Logiques de Description Appliquées

- François de Bertrand de Beuvron
francois.debertranddebeuvron@insa-strasbourg.fr

· Laboratoire de Génie de la Conception (LGECO)
http://www.insa-strasbourg.fr/fr/lgeco/

· INSA Strasbourg
http://www.insa-strasbourg.fr/

Méta-Plan

- INSA Strasbourg
- LGECO
- LICIA
- Logiques de description
- Conclusion et perspectives

Méta-Plan

- INSA Strasbourg
- LGECO
- LICIA
- Logiques de description
- Conclusion et perspectives

Institut National des Sciences Appliquées de Strasbourg

Architectes + Ingénieurs

INSA STRASBOURG

[Comprendre le monde, agir dans l'entreprise]

Le réseau des INSA

■ INSA de Lyon :	4 200 élèves, créé en 1957
■ INSA de Rennes :	1 270 élèves, créé en 1967
■ INSA de Rouen :	1 300 élèves, créé en 1985
■ INSA de Toulouse :	1 830 élèves, créé en 1967
■ **INSA de Strasbourg :**	**1 250 élèves, créé en 2003,**

(ex ENSAIS)

Rouen
INSA
Strasbourg
INSA
Rennes
INSA
Lyon
INSA
Toulouse
INSA

8 Spécialités d’ingénieurs et architectes

∀Architecture

∀Génie Civil

∀Topographie

∀Génie Mécanique

∀Plasturgie

∀Mécatronique

∀Génie électrique

∀Habitat et énergie

LES LABORATOIRES OU ÉQUIPES DE RECHERCHE

- **Laboratoire de génie de la conception** (EA 3938)
  - équipe de recherche en innovation, conception et intelligence artificielle (LICIA)
  - équipe de recherche en d'Ingénierie des Surfaces de Strasbourg (LISS)
  - équipe de recherche eau, sol, aménagement (ERESA)

- **Laboratoire des systèmes photoniques LSP**
(EA commune ULP/INSA 3426)

- **Equipe de recherche en génie électrique ERGE** (labo GREEN UMR 7037)

- **Equipe de recherche en photogrammétrie architecturale et géomatique PAGE** (laboratoire multisites MAP UMR 694)

Equipe de recherche en architecture, morphogénèse urbaine et projet Amup commune avec l'Ensas, habilitée par le Ministère de la culture et de la communication.

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE STRASBOURG

francois.debertranddebeuvron@insa-strasbourg.fr

Méta-Plan

- INSA Strasbourg
- LGECO
- LICIA

- Logiques de description

- Conclusion et perspectives

Garde INSA

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE STRASBOURG

ARCHITECTES + INGÉNIEURS

Laboratoire de Génie de la Conception

INSA
STRASBOURG

[Comprendre le monde, agir dans l'entreprise]

Composition du laboratoire

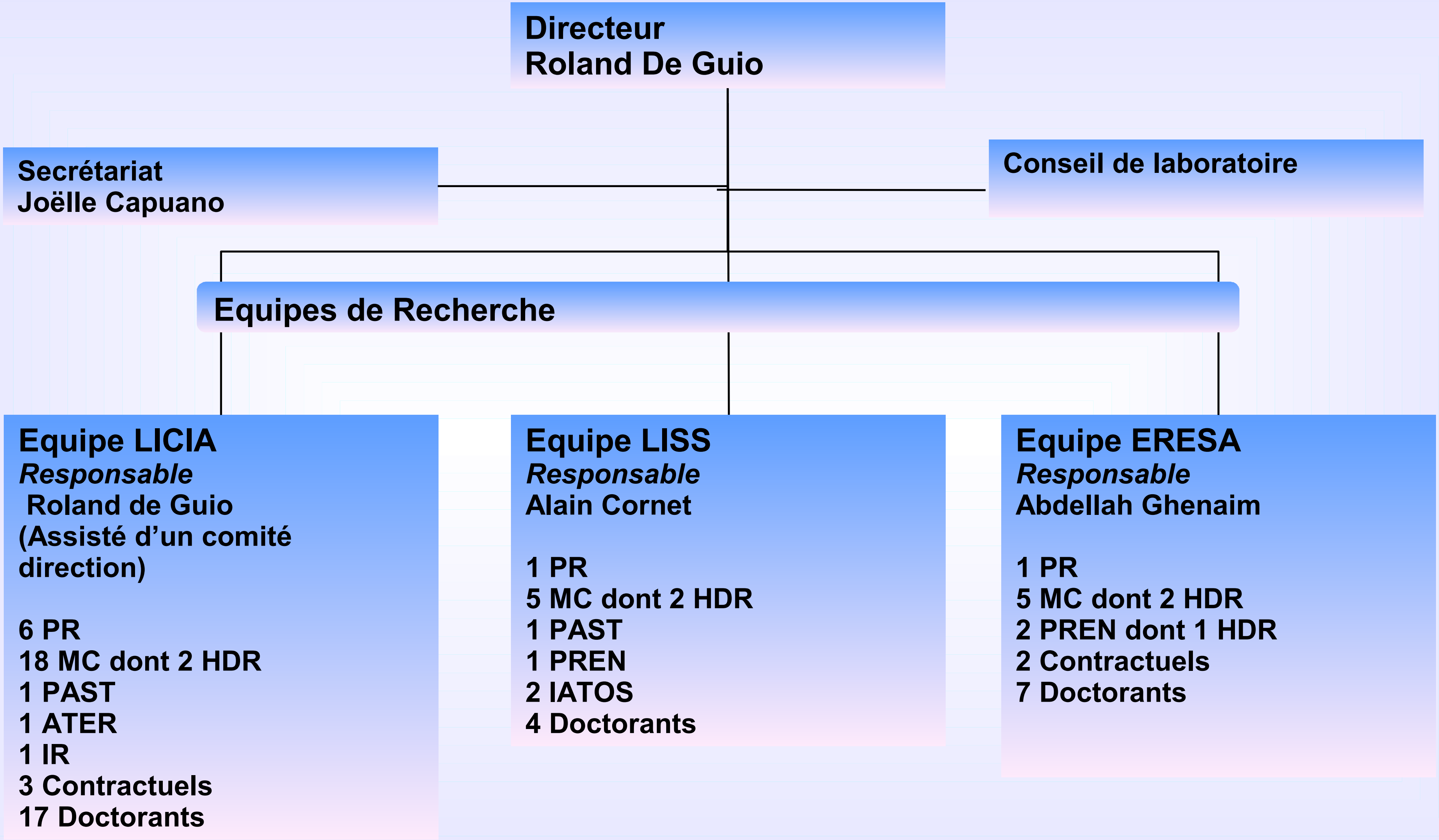
- **Enseignant-chercheurs** : 36
  - PR : 8
  - MC : 28 dont 6 HDR
- **PAST** : 2
- **PREN** : 3 dont 1 HDR
- **ATER** : 1

- **IATOS** : 4 (2.1 etp)
- **Contractuels** : 5

- **Doctorants** : 28

79 personnes en janvier 2008

Organigramme



Directeur
Roland De Guio

Secrétariat
Joëlle Capuano

Conseil de laboratoire

Equipes de Recherche

Equipe LICIA
Responsable
Roland de Guio
(Assisté d'un comité direction)

6 PR
18 MC dont 2 HDR
1 PAST
1 ATER
1 IR
3 Contractuels
17 Doctorants

Equipe LISS
Responsable
Alain Cornet

1 PR
5 MC dont 2 HDR
1 PAST
1 PREN
2 IATOS
4 Doctorants

Equipe ERESA
Responsable
Abdellah Ghenaim

1 PR
5 MC dont 2 HDR
2 PREN dont 1 HDR
2 Contractuels
7 Doctorants

Méta-Plan

- INSA Strasbourg
- LGECO
- LICIA
- Logiques de description
- Conclusion et perspectives

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE STRASBOURG

ARCHITECTES + INGÉNIEURS

INSA STRASBOURG

LGECO - Equipe LICIA

Ingénierie de conception, cognition et intelligence artificielle

47 personnes au 1er janvier 2008

- Responsable : Roland De Guio
- **6 PR :**

Roland de Guio	PR2 61ème
Mickaël Gardoni	PR2 60ème
Bernard Keith	PR2 61ème
Michel Sonntag	PR1 70ème
Emmanuel Caillaud	PR2 60ème (ULP)
Dominique Knittel	PR2 60ème (ULP)

- **1 PAST** : Nikolai Khomenko
- **1 ATER** : Anne-Sophie Lichtle (post-doc)
- **1 IR** : Sébastien Dubois
- **3 Contractuels** :
- Philippe Bouche
- Adrien Guenebaut (post-doc)
- Andri Rasakarisoa

- **17 Doctorants**

- **18 MC dont 2 HDR**

Marc Barth	60ème	HDR
Denis Cavallucci	60ème	
Amadou Coulibaly	60ème	
Bertrand de Beuvron	27ème	
Nathalie Gartiser	06ème	
Virginie Goepp	61ème	
François Kieffer	60ème	
Laurence Meylheuc	60ème	
David Oget	70ème	
Olivier Piccin	60ème	
Pierre Renaud	60ème	
Eric Schenk	06ème	
Marc Vedrines	60ème	
Cecilia Zanni	27ème	
Rémy Houssin	60ème	(ULP)
Ivana Rasovska	61ème	(ULP)
Bertrand Rose	60ème	(ULP)
François Rousselot	27ème	HDR (UMB)

Nos centres d’intérêts en recherche (LICIA)

LICIA

- Méthodes de résolution de problèmes par les contradictions et supports informatiques
  - Formulation de problème : construction des réseaux de problème et de contradictions
  - Analyse de grands réseaux de problèmes/ contradictions/paramètres
  - Capitalisation des résultats de la résolution de problème contexte industriel de conception
  - Lien des standards inventifs, des bases de connaissance de la TRIZ
  - Amélioration continue du processus de résolution de problème et de ses supports
  - Modèles génériques pour les systèmes de production
- Management de l’innovation
  - Prévision des barrières technologiques (sur la base d’une approche système et des contradictions)
  - Intégration des méthodes de résolution de problème d’invention dans les entreprises
- Conception collaborative et ingénierie concurrente
  - Conception de l’architecture des systèmes d’information et de production
  - Résolution des conflits en ingénierie collaborative et concurrente
  - Conception de produits et d’outils en plasturgie (modélisation)
  - Processus de maintenance et de recyclage des produits – cadre PLM
  - Conception de robots médicaux chirurgicaux

Axe CI (Conception Intégrée)

Ingénierie Intégrée

Mise en cohérence de pratiques au cours du cycle du produit

(ex.: alignement procédé/produit, maintenabilité/produit, etc.)

Conception Collaborative

Partage de pratiques dans une même étape du cycle du produit

(ex. : modélisation & évaluation de performances en conception de produit, conception préliminaire de système de production et d'informations, etc.)

Axe SHS (Sciences Humaines et Sociales)

Organisation

- Intégration de l’innovation et de la conception dans l’organisation et la stratégie de l’entreprise

Compétence

- Conception de l’architecture cognitive de l’entreprise

Cognition

- Interaction OTSM-TRIZ et développement cognitif (recherche sur les métacognitions en conception et résolution de problèmes)

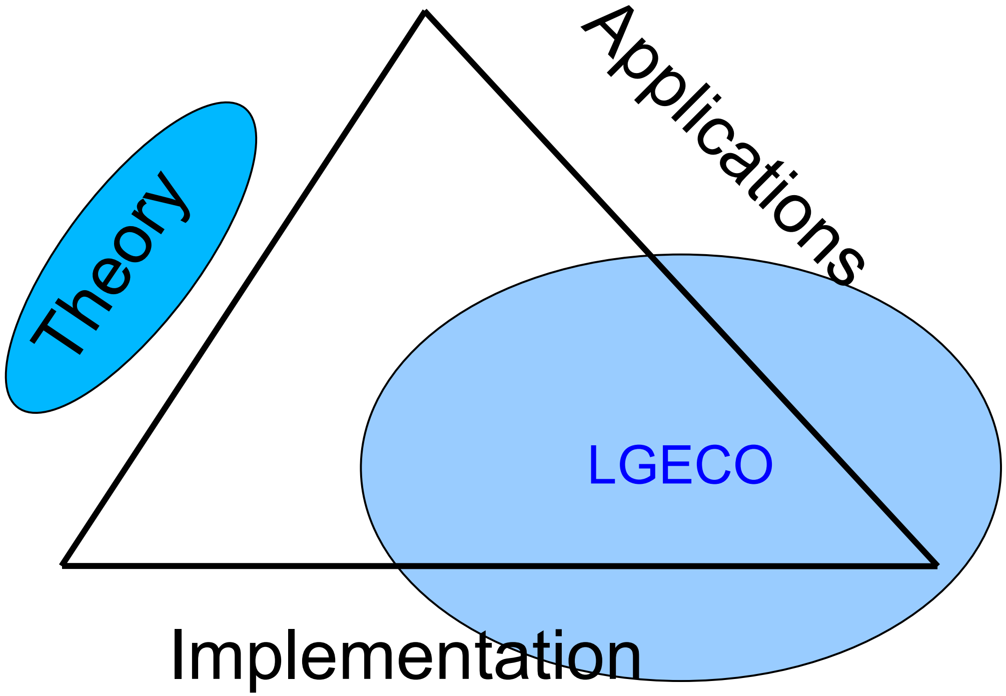
Axe COIA : ingénierie de la conception, cognition et intelligence artificielle

- Formalisation du raisonnement de formulation de problèmes
- Analyse et représentation des réseaux de contradiction
- Conception des systèmes de gestion de l’innovation
- Intégration des méthodes de conception inventives

Méta-Plan

- INSA Strasbourg
- LGECO
- LICIA
- Logiques de description
- Conclusion et perspectives

Description Logic



DLs as FOL Fragment (i)

- Atomic Concept C ↔ unary predicate
  - Man ↔ man(X)
- Atomic Role R ↔ binary predicate
  - hasChild ↔ hasChild(X,Y)
- Individual i ↔ constant
  - jean ↔ jean
- Operators (for forming concepts and roles)
  - A specific set of operators define a specific description logic
- Concept (expression) ↔ ***FOL formulae with one free variable***

DLs as FOL Fragment (ii)

- *$\mathcal{ALC}$*
  - Boolean operators

    ⊓, ⊔, ¬,

    ¬Person ⊔Man ⊔Woman

    ¬person(X) ∨ man(X) ∨ woman(X)
  - Only atomic roles
  - restricted role quantifiers

    ∃, ∀

    Person ⊓ ∀hasChild.Woman

    person(X) ∧ ∀Y [hasChild(X,Y) => woman(Y)]
- Ex : Person all of whose children are either Doctors or have a child who is a Doctor
  - Person ⊓ ∀hasChild.(Doctor ⊔∃hasChild.Doctor)

DL Family

- Additional letters indicate other extensions, e.g.:
  - R_+ transitive roles (e.g. ancestor is transitive)
  - $\mathcal{H}$ for role hierarchy (e.g., hasDaughter $\sqsubseteq$ hasChild)
  - $\mathcal{O}$ for nominals/singleton classes (e.g., {Italy})
  - $\mathcal{I}$ for inverse roles (e.g., isChildOf ↔ hasChild^{-1})
  - $\mathcal{N}$ for number restrictions (e.g., $\rhd$2hasChild, $\preceq$3hasChild)
  - $\mathcal{Q}$ for qualified number restrictions (e.g., $\rhd$2hasChild.Doctor)
  - $\mathcal{F}$ for functional number restrictions (e.g., $\preceq$1hasMother)
- $\mathcal{S}$ often used for $\mathcal{ALC}$ extended with transitive roles
- $\mathcal{SHIQ}$=$\mathcal{ALC}$+ transitive roles + role hierarchy + inverse + qualified number restriction

DL Knowledge Base

- A TBox is a set of “schema” axioms

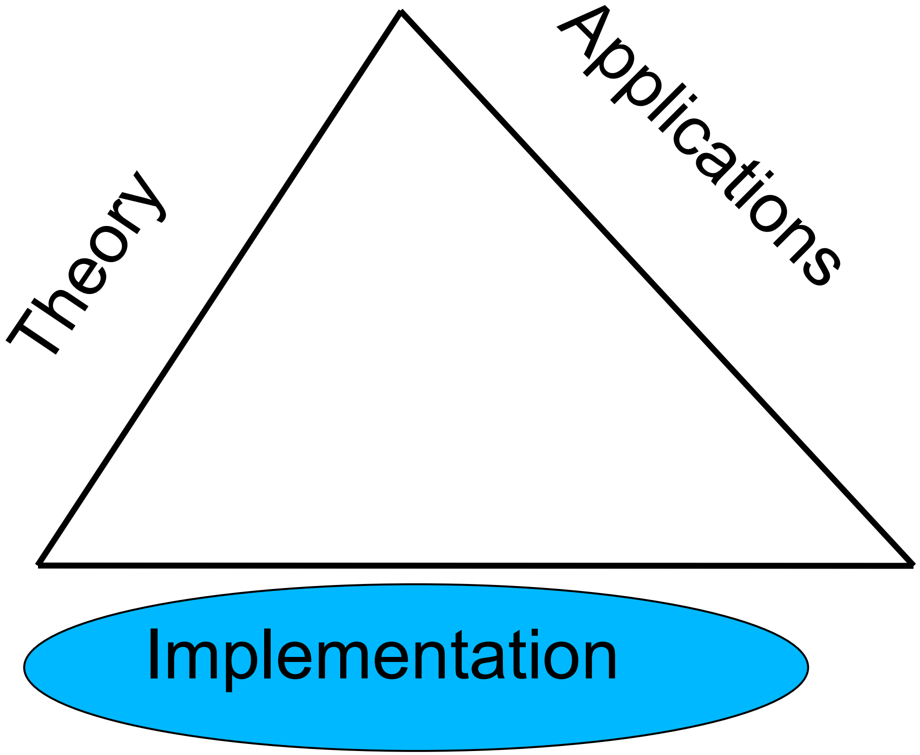
  $\{\text{Doctor} \Rightarrow \text{Person},$
  $\text{HappyParent} \Leftrightarrow \text{Person} \sqcap \forall\text{hasChild}.(\text{Doctor} \sqcup \exists\text{hasChild}.\text{Doctor})\}$

- An ABox is a set of “data” axioms (ground facts)

  {John:HappyParent,
  John hasChild Mary}

- A Knowledge Base (KB) is just a TBox and an ABox

Description Logic



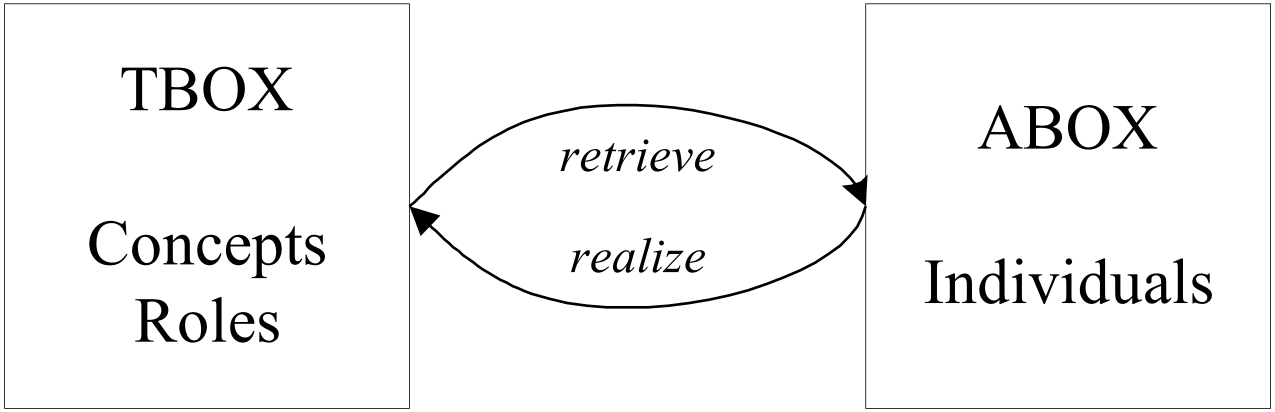
DL Services for Tboxes

- Coherence test (satisfiability of concepts)
  - Ex : HappyParent $\sqcap$ $\forall$hasChild.$\bot$ is not satisfiable
- Subsumption test
  - C1 subsume C2 $\Leftrightarrow$ C2 $\wedge$ $\neg$ C1 inconsistent
- Compatibility test
  - C1 and C2 disjoint $\Leftrightarrow$ C2 $\wedge$ C1 inconsistent
- Classification
  - Use subsumption to build concept hierarchy

DL Services for ABoxes

- Instance checking
  - test if instance is necessarily (in all models) in the extension of a concept
- Retrieval
  - retrieve all instance of a concept
- Realization
  - retrieve all (or least) concepts witch the individual is an instance of)

DL system



TBOX

Concepts
Roles

retrieve

realize

ABOX

Individuals

satisfiability
subsumption
coherence
classification

coherence

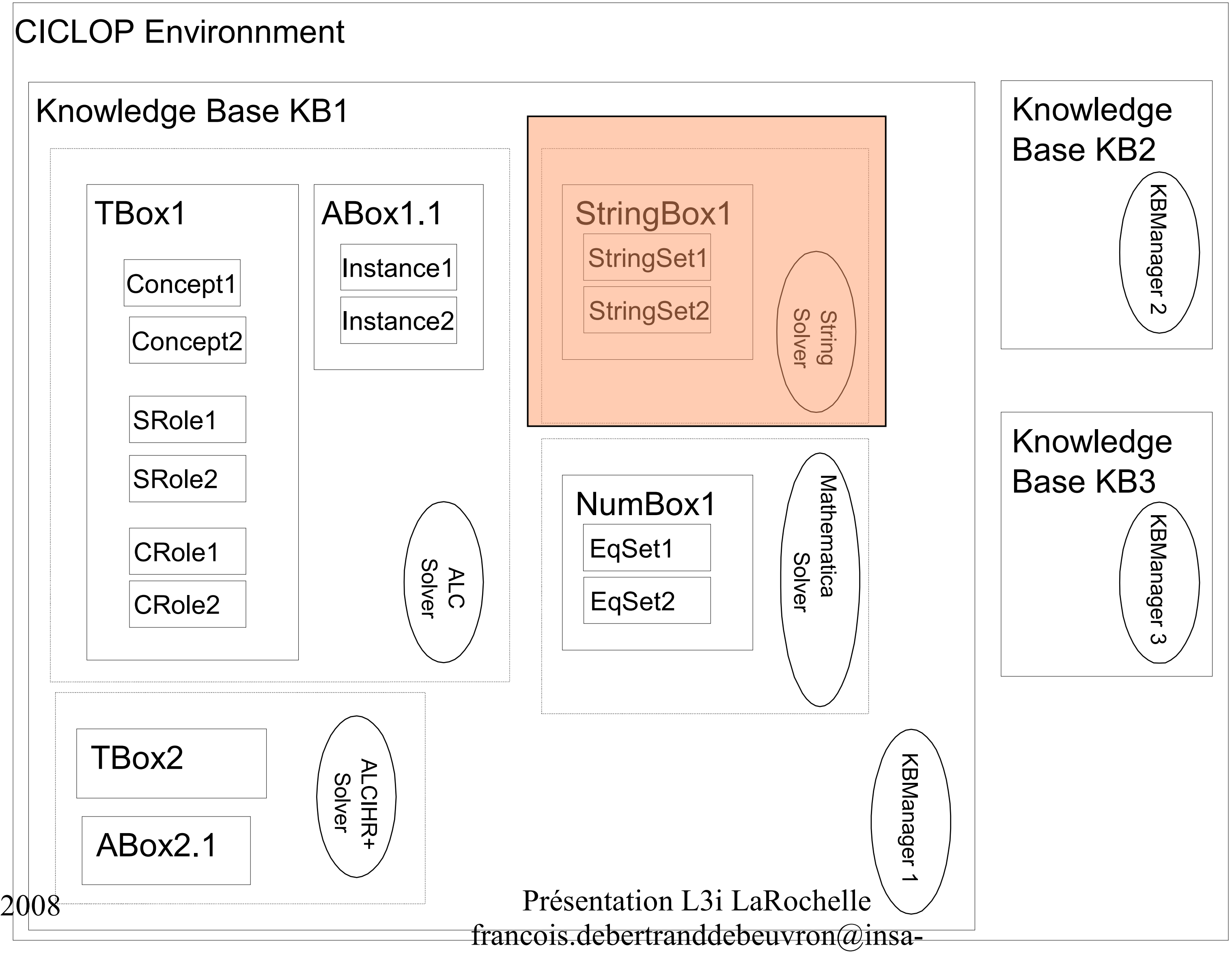
Ciclop Implementation

- Pure java
- Textual interface
- Graphical interface
- Java API
- Simple, file based client/server interface
- CORBA client/Server architecture
- XML support

CICLOP Expressivity

- $\mathcal{ALC}f\mathcal{IHR}_+$
  - Functional role (attribute)
    - Ex : *father*
  - Inverse role
  - Primitive Role Hierarchy
    - Ex : *father* => *parent*
  - Transitive role
    - $(x\ R\ y \wedge y\ R\ z \Rightarrow x\ R\ z)$
    - Ex : *ancestor* is transitive ; *parent* => *ancestor*
  - Concrete domains for String and Number

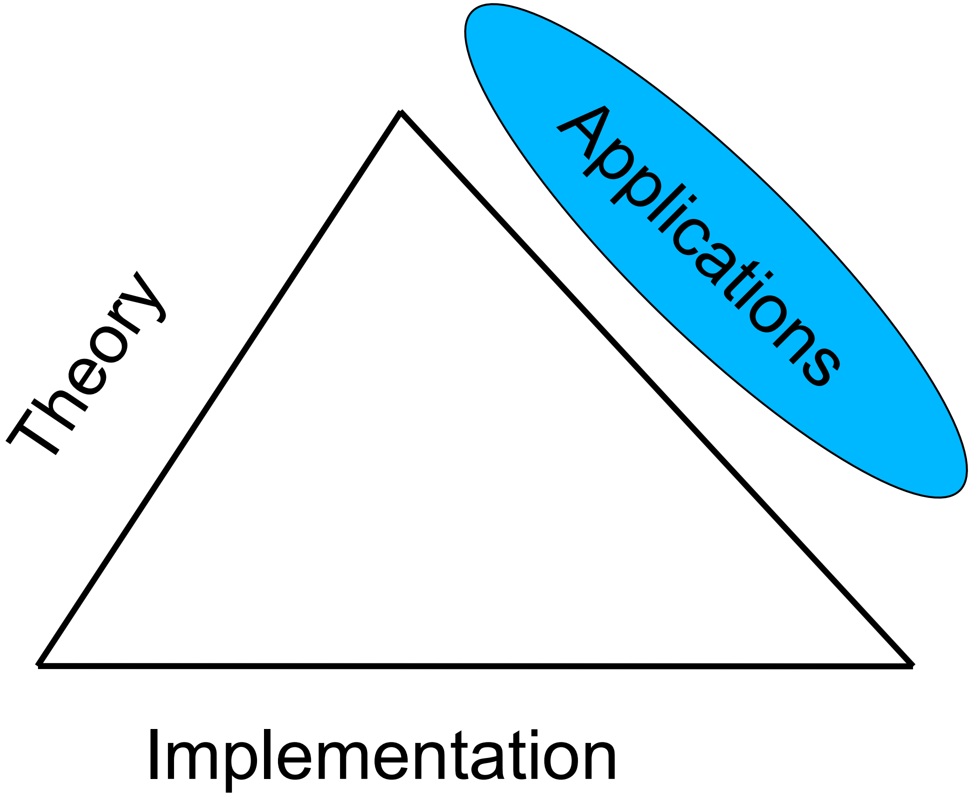
CICLOP (Multi-Tbox support)



CICLOP Services

- All standard services
  - Concept consistency, classification, Abox consistency, instance checking/realization/retrieval
- Multiple models computation/storage (V2)
  - Efficient caching of previous proofs in Abox
  - Efficient Abox incremental definition
  - Using computed models in applications

Description Logic

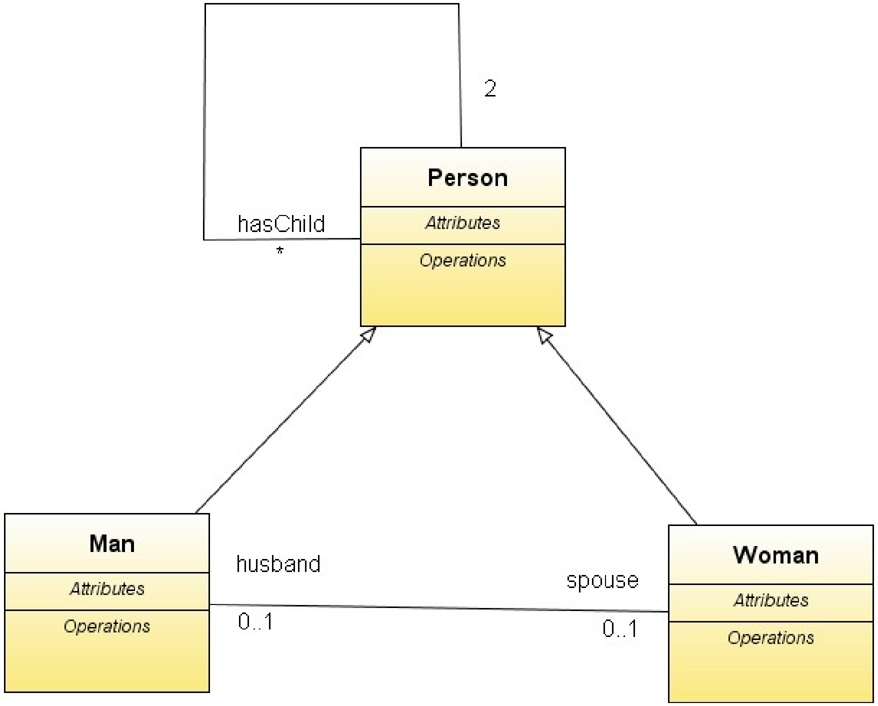


Applications

- Conceptual Modeling
- Query Optimization and View Maintenance
- Natural Language Semantics
- Terminologies and Ontology
- Information Access and Intelligent Interfaces
- Formal Specifications in Engineering
- Configuration
- Planning

Conceptual Modeling

- UML



DL vs OOR

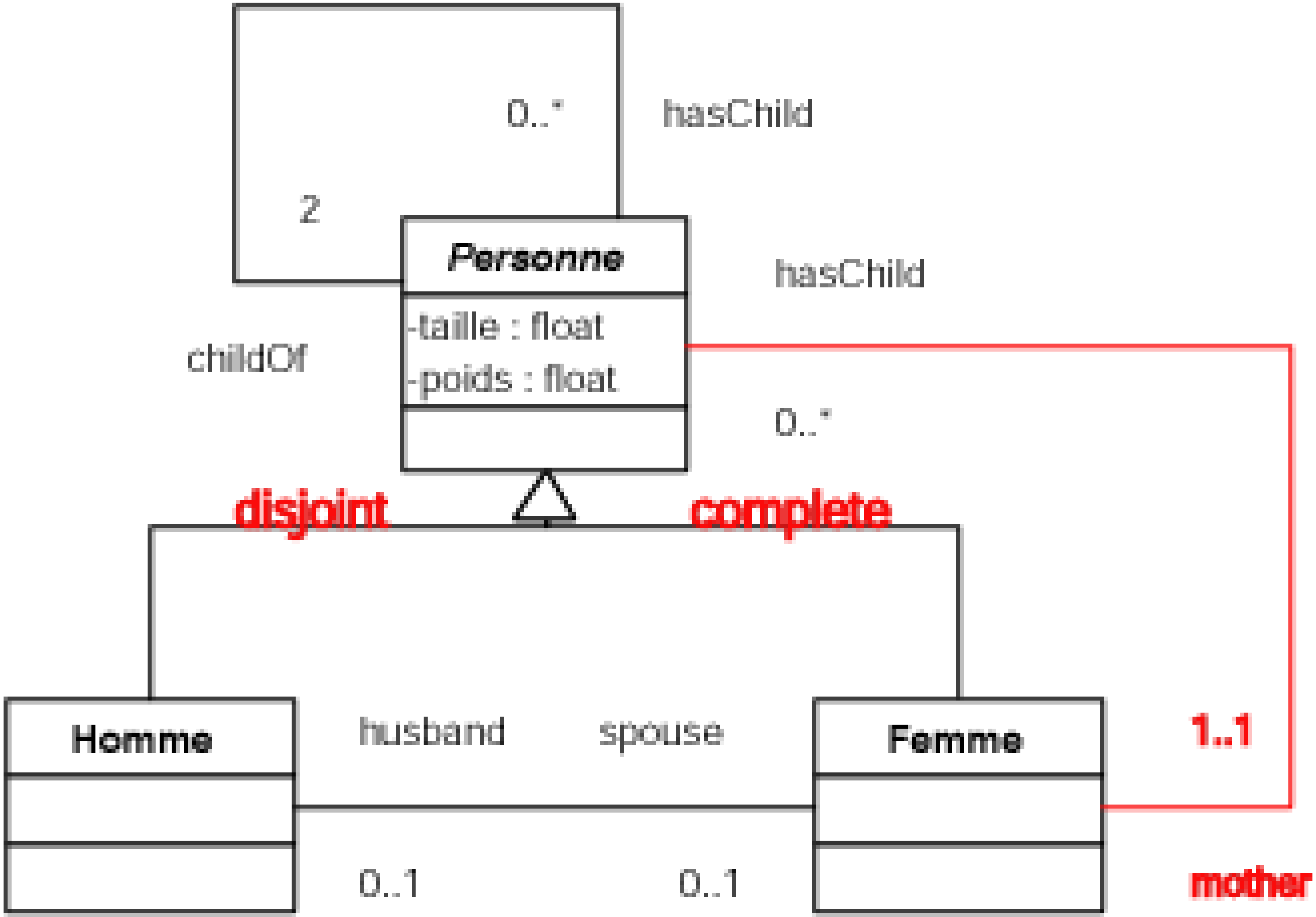
- Description Logic
  - Well defined semantic
  - Automatic classification
  - Open world assumption
    - Not consistent with standard DBMS
    - + Reasoning with incomplete knowledge
  - Extended expressivity

- Object Oriented Rep.
  - Many ambiguities
  - Explicit class hierarchy
  - Closed word assumption
    - Easy to interface with object-oriented or relational Data Base Management System

Extended expressivity

- Man and Woman are a partition of Person
- A person has only one mother
- Mother is a specific case of hasChild
- hasChild is a specific case of ancestor
- ancestor is transitive

UML (wrong)

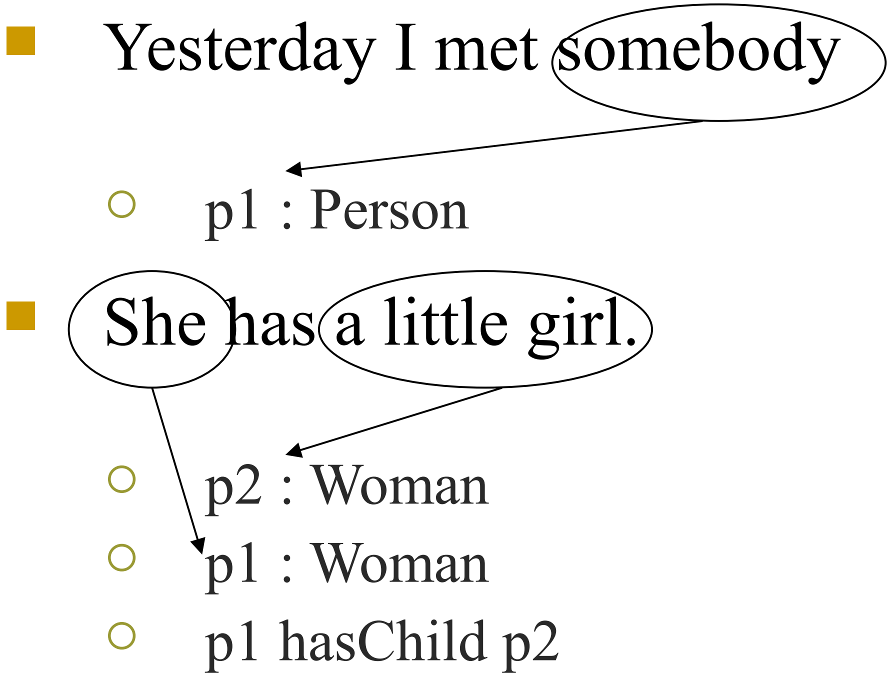

0..*
hasChild
2
Personne
-taille : float
-poids : float
hasChild
childOf
0..*
disjoint
complete
Homme
husband
spouse
Femme
1..1
0..1
0..1
mother

DL vs OOR

- Description Logic
  - Well defined semantic
  - Automatic classification
  - Open world assumption
    - Not consistent with standard DBMS
    + Reasoning with incomplete knowledge
  - Extended expressivity

- Object Oriented Rep.
  - Many ambiguities
  - Explicit class hierarchy
  - Closed word assumption
    - Easy to interface with object-oriented or relational Data Base Management System

Reasoning with incomplete knowledge



- Yesterday I met somebody
  - p1 : Person
- She has a little girl.
  - p2 : Woman
  - p1 : Woman
  - p1 hasChild p2

Ciclop Exemple (roles)

- (DefineAtomicRole isAncestorOf :domain Person :range Person :inversible true :inverseName isDescendantFrom :transitive true)
- (DefineRole hasChild isAncestorOf :inversible true :inverseName childOf)
- (DefineRole mother childOf)
- (DefineAtomicRole marriedWith :feature true :symetric true)

Ciclop exemple (concepts 1)

- (DefineConcept Person (and
  (or Man Woman)
  (Some mother Person)))

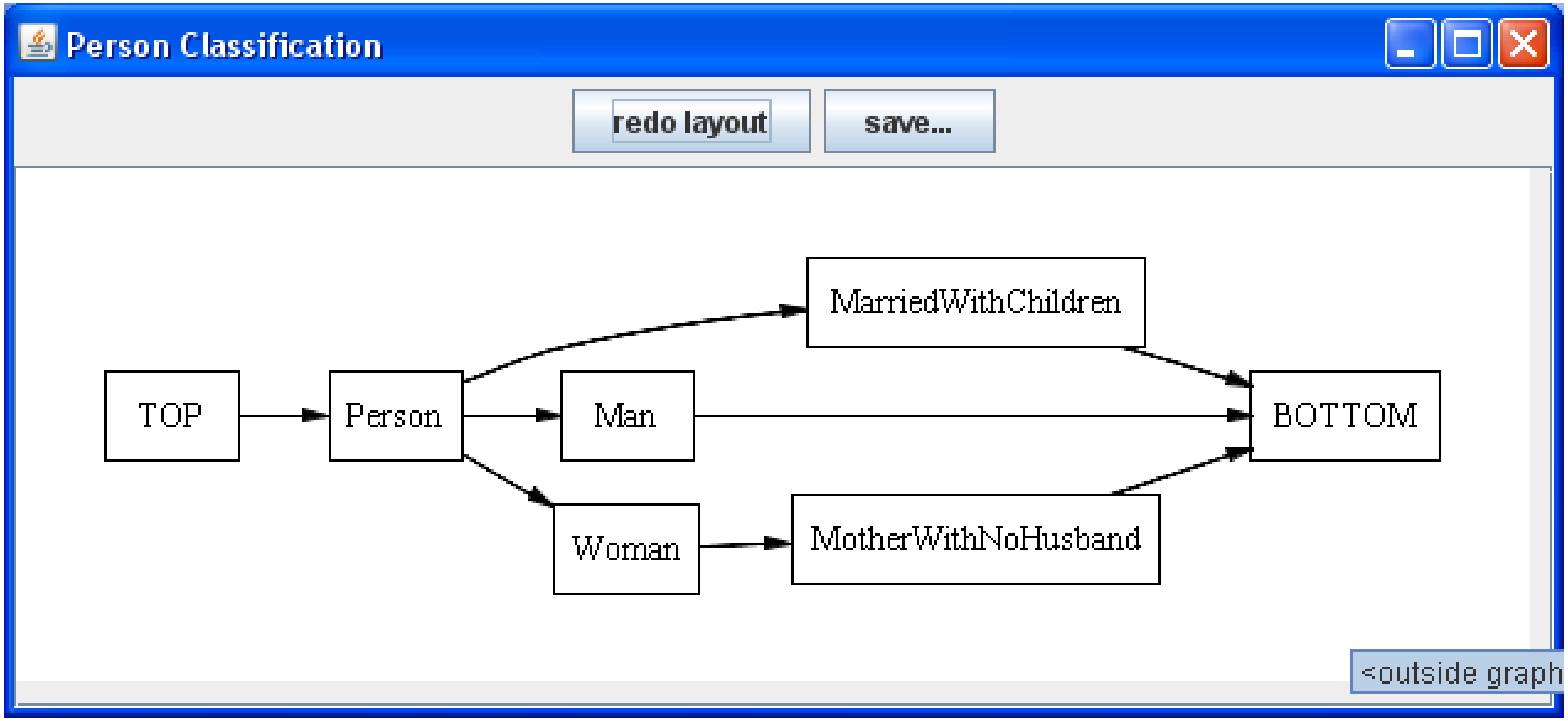
- (DefineConcept Man (and
  Person
  (not Woman)
  (all marriedWith Woman)))

- (DefineConcept Woman (and
  Person
  (not Man)
  (all marriedWith Man)))

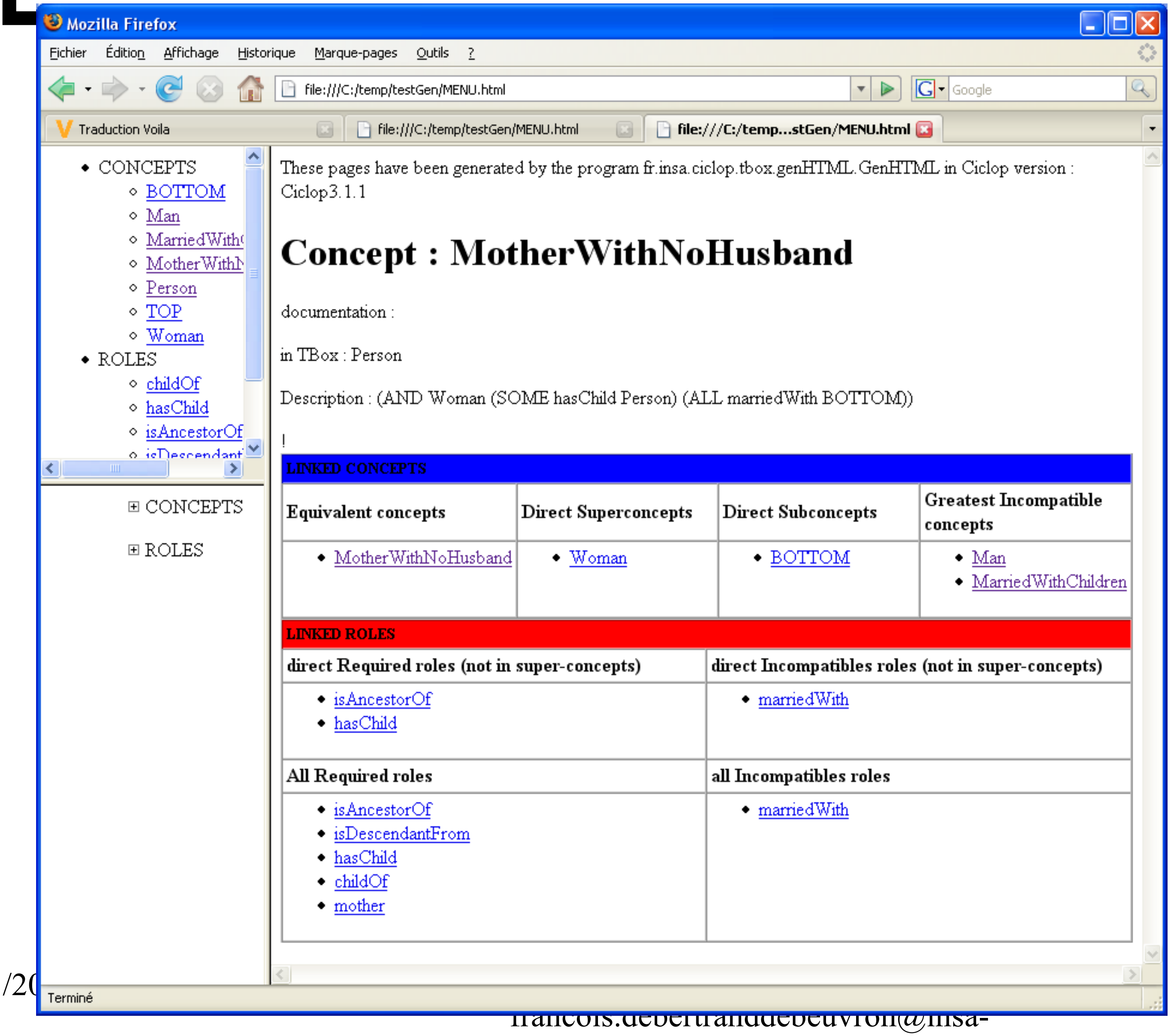
Ciclop exemple (concepts 2)

- (DefineConcept MarriedWithChildren (and
  Person
  (Some marriedWith Person)
  (Some hasChild Person)))
- (DefineConcept MotherWithNoHusband (and
  Woman
  (Some hasChild Person)
  (all marriedWith BOTTOM)))

Classification



Analyse


Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils ?
file:///C:/temp/testGen/MENU.html
Google
Traduction Voila
file:///C:/temp/testGen/MENU.html
file:///C:/temp...stGen/MENU.html
Terminé

- CONCEPTS
  - BOTTOM
  - Man
  - MarriedWith(
  - MotherWithN
  - Person
  - TOP
  - Woman
- ROLES
  - childOf
  - hasChild
  - isAncestorOf
  - isDescendant

⊞ CONCEPTS

⊞ ROLES

These pages have been generated by the program fr.insa.ciclop.tbox.genHTML.GenHTML in Ciclop version : Ciclop3.1.1

Concept : MotherWithNoHusband

documentation :

in TBox : Person

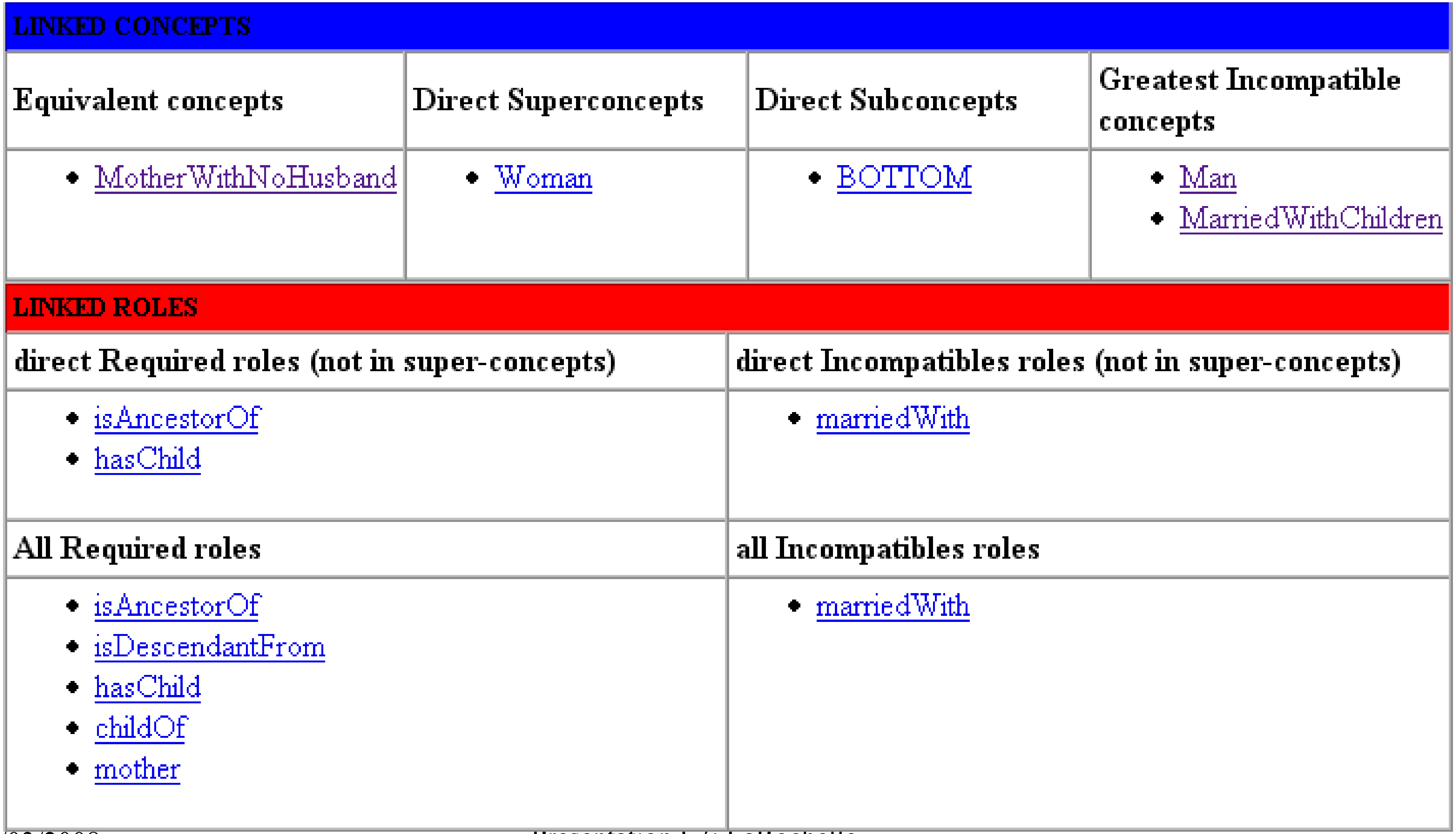
Description : (AND Woman (SOME hasChild Person) (ALL marriedWith BOTTOM))

!

LINKED CONCEPTS			
Equivalent concepts	**Direct Superconcepts**	**Direct Subconcepts**	**Greatest Incompatible concepts**
• MotherWithNoHusband	• Woman	• BOTTOM	• Man • MarriedWithChildren

LINKED ROLES	
direct Required roles (not in super-concepts)	**direct Incompatibles roles (not in super-concepts)**
• isAncestorOf • hasChild	• marriedWith
All Required roles	**all Incompatibles roles**
• isAncestorOf • isDescendantFrom • hasChild • childOf • mother	• marriedWith

Analyse



LINKED CONCEPTS			
Equivalent concepts	**Direct Superconcepts**	**Direct Subconcepts**	**Greatest Incompatible concepts**
• MotherWithNoHusband	• Woman	• BOTTOM	• Man • MarriedWithChildren

LINKED ROLES	
direct Required roles (not in super-concepts)	**direct Incompatibles roles (not in super-concepts)**
• isAncestorOf • hasChild	• marriedWith
All Required roles	**all Incompatibles roles**
• isAncestorOf • isDescendantFrom • hasChild • childOf • mother	• marriedWith

Applications

- Conceptual Modeling
- Query Optimization and View Maintenance
- Natural Language Semantics
- Terminologies and Ontology
- Information Access and Intelligent Interfaces
- Formal Specifications in Engineering
- Configuration
- Planning

OWL (from Horrocks)

The Web Ontology Language OWL

- Semantic Web led to requirement for a “web ontology language”
- set up Web-Ontology (**WebOnt**) Working Group
  - WebOnt developed **OWL** language
  - OWL based on earlier languages **OIL** and **DAML+OIL**
  - OWL now a W3C **recommendation** (i.e., a standard)
- OIL, DAML+OIL and OWL based on **Description Logics**
  - OWL effectively a “Web-friendly” syntax for ***SHOIN***

OWL syntax

$\text{Person} \sqcap \forall \text{hasChild}.(\text{Doctor} \sqcup \exists \text{hasChild}.\text{Doctor})$:

```
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:allValuesFrom>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#hasChild"/>
            <owl:someValuesFrom rdf:resource="#Doctor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:allValuesFrom>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Applications

- Conceptual Modeling
- Query Optimization and View Maintenance
- Natural Language Semantics
- Terminologies and Ontology
- Information Access and Intelligent Interfaces
- Formal Specifications in Engineering
- Configuration
- Planning

Station LIKES

- Développée au LGECO
  - François Rousselot
- Peu de ressources linguistiques: pas de dictionnaire
  - circonstances, transportabilité
- Simplicité d'utilisation
  - Usage par des linguistes
  - Parenté avec INTEX

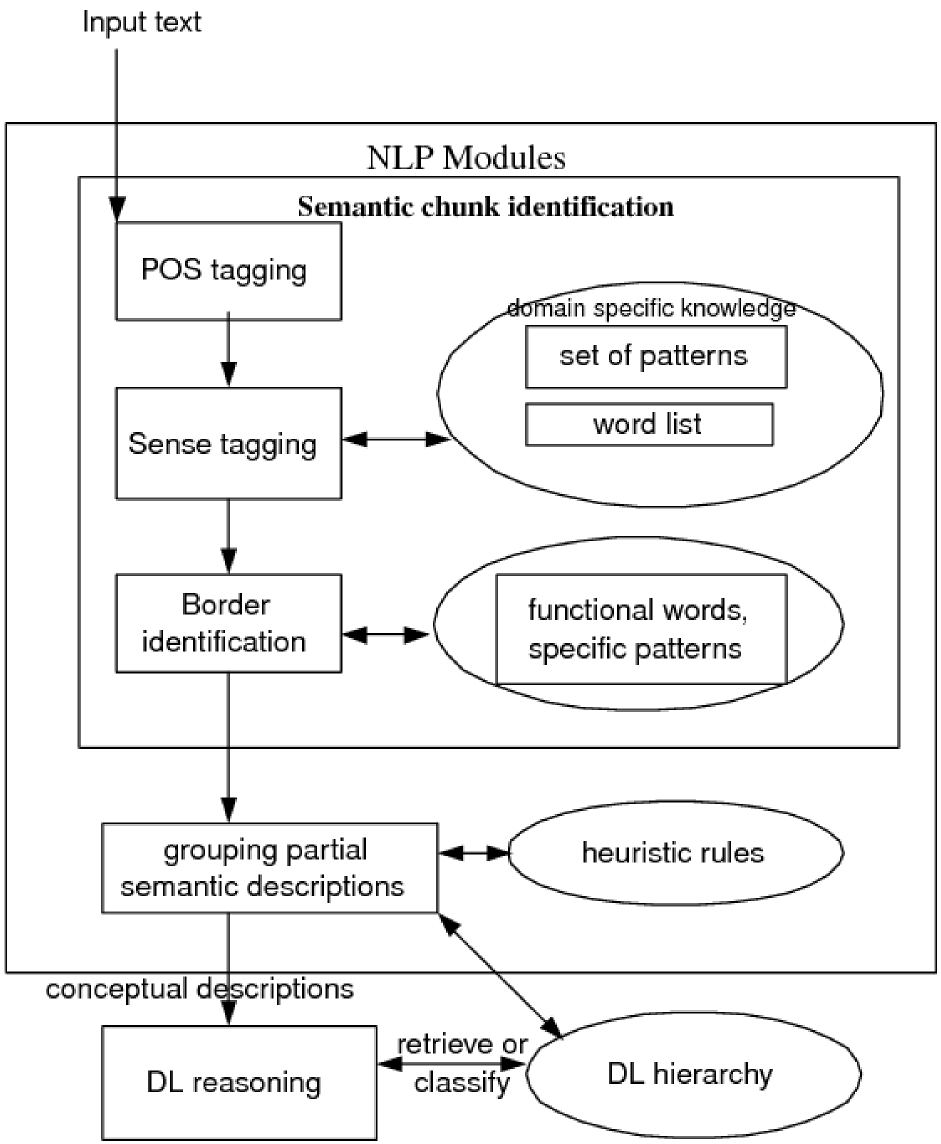
Fonctionnalités de LIKES

- Création de corpus
  - Sélection de un ou plusieurs textes dans la même langue.
  - textes en Format .txt ou Html ou XML (bientôt en word)
- Découpage:
  - le corpus est découpé en formes (tokens) en phrases et en paragraphes.
- La liste des formes différentes (types) peut être triée
  - alphabétiquement
  - **par fréquence décroissante**
- Recherche de motifs
- Retour au contexte.

Automates en LIKES

- LIKES par une interface graphique permet d'éditer des graphes d'automates.
- Tout ce qui a été vu peut être rentré sous forme d'automate.
- De façon transparente, chaque expression régulière est transformée en automate qui est ensuite déterminisé puis minimalisé.

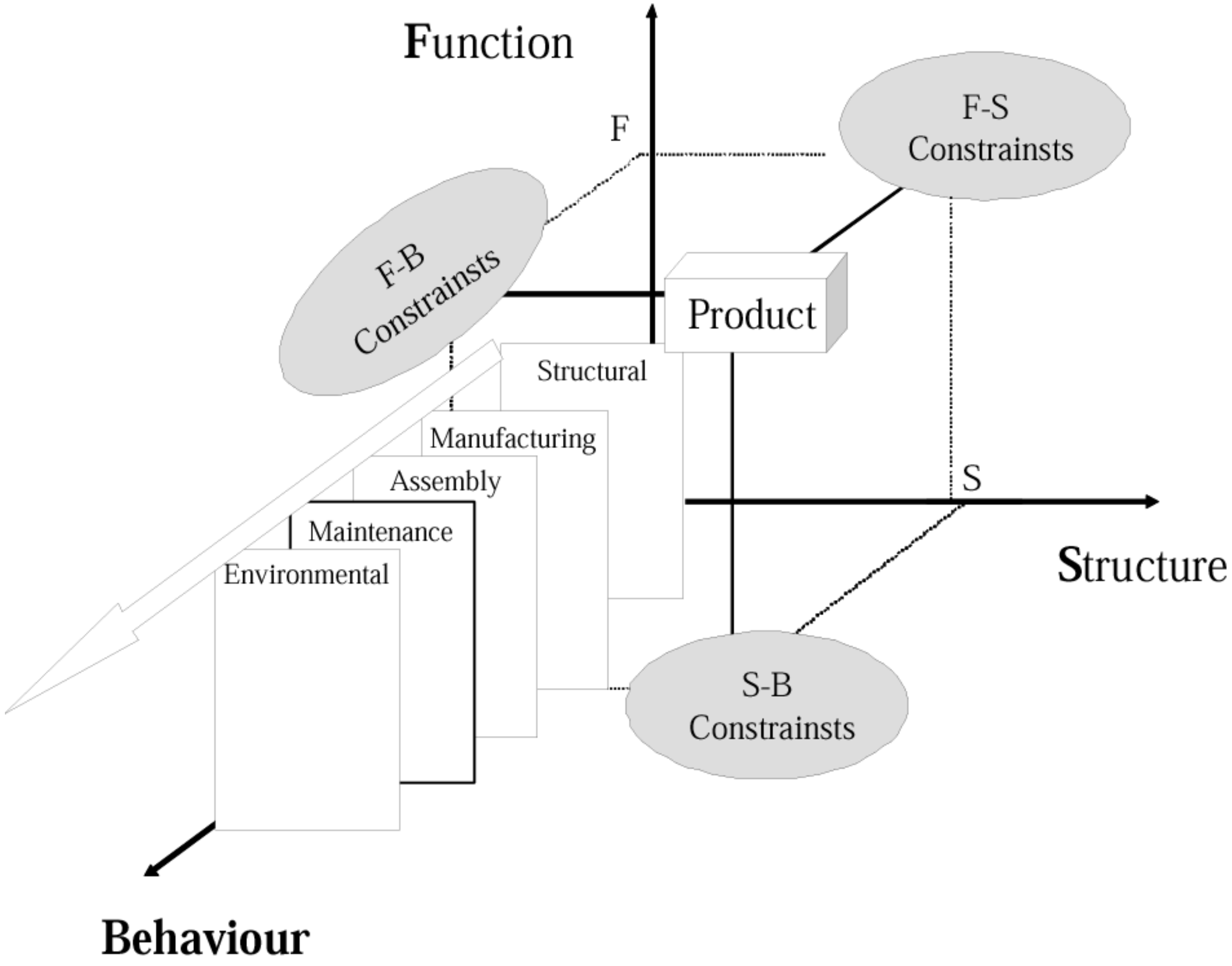
Semantic tagging



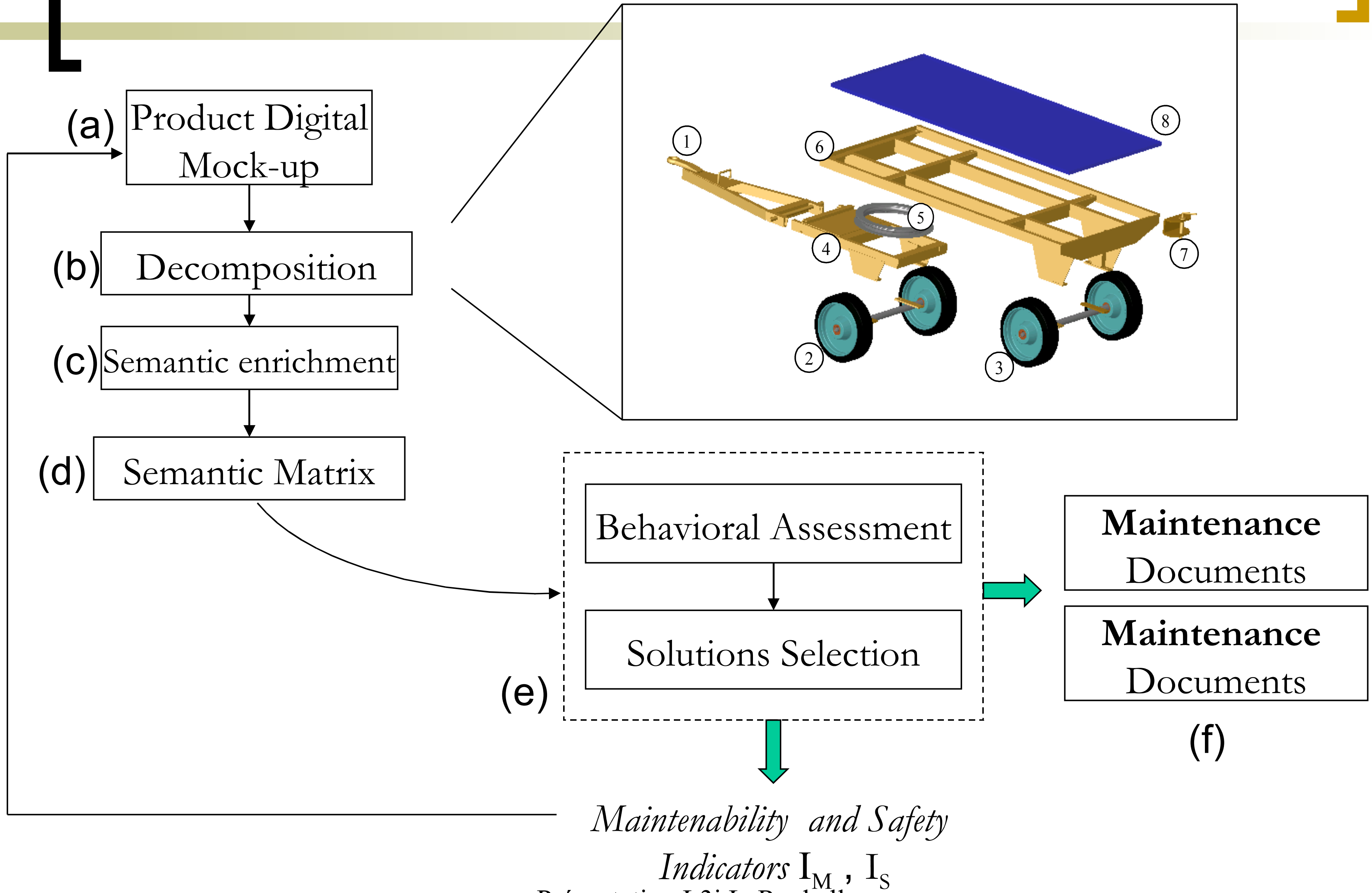
Applications

- Conceptual Modeling
- Query Optimization and View Maintenance
- Natural Language Semantics
- Terminologies and Ontology
- Information Access and Intelligent Interfaces
- Formal Specifications in Engineering
- Configuration
- Planning

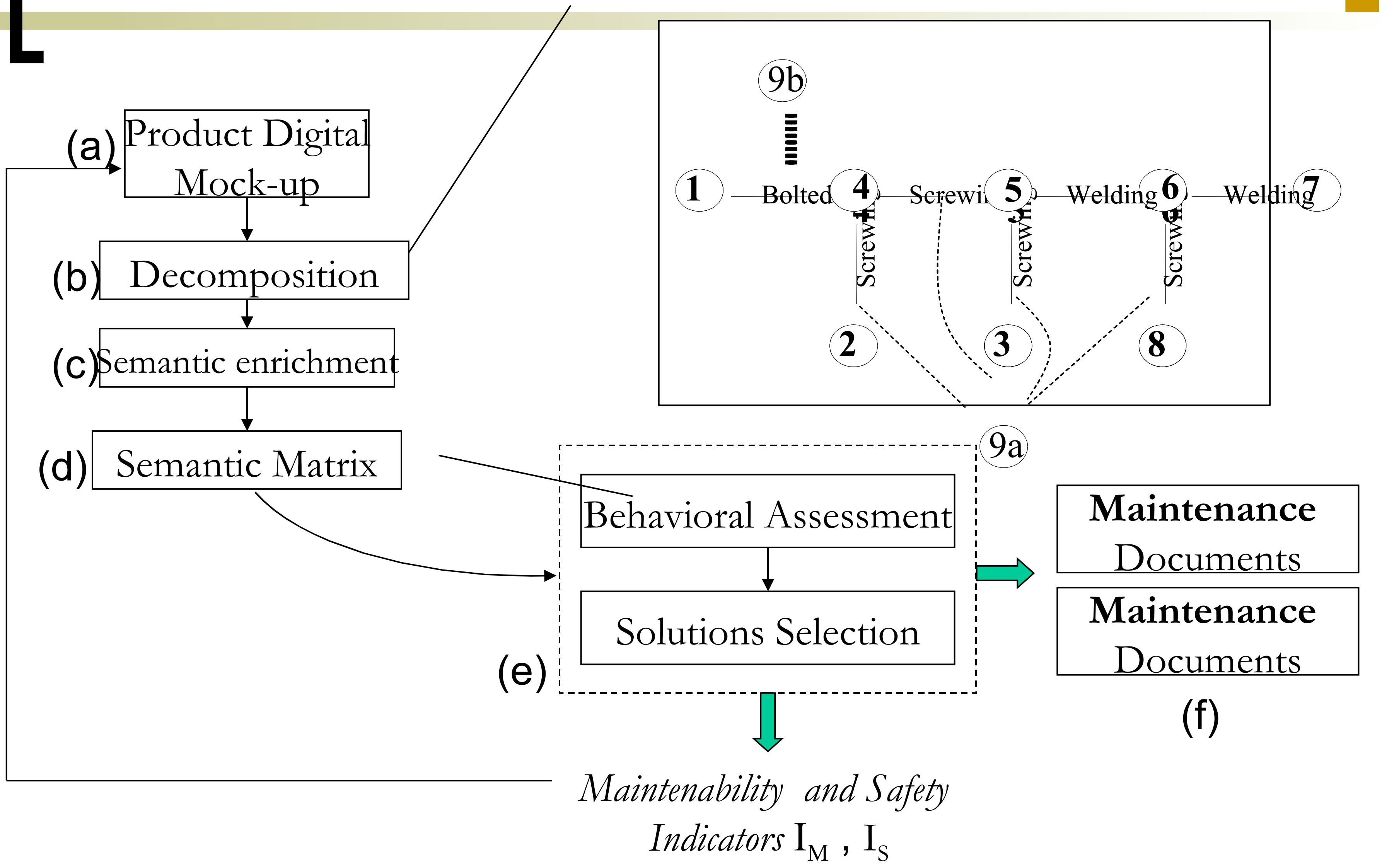
FSB framework



Mise en œuvre de l'Evaluation Comportementale



Mise en œuvre de l'Evaluation Comportementale



(a) Product Digital Mock-up

(b) Decomposition

(c) Semantic enrichment

(d) Semantic Matrix

Behavioral Assessment

Solutions Selection

(e)

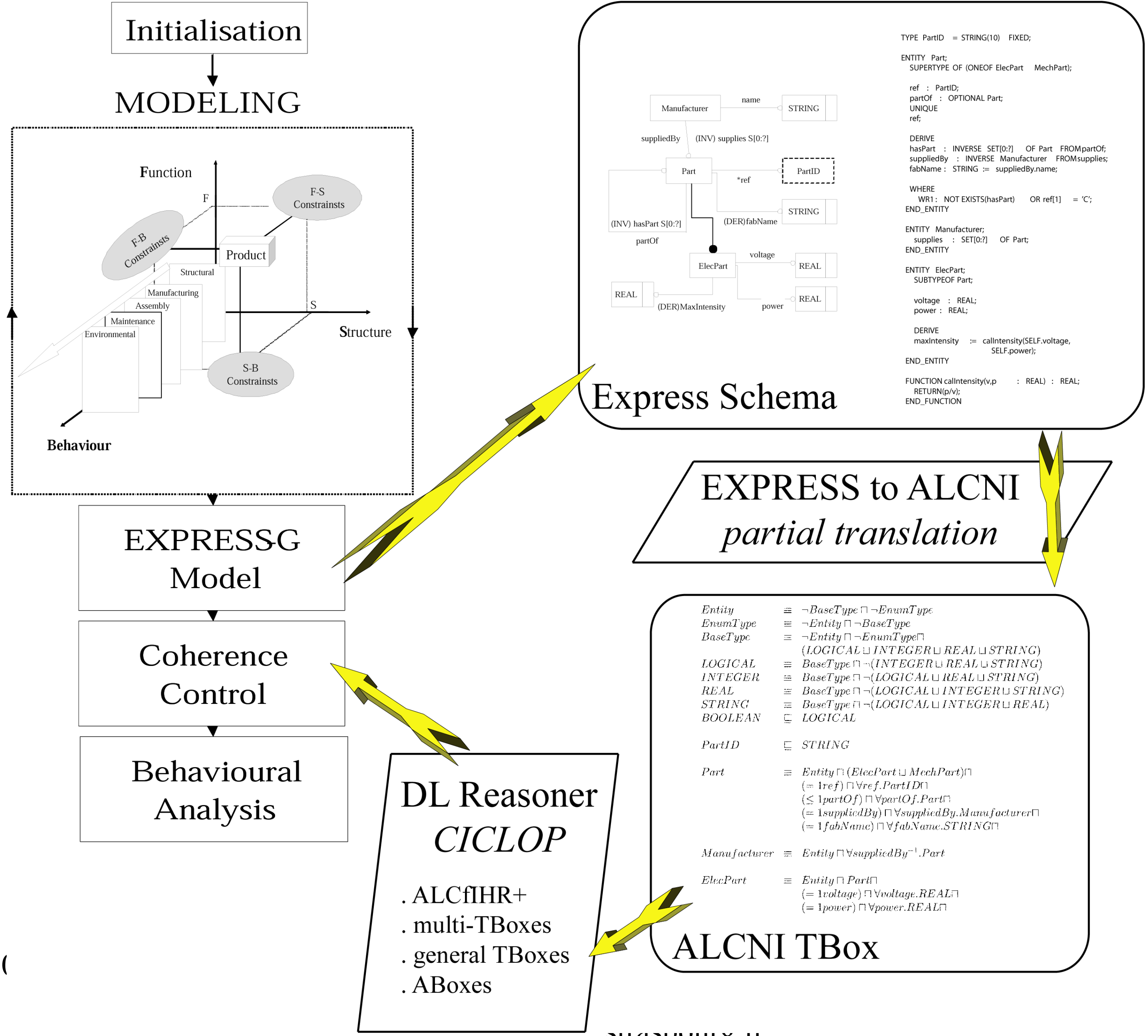
Maintenance Documents

Maintenance Documents

(f)

Maintenability and Safety Indicators I_M , I_S

Embedding DL within CAD Systems



Initialisation

MODELING

EXPRESS-G Model

Coherence Control

Behavioural Analysis

```
TYPE  PartID   = STRING(10)   FIXED;

ENTITY  Part;
   SUPERTYPE OF (ONEOF ElecPart    MechPart);

   ref   :  PartID;
   partOf   :  OPTIONAL Part;
   UNIQUE
   ref;

   DERIVE
   hasPart   :  INVERSE  SET[0:?]    OF Part   FROMpartOf;
   suppliedBy   :  INVERSE  Manufacturer   FROMsupplies;
   fabName :  STRING  :=  suppliedBy.name;

   WHERE
      WR1 :  NOT EXISTS(hasPart)      OR ref[1]    = 'C';
 END_ENTITY

 ENTITY  Manufacturer;
    supplies    :  SET[0:?]    OF Part;
 END_ENTITY

 ENTITY  ElecPart;
    SUBTYPEOF Part;

    voltage   :  REAL;
    power :  REAL;

    DERIVE
    maxIntensity    :=  callntensity(SELF.voltage,
                              SELF.power);
 END_ENTITY

 FUNCTION callntensity(v,p        :  REAL)  :  REAL;
    RETURN(p/v);
 END_FUNCTION
```

Express Schema

EXPRESS to ALCNI

partial translation

$$
\begin{array}{lll}
Entity & \equiv & \neg BaseType \sqcap \neg EnumType \\
EnumType & \equiv & \neg Entity \sqcap \neg BaseType \\
BaseType & \equiv & \neg Entity \sqcap \neg EnumType \sqcap \\
 & & (LOGICAL \sqcup INTEGER \sqcup REAL \sqcup STRING) \\
LOGICAL & \equiv & BaseType \sqcap \neg(INTEGER \sqcup REAL \sqcup STRING) \\
INTEGER & \equiv & BaseType \sqcap \neg(LOGICAL \sqcup REAL \sqcup STRING) \\
REAL & \equiv & BaseType \sqcap \neg(LOGICAL \sqcup INTEGER \sqcup STRING) \\
STRING & \equiv & BaseType \sqcap \neg(LOGICAL \sqcup INTEGER \sqcup REAL) \\
BOOLEAN & \sqsubseteq & LOGICAL \\
 & & \\
PartID & \sqsubseteq & STRING \\
 & & \\
Part & \equiv & Entity \sqcap (ElecPart \sqcup MechPart) \sqcap \\
 & & (= 1 ref) \sqcap \forall ref.PartID \sqcap \\
 & & (\leq 1 partOf) \sqcap \forall partOf.Part \sqcap \\
 & & (= 1 suppliedBy) \sqcap \forall suppliedBy.Manufacturer \sqcap \\
 & & (= 1 fabName) \sqcap \forall fabName.STRING \sqcap \\
 & & \\
Manufacturer & \equiv & Entity \sqcap \forall suppliedBy^{-1}.Part \\
 & & \\
ElecPart & \equiv & Entity \sqcap Part \sqcap \\
 & & (= 1 voltage) \sqcap \forall voltage.REAL \sqcap \\
 & & (= 1 power) \sqcap \forall power.REAL \sqcap
\end{array}
$$

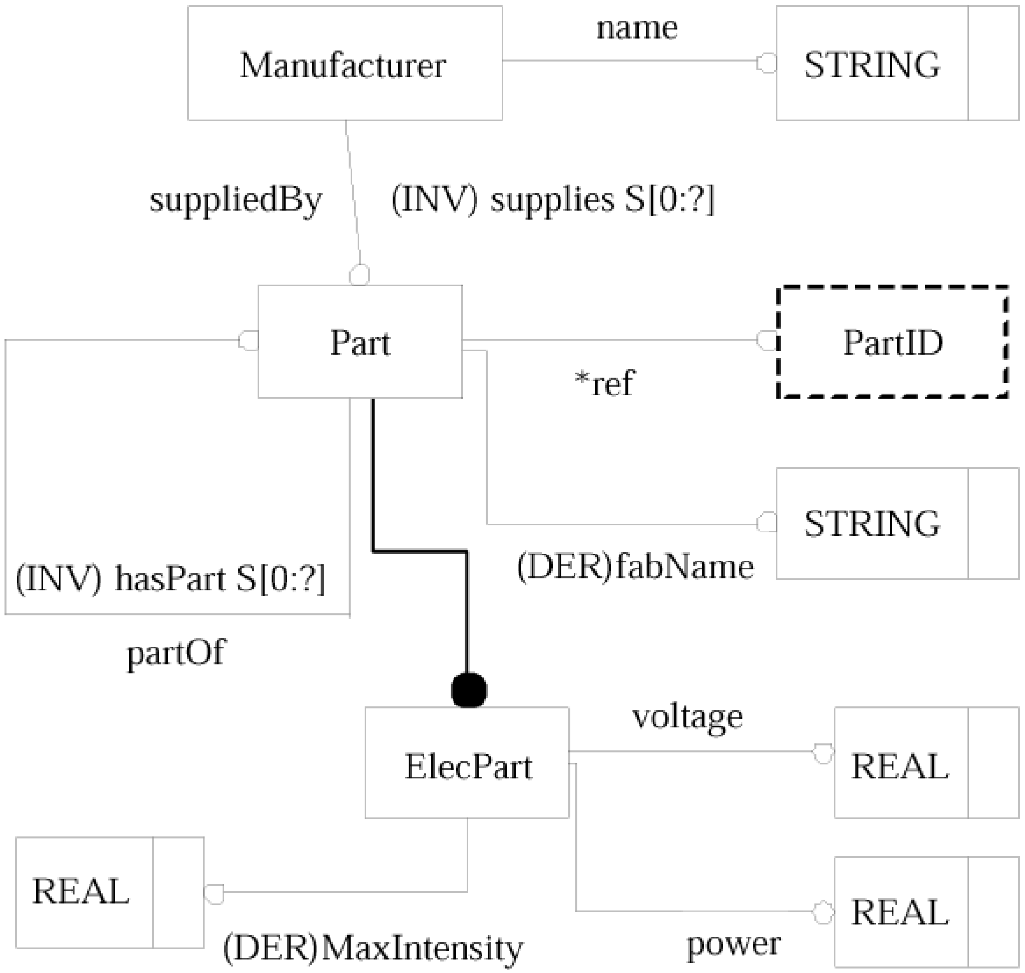
ALCNI TBox

DL Reasoner

CICLOP

. ALCfIHR+
. multi-TBoxes
. general TBoxes
. ABoxes

Express-G model (STEP)



ALCNI translation of Express-G

$$
\begin{array}{lcl}
Entity & \equiv & \neg BaseType \sqcap \neg EnumType \\
EnumType & \equiv & \neg Entity \sqcap \neg BaseType \\
BaseType & \equiv & \neg Entity \sqcap \neg EnumType \sqcap \\
 & & (LOGICAL \sqcup INTEGER \sqcup REAL \sqcup STRING) \\
LOGICAL & \equiv & BaseType \sqcap \neg (INTEGER \sqcup REAL \sqcup STRING) \\
INTEGER & \equiv & BaseType \sqcap \neg (LOGICAL \sqcup REAL \sqcup STRING) \\
REAL & \equiv & BaseType \sqcap \neg (LOGICAL \sqcup INTEGER \sqcup STRING) \\
STRING & \equiv & BaseType \sqcap \neg (LOGICAL \sqcup INTEGER \sqcup REAL) \\
BOOLEAN & \sqsubseteq & LOGICAL \\
 & & \\
PartID & \sqsubseteq & STRING \\
 & & \\
Part & \equiv & Entity \sqcap (ElecPart \sqcup MechPart) \sqcap \\
 & & (= 1 ref) \sqcap \forall ref.PartID \sqcap \\
 & & (\leq 1 partOf) \sqcap \forall partOf.Part \sqcap \\
 & & (= 1 suppliedBy) \sqcap \forall suppliedBy.Manufacturer \sqcap \\
 & & (= 1 fabName) \sqcap \forall fabName.STRING \sqcap \\
 & & \\
Manufacturer & \equiv & Entity \sqcap \forall suppliedBy^{-1}.Part \\
 & & \\
ElecPart & \equiv & Entity \sqcap Part \sqcap \\
 & & (= 1 voltage) \sqcap \forall voltage.REAL \sqcap \\
 & & (= 1 power) \sqcap \forall power.REAL \sqcap
\end{array}
$$

DL for CAD (requirements)

- DL KB for a flexible « feature » system within CAD
- Incremental definition of components
- Automatic classification of components
- Semantic based indexation of component library
- Inconsistency detection

Applications

- Conceptual Modeling
- Query Optimization and View Maintenance
- Natural Language Semantics
- Terminologies and Ontology
- Information Access and Intelligent Interfaces
- Formal Specifications in Engineering
- Configuration
- Planning

a KB for configuration [Schlick]

- Open/closed world reasoning
- Two kind of concepts
  - abstract one
  - concrete one
- abstract concepts express choices:
  - of functionalities, of esthetical style etc.
  - General choices
- concrete concepts represent real objects

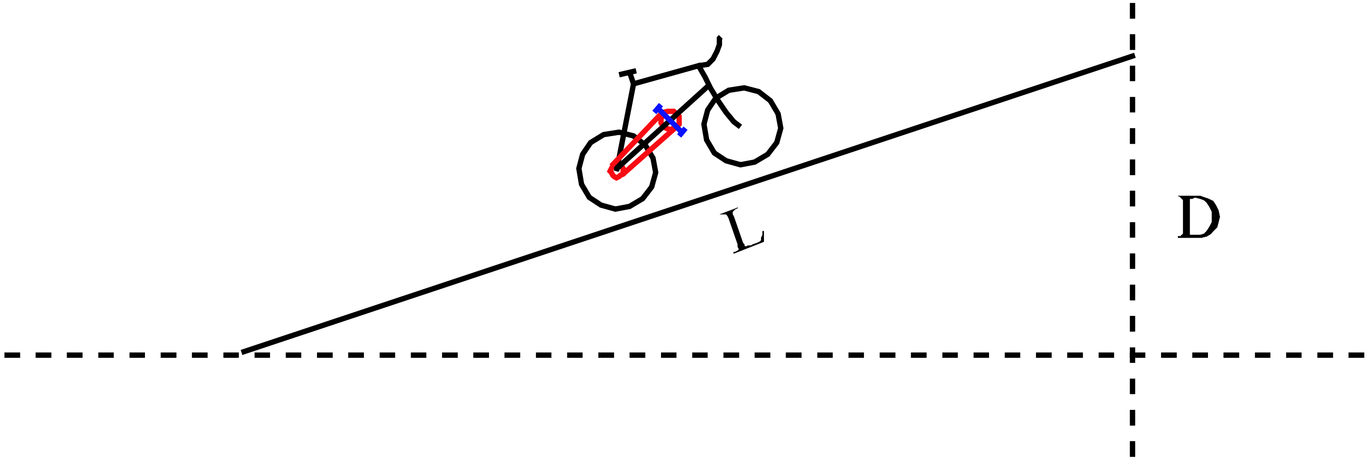
Enhancements of Description Logics for Configuration

- A concept is concrete iff an individual which is its instance is both complete and sufficiently specific for the purpose of the intended application.
- Vocabulary closure
- Domain closure
- Syntactic closure of the terminology
- Forcing values

Applications

- Conceptual Modeling
- Query Optimization and View Maintenance
- Natural Language Semantics
- Terminologies and Ontology
- Information Access and Intelligent Interfaces
- Formal Specifications in Engineering
- Configuration
- Planning
- Intermède cyclo-touristique

Parcours



Ciclop parcours

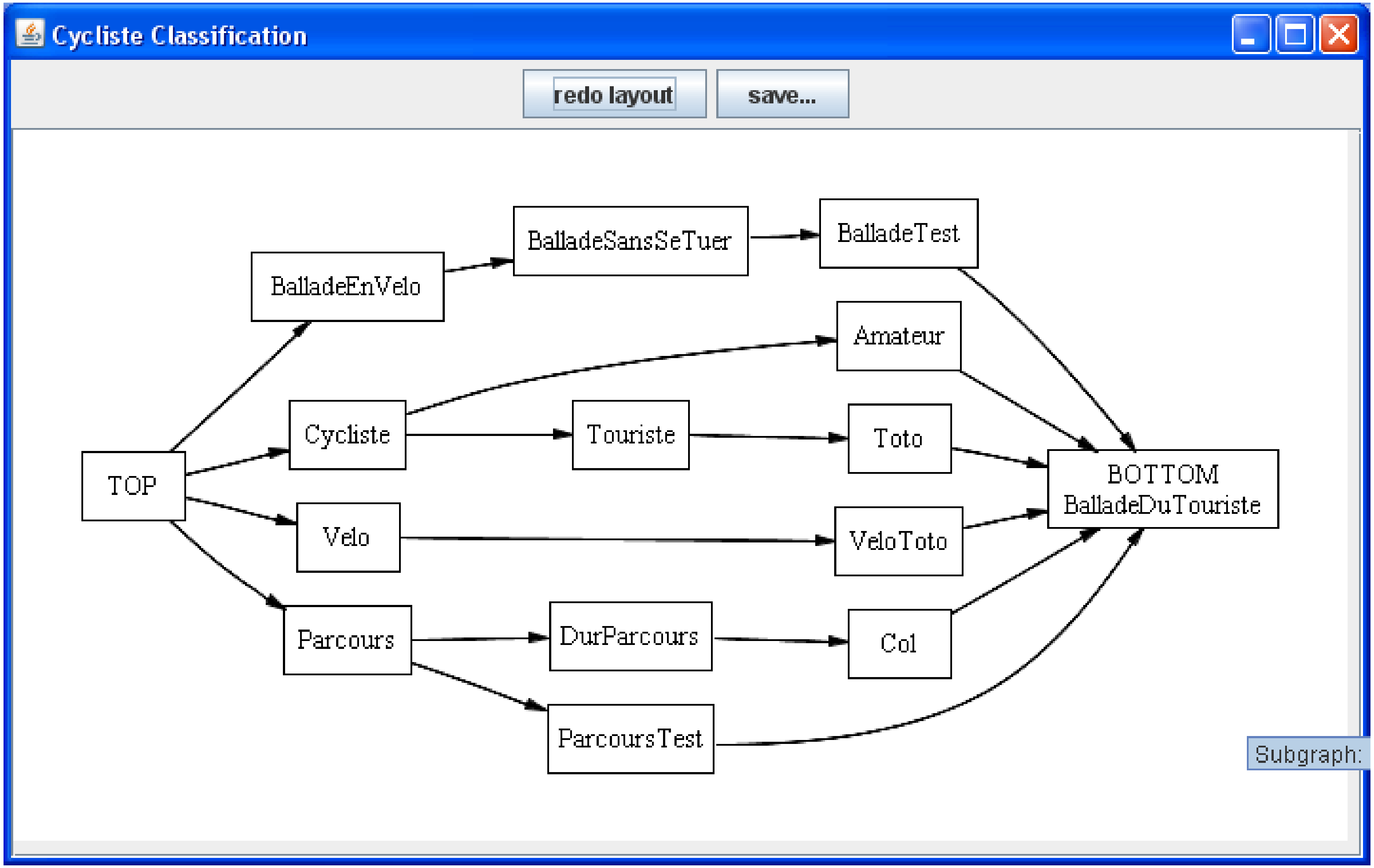
-
  ```
  (defconcept Parcours
    (and
     (not Velo)
      (CSome (path denivele)=X "True")
      (CSome (path longueur)=X "X >= 0")))
  ```

-
  ```
  (defconcept Col
    (and Parcours
     (CAll (path denivele)=D , (path longueur)=L
          "D > L * 10/100")))
  ```

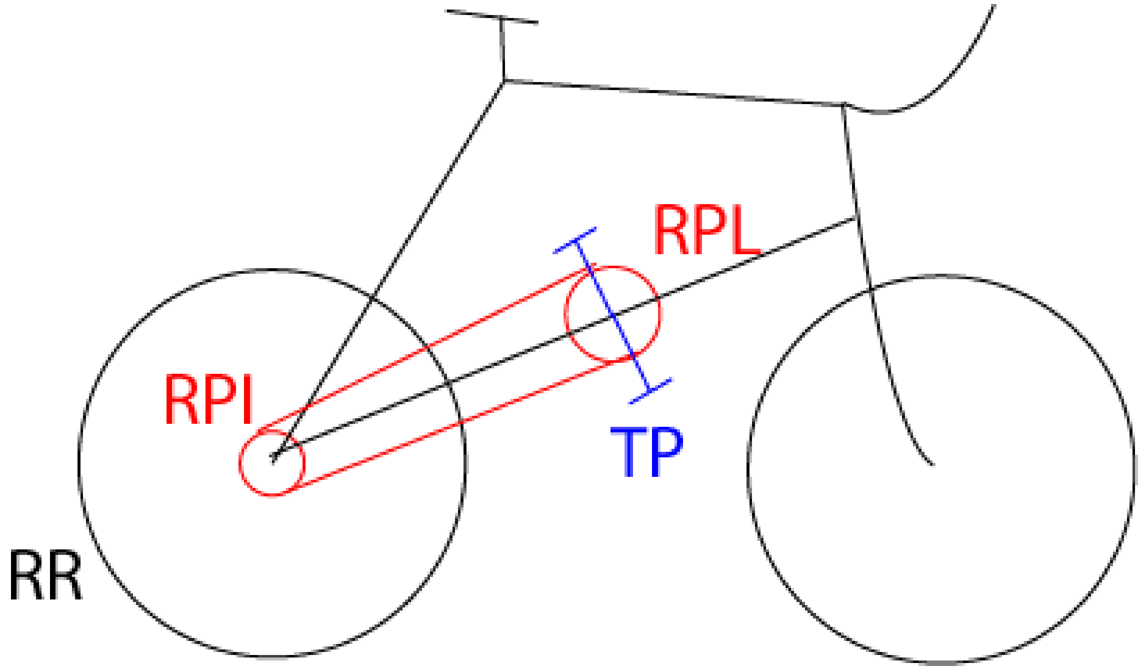
Ballade

```
(defconcept Ballade
 (and
  (not Velo) (not Parcours) (not Cycliste)
  (Some qui Cycliste)
  (Some ou Parcours)
  (Some sur Velo)
  (CSome (path duree)=X "X >= 0")
  (CSome (path qui poids)=PC ,
         (path sur poids)=PV ,
         (path ou denivele)=D ,
         (path energieRequise)=E
         "E == 981/100  (PC + PV)  D")))
```

Ballades classifiées



Vélo



(path energieRequise)=E "2 Pi RR RPL E == L RPI TP F")

vélo

```
(defConcept Velo
  (and
    (CSome (path rayonPlateau)=X "X >= 0")
    (CSome (path rayonPignon)=X "X >= 0")
    (CSome (path rayonRoue)=X "X >= 0")
    (CSome (path taillePedale)=X "X >= 0")
    (CSome (path poids)=X "X >= 0")))
```

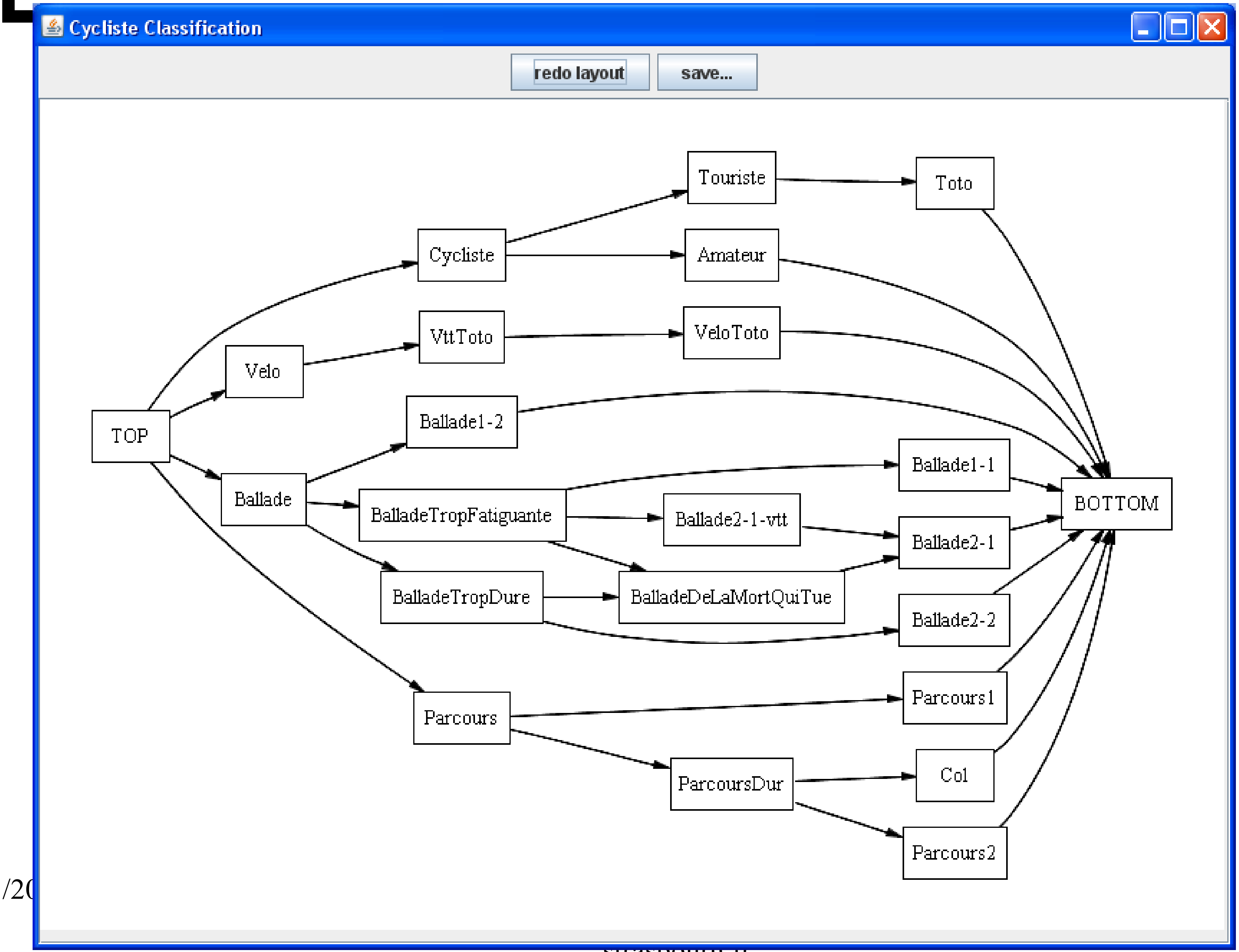
Un type de vélo plus particulier

-
```
(defconcept VeloToto
  (and Velo
    (CSome (path rayonPlateau)=X "X == 10/100")
    (CSome (path rayonPignon)=X "X == 5/100")
    (CSome (path rayonRoue)=X "X == 40/100")
    (CSome (path taillePedale)=X "X ==20/100")
    (CSome (path poids)=X "X == 15")))
```

Un autre vélo

```
(defconcept VttToto
  (and Velo
    (CSome (path rayonPlateau)=X "X == 10/100")
    (or
      (CSome (path rayonPignon)=X "X == 5/100")
      (CSome (path rayonPignon)=X "X == 30/100")
    )
    (CSome (path rayonRoue)=X "X == 40/100")
    (CSome (path taillePedale)=X "X ==20/100")
    (CSome (path poids)=X "X == 15")))
```

Ballades re-classifiées



Applications

- Conceptual Modeling
- Query Optimization and View Maintenance
- Natural Language Semantics
- Terminologies and Ontology
- Information Access and Intelligent Interfaces
- Formal Specifications in Engineering
- Configuration
- Planning
- Conception innovante et ontologies

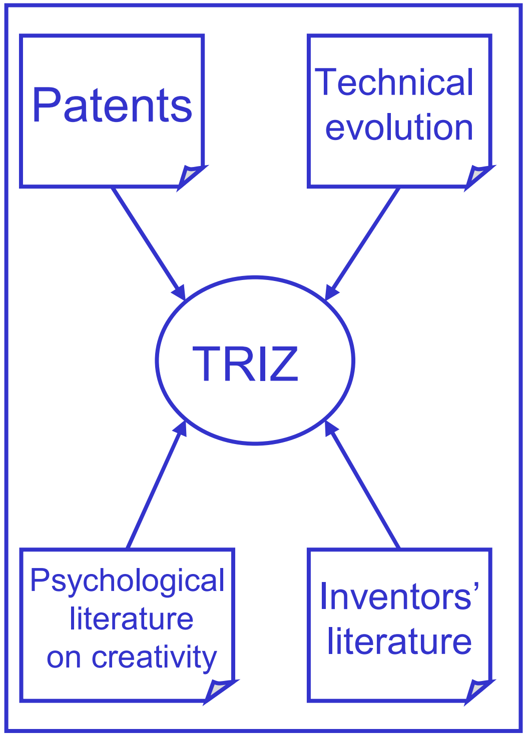
DL-TRIZ (Alexis Bultey)

A problem solving environment based on TRIZ ontologies

- TRIZ ?
- The Substance-Field Analysis
- System Architecture
- Preliminary results and perspectives

What is TRIZ

- Theory of Inventive problem solving introduced by Altshuller (1946)



- What is TRIZ ?

TRIZ is based on :

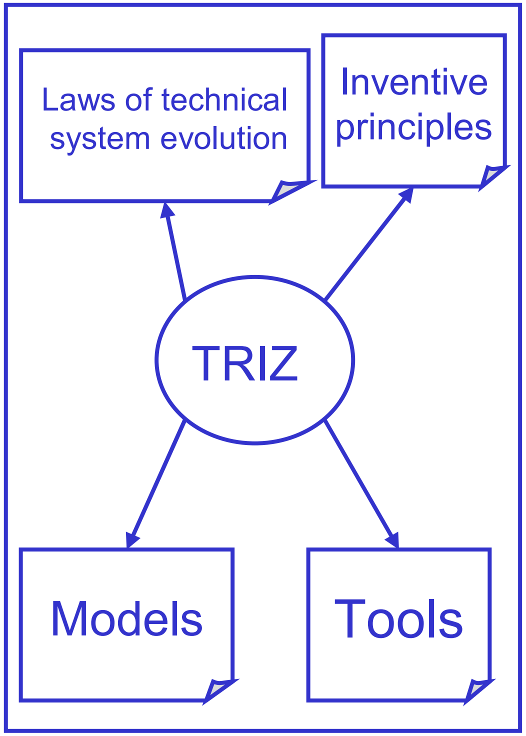
❖ 200,000 patents analysis,

❖ technical evolution literature,

❖ inventors biography,

❖ and creative methods from psychological movement.

What is inside TRIZ

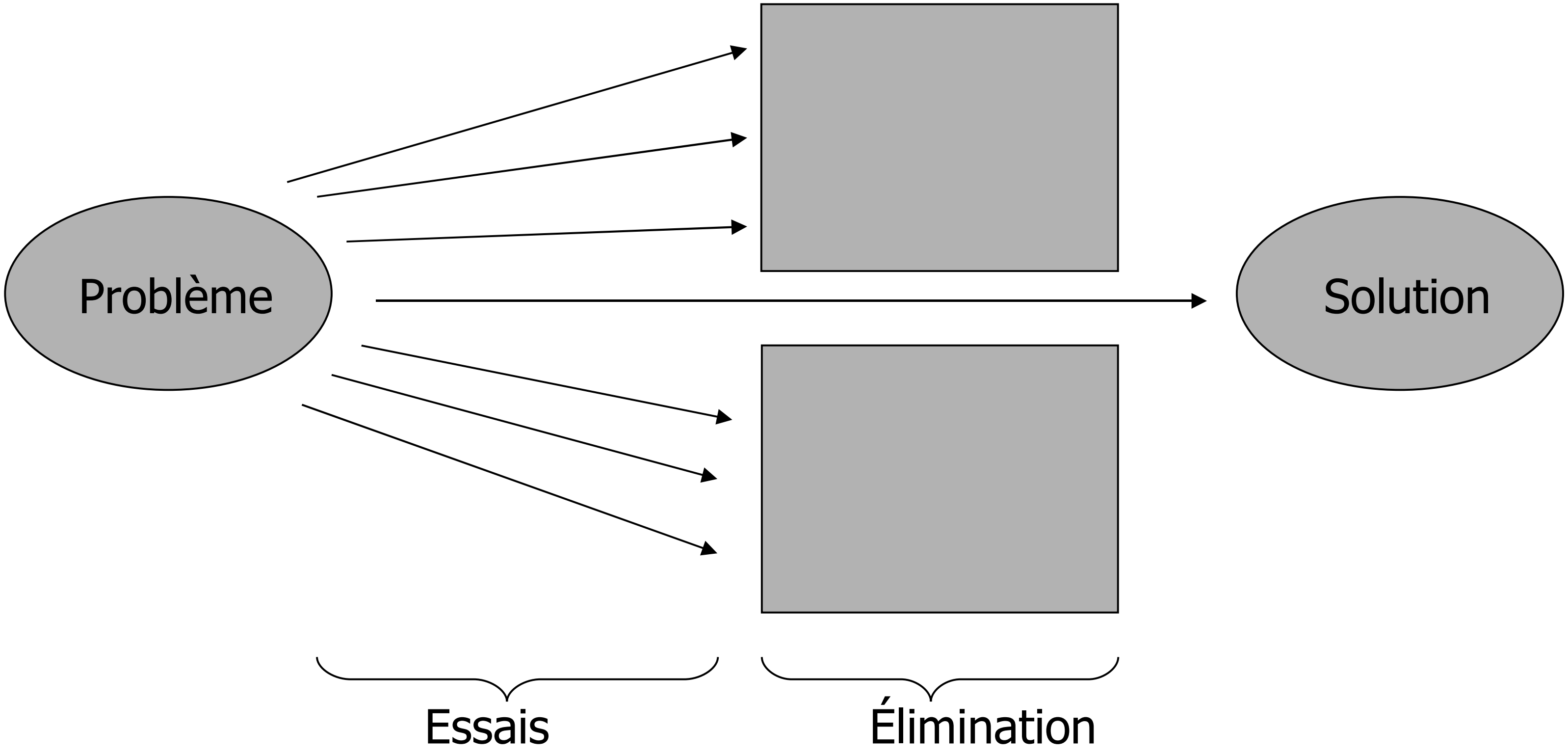


TRIZ includes:

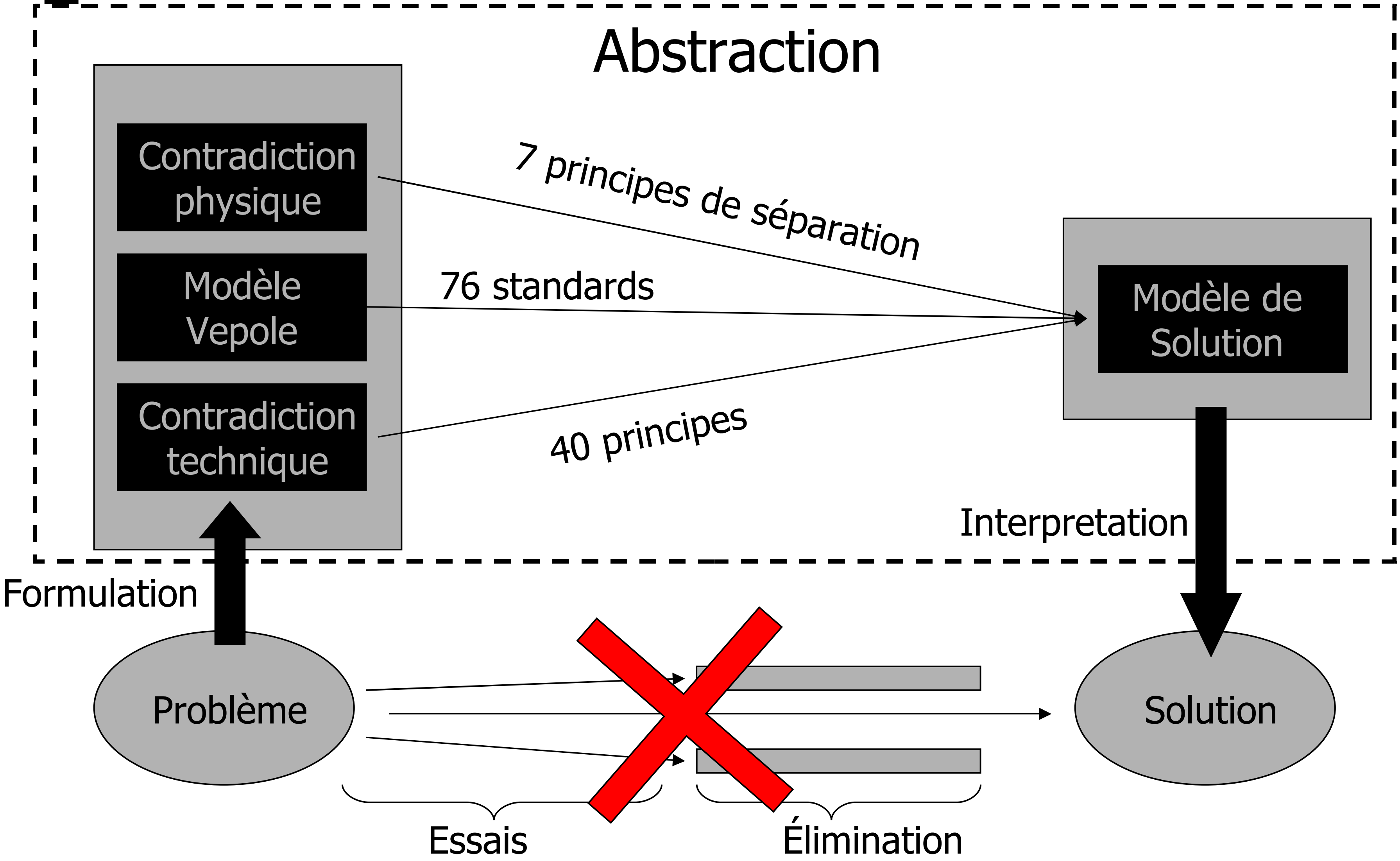
- some laws of technical system evolution
- some inventive principles,
- some general models for problem formulation and solving,
- and some methods, tools to help and assist the solving process

Ce que TRIZ n'est pas

- Méthode par essai-validation (brain storming)



Le principe TRIZ



DL-TRIZ (Alexis Bultey)

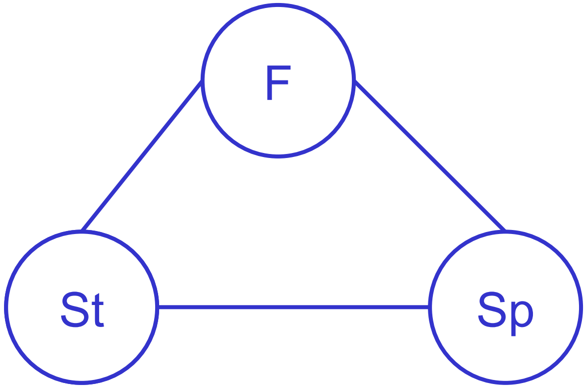
A problem solving environment based on TRIZ ontologies

- TRIZ ?
- The Substance-Field Analysis
- System Architecture
- Preliminary results and perspectives

Substance-Field Analysis

- A minimal technical system can be modeled by a complete S-Field.
- A complete S-Field contains at least a tool substance (St), a product substance (Sp) and a Field (F).

S-Field modeling



A complete S-Field model

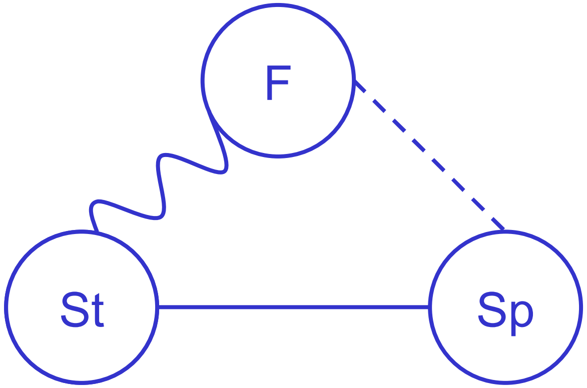
= minimal technical system

Substance-Field Analysis

The “mutual interaction” are the universal form of connection between the elements of a S-Field.

The mutual interaction could be qualified as inefficient, harmful or useful.

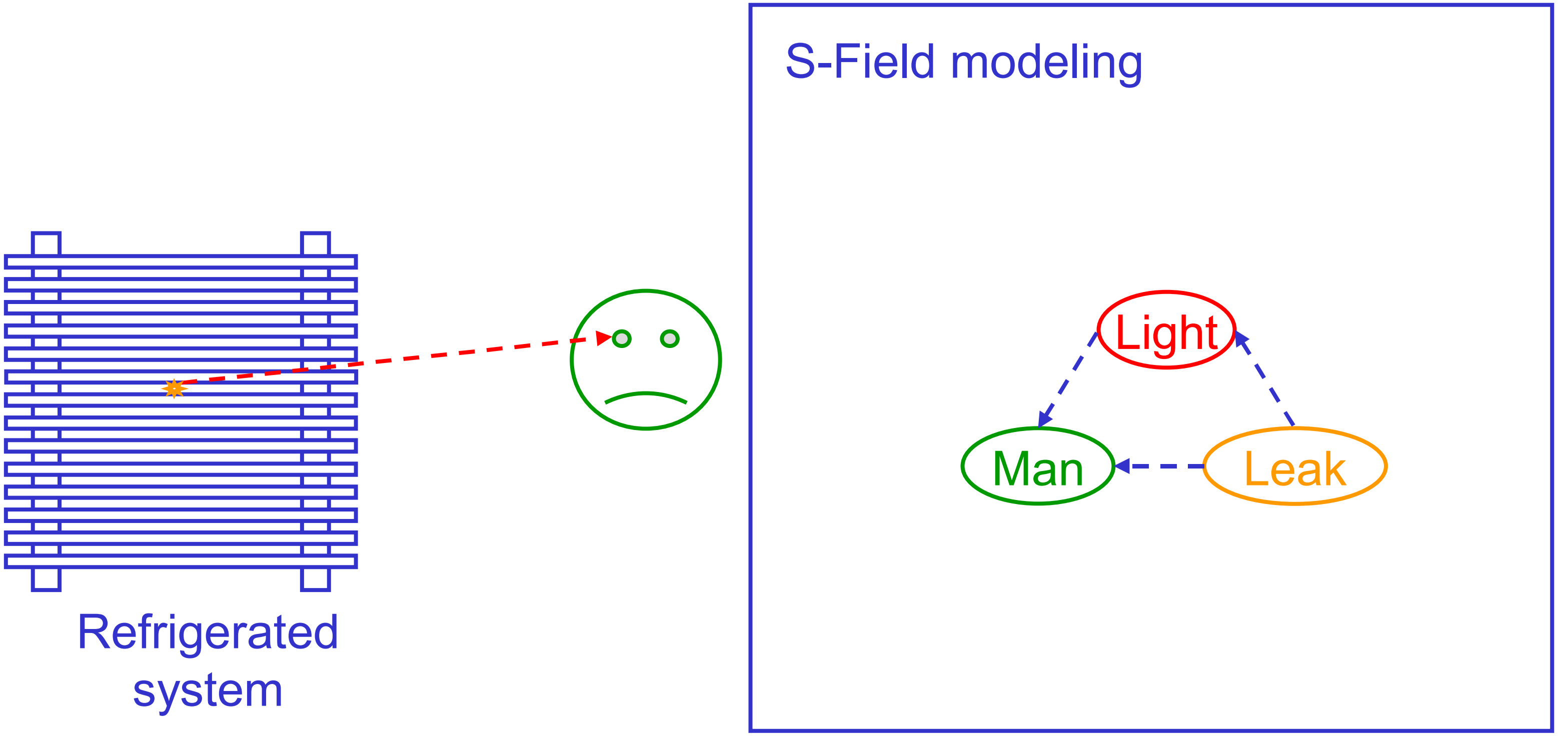
S-Field modeling



A complete S-Field model

= minimal technical system

SFA (exemple)



Refrigerated system

S-Field modeling

Light

Man

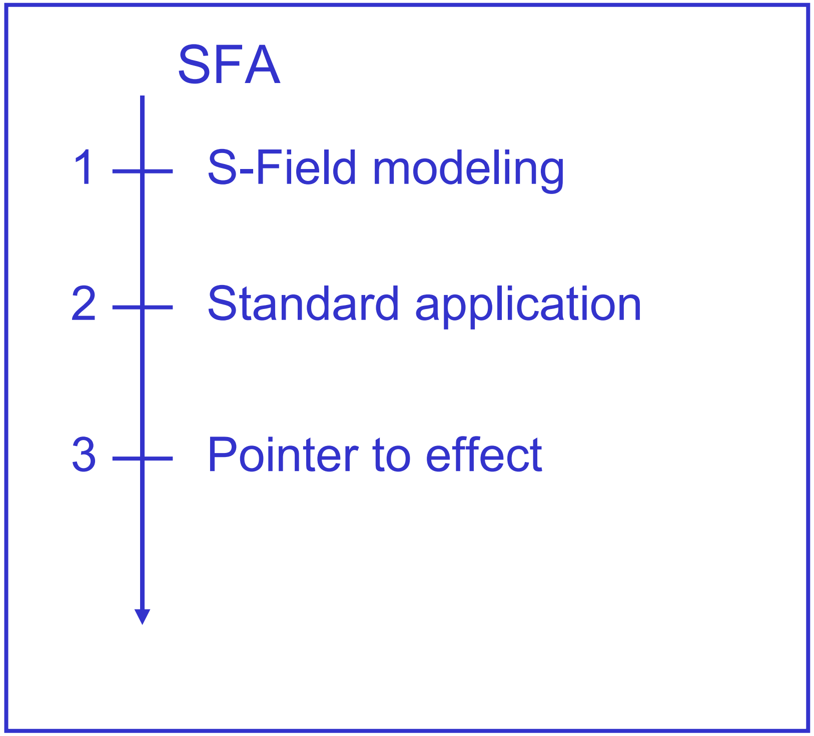
Leak

SFA problem solving

The SFA basis :

The SFA is subdivided in 3 phases:

3. a problem formulation phase
4. a resolution phase
5. an interpretation phase


SFA

1 S-Field modeling

2 Standard application

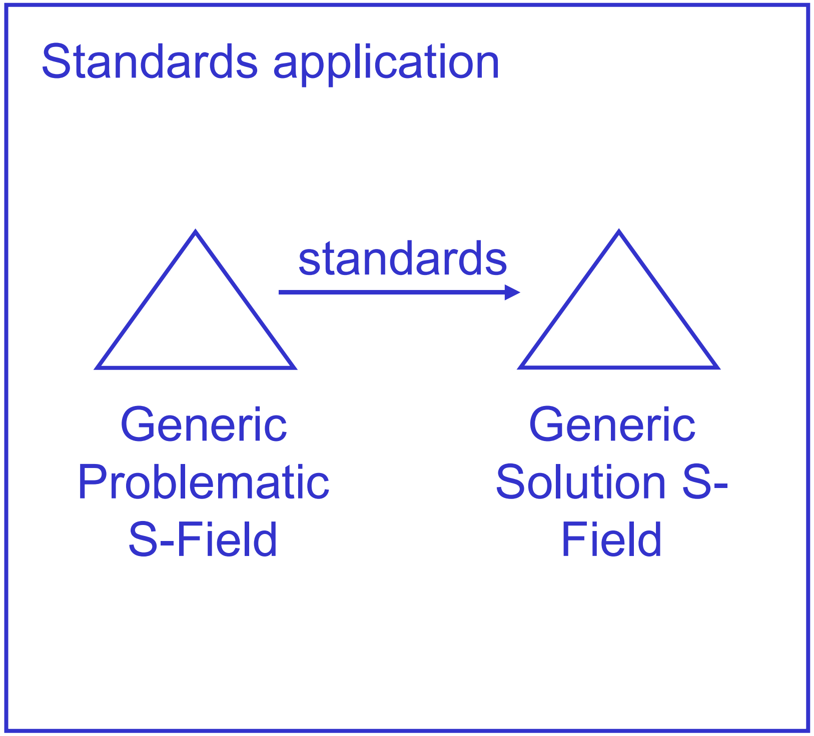
3 Pointer to effect

SFA resolution

The resolution phase:

It links a generic problematic S-Field to a generic solution S-Field, by means of standards.

There is 76 standards which represent some generic way of resolution used by former inventors.



Standards application

standards

Generic Problematic S-Field

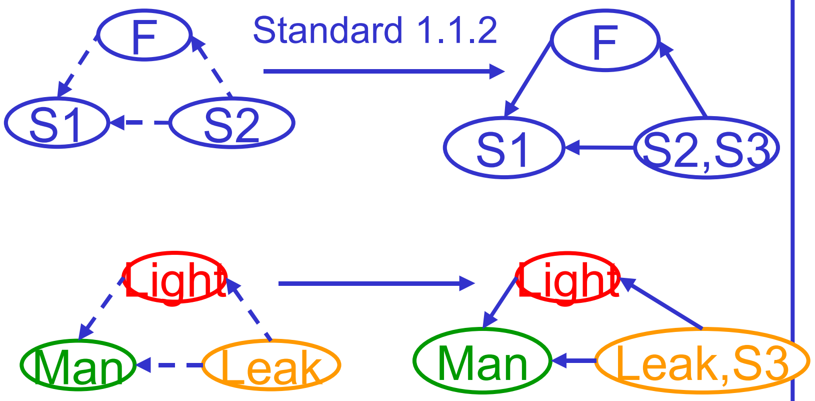
Generic Solution S-Field

SFA resolution (exemple)

The resolution phase on an example:

Standard 1.1.2 : **if** there is a S-Field which is not easy to change and there is no restriction on the introduction of an additive to given substance, **then** the problem is to be solved by a transition to an internal complex S-Field, introducing additive in the substance S1 or S2 enhancing controllability or imparting the required properties of the S-Field.

Standards application

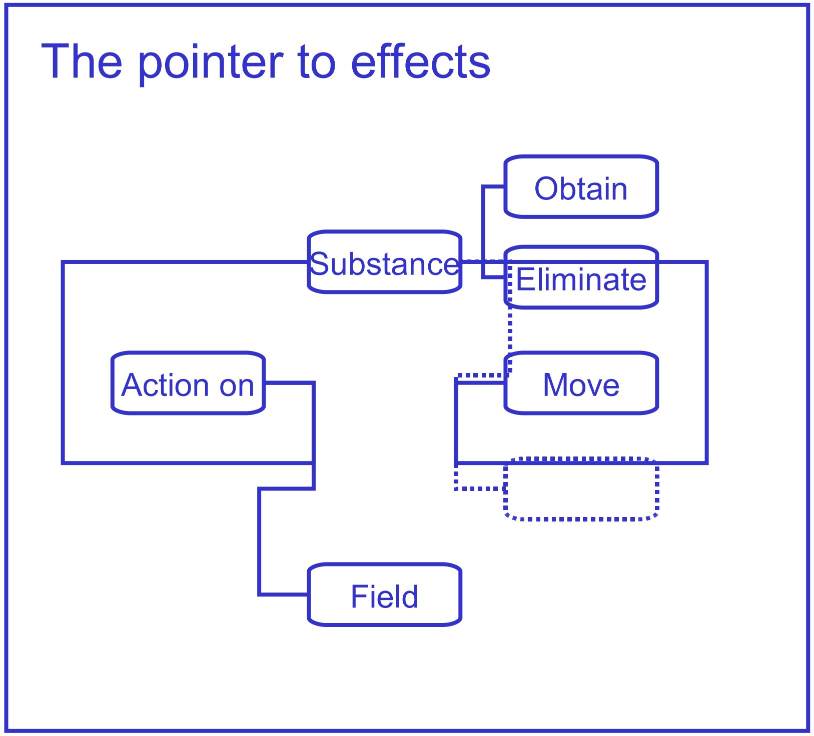


SFA interpretation

The interpretation phase:

The pointer to effects enables to instantiate the solution S-Field with some physical, chemical or geometric effect.

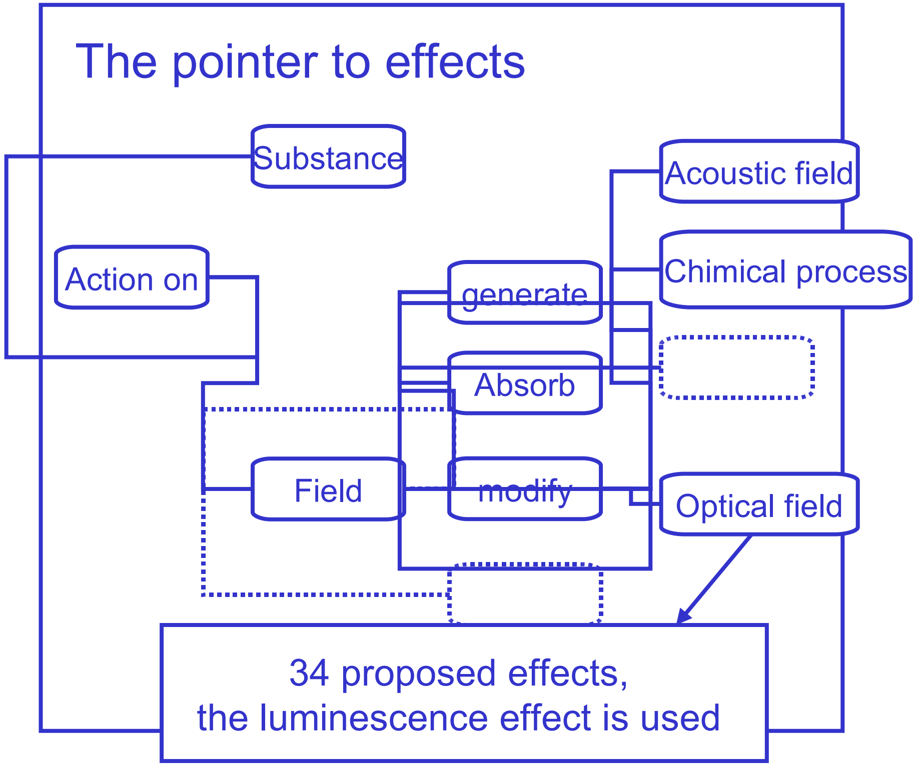
This pointer to effects is classified with a functional index.


The pointer to effects

SFA interpretation (exemple)

The interpretation phase:

In TRIZ literature according to the problem specification, the additive used is a luminescent substance.

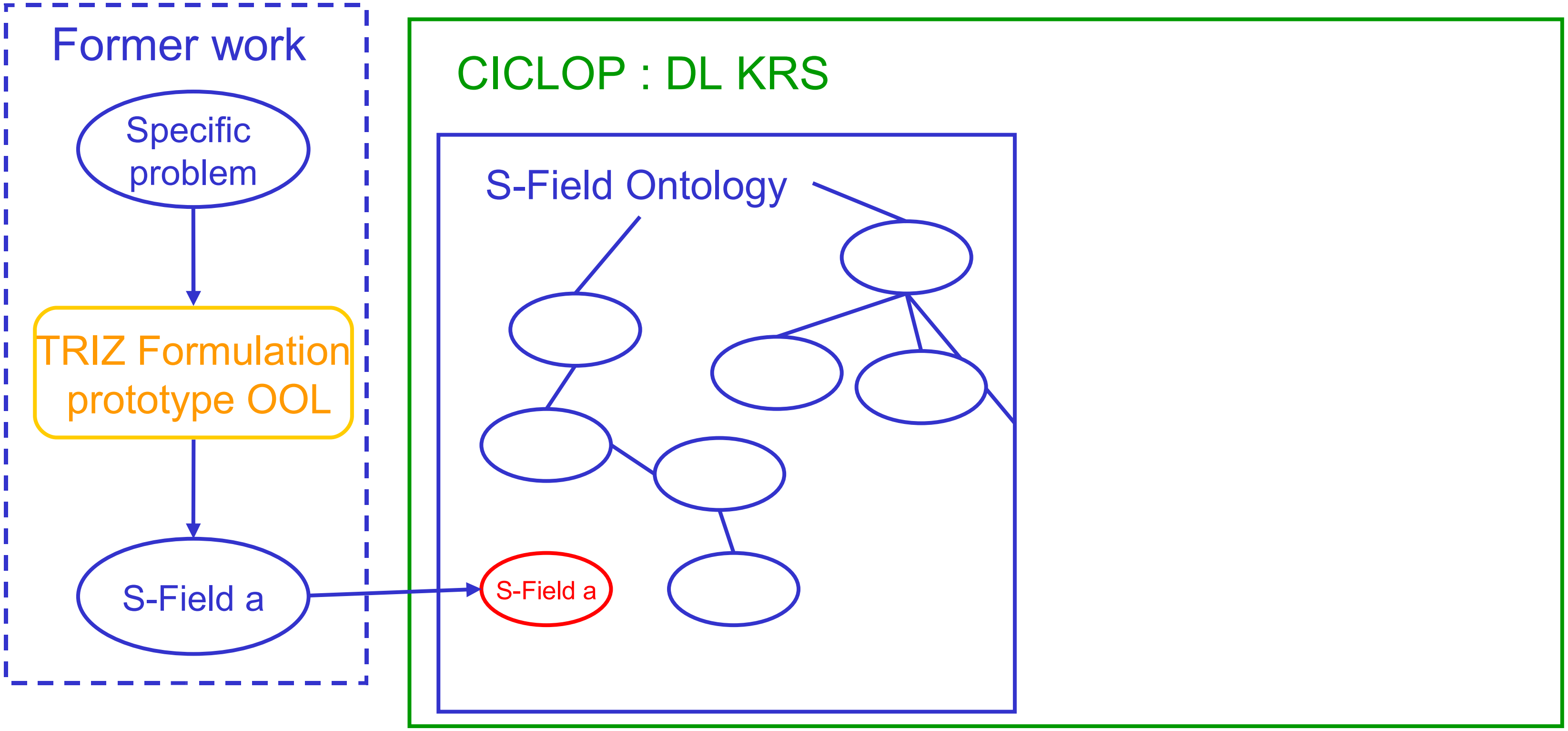


DL-TRIZ (Alexis Bultey)

A problem solving environment based on TRIZ ontologies

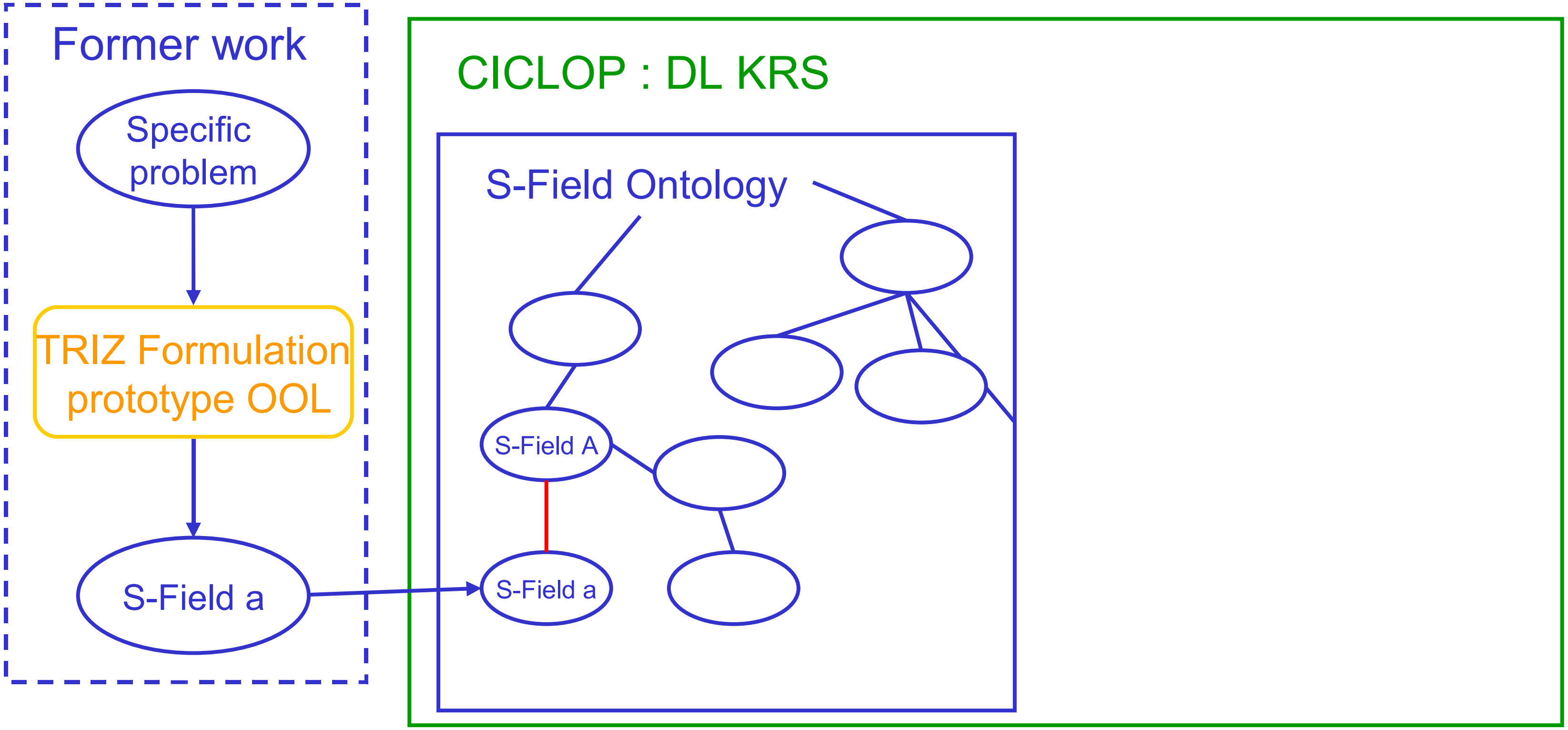
- TRIZ ?
- The Substance-Field Analysis
- System Architecture
- Preliminary results and perspectives

Project achitecture



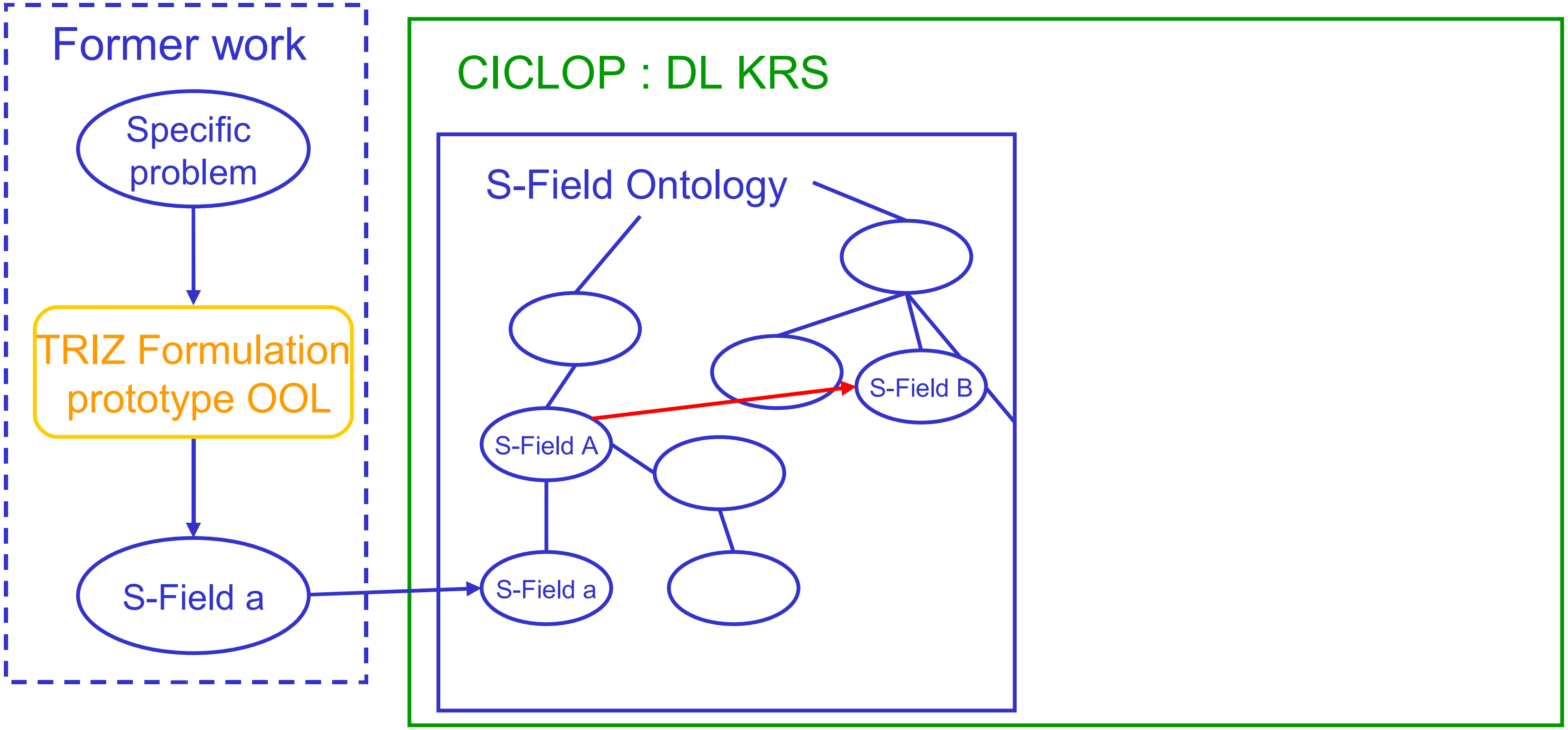
DL definition of S-Field a

Project achitecture



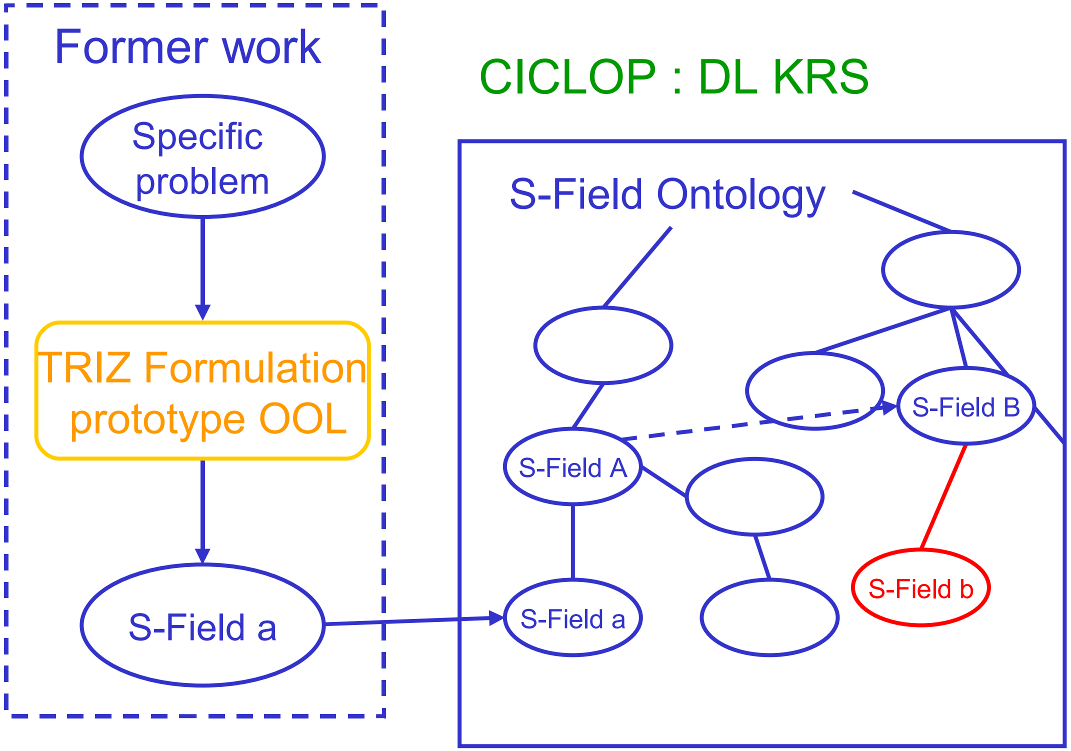
Subsumption reasonning, S-Field a is classified by CICLOP, and CICLOP finds its generic problematic S-Field.

Project achitecture



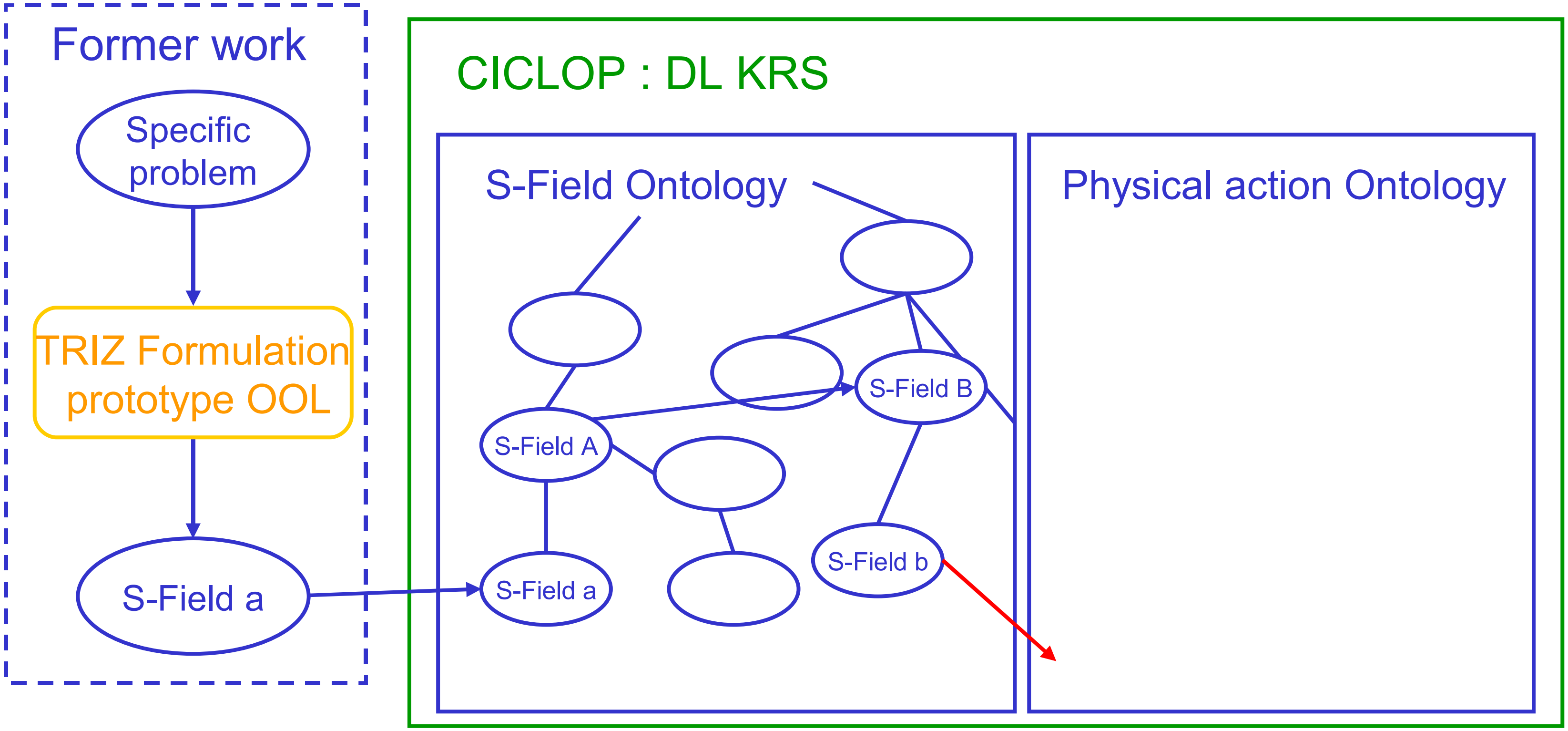
The standard is applied thanks to the application rules engine, it associates the generic problematic S-Field A to the generic solution S-Field B.

Project achitecture



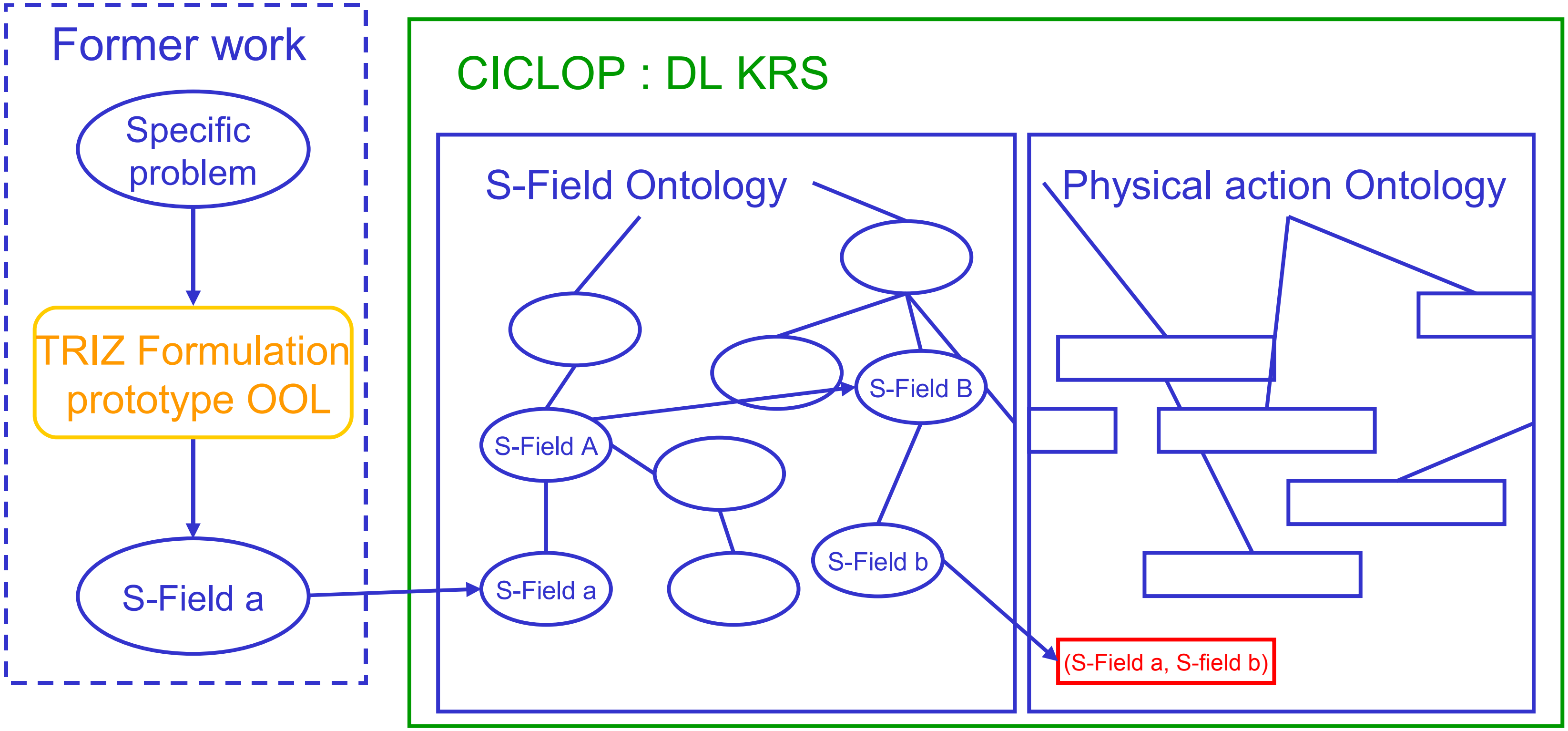
The generic S-Field B is specified by the substances and the fields contained in S-Field a, and it gives the S-Field b.

Project achitecture



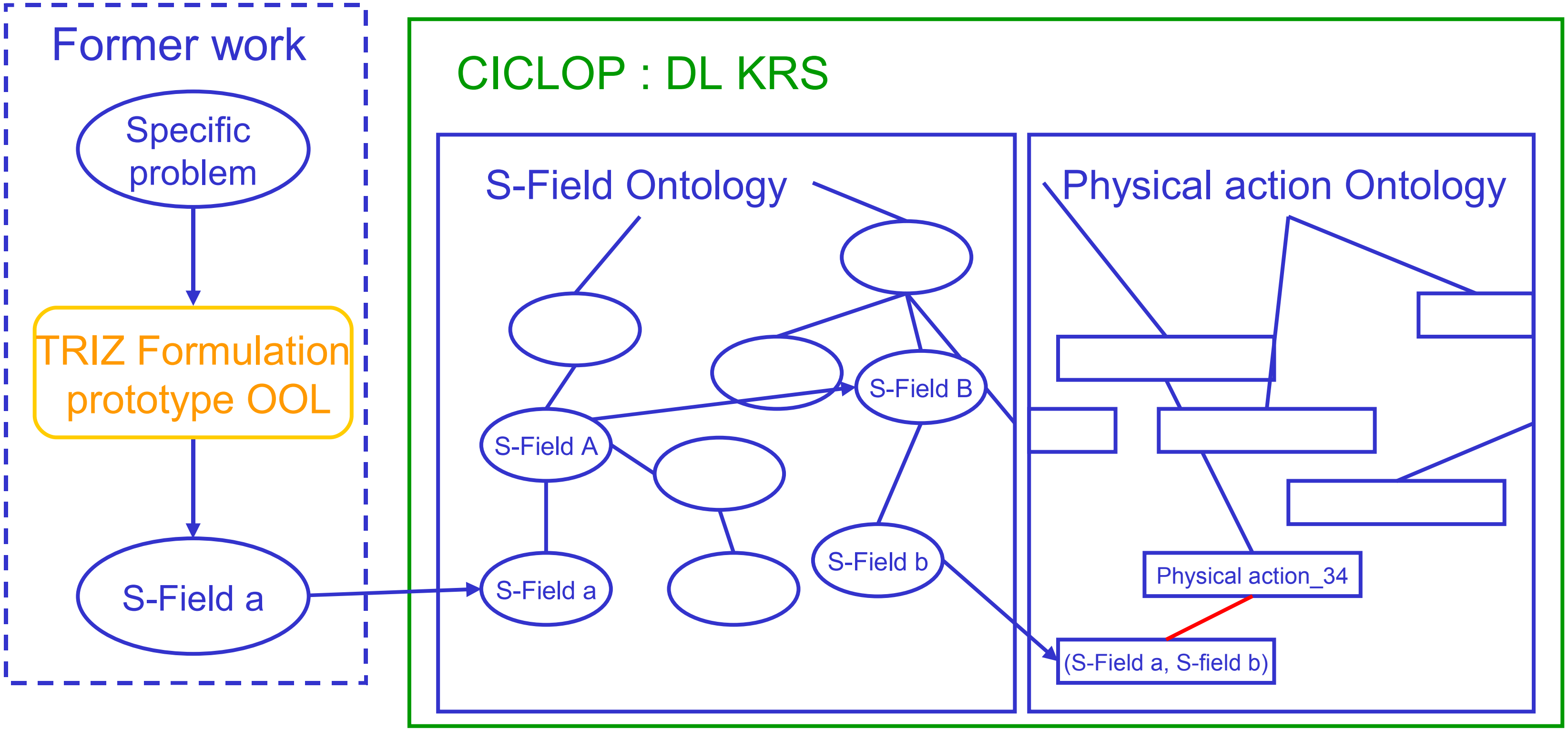
The application rules engine creates a couple of S-Field named Required-Action (S-Fiels a, S-Field b).

Project achitecture



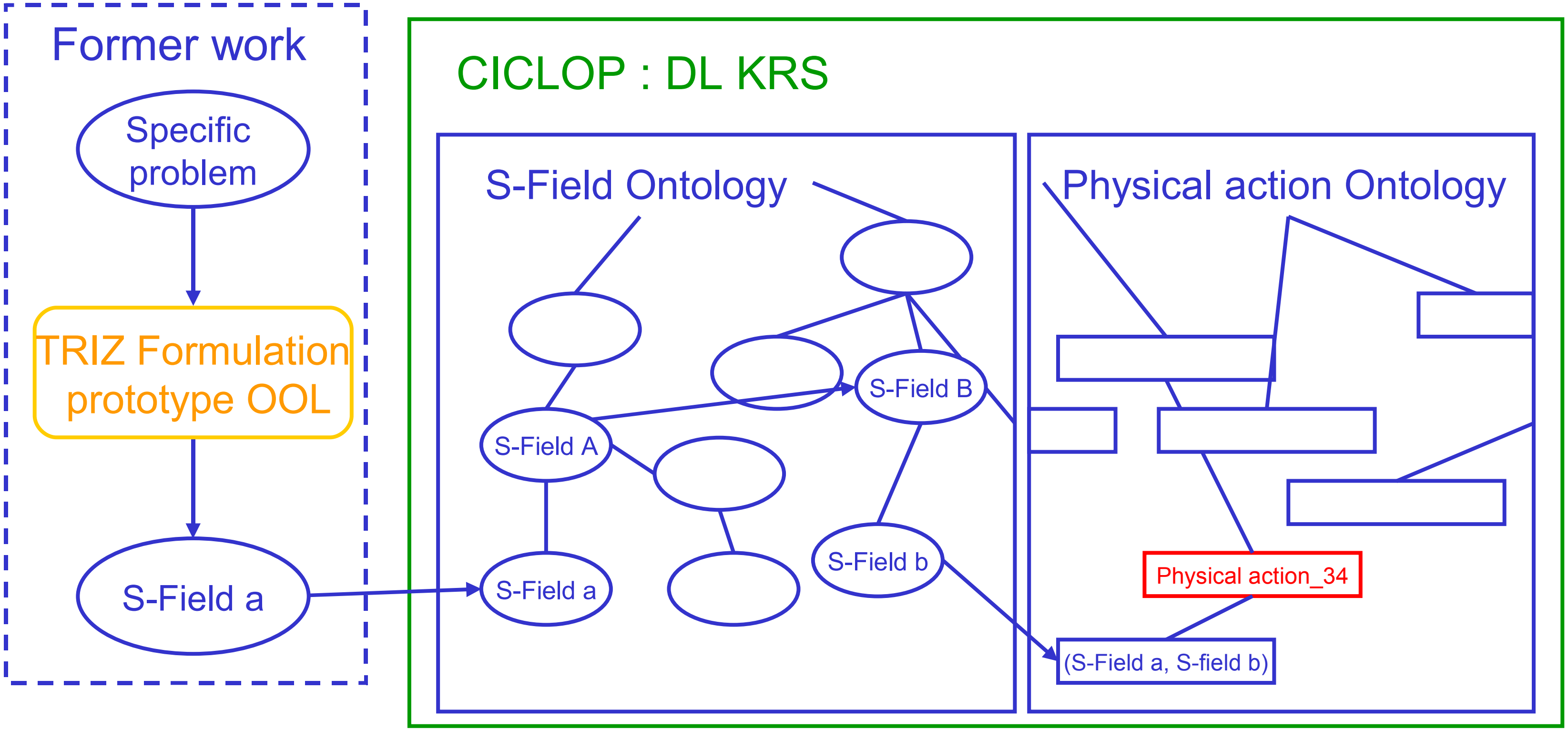
The required action is added to the physical action Ontology

Project achitecture



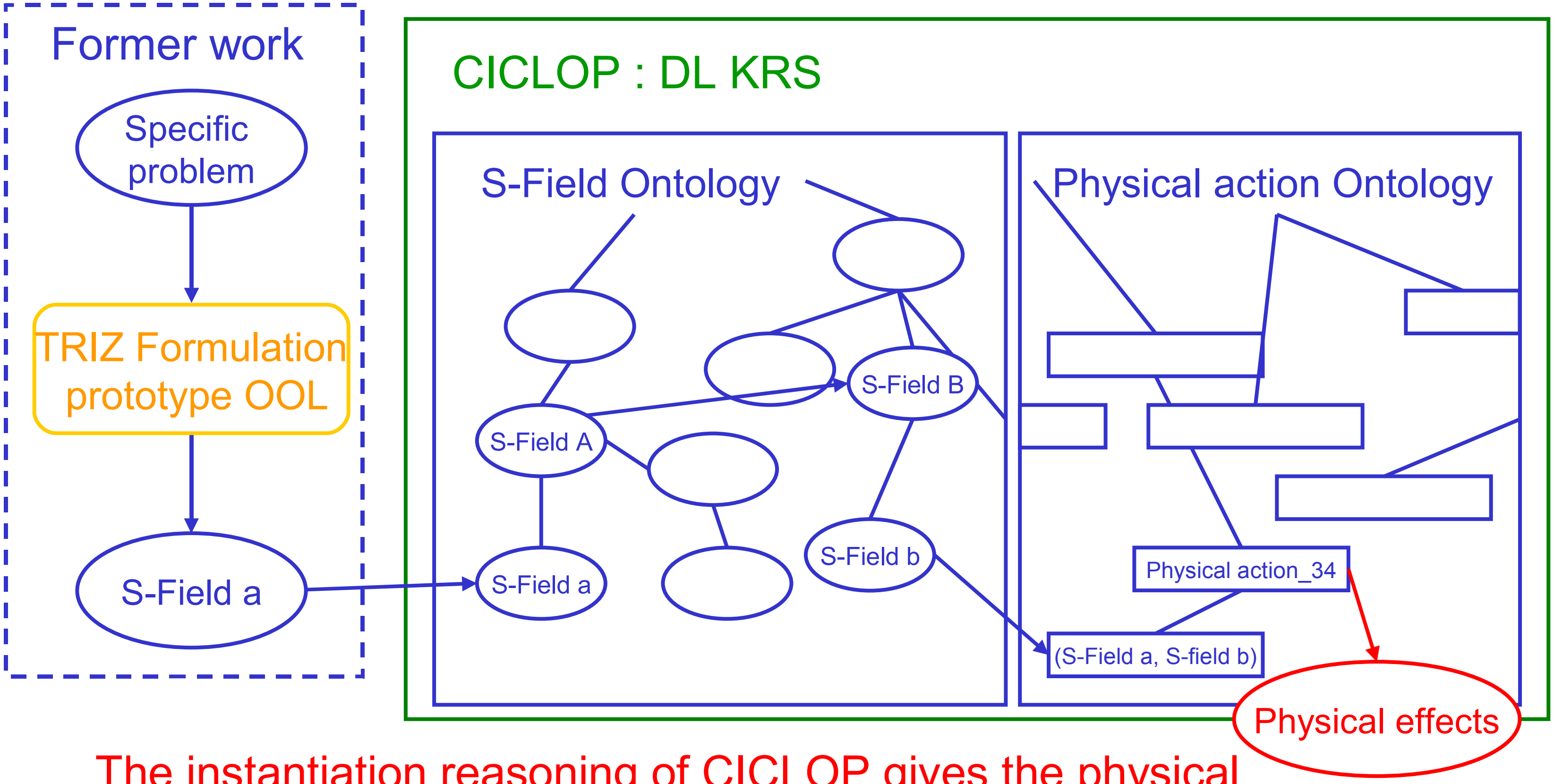
Ciclop classifies the required action is the physical action Ontology

Project achitecture



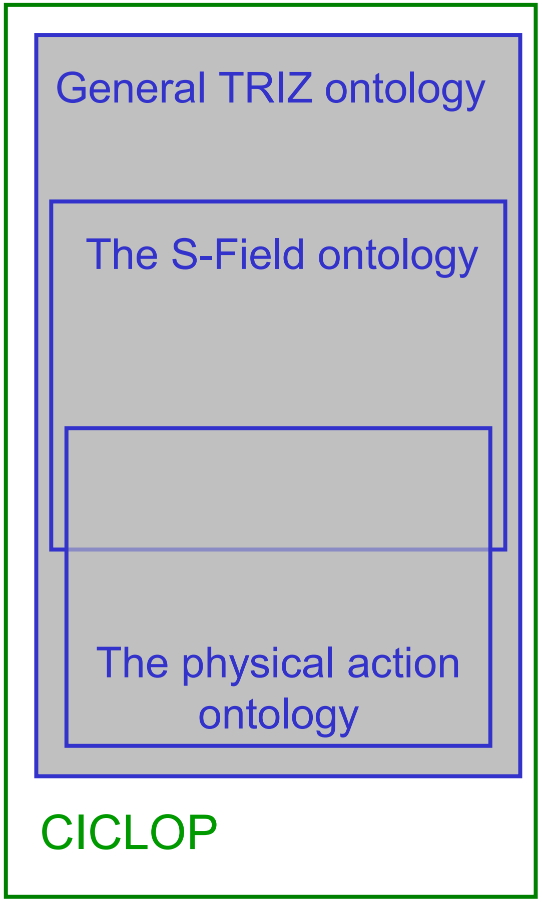
Lot of semantic connection could be tested, the semantic equivalence, the semantic kinship, …

Project achitecture



The instantiation reasoning of CICLOP gives the physical effects which could be used to have the solution S-Field b.

Ontologies



CICLOP contains a general TRIZ ontology obtained from the ontology of the formulation prototype, it’s a translation in DLs.

The general TRIZ ontology has been specified for the SFA, the specific term of SFA have been added to this ontology and form the S-Field ontology.

This S-Field ontology is specify with some physical terms (matter, bonding fields) in order to describe physical effects and to have the physical action ontology which will play the role of the Pointer to Effect.

DL-TRIZ (Alexis Bultey)

A problem solving environment based on TRIZ ontologies

- TRIZ ?
- The Substance-Field Analysis
- System Architecture
- Preliminary results and perspectives

Preliminary results

- The TRIZ ontologies
  - the S-Field ontology supports the S-Field modeling
  - the physical action ontology supports the S-Field interpretation
- From 250 effects, the environment converge to a restricted list according to the matter and field considered in the problem.
- The TRIZ proposed solutions are included in the list computed by the system.
- The selection is quite acurate (6 selected effects from 250)

Perspectives

- The application rules engine
  - a prototype based on SNARK system has been realized
  - the prototype have to be added to CICLOP
- The rules extraction
  - the algorithms of standards application of Tsourikov

Méta-Plan

- INSA Strasbourg
- LGECO
- LICIA
- Logiques de description
- Conclusion et perspectives

Conclusion et perspectives

- Un système de gestion de bases de connaissances fondé sur les logiques de description
- Validé par les applications
  - Conception intégrée
  - Conception innovante
- Collaborations
  - Raisonneur
  - Logiques floues
  - Bases de données

Merci de votre attention

- questions ?