UMR IRISA

Activity report 2013

Dpt 7: DATA AND KNOWLEDGE MANAGEMENT

Team LIS

Logical Information Systems

Rennes

1 Team

Head of the team
Olivier Ridoux, Professor, Université Rennes 1

Administrative assistant
Élodie Lequoc

Université Rennes 1 personnel
Yves Bekkers, Professor
Annie Foret, Assistant Professor, HDR
Sébastien Ferré, Assistant Professor
Benjamin Sigonneau, Research Engineer (part time: 70%)

Insa personnel
Mireille Ducassé, Professor
Peggy Cellier, Assistant Professor

PhD students
Mouhamadou Ba, ARED/Insa grant, since October 2012
Soda Marème Cissé, MENRT/Rennes 1 grant, since October 2012

Master students
Joris Guyonvarc'h, INSA Rennes, February-June 2013
Vincent Piet, ESIR, June-September 2013

Associate members
Erwan Quesseveur, Assistant Professor, Université Rennes 2, Associate Member
François Le Prince, President of ALKANTE and Associate Professor, Université Rennes 2, Associate Member

2 Overall Objectives

2.1 Overview

The LIS team aims at developing *formal* methods for handling complex data sets in a *flexible* and *precise* way. "Flexible" means that the content determines the shape of the container. Very often, it is the opposite that is observed; e.g., the tree-like shape of a hierarchical file system enforces the tree-like shape of software packages. "Precise" means that any subset of the data set can be easily characterized. Again, it is the opposite that is often observed; e.g., in a hierarchical file system only sub-trees can be easily characterized. More and more information is available on the Web, and more and more information can be stored on a single machine. However, whereas the related low-level technology is developing, and performance is increasing, little is done for organizing the ever-growing amount of information. Therefore, the LIS team addresses the issues of organizing and querying information in general. The solutions are to be both formal and practical. Operational issues such as index technologies are important, but we are convinced that their scope is too limited to solve the crucial issues.

At a formal level, *queries* and *answers* are two key notions. It is nowadays standard to consider queries as logical formulas and answers as special models of queries. Computing the preferred model of a query in some context is conceptually easy, and it warrants flexibility. However, the opposite is not that easy in general; given a subset of the data, how can we compute a query of which it is a model? Given two different subsets of the data, how can we compute a query that explains the difference? Knowing this would warrant precision. The LIS team proved that *formal concept analysis* (FCA [GW99]) is a powerful framework for analyzing $\langle query, answer \rangle$ pairs. *Formal concepts* formalize the association between a query and its answers. Formal concepts are structured into a lattice which provides navigation links between concepts.

However, standard FCA cannot deal with queries considered as logical formulas (recall that this is the key for flexibility). Therefore a variant of FCA for logical description has been developed [7] altogether with the generic notion of *Logical information system* (LIS) that provided a reconstruction of all information system operations based on logical concept analysis. In particular, some data-mining operations are native in LIS [7, 5].

The mottoes of the LIS research are:

1. *Never impose* a priori *a structure on information.* E.g., do not use hierarchical structures. Imposing *a priori* a structure causes the *tyranny of the dominant decomposition*[TOHS99]. For instance, the usual class-based organisation of source code makes highly visible the connections between methods of the same class, but masks the possible connections between methods in different classes.

   Instead, consider pieces of information as a bulk. Structure should emerge *a posteriori* from the contents or the point of view. As a consequence, updating the contents may change the structure: we accept it.

2. *Consider every possible rational classification, and permit changes at any time.* Here, rational means that what makes a piece of information belong or not to a class depends on the very piece of information, not on other pieces. The concept lattice induced by FCA is precisely a means to grasp all possible rational classifications.

3. *Rare events are as important as the frequent ones.* One cannot say *a priori* if a piece of information is interesting because it presents a frequent pattern, or because it presents a rare pattern.

[GW99] B. Ganter, R. Wille, *Formal Concept Analysis — Mathematical Foundations*, Springer, 1999.

[TOHS99] P. Tarr, H. Ossher, W. Harrison, S. Sutton, "N Degrees of Separation: Multi-Dimentional Separation of Concerns", *in: ICSE*, IEEE Computer Society, p. 107–119, 1999.

So, rare events must not be masked by statistical artefacts. Statistics is not forbidden, but it is only a complement of a symbolic logic approach.

4. *Queries should be possible answers.*

   In usual information systems (say relational databases or Web browsers) there is a strict dichotomy between queries (they are intensional expressions), and answers (they are strictly extensional expressions, i.e. sets of things). We contend that a good answer must be a mix of extensional and intensional answers. E.g. the good answer to "I would like to buy a book" is seldom the whole catalog of the bookshop; it is more relevant to answer such a query with other queries, like "Is this for a child" or "Do you prefer novels or documents".

   Note that hierarchical file systems already do that. Queries (i.e., *filepaths*) yield answers that contain other queries (i.e., *sub-directories*). One of the LIS achievements is a formalization of this behaviour that does not rely on an *a priori* hierarchical structure.

Our research is intended to be *vertical* in the sense that all aspects of information systems are of interest: design, implementation, and applications.

On the implementation side, the LIS team develops systems that present the LIS abstraction at the user level [9].

On the application side, the LIS team explores the application of LIS to *Geographical information systems* (GIS). The intuition here is that the traditional layered organization of information in GIS suffers a rigid structure of thematic layers. Moreover, GIS applications usually cope with highly heterogeneous information and large amount of data; this makes them an interesting challenge for LIS. The team also works on a data-mining interpretation of bug tracking. In this case, the intuition is that pieces of information relevant to software engineering, e.g. programs, specifications or tests, can be explored very systematically by a LIS. More generally, applications to software engineering are important for the team. A recent trend of application is the assistance to group decision and negotiation. In particular LIS provide new technological support to *social choice.* For example, in committee decision making, navigating with LIS in the facts recorded in a context allows decision makers to treat all candidates in a fair way [3].

2.2 Key Issues

In its current state, LIS studies the following key issues:

- The LIS formalism is generic w.r.t. the logic used for describing pieces of information.

  What are the appropriate logics for the application fields that we have chosen? (GIS and error localization) Do we need a brand new logic for every application, or is there something that different applications can share?

- Genericity of LIS w.r.t. logic opens the door for creating ad hoc logics for describing pieces of information of an application. We already have proposed the framework of *logic functors* for helping a user build safely ad hoc logics. Logic functors are certified logic components that can be composed to form certified implementations of a logic.

What are the useful logic functors? How can we be sure that a toolbox of logic functors is complete for a given purpose?

Can the idea of certified composition be applied to another domain? Given a domain *foo*, *foo* functors would be certified *foo* components that can be assembled to form certified implementations of *foo* systems.

Is it possible to certify other properties than meta-logical properties? E.g. is it possible to characterize complexity, or other non-functional properties like security?

- A family of non-commutative logics has developed over the years in the domain of computational linguistics, e.g. Lambek logic, pregroups. As for LIS, a great amount of creativity is expected for extending this family with ad hoc logics that would tackle fine-grained linguistics phenomena.

  Is it possible to build up an implementation of these logics using logic functors?

  Some LIS applications deal with objects that are sequential by nature (say, texts).

  Can these non-commutative logics primarily developed for computational linguistics help in LIS applications?

- Hierarchical file systems have a preferred metaphor which is the tree.

  What is the proper metaphor for LIS?

  The tree is also the graphical metaphor of hierarchical file systems.

  What is the graphical metaphor for LIS?

  Knowing this is crucial for the acceptance of LIS in end-user applications.

- Geographical information systems also suffer the *tyranny of the dominant decomposition.* Here, the dominant decomposition is in rigid thematic layers that inherit from plastic sheets of ancient map design. These layers are omnipresent in the design and interface of GIS applications.

  How can LIS abstract these layers, and still display layers when needed?

  Mining geographical information is difficult because of the layers and because it must cope with complex spatial relations.

  What is the proper modeling of these relations that will permit efficient LIS operations, including data-mining?

- Up to now, mining execution traces for bug tracking has used poor trace representations and ad hoc algorithms.

  How can the theoretical and practical framework of LIS help benefit from the wide range of information of program development environments?

- The file system implementation of LIS can handle around 1 million elementary pieces of information, which corresponds approximately to a full homedir with 10 to 20 thousands files. This is rather small compared to relational database capabilities, but already large compared to other approaches based on formal concept analysis.

How can it handle more? Can we reach 100 million in the next few years?

3 Scientific Foundations

3.1 Logics for Information Systems

Keywords: Syntax, interpretation, semantics, subsumption.

Glossary :

Syntax Definition of the well-formed statements of a language. Statements are finite.

Interpretation Complete description of a world. Interpretations can be arbitrary mathematical constructs, and so can be infinite. Interpretations are models of statements, namely the worlds in which the statement is true. Statements are features of interpretations, namely the statements that are true in the world.

Semantics A binary relation between syntactic statements and interpretations.

Subsumption A relation which states that a property is more specific than another property.

Logic is the core of Logical Information Systems. However, this does not say everything because every particular usage of logic is also a point of view on logic. For instance, logic in Logic Programming is not the same as in Description Logics. This section describes the point of view on logic from information systems.

Logic is a wide domain that is concerned with formal representation and reasoning. The point of view on logic in logical information systems can be characterized by two things. Firstly, we are interested in the individual description of objects (e.g., files, pictures, program functions or methods), so that we need to represent concrete domains and data structures. This entails two levels of statements: (1) statements about objects, and (2) statements about the world (e.g., ontologies and *subsumption*). Subsumption helps to decide when an object is an answer to a query. Secondly, we need automated reasoning facilities as the subsumption must be decided between any object and a query in information retrieval. This forces us to only consider decidable logics, unless consistency or completeness are weakened.

Properties of a Logic A characteristic of logic is the ability to derive new statements from known statements. Such a derivation is valid w.r.t. semantics only if every model of the known statements is also a model of the new statements. This ability opens the room for *reasoning*, i.e. the production of valid statements by working at the syntactical level only. Reasoning is formalized by *inference systems* (e.g., axioms and rules). An inference system is *consistent* if it produces only valid statements; it is *complete* if it produces all valid statements. Reasoning is *decidable* if a consistent and complete inference system can be realized by an algorithm.

Examples of Logics for Information Systems Proposition logic is a possible logic for an information system, but it needs a lot of encoding for handling structured information. Instead, non-standard logics have been defined for some structured domains.

A large family of logics that comes into our scope is the family of Description Logics (DL) [Bra79,CLN98], which have been widely studied, implemented, and applied in knowledge and information management. Moreover, their semantic structure is especially well-suited to be used in a LIS. The semantics of proposition logic is often exposed in terms of truth values and truth tables. To the contrary, the semantics of description logic is defined in terms of sets of objects that are close to answers to a query. DL are, therefore, of a special interest for the LIS team.

Another family of interest is *categorial grammars*. Many substructural logics come into this scope, among which non-commutative linear logic or Lambek Calculus[Lam58] that handle various concatenation principles (or ordered conjunction) in categorial grammars where logic is used both for attaching formulas to objects and for parsing seen as deduction.

At an empirical level, the categorial approach comes very close to the LIS approach. Categorial grammars correspond to LIS contents, because they both attach formulas to objects, and sentence types correspond to queries. The difference is that the answer to a LIS query is an unordered set, whereas a sentence generated by a categorial grammar is an ordered sequence. We expect a cross-fertilization of both theories in the future, especially in the LIS applications where the objects are naturally ordered.

3.2 Concept Analysis

Keywords: Objects, descriptors, context, instance, property, extension, intension, concept.

Glossary :

Objects A set of distinguished individuals.

Descriptors A set of distinguished properties.

Context A set of objects associated with descriptors.

Instance An object is an *instance* of a descriptor if it is associated with it in a given context.

Property A descriptor is a *property* of an object if it is associated with it in a given context.

Extension The *extension* of a collection of descriptors is the set of their common instances. Extent is a synonym.

Intension The *intension* of a collection of objects is the set of their common properties. Intent is a synonym.

Concept Given a context, and extensions and intensions taken from it, a *concept* is a pair (E, I) of an extension E and an intention I that are mutually complete; i.e., I is the intention of the extension, and E is the extension of the intention.

[Bra79] R. J. Brachman, "On the Epistemological Status of Semantic Nets", *in: Associative Networks: Representation of Knowledge and Use of Knowledge by Examples*, N. V. Findler (editor), Academic Press, New York, 1979.

[CLN98] D. Calvanese, M. Lenzerini, D. Nardi, "Description Logics for Conceptual Data Modeling", *in: Logics for Databases and Information Systems*, J. Chomicki, G. Saake (editors), Kluwer, p. 229–263, 1998.

[Lam58] J. Lambek, "The Mathematics of Sentence Structure", *American Mathematical Monthly 65*, 1958, p. 154–170.

Formal Concept Analysis Formal Concept Analysis (FCA) is part of the mathematical branch of applied lattice theory [Bir40,DP90]. It can be seen as a reformulation by Wille of Galois lattices [BM70] that emphasizes lattices as conceptual hierarchies [Wil82]. The mathematical foundations of FCA have been extensively studied by Ganter and Wille [GW99].

FCA mainly aims at the automatic construction of *concepts* and their classification according to a generalization ordering, given a flat representation of data. The adjective *formal* means that concepts are given a mathematical definition, which reflects the usual philosophical meaning of a "concept". The basic notions of FCA are those of *formal context*, and *formal concept*.

A *formal context* is a binary relation between a set of objects, and a set of attributes. Through this relation attributes can be seen as properties of objects, and reciprocally, objects can be seen as instances of attributes. This is a very general settings that applies to various domains such as data analysis, information retrieval, data-mining or machine learning. In all these domains, the objects of interest are described by sets of attributes, and the objective is to relate in some way sets of objects and sets of attributes. In information retrieval a set of attributes is a query, whose answers is a set of objects. In machine learning a set of objects is a set of positive examples, whose characterization is a set of attributes.

A *formal concept* is the association of a set of objects, the *extent*, and a set of attributes, the *intent*. This comes close to the classical definition of concept in philosophy, but in FCA the relationship between extent and intent is formally defined. The extent must be the set of instances shared by all attributes of the intent; and the intent must be the set of properties shared by all objects in the extent.

The fundamental theorem of FCA says that the set of all concepts forms a complete lattice when they are ordered according to the set inclusion on extents (or intents). This is called the *concept lattice*, and it can be computed automatically from the formal context. The concept lattice is the structure that is implicit in any formal context. It contains all the information contained in the formal context; the latter can be rebuilt from the former. In data analysis, the concept lattice permits a flexible classification of data (where a concept is a class), because concepts are not organized as a strict hierarchy. In information retrieval and data-mining it is used as a search space for answers.

Logical Concept Analysis In Formal Concept Analysis (FCA) object properties are restricted to Boolean attributes. In many applications there is a need for richer properties, where properties are not independent. For instance, if a book has been published in 2000, it can be given the property `year = 2000`, and has then the implicit properties `year in 1990..2000` and `year in 2000..2010`. This means that properties are statements about objects that can

[Bir40] G. Birkhoff, *Lattice Theory*, American Mathematical Society, 1940.

[DP90] B. A. Davey, H. A. Priestley, *Introduction to Lattices and Order*, Cambridge University Press, 1990.

[BM70] M. Barbut, B. Monjardet, *Ordre et classification — Algèbre et combinatoire (2 tomes)*, Hachette, Paris, 1970.

[Wil82] R. Wille, *Ordered Sets*, Reidel, 1982, ch. Restructuring lattice theory: an approach based on hierarchies of concepts, p. 445–470.

[GW99] B. Ganter, R. Wille, *Formal Concept Analysis — Mathematical Foundations*, Springer, 1999.

be subject to reasoning, exactly like logical statements. Other examples of useful properties are strings and string patterns, spatial descriptions for locating objects, or patterns over the programming type of functions and methods.

FCA has been extended by other authors to handle multi-valued contexts [GW99], but this extension takes the form of a preprocessing stage that results in a standard formal context, and forgets all logical relations between properties. Moreover it is limited in practice to valued attributes with finite domains of attributes. In 2000 we proposed a logical generalization of FCA, named Logical Concept Analysis (LCA) [7], that is the abstraction of FCA w.r.t. object descriptions and concept intents. This makes LCA an abstract component, and makes FCA the composition of LCA with a logic component. LCA makes the theory of concept analysis easily reusable in various applications.

For good composability of LCA and logics, they must agree on the specification of logics. What LCA needs from a logic is:

- a language of formulas (or statements), L, for the representation of object descriptions and concept intents,
- a procedure, $\sqsubseteq$, for deciding the subsumption between 2 formulas; $\sqsubseteq$ means “is subsumed by”, “is more specific than”, “entails”,
- a procedure, $\sqcup$, for computing the least common subsumer of 2 formulas; it is a kind of logical disjunction,
- a formula, $\perp$, that is the most specific according to subsumption (logical contradiction).

This specification provides everything required to extend fundamental results of FCA to LCA (formal context, extent, intent, concept, complete lattice of concepts). For information retrieval and the expression of queries, it is useful to add, to this specification, operations such as logical conjunction, and logical tautology (the most general formula).

Any formal context defines a logic whose subsumption relation is isomorphic to the concept lattice that is derived from the formal context. An interesting result is that the *contextualized logic* (the logic defined by the logical context) is a refinement or extension of the logic used by LCA. Everything true in the logic is also true in the contextualized logic (because it is *eternal truth*); and everything true only in the contextualized logic says something that is true in the context, but not in general (because it is *instant truth*). Thus, contextualized logic forms the basis for data-mining and machine learning tasks, whose aim is to discover outstanding regularities in a given context [7, 5].

3.3 Logical Querying, Navigation, and Data-mining

Keywords: Querying, navigation, data-mining.

Glossary :

Querying The process that takes a query (e.g., a logical formula), and returns the collection of objects that satisfy the query (e.g., the extent of the query).

[GW99] B. Ganter, R. Wille, *Formal Concept Analysis — Mathematical Foundations*, Springer, 1999.

Navigation The process of moving from place to place, where each place indicates objects they contain (i.e. *local objects*) and other places where it is possible to move (i.e. *neighbouring places*).

Data-mining The process of extracting outstanding regularities from data (e.g., a context) hoping to discover new and useful knowledge.

In most information systems, querying and navigation are two disconnected means for information retrieval. With querying, users formulate queries which belong to more or less complex querying languages, from simple words as in Google to highly structured languages like SQL. The system returns a set of answers to the query. This permits expressive search criteria over large amounts of data, but lacks interactivity because the dialogue is only one-way. If the answers are not satisfying, users have to imagine new queries and formulate them, which requires *a priori* knowledge of both querying language and data. With navigation, users move from place to place following links. The most common systems are folder hierarchies (e.g., file systems, bookmarks, emails), and hypertext. As opposed to querying, navigation provides interactivity by making suggestions at each step, but offers limited expressivity because navigation structures are rigid. In a hierarchy, selection criteria are presented in a fixed order. For instance, if pictures are classified first by date, then by type, one cannot easily find all landscape pictures.

The need for combining querying and navigation has already been recognized. Most proposals, however, are unsatisfying. Indeed, either querying and navigation cannot be mixed freely in a same search, or consistency of querying is not maintained. An example of the former is SFS [GJSO91], once a querying step is done, there is no more navigation. An example of the latter is HAC [GM99], some query answers may not satisfy the query. A proposal based on FCA has not these drawbacks [GMA93], and we have generalized it to work within LCA, which allows us to use logical formulas for object description and queries [7]. Logic brings expressivity in querying, and concept analysis brings the concept lattice as a navigation structure (i.e., navigation places are formal concepts). The advantages of this navigation structure is that (1) it is automatically derived from data, the logical context (see motto 1), (2) it is complete as navigation alone makes it possible to reach any object (see motto 3), and (3) it is flexible because selection criteria can be chosen in any order, thus allowing user to express their preferences (see motto 2). Querying and navigation can be freely mixed (see motto 4) in a same search because every logical formula points to a formal concept, and every formal concept is labelled by a logical formula. Put concretely, this means that a user can at each step of his search: either modify by hand the current query and reach a new place, or follow a suggested link that will modify the current query and reach a new place.

The critical operation is the computation of navigation links, which correspond to edges in

[GJSO91] D. K. Gifford, P. Jouvelot, M. A. Sheldon, J. W. J. O'Toole, "Semantic file systems", *in: ACM Symp. Operating Systems Principles*, ACM SIGOPS, p. 16–25, 1991.

[GM99] B. Gopal, U. Manber, "Integrating Content-Based Access Mechanisms with Hierarchical File Systems", *in: third symposium on Operating Systems Design and Implementation*, USENIX Association, p. 265–278, 1999.

[GMA93] R. Godin, R. Missaoui, A. April, "Experimental Comparison of Navigation in a Galois Lattice with Conventional Information Retrieval Methods", *International Journal of Man-Machine Studies 38*, 5, 1993, p. 747–767.

the concept lattice. Indeed, the worst-case time complexity for computing the concept lattice is exponential in the number of objects, which makes it intractable in most interesting cases. We demonstrated both in theory and practice that this computation is not necessary. A key feature of LIS is that its semantics is expressed in terms of LCA, though it is not required to actually build the concept lattice. This is opposed to most (all?) previous proposals for using LCA in information retrieval.

The concept lattice upon which our navigation is based is also a rich structure for data-mining and machine learning [Kuz04]. Here again, we have combined existing techniques with logic [7, 5, 2], and applied them to the automatic classification of emails, and the prediction of the function of proteins from their sequence [5].

3.4 Genericity and Components

Keywords: Abstraction, reusability, composability, component.

Glossary :

Abstraction a mechanism and practice to reduce and factor out details so that one can focus on few concepts at a time.

Reusability the likelihood a segment of structured code can be used again to add new functionalities with slight or no modification. Reusable code reduces implementation time, it increases the likelihood that prior testing and use has eliminated bugs and it localizes code modifications when a change in implementation is required.

Composability a system design principle that deals with the inter-relationships of components. A highly composable system provides recombinant components that can be selected and assembled in various combinations to satisfy specific user requirements.

Component a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties.

The application scope of Logical Information Systems is very large, and we do not expect that one design (e.g., one logic) will fit all possible applications. That is why we emphasize genericity, and we use the plural in "logical information systems". The need for genericity is not limited to theoretical results and design, but extends to the concrete implementation of LIS.

Genericity requires programming facilities for *abstraction*, *composability*, and *reusability* of *software components.*

In LIS, abstraction is of the upper importance in the design of logical concept analysis; LCA is an abstraction of FCA. It is also at the heart of the *logic functor* framework and of its implementation; a logic functor is an abstraction of a logic (see Section 3.5). Reusability and composability are the expected outcomes of this framework. It is expected to make things easier to the designer of a LIS application. Composability is also at the heart of the very notion of formal context, and thus at the heart of concept analysis. Indeed, the flat structure

[Kuz04] S. O. Kuznetsov, "Machine Learning and Formal Concept Analysis.", *in: Int. Conf. Formal Concept Analysis*, P. W. Eklund (editor), *LNCS 2961*, Springer, p. 287–312, 2004.

of formal concepts makes it trivial to extend a context or merge two contexts, and the burden of giving a structure to the context is left to the construction of the concept lattice.

A generic implementation of LIS can be seen as a central component that is parameterized by several application-dependent components: at least a logic, and a transducer for importing data. These parameter components can be linked at compilation time (plugins). The central component as well as parameter components can themselves be the result of the composition of smaller components.

3.5 Logic Functors

Keywords: Logics, genericity, composability.

The genericity w.r.t. logic implies that for every new application a logic has to be found for describing objects in a logic context. Either a suitable logic is already known, or it must be created. Creating a logic requires designing a syntax, a semantics, algorithms for subsumption and other procedures, and proving that these algorithms are correct w.r.t. semantics. This definitely requires logic expertise and programming skills, especially for the subsumption procedure that is a theorem prover for which consistency and completeness must be proven. However, application developers and logic experts are likely to be different persons in most cases. Moreover, creating new logics from scratch for each application is unsatisfying w.r.t. reusability as these logics certainly share common parts. For instance, many applications need propositional reasoning, only changing the notion of what is a propositional variable.

We introduced high-level logic components, named *logic functors* [6], in order to make the creation of a new logic the mere composition of abstract and reusable components. All logics share a common specification that contains all useful procedures (e.g., subsumption); logic functors are functions from logics to logics, implemented as parameterized modules. Some logic functors take no parameter, and provide stand-alone but reusable logics: this is the case of concrete domains such as integers or strings. Other logic functors take one or several logics as parameters. For instance the functor `Prop(X)` is propositional logic abstracted over its atoms. This makes it possible to replace atoms in propositional logic by the formulas of another logic (e.g., valued attributes, terms from a taxonomy).

Logics are built by applying logic functors to sub-logics, which can themselves be defined as a composition of logic functors. For instance, the propositional logic where atoms are replaced by integer-valued attributes (and allowing for integer intervals) can be defined by the expression

```
L = Prop(Set(Prod(Atom,Interval(Int)))).
```

This results in a concrete software component `L` that is fully equipped with implementations of the logic specification procedures. This component can then be composed itself with LCA or a LIS system.

3.6 Categorial Grammars

Keywords: Categorial grammar, identification in the limit.

Categorial grammars are used for natural language modeling and processing; they mainly handle syntactic aspects, but Lambek variants also have a close link with semantics and Lambda-calculus. Formally, a *categorial grammar* is a structure $G = (\Sigma, I, S)$ where: Σ is a finite alphabet (the words in the sentences); I is a function that maps a finite set of types to each element of Σ (the possible categories of each word, a lexicon); S is the *main type* associated to correct sentences. A *k-valued categorial grammar* is a categorial grammar where, for every word $a \in \Sigma$, $I(a)$ has at most k elements. A *rigid categorial grammar* is a 1-valued categorial grammar. Rigidity is a useful constraint to get learnable subclasses of grammars (and related algorithms).

Each variant of categorial grammar formalism is also determined by a derivability relation on types $\vdash$ (which can be seen as a subcase of *linear logic* deduction in the case of Lambek grammars). Given a categorial grammar $G = \langle \Sigma, I, S \rangle$, a sentence w on the alphabet Σ belongs to the language of G whenever the words in w can be assigned by I a sequence of types that derive (according to $\vdash$) the distinguished type S.

A simplified example is $G_1 = (\Sigma_1, I_1, S)$ with $\Sigma_1 = \{John, Mary, likes\}$ $I_1 = \{John \mapsto \{N\},\ Mary \mapsto \{N\},\ likes \mapsto \{N \backslash (S/N)\}\}$ the sentence “John likes Mary” belongs to the language of G_1 because $N, N\backslash(S/N), N \vdash S$ due to successive applications of the two elimination rules : $X, X \backslash Y \vdash Y$ and $Y, Y/X \vdash Y$. Type constructors / and \ can be seen as oriented logic implications, the elimination rules are analogues of the “Modus Ponens” logic rule. An interesting issue is how the underlying rules or logics may compose (this is the design of logic functors) to deal with more fine-grained linguistic phenomenon.

Since they are lexicalized, such grammar formalisms seem well-adapted to automatic acquisition or completion perspectives. Such studies are performed in particular in Gold’s paradigm.

Identification in the limit in the model of Gold consists in defining an algorithm on a finite set of (possibly structured) sentences that converges to obtain a grammar in the class that generates the examples. Let $\mathcal{G}$ be a class of grammars that we wish to learn from positive examples; let $\mathcal{L}(G)$ denote the language associated with a grammar G; a *learning algorithm* is a function ϕ from finite sets of (structured) strings to $\mathcal{G}$, such that for any $G \in \mathcal{G}$ and $\langle e_i \rangle_{i \in \mathbb{N}}$ any enumeration of $\mathcal{L}(G)$, there exists a grammar $G' \in \mathcal{G}$ such that $\mathcal{L}(G') = \mathcal{L}(G)$ and $n_0 \in \mathbb{N}$ such that $\forall n > n_0\ \phi(\{e_0, \ldots, e_n\}) = G'$.

Categorial grammars have close connections with logic. We therefore consider different ways of connecting computational linguistic data and LIS, such as implementing parameterized pregroups as a logic functor [10].

4 Application Domains

4.1 Geographical Information Systems

Participants: Soda Marème Cissé, Olivier Ridoux, Peggy Cellier, Erwan Quesseveur, François Le Prince, Sébastien Ferré.

Geographical Information Systems (GIS) is an important, fast developing domain of Information technology, and it is almost absent from INRIA projects. It is especially important for local communities (e.g. region and city councils).

Geographical information systems [LT92] handle information that are localized in space (*geolocalized*). GIS form an area which incorporates various technologies such as web, databases, or imaging. One characteristic of GIS is their organization as *layers*. This is inherited from the plastic sheets that where used until recently for drawing maps. A layer represents the road system, another the fluvial system, another the relief, etc. This is another instance of the tyranny of the dominant decomposition, and is not satisfactory: to which layer belong bridges, into which layer can we represent a multimodal network? Moreover, mining GIS is known to be difficult for the same reason; the layer structure makes inter layer relationships difficult to discover.

The first advantage of applying LIS to GIS is to allow cross-layer navigation. Another advantage is to permit a logical handling of scales. In current GIS systems, scales are treated as different layers, and it is difficult to keep the consistency between all layers that describe the same object. Another advantage that we have observed in a preliminary work is that LIS helps cleaning a data-base. This was not expected, and opens an interesting research area. Another characteristic of GIS is an intensive usage of topological relations (touches, overlaps, etc) and geographical relations (North, upstream, etc). Logic offers a rich language for expressing these relations and combining them.

4.2 Group Decision and Negotiation

Participants: Mireille Ducassé, Peggy Cellier.

Group decision and negotiation focuses on complex and self-organizing processes that constitute multiparticipant, multicriteria, ill-structured, dynamic, and often evolutionary problems. Group decision and negotiation refers to the whole process or flow of activities relevant to reaching a group decision, and not merely to the final choice - aspects of the process in group decision and negotiation include scanning, communication and information sharing, problem definition (representation) and evolution, alternative generation, and social-emotional interaction. Group decision support systems (GDSS) and negotiation support systems (GDNSS) are amongst the major approaches to address the problems.

In the current thread of research, we are showing that Logical Information Systems provide an innovative technological support for most of the above mentioned aspects of GDSS. In particular, the navigation and filtering capabilities of LIS help detect inconsistencies and missing knowledge during meetings. The updating capabilities of LIS enable participants to add objects, features and links between them on the fly. As a result the group has a more complete and relevant set of information. Furthermore, the compact views provided by LIS and the OLAP features (see Section 5.3) help participants embrace the whole required knowledge. The group can therefore build a shared understanding of the relevant information previously distributed amongst the participants. Lastly, the navigation and filtering capabilities of LIS are relevant to quickly converge on a reduced number of targets. A future trend of research will be to investigate how LIS can also support negotiation.

[LT92] R. Laurini, D. Thompson, *Fundamentals of Spatial Information Systems*, Elsevier, Academic Press Limited, 1992.

5 Software

5.1 Camelis and Sewelis

Participants: Sébastien Ferré.

Camelis is a stand-alone application that allows to store, retrieve and update objects through a graphical interface. Its main purpose is to experiment with the LIS paradigm. In particular, it has been very useful for refining the query-answer principle in special circumstances (e.g. when there are many answers, or when there are few answers). It is currently used as a personal storage device for handling photos, music, bibliographical references, etc, up to tens of thousands of objects. It implements as closely as possible the LIS paradigm. It is generic w.r.t. logics, and is compatible with our library of logic functors, LogFun (see Section 5.2). It is available on Linux and Windows, and comes with a user manual.

An important extension, Sewelis, has been developped to browse RDF(S) graphs, a Semantic Web standard. It uses a query language whose expressivity is similar to SPARQL, the reference query language of the Semantic Web. The LIS navigation has been proved safe (i.e., does not lead to dead-ends), and complete (i.e., can reach all conjunctive queries), so that users can perform complex searches easily and safely [4]. Sewelis also supports the guided creation and update of objects, according to the UTILIS approach [11].

5.2 LogFun

Participants: Sébastien Ferré.

The formal definition of a LIS is generic with respect to the logic used for object descriptions and for queries. The counterpart is that it is up to the user to design and implement a logic solver to plug in a LIS. This is too demanding on the average user, and we have developed a framework of *logic functors* that permits to build *certified* logic solvers (see Section 3.5).

LogFun is a library of *logic functors* and a *logic composer*. A user defines a logic using the logic functors, and produces a certified software implementation of the logic (i.e., parser, printer, prover) by applying the logic composer to the definition. For instance, using a functor *Interval* for reasoning on intervals (e.g. $x \in [2,5] \Longrightarrow x \in [0,10]$), and a functor *Prop* for propositional reasoning (e.g. $a \wedge b \Longrightarrow a$), a user can define logic *Prop(Interval)*. In this logic, a theorem like $x \in [2,5] \vee x \in [7,9] \Longrightarrow x \in [0,10]$ can be proven. Note that $[2,5] \cup [7,9]$ *is not* an interval, so that *Prop(Interval)* is an actual extension over *Interval*.

What the logic composer does when building logic *Prop(Interval)* is to compose the solver of *Interval* and the generic solver of *Prop*, and build a solver for *Prop(Interval)*. It also type-checks *Prop(Interval)* to produce its certificate using the certificates of *Interval* and *Prop*. In this example, the certificate says that *Prop(Interval)* is complete: everything that could be deduced from the meaning of *Prop(Interval)* can be proved by its solver. In other circumstances, the certificate indicates that the logic defined by the user is incomplete, w.r.t. the semantics and solvers that come with the functors. In this case, the certificate also indicates what hypotheses are missing for completeness; this may help users to define a more complete variant of their logic.

Logic functors offer basic bricks and a building rule to safely design new logics. For instance, in a recent application of LIS to geographical information system, a basic reasoning capability on locations was needed. The designer of the application, not a LIS or LogFun author, could build a relevant ad hoc logic safely and rapidly.

5.3 Abilis

Participants: Benjamin Sigonneau, Mireille Ducassé.

Abilis provides the LIS functionalities as a Web application. The advantage of a Web application is that users do not have to bother about set up, and we hope that this will foster the diffusion of logical information systems. We plan to publish a number of datasets, for example the collection of LIS publications, and to allow people to create their own datasets. Each dataset is defined by a logical context, and a set of users. Abilis is developed in the Ocsigen framework, based on a thin client – thick server architecture. Abilis is based on Camelis, which provides the LIS API, and where the graphical user interface is replaced by a XHTML Web interface.

Abilis improves Camelis on two aspects: multi-user access and visualization. Multi-user access is crucial in a Web application, and requires the management of users and access rights. Anonymous users can browse a context, while advanced users can also update, create and delete contexts. The development of multi-user access is led by Benjamin Sigonneau. Regarding visualization, users can create complex views of the extension (the query answers), reusing ideas and concepts from OLAP. A key characteristic of LIS, the consistency between the query, the navigation tree, and extension views, is retained. Users can partition the extension by selecting dimensions, project it by selecting a measure, and aggregate the results (e.g., count, sum). The resulting OLAP cubes can be displayed as tables, charts, or on a map.

5.4 Portalis

Participants: Yves Bekkers, Benjamin Sigonneau.

Portalis aims to produce *on the shelve software bricks* that can be used to construct web services built on top of our logical information systems. The purpose of this sub-project is to facilitate scientific popularization and industrial transfer of tools produced by the LIS team. The logical information systems are written in OCaml, and need adapters to more standard interfaces.

The first software brick is an HTTP server that wraps Camelis core functions and provides a dedicated API to access them. We call it the *LIS server.* Each Camelis function is called using an HTTP request. The answer is then marshalled in a dedicated XML format and sent back as an HTTP response.

A second brick implements a Java layer on top of the first brick. It is meant to easily build clients. It provides a Java function for each Camelis function. The Java function serializes its parameters, calls the corresponding HTTP request, and deserializes the XML answer in the form of Java objects.

On top of that, a portal offers features that are essential to several real-world use cases: using different Camelis contexts at once, handling users and their access rights. It also exports an HTTP API to allow people who would want to write clients in a language different from Java to benefit from these functions.

This year, the following advances were made on Portalis:

- The LIS server was completed to cover all of Camelis functions. The network query protocol is documented. The Java bindings were updated accordingly.
- The LIS server was made more scalable by using threading.
- Users are now handled in the portal.
- Two clients were written: an Android client (tested on a tablet) and an HTML5 client. Both clients serve as a demo of what can be done using Portalis bricks, and also as a test of the system.

5.5 Typed grammars

Participants: Denis Béchet [LINA-Nantes], Annie Foret [contact point].

A Pregroup ToolBox is under development on the gforge Inria as a collaborative work with LINA. It includes a generic pregroup parser (LINA) and grammar lexicon definitions and manipulation tools based on XML. An interface with Camelis has been developped (from Camelis to the Pregroup XML format, and the other way round). It has been used to define and experiment grammar prototypes for different natural languages.

5.6 SQUALL: a Semantic Query and Update High-Level Language

Participants: Sébastien Ferré.

SQUALL (Semantic Query and Update High-Level Language) is a controlled natural language (CNL) for querying and updating RDF graphs [6]. The main advantage of CNLs is to reconcile the high-level and natural syntax of natural languages, and the precision and lack of ambiguity of formal languages. SQUALL has a strong adequacy with RDF, and covers all constructs of SPARQL, and most constructs of SPARQL 1.1. Its syntax completely abstracts from low-level notions such as bindings and relational algebra. It features disjunction, negation, quantifiers, built-in predicates, aggregations with grouping, and n-ary relations through reification.

SQUALL is available as a Web application at `http://lisfs2008.irisa.fr/ocsigen/squall/` under two forms: one that translates SQUALL sentences to SPARQL, and another one that directly return query answers from a SPARQL endpoint.

5.7 PEW: Possible World Explorer

Participants: Sébastien Ferré, Sebastian Rudolph.

The Possible World Explorer (PEW) targets ontology designers, and aims to help them correct and complete their ontologies. It reuses the query-based faceted search principles of Sewelis for exploring the "possible worlds" (i.e., models) of an OWL ontology. Users are guided in the incremental construction of class expressions, such that only satisfiable classes are reachable. All classes made of qualified existential restrictions, nominals, intersections, unions, and atomic negations are reachable.

PEW not only supports the exploration of an ontology's possible worlds, but also supports its completion by the addition of axioms [8]. When a class is found satisfiable, and this contradicts domain knowledge (e.g., a man that is not a person), the undesirable possible worlds can be excluded ("pew pew!") by asserting an axiom saying that this class is unsatisfiable (e.g., every man is a person). This could be made a game, where the player would strive to exclude as many undesirable worlds as possible. The benefits are to complete the ontology with more knowledge, and therefore to improve its deduction power.

In addition to asserting negative axioms (about things that should not exist), PEW also allows for the definition of named classes (OWL equivalent class axioms), and for the creation of named individuals as instances of class expressions (OWL class assertion axioms).

6 New Results

6.1 Group Decision Support Systems

Participants: Mireille Ducassé, Peggy Cellier.

Group work represents a large amount of time in professional life while many people feel that much of that time is wasted. This amount of time is still increasing because problems are becoming more complex and are meant to be solved in a distributed way. Each involved person has a local and partial view of the problem, no one embraces the whole required knowledge. In this thread of research we develop decision processes taking benefits of Logical Information Systems capabilities.

Reasoning on multiple criteria is a key issue in group decision to take into account the multidimensional nature of real-world decision-making problems. In order to reduce the induced information overload, in multicriteria decision analysis, criteria are in general aggregated, in many cases by a simple discriminant function of the form of a weighted sum. It requires to, a priori and completely, elicit preferences of decision makers. That can be quite arbitrary. In everyday life, to reduce information overload people often use a heuristic, called *"Take-the-best"*: they take criteria in a predefined order, the first criterion which discriminates the alternatives at stake is used to make the decision [MGG10]. Although useful, the heuristic can be biased. We propose the Logical Multicriteria Sort process to support multicriteria sorting within islands of agreement. It therefore does not require a complete and consistent a priori set of preferences, but rather supports groups to quickly identify the criteria for which an agreement exists. The process can be seen as a generalization of Take-the-best. It also proposes to consider one criterion at a time but once a criterion has been found discriminating it is recorded, the process

[MGG10] J. N. Marewski, W. Gaissmaier, G. Gigerenzer, "Good judgments do not require complex cognition", *Cognitive Processing 11*, 2, 2010, p. 103–121.

is iterated and relevant criteria are logically combined. Hence, the biases of Take-the-best are reduced. The process is supported by Logical Information Systems, which give instantaneous feedbacks of each small decision and keep tracks of all of the decisions taken so far. The process is incremental, each step involves low information load. It guarantees some fairness because all considered alternatives are systematically analyzed along the selected criteria [1].

We also investigate how Logical Information Systems can help meeting organizers (*facilitators*) build on experience when preparing meetings. Thanks to Utilis [11], features of meetings similar to the one under construction are automatically suggested without having to ask for suggestions. Suggestions take into account the whole information about all the meetings already recorded in the system as well as facilitation knowledge. Usual techniques and processes that facilitators like to use are naturally suggested. An unusual technique is suggested for example if the facilitator enters a keyword that is a feature of that technique. Although this is still work in progress, we have already shown contributions that make believe that it is worth investigating further. The main one is that it builds on the facilitator very practice. Other important features are flexibility and adaptability [4].

6.2 Scalable Query-based Faceted Search on top of SPARQL Endpoints for Guided and Expressive Semantic Search

Participants: Joris Guyonvarc'h, Sébastien Ferré, Mireille Ducassé.

Because the Web of Documents is composed of unstructured pages that are not meaningful to machines, search in the Web of Documents is generally processed by keywords. However, because the Web of Data provides structured information, search in the Web of Data can be more precise. SPARQL is the standard query language for querying this structured information. SPARQL is expressive and its syntax is similar to SQL. However, casual users can not write SPARQL queries. Sewelis is a search system for the Web of Data that makes it possible to explore data progressively and more user-friendly than SPARQL. Sewelis guides the search with a query built incrementally because users only have to select query elements in order to complete the query. However, Sewelis does not scale to large datasets such as DBpedia, which is composed of about 2 billion triples. We have introduced Scalewelis as a search system for the Web of Data that is similar to Sewelis but scalable [9]. This is made possible by relying on efficient SPARQL engines to compute the suggested query elements at each navigation step. Moreover, Scalewelis is data independent because it connects to SPARQL endpoints. We took part in a challenge on DBpedia with Scalewelis [13]. We were able to answer 70 questions out of 99 with acceptable response times, recall 33%, and precision 33%.

6.3 Guided Composition of Bioinformatic Workflows with Logical Information Systems

Participants: Mouhamadou Ba, Sébastien Ferré, Mireille Ducassé.

In a number of domains, particularly in bioinformatics, there is a need for complex data analysis. For that issue, elementary data analysis operations called tasks are composed as workflows. The composition of tasks is however difficult due to the distributed and heterogeneous resources of bioinformatics. We presented a doctorial work [3] that addresses the composition of tasks using Logical Information Systems (LIS). LIS let users build complex queries and updates over semantic web data through guided navigation, suggesting relevant pieces and updates at each step. The objective is to use semantics to describe bioinformatic tasks and to adapt the guided approach of Sewelis, a LIS semantic web tool, to the composition of tasks. We aim at providing a tool that supports guided composition of semantic web services in bioinformatics, and that will support biologists in designing work ows for complex data analysis.

6.4 Representation of Complex Expressions in RDF

Participants: Sébastien Ferré.

Complex expressions, as used in mathematics and logics, account for a large part of human knowledge. It is therefore desirable to allow for their representation in RDF and for their exploration through semantic search. We have proposed [5][1] an RDF vocabulary that fulfills three objectives. The first objective is the accurate representation of expressions in standard RDF, so that expressive mathematical search is made possible. We here propose a syntactic extension of Turtle and SPARQL for the concise notation of such expressions. For example, the query `int(...?X ^ 2...,?X)` retrieves all integrals in x whose body contains the subexpression x^2. The second objective is the automated generation of expression labels that are close to usual mathematical notations (e.g., infix operators, symbols). The third objective is the compatibility with existing practice and legacy data in the Semantic Web community for the representation of expressions and structures (e.g., OWL/RDF, SPIN). We have illustrated the use of this vocabulary on mathematical search using SEWELIS, which shows a number of benefits compared to state-of-the-art in mathematical search.

6.5 SQUALL: A Controlled Natural Language as Expressive as SPARQL

Participants: Sébastien Ferré.

The Semantic Web is now made of billions of triples, which are available as Linked Open Data (LOD) or as RDF stores. The most common approach to access RDF datasets is through SPARQL, an expressive query language. However, SPARQL is difficult to learn for most users because it exhibits low-level notions of relational algebra such as union, filters, or grouping. SQUALL (Semantic Query and Update High-Level Language) is a high-level language for querying and updating an RDF dataset. It has a strong compliance with RDF, covers all features of SPARQL 1.1, and has a controlled natural language syntax that completely abstracts from low-level notions. SQUALL is available as two web services: one for translating a SQUALL sentence to a SPARQL query or update, and another for directly querying a SPARQL endpoint such as DBpedia. It has been evaluated on the questions of the QALD-2 challenge [6]. We also took part in the QALD-3 challenge on DBpedia with SQUALL [12]. We could reformulate in SQUALL all 99 questions, and obtained the highest scores, recall 88%, and precision 93%, while retaining the conciseness and readability of questions. However, the

[1]A long version of this poster is available at `http://hal.inria.fr/hal-00812197`.

reformulation requires the manual mapping for English words to URIs, what other participants of the challenge try to do automatically.

6.6 Categorial grammar hierarchies

Participants: Annie Foret.

The notion of k-valued categorial grammars in which every word is associated to at most k types is often used in the field of lexicalized grammars as a fruitful constraint for obtaining interesting properties like the existence of learning algorithms. This constraint is reasonable only when the classes of k-valued grammars correspond to a real hierarchy of generated languages. Such a hierarchy has been established earlier for the classical categorial grammars. The new contribution [2] studies an extension of Lambek grammars with respect to such hierarchies. Another interest of the extended calculus is to allow some parallels in grammar design (type assignments, acquisition methods) between both frameworks (pregroups and (L)). Those hierarchies are of particular interest for a LIS view of a grammar, which helps to browse, update, create and maintain or test related linguistic data.

6.7 Closed Sequential Pattern Mining under Multiple Constraints

Participants: Peggy Cellier, Nicolas Béchet [IUT Vannes], Thierry Charnois [University of Paris 13], Bruno Crémilleux [University of Caen], Solen Quiniou [University of Nantes].

We tackle the problem of sequential pattern extraction under multiple constraints and condensed representation [10, 11]. We proposed two algorithmes allowing to treat sequences of itemsets with several kinds of constraints. The first algorithm computes the closed sequential patterns under the gap constraint. The second one extracts sequential patterns of itemsets under several other constraints. In addition, we studied the problem of the closure for sequential patterns under constraints. A tool (sdmc) has been developped and is available at `https://sdmc.greyc.fr`.

6.8 Graph Mining Under Linguistic Constraints to Explore Large Texts

Participants: Peggy Cellier, Thierry Charnois [University of Caen], Dominique Legallois [University of Caen], Solen Quiniou [University of Nantes].

We continue the work to explore large texts by highlighting coherent sub-parts [8]. The exploration method relies on a graph representation of the text according to Hoey's linguistic model which allows the selection and the binding of adjacent and non-adjacent sentences. The main contribution of our work consists in proposing a method based on both Hoey's linguistic model and a special graph mining technique, called CoHoP mining, to extract coherent subparts of the graph representation of the text. We have conducted some experiments on several English texts showing the interest of the proposed approach.

6.9 A Case-Based Reasoning Approach for the Interactive Update of Objects in RDF(S) Bases

Participants: Alice Hermann, Mireille Ducassé, Sébastien Ferré, Jean Lieber [LORIA].

With existing tools, when creating a new object in the Semantic Web, users benefit neither from existing objects and their properties, nor from the already known properties of the new object. We have proposed UTILIS [7], an interactive process to help users add new objects. While creating a new object, relaxation rules are applied to its current description to find similar objects, whose properties serve as suggestions to expand the description. A user study conducted on a group of master students shows that students, even the ones disconcerted by the unconventional interface, used UTILIS suggestions. In most cases, they could find the searched element in the first three sets of properties of similar objects. Moreover, with UTILIS users did not create any duplicate whereas with the other tool used in the study more than half of them did. Another user study was conducted with a visiting student from India, Preeti Dahiya. She intensely used UTILIS in SEWELIS for the annotation of the 573 episodes the Hindi serial "Sasural Genda Phool". After a first phase for learning the tool and defining a schema for describing episodes and related people, Preeti produced around 18,000 triples and 8,000 objects in about 2-3 weeks. The time for describing an episode went decreasing from 3 hours for the 1st episode to less than 10 minutes after 200 episodes and a bug fix.

7 Contracts and Grants with Industry

7.1 Portalis: funding for maturation (FEDER Région Bretagne)

Participants: Yves Bekkers, Benjamin Sigonneau, Sébastien Ferré.

As part of the implementation of the pole "Regional Competitiveness and Employment (2007-2013)" in Brittany, we obtained a funding from FEDER and Région Bretagne for an engineer for a year to participate in CAMELIS transfer to industry. Benjamin Sigonneau has been appointed on this founding. It started in October 2012.

Presently, we have discussions with Mediadone, a company specializing in processing, indexing and image enhancement. Mediadone provides tools for interactive and enriched WebTV. The company is interested in using Portalis bricks to build an intelligent navigation tool based on the use of Camelis. Mediadone has already acquired a license for the commercial exploitation of CAMELIS.

8 Other Grants and Activities

8.1 International Collaborations

- In 2013-2014, during a "délégation CNRS", Annie Foret is a visitor of *The Institute for Language, Cognition and Computation (ILCC)* in Edinburgh. The visit is hosted by Mark Steedman's group, working on Categorial Grammars. She has given a talk in October

2013 at the ILCC seminar : "On some classes of Categorial Grammars and on Logical Information Systems".

- Sébastien Ferré is a member of the management committee of the COST action MUMIA (IC1002 - Multilingual and multifaceted interactive information access). MUMIA aims to coordinate collaboration between the following disciplines: machine translation, information retrieval, and faceted search. The objectives of the action is to foster research and development for next generation search technologies. The domain of patent search has been selected as a common use case, as it provides highly sophisticated and information intensive search tasks that have significant economic ramifications.

  In 2013, he gave a presentation at the Working Group, which was co-located with the conference IRFC in Limassol, Cyprus. The topic of the presentation was the work of Joris Guyonvarc'h showing how query-based faceted search (SEWELIS) can scale to large RDF datasets like DBpedia.

- The LIS team received the visit of Sergei Kuznetsov (High School of Economic, Moscow) on july 8th. He gave a conference on “Pattern structures for knowledge discovery” at IRISA.

8.2 National Collaborations

- Annie Foret is a member of a group working with "The French Cour de Cassation". This multi-disciplinary project on "Développement d'un prototype de logiciel d'aide à la prise de décision lors de la rédaction de jugements" has received funds from ENS CHRC (Collèg de Recherche Hubert Curien). The proposal was submitted by François Schwarzentruber at ENS Rennes.

  Annie Foret is an external collaborator of LINA (research lab. Nantes), in TALN team (Natural Language Processing), and member of “Agence Universitaire de la Francophonie” (AUF) , LTT network on “Lexicologie, terminologie et traduction”. Annie Foret is member of ATALA (Association pour le Traitement automatique des Langues), and of SIF (Société Informatique de France).

- Peggy Cellier collaborates with the members of the ANR project HYBRID[2] on the part about data mining for natural language processing.

- Pierre Allard, a former PhD student working at Uneek (Nantes), gave a conference to ESIR students about data-mining.

9 Dissemination

9.1 Scientific Responsabilities

- Olivier Ridoux, Annie Foret, and Sébastien Ferré are members of the committee of the DKM scientific department (Data and Knowledge Management) at IRISA.

[2]http://hybride.loria.fr

- Annie Foret has been elected as a member of the scientific committee of ISTIC-Rennes1. She is a member of the committee "Développement Durable (Sustainable development)"
- Mireille Ducassé has served in the program committee of GDN 2014, international conference on Group Decision and Negotiation, Toulouse. She served in two PhD defense committees: reviewer for Azzeddine Amiar, Université de Grenoble, Lydie du Bousquet and Yliès Falcone, PhD co-supervisors, november 2013; examiner for Aymeric Hervieu, Université de Rennes 1, Benoit Baudry and Arnaud Gotlieb, PhD co-supervisors, december 2013. She has been a member of 3 recruitment committees ("comité de sélection") in computer science at university of Rennes 1, ENSSAT Lannion and INSA Rennes.
- Olivier Ridoux has served as a member of two PhD committees (was President once). He is a member of the "Conseil de laboratoire" at IRISA.
- Sébastien Ferré and Peggy Cellier are members of the organization committee of EGC 2014 to be held in Rennes in January 2014. The presidents of the committee are Arnaud Martin and René Quiniou.
- Sébastien Ferré is a member of the Editorial Board of the International Conference on Formal Concept Analysis (ICFCA). He was also in 2013 a member of the program committee of: the conference IRFC (co-organized by MUMIA), the conference CLA (Concept Lattices and Applications), the FCA-related workshops FCA4AI and FCAIR, the workshop GKR@IJCAI, and the PhD symposium of ESWC. Finally, he served as an external reviewer for the journal SOSYM.

  Sébastien is a supervisor of the PhD of Mouhamadou Ba. He also served as an examiner in the PhD defense committee (Rennes 1 Lannion, December 12th) of Katia ABBACI on "Contribution à l'interrogation flexible et personnalisée d'objets complexes modélisés par des graphes". He served as a local member of the selection committee for an associate professor position at ENSSAT Lannion (ref. MCF1720), and as an external member of the selection committee for an associate professor position at the University of Vannes (ref. MCF0134).
- Peggy Cellier was one of the two program chairs for ICFCA 2013 (Int. Conf. on Formal Concept Analysis). She has also served in the program committee of SEKE 2013 (Software Engineering ans Knowledge Engineering) and ICCS 2013 (Int. Conf. on Conceptual Structures) and as an external reviewer for the doctoral symposium of ESWC 2013 (Extended Semantic Web Conference). Peggy is also a supervisor of the PhD of Soda Marème Cissé.
- Annie Foret has been a *program committee* member of the *Formal Grammar* 2013 International Conference. She has been a program committee member of the *Mathematics of Linguistics* (MOL) 2013 International Conference. She belongs to the program committee of next *Formal Grammar* international conference. Annie Foret has been a *thesis reviewer (rapporteur)* and jury member for a PHD in Bordeaux in 2013, by Noémie-Fleur Sandillon-Rezer on "Learning categorial grammars : tree transducers and clustering of grammatical inference". She has been a *thesis reviewer (rapporteur)* and jury member for

a PHD in Nancy in 2013, by Florent Pompigne on "Modélisation logique de la langue et Grammaires Catégorielles Abstraites" She is member of a thesis committee for Ophélie Lacroix in Nantes (annual report before the PHD).

9.2 Involvement in the Scientific Community

- Mireille Ducassé has given an invited talk at the International Conference on Communication Systems, October 2013, organized by B.K. Birla Institute of Engineering and Technology, Pilani, Inde, “Fair and Fast Convergence on Islands of Agreement in Multi-criteria Group Decision Making by Logical Navigation”.

- Sébastien Ferré has given an invited talk at the International Conference on Formal Concept Analysis, organized by Bernhard Ganter at TU Dresden: “Scaling Conceptual Navigation in Expressivity, Usability, and Efficiency”.

- Olivier Ridoux organized with the University Library an exhibition of scientific comic books, mangas and graphic novels. He is also active in the development of the "Maison pour la science en Bretagne" for the scientific training of primary and secondary teachers in Brittany.

- Mouhamadou Ba and Sébastien Ferré took part in the ReNaBiGo seminar on workflows, organized by the GenOuest platform and closely related to the PhD subject of Ba.

- Soda Marème Cissé took part in the foundation meeting of GDR (Research Group) MAGIS (Methods and Application for Geomatics and Spatial Information) organised in Lyon where different thematics on geographic information (Transport and Mobility, Environments, Uses, etc.) have been presented. Future work and future collaborations between thematics have been proposed during these days.

9.3 Teaching

- Mouhamadou Ba teaches algorithms and Java at Licence 1 level.

- Yves Bekkers teaches programming languages: functional programming, logic programming (Prolog, lambdaProlog), artificial intelligence, relational databases, XML, new technologies for programming distributed applications on the Web. After being a specialist of logic programming, his current principal interest is teaching distributed application design using tools based on model technologies (Object programming, XML, SGBDR) which allows building applications from one need to the other using numerous bridges allowing passing automatically from one model to another.

- Peggy Cellier is a member of the “Conseil de composante IRISA/INSA”. At Insa, she teaches database and script languages at licence level; symbolic data mining and formal methods for software engineering at Master level. Since September, 2013 she is responsible of the internships of computer science students.

- Soda Marème Cissé teaches office automation and programming in Java at Licence 1 level of University Rennes 1. She also teaches programming in JavaScript and Java at Licence 3 Miage level of University Rennes1.
- Mireille Ducassé is the director of international relations of the INSA of Rennes since december 2010. As such, she is a member of the direction of the Insa of Rennes.

  She taught two lectures to bachelor students at B.K. Birla Institute of Engineering and Technology, India, "Logical Information Systems, Formal and Logical Concept Analysis", October 2013.

  At Insa, she is responsible of three courses, taught in English: *Formal Methods for Software Engineering* (with the "B formal method") and *Constraint Programming* at Master 1 level, as well as *Participatory Design* at Master 2 level.
- Sébastien Ferré teaches symbolic data mining and compilation at the master level. He also teaches formal methods for programming and software engineering at the license level. He started a course on the Semantic Web for master-level students in MIAGE. He is vice-director of the MIAGE at ISTIC, and became in charge of Master 1 Internship, after Annie Foret went to Edinburgh.
- Annie Foret teaches university courses including formal logic, functional programming, and databases. She has been in charge of Master 1 Internship until August.
- Olivier Ridoux teaches data-bases, algorithmics, the theory of formal languages and compilation in the engineering school ESIR. He is also in charge of the innovation training in the school, in which he also teaches sustainable developpement w.r.t. IT, and disruptive innovation (à la Clayton Christensen) w.r.t. scientific revolution (à la Thomas Kuhn). He also teaches logic and constraint programming at the Master level, and an introduction to the principles of IT systems at the Bachelor level. He is also in charge of the institutional communication of ESIR.

10 Bibliography

Major publications by the team in recent years

[1] D. Béchet, A. Foret, "A Pregroup Toolbox for Parsing and Building Grammars of Natural Languages", *Linguistic Analysis Journal 36*, 2010.

[2] P. Cellier, S. Ferré, O. Ridoux, M. Ducassé, "A Parameterized Algorithm to Explore Formal Contexts with a Taxonomy", *Int. J. Foundations of Computer Science (IJFCS) 19*, 2, 2008, p. 319–343.

[3] M. Ducassé, S. Ferré, "Fair(er) and (almost) serene committee meetings with Logical and Formal Concept Analysis", *in: Proceedings of the International Conference on Conceptual Structures*, P. Eklund, O. Haemmerlé (editors), Springer-Verlag, July 2008. Lecture Notes in Artificial Intelligence 5113.

[4] S. Ferré, A. Hermann, "Reconciling faceted search and query languages for the Semantic Web", *Int. J. Metadata, Semantics and Ontologies 7*, 1, 2012, p. 37–54.

[5] S. Ferré, R. D. King, "A dichotomic search algorithm for mining and learning in domain-specific logics", *Fundamenta Informaticae – Special Issue on Advances in Mining Graphs, Trees and Sequences 66*, 1-2, 2005, p. 1–32.

[6] S. Ferré, O. Ridoux, "A Framework for Developing Embeddable Customized Logics", *in: Int. Work. Logic-based Program Synthesis and Transformation*, A. Pettorossi (editor), *LNCS 2372*, Springer, p. 191–215, 2002.

[7] S. Ferré, O. Ridoux, "An Introduction to Logical Information Systems", *Information Processing & Management 40*, 3, 2004, p. 383–419.

[8] S. Ferré, S. Rudolph, "Advocatus Diaboli - Exploratory Enrichment of Ontologies with Negative Constraints", *in: Int. Conf. Knowledge Engineering and Knowledge Management (EKAW)*, A. ten Teije et al. (editor), *LNAI 7603*, Springer, p. 42–56, 2012.

[9] S. Ferré, "Camelis: a logical information system to organize and browse a collection of documents", *Int. J. General Systems 38*, 4, 2009.

[10] A. Foret, "A modular and parameterized presentation of pregroup calculus", *Information and Computation Journal 208*, 5, may 2010, p. 395–604.

[11] A. Hermann, S. Ferré, M. Ducassé, "An Interactive Guidance Process Supporting Consistent Updates of RDFS Graphs", *in: Int. Conf. Knowledge Engineering and Knowledge Management (EKAW)*, A. ten Teije et al. (editor), *LNAI 7603*, Springer, p. 185–199, 2012.

Articles in referred journals and book chapters

[1] M. Ducassé, P. Cellier, "Fair and Fast Convergence on Islands of Agreement in Multicriteria Group Decision Making by Logical Navigation", *Group Decision and Negotiation*, To appear.

[2] A. Foret, *CATEGORIES AND TYPES IN LOGIC, LANGUAGE AND PHYSICS - Festschrift on the occasion of Jim Lambek's 90th birthday*, edition Casadio C., Coecke B., Moortgat M., Scott P. (editors), in print, Springer, Lecture Notes in Computer Science, New York, 2013, ch. On Associative Lambek Calculus extended with basic proper axioms.

Publications in Conferences and Workshops

[3] M. Ba, "Guided Composition of Tasks with Logical Information Systems - Application to Data Analysis Workflows in Bioinformatics", *in: Extended Semantic Web Conf.*, P. Cimiano, O. Corcho, V. Presutti, L. Hollink, S. Rudolph (editors), *LNCS 7882*, Springer, p. 661–665, 2013.

[4] M. Ducassé, "Helping Facilitators Build on Experience When Preparing Meetings With Logical Information Systems", *in: Group Decision and Negotiation Conference*, B. Martinovski (editor), Department of Computer and Systems Sciences, Stockholm University, p. 139–143, 2013. Extended abstract.

[5] S. Ferré, "Representation of Complex Expressions in RDF", *in: Extended Semantic Web Conf. (ESWC Satellite Events)*, P. Cimiano, M. Fernández, V. Lopez, S. Schlobach, J. Völker (editors), *LNCS 7955*, Springer, p. 273–274, 2013.

[6] S. Ferré, "SQUALL: A Controlled Natural Language as Expressive as SPARQL 1.1", *in: Int. Conf. Applications of Natural Language to Information System (NLDB)*, E. Métais, F. Meziane, M. Saraee, V. Sugumaran, S. Vadera (editors), *LNCS 7934*, Springer, p. 114–125, 2013.

[7] A. Hermann, M. Ducassé, S. Ferré, J. Lieber, "Une approche fondée sur le raisonnement à partir de cas pour la mise à jour interactive d'objets du Web sémantique", *in: 21ème atelier Français de Raisonnement à Partir de Cas (RàPC)*, Lille, France, 2013, `http://hal.inria.fr/hal-00910294`.

[8] S. Quiniou, P. Cellier, T. Charnois, D. Legallois, "Graph Mining under Linguistic Constraints to Explore Large Texts", *in: Complementary Proceedings of the 14th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing'13)*, 2013.

Internal Reports

[9] J. Guyonvarch, S. Ferre, M. Ducassé, "Scalable Query-based Faceted Search on top of SPARQL Endpoints for Guided and Expressive Semantic Search", *Research report number PI-2009*, LIS - IRISA, October 2013, `http://hal.inria.fr/hal-00868460`.

Miscellaneous

[10] N. Béchet, P. Cellier, T. Charnois, B. Crémilleux, "Extraction de motifs séquentiels sous contraintes multiples", *in: Extraction et gestion des connaissances (EGC'2013) (Session poster)*, A. F. Gelbukh (editor), 2013.

[11] N. Béchet, P. Cellier, T. Charnois, B. Crémilleux, "SDMC : un outil en ligne d'extraction de motifs se quentiels pour la fouille de textes", *in: Extraction et gestion des connaissances (EGC'2013) (Session démo)*, A. F. Gelbukh (editor), 2013.

[12] S. Ferré, "squall2sparql: a Translator from Controlled English to Full SPARQL 1.1", Work. Multilingual Question Answering over Linked Data (QALD-3), 2013, See Online Working Notes at `www.clef2013.org`.

[13] J. Guyonvarc'h, S. Ferré, "Scalewelis: a Scalable Query-based Faceted Search System on Top of SPARQL Endpoints", Work. Multilingual Question Answering over Linked Data (QALD-3), 2013, See Online Working Notes at `www.clef2013.org`.