# System Design

## Design Goals

In the process of making our application, we have encountered several design goals that might fulfill our criteria and our desires. Of course that in trying to accomplish our standards, we had some drawbacks and we needed to decide some trade-offs. Our chosen goals are the following:

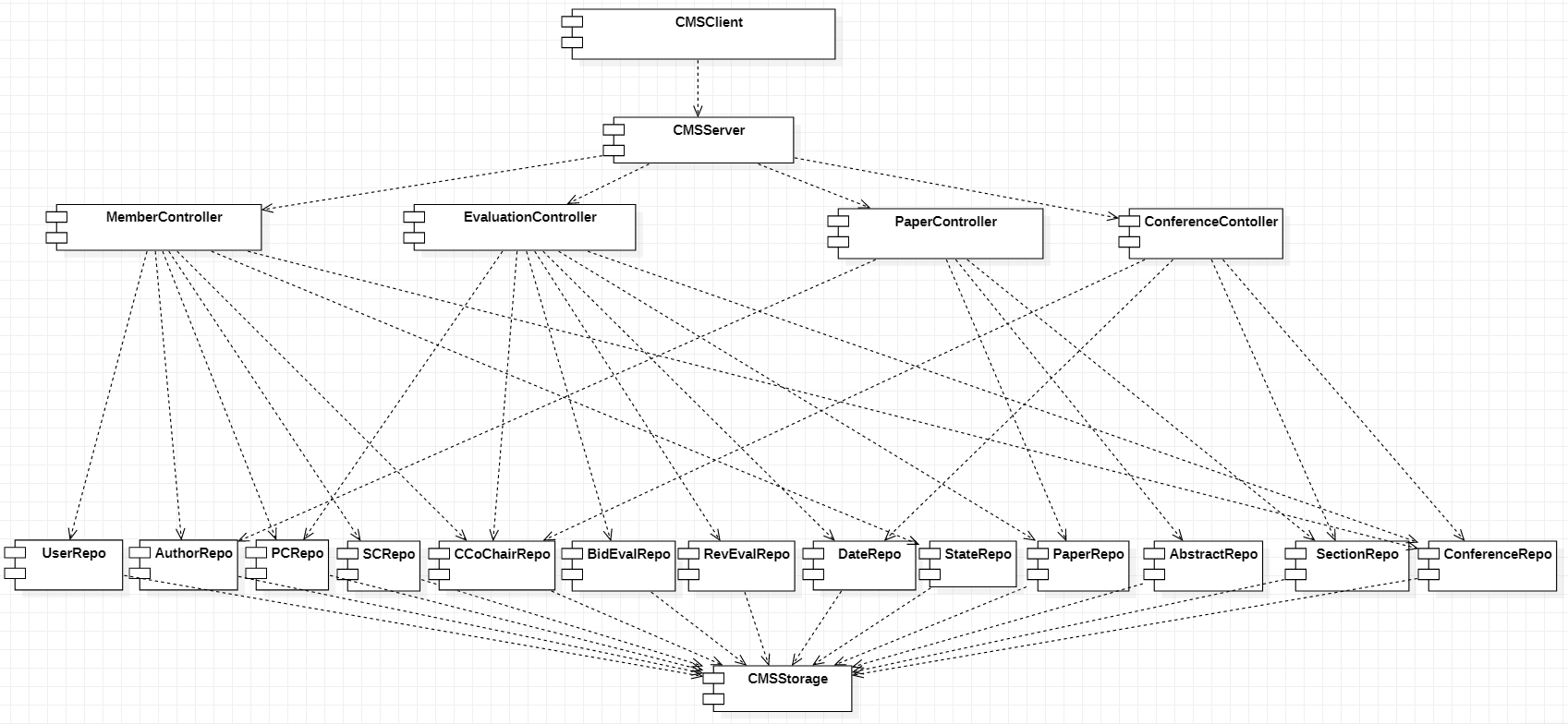
* Modifiability
* Reusability
* Understandability
* Traceability of requirements
* User-friendliness
* Ease of use
* Low cost
* Functionality

## Subsystem Decomposition

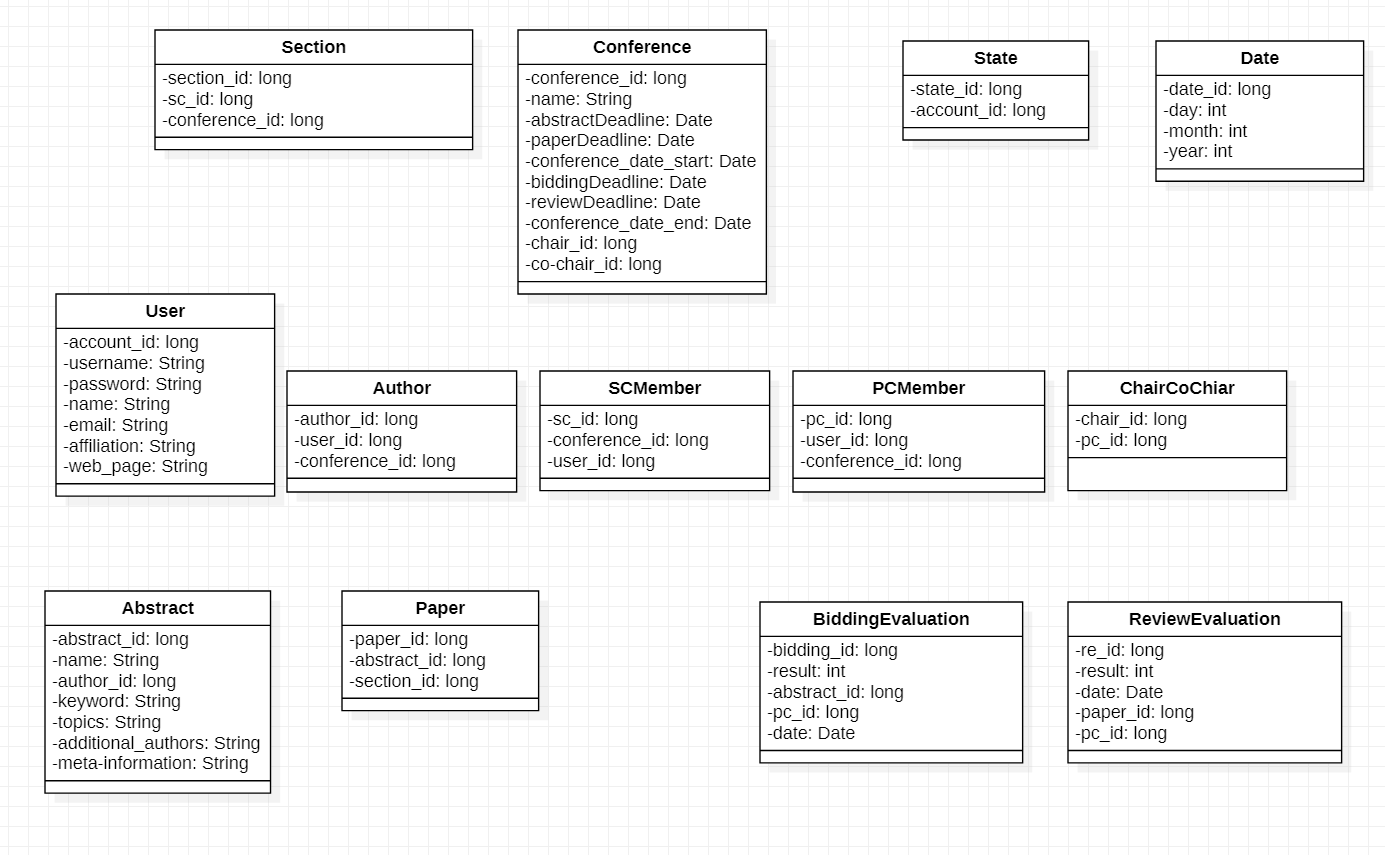
In this part we have considered the diagrams made in the analysis model, for the functional view, namely the use case diagrams.

In the following diagram, we will divide our system into subsystems identifying the services and repositories that we need to take into consideration.

Figure 1. CMS subsystem decomposition, conference organization part (UML component diagram, layers shown as UML packages).



For the architectural part we considered the Model-View-Controller + the Repository design patterns, as seen above.

Next we will detail the model part given the fact that we are going to use an ORM, Object-Relational Mapping. For each entity we will have some fields and we consider futile to add to the repositories the getters and setters methods since the framework that we will use will implement this automatically and will get the data from a database.

The only methods that we are going to implement are going to be located in controllers.

For each controller we are going to specify some functions and the preconditions as well as the postconditions.

MemberController

Login(username, password)

pre:

username – string not null

password – string not null

UserRepo.getAll().filter(p->p.getUsername() == username) != NULL

UserRepo.getAll().filter(p->p.getUsername() == username). getPassword() == password

State == NULL

post:

State = new State(username)

LogOut()

pre:

State != NULL

post:

State == NULL

CreateAccount(username, Password, Email)

pre:

username: string not null

password: string not null

email: string not null

State == NULL

UserRepo.getAll().filter(p->p.getUsername() == username) == NULL

UserRepo.getAll().filter(p->p.getEmail() == email) == NULL

post:

UserRepo.add(username,password,email)

UpdateInfo(Username,Email,WebSite,Affiliation)

pre:

username: string not null

email: string not null

website: string not null

affiliation: string not null

UserRepo.getAll().filter(p->p.getUsername() == username) != NULL

post:

UserRepo.getAll().filter(p->p.getUsername() == username).setEmail(Email)

UserRepo.getAll().filter(p->p.getUsername() == username).setWebsite(Website)

UserRepo.getAll().filter(p->p.getUsername() == username).setAffiliation(affiliation)

SendEmail(Username,Title,Message,ReceveingEmail)

pre:

username: string

title: string not null

message: string not null

receveingEmail: string not null

receivingEmail.contains(‘\*.[@]\*.[.]\*.) <- regex

State != NULL

post:

an email was sent to the receiving email with the Title and Message.

Search(searchString)

Pre:

searchString: string not null

post:

return UserRepo.getAll().filter(p->p.getUsername().contains(searchString)).getUsername()

PaperController

addAbstract(author\_id, name,Abstract, keyword, topics, additionalAuthors)

pre:

author\_id: long not null

abstract: string not null

name: string not null

keyword: string not null

topics: string not null

additionalAuthors: string not null

meta-information: string not null

AuthorRepo.get(author\_id) != null

post:

AuthorRepo.add(author\_id,name,abstract)

Search(searchString)

Pre:

searchString: string not null

post:

return AbstractRepo.getAll().filter(p->p.getName().contains(searchString)).getName()

EvaluationController

BidAbstract(username, abstractName, result, date)

Username: string not null

abstractName: string not null

result: integer not null

date: Date not null

PcRepo.getAll().filter(p->p.getUserId() == UserRepo.getAll().filter(u->u.getUsername() == username).getUserId()) != NULL

AbstractRepo.getAll().filter(p-> p.getName() == abstractName) != NULL

Date < ConferenceRepo.getBidDeadline()

Date > ConferenceRepo.getPaperDeadline()

Post:

BidEvalRepo.add(AbstractRepo.getAll().filter(p-> p.getName() == abstractName).getId(),

PcRepo.getAll().filter(p->p.getUserId() == UserRepo.getAll().filter(u->u.getUsername() == username).getUserId()).getId(), result, date)

ReviewPaper(user\_id, paper\_id, result, date)

User\_id: long not null

Paper\_id: long not null

result: integer not null

date: Date not null

PcRepo.getAll().filter(p->p.getUserId() == user\_id) != NULL

PaperRepo.getAll().filter(p->p.getId() == paper\_id) != NULL

[PcRepo.getAll().filter(p->p.getUserId() == user\_id).getId(), paper\_id] IN BidEvalRepo.getAll()

Date < ConferenceRepo.getReviewDeadline()

Date > ConferenceRepo. getBidDeadline ()

Post:

BidEvalRepo.getAll().filter(userId == userId AND paperId == PaperId).setResult(result)

BidEvalRepo.getAll().filter(userId == userId AND paperId == PaperId).setDate(date)

AssignReviewEvaluation(user\_id, pc\_id, paper\_id)

pre:

user\_id long not null

pc\_id long not null

paper\_id long not null

CCoChairRepo.getAll().filter(p->p.getUserId() == user\_id) != NULL

PCRepo.getAll().filter(p->p.getId() == pc\_id) != NULL

PaperRepo.getAll().filter(p->p.getId() == paper\_id) != NULL

RevEvalRepo.add(CCoChairRepo.getAll().filter(p->p.getUserId() == user\_id )   
.getId(),pc\_id,paper\_id,result:NULL,date:NULL)

ConferenceController

ChangePaperDeadline(user\_id, date)

Pre:

user\_id long not null

date Date not null

CChairRepo.getAll().filter(p->p.getUserId() == user\_id) != NULL

Date > ConfRepo.getAbstractDeadline()

Date < ConfRepo.getBidDeadline()

Post:

ConfRepo.setPaperDeadline(date)

CreateConference(user\_id,chair\_id, co\_chair\_id, ConfTitle, abstractDeadline, paperDeadline, bidDeadline,reviewDeadline, conf\_end\_date)

Pre:

User\_id long not null

Chair\_id long not null

Co\_chair\_id long not null

ConfTItle string not null

abstractDeadline Date not null

paperDeadline Date not null

bidDeadline Date not null

ReviewDeadline Date not null

Conf\_end\_date Date not null

UserRepo.getAll().filter(p->p.getId() = chair\_id) != NULL

UserRepo.getAll().filter(p->p.getId() = co\_chair\_id) != NULL

SCRepo.getAll().filter(p->p.getUserId == user\_id) != NULL

ConfRepo.getAll().filter(p->p.getName() == confTitle) == NULL

nowDate() < abstractDeadline < paperDeadline < bidDeadline < reviewDeadline < conf\_end\_date

post:

new Conference(chair\_id, co\_chair\_id, ConfTitle, startDate: now(), conf\_end\_date, abstractDeadline,paperDeadline,bidDeadline,reviewDeadline)

pc\_id1= PCRepo.add(chair\_id,conference\_id).getId()

ChairRepo.add(pc\_id1,conference\_id)

pc\_id2 = PCRepo.add(co\_chair\_id,conference\_id).getId()

ChairRepo.add(pc\_id2,conference\_id)