I.J. Intelligent Systems and Applications, 2013, 03, 13-22
Published Online February 2013 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2013.03.02

Using Description Logics to specify a Document Synthesis System

Souleymane KOUSSOUBE Dr.**, Roger NOUSSI** Dr.**, Balira O. KONFE**, Dr., Prof

Laboratoire Africain d'Informatique et de Math ématiques appliqu ées (LAIMA), Institut Africain d'Informatique (IAI),
BP: 2263 Libreville Gabon
Email: skoussoube@gmail.com, roger_noussi@yahoo.fr, obalira@gmail.com

Abstract — this paper deals with an automatic document's synthesis system. Our approach is based on the prior formal description of the semantics of the main elements (document, reader and his request) in the synthesis system. In this approach, semantic capture is based on ontology definition that is specified formally using Description Logics (DL). The DL inference techniques associated to production rules are then used to compute a document synthesis. Moreover, DL inference techniques are used to reason about each component.

***Index Terms* —** Description Logics, Ontology, Ontology Matching, Document Synthesis

I. Introduction

Large amount of textual documents are now available on electronic media. Moreover, the development of multimedia technologies leads to the improvement of functions used on these documents [1, 2, 3].

Generally, the new systems on these media provide support to text assimilation. They take into account the content and the meaning of the text. Ongoing works to build such systems follow two main directions.

On one hand, we have systems dealing with texts that have specific format, mostly tabular format [4].

The specificity of the format facilitates the capture of the semantic of the text. An extended presentation of works following this approach during the last decade can be found in [5].

In the other hand, there are works devoted to linguistic analysis and statistic techniques, which produce an automatic abstract of a text [6]. Statistic techniques are used here to have an insight on the semantic of the document.

However, these methods do not use a uniform and unique formalism to describe the different components of the synthesis process. So any inference mechanism taking into account all the components of the synthesis systemis proposed.

Let us note that, the main challenge in the document's synthesis is to describe suitably the semantic of the document [7] and the semantic of the reader's request.

This paper deals with the development of an automatic textual document synthesis system. It proposes functions to build personalized views of a document. It aims to assist the reader by taking into account: his profile, his acquisition constraints (time, volume of information, etc...), his synthesis request, and the history of his work on the document.

Our approach is based on the prior formal description of the semantics of the main elements (document, reader and his request) in the synthesis system. In this approach, semantic capture is based on ontology definition that is specified formally using Description Logics (DL). The DL inference techniques associated to production rules are then used to compute a document synthesis. DL inference techniques are used to reason about each component of the system in one hand, and to achieve a formal reasoning mechanism using both, the documents ontology, the reader profile and the reading projects ontology.

To tackle the problem of the document's semantic; we assume that the document is related to a particular domain. We use the ontology of this domain as a support to capture the semantic of the document. The focus on a particular domain also gives the possibility to take advantage of available expertise.

The remainder of this paper is organized as follows: we first present Ontology and Description Logics.

The section 3 is devoted to the specification of the document domain ontology by using DL language. In section 4, we present the reader: his profile, his request and the reading project ontology. In the last section, we describe our synthesis system including, the matching component that translates a reader request into a reading project and finally, the synthesis engine.

II. Ontologies and Description Logics

Semantic capture and its efficient use for document synthesis are the main concerns of this paper. In this section, we give a short presentation of ontology and DL that are used for this purpose. Ontology provides a shared conceptualization of a domain. DL [8, 9, 10] allows to formally expressing the knowledge represented graphically in semantic networks.

2.1 Ontology

2.1.1 Definition

Ontology is an explicit specification of a conceptualization [11, 12]: A conceptualization is an abstract, simplified view of a domain. That is, ontology is mainly a description of the concepts (definitions) and relationships existing within a domain. Ontology are designed for the purpose of enabling knowledge sharing and reuse.

Dentitions consist in associating the names of entities within a domain (e.g., classes, relations, functions, or other objects) with human-readable texts. These texts describe the meaning of the names. Dentitions also include formal axioms constraining the interpretation of well-formed terms.

Formally, ontology is the statement of a logical theory. Pragmatically, a common ontology de.nes the vocabulary with which queries and assertions are exchanged among agents. Ontological commitments are agreements to use the shared vocabulary in a coherent and consistent way.

A commitment to a common ontology is a guarantee of consistency, but not of completeness, with respect to queries and assertions using the vocabulary de.ned in the ontology.

Ontology are often equated with taxonomic hierarchies of classes, and do not add any knowledge about the world. However, as said above, to specify a conceptualization, one needs to state axioms that do constrain the possible interpretations for the defined terms.

2.1.2 The use ontology in our synthesis system

The semantic capture of the information needed for our synthesis system is based on the prior construction of the following ontology:

1. The document ontology, which provides a formal and sound description of the document's content. This ontology is developed in next section.

2. The reader profile ontology. This ontology describes the different classes of potential readers (section 4.1).

3. The reading project ontology, which allows a formal and standard description of the reader's request. Section 4.2 is devoted to this ontology.

For each ontology, we first propose the settings that can be seen as meta-ontology (high level concepts and relationships).

2.2 Description Logics and Knowledge-Based systems

Description Logics (DL) [8] are knowledge representation formalisms used to describe concepts in a given domain. A knowledge base (**KB**) described in DL has two components, the Tbox and the Abox. The TBox introduces the terminology, i.e., the vocabulary of an application domain, while the Abox contains assertions about named individuals in terms of this vocabulary. The vocabulary consists of concepts, denoting sets of individuals (identified objects of the domain), and roles (binary relationships between individuals). In addition to atomic concepts and roles, all DL systems allow building complex descriptions of concepts and roles. Depending on provided operators, there are several DL languages, the **Attributive Language (*AL*)** being the minimal one. We summarize here the syntax and the semantics of some DL languages.

2.2.1 Syntax and semantics of DL languages

Concepts and roles are inductively de.ned from a set N_C of concepts names (atomic concepts), a set N_R of roles names (atomic roles) and a set of operators.

In the following, unless otherwise stated, A and B are elements of N_C; *r* and *s* are components of N_R;

C and *D* are concepts descriptions and n is a positive integer.

The minimal language ***AL*** contains the atomic concepts, the universal concept, the bottom concept, atomic negation, intersection, value restriction and limited existential quantification.

$$AL = \{A, \mathrm{T}, \bot, \neg A, C \bigcap D, \forall r \bullet C, \exists r \bullet T\}$$

Extending ***AL*** by any subset of the constructor in the table below, yields a particular ***AL***-language. Each ***AL***-language is denoted by a string of the form:

$$AL[U][\varepsilon][N][C][Q][I]$$

Hence, *ALCQI* is the language obtained from AL by adding full negation(*C*), qualified number restriction (*Q*) and Inverse of role (*I*).

In order to define a formal semantics of concepts descriptions, we consider an interpretation *I* that consists of a non-empty set Δ^I (the domain of the interpretation) and an interpretation function $\bullet^I$, which assigns to every atomic concept *A*, a set $A^I \subseteq \Delta^I$ and to every atomic role *r* a binary relation $r^I \subseteq \Delta^I \times \Delta^I$.

For example, we have the following interpretations:

- T is the whole domain, i.e. all the individuals in the domain.
- $\forall r \bullet C$ is the set of individuals who are related through r only to individuals satisfying C.
- $\exists r \bullet T$ is the set of individuals related through *r* with other individuals of the domain.
- $\leq nr \bullet C$ is the set of individuals who are related through *r* to most n individuals satisfying *C*.

The following table summarizes the syntax and the semantics of DL.

Table 1: Syntax and semantic of DL

Syntax	Definition	Semantic	Symbol
T	Universal concept	$\mathrm{T}^I = \Delta^I$	*AL*
$\perp$	Bottom concept	$\perp^I = \emptyset$	*AL*
$\neg A$	Atomic negation	$(\neg A)^I = \Delta^I \setminus A^I$	*AL*
u D	intersection	$(C \cap D)^I = C^I \cap D^I$	*AL*
$\forall r \bullet C$	Value restriction	$(\forall r \bullet C)^I = \{a \in \Delta^I \mid \forall b \bullet$ $(a,b) \in r^I \Rightarrow b \in C^I \}$	*AL*
$\exists r \bullet T$ or $\exists r$	Limited existential quantification	$(\exists r \bullet \mathrm{T})^I = \{a \in \Delta^I \mid \forall b \bullet$ $(a,b) \in r^I \}$	*AL*
$\exists r \bullet C$	Full existential quantification	$(\exists r \bullet C)^I = \{a \in \Delta^I \mid \forall b \bullet$ $(a,b) \in r^I \wedge b \in C^I \}$	ε
t	union	$(C \cup D)^I = C^I \cup D^I$	*U*
$\neg C$	Full negation	$(\neg C)^I = \Delta^I \setminus C^I$	*C*
$\leq nr \bullet C$	At most qualified number restriction	$(\leq nr \bullet C)^I = \{a \in \Delta^I \mid \lvert\{b \mid$ $(a,b) \in r^I \}\rvert \leq n \wedge b \in C^I \}$	*Q*
$\geq nr \bullet C$	At least qualified number restriction	$(\geq nr \bullet C)^I = \{a \in \Delta^I \mid \lvert\{b \mid$ $(a,b) \in r^I \}\rvert \geq n \wedge b \in C^I \}$	*Q*
$\leq nr$	At most unqualified number restriction	$(\leq nr)^I = \{a \in \Delta^I \mid \lvert\{b \mid$ $(a,b) \in r^I \}\rvert \leq n \}$	*N*
$\geq nr$	At least unqualified number restriction	$(\geq nr)^I = \{a \in \Delta^I \mid \lvert\{b \mid$ $(a,b) \in r^I \}\rvert \geq n \}$	*N*
r^-	Inverse role	$(r^-)^I = \{(b,a) \in \Delta^I \times \Delta^I \mid$ $(a,b) \in r^I \}$	*I*

2.2.2 Inference techniques in Description Logics

t the terminological level, there are 4 inference operators:

- **Satisfiability**: A concept *C* of a terminology *T* is satisfiable if and only if there exists a model *I* of *T* such that $C^I \neq \{\ \}$
- **Subsumption**: A concept *C* is subsumed by a concept D (*C* v D) for a terminology *T* if and only if $C^I \subseteq D^I$ for any model *I* of *T*.
- **Equivalence**: A concept *C* is equivalent to a concept *D* ($C \equiv D$) for a terminology *T* if and only if $C^I = D^I$ for each model *I* of *T*.
- **Disjunction**: Concepts *C* and *D* are disjoined by report/ratio terminology *T* if and only if $C^I \setminus D^I$ = { }; for each model *I* of *T*.

The 4 types of problems can be brought back to problems of subsumption or satisfiability. Consequently the design of inference engines requires, very often only one type of algorithm.

At factual level, there are also 4 inference operators:

- **Coherence**: An Abox A is coherent with reference to (wrt) a Tbox *T* if and only if there exists a model I of *A* and *T*.
- **Checking of authority**: To check by inference if an assertion *C (a)* is true for any model *I* of an Abox *A* and a Tbox *T* .
- **Checking of role**: To check by inference if an assertion *r(a; b)* is true for any model *I* of an Abox *A* and a Tbox *T* .
- **Recovery problem**: For an Abox *A*, a concept *C* of a terminology *T* , infer the individuals a such that *C (a)*

One can find non standard inference operators in [13]

III. The Document Domain Ontology

3.1 The document semantic and the document ontology

In her work on small advertisements sub-language, M. COURANT [14] shows that, the mastery of semantics makes it possible to communicate a right message even if some of the linguistic constraints are dropped. It appears that an advertisement can undergo significant variations of form while preserving the integrity of the message conveyed. Thus, some lexemes are considered as noise and are not taken into account in the processing of the advertisement.

In the same way, non compliance with the syntactic rules has limited influence on the comprehension of the advertisement.

Our synthesis system aims to produce a reduced version of a document while preserving the initial message. One can imagine that the reverse approach is used to make the speech more expressive. In both cases, it is fundamental to preserve the integrity of the speech we want to transmit. This preservation is based on the mastery of document semantic.

We use the following sub-language characterization [1, 15] as a methodological framework to capture the semantic of a document:

1. Restricted domain of reference: The set of objects and relations to which the linguistic expressions are associated, is assumed to be restricted.

2. Restricted targets: The linguistic exchanges are strongly related to specialized targets.

3. Restricted users community: This community is composed of users sharing specialized knowledge's. The expression of this knowledge is strongly sullied with familiar uses.

4. Restricted communication mode: The expression is constrained with material or technical conditions such as the limitation of the number of lines or words etc.

Generally, a document deals with a precise domain. Thus, the .first item of the sub-language characterization is fulfilled as soon as a document is chosen. The second and the fourth items can be link to the reading project. The third item refers to the reader and leads to the description of his profile and history.

Our semantic capture is based on the formal specification of the domain and document ontology, the reader profile ontology and the reading project ontology.

As stated above, a document is supposed to deal with a precise domain. The set of concepts that are explicitly used in the document are a subset of the concepts of the domain.

In the context of the synthesis, taxonomic relationships and aggregation relationships are of great interest.

It is also useful to clearly identify the document. The following figure gives the major high level components of the document semantic (document meta-ontology).

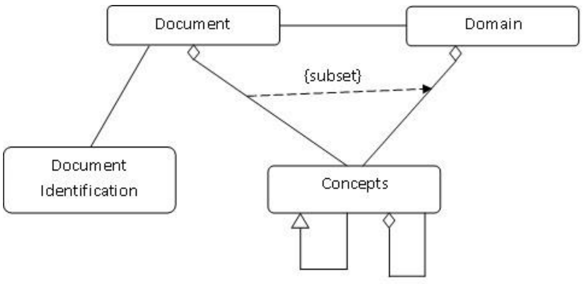


Fig. 1 : Major component of Document meta-ontology

Particular document ontology can be seen as an instance of the above meta-ontology. It can be expressed by UML-like representation diagram or, more formally, in a DL language.

3.2 Example

In this section, we illustrate the document ontology by using an extract of the semantic representation of a document containing **a course on problems solving techniques in Computer Science**.

The .figure 2 below is the UML-like representation diagram of this document.

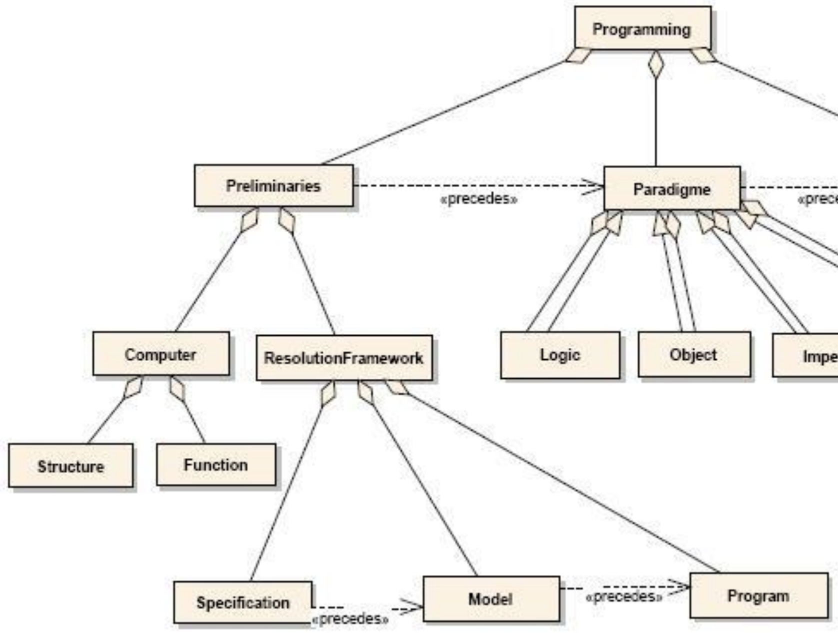


Fig. 2 : UML-like representation diagram

To obtain the formal ontology, the translation of the UML-like representation into the DL notation can be done using the translation rules presented in [16].

However these rules cannot be applied completely because they would introduce a different role for each aggregation. In our case, all the aggregation relationship between concepts has the same semantic. So a unique role must be used to represent this relationship.

The transcription of the example above is given below:

1. Programming v9IsComposedOf • Paradigm u 9IsComposedOf • Language u9IsComposedOf • Preliminary.

The expression above states the fact that the programming concept is composed of preliminary, language and paradigm.

2. Preliminaries v9IsComposedOf • Computer u 9IsComposedOf • resolutionFramework u9IsComposedOf • Paradigm.

The expression above speci.es the fact that concept Preliminaries is composed of computer, resolution framework and paradigm.

3. Paradigm v9IsComposedOf • logic u 9IsComposedOf • functional u9IsComposedOf • object u9IsComposedOf • imperative. u9IsComposedOf • language.

The relation above expresses the fact that a paradigm is composed of logical, functional, object or imperative approach.

4. logic v Paradigm,
5. object v Paradigm,
6. functional v Paradigm,
7. imperative v Paradigm

The subsumption relationships above specify that logic, object, functional and imperative are sort of paradigm.

In the next section, we provide materials for the formal specification of the readers' profiles and they request.

IV. The Readers

4.1 Readers profiles

4.1.1 The Framework

In order to produce accurate results for synthesis request, it is necessary to classify the readers. This classification takes into account their knowledge of the domain's topic and their contextual objectives. In the following, we provide a framework for the formal specification of reader profiles using a DL language.

Examples of atomic profiles are: *Student, Data Processing Specialist, Journalist, And Scientific Chronicler.*

The Tbox of the reader profiles will be built as follows:

- Each atomic profile is represented as a concept in the Tbox,
- Any subset of domain concepts can be used to characterize of a reader profile. Let K be a set describing the knowledge levels and C, a concept of the document domain. The expression $\exists k \bullet C$ where $k \in K$; defines the profile of a reader having the level k of knowledge of the concept C: For example,

  K = {*Expert, Specialist, Has Good Knowledge, Beginner, Has No Knowledge, Not Initiated*}

Table 2: syntax and interpretation of profile expressions

Syntax	Interpretation
$\boldsymbol{P} \equiv P_1 \cup P_2$	A reader has the profile P if he has P_1 or P_2 or the both
$\boldsymbol{P} \equiv P_1 \cap P_2$	A reader has the profile P if he has the profile P_1 and the profile P_2
$\boldsymbol{P} \equiv \neg P_1$	A reader has the profile P if he does not have the profile P_1

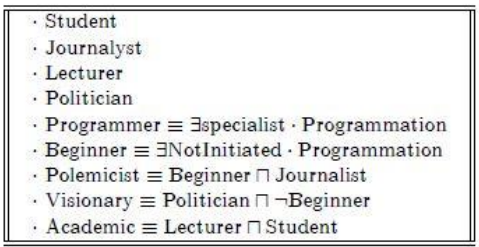
- Composite profiles are built with the standard DL operators and with the following interpretation :

  Given two profiles P_1 and P_2,
- The interpretation of subsumption between profiles concept noted $P_1 \subseteq P_2$ has the following interpretation: the profile P_1 is more specialized or is as specialized as the profile P_2.

The subsumption relationship also allows defining in uncompleted way, a profile through the assertion $P \subseteq P_1$ or $P_1 \subseteq P$.

4.1.2 Example of reader profile TBOX

Given the atomic profiles Student, Journalist, Lecturer, Politician; we have a following Tbox:



- Student
- Journalyst
- Lecturer
- Politician
- Programmer ≡ ∃specialist · Programmation
- Beginner ≡ ∃NotInitiated · Programmation
- Polemicist ≡ Beginner ⊓ Journalist
- Visionary ≡ Politician ⊓ ¬Beginner
- Academic ≡ Lecturer ⊓ Student

Fig. 3: Example of a profile TBOX

4.2 The Reading Project (RP) Ontology

The reader request is expressed in a free format (natural language): The analysis step translates this request into an expression of the reading project ontology. Some items of the initial request can be considered as "noise" so, will not be taken into account during this step.

To represent, the reading project in DL Language, we use the following notations. We assume that the reading project consists in searching, learning, summarizing, reading particular concepts, solving exercises etc.

The actions (search, learn, summarize, read, solve ...) are roles applied to the concepts defined in the representation of the document.

Let A be a set of reading actions defined by:

A = {*search; learn; summarize; read; SolveExercises; etc*}

Let C be a set of domain concept like

C = {*Programming; Paradigm; Language; Preliminary; etc*}

An atomic Reading Project (*RP*) is represented by the DL expression:

$$RP \equiv \exists a \bullet c$$

where $a \in A$ and $c \in C$.

Composite reading project are built with the standard DL operators and with the following interpretation:

Given two reading projects RP_1 and RP_2

Tab 3: Syntax and interpretation of reading project expressions

Syntax	Interpretation
$\boldsymbol{RP} \equiv RP_1 \cup RP_2$	Any synthesis of RP_1, or RP_2; or of the both, is a synthesis of RP
$\boldsymbol{RP} \equiv RP_1 \cap \mathrm{RP}_2$	A synthesis of RP must be a synthesis of RP_1 and RP_2
$RP_1 \subseteq \mathrm{RP}_2$	Any synthesis of RP_1 is synthesis of RP_2

Examples of atomic reading project are:

$$RP_1 \equiv 9\text{learn} \bullet \text{Preliminary}$$

$$RP_2 \equiv 9\text{summarize} \bullet \text{Paradigm}$$

An example of composite reading project:

$$RP_3 \equiv 9\text{learn} \bullet \text{Preliminaryu}9\text{summarize} \bullet \text{Paradigm}$$

The reading project RP_3 consists in learning the preliminaries and summarizing the paradigms.

V. The Synthesis System

In this section we present the main functions and the architecture of our synthesis system. We first show how to translate the reader request into a formalism that can take into account the domain expertise then, the synthesis rules.

5.1 Mapping the reader request on the Reading Project (R_P)

We transform the reader request into a formal and normalized expression (Reading Project) by using the following steps:

Step 1 The Request analysis allows the extraction of the significant elements of the reader request. These elements are composed of concepts and actions on these concepts. It may happen that some parts of the initial request do not contribute to the final result and is therefore considered as noise. We assumed that the obtained reader request is without noise.

Step 2 This step is devoted to matching the output of step 1 into a formal and normalized expression of reading project ontology. Matching is the process of finding relationships or correspondences between entities of different Ontology [17, 18]. The matching process can be seen as a function *M* which, from a pair of ontology to match o and o', an input alignment *A*, a set of parameters *p*, and a set of oracles and resources *r*, returns an alignment *A*' between these ontology:

$$A^{'} = M(o, o^{'}, A, p, r)$$

This can schematically represented as illustrated below:

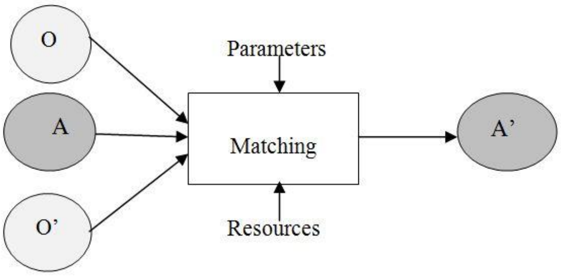
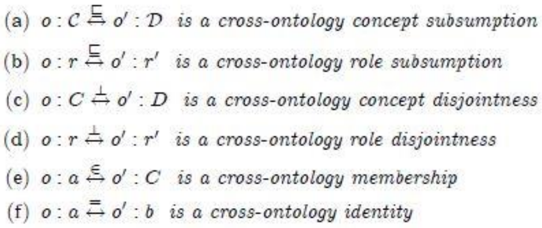


Fig. 4: Matching process

The alignment is a set of correspondences between two or more ontology (by analogy with molecular sequence alignment). The alignment is the output of the matching process.

The correspondences can take the following forms:



(a) $o : C \overset{\sqsubseteq}{\leftrightarrow} o' : D$ *is a cross-ontology concept subsumption*

(b) $o : r \overset{\sqsubseteq}{\leftrightarrow} o' : r'$ *is a cross-ontology role subsumption*

(c) $o : C \overset{\perp}{\leftrightarrow} o' : D$ *is a cross-ontology concept disjointness*

(d) $o : r \overset{\perp}{\leftrightarrow} o' : r'$ *is a cross-ontology role disjointness*

(e) $o : a \overset{\in}{\leftrightarrow} o' : C$ *is a cross-ontology membership*

(f) $o : a \overset{=}{\leftrightarrow} o' : b$ *is a cross-ontology identity*

The obtained alignment is then used to translate the initial request into the reading project

Step 3 additional operations consisting in extending the reading project ontology using the alignment. The new reading project ontology takes into account the concepts and the actions arising from the reader request.

The following figure summarizes the overall matching process.

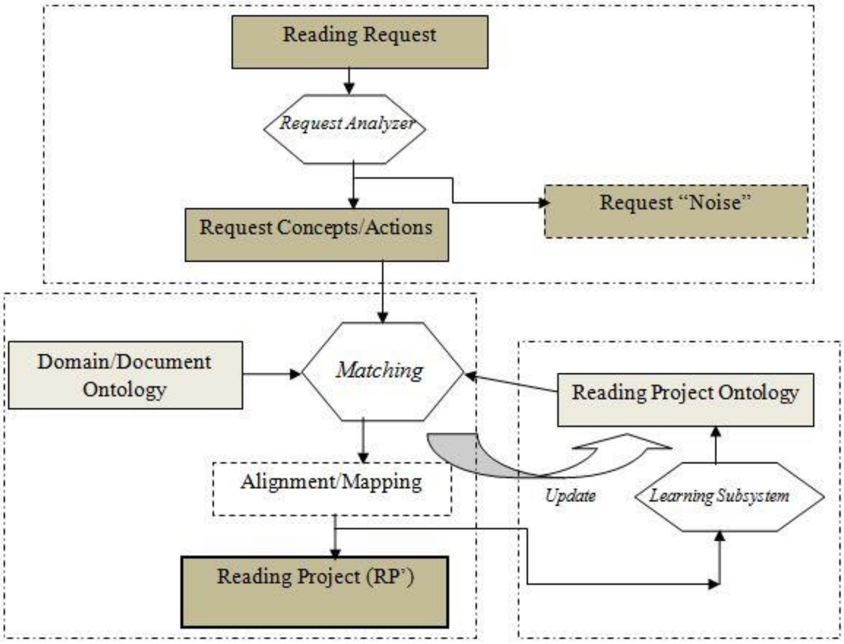


Fig. 5: The Matching subsystem

5.2 Inference system

5.2.1 Expert knowledge and synthesis rules

Expert knowledge is knowledge based on a good practice of the subject covered by the document. They are used to compute a suitable synthesis.

To define the synthesis result, let us call

$$f : P \times R_p \to S,$$

The synthesis expert function, where P is the set of reader profiles, R_p the set of reading projects and S; the set of synthesis elements.

$$f(P, R\_P) = S$$

Practically, the synthesis expert function f is de.ned in a dynamic and recurrent way. Production rules formalism can be used advantageously to build the function f because these rules can be written in associative way.

Synthesis production rules are of the form

IF < Condition > THEN < Synthesis Action >

where < Condition > is a predicate on reader profile and/or reading project, < Synthesis Action > is a prescription on the synthesis (synthesis action).

< Condition > can be specified as follows:

< Condition >::= < Item Condition > j < Item Condition > and < Condition >

where

< Item Condition >::= < READER_PROFILESpecif > j < READER_PROJECTSpecif > .

< Synthesis Action > is defined as:

< Synthesis Action >::= < Item Synthesis Action > j < Item Synthesis Action > and < Synthesis Action >.

A synthesis action is given in the following non exhaustive forms:

1. The mapping

$$C \mapsto D$$

This means, the concept D is a possible synthesis of the concept C.

2. The Forget function

$$Forget(C, Exp)$$

This means, in the concept *C*; forget the expression *Exp*. The semantic of the operator *Forget* is similar to the semantic de.ned in [19]

For example:

Rule1: *IF READER is ¬Academic and READING_PROJECT is 9learn ● Programming THEN Programming 7! (9IsComposedOf ● Paradigm u 9IsComposedOf ● Preliminary)*

Rule2: *IF READER is ¬Academic and READING_PROJECT is 9learn ● Programming THEN Forget(Paradigm; 9IsComposedOf: Functional u9IsComposedOf ● Object).*

- Rule 1 states: if the reader is not an academic and want to learn Programming then Programming can be brought back to Preliminary and Paradigm.
- Rule 2 states: in the same condition, drop Functional and Object concepts from the Paradigm concept.

5.2.2 The system architecture

The figure below summarizes the overall synthesis system. The inputs of this system are: the reading project and expert knowledge and synthesis rules.

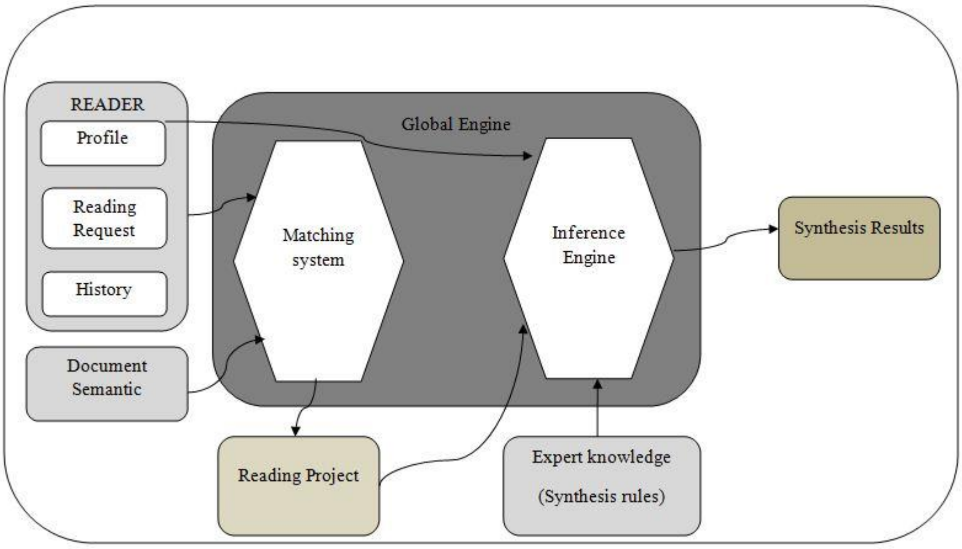


Fig. 6: The overall system

Our systemhas the following components :

- The matching subsystem translates the reader request into a reading ontology expression. The inputs of this subsystem are : the reader pro.le, the reader request, his history and the document semantic.

  The output is the reader.s reading project.

- The inference subsystem produces a document synthesis by using the reading project arising from thematching subsystem, the reader pro.le and the expert knowledge.

VI. Conclusion

In this paper, the main topic is to propose a new approach for document synthesis using ontology and Description Logics.

We use a unique formalism (DL) to specify the semantic of the document, the reader profile and his reading project. Then, DL is associated to productions rules formalism to describe the synthesis knowledge.

In .ne, our approach allows to reason (compare, infer) on document domain ontology, reader pro.les, reader requests and expert knowledge.

For an illustration, a prototype is in development in Laboratoire Africain d'Informatique et de Mathématiques Appliquées (LAIMA).

Further works address investigations of textual aspects of the document and noise on the reader requests.

Another research direction is to improve document synthesis using classical approaches (statistical approach for example) associated to our ontological methodology.

References

[1] [J. G. MEUNIER, S. B. GASTALDYL, L. C. PAQUIN, La gestion et l'analyse des textes par Ordinateur; leur spéci.cité dans le traitement de l'information, ICO Québec vol. 6(1-2), printemps 1994, pp 19-28.

[2] J. C. HOCHON, F. EVRARD, Lecture professionnelle et gestion personnalisée de documents textuels, ICO vol. 6(1-2), printemps 1994, pp 9-18.

[3] R. NOUSSI and S. KOUSSOUBE, Using Abstraction for automatic synthesis of electronic documents. CARI.96 Libreville, October 1996;

[4] S. MUKHERJEE, G. YANG, W. TAN; I. V. RAMAKRISHNAN, Automatic discovery of semantic structures in HTML Documents, International Conference on Document Analysis and Recognition (ICDAR) 2003.

[5] K. SPARCK JONES, Automatic summarizing: a review and discussion of the state of the art, Technical report published by the University of Cambridge computer laboratory 2007.

[6] A. BOSSARD, Contribution au résumé automatique multi document, Thèse de l'Université Paris XIII, école doctorale Galilée, Juin 2010.

[7] J. R. PUNIN M. S. KRISHNAMOORTHY, Semantics of Links and Document Structure Discovery, Proceedings of the WWW2002 International Workshop on the Semantic Web, Hawaii, May 7, 2002.

[8] F. BAADER, D. CALVANESE, D. MCGUINNESS, D. NARDI, P. SCHNEIDER, "The description logic handbook: Theory, implementation and applications", Cambridge University Press (ISBN-13: 9780521781763 j ISBN-10: 0521781760), 2003.

[9] I. HORROCKS, U. SATTELER S. TOBIES, Practical reasoning for expressive description logics, Logic for Programming and Automated Reasoning, pp. 161-180, 1999.

[10] A. NAPOLL, une introduction aux logiques de description, Rapport de recherche, lNRIA Lorraine, N_3314; 72 p, Décembre 1997.

[11] T. R. GRUBER Toward principles for the design of ontology used for knowledge sharing. International Journal of Human-Computer Studies 45, 907.928 (1995).

[12] D. ROMAN, U. KELLER, H. LAUSEN, J. DE BRUIJN, R. LARA, M. STOLLBERG, A. POLLERES, C. FEIER, C. BUSSLER, D. FENSEL Web Service Modeling Ontology Applied Ontology Journal 1(1), 77.106 (2005).

[13] S. BRANDT; ANNI-YASMIN TURHAN, "Using Non-standard Inferences in Description Logic. What does it buy me", Proc. of KI-2001 Workshop on Applications of Description Logics (KIDLWS.01). Volume 44 of CEUR (http://ceur-ws.org/)/. 2001.

[14] M. COURANT, La compréhension de petites annonces dans le système-expert havane, Thèse de doctorat de l'université de Rennes I, 1987.

[15] R. KITTREDGE The significance of sublanguage for automatic translation, ed Serguei Nirensburg-, Machine translation: Theoretical and methodological issues, Studies in natural language processing, pp. 59-67, Cambridge University Press, 1987.

[16] D. BERARDI, D. CALVANESE, G.DE GIACOMO, "Reasoning on UML Class Diagrams

using Description Logic Based Systems" Proc. of the KI.2001 Workshop on Applications of Description Logics. Volume 44 of CEUR Electronic Workshop Proceedings, http://ceur-ws.org/ Vienna, September 18, 2001.

[17] A. ZIMMERMANN and C. LE DUC Reasoning with a network of aligned ontology, Proceedings of the 2nd International conference on Web reasoning and Rule systems (RR2008), 2008.

[18] J. EUZENAT, P. SHVAIKO, Ontology Matching, Springer ISBN 978-3-540-49611-3, 2007.

[19] C. LE DUC and N. LE THANH Combining Revision production rules and Description logic, Proceedings of the seventh International conference on Knowledge-Based Intelligent Information and Engineering Systems, 2003.

Authors' Profiles

KOUSSOUBE Souleymane (1951—), male, Institut Africain d'informatique Professor, Ph.D., his research directions include ontology, expert knowledge Business intelligence, intelligence computation. Machine learning, Dataming.

NOUSSI Roger (1960 —), male, Institut Africain d'informatique Professor, Ph.D., his research directions include ontology, knowledge representation Problem frames Domain theory and intelligence computation.

KONFE Balira Ousmane (1974—), Institut Africain d'informatique, Professor, Ph.D supervisor for Ph.D. candidate, his research directions include Biomathematics modeling, global optimization, parameters identification technique, intelligence computation and optimal control.

How to cite this paper: Souleymane KOUSSOUBE, Roger NOUSSI, Balira O. KONFE,"Using Description Logics to specify a Document Synthesis System", International Journal of Intelligent Systems and Applications(IJISA), vol.5, no.3, pp.13-22, 2013.DOI: 10.5815/ijisa.2013.03.02