# MDE
# to the people

Jordi Cabot

jordi.cabot@inria.fr

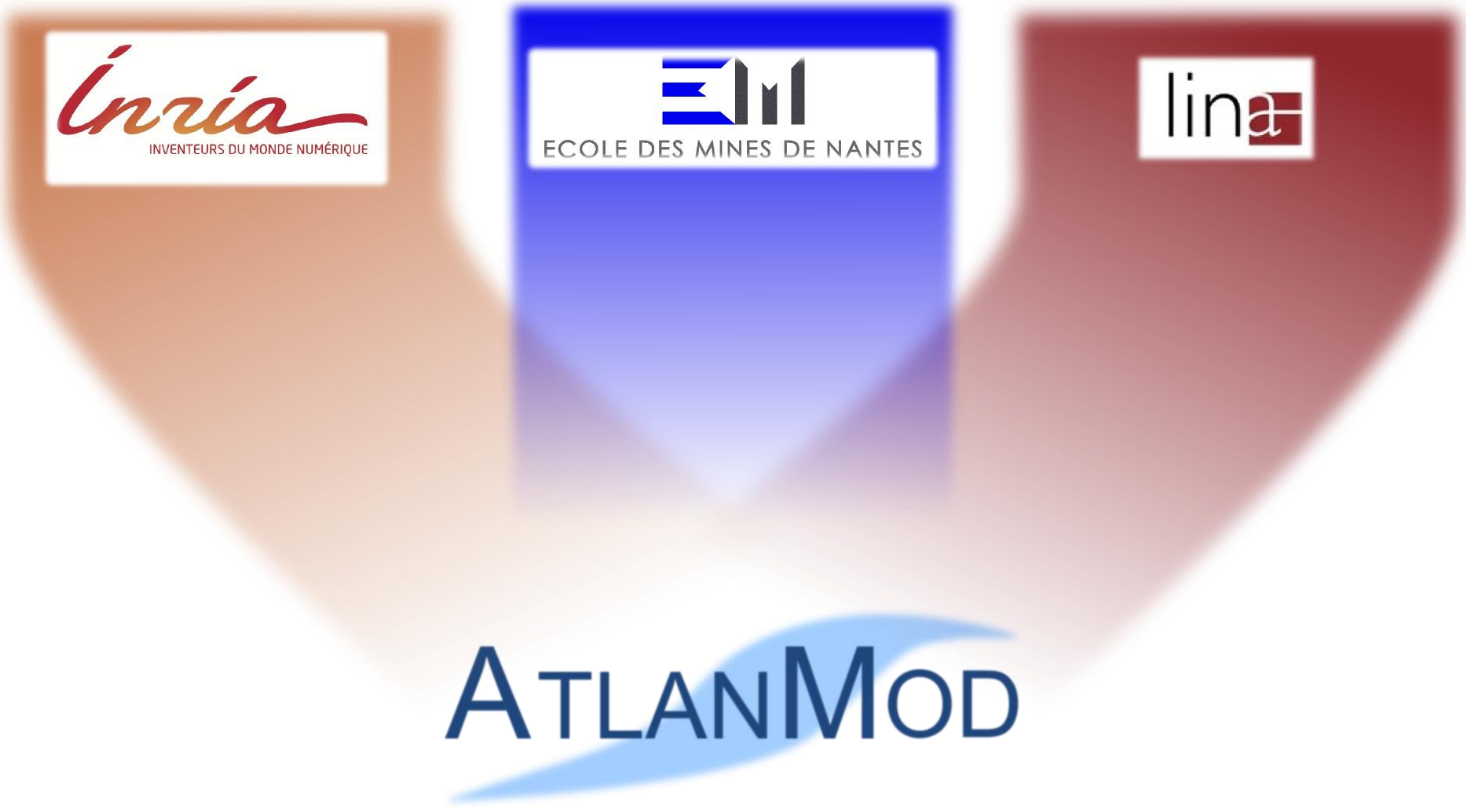
Barcelona – UPC
22/03/2013

AtlanMod

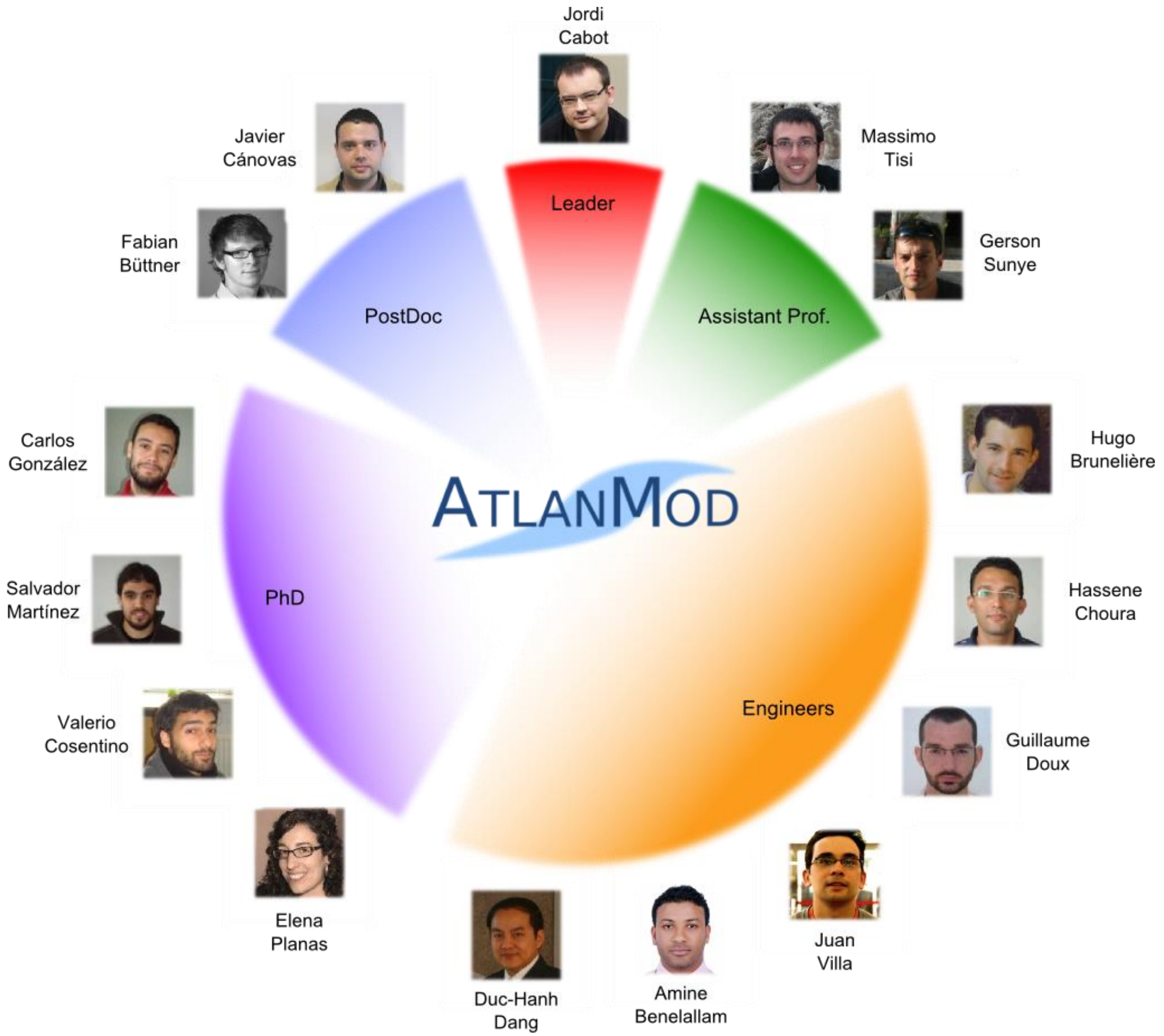
# AtlanMod

ECOLE DES MINES DE NANTES
Nantes
Vannes
Saint-Nazaire
Angers
Tours
Dijon
Basel
Poitiers
France
Lausanne
Genève
Schweiz
Suisse
Svizzera
(Switzerland)
Bourg-en-Bresse
Roanne
La Rochelle
Bay of Biscay
Bordeaux
Agen
Ferrol
Viveiro
Gijón
Santander
Donostia-San Sebastián
A Coruña
Oviedo
Llanes
Bilbao
Santiago de Compostela
Lugo
León
Pamplona
Vitoria-Gasteiz
Andorra
Ourense
Burgos
Vigo
Valladolid
Zaragoza
Lleida
Sabadell
Braga
Barcelona
Porto
Salamanca
Tarragona
Ávila
Madrid
Coimbra
Balearic Sea
Ciutadella
Portugal
España
(Spain)
València
Palma
Maó
Albacete
Lisboa
Badajoz
Xàbia
Setúbal
Alacant
Córdoba
Murcia
Sevilla
Écija
Cartagena
Granada
Portimão
Huelva
Utrera
Jerez de la Frontera
Málaga
Gibraltar
Alboran Sea
Oran
Relizane
Alger
Tizi Ouzou
Béjaïa
Jijel
Annaba
Bouira
Sétif
Constantine
Bordj Bou

# The team


Inria
INVENTEURS DU MONDE NUMÉRIQUE
ECOLE DES MINES DE NANTES
lina
ATLANMOD

# The people



Jordi Cabot

Leader

Massimo Tisi

Gerson Sunye

Assistant Prof.

Javier Cánovas

Fabian Büttner

PostDoc

AtlanMod

Hugo Brunelière

Hassene Choura

Guillaume Doux

Juan Villa

Amine Benelallam

Duc-Hanh Dang

Engineers

Carlos González

Salvador Martínez

Valerio Cosentino

Elena Planas

PhD

# Our Research

# Research

- MDE as a software engineering paradigm to improve software production, evolution and operation.
- MDE based on the rigorous use of  software models and model manipulation operations.
- AtlanMod researches core MDE techniques and their adaptation to specially relevant industrial challenges.

# Research

- **Application-driven research**
  - Constant collaboration with companies

- **Open source** community via **Eclipse**
  - MoDisco, AM3, EMF Facet, ATL, AMW, etc

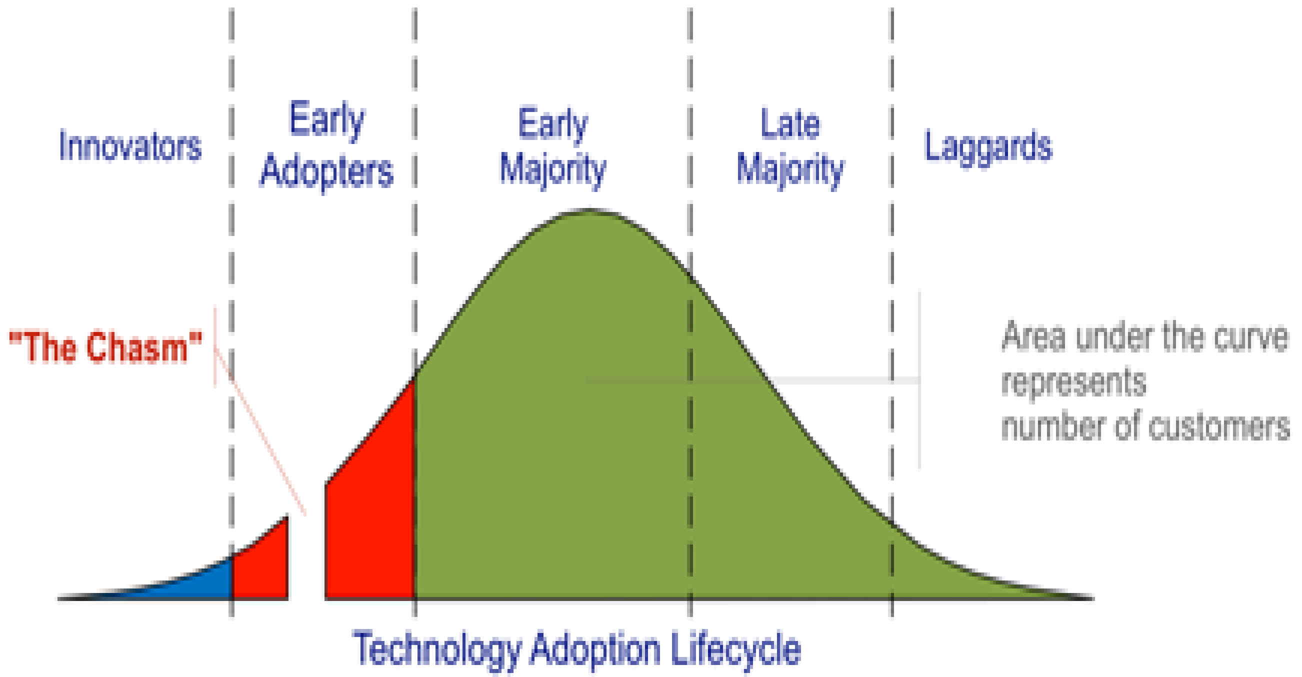
# We have advanced a lot on the core techniques

- UML and profiles
- DSLs & Language workbenches
- Model-to-model and model-to-text transformations
- Model management and evolution
- …

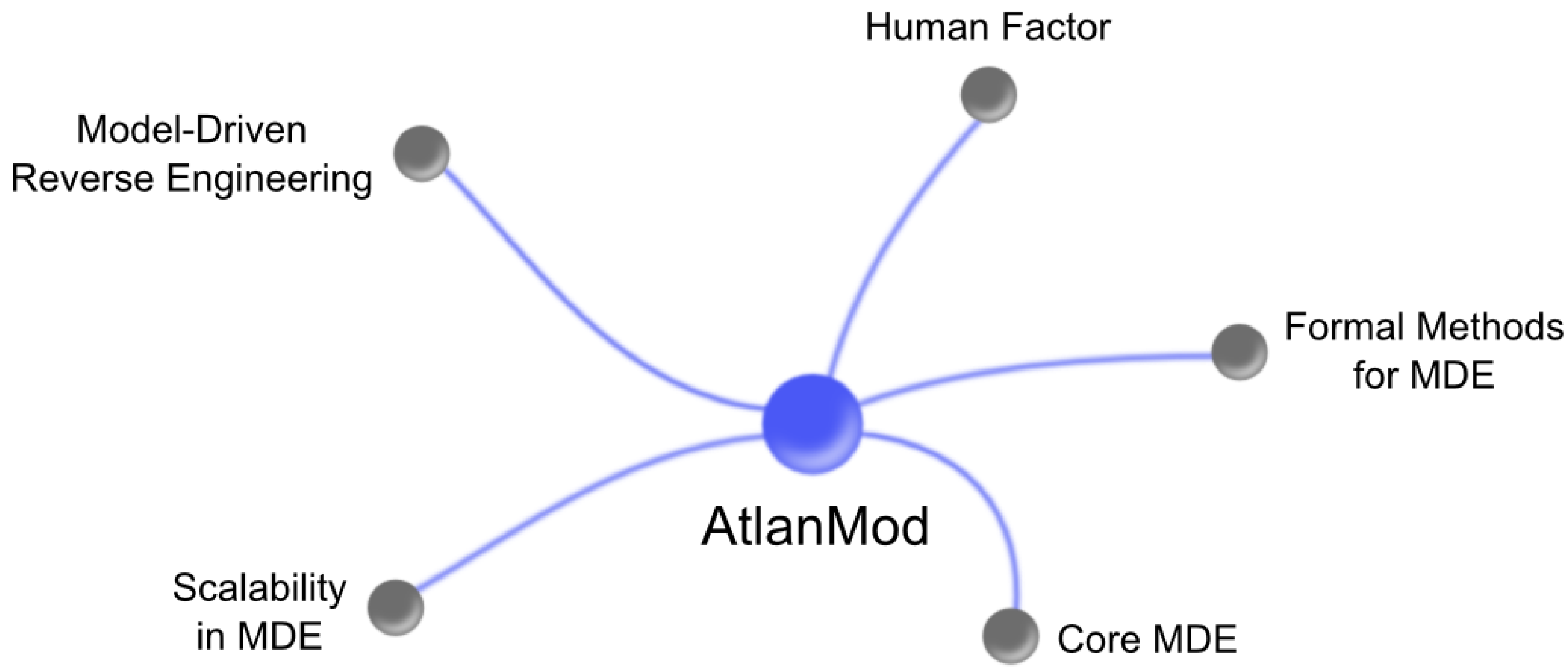
# But it’s clearly not enough

- Modeling will be commonplace in 3 years time – S. Mellor

Though he is giving the same answer for the last 20 years



# Our place in MDE



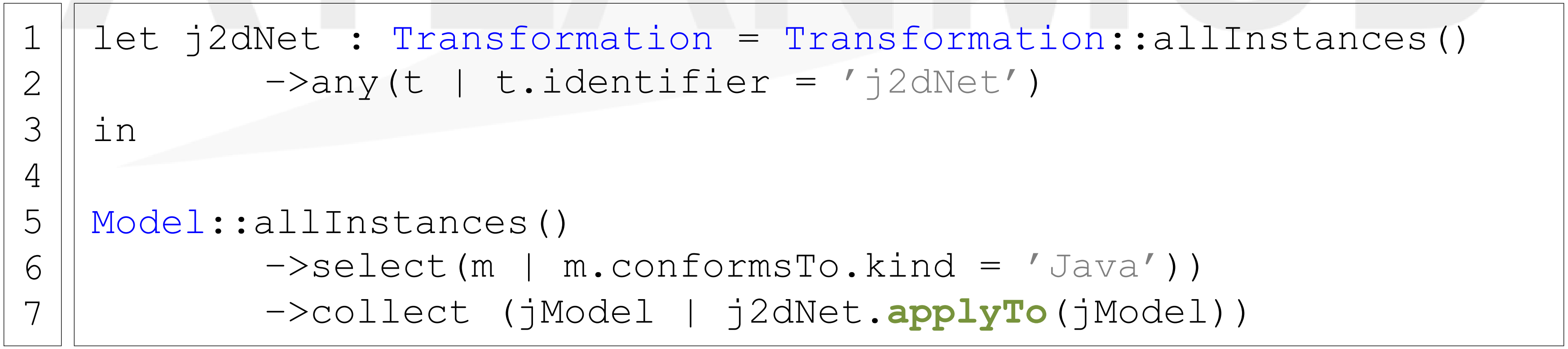
# Core MDE

- **Model Transformations**
  - Refactoring of transformations
  - Bidirectional transformations
  - Reactive ATL

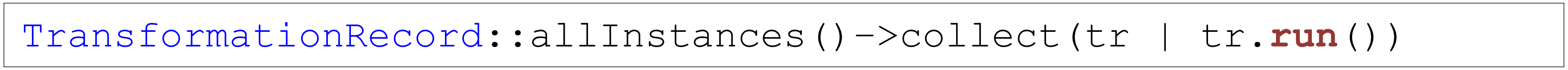
- **Model management**
  - EMF Profiles
  - DSL for querying and manipulating model repositories

# Example

- Model to Model transformations (M2M)



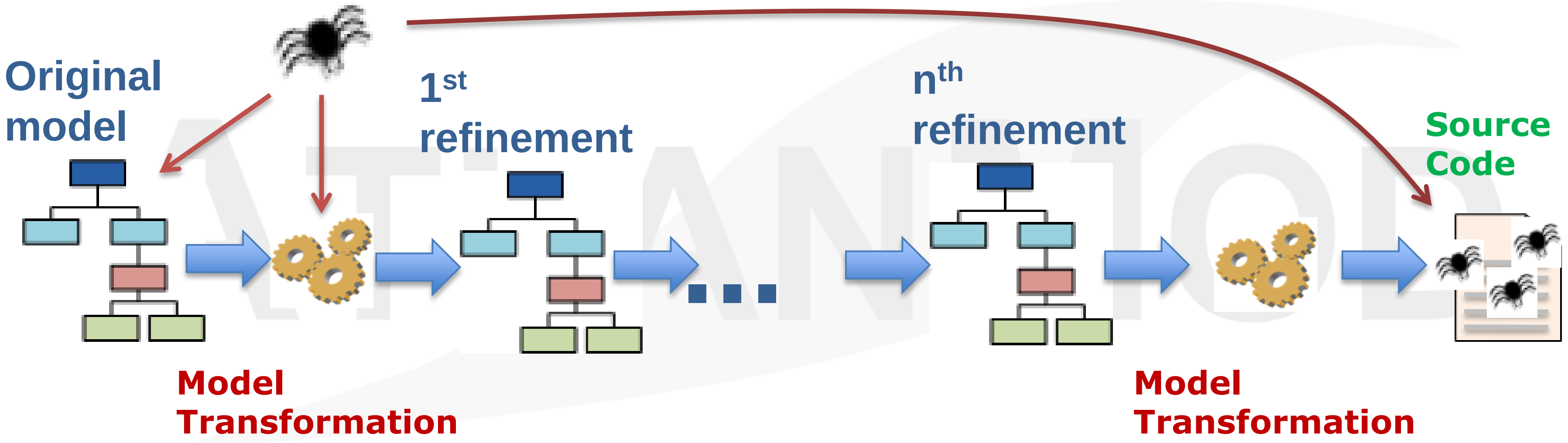
```
let j2dNet : Transformation = Transformation::allInstances()
        ->any(t | t.identifier = 'j2dNet')
in

Model::allInstances()
        ->select(m | m.conformsTo.kind = 'Java'))
        ->collect (jModel | j2dNet.applyTo(jModel))
```



```
TransformationRecord::allInstances()->collect(tr | tr.run())
```

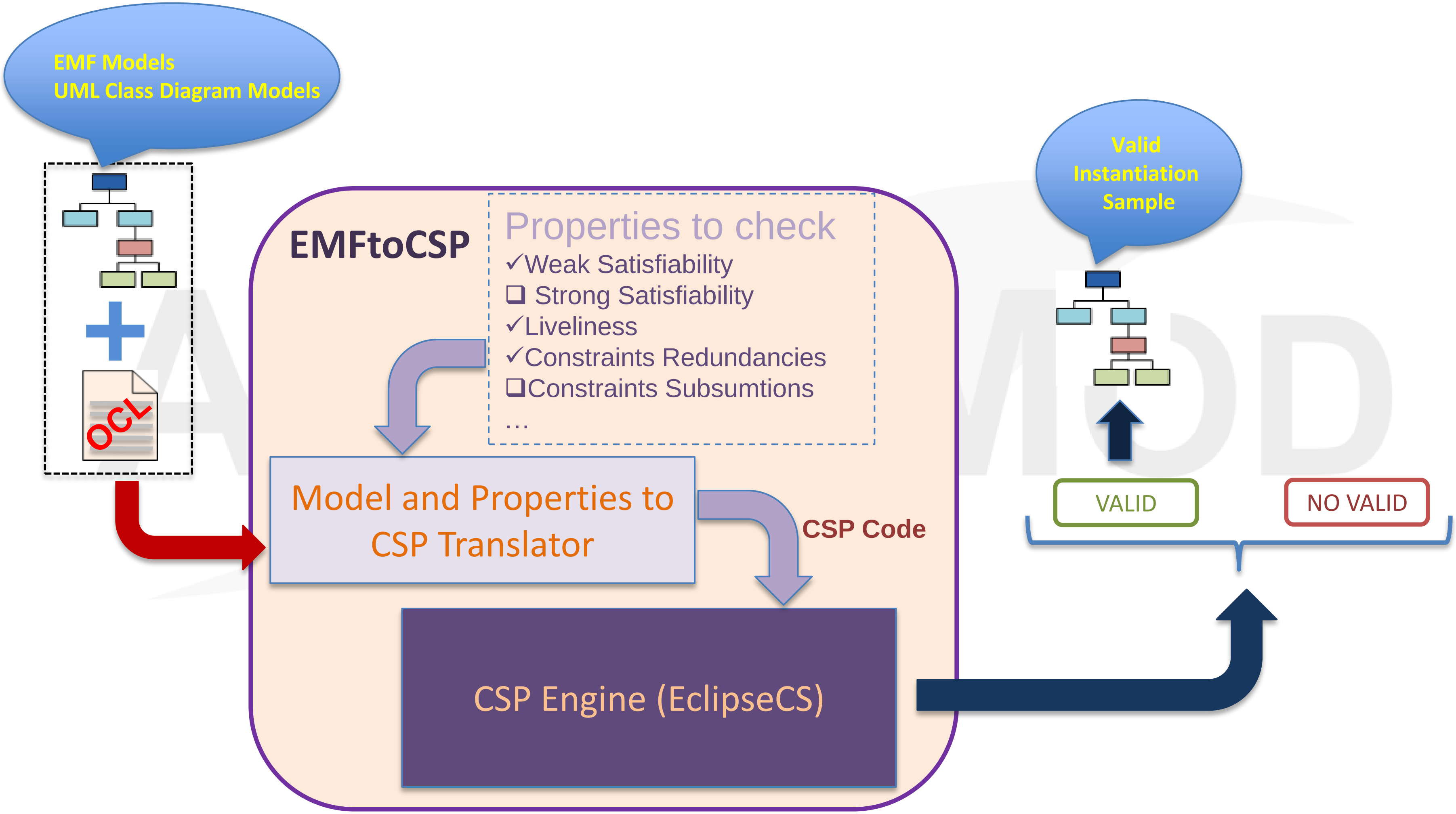
# Quality - Importance

MDE-based software development process



Errors in models will lead to errors in the resulting software

# Quality - EMFtoCSP



EMF Models
UML Class Diagram Models

EMFtoCSP

Properties to check

✓Weak Satisfiability

❑ Strong Satisfiability

✓Liveliness

✓Constraints Redundancies

❑Constraints Subsumtions

…

Model and Properties to CSP Translator

CSP Code

CSP Engine (EclipseCS)

Valid Instantiation Sample

VALID

NO VALID

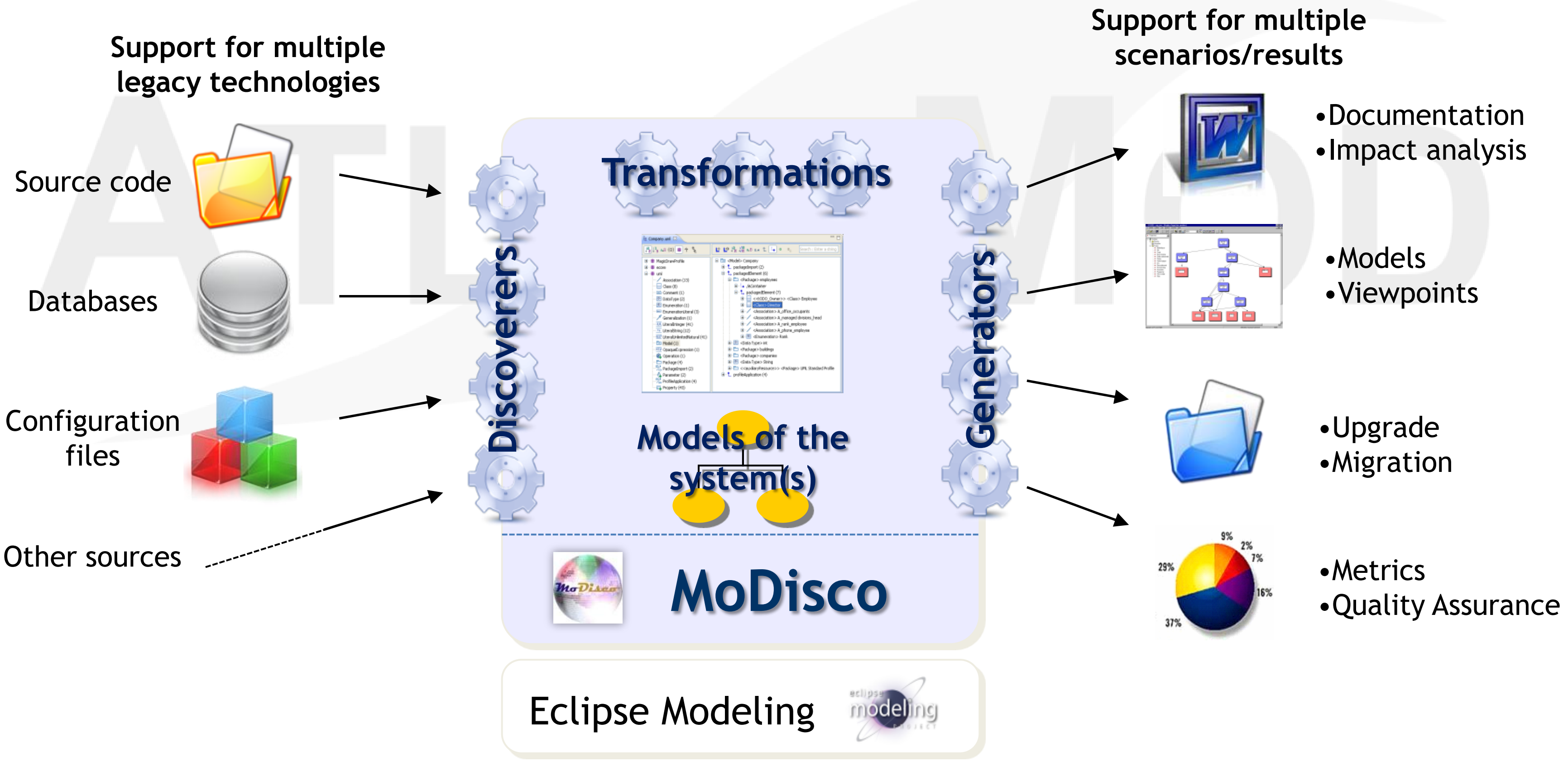
# Quality

- Verification of MT
  - With CSPs but also SMTs

- Testing of MT

# Reverse Engineering

- As old as CS itself. Always relevant
- First level models : zero information loss



Support for multiple legacy technologies

Source code

Databases

Configuration files

Other sources

Discoverers

Transformations

Models of the system(s)

Generators

MoDisco

Eclipse Modeling

Support for multiple scenarios/results

- Documentation
- Impact analysis

- Models
- Viewpoints

- Upgrade
- Migration

- Metrics
- Quality Assurance

# Reverse Engineering

- Reverse engineering of security polices
- Reverse engineering of business rules
- Moving applications to the cloud

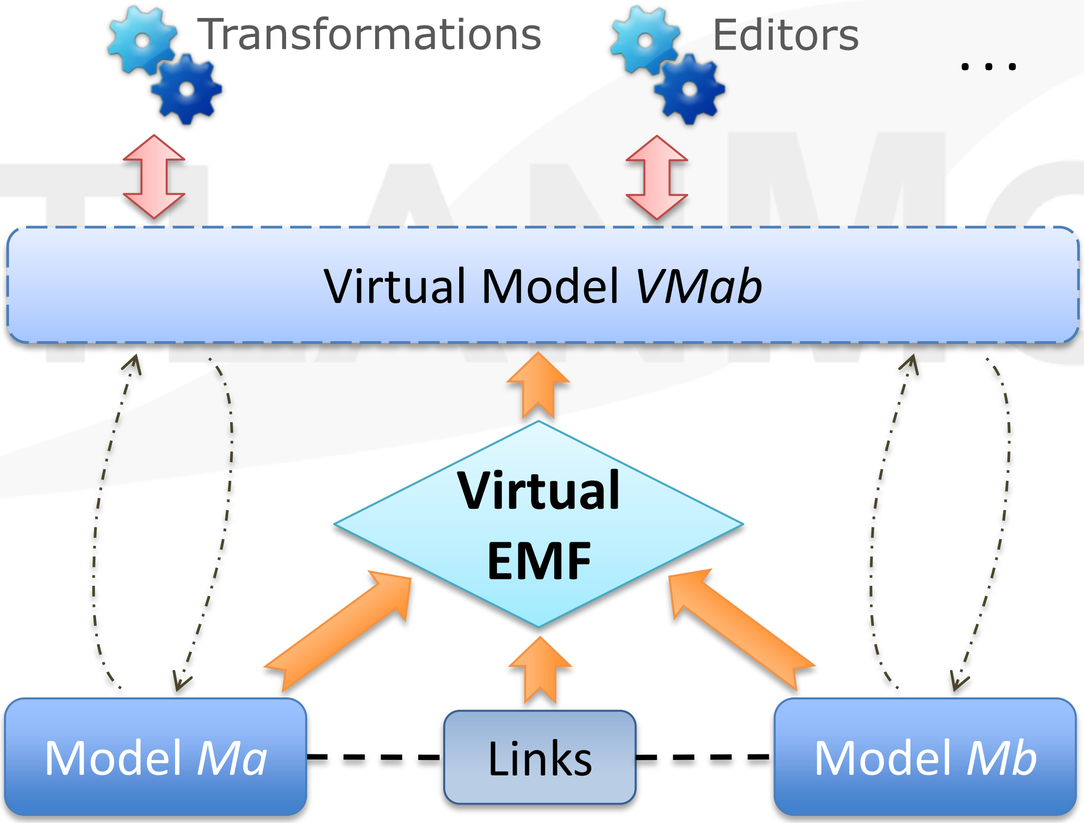
# Scalability

- Scalability important both at the model (loading very large models) and model manipulations level (executing complex transformations on large models)

- Key problem in industrial scenarios but far from a trivial one

# Virtual Models (i)

- *"a virtual model is a model whose (virtual) model elements are proxies to elements contained in other models"*



# Scalability

- Scalability important both at the model (loading very large models) and model manipulations level (executing complex transformations on large models)

- Key problem in industrial scenarios but far from a trivial one

# Scalability for MT

- Incremental ATL
- Lazy ATL
- Parallel ATL

# Human Factors (the DSL case)

# In General

- Research is not aligned with the real needs of end-users
  - Architects and NFRs
  - Ikerlan
- This is a very important problem when creating DSLs
  - Quality of DSL = user experience
  - Evaluating user experience is a challenging task
  - We cannot develop specific techniques for each different DSL
  - Need the participation of users

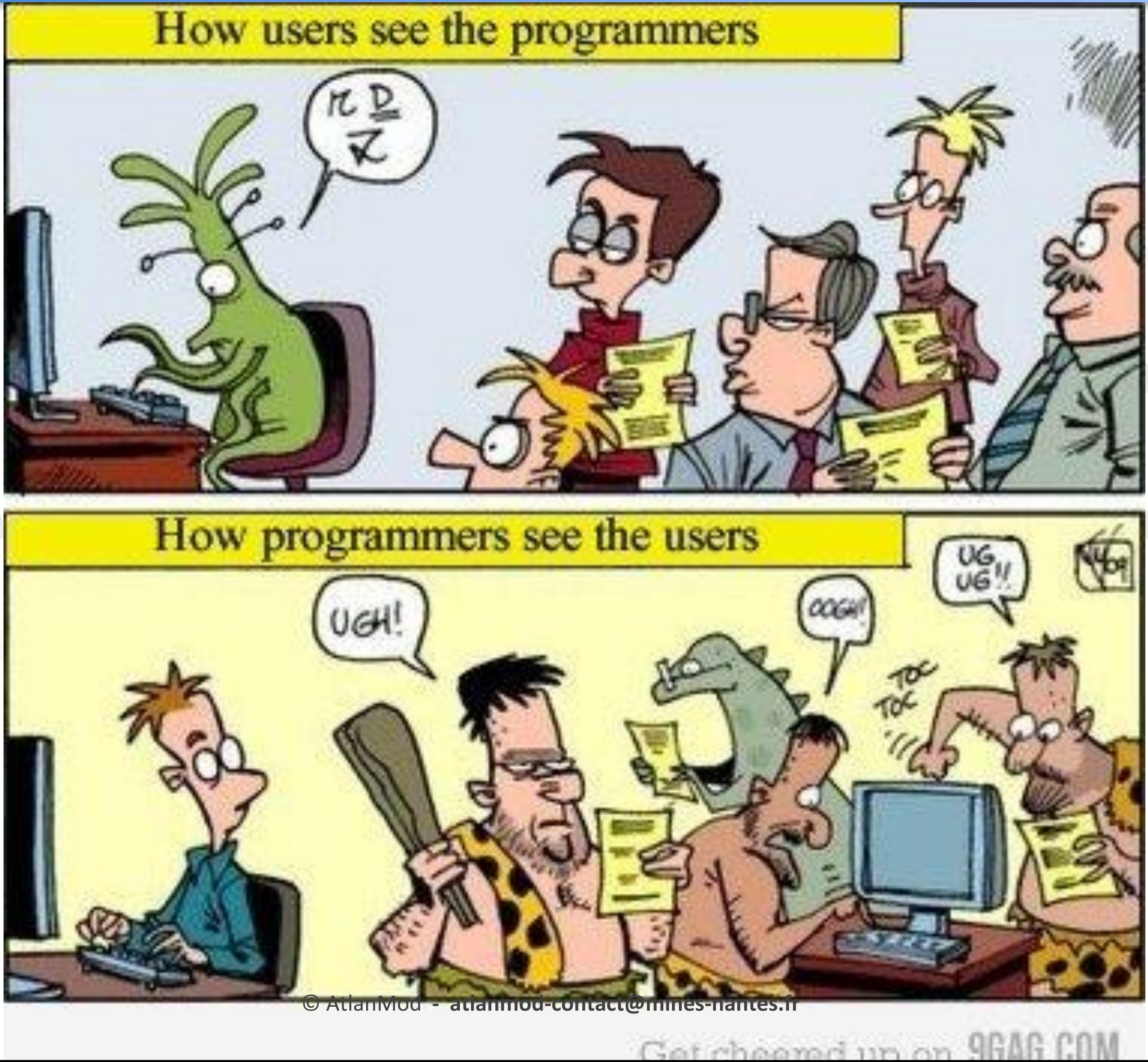
# Quality

- We know what quality properties make sense for models (e.g. satisfiability) but they do not translate well to DSLs.

- Quality of DSL = user experience

- Evaluating user experience is a challenging task
  - We cannot develop specific techniques for each different DSL
  - Need the participation of users

# Quality: Dealing with users is not easy


How users see the programmers
How programmers see the users
UGH!
OOGH!
UG UG!!
TOC TOC
Get cheered up on 9GAG.COM

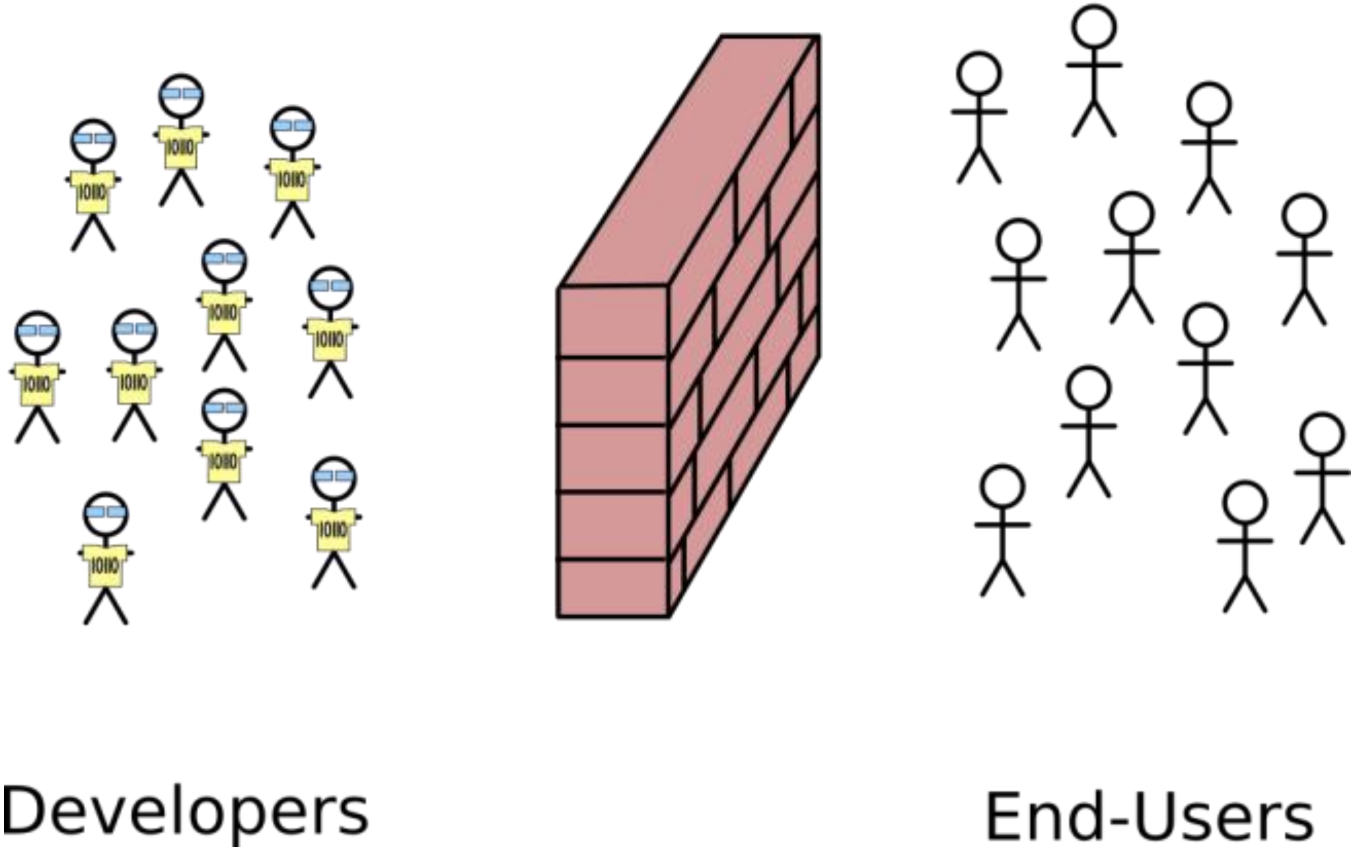
# The DSL case - Before

# Process

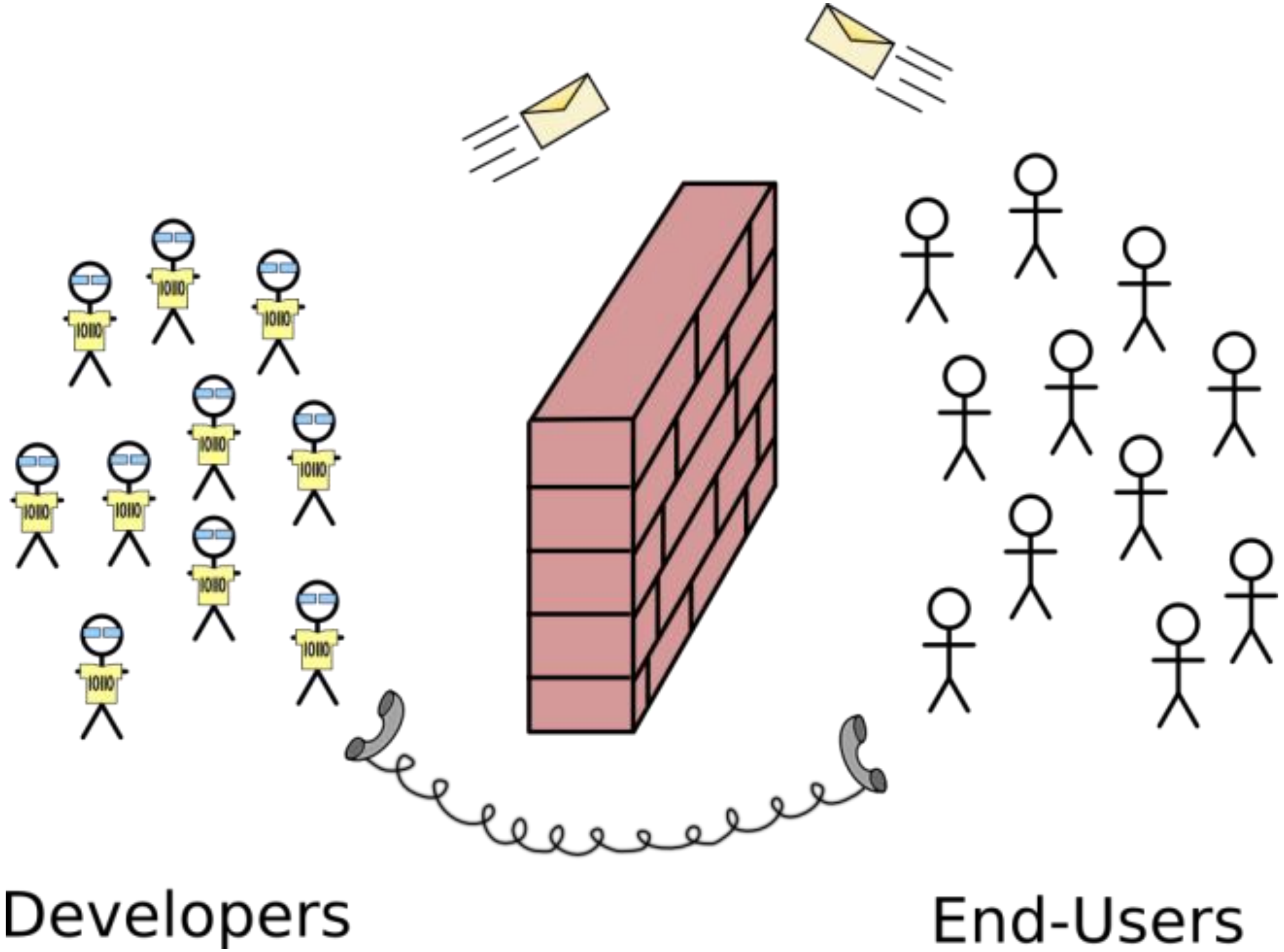
- DSLs are domain-specific but still it’s a non domain-expert who creates the DSL

- *Collaboro* aims to enable a more collaborative process

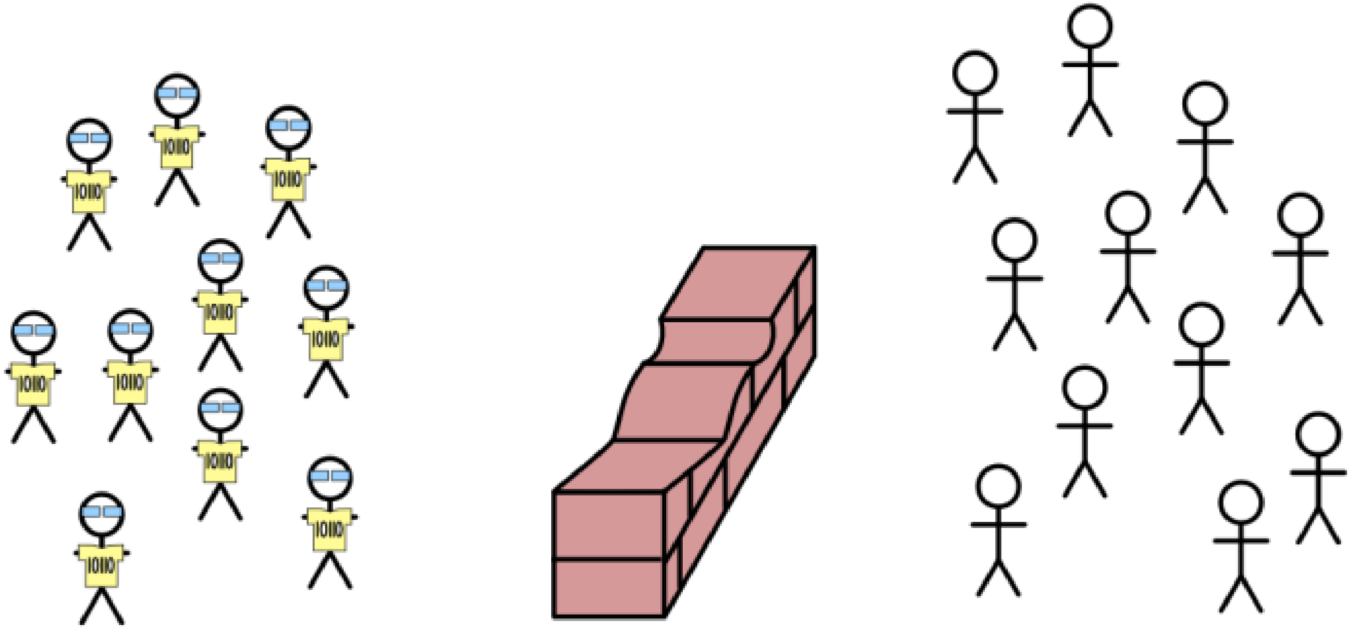
# Motivation



# Motivation



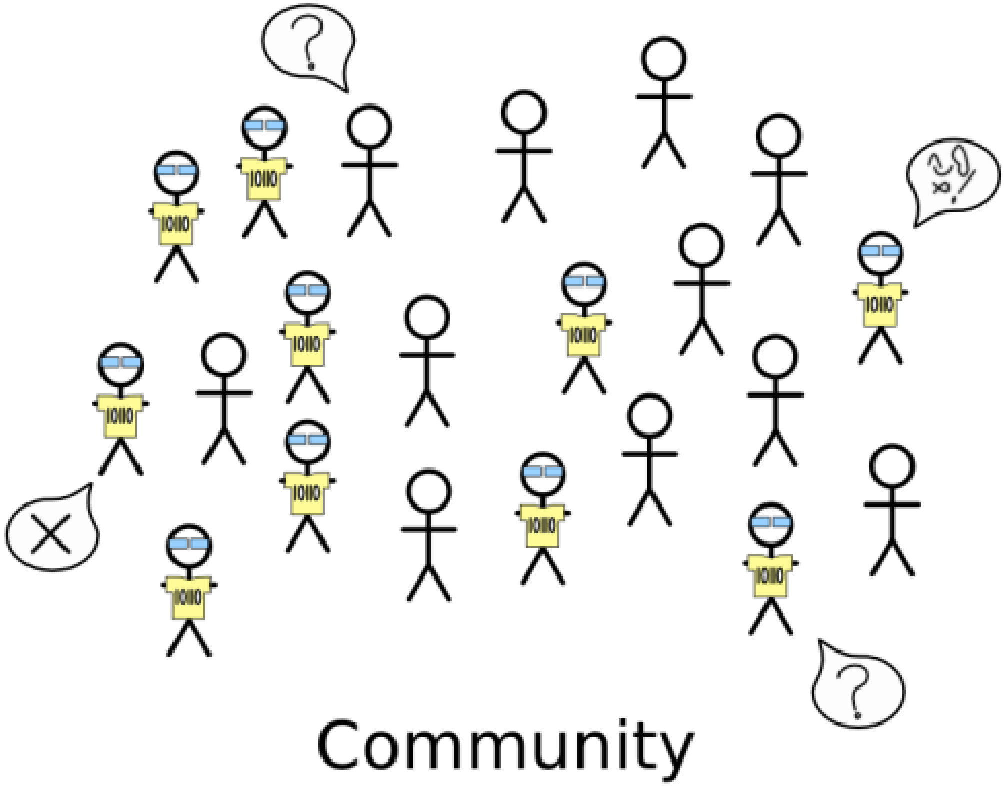
# Motivation



Developers

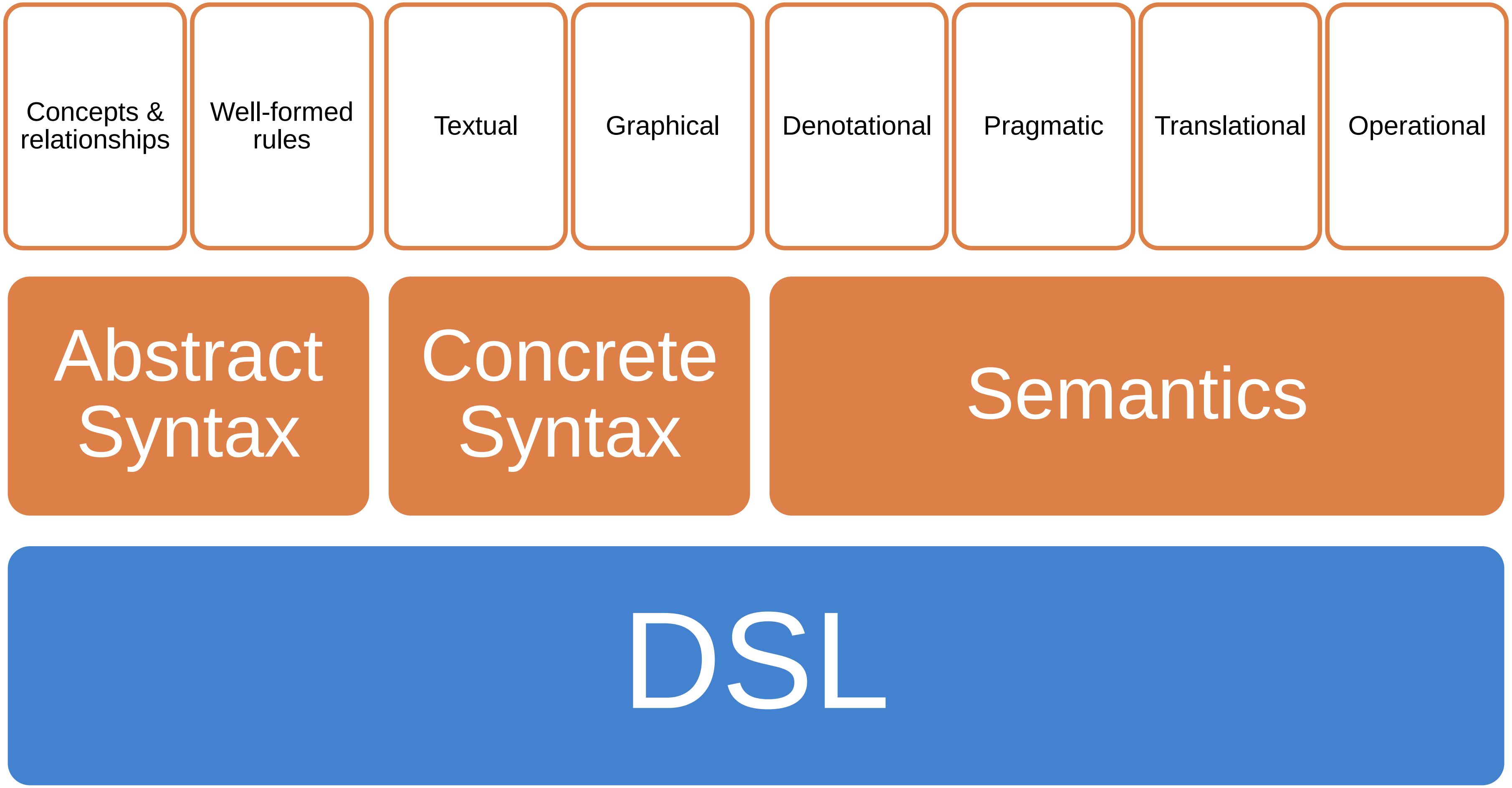
End-Users

# Motivation



Group of people involved with the DSML under development, which includes both technical level users and domain expert users

# Anatomy



Concepts & relationships

Well-formed rules

Textual

Graphical

Denotational

Pragmatic

Translational

Operational

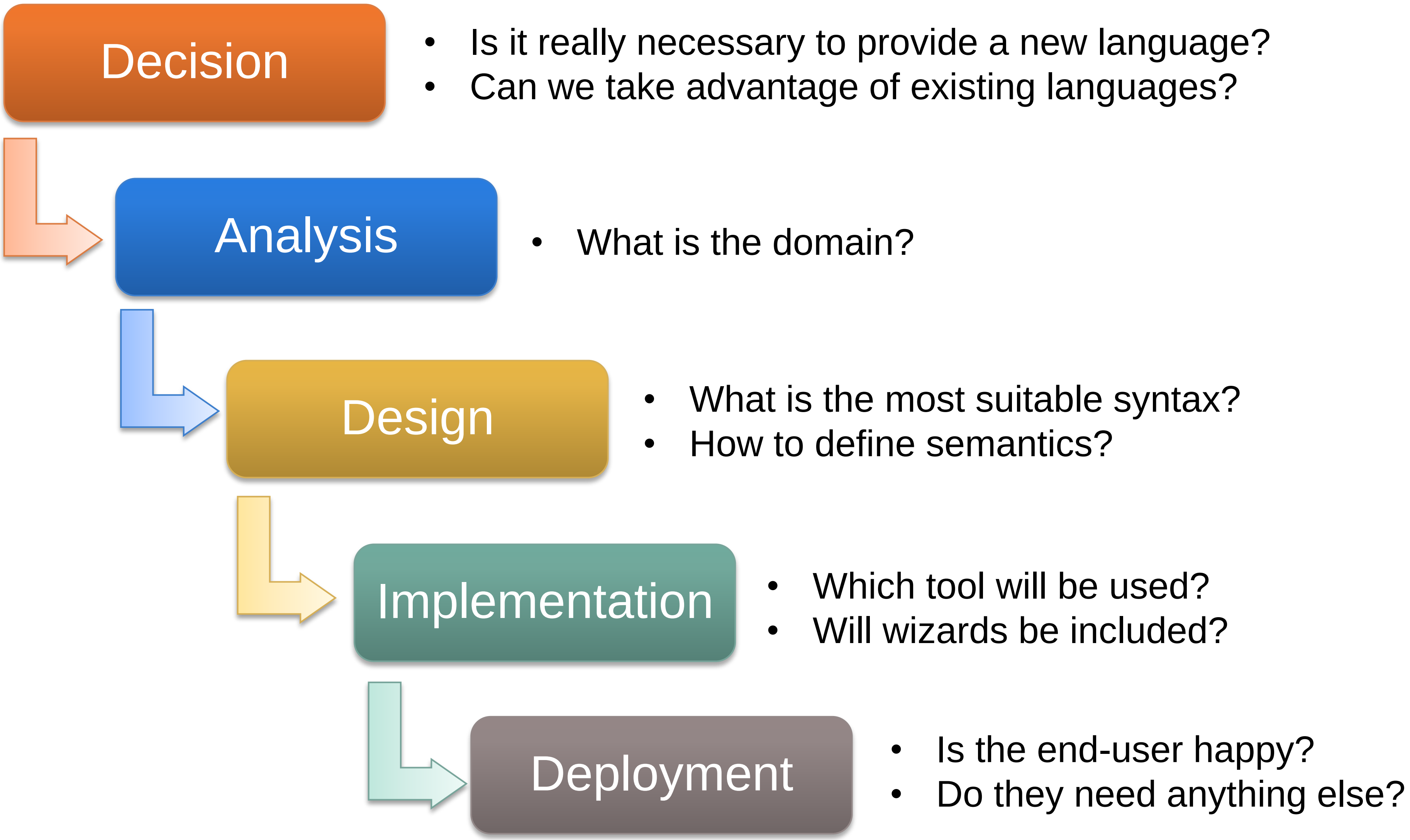
Abstract Syntax

Concrete Syntax

Semantics

DSL

# Current development process



Decision

- Is it really necessary to provide a new language?
- Can we take advantage of existing languages?

Analysis

- What is the domain?

Design

- What is the most suitable syntax?
- How to define semantics?

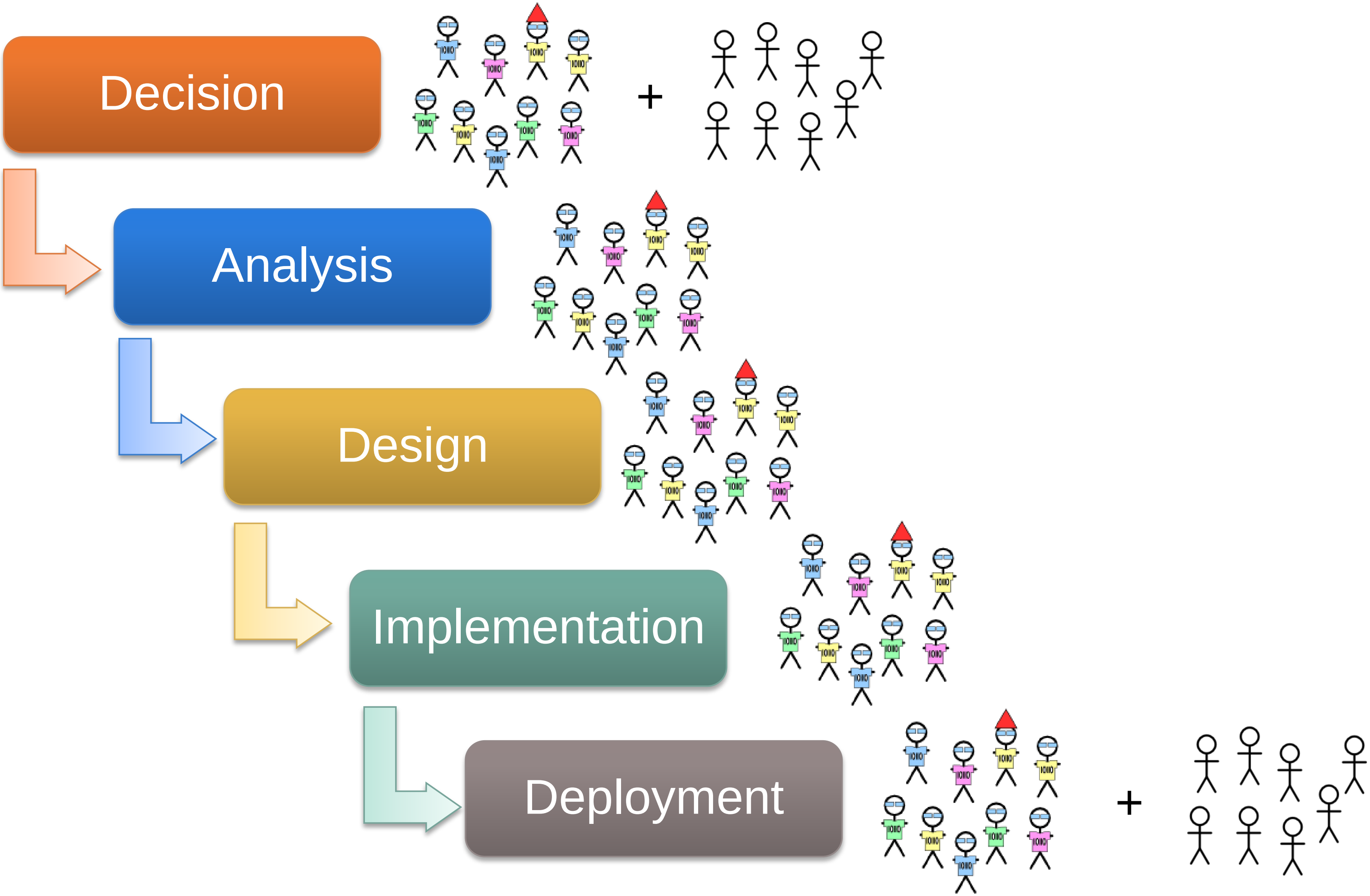
Implementation

- Which tool will be used?
- Will wizards be included?

Deployment

- Is the end-user happy?
- Do they need anything else?

# Current development process



Decision

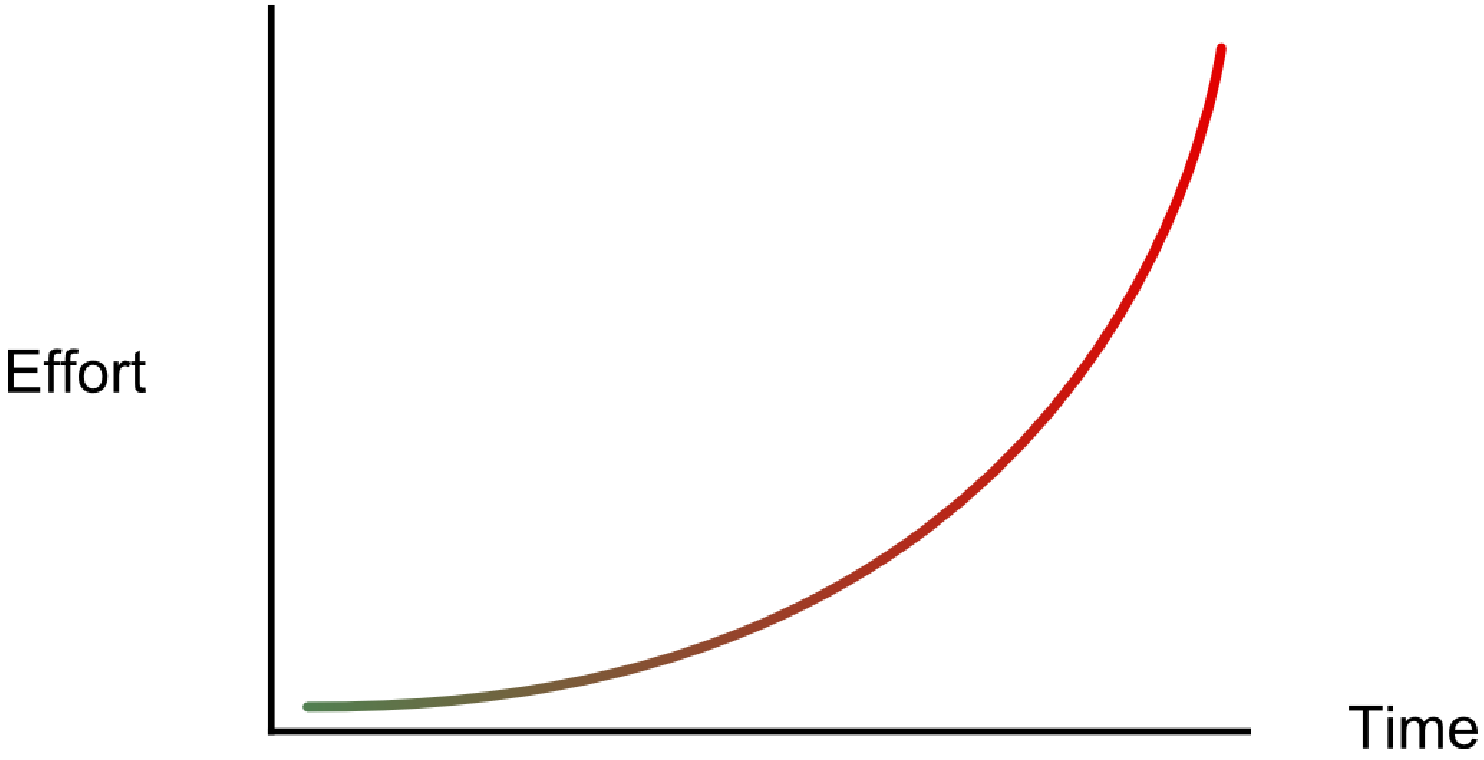
Analysis

Design

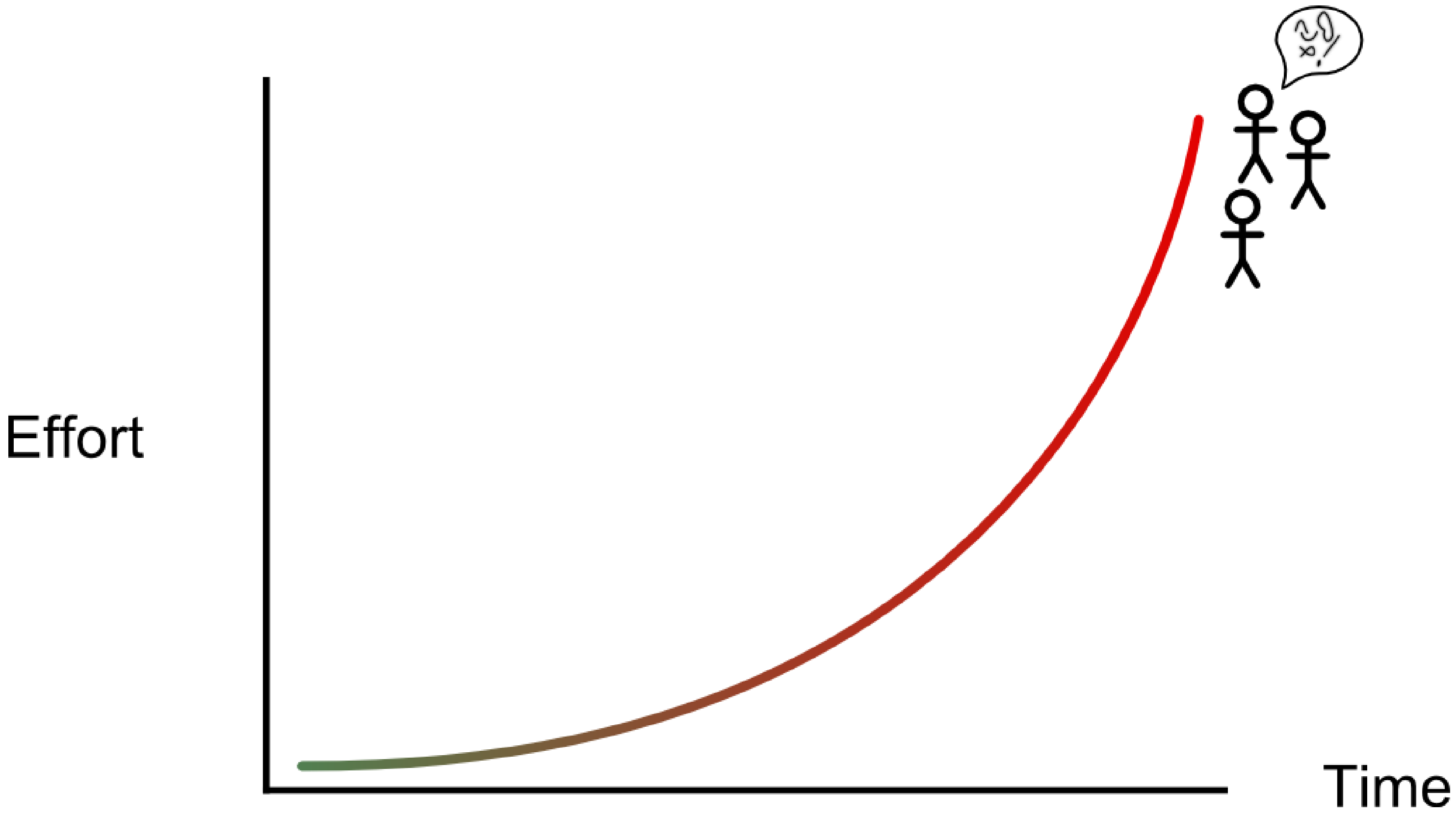
Implementation

Deployment

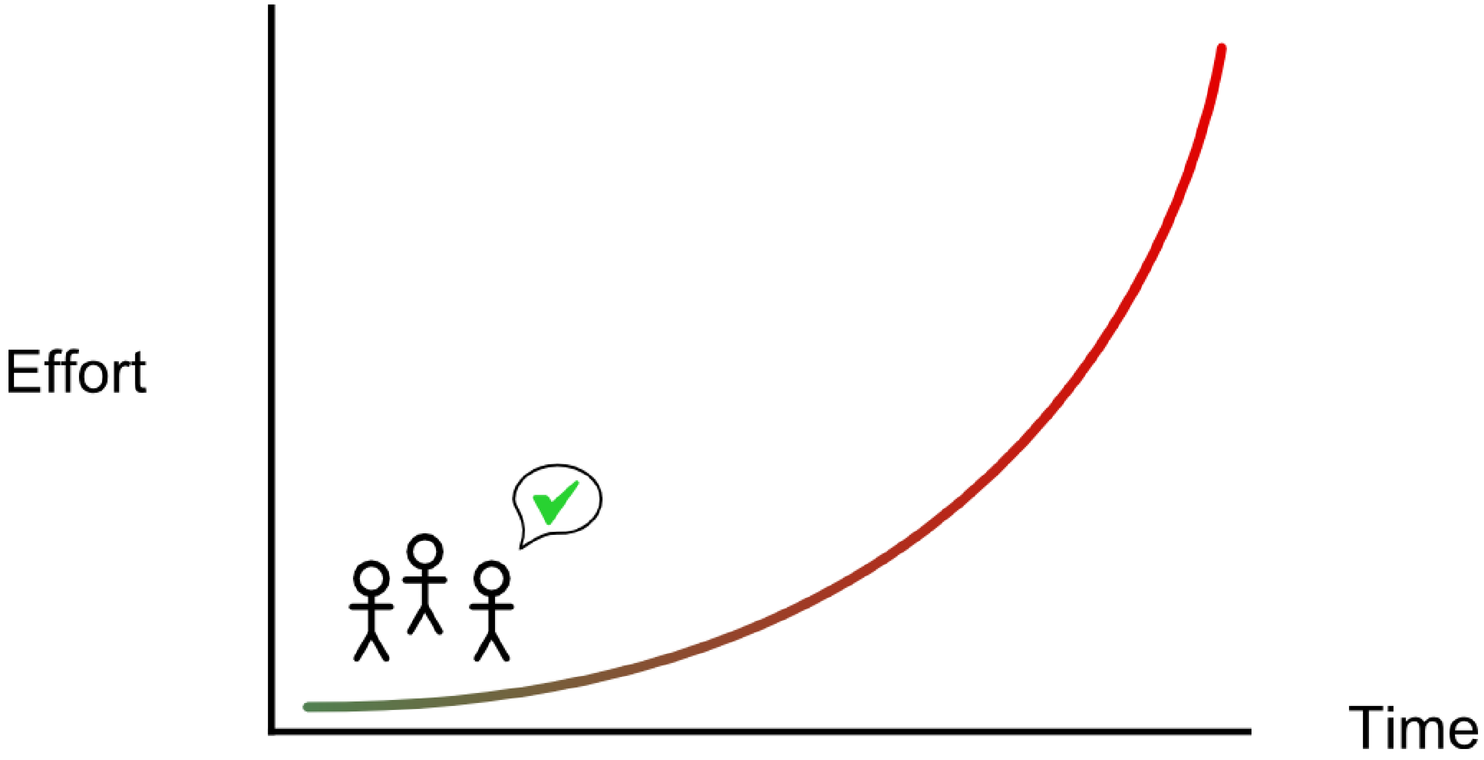
# Boehm's graph



# Current end-user participation



# The sooner the better

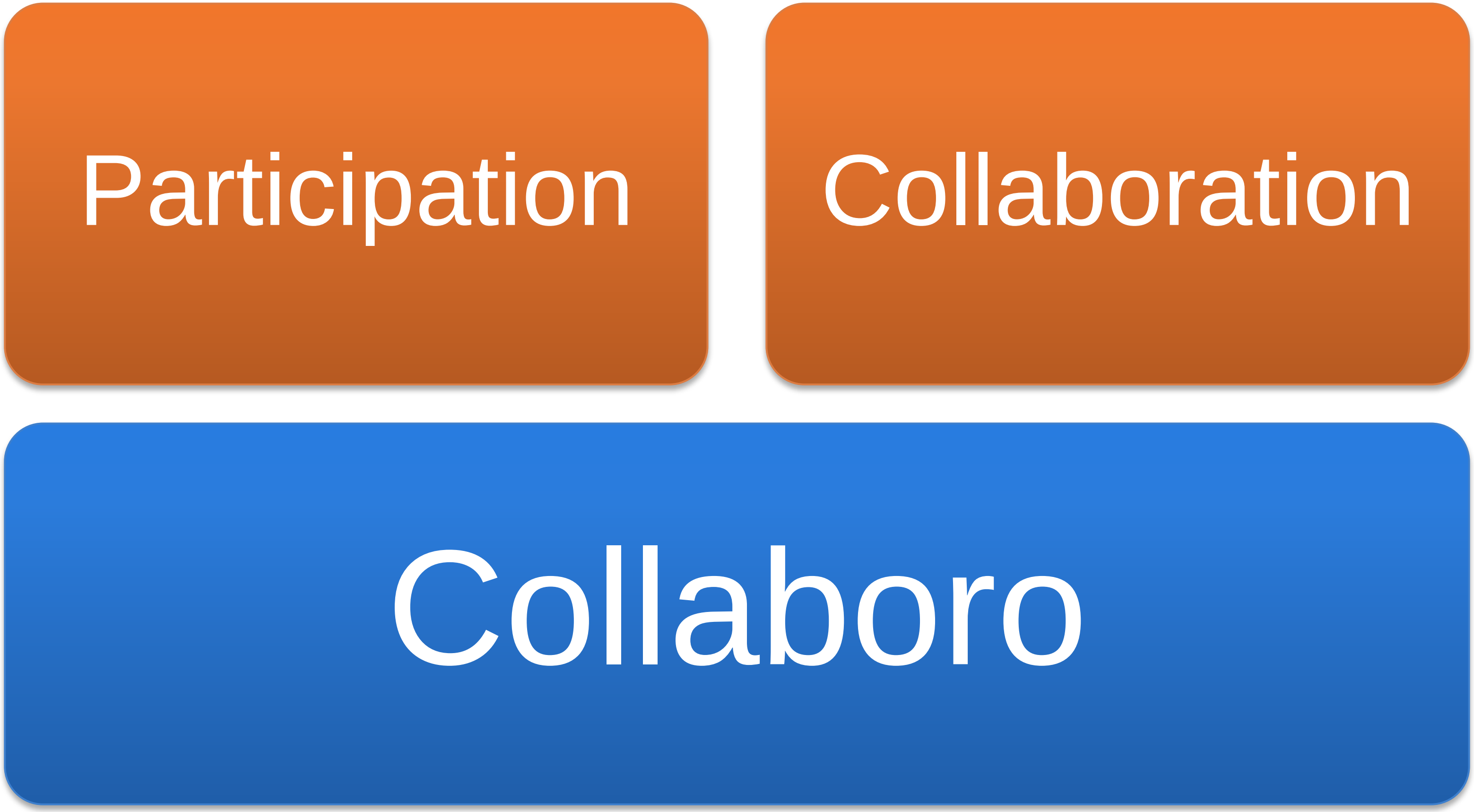


# How? Existing tools

# Existing tools

No support for DSLs

# Our proposal



Participation

Collaboration

Collaboro

# But… technically?

## Participation

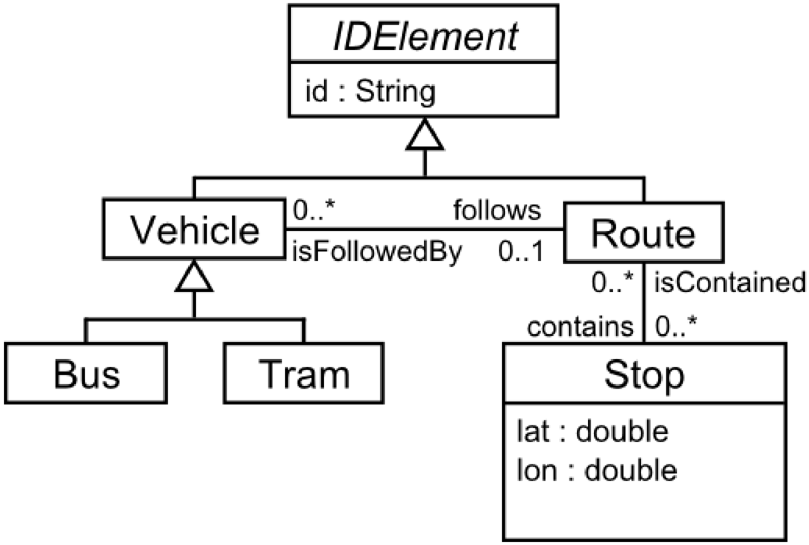
- Providing means to discuss about language elements
- Overcoming technical barriers

## Collaboration

- Suitable environment
- Fostering end-user discussion
- Facilitating voting processes

# Example: before

Abstract Syntax



Concrete Syntax Example

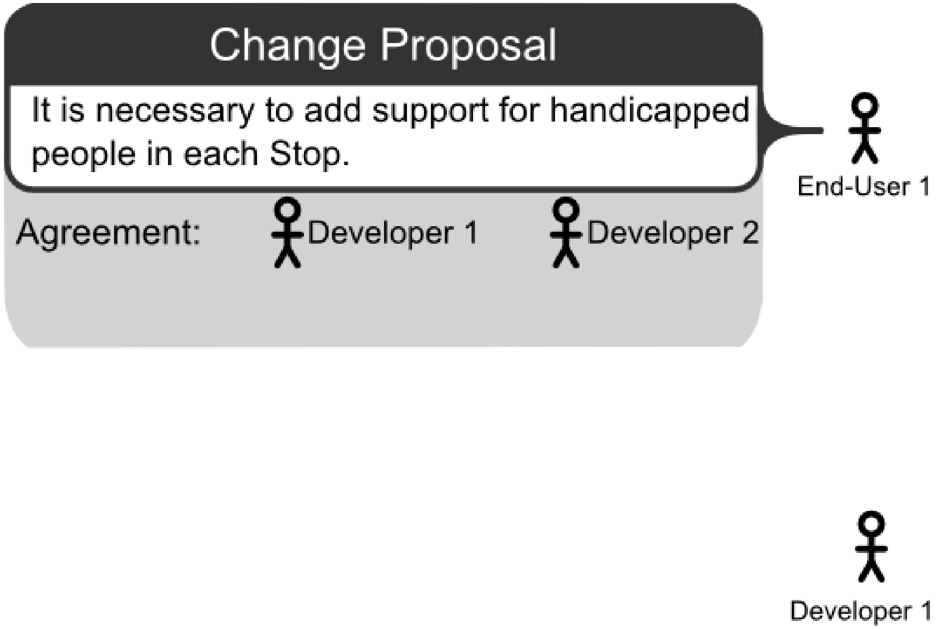
```
tram 1:     route A:                 stop 001:
  route A;    stops : 001, 002;        lat: 23.1082
...         ...                        lon: 12.9883

                                     ...
```

# Example

End-User 1

Developer 1

# Example



Change Proposal

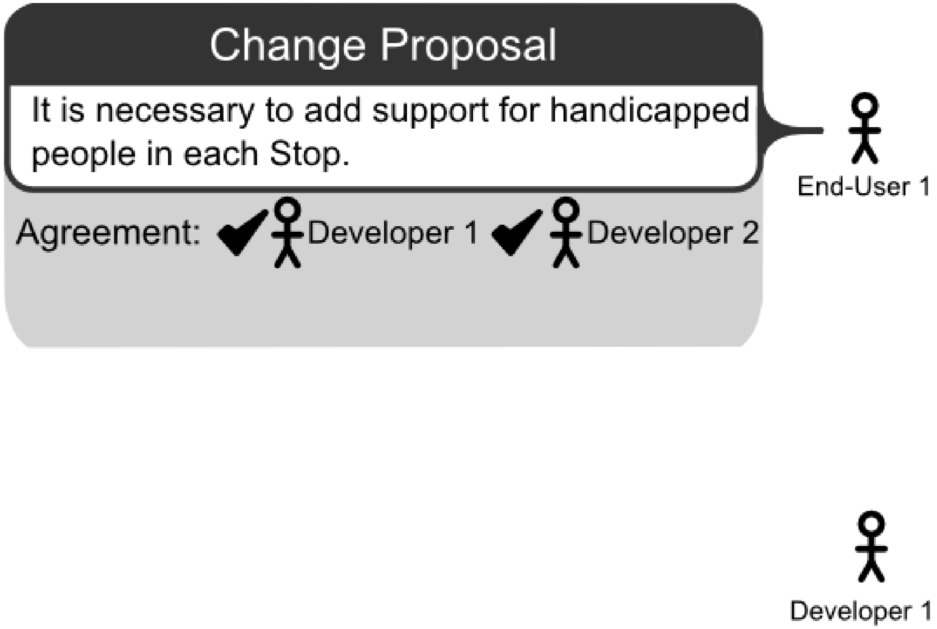
It is necessary to add support for handicapped people in each Stop.

Agreement: Developer 1 Developer 2

End-User 1

Developer 1

# Example



Change Proposal

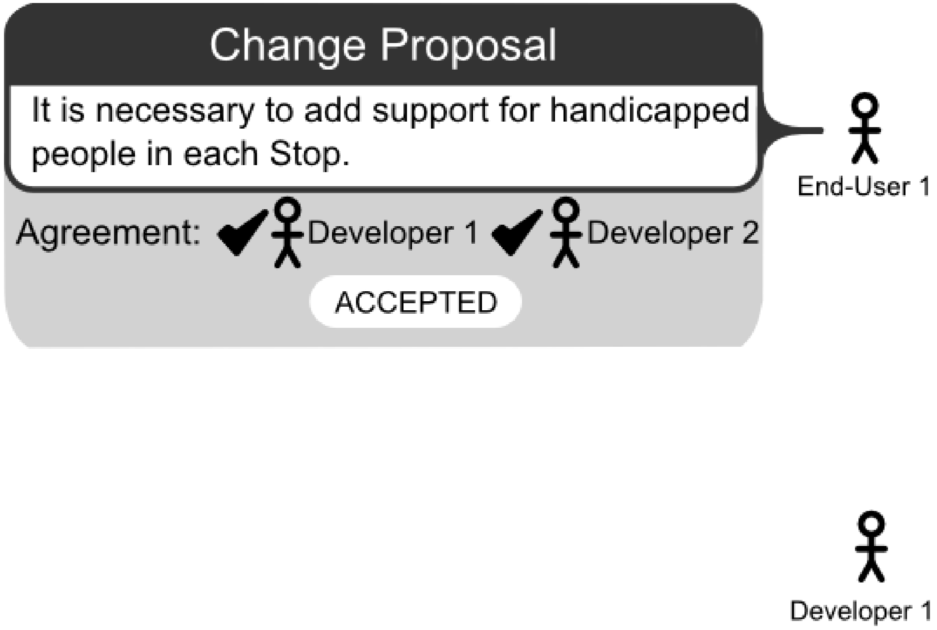
It is necessary to add support for handicapped people in each Stop.

Agreement: ✓ Developer 1 ✓ Developer 2

End-User 1

Developer 1

# Example



Change Proposal

It is necessary to add support for handicapped people in each Stop.

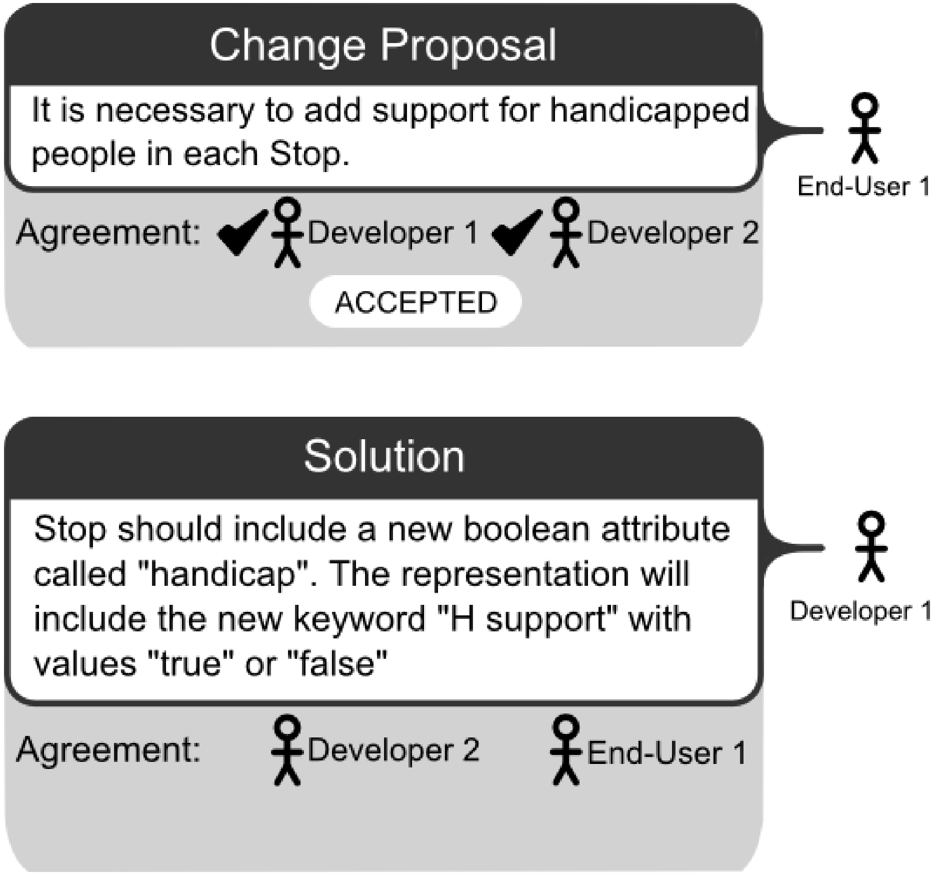
Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

End-User 1

Developer 1

# Example



Change Proposal

It is necessary to add support for handicapped people in each Stop.

End-User 1

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

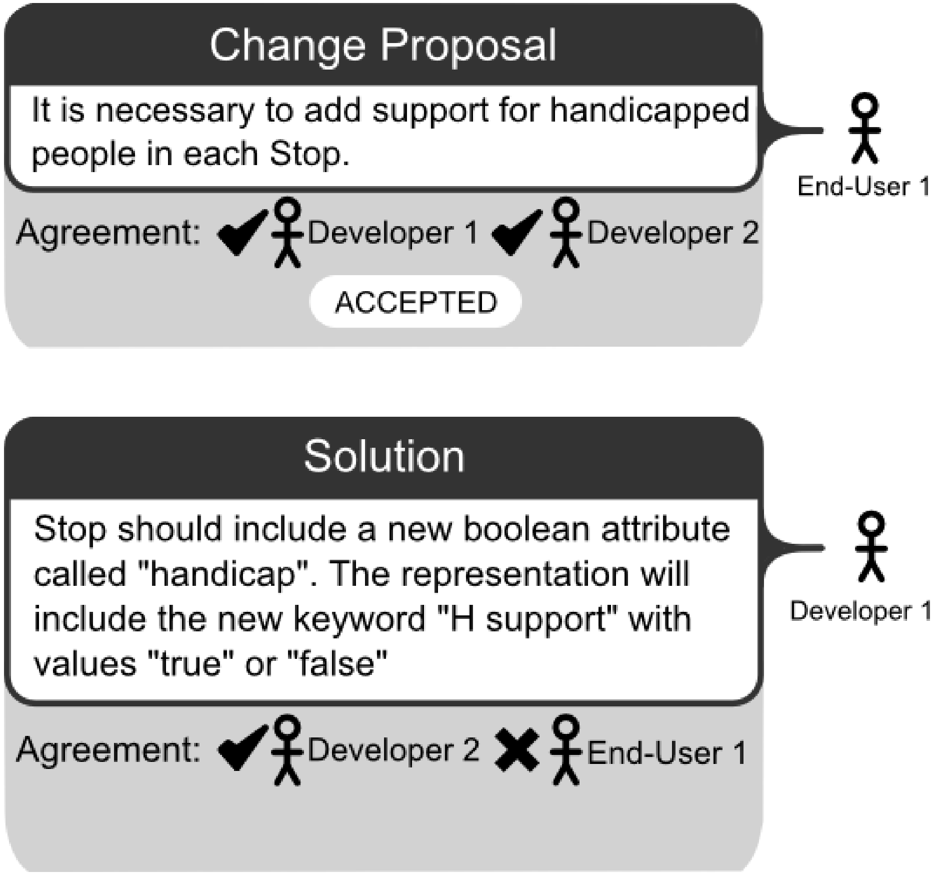
Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Developer 1

Agreement: Developer 2 End-User 1

# Example



Change Proposal

It is necessary to add support for handicapped people in each Stop.

End-User 1

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

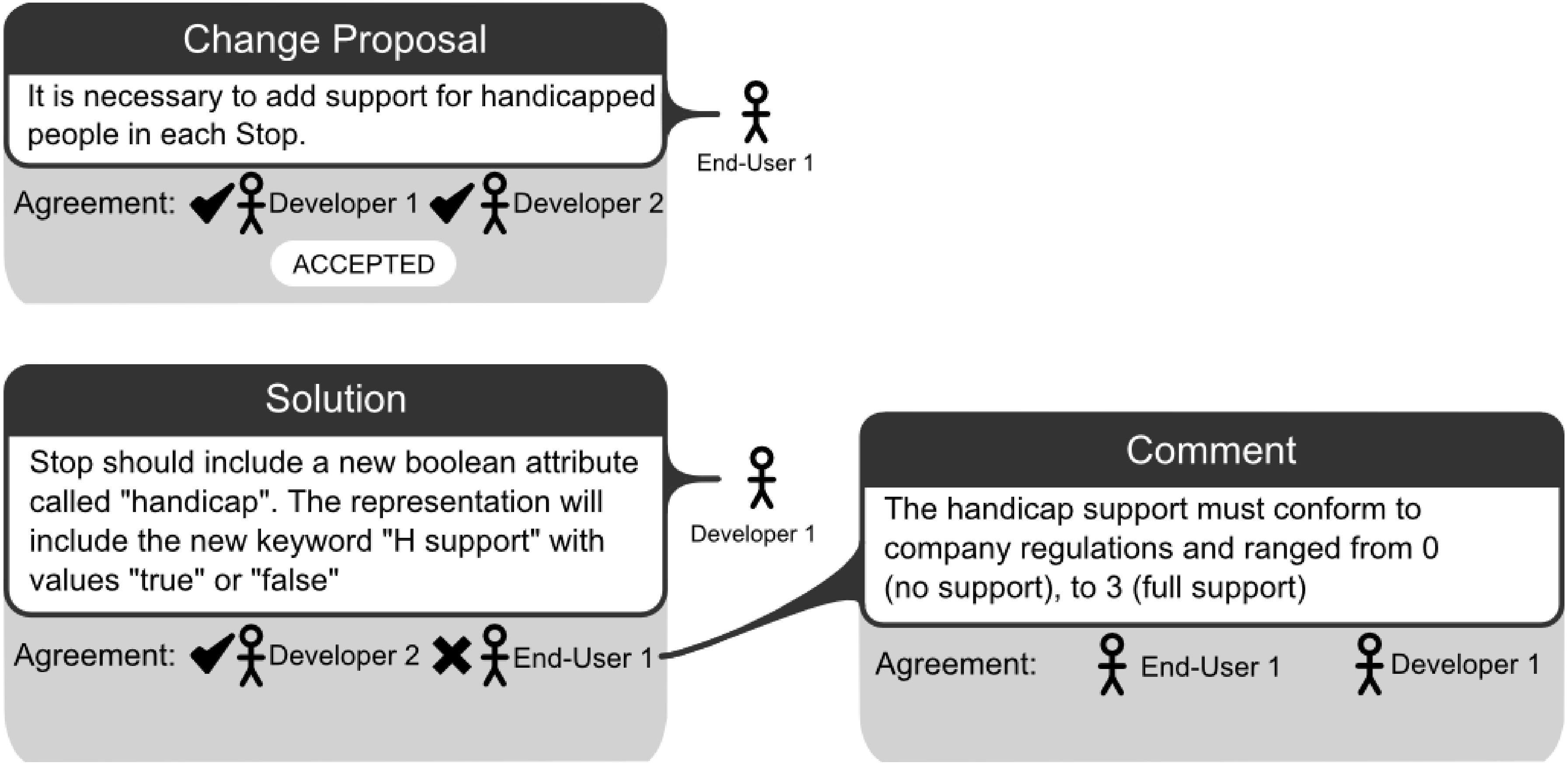
Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Developer 1

Agreement: ✓ Developer 2 ✗ End-User 1

# Example



Change Proposal

It is necessary to add support for handicapped people in each Stop.

End-User 1

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Developer 1

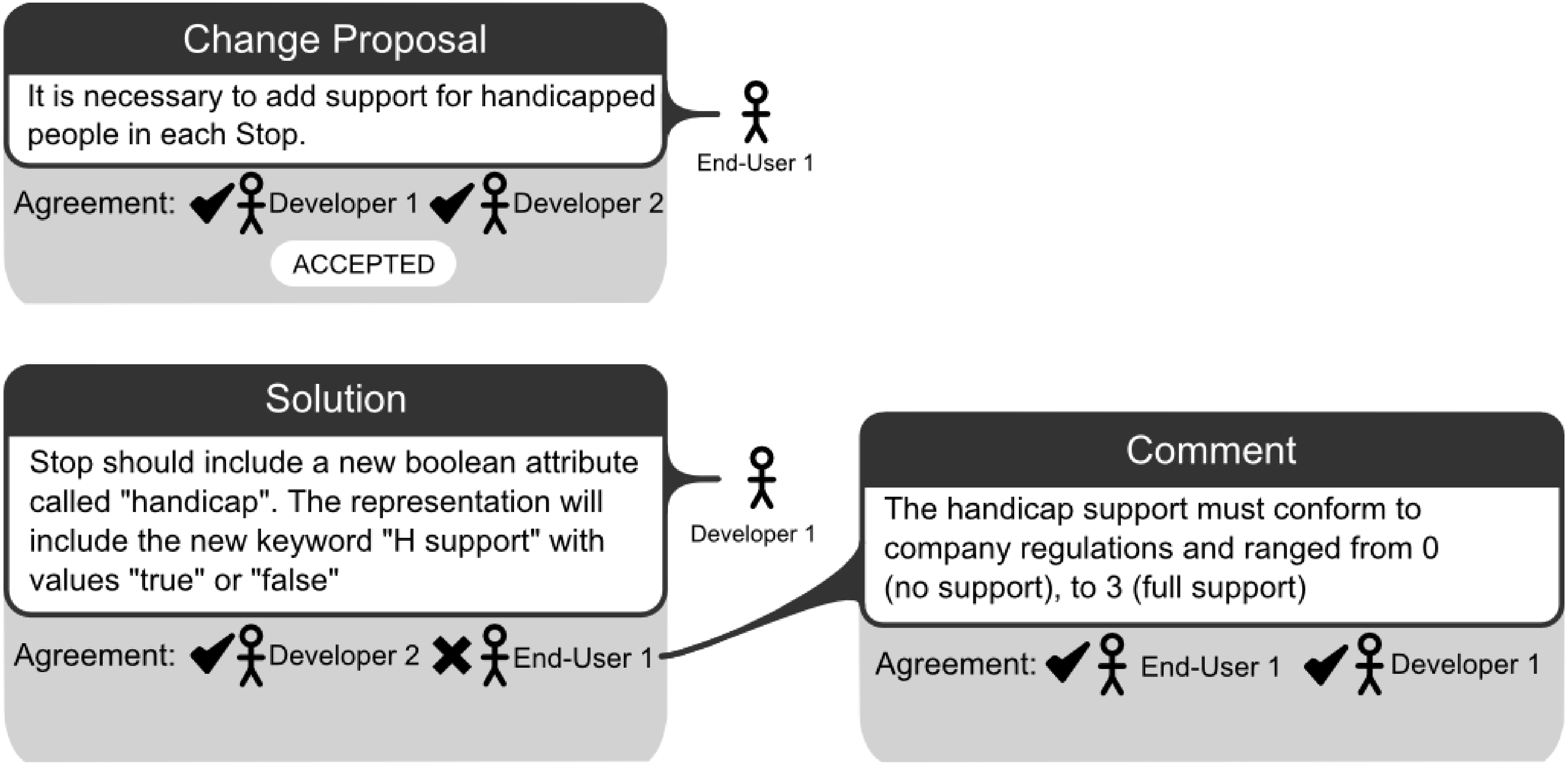
Agreement: ✓ Developer 2 ✗ End-User 1

Comment

The handicap support must conform to company regulations and ranged from 0 (no support), to 3 (full support)

Agreement: End-User 1 Developer 1

# Example



Change Proposal

It is necessary to add support for handicapped people in each Stop.

End-User 1

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Developer 1

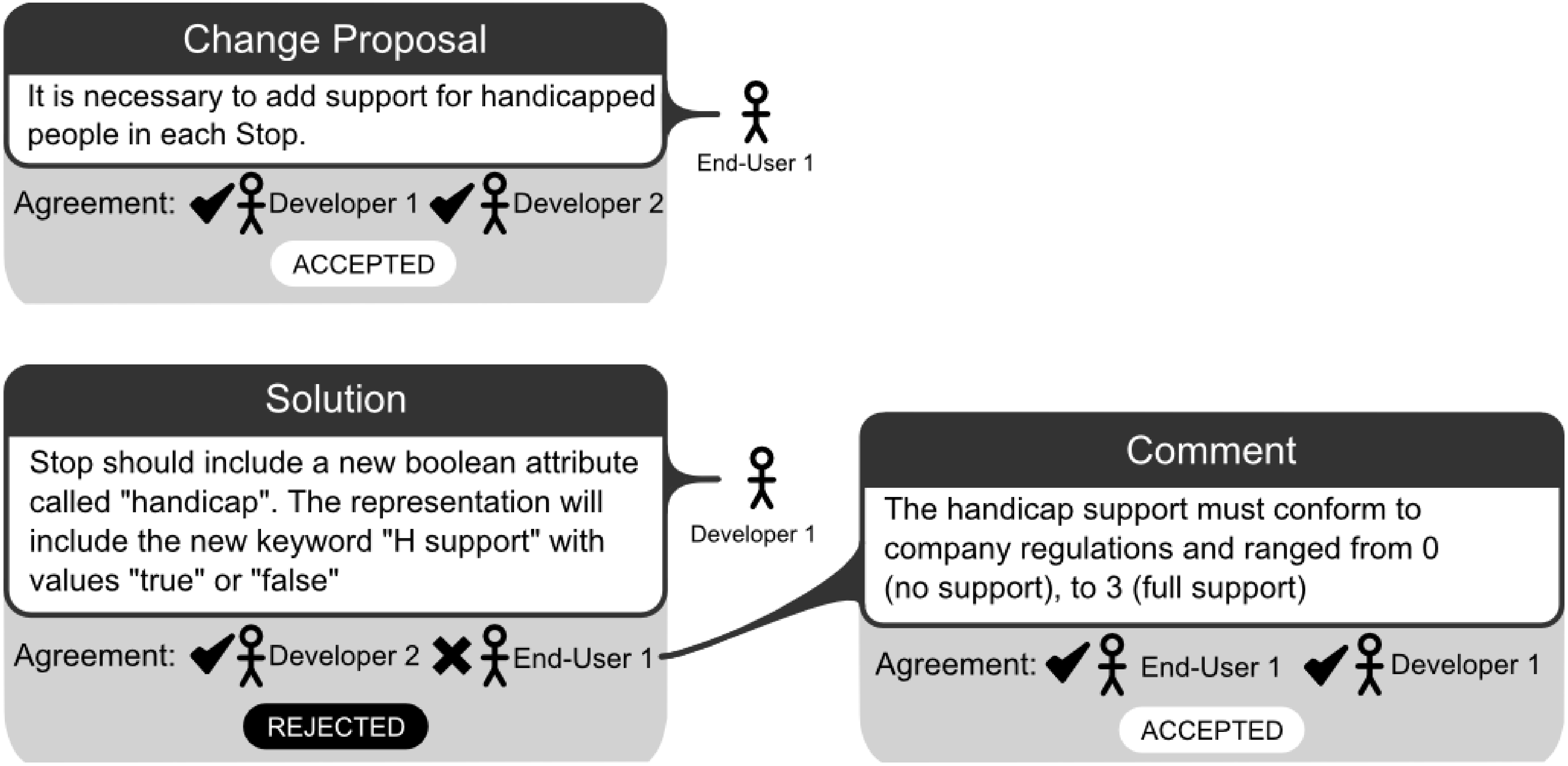
Agreement: ✓ Developer 2 ✗ End-User 1

Comment

The handicap support must conform to company regulations and ranged from 0 (no support), to 3 (full support)

Agreement: ✓ End-User 1 ✓ Developer 1

# Example



Change Proposal

It is necessary to add support for handicapped people in each Stop.

End-User 1

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Developer 1

Agreement: ✓ Developer 2 ✗ End-User 1

REJECTED

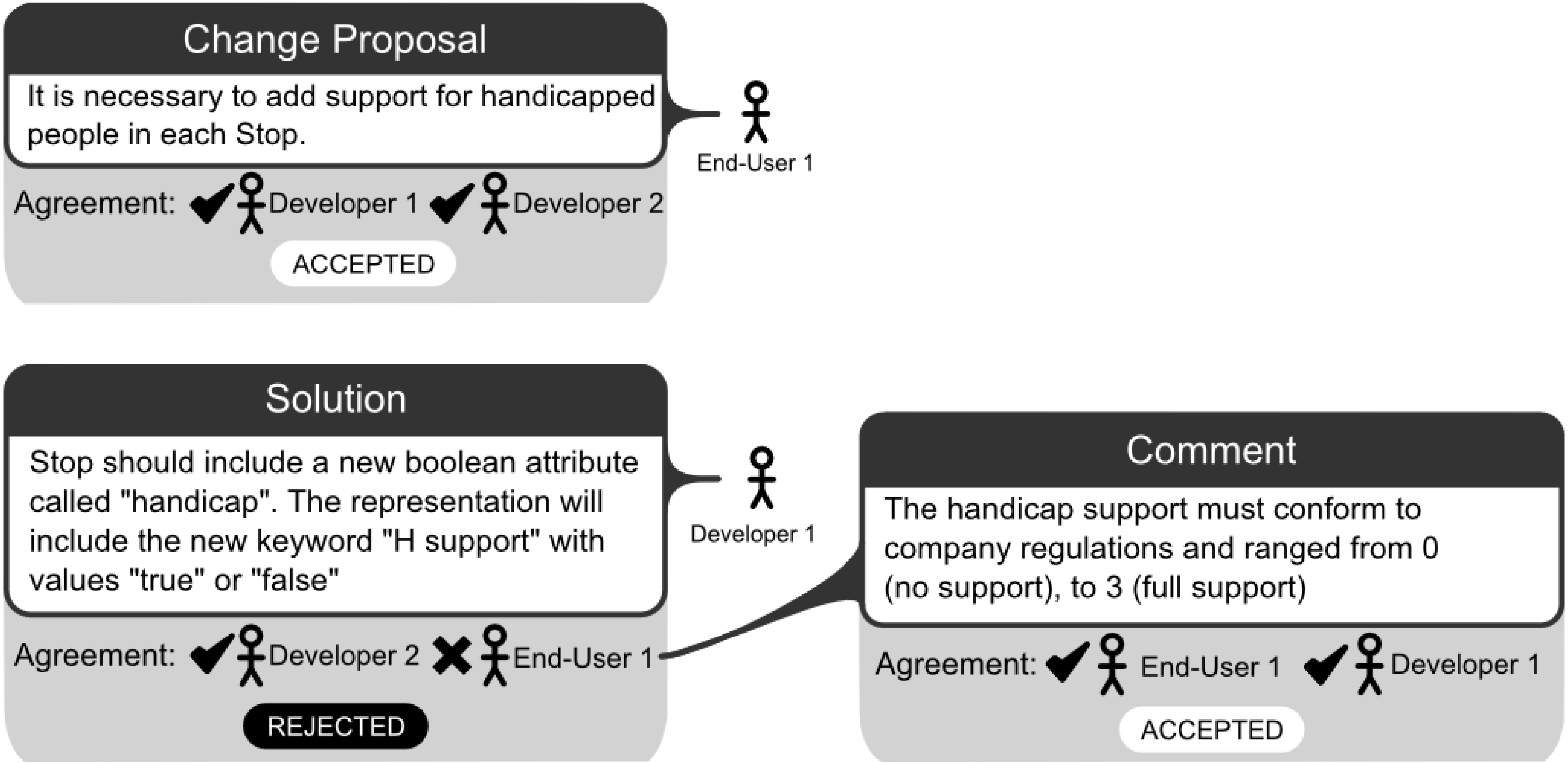
Comment

The handicap support must conform to company regulations and ranged from 0 (no support), to 3 (full support)

Agreement: ✓ End-User 1 ✓ Developer 1

ACCEPTED

# Example



Change Proposal

It is necessary to add support for handicapped people in each Stop.

End-User 1

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Developer 1

Agreement: ✓ Developer 2 ✗ End-User 1

REJECTED

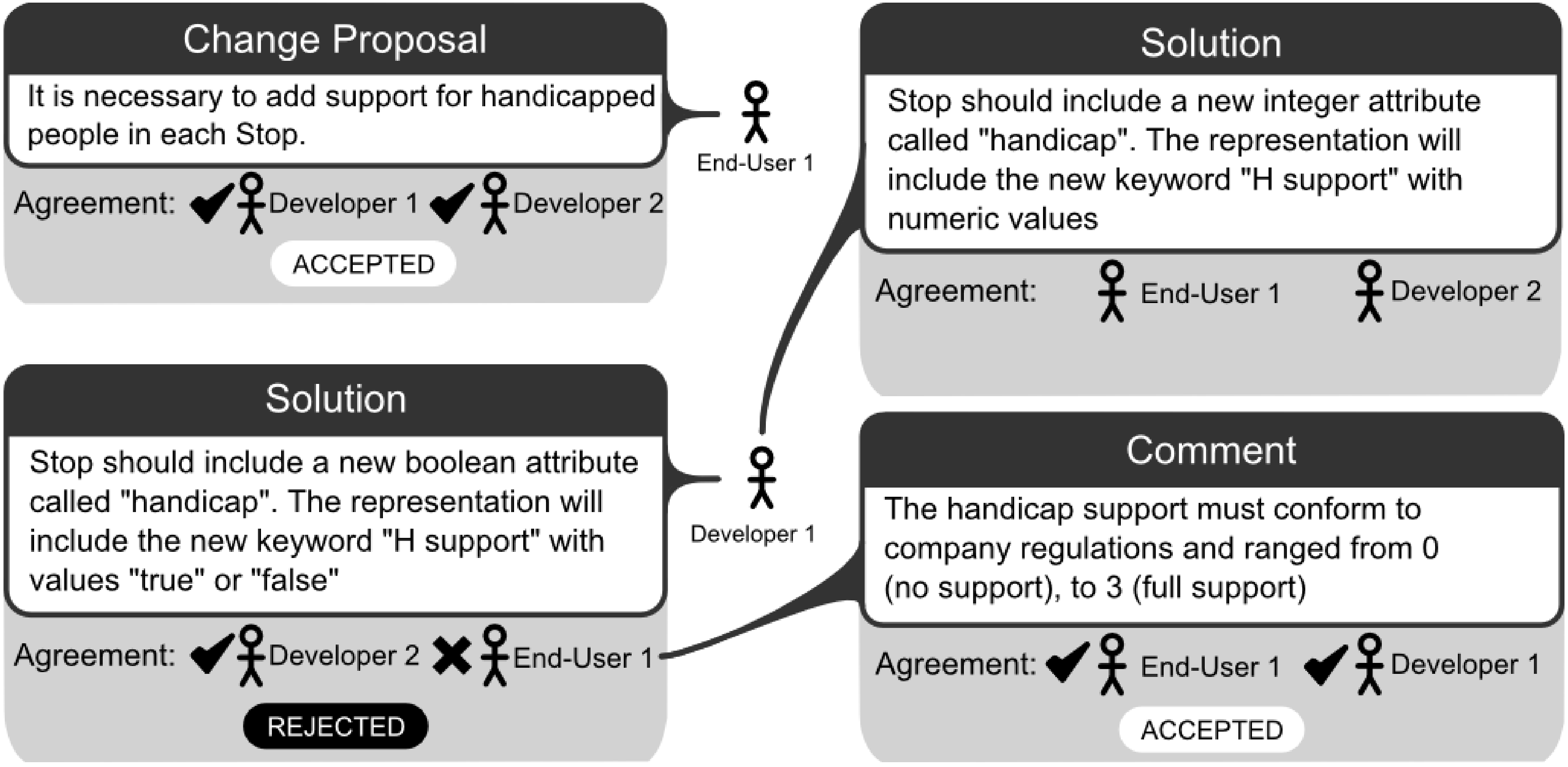
Comment

The handicap support must conform to company regulations and ranged from 0 (no support), to 3 (full support)

Agreement: ✓ End-User 1 ✓ Developer 1

ACCEPTED

# Example



Change Proposal

It is necessary to add support for handicapped people in each Stop.

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

Solution

Stop should include a new integer attribute called "handicap". The representation will include the new keyword "H support" with numeric values

Agreement: End-User 1 Developer 2

Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Agreement: ✓ Developer 2 ✗ End-User 1

REJECTED

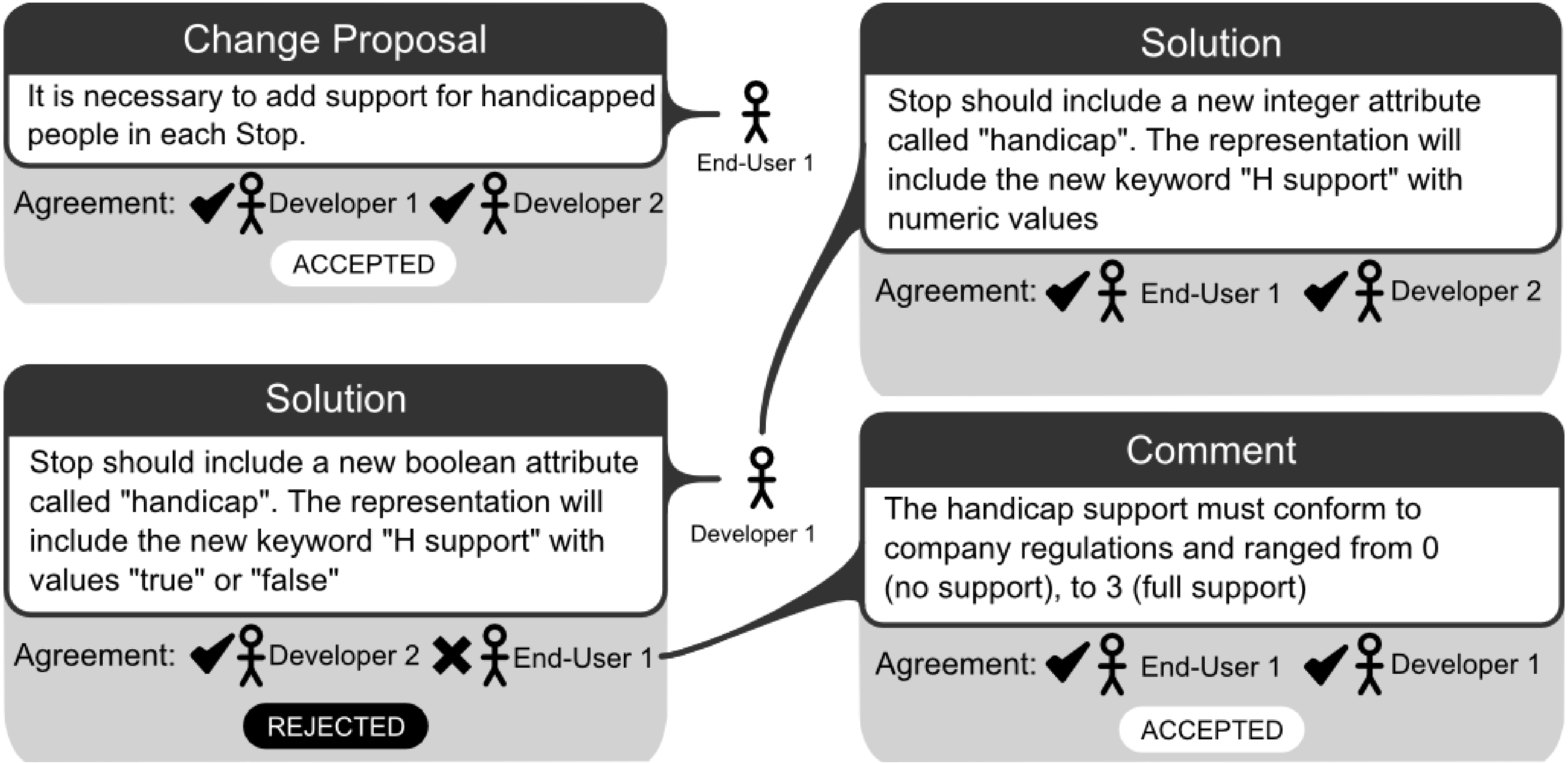
Comment

The handicap support must conform to company regulations and ranged from 0 (no support), to 3 (full support)

Agreement: ✓ End-User 1 ✓ Developer 1

ACCEPTED

# Example



## Change Proposal

It is necessary to add support for handicapped people in each Stop.

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

## Solution

Stop should include a new integer attribute called "handicap". The representation will include the new keyword "H support" with numeric values

Agreement: ✓ End-User 1 ✓ Developer 2

## Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Agreement: ✓ Developer 2 ✗ End-User 1

REJECTED

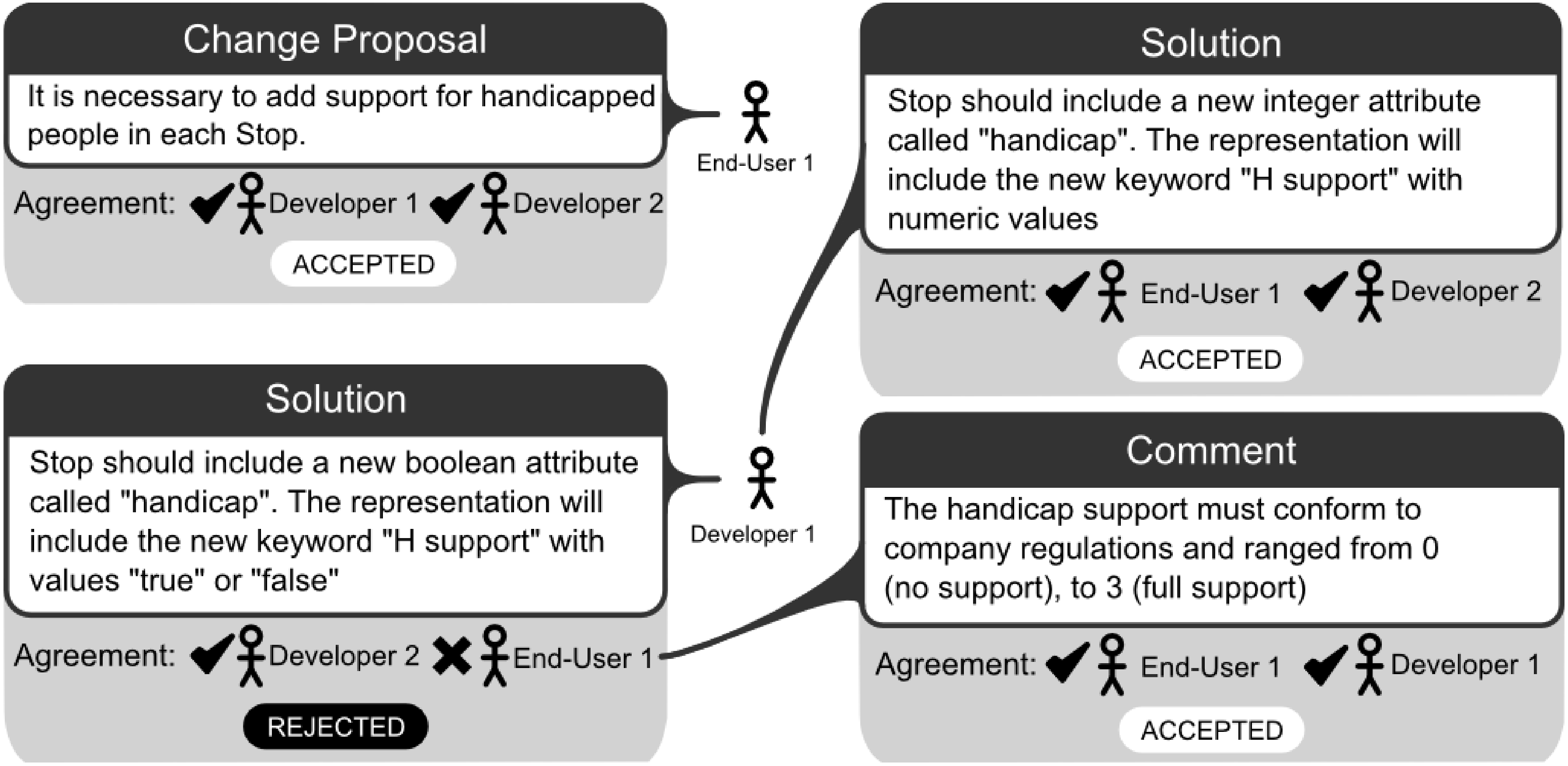
## Comment

The handicap support must conform to company regulations and ranged from 0 (no support), to 3 (full support)

Agreement: ✓ End-User 1 ✓ Developer 1

ACCEPTED

# Example



## Change Proposal

It is necessary to add support for handicapped people in each Stop.

Agreement: ✓ Developer 1 ✓ Developer 2

ACCEPTED

End-User 1

## Solution

Stop should include a new integer attribute called "handicap". The representation will include the new keyword "H support" with numeric values

Agreement: ✓ End-User 1 ✓ Developer 2

ACCEPTED

Developer 1

## Solution

Stop should include a new boolean attribute called "handicap". The representation will include the new keyword "H support" with values "true" or "false"

Agreement: ✓ Developer 2 ✗ End-User 1

REJECTED

## Comment

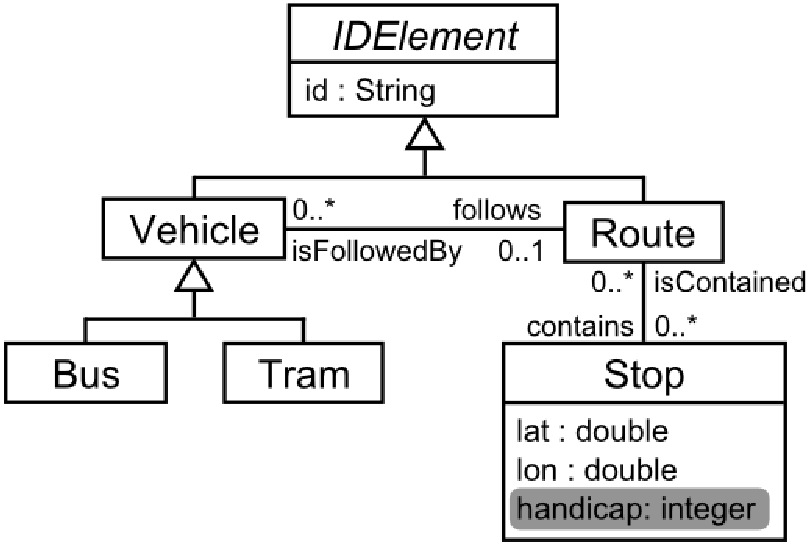
The handicap support must conform to company regulations and ranged from 0 (no support), to 3 (full support)

Agreement: ✓ End-User 1 ✓ Developer 1

ACCEPTED

# Example: after

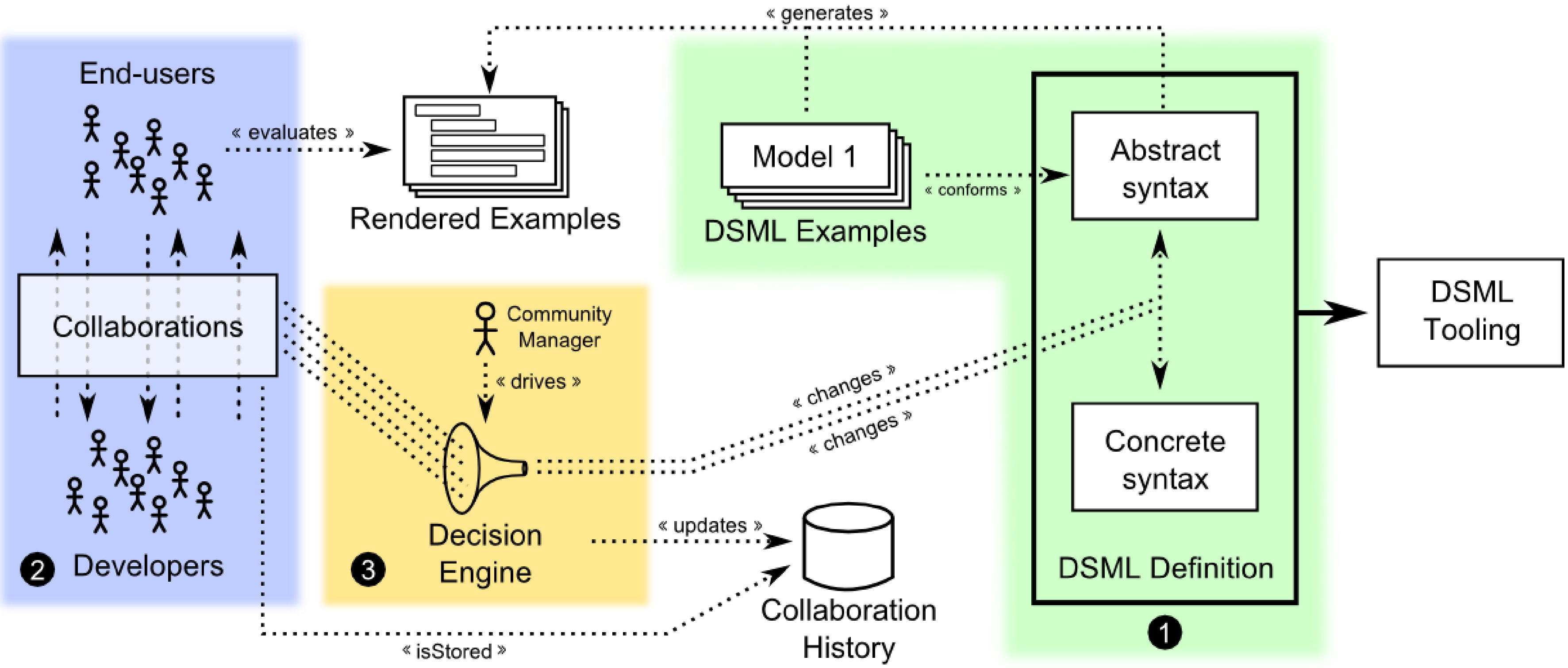
Abstract Syntax



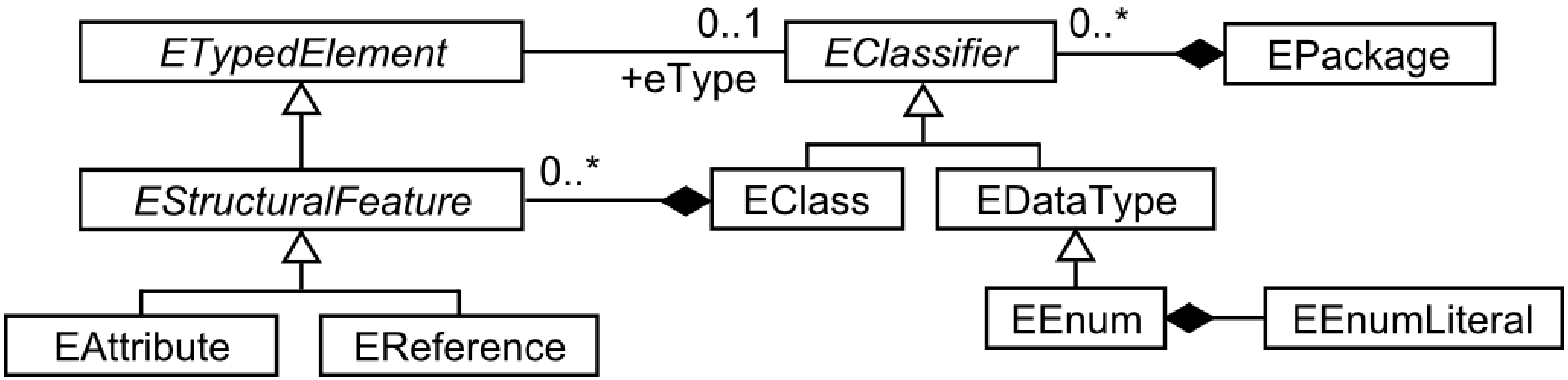
Concrete Syntax Example

```
tram 1:      route A:              stop 001:
  route A;     stops : 001, 002;     lat: 23.1082
...          ...                     lon: 12.9883
                                     H support: 3
                                   ...
```

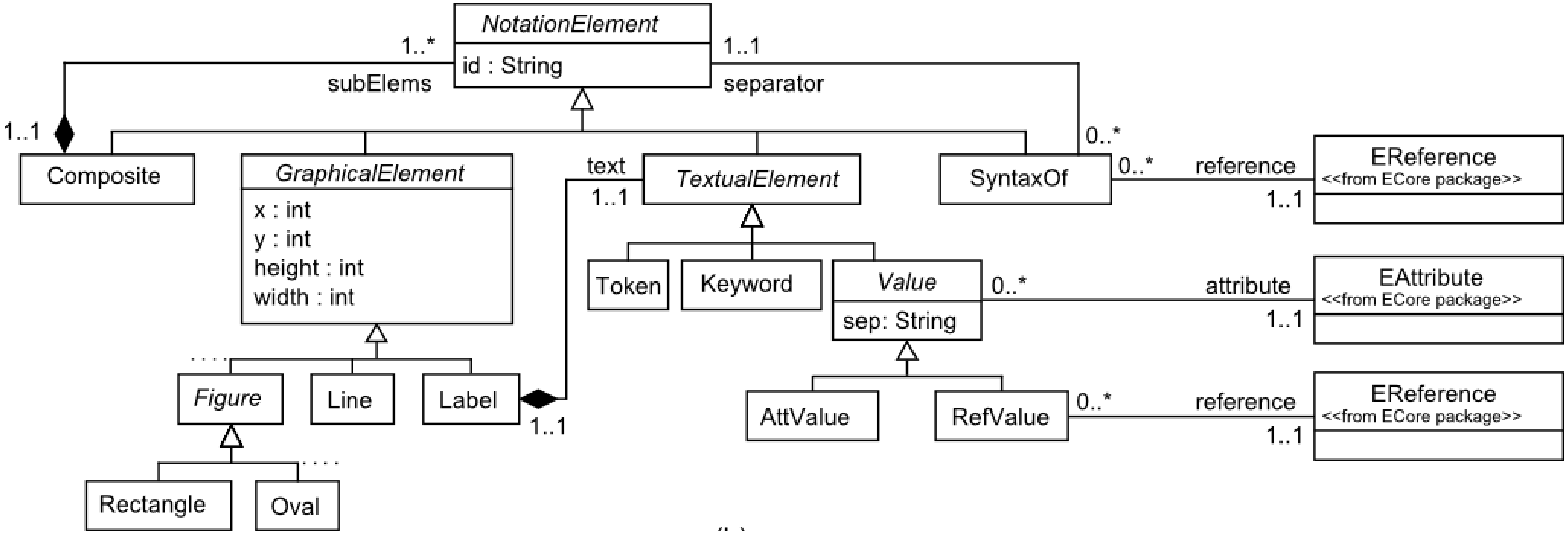
# Collaboro process



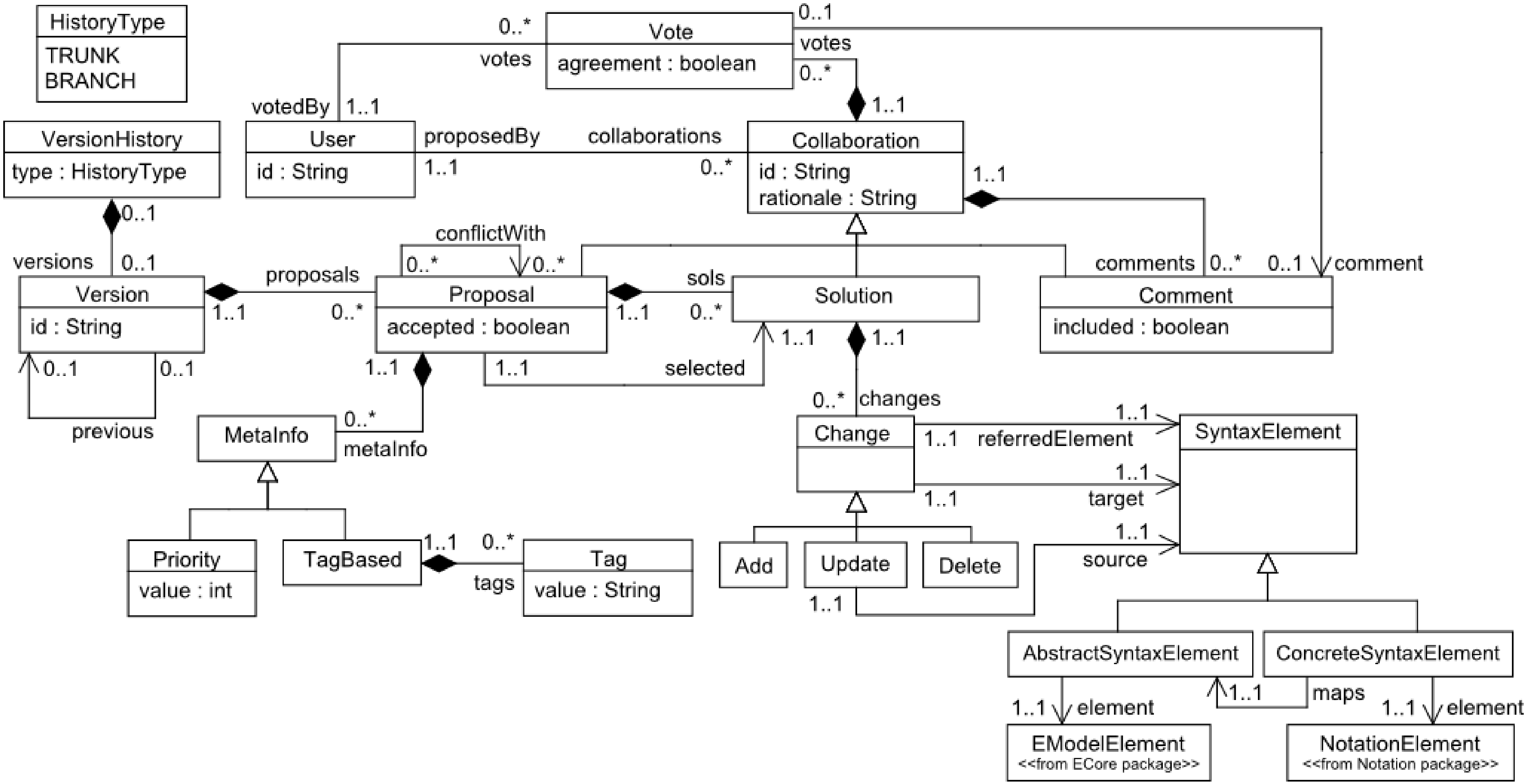
# Discussing the abstract syntax



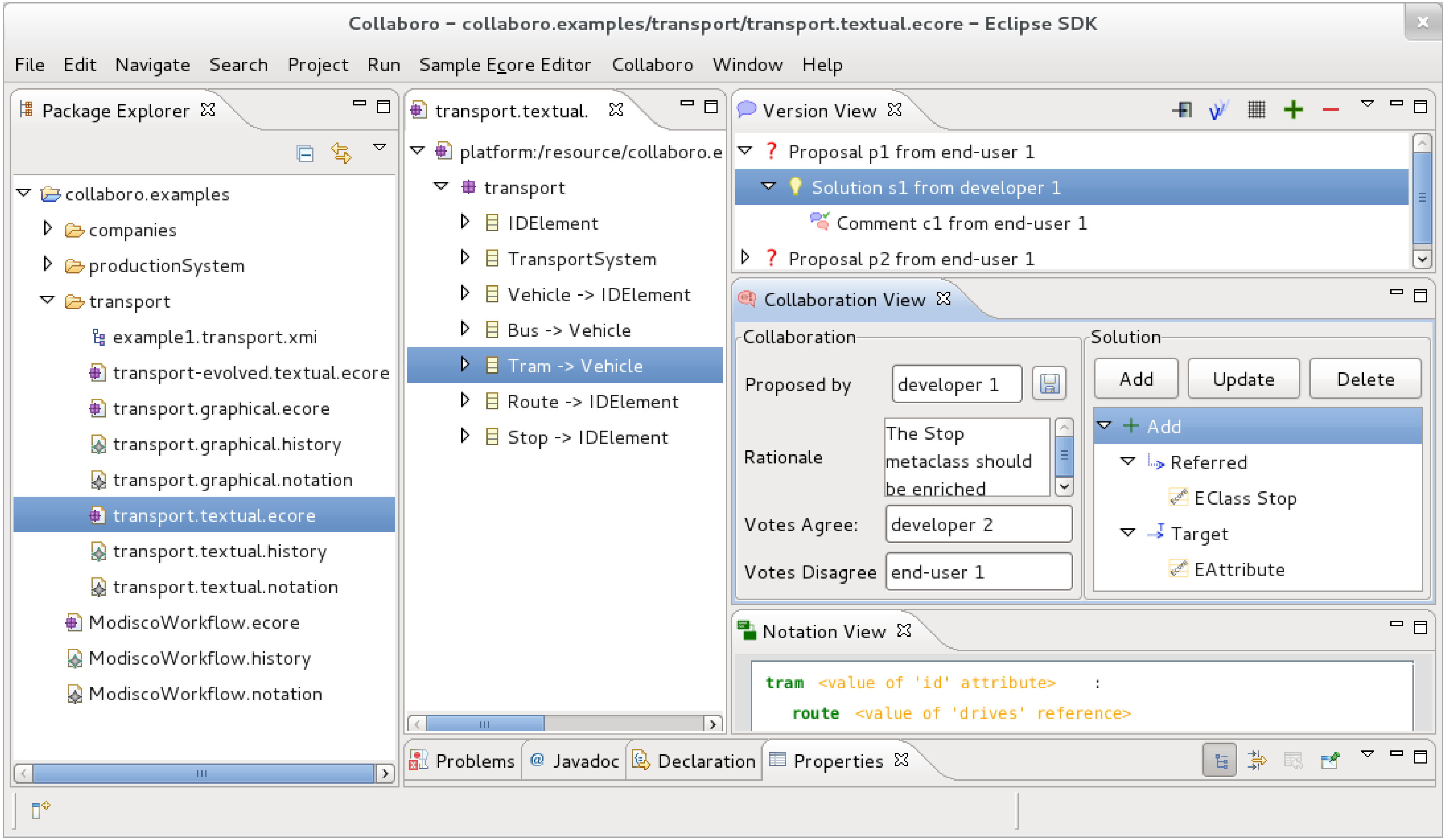
# Discussing the concrete syntax



# Representing collaborations



# Environment



Proposed by: developer 1

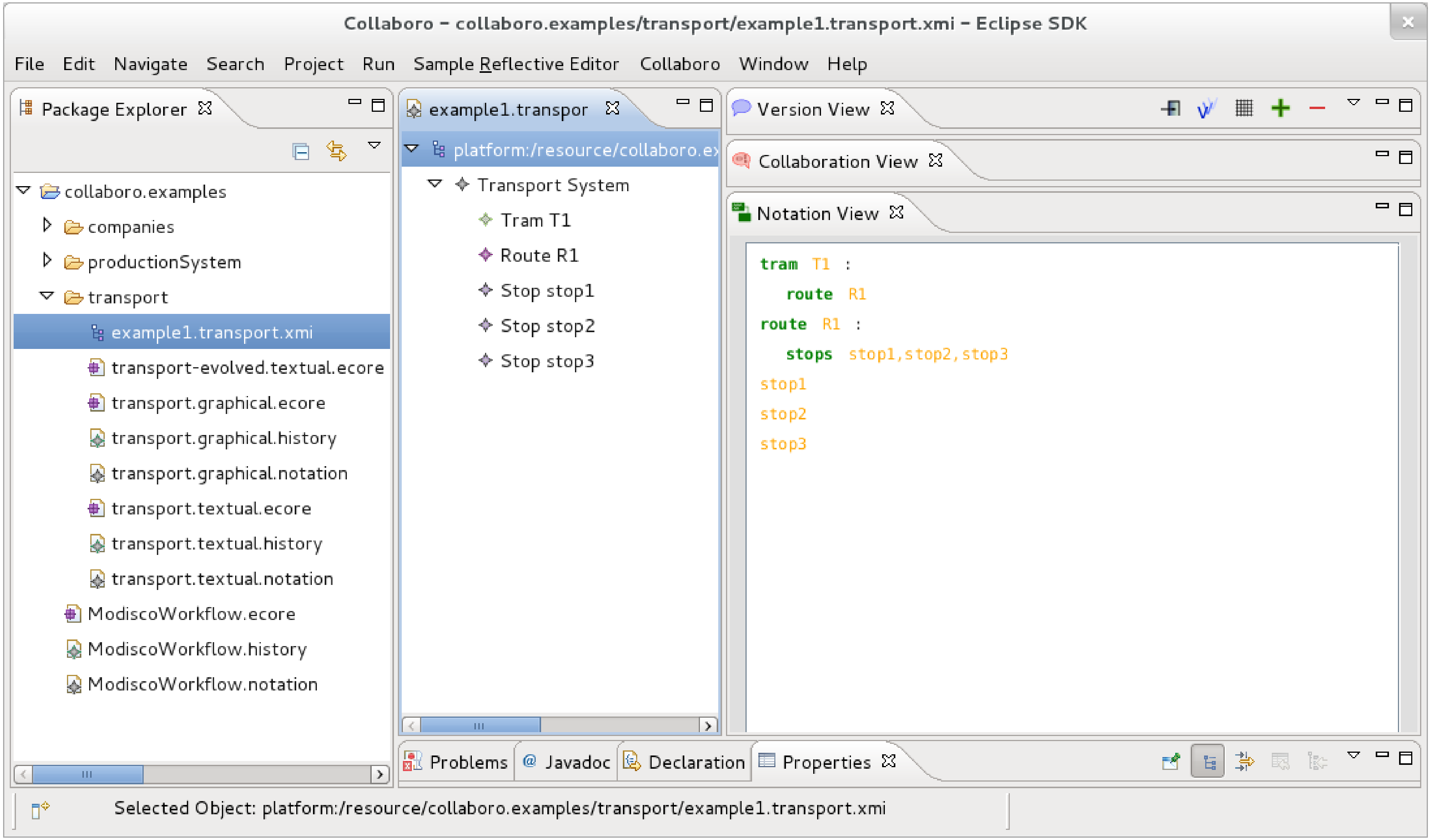
Rationale: The Stop metaclass should be enriched

Votes Agree: developer 2

Votes Disagree: end-user 1

```
tram  <value of 'id' attribute>    :
   route  <value of 'drives' reference>
```

# Environment



Collaboro – collaboro.examples/transport/example1.transport.xmi – Eclipse SDK

- collaboro.examples
  - companies
  - productionSystem
  - transport
    - example1.transport.xmi
    - transport-evolved.textual.ecore
    - transport.graphical.ecore
    - transport.graphical.history
    - transport.graphical.notation
    - transport.textual.ecore
    - transport.textual.history
    - transport.textual.notation
  - ModiscoWorkflow.ecore
  - ModiscoWorkflow.history
  - ModiscoWorkflow.notation

- platform:/resource/collaboro.ex
  - Transport System
    - Tram T1
    - Route R1
    - Stop stop1
    - Stop stop2
    - Stop stop3

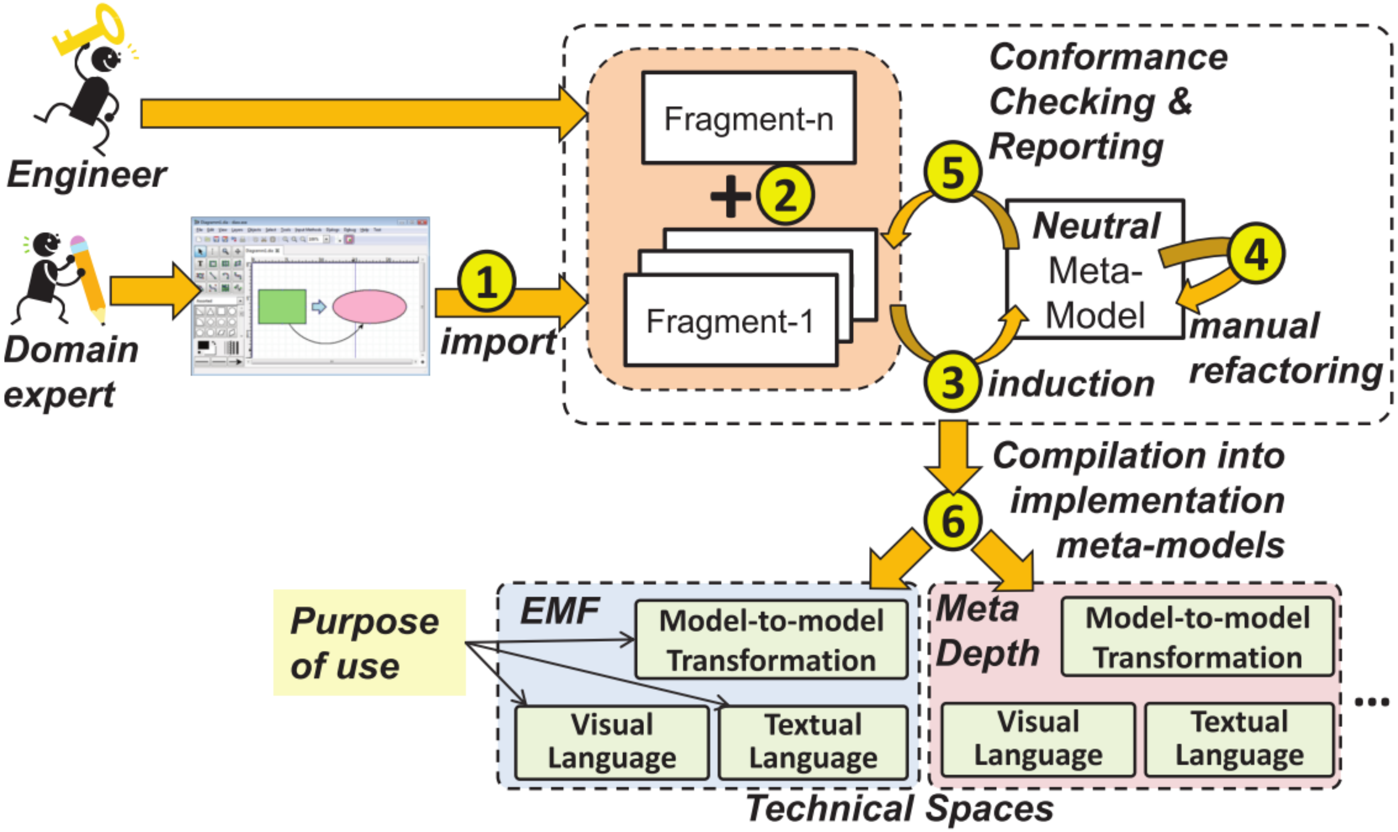
```
tram  T1  :
   route  R1
route  R1  :
   stops  stop1,stop2,stop3
stop1
stop2
stop3
```

Selected Object: platform:/resource/collaboro.examples/transport/example1.transport.xmi

# Not enough…

- Engagement is limited
  - End-users are required to express changes at high-level of abstraction
  - Solution: Example-driven collaboration
- Collaboration strategies
  - How to adapt the collaboration protocol?
  - Solution: Mechanism to define a democratic process
- Good notations
  - What is exactly a good notation?
  - Need of experimentation on this field
- Semantics
  - What happens with semantics?
  - Solution: Mechanisms to make easier the discussion about semantics

# Example-driven Bottom-up



Engineer

Domain expert

1 import

Fragment-n

+ 2

Fragment-1

Conformance Checking & Reporting

5

Neutral Meta-Model

4 manual refactoring

3 induction

Compilation into implementation meta-models

6

Purpose of use

EMF

Model-to-model Transformation

Visual Language

Textual Language

Meta Depth

Model-to-model Transformation

Visual Language

Textual Language

…

Technical Spaces

# The DSL case - During

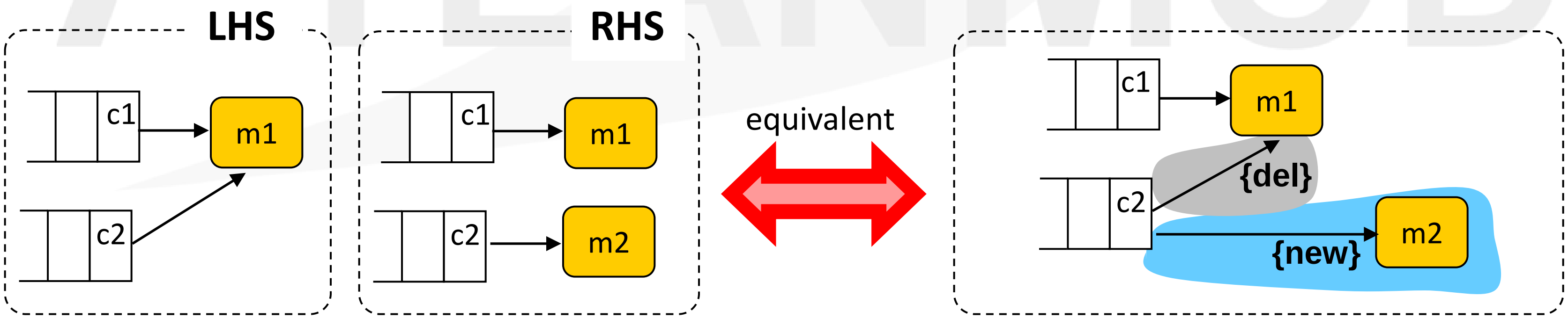
# Automatic testing of the user experience (interactive)

- Specially for concrete syntaxes
- Reuse what we know from web interaction and design
  - Small changes can make a huge difference
  - Even seasoned designers fail to predict upfront what would work
  - Different user profiles may require different concrete syntaxes
- What about A/B testing for DSLs?

# Automatic testing of the user experience (interactive)

- Which syntax for expressing transformation rules is better?
  - It’s up to the users to choose!!!

Rule *newMachine* Rule *newMachine*



*Extended (declarative) notation* *Compacted (operational) notation*

# The DSL case - After

# Corpus analysis (post mortem)

- Analysis of repositories of DSL models (i.e. instances of the DSL under analysis)

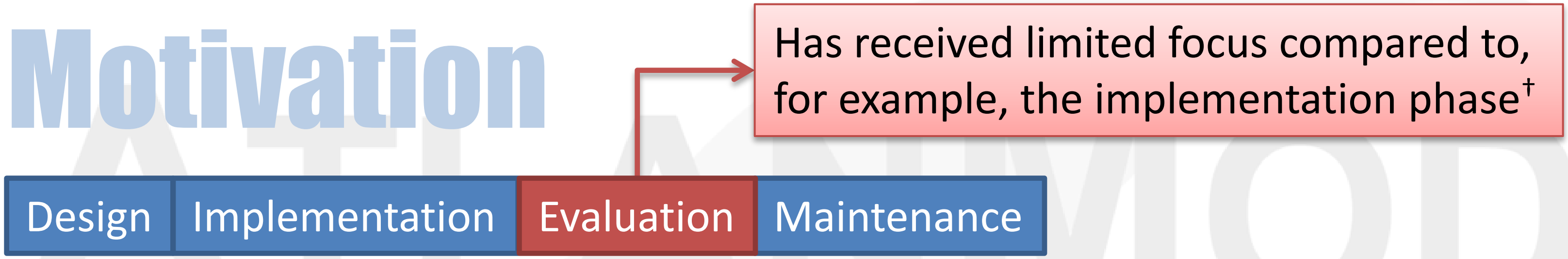
- We can analyze:
  - (meta) classes that are never used <- irrelevant?
  - Clusters in the DSL <- two subDSLs?
  - Complex structures (clones) that appear often <- is the DSL missing an important element?

# Corpus-based DSL Analysis

## Definition

*Domain specific languages* (DSLs) – Languages tailored for a specific domain

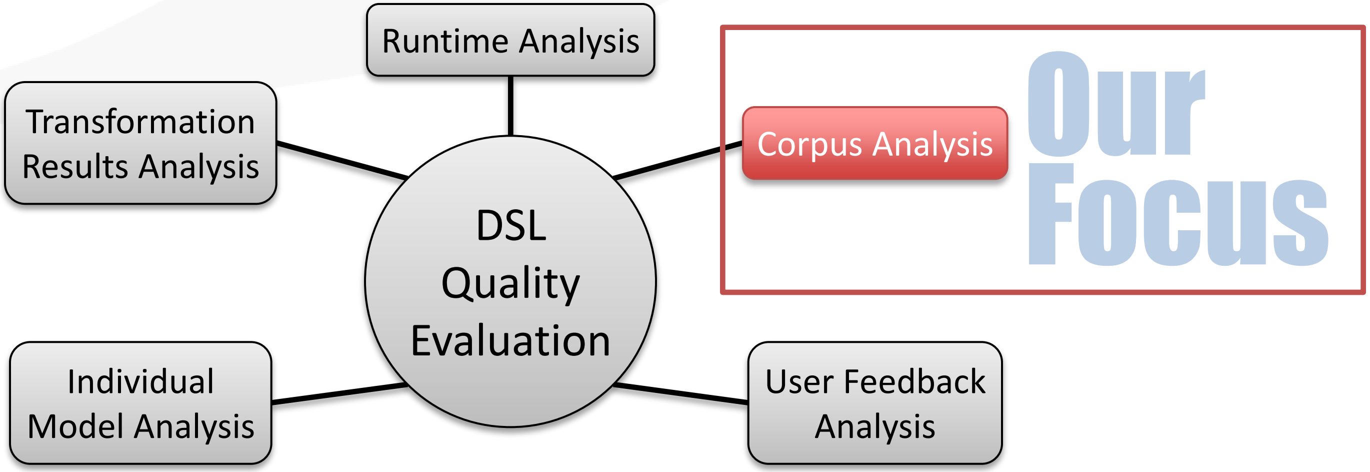
## Motivation



Has received limited focus compared to, for example, the implementation phase[†]

Design | Implementation | Evaluation | Maintenance

DSL lifecycle



DSL Quality Evaluation

Runtime Analysis

Transformation Results Analysis

Corpus Analysis

Our Focus

Individual Model Analysis

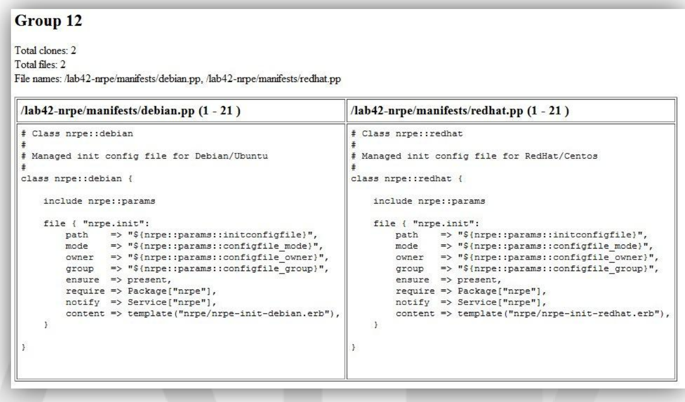
User Feedback Analysis

[†] Gabriel et al. (2010) *Do software languages engineers evaluate their languages*?

# Corpus-based DSL Analysis

## Clone Analysis

*Identify duplication within the language*



**Group 12**

Total clones: 2
Total files: 2
File names: /lab42-nrpe/manifests/debian.pp, /lab42-nrpe/manifests/redhat.pp

**/lab42-nrpe/manifests/debian.pp (1 - 21 )**

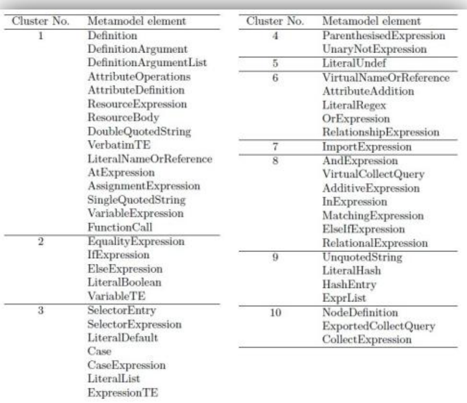
```
# Class nrpe::debian
#
# Managed init config file for Debian/Ubuntu
#
class nrpe::debian {

    include nrpe::params

    file { "nrpe.init":
        path    => "${nrpe::params::initconfigfile}",
        mode    => "${nrpe::params::configfile_mode}",
        owner   => "${nrpe::params::configfile_owner}",
        group   => "${nrpe::params::configfile_group}",
        ensure  => present,
        require => Package["nrpe"],
        notify  => Service["nrpe"],
        content => template("nrpe/nrpe-init-debian.erb"),
    }

}
```

**/lab42-nrpe/manifests/redhat.pp (1 - 21 )**

```
# Class nrpe::redhat
#
# Managed init config file for RedHat/Centos
#
class nrpe::redhat {

    include nrpe::params

    file { "nrpe.init":
        path    => "${nrpe::params::initconfigfile}",
        mode    => "${nrpe::params::configfile_mode}",
        owner   => "${nrpe::params::configfile_owner}",
        group   => "${nrpe::params::configfile_group}",
        ensure  => present,
        require => Package["nrpe"],
        notify  => Service["nrpe"],
        content => template("nrpe/nrpe-init-redhat.erb"),
    }

}
```

## Relationship Analysis

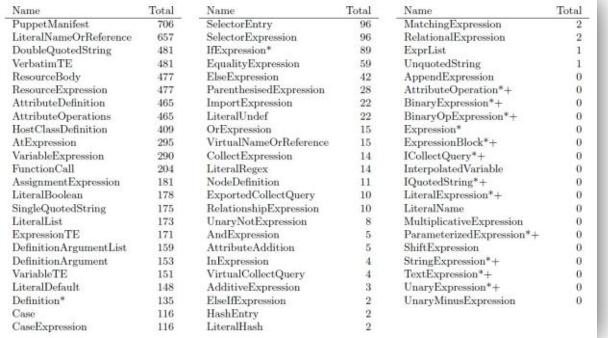
*Identify metamodel element relationships*



| Cluster No. | Metamodel element |
|---|---|
| 1 | Definition |
| | DefinitionArgument |
| | DefinitionArgumentList |
| | AttributeOperations |
| | AttributeDefinition |
| | ResourceExpression |
| | ResourceBody |
| | DoubleQuotedString |
| | VerbatimTE |
| | LiteralNameOrReference |
| | AtExpression |
| | AssignmentExpression |
| | SingleQuotedString |
| | VariableExpression |
| | FunctionCall |
| 2 | EqualityExpression |
| | IfExpression |
| | ElseExpression |
| | LiteralBoolean |
| | VariableTE |
| 3 | SelectorEntry |
| | SelectorExpression |
| | LiteralDefault |
| | Case |
| | CaseExpression |
| | LiteralList |
| | ExpressionTE |
| 4 | ParenthesisedExpression |
| | UnaryNotExpression |
| 5 | LiteralUndef |
| 6 | VirtualNameOrReference |
| | AttributeAddition |
| | LiteralRegex |
| | OrExpression |
| | RelationshipExpression |
| 7 | ImportExpression |
| 8 | AndExpression |
| | VirtualCollectQuery |
| | AdditiveExpression |
| | InExpression |
| | MatchingExpression |
| | ElseIfExpression |
| | RelationalExpression |
| 9 | UnquotedString |
| | LiteralHash |
| | HashEntry |
| | ExprList |
| 10 | NodeDefinition |
| | ExportedCollectQuery |
| | CollectExpression |

## Instance Analysis

*Identify metamodel element usage*



| Name | Total |
|---|---|
| PuppetManifest | 706 |
| LiteralNameOrReference | 657 |
| DoubleQuotedString | 481 |
| VerbatimTE | 481 |
| ResourceBody | 477 |
| ResourceExpression | 477 |
| AttributeDefinition | 465 |
| AttributeOperations | 465 |
| HostClassDefinition | 409 |
| AtExpression | 295 |
| VariableExpression | 290 |
| FunctionCall | 204 |
| AssignmentExpression | 181 |
| LiteralBoolean | 178 |
| SingleQuotedString | 175 |
| LiteralList | 173 |
| ExpressionTE | 171 |
| DefinitionArgumentList | 159 |
| DefinitionArgument | 153 |
| VariableTE | 151 |
| LiteralDefault | 148 |
| Definition* | 135 |
| Case | 116 |
| CaseExpression | 116 |
| SelectorEntry | 96 |
| SelectorExpression | 96 |
| IfExpression* | 89 |
| EqualityExpression | 59 |
| ElseExpression | 42 |
| ParenthesisedExpression | 28 |
| ImportExpression | 22 |
| LiteralUndef | 22 |
| OrExpression | 15 |
| VirtualNameOrReference | 15 |
| CollectExpression | 14 |
| LiteralRegex | 14 |
| NodeDefinition | 11 |
| ExportedCollectQuery | 10 |
| RelationshipExpression | 10 |
| UnaryNotExpression | 8 |
| AndExpression | 5 |
| AttributeAddition | 5 |
| InExpression | 4 |
| VirtualCollectQuery | 4 |
| AdditiveExpression | 3 |
| ElseIfExpression | 2 |
| HashEntry | 2 |
| LiteralHash | 2 |
| MatchingExpression | 2 |
| RelationalExpression | 2 |
| ExprList | 1 |
| UnquotedString | 1 |
| AppendExpression | 0 |
| AttributeOperation*+ | 0 |
| BinaryExpression*+ | 0 |
| BinaryOpExpression*+ | 0 |
| Expression* | 0 |
| ExpressionBlock*+ | 0 |
| ICollectQuery*+ | 0 |
| InterpolatedVariable | 0 |
| IQuotedString*+ | 0 |
| LiteralExpression*+ | 0 |
| LiteralName | 0 |
| MultiplicativeExpression | 0 |
| ParameterizedExpression*+ | 0 |
| ShiftExpression | 0 |
| StringExpression*+ | 0 |
| TextExpression*+ | 0 |
| UnaryExpression*+ | 0 |
| UnaryMinusExpression | 0 |

## DSLs under evaluation/consideration:

puppet labs

ATL

Canappi

ALLOY

OCL