2014 Fourth International Conference on Advanced Computing & Communication Technologies

http://ieeexplore.ieee.org/document/6783506/ DOI: 10.1109/ACCT.2014.34

Conflict-free collaborative decision-making over Mind-Mapping

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

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*—As the integration of disaster management, distributed computing, and collaborative technologies, collaborative disaster management systems can overcome the challenges, such as data replication and updating, real-time information dissemination, distributed resources sharing, and collaborative decision-making. Collaborative decision-making systems over Mind-Mapping are modeled by a set of sites connected by a communication network with each site hosting a replica of shared data. When a site executes a modification, it generates a corresponding operation that is performed immediately on its local copy, and then is propagated to all sites in order to be executed remotely. However, if the consistency is not properly guaranteed it can lead to divergences, practically, in the case of concurrent decisions over the shared Mind-Mapping. This paper aims to provide a solution to the problem of consistency in collaborative decision-making over Mind-Mapping. This solution allows distributed users to work together to reach a common goal without central servers and coordination among them is achieved in a Peer-to-Peer manner. A prototype developed about nuclear disaster scenario demonstrates the effectiveness of our approach.**

Keywords*— disaster management; collaborative editing; collaborative decision-making; distributed system; mind-mapping; consistency.

I. INTRODUCTION

Over the last few years, there has been a continuous evolution in the practice of disaster management. In disaster management system it is generally understood that situations may be complicated and a challenge to manage effectively, but that a set of conflicts will eventually be resolvable [1, 2]. As the integration of disaster management domain, distributed computing, and collaborative technologies, collaborative disaster management systems [3] can overcome the challenges, such as data replication and updating, real-time information dissemination [4], distributed resources sharing, distributed data editing [5] and Collaborative decision-taking [6]. Therefore, collaborative decision-taking systems provide a new means of organizing and sharing data. Collaborative decision-making systems, in which modern communication technologies allow distributed users from around the world to work on the same data in order to construct identical decision about a common subject by different viewpoints and skills. Collaborative decision-making systems are fundamental to all organizational processes, and have been the subject of research in management and in decision and computer sciences for years [7, 8].

Mind-Mappings [9, 10] are special kinds of document representations which have not been utilized for collaborative decision-making on distributed networks before, to the best of our knowledge. The basic mechanism of collaborative decision-making system over Mind-Mapping is modeled by a set of sites connected by a communication network with each site hosting a replica of shared data. When a site executes a modification, it generates a corresponding operation that is performed immediately on its local copy, and then is propagated to all sites in order to be executed remotely.

The major benefits of such systems include reducing task completion time, reducing errors, getting different viewpoints and skills, and obtaining an accurate decision. Moreover, these systems offer flexibility and convenience where it is easy for users to contribute from anywhere and anytime in the world with effective and efficient work processes that help in developing different decisions via structured formalism [11]. A collaborative decision-making system can stand as a collaborative editing system where decisions are characterized by a sequence of operations.

Recently, a Commutative Replicated Data Type (CRDT) [12, 13] is invented as a convergence philosophy that ensures consistency maintenance of replica in a collaborative editing system without any difficulty over distributed networks. This method supposes that all concurrent operations commute [14]. During collaborative decision-making the consistency is very important and requires many rules. However, if the consistency is not properly guaranteed it can lead to divergences, practically, in the case of concurrent decisions over the shared Mind-Mapping, Thus, the shared Mind-Mapping which replicated in all sites diverges and the same sequence of operations generates inconsistent results. For instance, an inconsistent system used in order to avoid natural or human disasters by taking collaborative decision will produce dangerous consequences. .

In this paper, we propose a novel approach called CMM. CMM (Collaborative Mind-Mapping) is a new CRDT designed for Mind-Mapping data type to ensure eventual consistency in collaborative decision-making systems. The main idea of CMM is to define a set of operations on a specified structure for an optimistic Mind-Mapping in a way that all concurrent operations generate the identical result when the system is idle. CMM method is designed not only for collaborative decision-making but further for supporting concurrent editing operations at large scale. A prototype of CMM is designed and implemented as an extension to FreeMind[15] to support scalable conflict-free collaborative decision-making.

The rest of this paper is organized as follows: Section 2 reviews the backgrounds and the most recent related work. Section 3 details the proposed solution CMM. Section 4 contains the experimental results aver a real scenario. Finally, section 5 concludes the paper and describes some future research directions.

II. Backgrounds and Related Work

A. Mind-Mapping

Mind-Mapping concept was originally invented by Tony Buzan [14] and is nowadays became an interesting research area by the use of large number of organizations in different domains including document editing, brainstorming, project planning, note taking and decision making.

A Mind-Mapping is a graphical representation used to represent ideas, concepts, words, tasks or other notions linked to and arranged radially around a main key concept. It is used to generate, visualize, structure and classify ideas, and as an aid in study, organization, problem solving, decision making, and writing [9].

In general, a Mind-Mapping is a diagram that includes a set of ideas and always has a central item which corresponds to the central idea the mind mapping is about. From the central item child-items cover sub-topics. The major benefits of Mind-Mapping utilization include: (1) Mechanism adapted to our biology [16], (2) Ideas link from around central idea [17], (3) open and flexible system [18], (4) Support for increasing productivity [19].

Several studies were published about creating and evaluating Mind-Mappings, for instance, in the fields of philosophy, aeronautic, and education but not yet in collaborative decision-making systems that integrate concurrent operations within virtual community.

Figure 1 shows an example of Mind-Mapping about a concept of social networking and a relationship between its uses and types.

Nowadays, many tools exist to support the Mind-Mappings creation. The most popular of them are MindManager [20] and FreeMind [15]. To create Mind-Mappings, tools support the following actions: inserting of concepts or attributes to a node, linking a node with another, and removing a node from the graph.

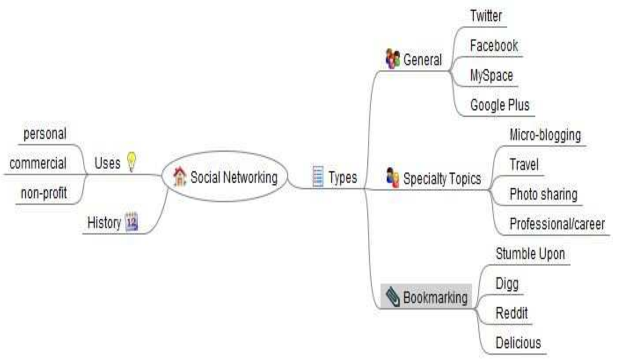

Fig. 1. A Mind-Mapping example of social networking

B. Consistency maintenance

For the context of consistency maintenance in distributed systems, several algorithms have been proposed based on Operation Transformation (OT) approach [21] such as GOTO [22], GOT [23], SOCT2 [24], SOCT4 [25], MOT2 [26]. Because of the use of victor clocks, such are algorithms are known for their inability to scale as well as the fact that their correctness is hard for verification [12]. This is mainly because remote operations are inefficient as well as history buffers are likely to grow for larger memberships. SOCT2 [24] is typically peer to peer collaborative environment that ensures the CCI [23] model. However, SOCT2 is designed only for text document structure. Further, there are no transformation functions for Mind-Mapping data are available.

Recently, CRDT [12] approach 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 [27]. CRDT has been successfully applied to different data representations types in scalable collaborative editing for linear data type (text document) [28], tree document structure data type [12],semi-structured data type [29] and set data type [11] but not yet on Mind-Mapping data type having a specific structure.

III. Proposition

In this section, we present our model, which aims to provide a solution to the problem of consistency in collaborative decision-making over Mind-Mapping within a virtual organization. It allows distributed users to work together to reach a common goal without a central servers and coordination among them is achieved in a P2P manner. The collaborative decisions of users are interpreted by a sequence of editing operations executed on a shared Mind-Mapping card. In order to achieve an eventual consistency after each concurrent decision, we define a new CRDT for Mind-Mapping data type where all concurrent decision commute without any requirement of merge algorithms or integration function, so the copies of all shared data converge systematically without central and complex control. Our System is based on an optimistic replication [30] and composed by a set of interconnected sites. The Mind-Mapping edited by users are replicated on each site having the same role and hosting the shared data. A site hosts a copy of Mind-Mapping, called a local replica and, can edit his copy by generating a set of updating operations. The updating is firstly performed on a local replica, and then it is eventually broadcasted to all other remote replicas. When a modification is received by a site, the modification is applied immediately. When the system is idle, all replicas converge and the obtained Mind-Mapping is identical in all sites of the network. In our context, no assumptions are made concerning the broadcast time of updating operations.

A. Data Structure

The Mind-Mapping is presented in a hierarchical structure that permits easy navigation from higher root level data abstractions to lower level details. In our context, a Mind-Mapping is created around a single problem and simulates a hierarchical tree structure, with a root value and sub-trees of children, represented as a set of linked nodes, where each node is a data consisting of idea and concept. In other words, a concept is assigned to each node of the considered tree.

From an organizational point of view, a root of tree corresponds to a central concept and is linked via lines to other ideas which in turn are linked with other associated concepts. By convention, the Mind-Mapping is read in clockwise direction which was done in accordance with the right-to-left reading direction. Each node can be accessed by a unique identifier which serves as a key. The key permits to indicate the position of the current node in the Mind-Mapping in relation to the main idea or central concept (see Figure 2 and 3).

Therefore, a node in the map is formally described by a pair (id, lbl), where id is the unique identifier of the current node position in which there is the label denoted by lbl. The value of any id associated with any node implicitly defines the path leading to it from the root of the card.

In this study we use the open source software FreeMind [15]. FreeMind is a free Mind-Mapping application written in Java and is licensed under the GNU General Public License. It provides extensive export capabilities.

Figure 2 shows an example of graphical description on a nuclear disaster using FreeMind, while the Figure 3 shows the representation of the same Mind-Mapping describing the nuclear disaster by showing only unique identifiers that characterize each node in our data structure.

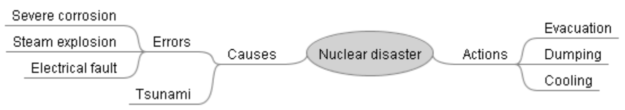

Fig. 2. An example of nuclear disaster reprsentation using Mind-Mapping

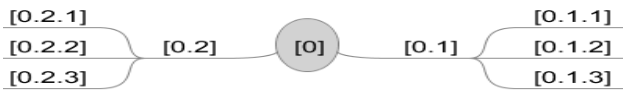

Fig. 3. Identifiers nodes sample

B. Mind-Mapping Updating

On collaborative editing systems users can modify the data by performing editing operations, such as insert and delete. To allow collaborative decision-making over shared Mind-Mapping, we define two basic operations:

- *InsNode (p, n, t):* creates and adds a node whose attribute value is t at the nth child of the node p of the Mind-Mapping. So, if we need to insert a node at the second position from the root, it's just we perform the operation: insNode ([0], 2, 'Mind').
- *DelNode (p, n):* removes the nth child of the node p from the Mind-Mapping. So, if we need to delete a node with position 2 from the root, it's just we perform the operation: delNode ([0], 2).

The updating operation of a given attribute can be considered or made equivalent as a delete of the existing value to be updated followed by an insert of the new value of the node. Indeed, this sequence of update changes the value of the attribute t of the node identified by the position n and the father p of the map. So, if we need to modify the value of a node with position 1 from the root, it's just a sequence serial of operations are executed: delNode ([0], 1) followed by insNode ([0], 1 , 'new value').

C. Algorithms

To maintain eventual consistency between distributed replicas, all concurrent operations must commute, i.e., whether the operations execution order on different sites, the final result must be the same when the system is idle. Commutativity of operations is one of the mains of the method CRDT. In our case, it is important to point out that we define the notions of commutativity as binary relations on operations in the sense of our formal definition, rather than simply for invocations as is usually done. There are four possible combinations of concurrent operations pairs which are respectively: (1) (insNode(p1,n1,t1), insNode(p2,n2,t2)), (2) (insNode(p1,n1,t1) ,delNode(p2,n2,t2)), (3) (delNode(p1,n1) , insNode(p2,n2,t2)) ,and (4) (delNode(p1,n1), delNode(p2,n2)).

We now detail the behavior of functions ensuring the operations commutativity. To do this, we use the following conventions: functionName (OD, OL) , where functionName is the name of the function with two concurrent operations, OD is an unexecuted remote operation, and OL is a local operation that has already been executed. By definition, both OD and OL operations are commutative if and only if OD >> OL = OL>> OD where the symbol >> denotes the precedence relation which means that when OD is executed before or after OL, this leads at any time to the same result. In what follows, we present the set of algorithms that ensures the commutativity of all operations pairs. The execution of two successive insert operations which permit to add both new nodes, it may lead to a conflict situation and generate divergent results. To overcome this problem, the procedure insCommute() outlined in Figure 4 which is defined mainly to guarantee the commutativity between given insert operations. The procedure takes an operation insNode(p1,n1,t1) as insNode(p2,n2,t2) as an input argument. its behavior can be summarized as follows: if both elements to be insert share the same parent and will be inserted at the same position, we check if it is the same attribute. If this is the case, nothing is done (execution of the operation DoNothing ()), otherwise it uses the lex() function for t1 and t2, which returns the lexicographical order of the attribute according to which the attribute is inserted at position n1+1 or n1.

```
insCommute(insNode(p1,n1,t1), insNode(p2,n2,t2) ){
if (p1=p2){
  if(n1=n2){
     if(t1=t2) {doNothing()}
       else if (lex(t1)> lex(t2)){insNode(p1,n1+1,t1)}
            else {insNode(p1,n1,t1)}
  }else if(n1>n2) {insNode(p1,n1+1,t1)}
       else{
         if(n1<n2) {insNode(p1,n1,t1)}
       }
}else {
 insNode(p1,n1,t1)
  }
```

Fig. 4. Consistency function for insert operations

Deleting and inserting nodes are also considered as concurrent operations that can lead to a conflict situation. The algorithms outlined in Figures 5 and 6 provide a solution to this problem regardless to the execution order in sites.

```
insDelCommute(insNode(p1,n1,t1), delNode(p2,n2)){
if (p1=p2){
  if(n1=n2){
     doNothing()
  }else if(n1>n2) {insNode(p1,n1-1,t1)}
       else{
         if(n1<n2) {insNode(p1,n1,t1)}
       }
}else {
 insNode(p1,n1,t1)
  }
```

Fig. 5. Consistency function for insert and delete operations

```
delInsCommute(delNode(p1,n1),insNode(p2,n2,t2) ){
if (p1=p2){
  if(n1=n2){
                          doNothing()
  }else if(n1>n2) {delNode(p1,n1+1)}
      else{
        if(n1<n2) {delNode(p1,n1,t1)}
      }
}else {
 delNode(p1,n1)
  }
```

Fig. 6. Consistency function for insert and delete operations

```
delDelCommute(delNode(p1,n1),delNode(p2,n2) ){
if (p1=p2){
  if(n1=n2){
                          doNothing()
  }else if(n1>n2) {delNode(p1,n1-1)}
      else{
        if(n1<n2) {delNode(p1,n1)}
      }
}else {
 delNode(p1,n1)
  }
```

Fig. 7. Consistency function for delete operations

If both elements share the same parent and position, then one of the insert or delete operations are cancelled by performing null operation DoNothing().

If both operations have the same path and the position of n1 is strictly lower than n2, then the result of the integration is due to the execution of the invoked operation that corresponds to the node identified by p1 and n1-1 or n+1 according to the type of the operation. Otherwise, it performs the operation as it is outside modification.

Therefore, it is important to note that the algorithms presented in Figures 5 and 6 lead to the identical result as they are developed on the idea of commutative operations between the couple (insNod () delNode), where the order of execution of operations is not required to maintain consistency of the Mind-Mapping.

To avoid inconsistency, the last case couples two delete operations that are running successively. The algorithm outlined in Figure 7 illustrates how to take into account the competitive aspect based on the principle of commutativity between two removal operations; the first one is local whilst the second is remote.

IV. EXPERIMENTATION

Successful disaster preparedness occurs through strong collaboration, detailed and well-understood plans of action, and written agreements in place before a disaster occurs. We validated and evaluated the performance of our optimistic solution through a real experimentation about

to the identified issues. Each expert has its own Mind-Mapping, Site 1 for the first expert and Site 2 for the second.

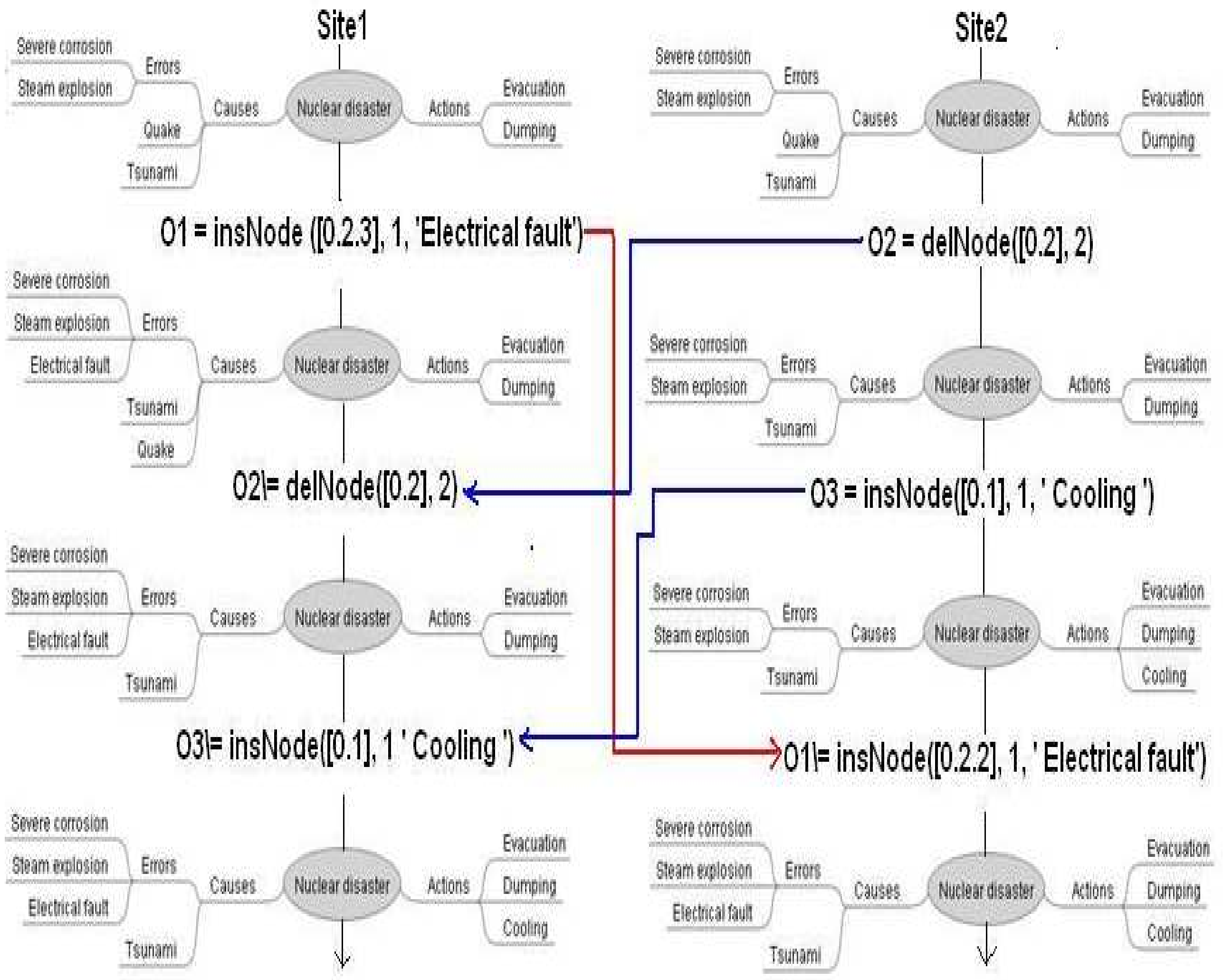

Fig. 8. Consistency and convergence of Mind-Mapping replicas after integrating of conccurrent operations

nuclear disasters. A nuclear disaster is an event that has led to significant consequences to people, the environment or the facility with tremendous amounts of nuclear radiation. The most famous examples are Chernobyl and Fukushima disasters which occurred respectively in 1986 in Ukraine and in 2011 in Japan. In fact, a prototype of CMM is designed and implemented as an extension to FreeMind. With CMM model, existing FreeMind becomes collaborative, flexible, more efficient, and can be deployed on distributed environment.

The scenario proceeds as follows:

Two experts collaborate in the diagnostic of the nuclear disaster causes as well as the set of actions to provide solutions

The initial Mind-Mapping is shared by both experts at the beginning of the collaborative session. The first expert inserts the node whose label is "Electrical fault" at position 1 of the branch "Errors" executing the operation O1 = insNode ([0.2.3], 1, 'Electrical fault'). The second expert generates two consecutive operations O2 = delNode([0.2], 2) to remove the second node of "Causes" and the operation O3 = insNode([0.1], 1, ' Cooling ') to insert the node “Cooling ”at the first position in the " Actions" branch, as shown in Figure 8.

In order to obtain consistent Mind-Mapping at both sites, each expert broadcasts their local and concurrent operations mutually. At Site 1, O2 is performed over O1 to give O2\ = delNode([0.2],2). In the same way, O3 is performed with preservation of $O2^{\backslash}$ to produce $O3^{\backslash}$= insNode([0.1], 1 ' Cooling '). As for O1, it is executed on site 2 over O3 to provide $O1^{\backslash}$= insNode([0.2.2], 1, ' Electrical fault'). It is noted that the new obtained Mind-Mappings from both experts converge to the same version after integrating the algorithms of our solution. Thus, the final results are identical regardless the operation execution order at sites.

V. Discussion

CMM is a CRDT for Mind-Mapping that supports collaborative decision-making and ensures eventual consistency. The proof that MMC ensures convergence is straightforward. Since nodes identifiers are unique, and operations are performed according to consistency program fragments presented above, the different sites can executes any sequences of delete and insert operations in any order and obtain an identical result.

Unlike previous approaches, CMM does not require either causal relation from the underlying network or tombstones, eliminating the burden of garbage collection. However, experiments are currently limited to two users with some operations. Therefore, we need to make more complex experiments to establish the scalability and efficiency of the method in presence of huge data.

VI. Conclusion

The demands of collaborative technologies are central to effective disaster management for relief organizations. Therefore, disaster management systems have changed direction and now recent studies are trying to integrate collaborative aspect in order to switch to a new generation of framework that is called collaborative disaster management, especially for collaborative decision-making. In fact, the proposed approach aims to provide a solution to the problem of consistency in collaborative decision-making over Mind-Mapping within a virtual organization. It allows distributed users to work together to reach a common goal without a central servers and coordination among them is achieved in a P2P manner. A prototype is implemented about nuclear disaster scenario as an extension to FreeMind that supports concurrent operations. The experimental results have demonstrated the effectiveness of our approach. In this regard, this study is the first to present a new CRDT for a Conflict-free collaborative decision-making over Mind-Mapping. For future work, we plan to deploy our model on a cloud computing infrastructure and to replay the experiment on a large community and other scenarios.

References

[1] M. Careem, C. De Silva, R. De Silva, L. Raschid, and S. Weerawarana, "Sahana: Overview of a disaster management system", International Conference on Information and Automation, IEEE Computer Society, pp. 361-366, 2006.

[2] L. Zheng, C. Shen, L. Tang, T. Li, S. Luis, and S.C. Chen, "Applying data mining techniques to address disaster information management challenges on mobile devices", 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp. 283-291, 2011.

[3] N. Khamis, A.M. Misfian, and R. Md Noor, "Towards sustainable software criteria: Rescue operation and disaster management system model", 10th IEEE International Conference on Networking, Sensing and Control, IEEE Computer Society, pp. 398-403, 2013.

[4] A. Utani, T. Mizumoto, and T. Okumura, "How geeks responded to a catastrophic disaster of a high-tech country: rapid development of counter-disaster systems for the great east Japan earthquake of March 2011", Special Workshop on Internet and Disasters. ACM, pp. 9, 2011.

[5] C. Sun, and D. Chen, "Consistency maintenance in real-time collaborative graphics editing systems", ACM Transactions on Computer-Human Interaction, vol. 9, no 1, pp. 1-41, 2002.

[6] N. Karacapilidis, and D. Papadias, "Computer supported argumentation and collaborative decision making: the HERMES system". Information systems, vol. 26(4), pp. 259-277, 2001.

[7] S. Kim, A Godble, R. Huang, et al, "Toward an integrated human-centered knowledge-based collaborative decision making system", IEEE International Conference on Information Reuse and Integration, IEEE Computer Society, pp. 394-401, 2004.

[8] C. Chen, and A. Chen, "An expert decision-making strategy based on collaborative cloud system", IEEE International Conference on Automation Science and Engineering, IEEE Computer Society, pp. 777-781, 2012.

[9] T. Buzan, Use Both Sides of Your Brain: New Mind-Mapping Techniques, Third Edition. Plume, 1991

[10] P.C. Shih, D.H Nguyen, S.H Hirano, D.F. Redmiles, and G.R. Hayes, "GroupMind: supporting idea generation through a collaborative mind-mapping tool", International conference on Supporting group work, ACM, pp. 139-148, 2009.

[11] 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.

[12] 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.

[13] H. Zarzour, and M Sellami, "B-Set: A synchronization method for distributed semantic stores", International Conference on Complex Systems, ICCS'12, IEEE Computer Society, November 5-6, 2012.

[14] H. Zarzour, and M Sellami, "p2pCoSU: A P2P Sparql/update for collaborative authoring of triple-stores", 11th International Symposium on Programming and Systems, ISPS'13, IEEE Computer Society, pp p. 128-136, 2013.

[15] FreeMind, http://freemind.sourceforge.net, 2013.

[16] B. Toni, Making the Most of your Mind, Pan Books, 1977.

[17] Z. Yan-lei, X. Shuang-jiu, T. Xu-bo, and D. Lei, "Mind Mapping Based Human Memory Management System", International Conference on Computational Intelligence and Software Engineering, IEEE Computer Society, pp. 1-4, 2010.

[18] I. Mahmud, and V. Veneziano, "Mind-mapping: An effective technique to facilitate requirements engineering in agile software development", 14th International Conference on Computer and Information Technology, IEEE, pp. 157-162, 2011.

[19] T-T. Kionga, J-M. Yunosb, B. Mohammad, W. Othmand, Y-M. Heonga, M-M. Mohamada, "The Development and Evaluation of the Qualities of Buzan Mind Mapping Module", Procedia - Social and Behavioral Sciences, Vol.59(17), pp. 188–196, 2012.

[20] MindManager, http://www.mindjet.com, 2013.

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

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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(8), pp.1162-1174, 2011.

[29] S. Martin, P. Urso, and S. Weiss, “Scalable XML collaborative editing with undo”, International Conference on Cooperative Information System, CoopIS, 2011,

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