Architecture Design Document for Conference Management System(CMS)

Ajinkya Bokade, CS17BTECH11001
Yash Khasbage, CS17BTECH11044
Jatin Chauhan, CS17BTECH11019
Happy Mahto, CS17BTECH11018
Himanshu Bishnoi, CS16BTECH11018

1. Overview

1.1: System Overview:

A conference management system(CMS) is a portal for hosting and managing conferences. It provides the functionality of paper submission, review for the paper, acceptance. CMS sends the notification on the updated status of their paper submission to the authors.

1.2: System Context

The system context is clearly defined in SRS. Academicians and people researchers in research labs are the significant sources of data. Users submit the paper for various conferences. The submissions are reviewed, accepted/rejected, and requested for Camera & Poster submissions. The respective conferences get the data of all the accepted papers. Data for rejected papers remain with the users.

1.3: Stakeholders

The main stakeholders are users registering for the conferences, authors submitting their papers, and CMS administrators who can create and update conference details.

1.4: Scope of this Document:

In this document, we describe the two possible architectures for CMS. With proper analysis under the perspective of implementation, scalability, querying, etc. we report the better architecture.

1.5: Definitions and Acronyms:

Acronyms:

CMS: Conference Management System
Cam-Pos Submission: Camera-ready and Poster Submission
DAL: Data Access Layer

Definitions:

Submit Paper: Submitting Abstract, Main-paper (pdf), Related Fields, Supplementary Material to a conference.
Accept-Reject Decision: It is a decision made by an Area Chair, based on the Reviews and Author response to Reviews.
Conference Admin: The user who manages the conference information and can finally download the conference data of Cam-Pos Submissions.
Website Admin: User who can accept the conference creation request and handles other managing issues regarding the portal.
Camera-ready and Poster submission: An accepted paper has to submit a camera-ready version of their paper along with a poster for presentation at the conference.

2. Architecture Design

2.1. Architecture 1: Data Repository Model (Shared-Data Style)

The architecture comprises a Data repository, where all the information about the user accounts, conferences, submitted paper rejected papers, reviewers are stored as well as various components that perform a usually disjoint set of tasks and inter-connect via the shared repository.

The following table describes all the components of the architecture:-

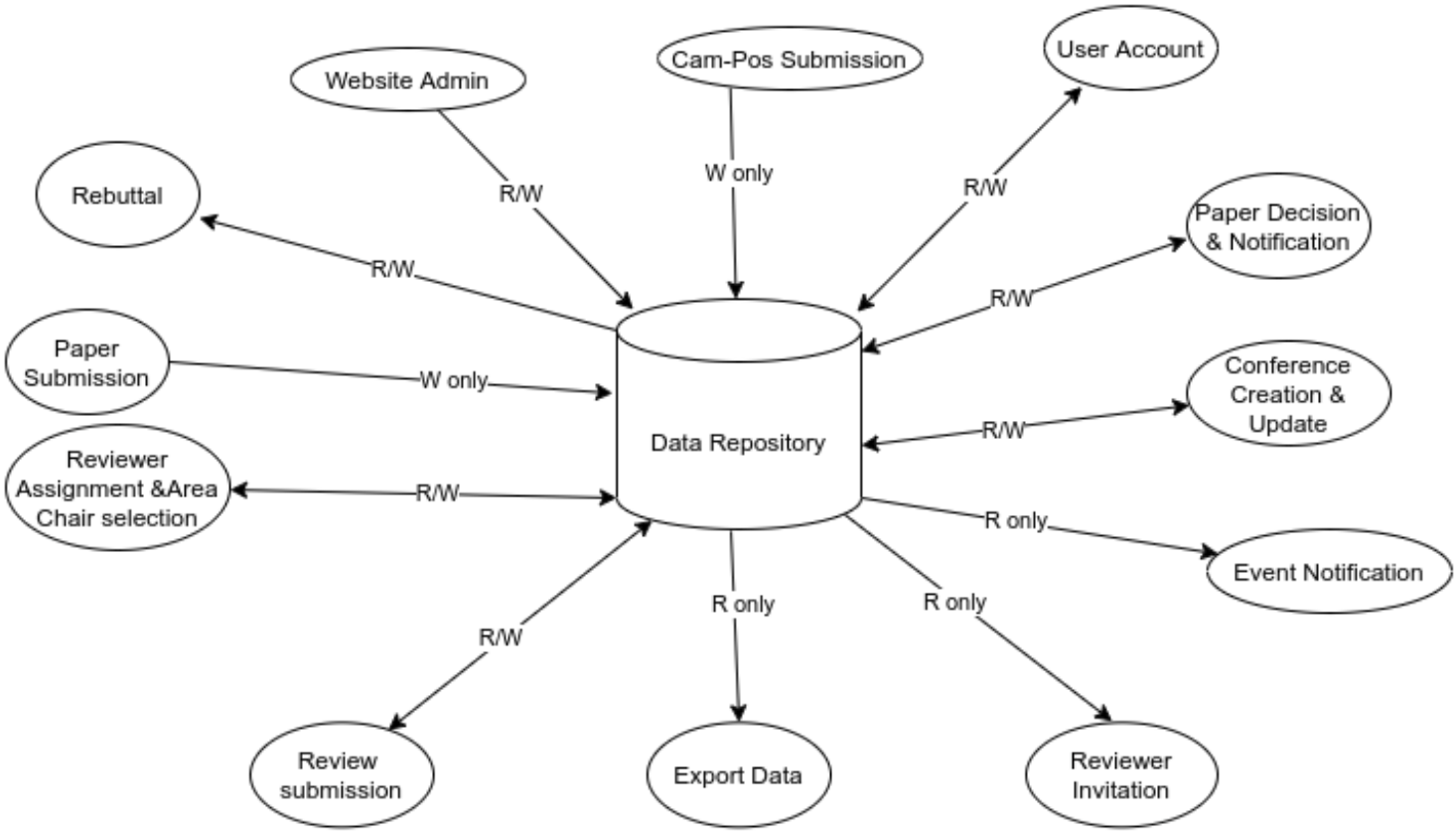
#	Component	Component Type	Description
1	Data Repository	Database	This module is the database containing information about the users, conferences, papers, etc.
2	Website admin	Processing (Database modification)	This module helps website admin to approve requested conferences and manage any portal-related activities.
3	User account	Processing (Database modification and access)	This module accesses the data repository to set/get user-related information which includes - login, signup, profile updates etc.

4	Paper decision & notification	Processing (Database modification and access)	This module accesses the data repository to get a paper submission and set the paper decision and notify the author.
5	Conference creation & update	Processing (Database modification and access)	This module accesses the data repository to request a new conference and get/set conference-related information.
6	Event notification	Processing (Database access)	This module accesses the data repository to get information about users and notify users about any event of a conference.
7	Reviewer invitation	Processing (Database access)	This module accesses the data repository to get information about users and send an invitation mail to become a reviewer.
8	Reviewer assignment & Area Chair selection	Processing (Database access and modification)	This module accesses the data repository to get information about users and conferences and sets reviewers and area chairs for the conference.
9	Paper submission	Processing (Database modification)	Adds the paper submission of the author in the data repository and handles related tasks such as withdrawal etc.
10	Rebuttal	Processing (Database access modification)	This module helps the author to obtain reviews from data repositories and upload a single rebuttal response.
11	Review submission	Processing (Database access modification)	This module helps the reviewer to get paper details and submit the review for the same to the data repository.
12	Export data	Processing (Database access)	This module helps to get all conference-related information as a compressed file that can be downloaded.
13	Cam-Pos submission	Processing (Database modification)	Adds the Cam-Pos submission of the author in the data repository.

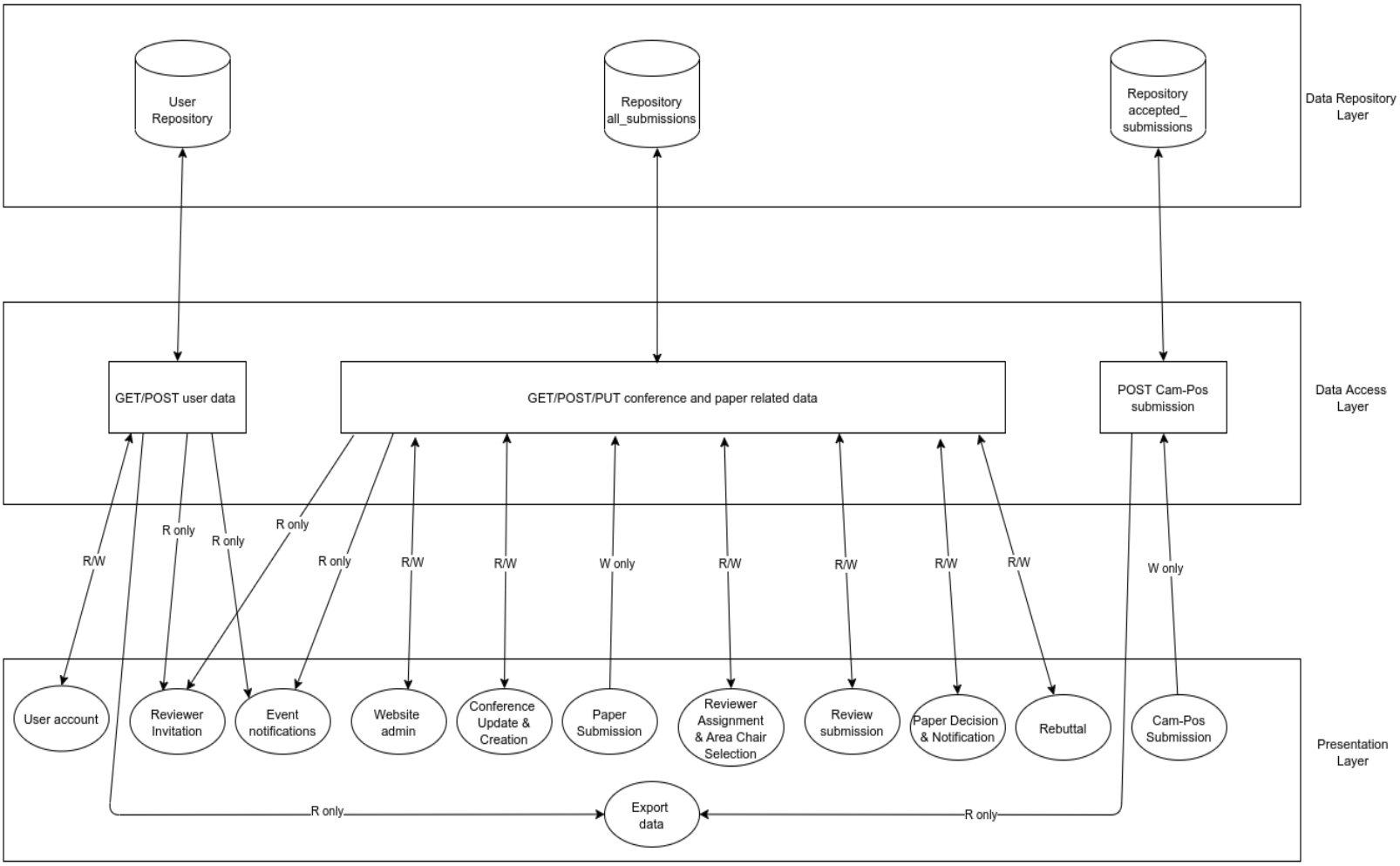
The following table describes all the connectors of the architecture:-

#	Connector	Connector Type	Description
1	R/W connectors	Database access/modification	Reads and Writes by modules to the data repository.
2	Read-only	Database access	Represent reading of relevant data from

	connector		the data repository.
3	Write-only connector	Database access, modification	Represent addition/modification (W) of the data repository.



2.2. Architecture 2: Multi-Repository Architecture (Combination of Shared-Data and Access Layer (3-Layer))



The following table describes all the components of the architecture:-

#	Component	Component Type	Description
1	User Repository	Database	Contains user-account information.
2	All Submissions Repository	Database	Contains conference-wise information for every paper submission
3	Accepted Submissions Repository	Database	Contains conference-wise information for every paper accepted along with the Cam-Pos submission.

4	User account	Processing (Database modification and access)	This module accesses the User repository to set/get user-related information.
5	Website admin	Processing (Database modification)	This module helps website admin to approve requested conferences and manage any portal-related activities, and accesses the All_submissions repository.
6	Conference creation & update	Processing (Database modification and access)	This module accesses the All_submissions repository to get/set conference-related information.
7	Event notification	Processing (Database access)	This module accesses the User and All_submissions repositories to get information about users and notify users about any event of a conference.
8	Reviewer invitation	Processing (Database access)	This module accesses the User and All_submissions repositories to get information about users and send an invitation mail to become a reviewer.
9	Reviewer assignment & Area Chair selection	Processing (Database access and modification)	This module accesses the All_submission repository to get information about users and conferences and sets reviewers and area chairs for the conference.
10	Paper submission	Processing (Database modification)	Adds the paper submission of the author in the All_submissions repository.
11	Rebuttal	Processing (Database access modification)	This module helps the author to obtain reviews from All_submissions repositories and upload a single rebuttal response.
12	Review submission	Processing (Database access modification)	This module helps the reviewer to get paper details and submit the review for the same to the All_submissions repository.
13	Export data	Processing (Database access)	This module helps to get all conference-related information as a compressed file that can be downloaded from Accepted_submissions repository.
14	Cam-Pos submission	Processing (Database modification)	Adds the Cam-Pos submission of the author in the Accepted_submissions repository.

The following table describes all the connectors of the architecture:-

#	Connector	Connector Type	Description
1	R/W connectors	Database access/modification	Reads and Writes by modules to the data repository.
2	Read-only connector	Database access	Represent reading of relevant data from the data repository.
3	Write-only connector	Database access, modification	Represent addition/modification (W) of the data repository.

This architecture differs from the earlier one due to two features:

1. 3 data repositories instead of one.
2. 3-layered architecture with a special data-access layer

We elaborate on the above-mentioned points in the following paragraphs.

The three repositories are:

1. **User repository:** It stores user account information for all users. It will contain information for every user registered on the portal, irrespective of the conferences he/she has submitted to.
2. **All Submissions Repository:** It contains all the paper submissions for each conference.
3. **Accepted Submissions Repository:** It contains all the papers accepted by the reviewers and area chairs.

Each component used by architecture can be scaled independently according to the conference needs and submission trends. For example, the Deep Learning community has seen a quadratic rise in the number of papers published. In this case, All_Submission has to increase accordingly. However, the number of good quality papers remains almost constant, and hence may not be scaled. Hence, Accepted_Submissions may not be scaled. This justifies the need for different database requirements.

The data access layer separates the Repository layer and the Presentation layer. The repository layer consists of the 3 repositories mentioned above. The presentation layer involves all the displays on the browser end.
The main advantage of having the data access layer is that, in case of changing the database, only the access layer has to be changed without affecting the rest of the system.

Each of the three databases contains only relevant entries which helps in querying in those databases faster as compared to the first architecture in which all information was stored in one database.

2.3. Comparing Architectures

Criteria	***Architecture 1***	***Architecture 2***
Security	Less Secure, since all the modules are directly accessing the data repository.	Higher security because of DAL, making it less susceptible to attacks.
Change in Data Repository Layer	Not Easy, since code for all modules need to be changed	Easy, since only DAL requires the code changes.
Addition in functionalities	Easy	Easy
Querying papers	Difficult and slower because of a unified data repository.	Easy and faster because of separate repositories.
Memory Efficient Scaling	No	Yes
Simplicity	Easy	Relatively Easier than Arch 1
Debugging	Easy	Relatively Easier (problems with different data repositories can be identified more easily).
Switching Databases	Difficult	Easy
R/W to data repository	Potential code duplication, thus inefficient	No code duplication, thus more efficient
E-mail Notification	Yes	Yes
Availability *	High	Relatively higher (Lesser load on the repositories, faster querying).
Data Currency **	Yes	Yes
Response Time *	Normal	Relatively Lower (faster database response)

* *Availability* and *Response Time* are measured subjectively and qualitatively here.
** *Data Currency* is measured in terms of search results such as - number of user's submissions to conferences etc.

Final Architecture of CMS

From the above table of architecture comparison, we see that architecture 2 is better in various aspects. It not only provides higher security, easy switching of databases and easier changes in the data repository layer but more importantly, provides much better performance in terms of querying research papers and provides memory-efficient scaling. The availability and response times of the second architecture are also expected to be better. The segregation on the storage side (repositories) can also help in reducing load as compared to the single repository case. We thus propose to use the ***second architecture*** for this project.