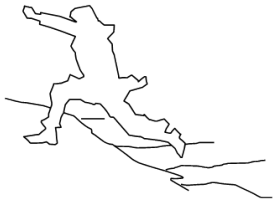
1. Methodology
   - transmission of domain knowledge and its verification
   - outcome: semantic model that is understandable for human (definitions) and machine (formal structure)
2. Tool
   - implementation of the methodology
   - evaluation of the methodology
3. Experiments proved worth of this approach.

Future work

1. Methodology
   - representation of an *entity* attached to a relation
   - *merge entity* and *initial definition*
   - more experiments
2. Tool
   - code generation
   - enhanced versioning
   - model-focused discussion
   - usability improvements

Please visit project websites:

- DMB source code:
  `https://gforge.cyfronet.pl/projects/dom-comp/`
- test deployment of the DMB: `http://gandalf.zagorz.net/dmb`



UrbanFlood

This thesis is related with the UrbanFlood, a project funded under the EU Seventh Framework Programme, Theme ICT-2009.6.4a. ICT for Environmental Services and Climate Change Adaption. Grant agreement no. 248767.

Parts of this work were elaborated as a final project during Erasmus scholarship at Universidad Politecnica de Catalunya in Barcelona under supervision of Pau Fernandez.
The project was defended with mark 8/10.