Ontologies-Based Platform for Sociocultural Knowledge Management

Papa Fary Diallo[1,2,3], Olivier Corby[2,3], Isabelle Mirbel[3]
Moussa Lo[1] and Seydina M. Ndiaye[1]

[1] Université Gaston Berger - UFR SAT - LANI, SENEGAL
[2] INRIA Sophia Antipolis, FRANCE
[3] Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, FRANCE
{diallo.papa-fary, moussa.lo, seydina.ndiaye}@ugb.edu.sn
olivier.corby@inria.fr
isabelle.mirbel@unice.fr

Abstract : In this paper, we present a sociocultural platform aiming at persevering and capitalizing sociocultural events in Senegal. This platform relies on Semantic Web technologies. First, we discuss the two ontologies we provided to support our platform: an upper-level sociocultural ontology (USCO) and a human time ontology (HuTO). To build our upper-level ontology we proposed a methodology based on the theory of Russian psychologist Lev Vygotsky called "Vygotskian Framework". We also present how the upper-level ontology can be matched in the Linked Open Data (LOD) cloud. On the other hand, we present the Human Time Ontology (HuTO) of which major contributions are (i) the modeling of non-convex intervals (repetitive interval) like *every Monday*, (ii) representation deictic temporal expressions which form specific relations with time speech and (iii) qualitative temporal notions which are temporal notions relative to a culture or a geographical position. Finally, we discuss the platform designed on top of Semantic MediaWiki to apply our scientific contributions. indeed, the platform allows Senegalese communities to share and co-construct their sociocultural knowledge.

Keywords : Upper Ontology . Domain Ontology . Semantic Web . LOD . Schema Matching . Cultural Information . Temporal Information.

1 Introduction

Due to the lack of knowledge of the African territories, it is very common to meet African youth knowing more about the geography of the West than their own countries. Thus, to refresh the memory of our fellow citizens and revive the many stories that accompany the daily life of the different African territories, we initiated the establishment of an online sociocultural encyclopedia.

Our goal is to develop a distributed infrastructure that will allow Senegalese communities to share sociocultural knowledge about tourism, economy, education, agriculture, etc. We place ourselves in the context of setting up a sociocultural ontology-based Semantic Web platform to enable communities to share their knowledge as in classical social network.

We propose to have a new point of view on the concept of community in the context of the social Web where the community typically represents a set of individuals sharing the same aspirations. Our point of view allows us to approach a community as an atomic entity and to focus this time on the sharing of knowledge between communities.

Thus, in (Diallo et al., 2011), we presented a methodology to develop our ontology which is structured around the "Vygotskian Framework" (Vygotsky, 1978; John-Steiner & Mahn, 1996) proposed by Russian psychologist Lev Vygotsky. This framework examines the relationship

between knowledge and the development of a society in three areas: (a) human (subject), (b) objects (buildings, parks, etc.) and (c) artefacts (abstract things).

The semantic representation is based on: (i) a sociocultural ontology and (ii) a human time modeling ontology. In this paper we focus on the design and development of these ontologies. The development of the sociocultural ontology has two parts. Firstly, creating an upper-level ontology from which we will create, secondly, a domain ontology for Senegalese sociocultural context. The human time modeling ontology developed, HuTO (Human Time Ontology), is an RDFS ontology to temporally annotate RDF resources using human time expressions that preserves the evolution of data (change in value) over time.

The term "ontology" is borrowed from philosophy, where an ontology is the study of existence. According to (Uschold, 1998), in the context of knowledge engineering, ontology "may take a variety of forms, but necessarily it will include a vocabulary of terms, and some specification of their meaning. This includes definitions and indications of how concepts are interrelated which collectively imposes a structure on the domain and constrains the possible interpretations of terms".

This paper continues by presenting our research context. After that, the third section presents the sociocultural ontology construction process. First, we present the Vygotskian framework and how we exploited it to model our upper ontology concepts and properties between these concepts. After that, we will present the schema matching done with Schema.org and DBpedia ontology. The fourth section presents HuTO our ontology to temporally annotate sociocultural knowledge. First, we define temporal logics chosen for our modeling. Second, we present the concepts and properties provided to model temporal expressions. Third, we discuss temporal data annotation. The fifth section presents how the scientific contributions of sections 3 and 4 have been implemented in a platform within Senegalese context. The sixth section presents the related work on modeling process and the related work on temporal information handling in the Semantic Web. We end with a conclusion and the perspectives of this work.

2 Research Context

The overall objective of the research is to strengthen the socio-economic development approach based on knowledge by implementing a sustainable framework to enable communities to identify and share their resources (in terms of skills, knowledge, potential, projects, etc.) and their needs (lack of infrastructure, resources, etc.). This is to have an overview of the opportunities and problems in the whole territory, information from the base (from folks on a daily basis in the different communities) rather than macro elements far away from the reality on the ground.

After the failure and the reconsideration of several models of development in recent decades, theoretical research seems to turn to approaches to development based on knowledge. Therefore, the specific objective is to enable Senegalese communities to share their knowledge through the social and semantic Webs technologies. Combining these two approaches produces "collective knowledge systems" (Gruber, 2007). Indeed, social Web technologies allow users to co-construct a corpus, "collective intelligence" (Gruber, 2007) and using Semantic Web technologies on this ”collective intelligence” through representation and reasoning for creating new knowledge and best way to learn and share through the Web.

Thereby, our work is based on two research areas. First, in social Web area where the work is to propose a framework for Senegalese communities to produce this "collective intelligence".

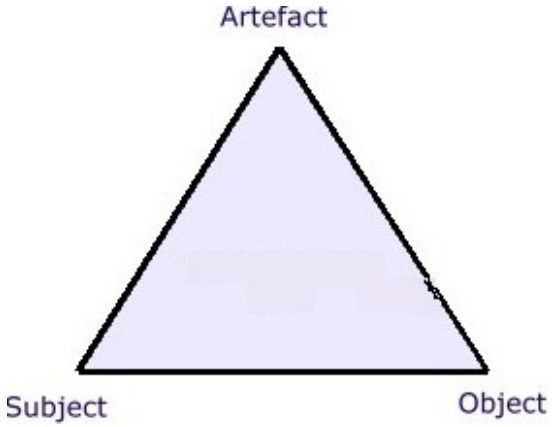

Figure 1: Vygotsky's mediating triangle.

Second, in Semantic Web area where the work is to propose ways to annotate (represent) these collective data. In this paper, we focus on the semantic aspect of our work.

The Semantic Web is underpinned by ontologies that are vocabularies of terms that characterize a domain and relations between those terms. The first step was to study an ontology in sociocultural context (Diallo et al., 2011) where the aims are to allow users to share and to co-construct their knowledge about aspects related both to society and culture. This study shows that there are no ontologies developed in this purpose. Furthermore, the handling of sociocultural data involves a lot of temporal notions among which are temporal notions shared across the world as well as notions specific to the culture of a country or a region. Thereby, in this paper we are interesting on sociocultural and temporal ontologies development.

Our aims in the sociocultural ontology development is to characterize features that represent the social aspects of a community, i.e. represent communities, their activities et general information around their existence. Temporal ontology development aims to annotate temporally those data mainly cultural activities.

3 Towards a Sociocultural Ontology

3.1 Vygotskian Framework

Vygotsky theory, also known as the first generation of Activity Theory, is a metaphorical space representing the location of cognitive development, a space occupied by subjects, experts, and any other agents capable of contributing in development. This theory considers human action (activity) as being mediated by objects such as tools that carry with them the cultural history of mankind. In this theory, individual development cannot be understood without reference to the social context within which such development is embedded. Humans do not interact directly with objects of environment. This relationship between them is mediated by cultural means, tools and signs. Human action has a tripartite structure (Fig. 1).

Vygotskian framework is not a methodology, much less a predictive theory; it is a philosophical framework or a clarifying tool for studying human practices as development processes. It offers at least the possibility to conceptualize in principle a scientific way of meta-cognitive

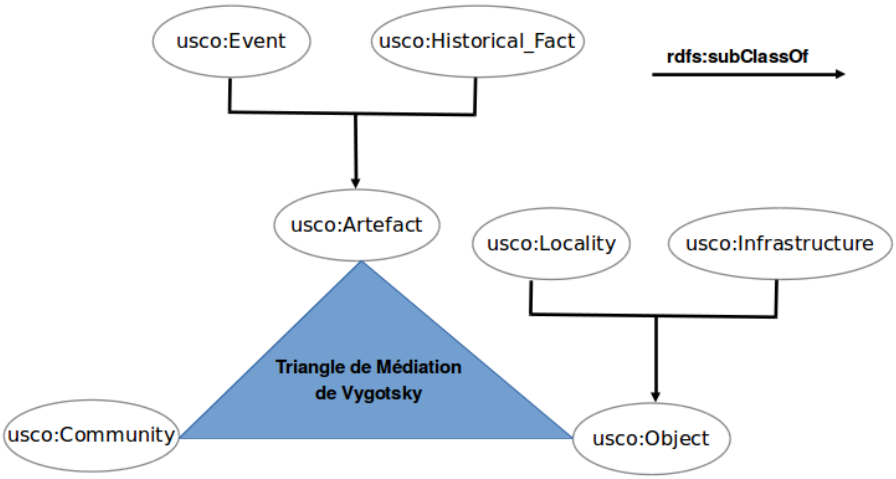

Figure 2: Sociocultural Upper-level Ontology.

processes (Ivic, 1994). Thus, it allows to bind this cognitive development dimension in general and to understand the origin of this capability to control its own internal processes as shown in the schema of Figure 1. It describes the transition from external and interpersonal control to individual intra-psychic control. Thus, the main objective of this theory is to overcome the gap between human and culture and society (Cole, 2010).

However with its Subject axis, Vygotsky's framework focuses on individual actions. Thus, we propose to replace this axis by a Community which is, in a sociocultural context, a group of people with a shared history, culture, ethnicity or interest and which wants to exchange or collaborate via the Web to share their knowledge of this area. Our communities are characterized by three components: 1) a common socio-cultural interest[1], 2) exchange, collaboration and sharing among members and 3) use of the Internet to interact. Therefore, Communities activities allow us to capitalize communities practices within their area.

Thus, we could say that the Vygotsky theory is a "socio-historical-cultural development theory" (Ivic, 1994). With the three axes of Vygotsky theory, we can model the different concepts of our ontology:

- Subject: it is the subject of the study. In our model, it is a group of humans in which we want to study their activities related to the environment.
- Object: they are objects in the human environment
- Artefact: it can be psychological tools produced by sociocultural evolution to which humans have access by being active in their community and not in isolation.

[1] Aspect related both to society and to culture.

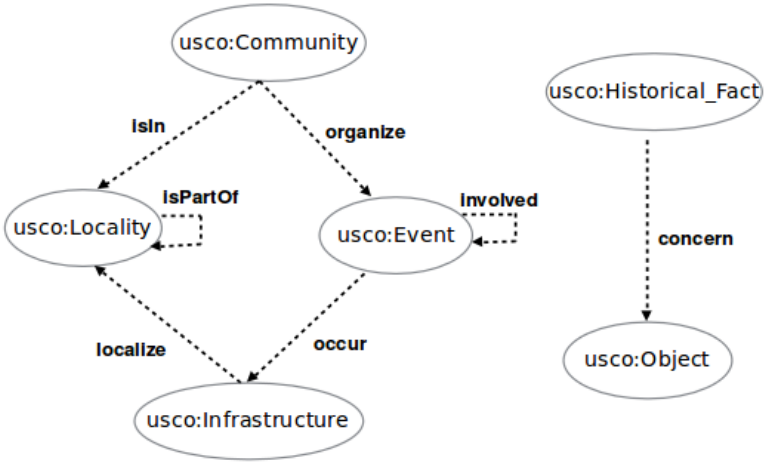

Figure 3: Sociocultural properties.

3.2 Upper-level Ontology

An upper-level ontology is an ontology which describes general concepts (classes) and properties (relations and attributes) which meanings are shared across the world. Thus the three axes of the Vygotsky process will be fundamental concepts in our upper-level ontology (Fig. 2):

- Community: in our modeling, community specializes Subject in the Vygotsky's mediating triangle. A community is a group of people sharing a common history, culture, ethnicity or interest and who want to exchange or collaborate via the Web to share their knowledge in relation with this area.
- Locality and Infrastructure: object axis in the Vygotsky's mediating triangle is divided in two parts. Locality is a place where humans live and Infrastructure is all buildings within the human environment.
- Event and Historical_Fact: our mediating tools are Events which are organized by Communities within their environment and Historical_Facts which are historical events which occur in the environment of the Community.

Concepts alone do not provide enough information. They should be associated with attributes that play an essential role in ontology development. They describe properties of concepts and instances. However, we do not detail all attributes of our various concepts. But with our schema matching in the next section, one can have an overview of attributes taken from the LOD. Another important thing in ontology development is to define relations between concepts. A design choice that must be made during ontology development is to define when knowledge should be modeled as attribute or as relation pointing to another concept. Usually it is an attribute when values are of a primitive type (integer, string, etc.), and it is a relation when values are of a type said complex, i.e., another concept of the ontology. Thus, Figure 3 illustrates the different relations that may exist between our concepts.

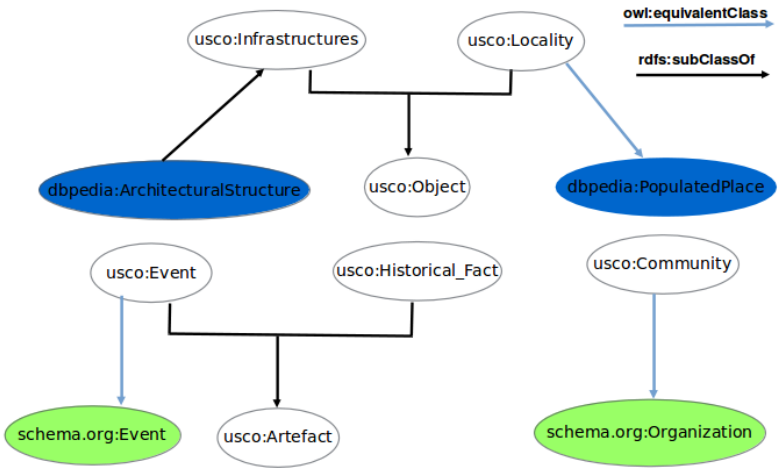

Figure 4: Schema matching.

With these properties, we can implement Vygotsky's mediating triangle at different levels (*Community - Event - Infrastructure* and *Community - Locality - Historical_Fact*). They allow representing different sociocultural knowledge:

- *Organize* and *occur* properties instantiate the Vygotsky's mediating triangle and allow us to know the different interests of the Community based on the Event organized and the way their Infrastructure is used. *Involved* property ties an Event and its sub-events.
- *Concern* and *isIn* properties also implement the Vygotsky's mediating triangle and provide different historical events about a Locality and what kind of historical events is related to a Community living space. *isPartOf* property ties a Locality and its sub-parts.
- By combining *concern*, *isIn* and *localize* properties, we have more information about Community environment. By combining *concern* and *occur* properties we can have an understanding of the use of certain Infrastructure by some Communities to organize their Event.

3.3 Schema Matching

Schema matching, in semantic computing, takes two graph structures (e.g., RDF schemas) and generates a mapping between nodes of these graphs that correspond semantically to each others. The idea is to find between two graphs which concepts are semantically equivalent and to do the mapping between them. As shown in (Noy, 2004), there are two methods for mapping discovery between ontologies. The first approach consists in relying on an upper-level ontology widely accepted by the community and to find mappings between this "shared ontology" and ours. The second approach relies on heuristics-based or machine learning techniques to find mappings. Our schema matching belongs to the first family of approaches by trying to find in the LOD vocabularies describing concepts similar to ours. Thus we will not reinvent the wheel. These mappings allow us to take advantage of all descriptions done on these schemas.

As shown in Figure 4, we have chosen two general vocabularies. The first one is Schema.org which provides a collection of schema for structured data markup on Web pages. It was an initiative of Yahoo, Bing and Google. It is widely used now by various organization for publishing data on the Web. In this way we linked mappings using the *owl:equivalentClass* constructor. *Schema.org:Organization* and *Community* as well as *Schema.org:Event* and *Event*. Thus, using a reasoner we are able to say that any individual that is an instance of *Community* or *Event* is also an instance of the equivalent concept in the schema matching.

The second vocabulary is DBpedia schema. DBpedia is a machine-readable knowledge base extracted from Wikipedia. At this moment[2], the English version of DBpedia knowledge base describes 4.58 million things, out of which 4.22 million are classified in a consistent ontology. It covers many domains; it represents real community agreement; it automatically evolves as Wikipedia changes. Using the *owl:equivalentClass* constructor, we have defined a mapping between *Locality* and *DBpedia:PopulatedPlace*. This mapping allows us by using a reasoner to say that any individual that is an instance of *DBpedia:PopulatedPlace* is also an instance of *Locality* which is useful when we want to create a domain ontology by adding a restriction to the domain (taking only the populated place in one country for example).

Excepted *DBpedia:PopulatedPlace*, we consider all *DBpedia:Place* sub-concepts as *Infrastructure* sub-con- cepts. Thus using the *rdfs:subClassOf* constructor, we have defined a mapping between *Infrastructure* concept to other *DBpedia:Place* sub-concepts as shown in Figure 4 (*DBpedia:ArchitecturalStructure* concept is a *DBpedia:Place* sub-concept). As *DBpedia:Place* concept has eleven sub-concepts excepted *DBpedia:PopulatedPlace*, we performed eleven schema matchings such as with *DBpedia:ArchitecturalStructure*. Thus a reasoner can deduce that if an individual is an instance of one of these concepts then it is also an instance of Infrastructure concept. This is also interesting when creating a domain ontology by adding a restriction to the domain (taking only the architectural structure in one country or area for example).

In this section, we explained our process to build a sociocultural ontology from scratch. The process is based on the extension of the sociocultural approach to the Cultural Historical Activity Theory (CHAT) to a community. Indeed, The CHAT is a psychological approach to understanding human evolution in the cultural, institutional and historical context of its environment. The sociocultural approach of the CHAT is person-centered, thus, we have extended this approach to a community to model the sociocultural knowledge of Senegalese communities. This allows to have the three main concepts of the ontology USCO : `Community`, `Artefact` and `Infrastructure`. To ease our process, we aligned USCO ontology with two vocabularies in the Linked Open Data, schema.org and DBpedia. This alignment allows to take advantage of descriptions done in these vocabularies.

Thus, USCO ontology is used to enable communities to share and co-construct their sociocultural heritage. This is done through the descriptions of social and cultural events organized by these communities, the descriptions of the resources available to these communities but also the descriptions of these communities themselves (see section 5).

[2]http://wiki.dbpedia.org/about, 06/07/2015

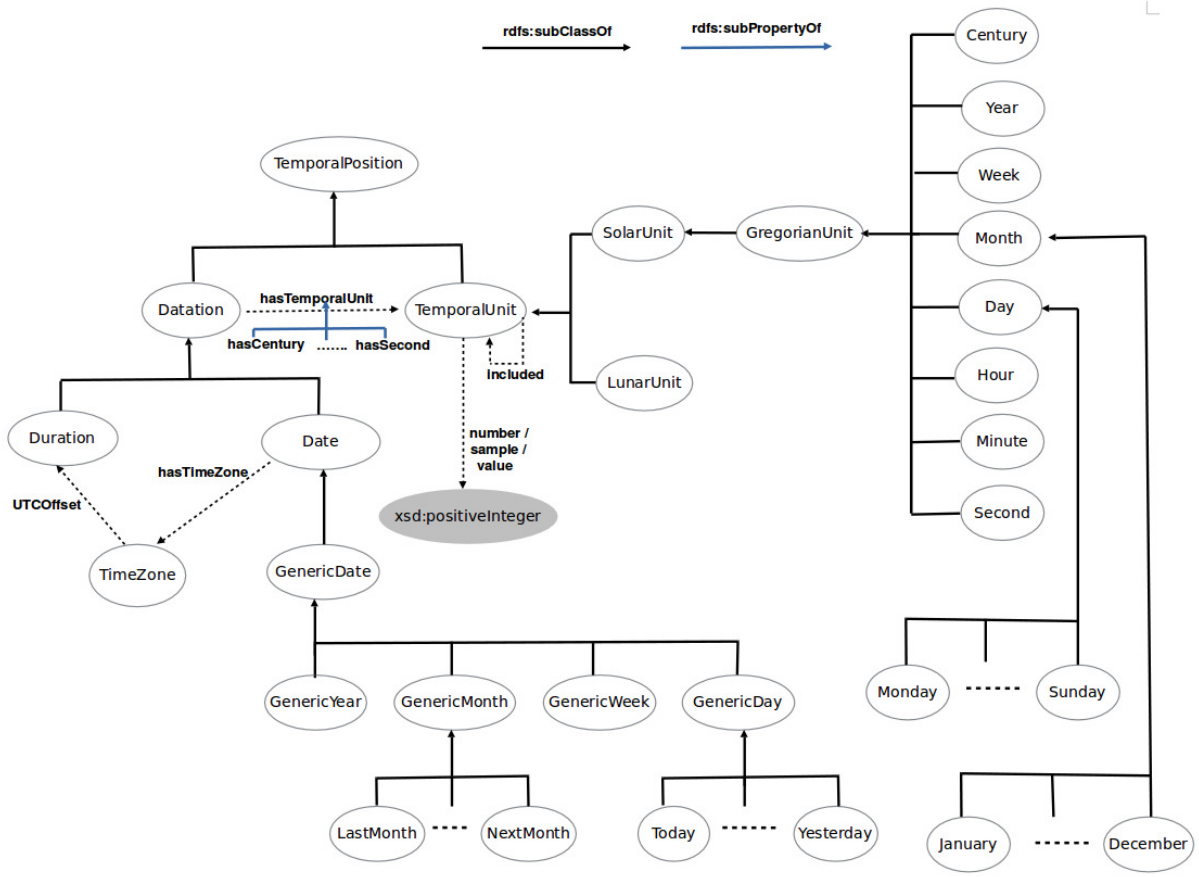

Figure 5: Main concepts and properties of HuTO's *TemporalPosition* concept.

4 HuTO: a Human Time Ontology

HuTO[3] is an ontology formalized in RDFS allowing temporal annotation of Semantic Web resources. This ontology allows users to define temporal anchors and to capture temporal changes associated with annotated resources. It makes possible temporal querying of the knowledge base using SPARQL queries. Specifically, HuTO allows:

- To model:
  1. **Explicit temporal expressions**: they are immediately anchored; e.g: *August 30, 2014*; *summer 2014*;
  2. **Deictic temporal expressions**: they form a specific relation with time speech; e.g: *today*; *last year*;
  3. **Duration**: they indicate a time interval; e.g, *2 hours*, *20 minutes*;
  4. **Cyclic temporal expressions**: known also as non-convex interval (Ladkin, 1987), allow to model repetitive intervals; e.g: *every Monday*, *every two months*;

[3]http://ns.inria.fr/huto/

5. **Qualitative temporal expressions** : they are used to define temporal notions relating to a culture or a geographical area which can be used as a time reference;
6. **Mixed temporal expressions**: they combine temporal expressions above; e.g: *two months of the last year*.

- To normalize temporal expressions in order to apply reasoning and query them.

Thereby, HuTO allows mainly to annotate of human time expressions. Indeed, as Hall (Hall, 1984) shows, human time is related to the culture insofar as time is an aggregate of concepts, phenomena and rhythms covering a very wide reality. Thus, the combination of these six criteria allows to express fixed time concepts, regular and linear (biological and physical time (Hall, 1984)) and subjective notions of time related to the perception of time (individual time (Hall, 1984)). Unfortunately, there is no approach which handles this aspect (see section 6).

In the temporal modeling in Semantic Web, we have identified two research domains : 1) temporal expressions modeling which allows to represent temporal entities as a date, an intervals, etc. and 2) temporal data annotation which allows to have time scope of resources.

4.1 Temporal Expressions Modeling

4.1.1 Overview

Before we can characterize the temporal expressions, we need to specify a temporal logic. The logic described here is based on time intervals and positions in a temporal coordinate system as defined by Hayes (Hayes, 1996). We chose intervals rather than time-point because as DOLCE (Masolo, 2003) and Time Interval Pattern[4] approaches we consider that intervals can be considered related to points. Indeed, time-points can be seen as intervals as short as possible. Thereby, time-point can be defined as a moment during which nothing can happen, which has no internal structure. This means that there is no real need to model time-points, since even minutes, which are the smaller human time units, have internal structures. In our modeling, we distinguish two types of intervals: 1) convex intervals that are continuous intervals and 2) non-convex intervals that are discontinuous intervals.

Positions in a temporal coordinate system allow to represent calendar dates et duration. They are components of the intervals and are used to better answer temporal questions.

4.1.2 Date, Calendar Date and Unit

In HuTO, the main concept to model a position of temporal coordinate system is *TemporalPosition* (Fig. 5) which has *Datation* and *TemporalUnit* as sub-concepts. These three concepts are abstract concepts, this means they have no direct instances. The *Datation* concept is used to model date and duration from which *Date* and *Duration* concepts are derived. *Date* concept allows to model dates such as the *xsd:dateTime* (examples 1a et 1b). It represents a specific date and time. A time zone expression may be added with *TimeZone* concept by giving the difference with UTC (Coordinated Universal Time). If no time zone value is present, it is assumed

[4]http://ontologydesignpatterns.org/wiki/Submissions:TimeInterval

to be UTC. *Duration* concept allows to define duration such as the *xsd:duration*. It represents a duration of time expressed as a number of years, months, days, hours, minutes, and seconds (example 1c).

As temporal units, we consider *LunarUnit* and *SolarUnit*. This approach allows to model different calendar units such as Chinese or Hebrew calendars but also Senegalese sociocultural events which all use solar and lunar units. It is possible to define other temporal units such as Maya or Aztec units which will be specializations of *TemporalUnit* concept. As we are interested in modeling Senegalese sociocultural events, in Figure 5 we detail Gregorian units for which defined granularities range from *Century* to *Second* and lunar unit which is used to model religious events in Senegal.

HuTO allows to model deictic dates with *GenericDate* concept (Fig. 5) which has as sub-concepts *GenericDay* (*Today*, etc.), *GenericWeek* (*lastWeek*, etc.), *Ge- nericMonth* (*NextMonth*, etc.) and *GenericYear* (*This- Year*, etc.).

To hierarchize units, we defined *included* property (Fig. 1) which is used to rank temporal units. For instance, for Gregorian units we have *included(Year, Century)* to mean *Year* unit is included in *Century* unit.

In example 1a as with *xsd:dateTime*, to specify a date one gives different units to the date. It is also possible to give the month unit as a digit (example 1b) or using a calendar month[5] (example 1a). A *GenericDate* it is a date which is dated in relation with the speech time . Thereby, this relation is done by providing the right date of the *GenericDate* (example 1d). To model a duration, one gives each unit related to the duration and the *number* property is used to have the magnitude (example 1c).

```
a. Date(Tuesday, February 17, 2015 at 10 pm)
  [a :Date;
     :hashour [a :Hour ;
                 :value 22] ;
     :hasDay [a :Tuesday ;
                :value 17] ;
     :hasMonth [a :February] ;
     :hasYear [a :Year ;
                 :value 2015]] .

b. Date(04 15) "April 15"
 [a :Date ;
    :hasDay [a :Day ;
               :value 4] ;
    :hasMonth [a :Month ;
                 :value 4]] .
```

[5]We use 24-hour format for representing hours

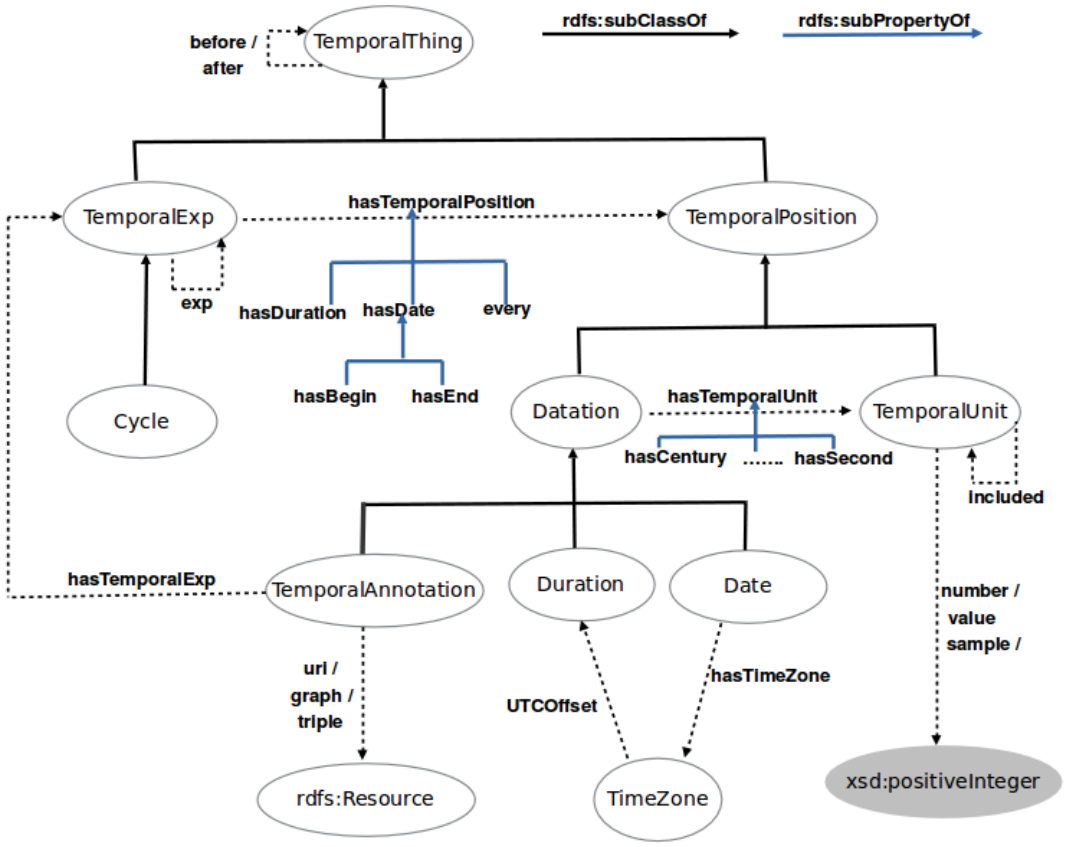

Figure 6: Main concepts and properties of intervals in HuTO's modeling.

```
c. 2 hours 30 minutes
  [a :Duration ;
     :hasHour [a :Hour ;
                 :number 2] ;
     :hasMinute [a :Minute ;
                   :number 30]] .

d. Date(Today)- Friday April 25, 2014
  [a :Today ;
     :hasDay [a :Friday ;
                :value 25] ;
     :hasMonth [a :April] ;
     :hasYear [a :Year ;
                 :value 2014]] .
```

Example 1: Modeling simple notions of dates.

4.1.3 Instant, Interval and Duration

A temporal element can be an instant, an interval or a duration. We have chosen to represent all temporal elements as intervals modeled using the concept *TemporalExp* (Fig. 6). Therefore, if the end or the duration of the interval is not specified then the considered interval is the date unit. For example, the date *Friday, August 15, 2014* is considered as *24 hours* interval. To specify the beginning and/or the end of an interval, one must use the *TemporalExp* concept with *hasBegin* and/or *hasEnd* properties. To model an interval with duration (Example 2a), *TemporalExp* concept is also used with *hasBegin* property to specify the beginning and *hasDuration* property to specify the duration.

The *Cycle* concept is used to model non-convex intervals (repetitive). No-convex intervals are characterized by two entities: the repetition frequency and the convex interval to repeat. This frequency is a sub-concept *TemporalUnit* which represents the time unit to which the cycle is repeated. The convex interval is connected to the *Cycle* concept by the relation *exp*.

Overall, the *exp* (Fig. 6) property is used to connect a non-convex interval (*Cycle*) to a convex interval (*TemporalExp*) (Example 2a) or another non-convex interval. It is used also to connect a convex interval to a non-convex interval (Example 2b).

Note also that with the proposed model, we distinguish between closed and infinite intervals:

- An interval is closed if:
  1. *hasDate* property is used;
  2. *hasBegin* and *hasEnd* properties are used together;
  3. One of *hasBegin* or *hasEnd* properties is used together with *hasDuration* property.
- Otherwise the interval is infinite.

The example 2a represents a non-convex interval. The frequency of its repetitive interval is yearly, thereby the unit of the *every* property is year. The day of its repetitive interval is a *Sunday* but the first (*number 1*). The *number* property is used when counting like in duration representation. The example 2b[6] represents a non-convex interval delimited in time. Thereby, the non-convex interval is contained in a convex interval. Thus, the convex interval has two parts: one to model the time duration and the second (*exp* property) to model the non-convex interval. The *sample* property is used to model sample of date as in the expression *every 8 hours*.

```
a. The first Sunday of every April.
  [a :Cycle ;
     :every [a :Year] ;
     :exp [a :TemporalExp ;
             :hasDate
                [a :Date ;
                   :hasDay [ :Sunday ;
                             :number 1] ;
                   :hasMonth
                     [a :April]]]] .
```

[6] In the representation, the context of *Today* has been omitted to avoid overloading the example.

```
b. Every 8H for 10 days starting from today
  [a :TemporalExp ;
     :hasBegin [a :Today] ;
     :hasDuration [a :Duration ;
                     :hasDay [a :Day ;
                                :number 10]];
     :exp [a :Cycle ;
             :every [a :Hour ;
                       :sample 8]]] .
```

Example 2: Modeling non-convex intervals.

4.1.4 HuTO and Allen's Interval Algebra

Allen's interval algebra is a calculus for temporal reasoning introduced in (Allen, 1981 ; 1983). The calculus defines thirteen basic relations between temporal intervals and provides a composition table that can be used as a basis for reasoning about temporal descriptions of events. Following Allen's definition, an interval is convex, closed and ordered i.e. the boundaries are specified and the left one is strictly less than the right one. Thus the thirteen relations are: *equal*, *before*/*after*, *during*/*contains*, *meet*/*metBy*, *start*/*startedBy*, *finishes*/*finishedBy* and *overlaps*/*overlappedBy*.

In HuTO, at this moment, we included *before* and *after* properties (Fig. 6). They allow us to support relative dates which allows to determine the relative order of events without necessarily determining their absolute date as in the expression *the Battle of Dekheulé took place after the Battle of Mékhé* (Example 3b). Note that it allows us to have two implicit information: 1) the date of the referenced resource and 2) the two Allen relations between resources. In HuTO, *before* and *after* properties may be expressed between intervals, between resources and between a resource and an interval.

The logical characteristics of `before` and `after` properties are done by rules (see section 4.3.2 and appendix). Indeed, we use both properties to order events or convex intervals between them. The temporal reasoning are performed by using rules on convex intervals

4.2 Temporal Data Annotation

Temporal data annotation consists in linking data (a resource, a triple or a named graph) to their temporal dimension. Thanks to our modeling choices, two dimensions co-exist: temporal and non-temporal. The temporal dimension is specified using HuTO concepts and the non temporal one, consists in the existing triples in the knowledge base. Thus, information retrieval is facilitated by considering or not the temporal aspect.

The example 3a represents a triple annotation. Thereby, a *TemporalAnnotation* concept is created which uses *hasTemporalExp* property to model the interval of occurrence and *triple* property for the RDF reification. The example 3b is an example of how one can use *after* Allen's property. The property is made between two *TemporalAnnotation* concepts and the value of the property *uri* of each *TemporalAnnotation* concept is

related to the resource. The example 3c represents a set of triple annotation. Thereby, a *TemporalAnnotation* concept is created which is annotated temporally and pointing in a named graph URI (*graph* property). The named graph regroups all triples that share the same time interval which has been annotated with the *TemporalAnnotation* concept. The example 3d[7] represents a resource annotation. Like to other annotations, a *TemporalAnnotation* concept is created which is temporally annotated and the value of its *uri* property is the URI of the resource to annotate.

```
a. Senghor was the President of Senegal from
   September 1960 to December 1980
   [a :TemporalAnnotation ;
      :hasTemporalExp
         [a :TemporalExp ;
            :hasBegin [a :Date ;
                       :hasMonth
                         [a :September] ;
                       :hasYear
                         [a :Year ;
                            :value 1960]] ;
            :hasEnd [a :Date ;
                     :hasMonth
                        [a :December] ;
                     :hasYear
                       [a :Year ;
                          :value 1980]]] ;
      :triple [ rdf:subject    <Senghor> ;
                rdf:predicate  <presidentOf> ;
                rdf:object     <Senegal>]] .

b. The Battle of Dekheule took place after
   the Battle of Mekhe
       [a :TemporalAnnotation ;
           :uri <BattleOfDerkheule>]
   :after
      [a :TemporalAnnotation ;
         :uri <BattleOfMekhe>] .
```

[7]Fanal is a Carnival and Ndar (Saint-Louis) is a city north of Senegal.

```
c. In 2011 the City of Dakar had 1,056,009
   inhabitants, it was the most populated
   and its mayor was Mr Sall.
  [a :TemporalAnnotation ;
     :hasTemporalExp
       [a :TemporalExp ;
          :hasDate [a :Date ;
                      :hasYear
                        [a :Year ;
                           :value 2011]]] ;
     :graph <http://example.org/g/>] .

<http://example.org/g/>{
       <Dakar>  <population>   1056009 ;
                <rang>         1 ;
                <mayor>       <Sall>} .

d. The first Saturday of each December,
   the Fanal of Ndar is organized
  [a :TemporalAnnotation ;
     :hasTemporalExp
       [a :Cycle ;
          :every [a :Year] ;
          :exp [a :TemporalExp ;
                  :hasDate
                    [a :Date ;
                       :hasDay
                         [a :Saturday ;
                            :number 1] ;
                       :hasMonth
                         [a :December]]]] ;
     :uri <FanalOfNdar>] .
```

Example 3: Temporal data annotation.

Overall, temporal annotation can be associated with a resource, a triple or a named graph. To annotate these objects, we create *TemporalAnnotation* (Fig. 6) which has two properties. The first one, *hasTemporalExp* property, allows to represent the temporal expression that annotates the resource. The second one is associated with the resource to annotate as follows:

1. If it is a resource, *uri* property is used with the resource URI (Example 3d);
2. If it is a triple, *triple* property is used with an RDF reification (Example 3a);
3. If it is a named graph, *graph* property is used with the named graph URI (Example 3c.).

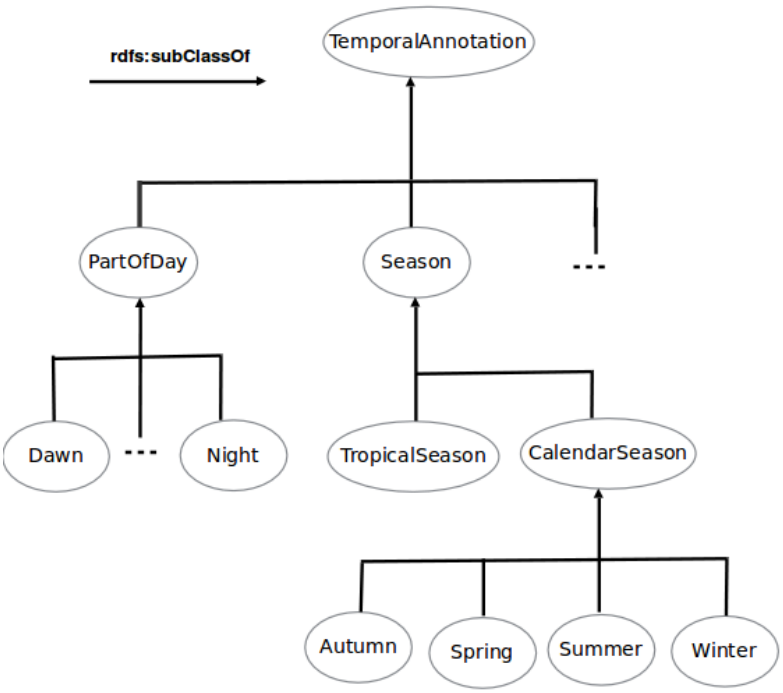

Figure 7: Qualitative modeling of time concepts in HuTO.

Note that HuTO also allows to model qualitative temporal notions (Fig. 7). HuTO allows a resource to be used as a temporal reference through the *TemporalAnnotation* concept. In this case, it is considered as a temporal object which can be used as a time anchor. Thereby, instead of to refer to the occurrence date one uses the annotated resource such as in the example 3b. It also is possible to model several temporal notions relative to one culture or to the geographical position.

For instance, the notion of *PartOfDay* is shared across the world but the duration depends to the geographical position. Thereby, the notion of *Night* means the same thing in Dakar (Senegalese capital) and Sydney but the intervals are not the same. On July 29, 2015 the *Night* is from midnight to 5am 30 and from 9pm to 11pm 59 at Dakar[8] while in Sydney[9] it is from midnight to 5am 30 and from 6pm 30 to 11pm 59. Thus to model this difference, *DakarNight* and *SydneyNight* concepts may be created as sub-concepts of *Night* concept and each of them will have its own schedule.

Thereby to model temporal domain notions, HuTO's qualitative notions can be used by specifying the good interval relative to geographical position. After that, the notion can be used such as in the example 3b as a time anchor. The same approach is done relative to the *Season*. African countries like in Senegal have a tropical season and the duration and period is relative to the geographical position.

HuTO also allows to define cultural temporal notions specific to one culture as a sub-concept

[8]http://www.timeanddate.com/astronomy/senegal/dakar

[9]http://www.timeanddate.com/astronomy/australia/sydney

of *TemporalAnnotation* concept. For instance in Senegal, the Gamou[10] (Mawlid) is an important reference date where many religious events are organized. To refer to the date of these events we use the expression *during Gamou*. Thus with HuTO modeling, the *Gamou* can be a sub-concept of *TemporalAnnotation* concept and one can use it as a time anchor to date those events.

Using HuTO has some advantages compared to other approaches. With regards to the modeling of temporal expressions, HuTO allows to represent complex statements. HuTO also allows to model closed and infinite intervals and to use a resource as a time anchor. With HuTO, one can define qualitative temporal notions relative to one culture or geographical position and use them as a time anchor. These aspects are not considered by other approaches. HuTO supports several temporal units and can be extended for other temporal units. Deictic temporal expressions can be modeled by using *GenericDate* or *TemporalAnnotation* concepts.

With regards to the temporal annotation, HuTO offers a representation that allows to annotate a resource, a triple or set of triples in a named graph. In our approach, the temporal dimension is separated from the other dimensions describing the resources. It facilitates information retrieval by considering or not the temporal aspect.

4.3 Temporal Reasoning and Rules

HuTO provides a conceptual model in RDFS to model temporal expressions and to annotate RDF resources. However many temporal expressions are expressed implicitly. The answers to many time-related queries are sometimes not explicitly represented but should be deduced. For this reason, we have proposed a set of rules to normalize the modeling of temporal data and also inference rules.

4.3.1 Temporal Data Normalization

Since HuTO is an RDFS ontology, we proposed rules expressed as CONSTRUCT SPARQL queries and aiming to deduce and explain the maximum of temporal information in order to enable reasoning about data.

Temporal information can be expressed in different ways. For example, a date can be represented either by using the day of week (Example 1a), or with digits (Example 1b). Thus, we have created rules to standardize these two types of writing. Thereby, regardless of the writing mode used, all possible representations will be added to the graph data. We also added two rules for determining leap years. These rules allow, for instance to calculate the number of days in the year which is useful for answering some types of queries.

HuTO allows a resource to be used such as temporal anchor (Example 4), therefore we defined a rule that adds explicitly the date of resources which use another resource such as temporal anchor.

[10]Period of 12 days to celebrate the birth of the Prophet of Islam

```
PREFIX hu: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?h ?m ?j }
WHERE{
 ?h ?o ?l .
 ?l a hu:TemporalAnnotation ;
    (hu:uri|hu:triple|hu:graph) ?y .
 ?x  a hu:TemporalAnnotation ;
    (hu:uri|hu:triple|hu:graph)  ?y ;
    hu:hasTemporalExp ?k .
 ?k ?m ?j .
 FILTER EXISTS{
  ?m rdfs:subPropertyOf+ hu:hasTemporalPosition}
 FILTER(?x != ?l)}
```

Example 4: Normalization rule for a resource using as temporal reference.

We also normalized intervals that are defined by duration by explicitly adding the end or the beginning date of the interval.

```
PREFIX hu: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?x hu:hasEnd
               [ ?j ?h ;
                 hu:hasMonth [a hu:Month ;
                                hu:value ?m2]}
WEHRE{
   ?x hu:hasBegin ?p ;
      hu:hasDuration/hu:hasMonth/hu:number ?m1 .
   ?p hu:hasMonth/hu:value ?m ;
      ?j ?h .

   BIND(?m + ?m1 - 1 as ?m2)

   FILTER( ?j != hu:hasMonth)
   FILTER NOT EXISTS{?p hu:hasYear ?g}
   FILTER NOT EXISTS{?x  hu:hasEnd ?h}
}
```

Example 5: Normalization rule for a duration expressed in month for a non-convex interval.

In this rule, we recover all properties related to the beginning date (*?p ?j ?h*) which are added to the end date created except the month. The month is calculated by adding to the beginning month (*?m1-1*) where *?m1* is the duration of the interval.

To check the consistency between *TemporalExp* and *Cycl*e concepts, we also add a verification query. In overall, if *Cycle* concept contains the *TemporalExp* concept then the frequency

unit (*every*) must be greater or equal to the largest unit of the time occurrence of the *TemporalExp*. Likewise, if *TemporalExp* concept contains a *Cycle* concept then the smallest unit of the time occurrence of the *TemporalExp* concept must be greater to the frequency unit of the *Cycle* concept. For instance in the example 2b, one cannot inverse the *TemporalExp* and the *Cycle* positions because one has *included(Hour, Day)* and *Day* is the smallest unit occurrence of *TemporalExp* concept and *Hour* is the frequency unit of the *Cycle* concept.

4.3.2 Implications and Inferences

Since RDFS does not model some basic inferences as transitivity or reflexivity, we have created inference rules to this result. Thus, we defined inference rules to *before* and *after* properties. For instance, *after* property is defined as transitive i.e. if *after(I1, I2)* and *after(I2, I3)* then we can infer *after(I1, I3)*. Further, *after* and *before* properties are set as inverse properties. Similarly if one relation (*after* or *before*) is expressed between two resources (respectively intervals), it is necessary to propagate the relation between the concerned intervals (respectively resources). For this, we developed propagation rules.

To verify the inclusion order of the *Cycle* and *TemporalExp* concepts, we defined *included* property that enables ranking date units. Thereby, in HuTO we have defined explicitly seven relations between units (*included(Year, Century)*, *included(Month, Year)*, etc). And we have defined two propagation rules: a rule for transitivity and another for transitivity by subordination i.e. if *included(d2, d1)* and *rdfs:subclassOf(d3, d2)* then *included(d3, d1)*.

Note that all these inference rules are expressed as CONSTRUCT queries in SPARQL interpreted as rules. They allow to add more information in the RDF graph either to clarify explicitly some information (normalization rules) or to add implicit information (reasoning rules). Using these rules in reasoners only affect the implications of RDF Schema (RDFS entailment).

For example, when asking the resources occurring on one date the rule in the **Example 4** is useful. Indeed, if there are resources expressed using temporal reference then without this rule we won't have those resources as answer.

The appendix section gives a general overview of HuTO's main concepts and main normalization and reasoning rules.

In summarizing, HuTO advantages are:

- To combine temporal expression modeling and temporal data annotation;
- To model convex and non-convex intervals and to also model closed and infinite intervals;
- To support the definition of several time units;
- To model deictic temporal expressions;
- To model qualitative temporal notions which can be used such as a time anchor.

5 Sociocultural Domain Ontology

In this section we explain how the scientific contributions of sections 3 and 4 can be used in a Senegalese context. Thereby, we have developed a domain ontology for USCO and HuTO ontologies which are integrated in a collaborative platform. A domain ontology describes a particular domain area. It represents one "world view" of a particular domain by describing concepts, relations between these concepts and instances that are existing things in this domain. As we said in the introduction, the domain ontology context is Senegal which is a west-African country and a former French colony.

The main sociocultural events organized by Senegalese are related to the ethnic groups or religious groups - more than 98% of Senegalese citizen are either Muslims (95%) or Catholics (3%). In the second section of this paper, we have included a mapping between *Schema.org:Organization* and *Community* concepts but ethnic group or religious group do not exist on *Schema.org:Organization*. Thus in the Senegalese context we have added two sub-concepts to the *Community*: *Religious-Community* and *EthnicCommunity*. Except these two concepts, other social groups can be used from *Schema.org:Organization*. For instance, there is *Schema.org:TeamsSport* which represents a social group practicing one sport. In Senegalese context, there are many sport teams practicing a "Lamb"[11] or wrestling. Likewise, there is *Schema.org:MusicGroup* which let us to model the traditional music group which sings historical epics.

We have also created *ReligiousEvent* and *EthnicEvent* concepts which are sub-concepts of *Event*. Others meaningful type of events have been found in *Schema.org*. Among these sub-concepts, *Schema.org:SaleEvent* allows to represent a "Louma" event that is a weekly sale market organized mainly during the weekend across the country.

According to (Corcho et Fernandez, 2003), "a domain ontology can be extracted from special purpose encyclopedias, dictionaries, nomenclatures, taxonomies, handbooks, scientific special languages, specialized knowledge bases, and from experts". Thus, as DBpedia provides also a knowledge base, the mapping done between *DBpedia:PopulatedPlace* and *Locality* concepts allows us to extract from DBpedia all populated place related to Senegal to populate our domain ontology by using a SPARQL query like this:

```
PREFIX onto: <http://dbpedia.org/ontology/>
PREFIX reso: <http://dbpedia.org/resource/>
SELECT *
WHERE{ ?l onto:type reso:Regions_of_Senegal .
       ?o onto:isPartOf ?l}
```

The fourth line from the above query returns all region types in Senegal. As we share with DBpedia ontology the same isPartOf property that links a *Locality* (*DBpedia:PopulatedPlace*) with its sub-parts, thus the fifth line shows the retrieved sub-part of the regions in DBpedia knowledge base. In DBpedia knowledge base, we have these following relationships between resources related to Dakar Region:

[11] Lamb is the national wrestling sport which regroups many teams organized by ethnic, cities, etc. We have also a national championship which duration is 9 months.

```
Grand_Yoff isPartOf Dakar_Departement
Dakar_Departement isPartOf  Dakar_Region
Dakar_Region isPartOf Senegal
```

As we know the upper *DBpedia:PopulatedPlace* concept related to Senegal is a *DBpedia:Region* and its sub-parts are *DBpedia:Department*. Thus the previous query returns Senegalese regions and their departments. Thus thanks to this DBpedia schema, one can populate our knowledge base and set up the right type for each Locality. Thus, the equivalence between *Locality* and *DBpedia:PopulatedPlace* concepts allows us to populate our domain ontology related to the localities from DBpedia knowledge base. *DBpedia:PopulatedPlace* type has 34 sub-types. Thus if no *DBpedia:PopulatedPlace* sub-type matches a given Senegalese administrative division we will create it. As DBpedia knowledge base is regularly updated, we run our queries from time to time to have updated information. The *Locality* concept is partially automatically updated from DBpedia and manually by users who can also update information extracted from DBpedia on our knowledge base.

As *DBpedia:ArchitecturalStructure* is an *Infrastructure* sub-concept, we model it in the same way as *DBpedia:PopulatedPlace* concept to populate partially and automatically *Infrastructure* concept:

```
PREFIX onto: <http://dbpedia.org/ontology/>
PREFIX reso: <http://dbpedia.org/resource/>
SELECT *
WHERE{ ?l  onto:location reso:Senegal ;
           a onto:ArchitecturalStructure}
```

The query above returns all Senegalese Architectural Structures from DBpedia knowledge base. Thus we can populate part of our knowledge base about *Infrastructure* concept by using this query.

With regards to the temporal notions, we have defined several qualitative temporal notions as *Gamou*, *WetSeason*[12], etc. These notions can be used by users to temporally annotate resources.

5.1 Sharing and Co-Construction Knowledge Platform

Previously, we showed how we updated automatically parts of our knowledge base. In this section we present the platform used for sharing and co-constructing knowledge. As the aim is to allow various Senegalese communities to share their sociocultural knowledge through the Web using Semantic Web technologies, we would like to support collaboration between users. This is why we rely on a Semantic MediaWiki (SMW). Wikis provide a collaborative environment to create, edit and update a Web site; therefore, semantic Wiki allows communities to collaborate in producing a set of textual and formal knowledge that is readable software agents.

We have chosen SMW[13] because it takes advantage of Wikipedia popularity (SMW is an extension of MediaWiki which is the Wiki system powering Wikipedia website). SMW also

[12] WetSeason is one of two tropical seasons we have in Senegal. It corresponds to the raining season.

[13] http://semantic-mediawiki.org.
In this paper we used SMW 2.1.3.

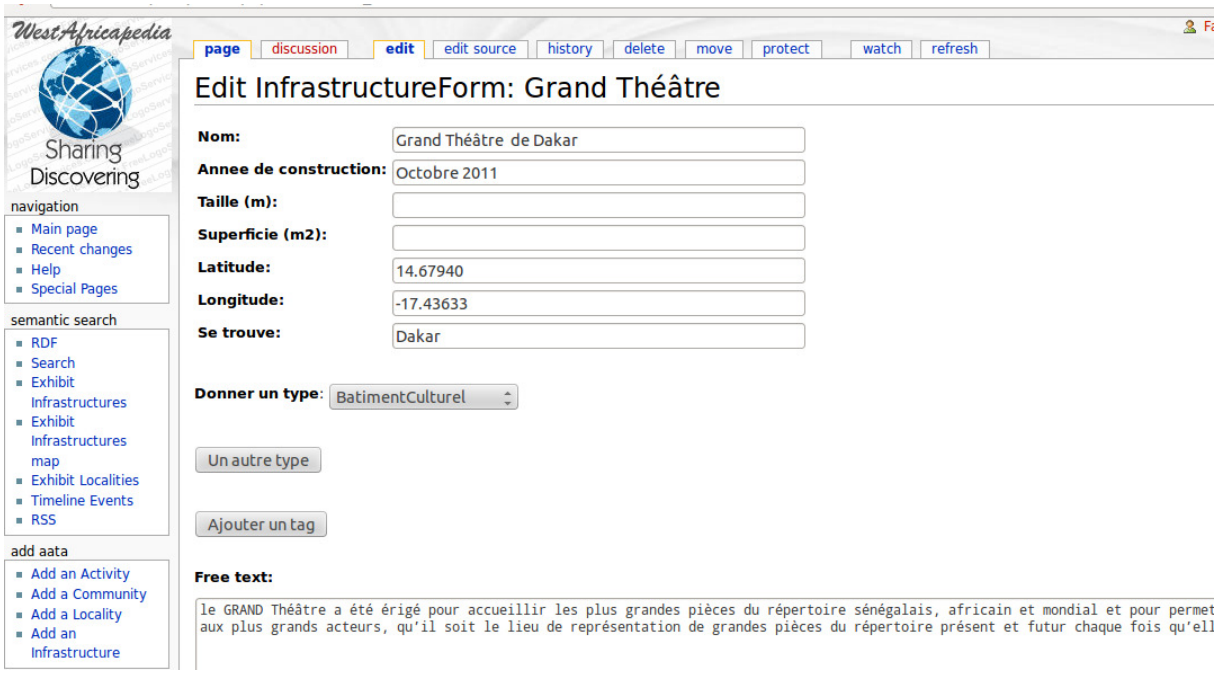

Figure 8: Filling form for Infrastructure type.

enables importing an ontology vocabulary and has many extensions, such as Semantic Forms which allow users to annotate pages. Further, SMW has a big community of contributions and several extensions can be used to make a user-friendly framework.

Others important extensions we have installed are Parser Functions and Variables extensions. The first one enhances the Wiki-text parser with helpful functions, mostly related to logic and string-handling. The second one allows to define a variable on a page and to dynamically take advantage of its value. To use our domain ontology, we have imported the vocabulary in SMW and using Semantic Form extension we provided support to users to annotate this ontology.

For installing native SMW we need to connect SMW with a RDF triple store, and to provide a SPARQL Endpoint. Thus we have chosen OpenLink Virtuoso[14] including a HTTP/SPARQL server and a back-end data- base engine. OpenLink Virtuoso is an hybrid database engine and middle-ware that covers relational data management, RDF linked data management, etc.

Having different views on our data we have used SIMILE[15] (Semantic Interoperability of Metadata and Information in unLike Environments) Widgets which let easily to create Web pages with advanced text search and filtering functionalities, interactive maps and other visualizations. SIMILE focuses on developing tools to increase the interoperability of disparate digital collections and developing robust, open source tools that empower users to access, manage, visualize and reuse digital assets. It seeks to enhance interoperability among digital assets, schema/vocabularies/ontologies, metadata, and services.

Note that the annotations are done with regards to USCO and HuTO ontologies. Indeed,

[14]http://virtuoso.openlinksw.com/.
In the paper we used virtuoso-opensource-develop-7.

[15]http://www.simile-widgets.org.

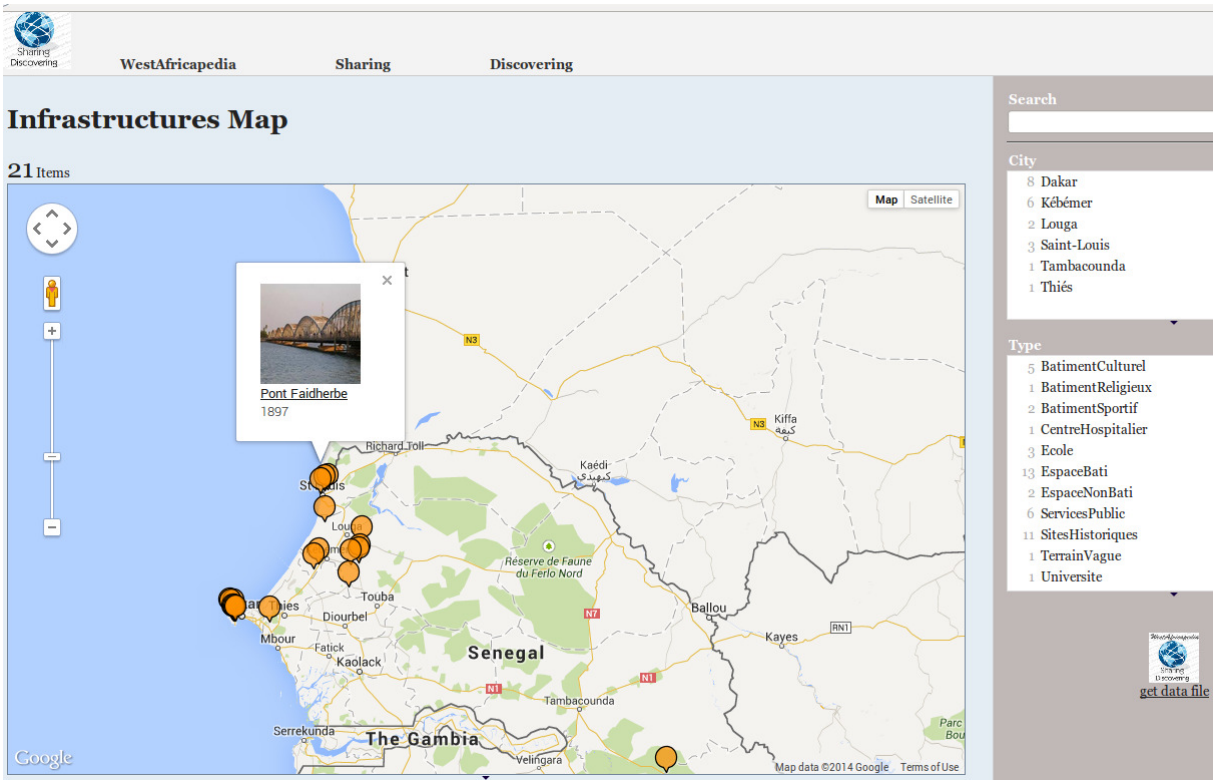

Figure 9: Senegalese map for Infrastructure.

SMW allows to import a vocabulary by giving its URL in a special namespace, thus, these two vocabularies are imported in the platform.

Figure 8 presents the form used to add an *Infrastructure* generated from the Semantic Form extension. Thereby, users do not need to know the SMW syntax to use the platform. On the bottom-left of the figure 8, one has several links for adding data about *Event*, *Community* and *Locality* which have their own form for filling data.

After having entered data in our Wiki, we want to present the data in catchy way i.e. using semantics on the data, to show implicit information about the data. For more user-friendliness, we have installed SIMILE Widget. For instance, we use Exhibit[16] which enables the web site to create dynamic exhibits i.e. it presents several filters on data to give many views of them (Fig. 9). Exhibit uses collections of data without using complex database and server-side technologies. Exhibit is used to show all *Infrastructure* within the Wiki with their position on Senegalese map (Fig. 9).

To facilitate information access in our platform, we developed a smart-phone and tablet Web page which allows, with user geo-localization, to map infrastructures on 5 Km radius around the user position. A click on one infrastructure gives information about it and events that occur in this infrastructure. Visiting localities, tourists can use it to have more information about these locations.

[16]http://www.simile-widgets.org/exhibit/

5.2 Data Validation

Wiki philosophy allows any user to edit/add content, but does not ensure that every editor respects the editing rules. Furthermore, using semantic Wiki, users should know and respect ontology syntax. SMW does not support inferences about disjoint classes as well as domain and range values. Using the Semantic Forms Extension, one constrains users to comply with the syntax when they use form to fill data. But, in the Free Text zone users can write all he/she wants. Thus, we do not have a way to control contributions for consistency in the knowledge base. To solve this problem, we created validation rules which:

- Generate warnings to help users to enter data correctly;
- Generate warnings to propose users to add more information relative to its contributions;
- Add metadata about wrong data within RDF file for helping in the consistency checking step.

Modeling validation rules in SMW is possible by using different extensions or components of MediaWiki and/or SMW (Bao et al., 2009, 2009b). Among these possibilities, we have chosen:

- Ask/Show query (a parser function part of SMW) is used to have an in-line query. The query string and any printout statements are directly given as parameters;
- Semantic Templates (part of SMW) allow mark-ups. This is helpful because it allows users to annotate pages without learning a specific syntax ;
- The Variable Extension (part of MediaWiki) is used to define the variables on page. Thus, it can be used in templates to specify the values variables ;
- Arraymap (part of MediaWiki) is a loop to get each value of a parameter when there are multiple values ;
- Parser Functions (part of MediaWiki) enhances the Wikitext parser with helpful functions, mostly related to boolean and string-handling.

Validation rule modeling is done by using Templates which are double-brace structures that can be placed in a page. It is a standard Wiki pages whose content is designed to be transcribed (embedded) inside other pages.

For instance, we defined consistency checking for disjoint classes. In the example below, we define the classes that are not disjoint or in other words those which are disjoint. Thus in this rule, we specify which classes (categories) can be used together. For example, an instance can be *CulturalBuilding* and *ReligiousBuilding*. Otherwise, the rule generates a warning message for the user and it adds metadata on the RDF file (Fig. 10). *WarningTemplate* template displays the warning message and *ErrorSyntaxTemplate* template adds metadata on the RDF file. *PropertyError* property regroups the wrong annotations relative to an instance (Fig. 10).

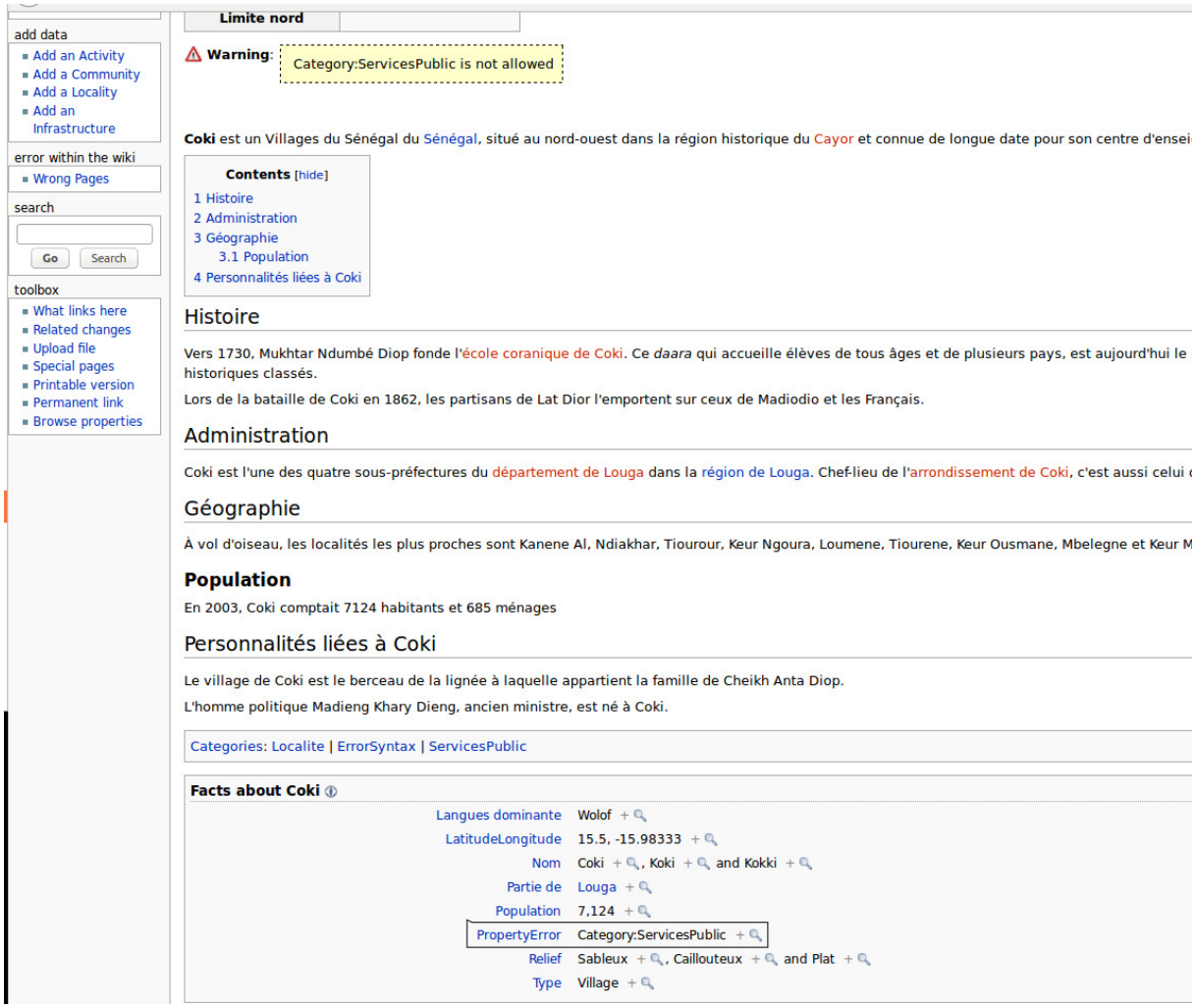

Figure 10: Page with errors.

```
{{#switch: {{{1|}}}
| Category:CulturalBuilding
| Category:ReligiousBuilding
| ...
| Category:WestLand=
| #default={{ErrorSyntaxTemplate|{{{1|}}}}}
    {{WarningTemplate|{{{1|}}} and
    Category:Infrastructure are disjoints}}}}
```

We have also defined rules for inverse properties. For example, if one *Locality* is set as *Est* border to one *Locality* then this one is also a *West* border to the first *Locality*. likewise, we have defined rules for verifying the domain and the range of properties.

We have created also pattern templates that enable pages created by users to be categorized by using validation rules.

5.3 Consistency Checking

In this step, we perform consistency checks on the data. Firstly, on the domain and the range of properties (for instance an *Event* occurs in one *Infrastructure* not in one *Locality*) and second on the structural consistency of the data (disjoint classes).

These consistency checks are expressed as DELETE queries in SPARQL and they are performed directly in the knowledge base before extracting the RDF file.

In the previous sub-section, we added metadata about annotations errors and we explained how warnings message are generated to help users to fix them. In this step, data correctness are check with regards to the ontology axioms. The best step to do it is during the user's annotations but with the Wiki philosophy any user can update data by entering free text. Thus, the metadata added during data validation allows us to optimize consistency checks since only instances which have *PropertyError* property are checked (Fig. 10).

In summarizing, the platform allows users to share and co-construct a "collective intelligence" which is annotated by using HuTO and USCO vocabularies to produce a "collective knowledge system". This collective knowledge system, structured mainly around USCO vocabulary, is queried out with Web applications to produce user-friendly views as Exhibit Map (fig. 9) and smartphone view. Mainly, user can create pages realtive to events, localities, communities and infrastructures and relative to the form of each entities, he/she can annotate these entities. For information retrieval, he/she can use Exhibit Map, research tool given by SMW and smartphone views.

6 Related work

In this section, we will present two related work domains. The first one is dedicated to the modeling process chosen to build our sociocultural ontology and the second one deals with temporal information modeling in the Semantic Web domain.

6.1 Sociocultural ontology

The first ontologies have been developed completely in a traditional way, without following a predefined method. From these early projects (Enterprise Ontology (Uschold et King, 1995), TOVE (Grüninger et Fox, 1994)) come from lists of recommendations constituting drafts or methodological frameworks. Since 1998, there has been the birth of more elaborate methodological frameworks inspired by the methods of the Knowledge Engineering (METHONTOLOGY (Gómez-Pérez et al, 1995), SENSUS (Swartout et al, 1997)) and based either on natural language processing (TERMINAE (Aussenac-Gilles et al, 2008)). Each group has its own methodology and there are no common guidelines because each ontology is built in a particular domain. Thus, there is no standard way or methodology for developing ontologies (Noy et McGuinness, 2001).

As we are interested to model sociocultural aspects, we have chosen to use CHAT (Cultural Historical Activity Theory) approach which is a conceptual framework that comes from the sociocultural tradition in Russian psychology. The main concept of this theory is the "Activity" which can be understood as the interaction between Human (Subject) and the World (Object).

However, there are different approaches related to CHAT. There is the sociocultural approach which focuses around a mediation process as the basic unit of analysis (Vygotsky, 1978 ; Cole, 1995 ; Wertsch et al., 1995 ; John-Steiner et Mahn, 1996 ; Cole, 2010) and there is also the activity theory approach which consider the activity as the basic unit of analysis (Leont'ev, 1979 ; Engeström, 2000 ; Kaptelinin, 2014).

The sociocultural approach (vygotskian framework) is person-centered and do not adequately address the cultural changes. While our aim is to propose a framework that would allow to model the sociocultural knowledge of Senegalese communities. In this sense, we are more interested in the action of the global community than the action of the entities that make up this community.

The activity theory is made following the framework proposed by Engeström (2000). Although Engeström's model concerns collective activities carried out by community (groups and organizations), he proposes to do it through the analysis of individual activities. In our study we are interested by communities activities and we do not consider dynamics within the community. We consider a community as an atomic entity and focus this time on the sharing of knowledge between communities.

Therefore, we proposed a methodology to identify features that represent the social aspects of a community (in the broadest sense) modeling these characteristics will guide us to model our sociocultural ontology. The approach we use is an extended version of the sociocultural approach. Indeed, as the sociocultural approach is person-centered we propose to center it to a community. The sociocultural approach examines the relationship between knowledge and development of a society in three areas: a) human (subject), b) objects (buildings, parks, etc.) c) artefacts (abstract things). Thus, we propose to use community in the place of human and then to model every activities and resources around communities.

We mean by methodology, work procedures and steps that describe why and how to conceptualize and build on artifact. Thus, we will rely on the extended version of sociocultural approach as a meta-model of our methodology.

To our best knowledge, there is no ontology about sociocultural domain except the one developed in (Kaladzavi et al., 2015). In this work, authors extended our work done in (Diallo et al., 2011) and this work for the Cameroonian sociocultural domain. In the Cameroonian context, authors wanted to model the dynamic within the community by identifying the role and task of each member of the community. Thereby they have adopted Engeström's model. In their work, they reused the concept of *Infrastructure* developed here.

Nevertheless, the approach described here can be seen as a combining four NeOn Methodology scenarios (Suárez-Figueroa, 2010 ; Suárez-Figueroa et al. 2012). Indeed, NeOn Methodology proposes a set of nine scenarios for building ontologies and ontology networks. Our modeling approach has two main steps. First, we have extended the sociocultural approach of the CHAT (Cultural Historical Activity Theory) to the notion of community. This step provides the three main concepts and their sub concepts. This step is equivalent to NeOn Methodology scenario 1 (Suárez-Figueroa, 2010 ; Suárez-Figueroa et al. 2012) where the ontology is developed from scratch (without reusing existing resources). The second step of our modeling is to use Linked Open Data (LOD) to complete our model. In this step, we reused and restructured Schema.org et DBpedia vocabularies. We reused some resources from these vocabularies and restructured them by extending and specializing these resources. The second step of our modeling corresponds mainly to two NeOn Methodology scenarios : scenario 5 and scenario

Table 1: Comparison between HuTO and three approaches ($\surd$ means that the ontology integrates this feature otherwise $\times$)

	OWL-Time	CNTRO	TimeML	HuTO
Non-convex Intervals	$\surd$	$\surd$	$\surd$	$\surd$
Infinite Intervals	$\surd$	$\times$	$\surd$	$\surd$
Deictic Time	$\times$	$\surd$	$\surd$	$\surd$
Temporal Reference	$\times$	$\surd$	$\times$	$\surd$
Temporal Units	$\surd$	$\surd$	$\surd$	$\surd$
Qualitative Representation	$\times$	$\times$	$\times$	$\surd$
Allen's Relations	$\surd$	$\surd$	partially	partially

8. However as described in the life cycle of NeOn Methodology, scenario 5 implies scenario 3 thereby our second step corresponds to the scenarios 3, 5 and 8. Scenario 3 corresponds to reuse ontological resources and scenario 5 corresponds to merge these resources with your resources and the scenario 8 corresponds to restructure resources that means extending and/or specializing resources.

6.2 Temporal ontologies in the Semantic Web

In Semantic Web (SW) data are produced with informal, semi-formal and formal temporal notions that must be understood by software agents. Thus, we distinguish two research areas in SW: modeling temporal expressions and temporal annotation resources. Modeling temporal expressions allows to represent a date, an interval, a repetitive interval (`every Wednesday John plays basketball`), relative or absolute temporal notions, etc. The temporal annotation resources allows to annotate temporally a knowledge base (expressed as triples in RDF) and that maintains the evolution of data (value change) over time.

6.2.1 Modeling temporal expressions

In this research area ones can find upper-level ontologies as GFO (Baumann et al., 2012), OpenCyc (Lenat et Guha, 1990 ; Witbrock et al., 2006), CIDOC CRM (Icoc, 2015), SUMO (Pease, 2007) et DOLCE (Masolo, 2003). These ontologies represent different temporal notions but their main drawback is that they don't model non convex intervals (repetitive intervals), deictic time and qualitative representation.

There are also domain ontologies as those we can find in the LOV[17] (Linked Open Vocabularies) and whose most complete and most reused is OWL-Time (Pan et Hobbs, 2005 ; Pan, 2007). There are also TimeML (Saurì, 2006) used in natural language processing and CNTRO (Tao et al., 2010 ; 2011) used in clinical domain. Table 1 compares HuTO with these three ontologies. This comparison is made followings seven criteria. Among these four approaches, only OWL-Time represents exceptions in non convex intervals. However, OWL-Time does not represents expressions like `Every month except July`. Regarding modeling of deictic time, OWL-Time does not take it into account and CNTRO represents it in strings which reduces the reasoning. Temporal reference allows a resource to be used as a date. Thus, only HuTO and CNTRO approaches are modeling this aspect. All approaches in the table 1 represents Gregorian temporal units. The advantage of HuTO approach compared to others is that it is not limited to the Gregorian units but it integrates also lunar units and our approach could be extended to other units as Maya or Aztec units. Only HuTO approach allows qualitative Representation which allows to model cultural or geographical temporal notions and that they can be used as temporal reference. As regards Allen's relations, TimeML and HuTO include only two relations but the others include all Allen relations and their logical characteristics. Unlike to TimeML, HuTO integrates the logical characteristics of *before* and *after* relations.

In the field of modeling temporal expressions, HuTO approach is more complete compared to others especially regarding to human time expressions which is important in the cultural domain.

6.2.2 Temporal annotation resources

In SW languages such as RDF, a statement (fact) is a binary relation that is used to connect two individuals (instances) or an individual and a value. Thereby to introduce a temporal dimension, it becomes necessary to manipulate a ternary relation. Therefore, ternary relation modeling is a special case of a more general problem which is modeling and querying of n-ary relations in the SW. Thereby, in the literature there are general approaches that try to address this problem in the SW as the N-ary relations approach (W3c, 2006), which proposes the introduction of a blank node related to the object and the subject of the triple. Thus, the blank node can be temporally annotated. There is also named graphs that allow to contextualize a set of triples by combining them in a single graph (URI) and this one can be temporally annotated. Another approach is the RDF reification which allows through *rdf:Statement* to add further information on a triple as time information.

There are also specific approaches about temporal modeling as 4D-Fluents (Welty et Fikes, 2006) which is an OWL ontology that proposes an approach based on perdurantism to model the temporal evolution of data. In this approach, authors consider that an object has a temporal part (*timeSlice*) that interacts with another object's temporal part. There is also the SOWL (Batsakis et Petrakis, 2011) approach which extends the 4D-Fluents by adding Allen's relations between *timeSlice*. Another approach is the Temporal RDF (Gutierrez et al., 2005 ; Hurtado et Vaisman, 2006) which extends RDF reification by adding a temporal dimension to the data. The graph can as well be accessed in two views, depending on whether one is interested in the temporality or knowledge of the modeled domain. In (Rula et al., 2014), authors propose a generic approach

[17] http://lov.okfn.org/dataset/lov/vocabs?tag=Time.

Table 2: Comparison in temporal annotation resources domain

	Objects Proliferation / Redundancy	Loss the semantic of triples	Extension of W3C technologies
RDF Reification	Yes	Yes	No
Named Graphs	Yes	No	No
N-ary Relations	Yes	Yes	No
Temporal RDF	Yes	No	Yes
4D-Fluents	Yes	Yes	No
TDL	No	No	Yes
HuTO	Yes	No	No

for extracting time information from the Web and their period of validity. In (Scheuermann et al., 2013), authors provide an empirical approach centered on user perspectives which helped to define different temporal patterns.

Table 2 compares seven approaches in temporal annotation resources domain followings three criteria. These criteria evaluate ontologies with respect to:

1. Objects proliferation (add an intermediate object to add temporality) and object redundancy (data duplication).

2. Loss the semantic of triples : temporality relates to a triple or set of triples (`address(John, Dakar)`) The loss of the semantics of the triples occurs when the relationship (semantics) between the two objects (`John` and `Dakar`) disappears when adding temporal information.

3. Extension of W3C technologies: these are syntactic and semantic extensions.

7 Conclusion

In this paper, we have presented our sociocultural platform. The platform integrates a sociocultural ontology and a temporal ontology in order to preserve and to capitalize sociocultural events in Senegal.

First, we presented a method for developing a sociocultural ontology in order to popularize and perpetuate the culture of a country through the sharing of the customs and history of different localities of the country. This method is based on the Vygotskian Framework which allowed us to model the sociocultural upper-level ontology. To complete this ontology, we proposed a

schema matching by using a "shared ontology". For this, we have chosen two concepts from Schema.org namely *Schema.org:Organization* and *Schema.org:Event*. We have also used DBpedia schema from which we use *DBpedia:PopulatedPlace* and *DBpedia:Architectural-Structure* concepts. We also described how we can develop a domain ontology by using these mappings in the Senegalese context. The schema matching with DBpedia allows us to populate our knowledge base by running queries on the DBpedia knowledge base. Thus running these queries from time to time, we update our knowledge base partially automatically.

Second, we have also proposed an ontology to handle many temporal information in our sociocultural domain ontology. Our ontology, HuTO (Human Time Ontology), has been specified in RDFS to temporally annotate RDF resources using human time expression.

With regards to temporal expression modeling, HuTO allows to model complex expressions. A distinction is made between closed and infinite intervals and between convex and non-convex intervals. Our ontology also includes *after* and *before* temporal relations as defined in (Allen, 1981; 1983). HuTO also allows a resource to be used as a time anchor for dating another resource. Different temporal units have been defined in HuTO which can be extended to consider other temporal units. Several deictic temporal expressions can be modeled by using HuTO.

With regards to temporal data annotation, HuTO proposes an approach that associates a distinct temporal dimension to the sociocultural data which makes the information retrieval easier by considering or not the temporal aspect. HuTO also allows to model the traceability of temporal changes on the data.

HuTO also integrates rules for normalization and reasoning. These rules allow to add more information in the RDF graph either to clarify explicitly some information (normalization rules) or to add implicit information (reasoning rules).

Third, we have presented our platform which integrates the two ontologies. The platform allows users to share their knowledge by annotating the domain ontology. This platform has been designed following two objectives. First, to provide a user-friendly framework to our users for collaborating and updating data. Therefore, we used Semantic MediaWiki (SMW) and some of its extensions. The second objective was to provide several tools (Exhibit map, Smartphone/tablet access) to enable different views on data. To have consistent data, we provided rules which help users to annotate in a good way. We also performed consistency checks on the data to delete errors annotation on the knowledge base.

Several directions remain to be pursued. First, we would like to propose query patterns and answers format more readable for users who are not RDF/SPAR- QL experts. Second, with regards to our time related ontology, we would like to study other relations defined in (Allen, 1981; 1983) and integrate those relevant in the sociocultural aspect. Last, but not least, we plan to integrate in HuTO uncertainty temporal notions as in the expression *in the late 1980s*.

Acknowledgements

This work is partially funded by Creatic4Africa Consortium, AUF (Agence Universitaire de la Francophonie) and LIRIMA (Laboratoire International de Recherche en Informatique et Mathématiques Appliquées). Special thanks to Diego Berrueta and Luis Polo for their help to design the first prototype. The anonymous referees of this paper provided a valuable list of comments that have been used to improve this paper.

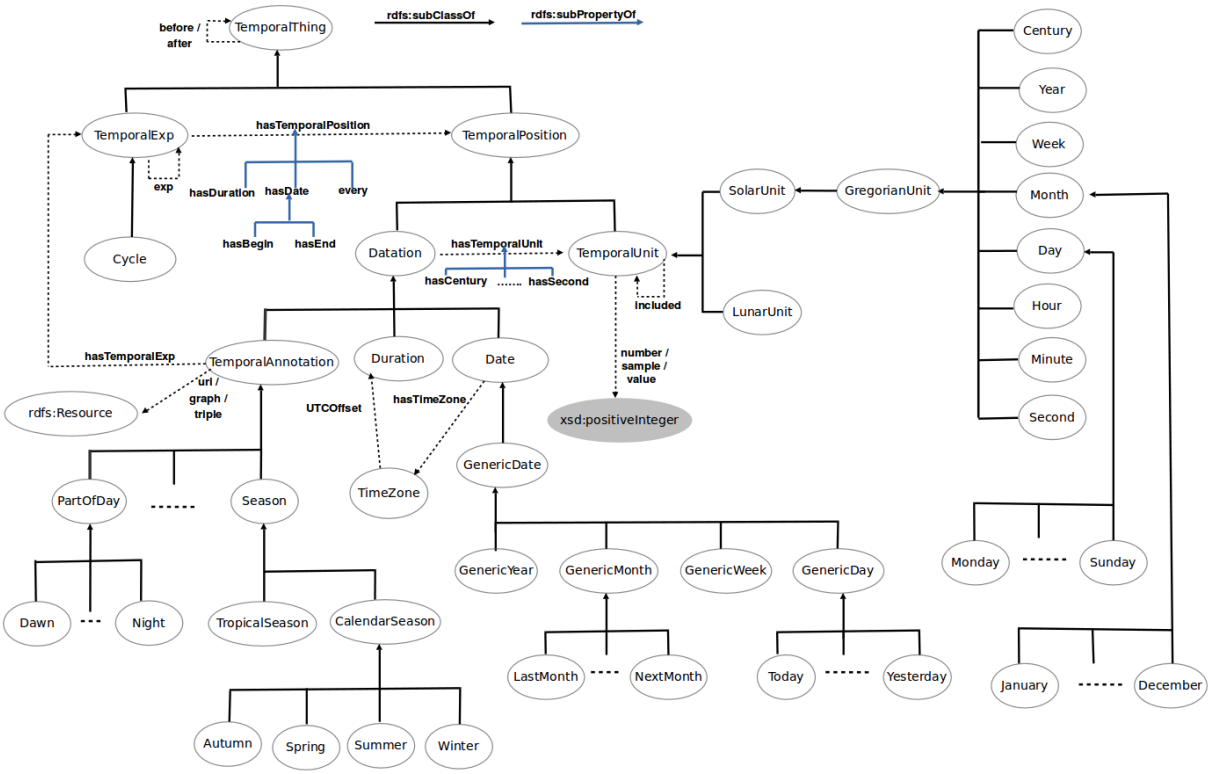

Figure 11: HuTO's main concepts.

Appendix

Figure 11 shows a general overview of HuTO's main concepts and properties.

Currently, we provided thirty five normalization and reasoning rules and one correction rule, but in this section we illustrate only some of them. Note that these rules concern only the Gregorian calendar.

Defining Leap Year

The two rules below check the leap years on the data and add the appropriate information. Thereby during the information retrieval, if a year doesn't have *leapYear* property set to *true* it means it is not a leap year. There are two rules: the first one is when **year mod 4 =0 and year mod 100 !=0**; the second one is when **year mod 400 = 0**.

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?x huto:leapYear true }
WHERE{ ?x a huto:Year ;
          huto:value ?y .
       BIND((?y / 4) as ?y1)
```

```
        BIND(ROUND(?y1) as ?y2)
        FILTER(?y1 = ?y2)
        BIND((?y / 100) as ?y3)
        BIND(ROUND(?y3) as ?y4)
        FILTER(?y3 != ?y4)
}
```

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?x huto:leapYear true }
WHERE{ ?x a huto:Year ;
          huto:value ?y .
        BIND((?y / 400) as ?y1)
        BIND(ROUND(?y1) as ?y2)
        FILTER(?y1 = ?y2)}
```

Adds the number of day, number of the month and the parity of the month

```
PREFIX huto: <http://ns.inria.fr/huto/>\\
CONSTRUCT{ ?x huto:number ?m ;
              huto:numberOfDay ?d ;
              huto:even ?e }
WHERE{
   ?x rdf:type ?o
   ?o rdfs:subClassOf huto:Month ;
      huto:number ?m ;
      huto:even ?e
   OPTIONAL{?x rdf:type/huto:numberOfDay ?d}
}
```

Adds the calendar months

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?x a ?c }
WHERE{ ?x  huto:number ?m ;
           rdf:type  huto:Month .
       ?c  huto:number ?m ;
           rdfs:subClassOf huto:Month .
}
```

We have created this kind of rules such as determining the number of day for the February month, the parity of months and the years. All these rules allow to add appropriate information which are necessaries to retrieve information such as *Which resources occur on odd month*, *Which resources occur from 01/22/2015 to 02/2/2015*, etc.

after and before properties are inverses

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?y huto:before ?x}
WHERE{ ?x huto:after ?y }
```

after is a transitive property (propagation rule)

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?x huto:after ?z }
WHERE{ ?x huto:after+ ?y .
       ?y huto:after+ ?z }
```

If Allen's relation is expressed on TemporalAnnotation then this rule adds appropriate information on the TemporalExp (case of after).

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?s huto:after ?o }
WHERE{
    [a huto:TemporalAnnotation ;
      (huto:uri|huto:triple|huto:graph) ?p1 ;
      huto:hasTemporalExp ?s]
  huto:after
    [a huto:TemporalAnnotation ;
       (huto:uri|huto:triple|huto:graph) ?p2 ;
       huto:hasTemporalExp ?o]
}
```

If Allen's relation is expressed on TemporalExp then this rule adds appropriate information on the TemporalAnnotation (case of after).

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?x huto:after ?y }
WHERE{
  ?s huto:after ?o

  ?x a huto:TemporalAnnotation ;
     (huto:uri|huto:triple|huto:graph) ?p1 ;
     huto:hasTemporalExp ?s .

  ?y a huto:TemporalAnnotation ;
     (huto:uri|huto:triple|huto:graph) ?p2 ;
     huto:hasTemporalExp ?o .
}
```

Equivalent rules have been provided for *before* property which has the same characteristics as *after* property.

Logical characteristics of Allen's relations. Annotation relying on hasBegin and hasEnd properties (case of after).

On the logical characteristics of Allen relations we check the data on which satisfy them. The two next rules check the data pairs which satisfy *after* logical characteristics. Equivalent rules have been provided for *before* property which has same characteristics as *after* property.

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?s huto:after ?o }
WHERE{
 ?s huto:hasTemporalExp/huto:hasEnd ?te .
 ?o huto:hasTemporalExp/huto:hasBegin ?tb .

 ?te huto:hasYear/huto:value ?y1 .
 ?tb huto:hasYear/huto:value ?y2 .

 FILTER(?y1 > ?y2 ||
        (?y1 = ?y2 ||
        EXISTS{
           ?te huto:hasMonth/huto:number ?m1 .
           ?tb huto:hasMonth/huto:number ?m2 .

           FILTER(?m1 > ?m2 ||
              (?m1 = ?m2 ||
              EXISTS{
                 ?te huto:hasDay/huto:value ?d1 .
                 ?tb huto:hasDay/huto:value ?d2 .
                 FILTER(?d1 > ?d2)
              })
            )
        })
    )
}
```

Logical characteristics of Allen's relations. Resources expressed with hasDate property (case of after).

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?s huto:after ?o }
WHERE{
  ?s huto:hasTemporalExp/huto:hasDate ?t1 .
  ?o huto:hasTemporalExp/huto:hasDate ?t2 .
```

```
  ?t1 huto:hasYear/huto:value ?y1 .
  ?t2 huto:hasYear/huto:value ?y2 .

  FILTER(?y1 > ?y2 ||
     (?y1 = ?y2 ||
     EXISTS{
        ?t1 huto:hasMonth/huto:number ?m1 .
        ?t2 huto:hasMonth/huto:number ?m2 .

        FILTER(?m1 > ?m2 ||
           (?m1 = ?m2 ||
              EXISTS{
                 ?t1 huto:hasDay/huto:value ?d1 .
                 ?t2 huto:hasDay/huto:value ?d2 .
                 FILTER(?d1 > ?d2)
       }))
 }))}
```

Logical characteristics of Allen's relations. Apply to the resources which have dates ordered (case of after).

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?resource1 huto:after ?resource2 }
WHERE{
  ?s1 huto:hasTemporalExp ?date1 ;
      (huto:uri|huto:graph|huto:triple) ?resource1 .

  ?s2 huto:hasTemporalExp ?date2 ;
      (huto:uri|huto:graph|huto:triple) ?resource2 .

   ?date1 huto:after ?date2
}
```

Normalize intervals (when using start on January without specifying the day it means the 1st January or when using end on January without specifying the day it means the 31st January)

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{ ?e huto:value ?d }
WHERE{
   ?x huto:hasMonth ?e
   ?e rdf:type ?o ;
      huto:numberOfDay ?n .
   ?o rdfs:subClassOf+ huto:Month
```

```
   BIND(IF(EXISTS{?s huto:hasBegin ?x}, 1,
          IF(EXISTS{?s huto:hasEnd ?x}, ?n, 0))
                       as ?d)

   FILTER NOT EXISTS{?x huto:hasDay ?p}
   FILTER(EXISTS{?s huto:hasBegin ?x} ||
          EXISTS\{?s huto:hasEnd ?x})
}
```

In our modeling approach, we consider any entity as an interval. Thus, this previous rule normalizes the approach. The rule normalizes both intervals expressed with *hasBeging* and *hasEnd* properties.

Normalize intervals described by duration

The two next rules are used to normalize intervals expressed by duration and *hasBegin* property. They add the appropriate end relative to the starting date. Note that, we created six rules about it because the rule is reliant on the format of the duration.

Note that equivalent rules have been provided for intervals expressed by duration and *hasEnd* property.

```
PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{
   ?x huto:hasEnd
      [ ?j ?l ;
        huto:hasDay [a huto:Day ;
                       huto:value ?d3] ;
        huto:hasMonth [a huto:Month ;
                         huto:number ?m2] ;
        huto:hasYear [a huto:Year ;
                        huto:value ?y1]]}
Where{
 ?x huto:hasBegin ?p ;
    huto:hasDuration/huto:hasDay/huto:number ?d.
 ?p huto:hasMonth/huto:number ?m ;
    huto:hasMonth/huto:numberOfDay ?nb ;
    huto:hasDay/huto:value ?d1 ;
    huto:hasYear/huto:value ?y ;
    ?j ?l .

  BIND(?d + ?d1 as ?d2)
  BIND(IF(?d2 > ?nb, ?d2 - ?nb, ?d2) as ?d3)
  BIND(IF(?d2 > ?nb, ?m + 1, ?m) as ?m3)
```

```
  BIND(IF(?m3 > 12, ?m3 - 12, ?m3) as ?m2)
  BIND(IF(?m > 12, ?y + 1, ?y) as ?y1)

  FILTER(?j != huto:hasMonth || ?j != huto:hasDay
          || ?j != huto:hasYear)
  FILTER NOT EXISTS{?x  huto:hasEnd ?h}
}


PREFIX huto: <http://ns.inria.fr/huto/>
CONSTRUCT{
    ?x huto:hasEnd
        [ ?j ?l ;
          huto:hasDay [a huto:Day ;
                       huto:value ?d3] ;
          huto:hasMonth [a huto:Month ;
                         huto:number ?m2]]}
WHERE{
 ?x huto:hasBegin ?p ;
    huto:hasDuration/huto:hasDay/huto:number ?d1.
 ?p  huto:hasMonth/huto:number ?m ;
     huto:hasMonth/huto:numberOfDay ?nb ;
     huto:hasDay/huto:value ?d ;
     ?j ?l .

 BIND(?d1 + ?d as ?d2)
 BIND(IF(?d2 > ?nb, ?d2 - ?nb, ?d2) as ?d3)
 BIND(IF(?d2 > ?nb, ?m + 1, ?m) as ?m2)

 FILTER(?j!=huto:hasMonth || ?j!=huto:hasDay)
 FILTER NOT EXISTS{?p huto:hasYear ?k}
 FILTER NOT EXISTS{?x huto:hasEnd ?h}
}
```

Correction Query

The query above checks both the consistency when *TemporalExp* concepts contains *Cycle* concept and when *Cycle* concepts contains *TemporalExp*. Note that if the query returns a result, one must again check if the entered values respect the scheduling because one can have multiple levels in an annotation. For instance, the expression **From 2015, every Friday at 17H one organizes a football match** has three levels. These levels are **From 2015**, **every Friday** and **at 17H**. Thereby, the query has to check each corresponding pair.

```
PREFIX huto: <http://ns.inria.fr/huto/>
DELETE{ ?p ?o ?k
        ?m ?f ?j
      }
INSERT{ ?p ?f ?j
        ?m ?o ?k
      }
WHERE{
   ?x (huto:every|huto:hasDate) ?p ;
      huto:exp/(huto:every|huto:hasDate) ?m .

     {?x a huto:Cycle .
      ?p  rdf:type ?y ;
          ?o ?k .
     }
 UNION
     {?x a huto:TemporalExp .
      ?p  ?o ?k .
      ?k  rdf:type ?y
     }

   ?m  ?f ?j .
   ?j rdf:type ?z .

   ?y  huto:included ?t .
   ?z  rdfs:subClassOf* ?t .

   FILTER(?y != ?t)
}
```