ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

Cloud Based CMIS with Offline Synchronization

Rekha M[1], Bharathi K[2], Cynthia A[3], Noorun Nazira[4]

Assistant Professor, Dept. of Information Science Engineering, Vemana Institute of Technology, Bengaluru, India[1]

U.G. Student, Dept. of Information Science Engineering, Vemana Institute of Technology, Bengaluru, India[2,3,4]

ABSTRACT: Performing offline synchronization betweendifferent databases is the most challenging task. The lack of a standard interface to content management systems made it difficult to integrate content from multiple repositories into a single application such as a portal, CRM system, or office desktop. We overcome this problem by creating an abstract layer using CMIS .The synchronization takes place with the help of CouchDB replication protocol. CMIS is a vendor neutral, language independent specification for working with ECM systems. The main benefit of CMIS is that the developers do not have to learn a new API every time they encounter a new type of repository. PouchDB enables applications to store data locally while offline, then synchronize it with CouchDB and when the application is back online, keeping the user's data in sync no matter where they next login.

KEYWORDS: Content management interoperability services (CMIS), Enterprise content management (ECM), Application Programming Interface (API).

I. INTRODUCTION

CMIS describes a domain model plus bindings that can be used by applications to handle the content stored in a repository. CMIS provides a common data model covering typed files and folders with generic properties that can be set or read. There is a set of services for adding and fetch documents (objects). There may be an access control system, a checkout and version control facility, and the ability to define generic relations. Protocol are defined using WSDL, SOAP, AtomPub, and JSON.

The CMISserver is built using common architectures of document management systems. The CMIS is implemented in many languages and provides an API that is programming language-agnostic, as REST and SOAP. PouchDB is a database that stores data locally, so that the users can enjoy all the functions of an application even when they are offline and the data is synchronized between clients, so wherever the user goes they can stay up-to-date. PouchDB is written in javascript is an open source project. PouchDB can also beexecuted in NodeJS and can be used as a direct interface to CouchDB servers.

The CouchDB Replication Protocol is used for synchronising JSON documents between any two databases over HTTP/1.1 by using the public CouchDB REST API and is based on the Apache CouchDB MVCC Data model. CMISconnectsdisparate repositories through a service-oriented interface, as shown in fig.1 CMIS is implemented by each ECM system to work with the content and metadata within its repository in a standardized fashion.

CMIS exposes services for:

- Discovering object type definitions and other repository information (including the optional capabilities that are supported by a particular repository).
- Creating, reading, updating, relating, and deleting objects.
- Filing documents into one or more folders (if the repository supports the optional multi-filing capability).
- Navigating and traversing the hierarchy of folders in a repository.
- Creating and accessing versions of documents.
- Querying a repository to retrieve one or more objects matching user-specified search criteria, including full-text search.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

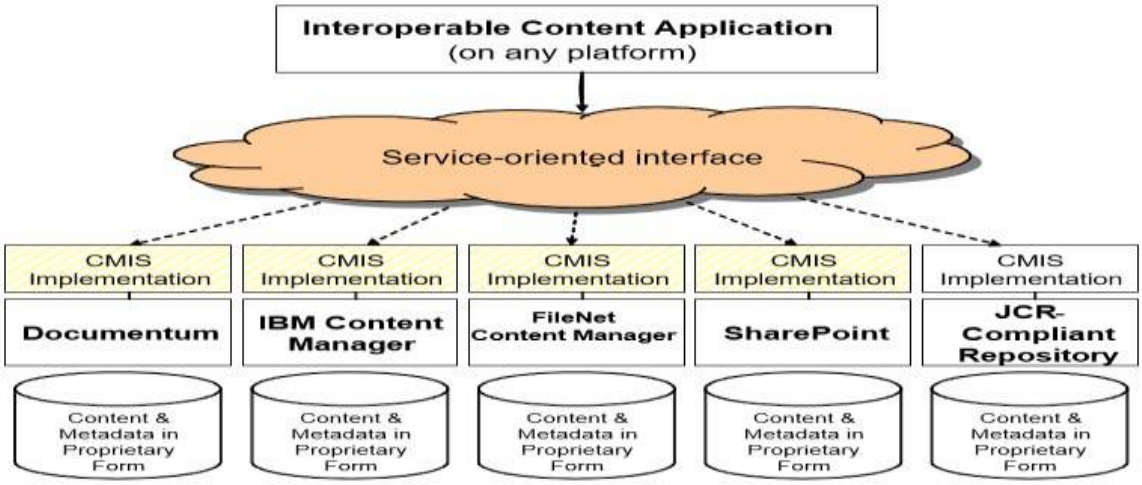

Fig.1: CMIS relies on a service-oriented interface to provide connections to disparate content repositories

II. RELATED WORK

- **Database synchronization between local and server by Rajni Jindal, Mtech CSE Department Bhai Gurdas Institute of Engineering & Technology, Sangrur,IJESRT, ISSN: 2277-9655, July, 2016**

The basic objective is to provide an algorithm to solve the problem , when all clients are relying on a single server. Due to planned server downtime or from server failures the database becomes unavailable; all of the remote workers will be disconnected from their data. Data is stored on their system (user system). When the user connected to the internet data automatically sink from their client system to the server in serial order and it also works on file handling. When the system is disconnected from the network all the files (images) uploaded by user, saved on the client machine folder when it is again connected to the server, automatically files (images) transferred from client to server.The synchronization refers to one of two distinct but related concepts: synchronization of processes, and synchronization of data. Process synchronization refers to the idea that multiple processes to a certain sequence of action. Process synchronization primitives are commonly used to implement data synchronization.

- **A Survey on Database synchronization algorithms for mobile device byMopati B. Kekgathetse, 10th April 2016, Vol.86. No.1, JATIT & LLS.**

Advances in technology have led to a new form of computing environment based on small,mobile devices. These devices come equipped with a lightweight database built unto them. Due to this property, processing of business information can now be achieved and as a result, more and more applications that depend on mobility of devices have emerged.

However, the mobile devices have limited processing power, rely on a finite battery source and have inadequate memory available. Moreover, it constantly access the network and this is a challenge because of limited bandwidth available to them. More often, mobile devices are occasionally disconnected from the network, this can be caused by a number of reasons; complete, unpredicted network cut, mobile data connectivity lapse or battery running low and contingency measures having to be executed as a result.Synchronization algorithms are procedures or methods adopted to facilitate data exchange between two or more entities. Synchronization process brings data to a consistent state across all entities involved. In the context of this paper, the entities in question refer to a mobile database client and a server-side database. The variations in these can be attributed to different design goals and specific problems that these

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

solutions were initially designed to address. Because mobile devices run on limited resources, it is inevitable that mobile data synchronization must account for this crucial requirement.

III. EXISTING SYSTEM

- POUCH TO COUCH SYNC

PouchDB is a JavaScript implementation of CouchDB. Its goal is to emulate the CouchDB API with near-perfect fidelity, while running in the browser. The synchronization takes place with the help of CouchDB replication protocol. PouchDB is used as local database and CouchDB is used as remote database.CouchDB sync has a unique design. Rather than relying on a master/slave architecture, CouchDB supports a multi-master architecture. A system is where any node can be written to or read from, and where and don't have to care which one is the "master" and which one is the "slave."CouchDB is an AP database, meaning that it's Partitioned, every node is Available, and it's only eventually Consistent. The data stored in pouchdb will be synchronized to couchdb when the user goes online.

- POUCH TO POUCH SYNC

Here, Pouchdb is used as local as well as remote database.

- Setting up sync:

To create local pouchDB, the following line is mentioned:
var localDB =new PouchDB('mylocaldb')
To create remote PouchDBs, the following line is mentioned:
var remoteDB =new PouchDB('http://localhost:5984/myremotedb')

- To perform unidirectional replication,simply do:

```
localDB.replicate.to(remoteDB).on('complete', function () {
}).on('error', function (err) {
});
```

- To perform bidirectional replication,simply do:

```
localDB.replicate.to(remoteDB);localDB.replicate.from(remoteDB);
```

IV. SYSTEM ARCHITECTURE

The System Architecture consists of Client app, pouchDB, sync, sync_CMIS server and other repositories given by thefig.2.whenever any new data is uploaded to the client app, if the client application is online a local copy of the data will be stored in the local database i.e., PouchDB. The replicator generates a replicationID before it replicates the data using CouchDB's replication algorithm, and the replication id is stored in the replication log. CMIS SYNC will compare replication ID of the local and remote database and if replication id is not matched with the ID in the remote database synchronisation will take place.

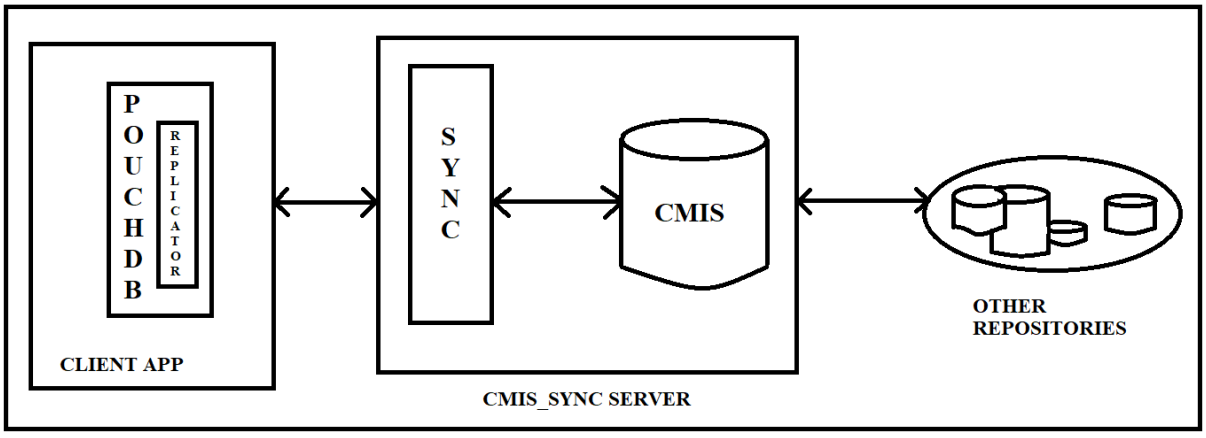

Fig. 2: System Architecture

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

V. IMPLEMENTATION

Implementation of system means the process of converting a new or revised system design into operational one.
The modules for developing an offline synchronization are

A. Module 1:Client App,
B. Module 2:PouchDB,
C. Module 3:Sync,
D. Module 4:Sync_CMIS Server

A) Client App

The first step is to develop a UI for a MemoDiary app. The UI consists of the following elements:

- An element for new list items.
- AngularJS template that will display each task in the MemoDiary app.

The next step is to render the UI. The MemoDiary app is used to store a list of items and its description. There is also a search option and delete option for each item in the list. This app will store the data locally once connected to its database. Over here pouchdb is used for storage purpose. The first page is the welcome page where sign in or sign up action takes place.Once the user enters the login details correctly, it will be redirected to the list master page where list of items shall be displayed.

B) Pouchdb

PouchDB is a JavaScript implementation ofCouchDB. Its goal is to emulate the CouchDB API with near-perfect fidelity, while running in the browser.

- Create connection for pouchDb server
- Connecting pouchDb server through app.
- Create local pouch database in app.
- Store all app data in that local pouchDb.

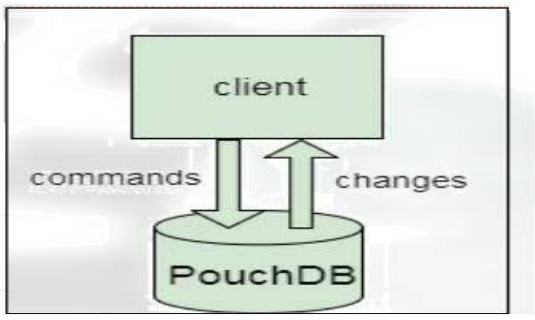

Fig. 3: Communication between client and pouchDB

The synchronization takes place with the help of CouchDB replication protocol.The replication algorithm work as follows Given a source and a target database, identify all current revisions (including deletions) in the source that do not exist in the target, and copy them (with contents, attachments and histories) to the target. Afterwards, all current revisions in the source exist at the target and have the same revision histories there.

i. Check source and target database existence by using HEAD request.
   HEAD /source HTTP/1.1
   Host: localhost:5984
ii. In case of non-existence of target,create target by PUT request
   PUT /target HTTP/1.1

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

Accept: application/json
Host: localhost:5984

iii. Generate a Replication ID. This value is used to track Replication History, resume and continue previously interrupted Replication process.

iv. Retrieve Replication Logs from Source and Target using GET Request and compare them.

GET /source/_local/b3e44b920ee2951cb2e123b63044427a HTTP/1.1
Accept: application/json
Host: localhost:5984

v. If any difference found in the replication logs fetch the changed document.

vi. When there are no more changes left to process and no more Documents left to replicate, the Replicator finishes the Replication process.

C) Sync

Most organizations have instituted backup processes, particularly for critical data. In some cases, they might use programs such as Systems Management Server or a third-party software product to perform this vital task.

In many cases, critical data files or file folders are shared among multiple users, such as a field sales force. Keeping current copies of all these files or folders can be a serious IT issue.

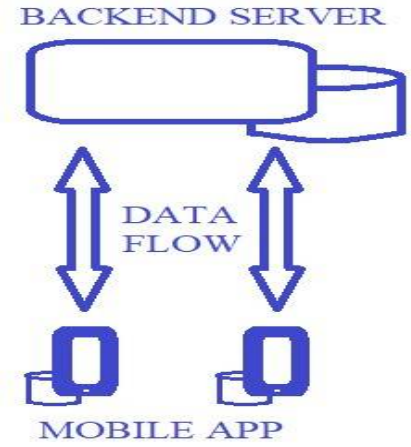

Fig. 4: Synchronization

Offline Synchronization makes it easier to ensure that critical files and folders on client computers and network servers are kept current and, in the process, helps ensure that vital data is backed up on a timely basis. Synchronization happens with the following steps:

- Specify the app local database in sync procedure.
- Specify the app server remote database in sync procedure.
- Sync with bidirectional replication. It will replicate with both databases.

D) Sync CMIS Server

Application connecting with CMIS_sync server. CMIS server get all app remote data it will replicate to local database.CMIS SYNC will compare replication ID of the local and remote database and if replication id is not matched with the ID in the remote database synchronisation will take place.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

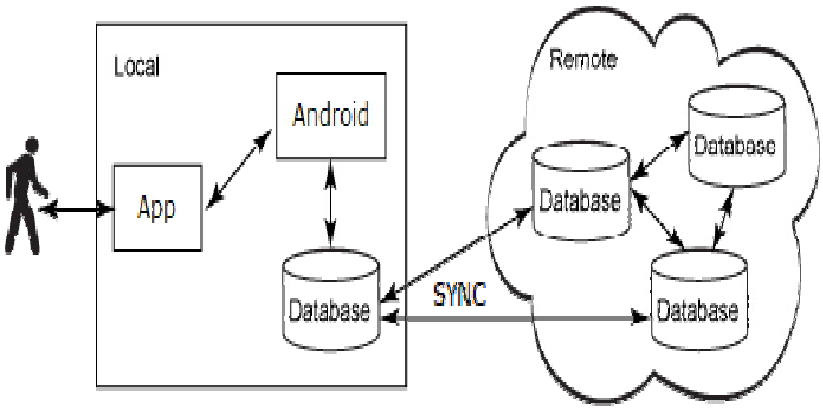

Fig. 5: Sync_CMIS Server

VI. CONCLUSION

A standard for content management systems data exchange would mean that different content management vendors could more easily exchange data and develop applications allowing the use of dispersed data repositories.The CMIS Workbench is distributed as a standalone java swing application. The CMIS specification is flexible enough to accommodate differences between implementations. A repository doesn't have to support ACL's and can still be CMIS-compliant. A CMIS repository typically uses named user accounts to control who can authenticate with the repository. As a developer, it will be necessary to meet all of the requirements of the application by staying strictly with pure CMIS API calls.Considering the fact that there are others standards that attempt to achieve the same goal as CMIS there is clearly and industry need for a standard. CMIS is implemented as a layer on top of an existing ECM solution and it exposes a relationship object for relating CMIS objects to one another.

REFERENCES

[1]. https://chemistry.apache.org/java/opencmis.html
[2]. http://docs.couchdb.org/en/2.0.0/replication/protocol.html
[3]. http://guide.couchdb.org/draft/replication.html
[4]. https://pouchdb.com/guides/replication.html
[5]. Database synchronization between local and server by Rajni Jindal, Mtech CSE Department Bhai Gurdas Institute of Engineering & Technology, Sangrur,IJESRT, ISSN: 2277-9655, July, 2016.
[6]. Survey on Database synchronization algorithms for mobile device byMopati B. Kekgathetse, 10th April 2016, Vol.86. No.1, JATIT & LLS.