12th ACS/IEEE International Conference on Computer Systems and Applications AICCSA 2015
http://ieeexplore.ieee.org/document/7507204/
DOI: 10.1109/AICCSA.2015.7507204

Consistency awareness in a distributed collaborative system for semantic stores

Hafed Zarzour*
Department of Computer Science
Mohamed Cherif Messaadia University
41000, Souk-Ahras, Algeria
hafed.zarzour@gmail.com

Lamia Berrezzek
Department of Computer Science
Mohamed Cherif Messaadia University
41000, Souk-Ahras, Algeria

Hafida Ghomrani
Department of Computer Science
Mohamed Cherif Messaadia University
41000, Souk-Ahras, Algeria

Tarek Abid
Department of Computer Science
Mohamed Cherif Messaadia University
41000, Souk-Ahras, Algeria
abidtarek@yahoo.fr

Mokhtar Sellami
LABGED, Department of Computer Science
Badji Mokhtar University
23000, Annaba, Algeria.
m.sellami@dgrsdt.dz

Abstract— In distributed collaborative systems for semantic stores editing, multiple users can add, delete and change RDF statements starting from the same replicas and achieving to the same results at the end of the collaborative session. To improve the performance for such systems, the development of an efficient awareness mechanism is very important in order to help users to better understand the semantic stores evolution. Moreover, maintaining the consistency in replicated architecture is one of the most significant problems. However, none of the existing approaches describes how to define the awareness mechanism for distributed semantic stores performing concurrent changes. In this paper, we propose a new powerful optimistic replication solution called AB-Set, which can ensure not only a consistency criteria when editing data but also use semantic web technologies to define an awareness mechanism for making users aware of the different status of the store they share and update regardless of the concurrency level.

Keywords— semantic Web; triple-store; awareness; eventual consistency; distributed collaborative system

I. INTRODUCTION

Today, distributed collaborative system become more and more an interdisciplinary research field including distributed systems, Computer Supported Cooperative Work (CSCW), sociology, ergonomics, organizational studies, management, and further disciplines. The focus of distributed collaborative system research is on enabling and facilitating coordinating collaboration among virtual organizations of remote users who jointly fulfill common tasks through the network. The goal of all distributed collaborative system research efforts is to increase effectiveness and efficiency of the collaborative work by supporting a huge number of distributed communities to build a huge amount of data in best manner way. In addition, users working on a collaborative tasks need to be aware of various aspects of the group and the tasks, which is called awareness [1]. The Semantic Web is an extension of the Web that we know today, where data can be processed by humans as well as machines, to find, share and integrate information more easily [2]. The Resource Description Format (RDF) [3] is a data model used to represent information about World Wide Web resources as a graph: a set of individual objects, along with a set of connections among those objects. SPARQL/UPDATE [4] is a language used to express updates to an RDF store. It is intended to be a standard mechanism by which updates to a remote RDF store can be described, communicated and stored. SPARQL/UPDATE is a companion language to SPARQL and is envisaged to be used in conjunction with the SPARQL query language. Both RDF and SPARQL/UPDATE are one of the pillars of the Semantic Web.

When editing a common semantic store, users keep track of many components which together make up their collaboration awareness [5]. These components give information about different status such as: how, where, when, what, and who. That is, when several users edit a shared data, they know how those events occur, where they are editing, when various events happen , what they are doing, and with whom they are editing. In distributed collaborative systems for semantic stores, multiple users can add, delete and change triples starting from the same replicas and achieving to the same results at the end of the collaborative session. To improve the performance for such systems, the development of an efficient awareness mechanism is very important in order to help collaborators to better understand the semantic stores evolution. Moreover, maintaining the consistency in replicated architecture is one of the most significant problems. A Commutative Replicated Data Type (CRDT) [6] is invented as a consistency method that ensures convergence maintenance of replica in distributed collaborative systems without any difficulty over distributed networks. This method supposes that all concurrent operations commute [7].

Many previous works have been developed based on CRDT concepts such as [6] [8] [9][10]. However, for the best of our knowledge, there is no previous work that tried to build a CRDT for semantic store which take into account the awareness metrics during the collaborative work.

978-1-5090-0478-2/15/$31.00 ©2015 IEEE

In this paper, we propose a new approach called AB-Set that extends B-Set [11]; in which a new replicated data type for triple-stores is defined. AB-Set is a powerful optimistic replication solution that can ensure not only a consistency criteria but also use semantic web technologies to define an awareness mechanism for making users aware of the different status of the store thy share and update regardless of the concurrency level.

This paper is organized as follows: Section 2 details the backgrounds and related work. Section 3 presents the main components of our solution. Section 4 provides a use case over a demonstrative prototype. Section 5 discusses the proposed approach. Section 6 concludes the paper and points to future works.

II. BACKGROUND AND RELATED WORK

A. Consistency and awareness issues

To illustrate the challenging nature of the problem related to the consistency and awareness during concurrent changes in distributed architecture, we will identify a conflict in a collaborative editing system of semantic stores. Let's consider three users (U1 , U2, U3) (see fig 1).

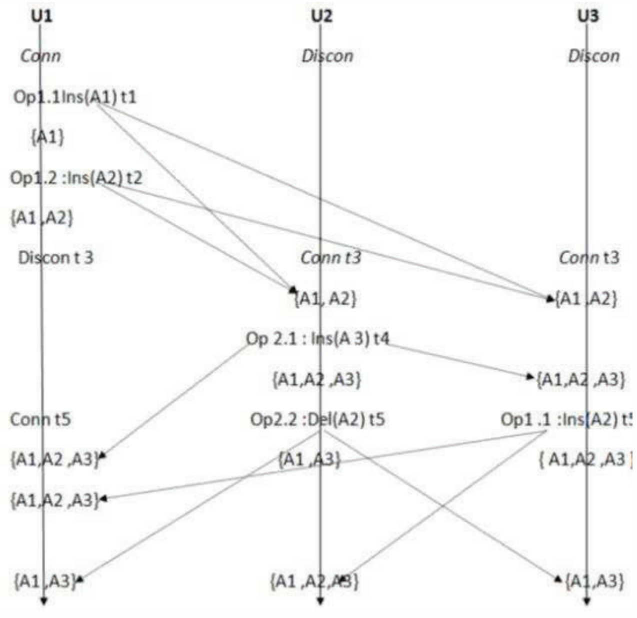


Fig. 1. Consistency and awareness criteria violation

Suppose that user U1 is connected and executes the first operation OP1.1 ADD(A1) at the time t1. Following that, he executes its second operation OP1.2 ADD (A2) at time t2, the result is: (A1, A2). At time t3, U2 and U3 connect and retrieve the contents of U1, all users share the same initial state (A1, A2), at this moment U1 disconnected. At time t4, U2 executes the operation Op2.1 ADD (A3). The obtained result is (A1, A2, and A3) for U2 and U3. At time t5, U1 and U2 reconnect and perform the operation Op2.2 DEL(A2). At the same time, U3 executes the operation OP3.1 ADD (A2). So, the result of each user is represented as follows: U1 (A1, A3) , U2(A1, A2 , A3), U3 (A1, A3). It is found that all users finish their session with different results. Since there is no awareness mechanism, the consistency criteria is violated, thus the convergence is not ensured.

B. Awareness in distributed systems

In recent years, as a result of the great collaborative working through large virtual communities of users, awareness mechanism has been increasingly supported by distributed and co-located systems such as: [12] [13] [14] [15]. Co-located systems have long been considered ideal for many types of group work, such as planning, decision-making, and design, since they provide a rich communication way, as well as promote awareness and coordination through the use of shared artifacts [16]. In [17], authors report that although awareness has been usually related to individuals' achievements, checking a system log, receiving a message as email, or conducting a debate are examples of work deftly used by participants within a group to become aware of their peers' thoughts.

Palantir [18] is an awareness framework providing software developers with insight into others' workspaces. It captures a number of events, for instance, inserted, deleted, changes in progress, changes committed, etc. The effect of the changes is computed and presented to the users by means of two metrics: the severity and impact metrics [19]. The severity metric measures how much a component in a user's workspace has changed when compared to its latest checked version in the repository or its version on the collaborators workspaces

State Treemap [20] is a divergence awareness tool; it allows the users to be aware of the differences among her documents and the others' documents by using the different document states. Edit Profile [21] is developed for linear document structure that offers awareness at different levels in used document. Users have the possibility to choose at which level they need to be aware of the modifications on a document. Thus, the parameters are computed based on the user selection of details.

In [22], authors propose the SCHO ontology, an unified formal ontology for constructing and sharing the causal history in the case of divergence awareness to allowing to localize where divergence is located and estimate how much divergence exists among the replicas. In [23], an awareness mechanism based on a notification purpose is developed for tracking modifications made to web content. Participants can create regions of interest on a web page that are saved as images. The system periodically checks if the highlighted regions visually differ from the saved ones with a certain severity index fixed by the participants. If an interesting region is updated, the participant is immediately notified by the system. However, none of the previously mentioned approaches describes how to define the awareness mechanism for distributed semantic stores performing concurrent changes.

C. Consistency in distributed systems

In the case of consistency in distributed systems, several methods have been investigated to extend the Operation

Transformation (OT) technique [24] such as GOTO [25], GOT [26], SOCT2 [27], SOCT4 [28], MOT2 [29].

Because of the use of victor clocks, such methods are known for their inability to scale as well as the fact that their correctness is hard for verification [6]. This is mainly because remote operations are inefficient as well as history buffers are likely to grow for larger memberships. However, SOCT2 is designed only for text document structure. Further, there are no transformation functions for Mind-Mapping data are available.

Recently, CRDT [6] technique is developed as a new consistency maintenance that is scalable and ensures coherence of replicas without synchronizing. The approach provides a simple mechanism for complex concurrency by defining specific types appropriated to each data type that are commutative for any performed set of operations in order to guarantee identical results. CRDT algorithms initially designed for P2P asynchronous collaboration are suitable for real-time collaboration [30]. CRDT has been successfully applied to different data types in scalable collaborative editing for linear data structure [8], tree document structure data type [6], semi-structured data type [9], and Mind-mapping description [31] as well as for image and video annotations [32] [33]; but not yet on semantic stores with respecting of awareness criteria.

III. PROPOSED APPROACH

Our system is called AB-Set. It is composed of a set of interconnected sites that can dynamically change the collaboration sate. The system uses an optimistic replication [34] and replicates the shared data on a sequence of replicas over the set of sites, where each site hosts a copy and has the same role. When a site modifies its local replica, it generates a corresponding operation. This operation is applied immediately on the local copy, and then it is eventually delivered through the network to all other sites. When received by a remote site, the operation is applied to their local replica as a local operation.

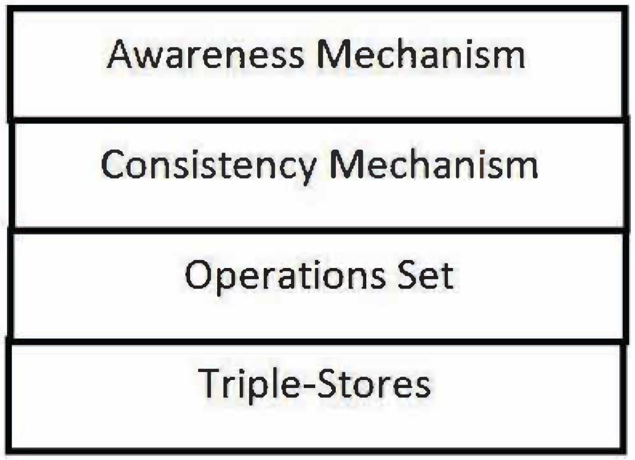


Fig. 2. Main layers of AB-Set

In order to ensure eventual consistency during collaboration sessions, all sites operate according to particular rules and structure which expressed in terms of layers shown from bottom to top. This structure, presented in Figure 2, comprises four layers and illustrates the main elements of the proposed awareness and concurrency architecture for semantic stores collaborative editing.

A. Triple-Stores

A Triple-store is a physical layer of our architecture which includes a set of RDF statements where each RDF statement is represented as the triple (subject, predicate, object) which signifies that the relationship denoted as predicate holds between the concepts denoted as subject and object, where predicate and object are resources or strings.

{subject, predicate ,object } ⇔ predicate(object, subject) witch equivalent of : ∃ object, ∃ subject where predicate(object, subject) .

For instance, the expression "Hafed comes from Algeria" is written according RDF statement definition as:

{Hafed, comesfrom, Algeria} ⇔ comesfrom(Algeria, Hafed).

B. Operations Set

On collaborative editing systems participants can modify the data by performing editing operations, such as insert and delete. An update is considered as a delete of old value followed by an insert of a new value.

In this model, user can insert or delete a triple t by using the following two statements: ADD(t) to create a triple t and DEL(t) to remove it.

The operations defined on AB-Set data structure are:

- L-ADD(t): is a local insert operation generated and executed on the same replica. It inserts a triple.
- D-ADD(t): is a distant insert operation generated and executed on another replica. It inserts a triple.
- L-DEL(t) : is a local delete operation generated and executed on the same replica . It deletes a triple.
- D-DEL(t) : is a distant delete operation generated and executed on another replica. It deletes a triple.

Initially, L-ADD(t) and L-DEL(t) operations are executed locally, and then launch D-ADD(t) and D-DEL(t) to be execute remotely, respectively.

C. Consistency mechanism

To ensure consistency, our idea is based on the CRDT [6] concepts. In the CRDT strategy, the total order is not required and concurrent modifications can be replayed in different orders. Thus, consistency is maintained by defining a data type where all concurrent modifications commute.

Definition 1: A Content of triple is a couple Content =<T, B> where T is a triple and B is a Boolean variable that can take two values, true if the triple is visible and false if the triple is hidden.

The boolean variable presents the visibility of a triple T, which defines the type of the executed operation ADD(t) (insert a triple t) or DEL(t) (remove a triple t). It used to ensure consistency between different replicas that execute concurrent updating operations.

Definition 2: An Optype of triple is used to describe the nature of the executed operation. It can have two values L or D. whilst L means that the operation is locally generated and performed, D means that the operation is distantly executed and has been generated on another replica.

Definition 3: A consistency component denoted by CC is pair of <Optype, content> where Optype corresponds to the type of the executed operation. It can have two values L or D and content is a couple <T, B> where T is a triple and B is a boolean variable signify the visibility of the triple T. CC=< (L, D), <t, (true, false) > >.

For instance: the component (<L, <(SMarwa, Speaks, English), true> means that the triple <Marwa, Speaks, English > has been locally inserted. In the case of delete operation, the component (<D, <(Marwa, Speaks, English), false> means that <(Marwa, Speaks, English) has been remotely removed from the site.

In order to maintain the consistency in all semantic stores, every operation generated and executed on a replica must be executed on all other semantic stores. This requires that every generated L-ADD (t) and L-DEL(t) will be broadcast to the other replicas, after reception on a replica the D-ADD(t) and D-DEL(t) are executed on the local replica of the semantic store.

D. Awareness mechanism

To keep a certain degree of awareness when updating the shared data, the following definitions are presented:

Definition 4: A given awareness operation is the set of information that provides the state of executed operation at any time. It is formalized as Aop=(idUser,time , X) , where idUser is the unique identifier attributed to the user that generated the operation, time is the time of execution of operation and X is a boolean variable witch be true if the content of operation has been updating by our model else it is false.

For instance, the statement <0030, 21/04/2013, 17:41, F> signifies that the identifier of user is 0030, the data and time in which the operation is executed in the local replica is '21/04/2012, 17:41', and F for false indicates that the triple has not been modified by the system.

Definition 5: A given awareness user is the set of information that provides the state of current user at any time. It is formalized as Auser = (idUser, state) , where idUser is the unique identifier attributed to the local user and state is a variable that presents the state connection of the user .

The state of each user is an interesting propriety that can be taken one of the set of the following values (connected, disconnected, absent, occupied). The following state characterizes the way in which the awareness mechanism is built.

- Connect: in this case the user can be aware of the workspace where he edits; where he can receive all information about every replica (<idUser, time, X>, <idUser, connected >)
- Occupy: in this case the user executes an operation, and he can receive all information about every replica (<idUser, time, X>, <idUser, occupied >)
- Disconnect: in this case the user can't be aware of the workspace where he edits because he is out.
- Absent: in this case the replica of this user can be in the same state with the other replicas; and he can receive all operations and information about every site.
- Reconnect: the user in this case is symbolized by (connect; disconnect)

For instance, the pair <0030, connected> signifies that the state of the user identified by 0030 as identifier is connected.

Definition 6: An awareness component denoted by AC is pair of (Aop, Auser), where Aop corresponds to an awareness related to the operation and Auser corresponds to an awareness related to the user that executed the operation. AC = <<idUser, time, X>, < idUser, state>>.

For instance, the association <<0030, 21/04/2013, 17:41, F>, <0030, connected>> indicates that the identifier of user is 0030, the time of executed operation in the local replica is 21/04/2012, 17:41 and F indicates that the triple has not been modified by the system and the state of user is connected.

E. General data structure

AB-Set is an optimistic approach that ensures eventual consistency for set structures without tombstones requirement. It tolerates a huge number of distributed replaces, and supports the awareness mechanisms that we have drawn above. A general data structure of AB-Set is a model of the form:

<CC, AC >, where CC and AC are the consistency and awareness components respectively.

To guarantee that all concurrent users operations commute, we define a set of rules in all possible combination cases. These rules describe the algorithms comportments of remote and local operations executed on a given replica.

Let us consider the following axioms:

- A triple t is locally removed if opType=L and B=false, and it is remotely removed if opType=R and B=false.
- A triple t is locally added if opType=L and B=true, and it is remotely added if opType=R add B=true.
- A triple X is locally initialize X=false.
- L-ADD (t):

If (t is locally added) then DoN( ); //Do Nothing;

else If (t is remotely added) then (S\(<D, <t, true>, <idUserD, Time, X> >)) ∪ (<L, <t, true>, <idUserS, Time, F> >) ;

else If (t is locally removed) then (S\(<L, <t, false>, <idUserS, Time, F> >)) ∪ (<L, <t, true>, <idUserS, Time, F> >);

else If (t is remotely removed) then (S\(<D, <t, false >, <idUserD, Time,X> >)) ∪ (<L, <t, true>, <idUserS, Time, F> >);

else (S ∪ (<L, <t, true >, <idUserS, Time, F> >)); Broadcast(D-ADD(t)).

- D-ADD(t):

  If (t is locally added) then (S\ (<L, <t, true >, <idUserS, Time, F> >)) ∪

  (<D, <t, false>, <idUserD, Time, T> >);

  else If (t is remotely added) then DoN();

  else If (t is locally removed) then (S\(<L, <t, false >,<idUserS, Time,F>)) ∪

  (<D, <t, false>, <idUserD, Time,T> >);

  else If (t is remotely removed) then (S\(<D, <t, false >,<idUserD, Time, X)) ∪

  (<D, <t, true>, <idUserS, Time, F> >);

  else (S ∪ (<D, <t, true>, <idUserD, Time, F> >)).

- L-DEL(t):

  If (t is locally added) then (S\ (<L, <t, true>, <idUserS, Time, F> >)) ∪

  (<L, <t, false >, <idUserS, Time, F> >);

  else If (t is remotely added) then (S\(<D, <t, true>,<idUserD,Time,X> >)) ∪

  (<L, <t, false>, <idUserS, Time, F> >);

  else If (t is locally removed) then DoN();

  else If (t is remotely removed) then (S\(<D, <t, false >,<idUserD, Time,X> >)) ∪

  (<L, <t, false>,<idUserS, Time, F> >);

  else DoN();

  Broadcast(D-DEL(t)).

- D-DEL(t):

  If (t is locally added) then (S\ (<L, <t, true>, <idUserS, Time, F> >)) ∪

  (<D, <t, false>, <idUserD, Time, F>>);

  else If (t is remotely added) then (S\(<D, <t,true >, <idUserD, Time, X>>)) ∪

  (<D, <t, false>, <idUserD, Time, F>>);

  else If (t is locally removed) then (S\(<L, <t, false >,<idUserS, Time, F>>)) ∪

  (<D, <t, false>, <idUserD, Time, F> >);

  else If (t is remotely removed) then DoN();

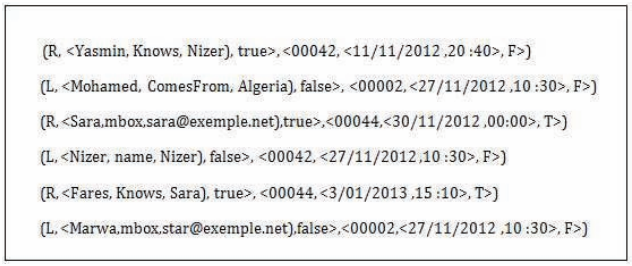
  else DoN().

- DoN(): this function in the case of L-DEL(t) and D-DEL(t) means that the delete operation hasn't effect if this triple hasn't been inserted yet.

- IdUserS: means that type of operation is local (created and executed by the same user).

- IdUserD: means that type of operation is remote (created by another user).

Figure 3 shows an example of the general data structure of AB-set:



(R, <Yasmin, Knows, Nizer), true>, <00042, <11/11/2012 ,20 :40>, F>)

(L, <Mohamed, ComesFrom, Algeria), false>, <00002, <27/11/2012 ,10 :30>, F>)

(R, <Sara,mbox,sara@exemple.net),true>,<00044,<30/11/2012 ,00:00>, T>)

(L, <Nizer, name, Nizer), false>, <00042, <27/11/2012,10 :30>, F>)

(R, <Fares, Knows, Sara), true>, <00044, <3/01/2013 ,15 :10>, T>)

(L, <Marwa,mbox,star@exemple.net),false>,<00002,<27/11/2012 ,10 :30>, F>)

Fig. 3. A sample of AB-Set genaral data structure's

Figure 4 shows a convergence situation with taking into account the awareness of criteria between distributed sites after executing concurrent operations.

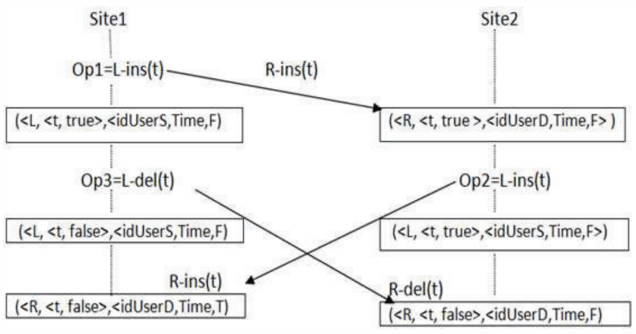

Site1
Site2
Op1=L-ins(t)
R-ins(t)
(<L, <t, true>,<idUserS,Time,F)
(<R, <t, true >,<idUserD,Time,F>)
Op3=L-del(t)
Op2=L-ins(t)
(<L, <t, false>,<idUserS,Time,F)
(<L, <t, true>,<idUserS,Time,F>)
R-ins(t)
R-del(t)
(<R, <t, false>,<idUserD,Time,T)
(<R, <t, false>,<idUserD,Time,F)

Fig. 4. Convergence and support of the awareness after running concurrent modifications.

Let us consider again the scenario of the Figure 1. When all concepts of AB-Set are used, the awareness is maintained and the consistency between all users is ensured, as illustrated in Figure 5; implying a useful solution in maintaining both consistency and awareness criteria.

IV. A SIMPLE USE CASE

We evaluated the performance of our optimistic solution through a real experimentation about a collaborative editing of distributed FOAF (Friend of a friend) [35] dataset. In this study, we edit the FOAF dataset to describe social network relating to Souk-Ahras university students within a virtual organization. For this reason, we have implemented AB-Set using the ARQ API's of the open source JENA Framework [36] that implements the W3C standard SPARQL/Update language for data manipulation (see fig 6).

The AB-Set software was designed to provide users the ability to update the shared data in a flexible manner by integrating a concurrency awareness mechanism. This enables to notify users about the status of distributed datasets.

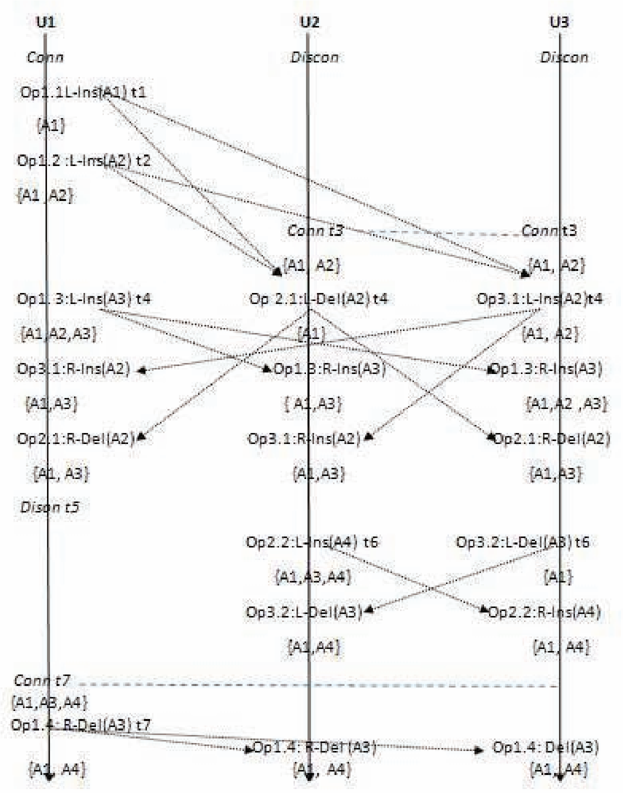
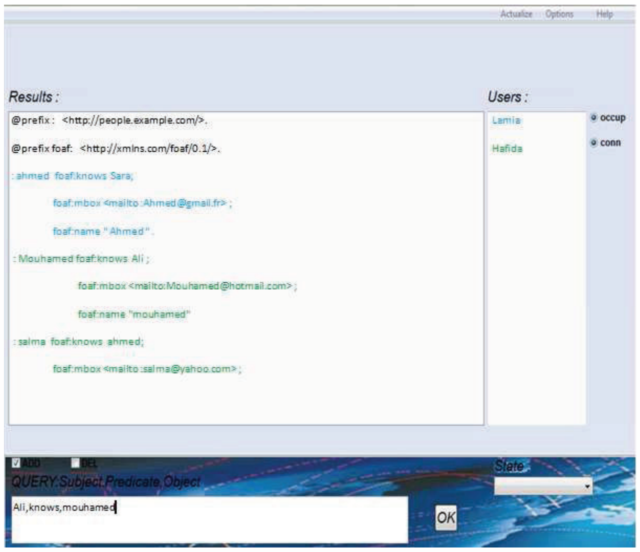


Fig. 5. Consistency after integrating AB-Set concepts.

A typical workflow that portrays how a collaborative editing of distributed FOAF datasets process, is carried out via the AB-Set system as follows:

1) Initially, AB-Set authenticates participants.
2) AB-Set replicates an initial FOAF document by creating a local copy for each participant.
3) AB-Set displays the FOAF documents to all participants and participants may begin to edit.
4) When operations are performed on a local FOAF, they are sent to all others participants.
5) Remote operations are received then integrated to their local copy, AB-Set resolves any possible conflicts
6) Regularly, AB-Set makes participants aware of the different status.



```
@prefix :  <http://people.example.com/>.

@prefix foaf:  <http://xmlns.com/foaf/0.1/>.

:ahmed  foaf:knows Sara;

        foaf:mbox <mailto :Ahmed@gmail.fr> ;

        foaf:name " Ahmed " .

: Mouhamed foaf:knows Ali ;

              foaf:mbox <mailto:Mouhamed@hotmail.com> ;

              foaf:name "mouhamed"

: salma  foaf:knows  ahmed;

        foaf:mbox <mailto :salma@yahoo.com> ;
```

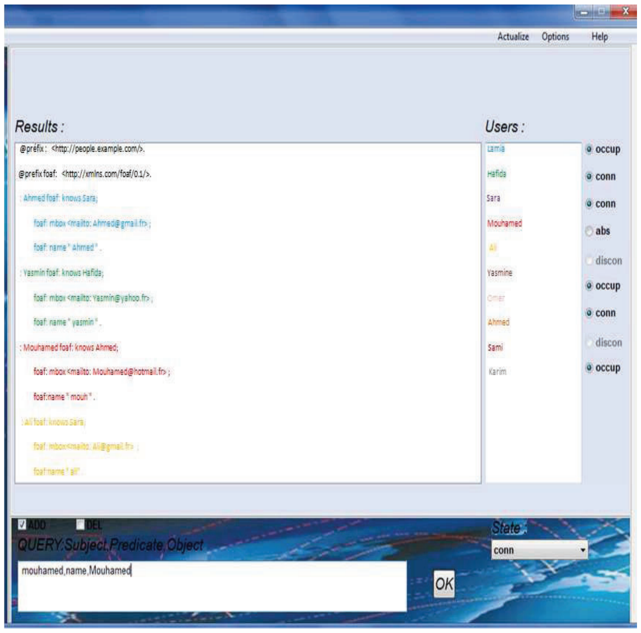
Lamia
Hafida

occup
conn

[✓ ADD] DEL

Ali,knows,mouhamed

Fig. 6. A prototype of AB-Set showing when two users are working collaboratively in shared data, the different colors are used to facilate the awareness about users contributions.



```
@préfix :  <http://people.example.com/>.

@prefix foaf:  <http://xmlns.com/foaf/0.1/>.

: Ahmed foaf: knows Sara;

    foaf: mbox <mailto: Ahmed@gmail.fr> ;

    foaf: name " Ahmed " .

: Yasmin foaf: knows Hafida;

    foaf: mbox <mailto: Yasmin@yahoo.fr> ;

    foaf: name " yasmin " .

: Mouhamed foaf: knows Ahmed;

    foaf: mbox <mailto: Mouhamed@hotmail.fr> ;

    foaf:name " mouh " .

: Ali foaf: knows Sara;

    foaf: mbox<mailto: Ali@gmail.fr> ;

    foaf:name " ali" .
```

Lamia
Hafida
Sara
Mouhamed
Ali
Yasmine
Omer
Ahmed
Sami
Karim

occup
conn
conn
abs
discon
occup
conn
discon
occup

[✓ ADD] DEL

conn

mouhamed,name,Mouhamed

Fig. 7. An example of collaborative editing of shared FOAF dadaset by serveral users.

Figure 7 shows an example of collaborative editing of shared FOAF dadaset by serveral users. Each user has his/her own status and the contribution of each one is distinguished by a specific color.

V. Discussion

AB-Set is a CRDT data type for distributed semantic stores that guarantees eventual consistency. Additionally, it provides a powerful awareness mechanism that informs participants about states of shared data stores. States indicate when a store replica is locally updated, when two replicas of the store are updated or when a store replica is updated locally and some modifications on that document were committed. The proof that AB-Set ensures convergence is straightforward. Since a boolean variable is attached to each triple and the new value of this variable depends of the set of rules of basic conjunction and disjunction operation according to operation type, such as illustrated above. All pairs of operations commutes, thus, AB-Set ensure eventual consistency. Unlike previous approaches, AB-Set does not require either causal relation from the underlying network or tombstones, eliminating the burden of garbage collection. However, the experiments are currently limited to small users' community with some operations for each one. Therefore, we need to make more complex experiments to establish the scalability and efficiency of the method in presence of huge data

VI. Conclsuion

In this study, we have developed a new approach called AB-Set that focuses on optimistic replication technique to allow a higher availability and performance in distributed collaborative system for semantic stores. AB-Set data structure is designed as a CRDT for replicating, sharing, and ensuring eventual consistency of data as well as providing an efficient awareness mechanism for understanding the activities of all collaborators. A prototype implementation of the proposed approach, for the collaborative FOAF editing, has also been presented and discussed. Consequently, the overall results of the presented use case were satisfactory.

We plan to test and understand users' perceived attitudes towards the use of the prototype that implements AB-Set. We plan also to design several scenarios to evaluate the usability of our solution in more real cases such as collaborative learning and collaborative semantic annotation. Finally, we think about adding others factors to the awareness mechanism for supporting the disconnection mode, particularly when a user reconnect after being disconnected.

References

[1] X. Peng, M.A. Babar, and C. Ebert, "Collaborative software development platforms for crowdsourcing. " IEEE software, vol. 2, pp. 30-36, 2014.

[2] T. Berners-Lee, J. Hendler and O. Lassila, "The Semantic Web," Scientific Am., vol. 284(5), pp. 28-37.

[3] Resource Description Framework, http://www.w3.org/RDF/, accessed 01 June 2015.

[4] SPARQL 1.1 Update, http://www.w3.org/TR/sparql11-update/, accessed 01 June 2015.

[5] J.M. Carroll, D. C. Neale, P. L. Isenhour, M. B. Rosson, and S. D. McCrickard "Notification and awareness: synchronizing task-oriented collaborative activity", International Journal of Human-Computer Studies, vol. 58, pp.605 -632, 2003.

[6] N. M. Preguic J. M. Marques, M. Shapiro, and M. Letia, "A commutative replicated data type for cooperative editing", International Conference On Distributed Computing Systems, ICDCS, IEEE Computer Society, pp.395-403, 2009.

[7] M. Shapiro, N. Preguica, C. Baquero, and M. Zawirski, "A comprehensive study of Convergent and Commutative Replicated Data Types", Research Report RR-7506, INRIA, January 2011.

[8] S. Weiss, P. Urso, and P. Molli, "Logoot-Undo: Distributed Collaborative Editing System on P2P Networks", IEEE Transactions on Parallel and Distributed Systems, vol. 21, no. 8, pp.1162-1174, 2010.

[9] H. Zarzour, and M Sellami, "srCE: a collaborative editing of scalable semantic stores on P2P networks", International Journal of Computer Applications in Technology, vol. 48, no.1, pp. 1-13, 2013.

[10] H. Zarzour, and M. Sellami, "B-Set: a synchronization method for distributed semantic stores", IEEE International Conference on Complex Systems, ICCS'12, Published on Xplore IEEE, Agadir, Maroc, November 5 - 6, 2012.

[11] S. Martin, P. Urso, and S. Weiss, "Scalable XML Collaborative Editing with Undo", Proc. International Conference on Cooperative Information System, CoopIS, 2010.

[12] I. Steinmacher, A. P. Chaves, and M. A. Gerosa, "Awareness support in distributed software development: A systematic review and mapping of the literature", Computer Supported Cooperative Work, vol. 22, pp. 113-158, 2013.

[13] P. Dourish and V. Bellotti "Awareness and Coordination in Shared Workspaces", Proceedings of the ACM Conference on Computer-Supported Cooperative Work, pp.107 -114, 1992.

[14] C.A. Ellis, S.J. Gibbs, and G.L. Rein, "Groupware: Some issues and experiences", Communications of the ACM, vol. 34, no. 1, pp.38 -58 1991.

[15] C. Gutwin, R. Penner, and K. Schneider, "Group Awareness in Distributed Software Development", ACM CSCW, pp. 72 -81, 2004.

[16] S.D. Scott, T. C. Graham, J. R. Wallace, M. Hancock, and M. Nacenta, "Local Remote Collaboration: Applying Remote Group AwarenessTechniques to Co-located Settings", In Proceedings of the 18th ACM Conference Companion on Computer Supported Cooperative Work & Social Computing. ACM, pp. 319-324, 2015.

[17] C. Souza and D. Redmiles, "The awareness network: To whom should i display my actions? and, whose actions should i monitor?" in ECSCW 2007, L. Bannon, I. Wagner, C. Gutwin, R. Harper, and K. Schmidt, Eds. Springer London, pp. 99-117, 200.

[18] A. Sarma, Z. Noroozi and A. Van Der Hoek, "Palantir: Raising Awareness among Configuration Management Workspaces," Proc. Int',l Conf. Software Eng., pp. 444-453, 2003.

[19] A. Doucette, C. Gutwin, and R. Mandryk, "Effects of arm embodiment on implicit coordination, co-presence, and awareness in mixed-focus distributed tabletop tasks," In Proceedings of the 41st Graphics Interface Conference, Canadian Information Processing Society. pp. 131-138, 2015.

[20] P. Molli, H. Skaf-Molli and C. Bouthier "State Treemap: an Awareness Widget for Multi-Synchronous Groupware", Proceedings of the Seventh International Workshop on Groupware, 2001.

[21] S. Papadopoulou, C. L. Ignat, G. Oster, and M. C. Norrie. Increasing Awareness in Collaborative Authoring through Edit Profiling. In Proceedings of the IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing - CollaborateCom 2006, Atlanda, USA, Nov. 2006.

[22] K. Aslan, N. Alhadad, H. Skaf-Molli, and P. Molli, "SCHO: An Ontology Based Model for Computing Divergence Awareness in Distributed Collaborative Systems." ECSCW 2011: Proceedings of the 12th European Conference on Computer Supported Cooperative Work, 24-28 September 2011, Aarhus Denmark. pp. 373-392, 2011.

[23] S. Greenberg, and M. Boyle, "Generating custom notification histories by tracking visual differences between web page visits," In Proceedings of Graphics Interface, Canadian Information Processing Society, pp. 227-234, 2006.

[24] C.A. Ellis, and S.J. Gibbs, "Concurrency control in groupware systems", ACM International Conference on Management of Data, ACM, pp. 399-407, 1989.

[25] C. Sun, and C.S. llis, “Operational transformation in real-time group editors: issues, algorithms, and achievements”, ACM Conference on Computer Supported Cooperative Work, ACM, pp. 59-68, 1998.

[26] Sun, C., Jia, X., Zhang, Y., Yang, Y., and Chen, D. (1998) `Achieving Convergence, Causality Preservation, and Intention Preservation in Real-Time Cooperative Editing Systems', ACM Transactions on Computer-Human Interaction, Vol. 5(1), pp. 63-108.

[27] M. Suleiman, M. Cart, and J. Ferri, “Concurrent operations in a distributed and mobile collaborative environment”, International Conference on Data Engineering, IEEE Computer Society, pp. 36-45, 1998.

[28] N. Vidot, M. Cart, J. Ferri, and M. Suleiman, “Copies convergence in a distributed real-time collaborative environment”', ACM Conference on Computer Supported Cooperative Work, pp. 171-180, 1998.

[29] M. Cart, and J. Ferri, “Asynchronous reconciliation based on operational transformation for p2p collaborative environments”, International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom, IEEE Computer Society, pp 127-138, 2007.

[30] M. Ahmed-Nacer, C.L. Ignat, G. Oster, H.G. Roh, and P. Urso, `Evaluating CRDTs for real-time document editing', ACM Symposium on Document Engineering, pp 103-112, 2011.

[31] H. Zarzour, T. Abid, and M. Sellami, “Conflict-free collaborative decision-making over Mind-Mapping”, 4th International Conference on Advanced Computing & Communication Technologies, ACCT’14, Rohtak, India, February 8-9, 2014.

[32] H. Zarzour and M. Sellami, “Achieving consistency in collaborative image annotation systems,” in Proc. IEEE 5th International Conference on Information and Communication Systems, Irbid, Jordan, April 1-3, 2014.

[33] H. Zarzour and M. Sellami, “Using commutative replicated data type for collaborative video annotation,” in Proc. IEEE 4th International Conference on Multimedia Computing and Systems, Marrakesh, Morocco, April 14-16, 2014.

[34] Y. Saito, and M. Shapiro, “Optimistic replication”, ACM Computing Surveys, ACM, vol.37(1), pp:42–81, 2005.

[35] http://www.foaf-project.org/ , accessed 01 June 2015.

[36] Apache Jena, https://jena.apache.org/, accessed 01 June 2015.