**Conceptual Modeling & Design. Project description.**

**“WeStudy” academic social network.**

Group composition

Joachim Alvarez-Rodriguez ([Joachim.Julio.Alvarez-Rodriguez@vub.be](mailto:Joachim.Julio.Alvarez-Rodriguez@vub.be))

Aldar Saranov ([Aldar.Saranov@ulb.ac.be](mailto:Aldar.Saranov@ulb.ac.be))

Concept

Two types of users are supposed to interact with the system - student, teacher. There is also a special kind of user, school. The user type and privileges are determined automatically at the login stage.

The school user represents a complete school (like the VUB). These kind of users are made by the creator of the application using legal means (Identification of the school and such). Once a school user is active he can create accounts for teachers and students). Teachers have assigned courses. Students can follow courses. Students can discuss topics with each other or ask the teacher for help.

Common actions of students and teachers:

* Login.
* Post at the course page, he/she is assigned to.
* Comment on posts of his/her course.
* Edit password and avatar.

Student’s actions only:

* Post hidden posts. (only visible to the creator and the teacher)
* Be able to request help from the teacher on one of the public posts.

Teacher’s actions only:

* Create course pages.
* View hidden posts.
* Remove any post from his/her course.
* Pin posts.
* Teachers have a star tag over their posts and comments.
* View posts where he/she is requested.

School supervisor’s actions:

* Are predefined by the application developers.
* Can create student and teacher accounts.

Implementation details

The application will be developed as an Android application. The database will be stored locally for simulating the data web-access.

CTT developing

Concurrent Task Tree is required to specify the relations and sequence between the tasks.

[Final CTT will be here]

Enabled task set

Enabled task sets were derived from the CTT and allow to observe the planned presentation layers of the application.

[ETS final screenshot]

Here following ETS are represented:

1. {Enter username, Enter password, Submit}
2. {Select join new courses, Select a course, Open profile, Logout}
3. {Type new password, Retype password, Upload avatar, Submit, Close profile, Logout}
4. {Type name filter, Select course, Join course, Logout}
5. {Toggle help-request, Select add comment, Stop browsing comments, Stop browsing posts, Logout}
6. {Enter content, Submit, Stop browsing comments, Stop browsing posts, Logout}
7. {Enter title, Enter content, Toggle public hidden, Toggle help request, Submit, Stop browsing posts, Logout}
8. {Select leave course, Select post, Select new post, Stop browsing posts, Logout}
9. {Yes or No, Stop browsing posts, Logout}
10. {Select join new course, Open course list page, Open request page, Open profile, Logout}
11. {Type name filter, Select course, Join course, Stop managing courses, Logout}
12. {Select a course among the assigned ones, Stop managing courses, Logout}
13. {Select leave course, Add post, Remove any post, Add comment, Remove any comment, Pin post, Manage courses, Stop managing courses, Logout}
14. {Add comment, Remove any comment, Stop managing courses, Logout}
15. {Type new password, Retype password, Upload avatar, Submit, Close profile, Logout}
16. {Select new teacher, Select new student, Show courses, Logout}
17. {Enter name, Submit, Stop managing courses, Logout}
18. {Yes or no, Stop managing courses, Logout}
19. {Select delete course, Select new course, Stop managing courses, Logout}
20. {Type name filter, Select course, Stop managing courses, Logout}
21. {Enter username, Enter e-mail, Submit, Logout}
22. {Enter username, Enter e-mail, Submit, Logout}

Object-role modeling

The ORM was derived from the application requirements as follows:

[Final ORM here]

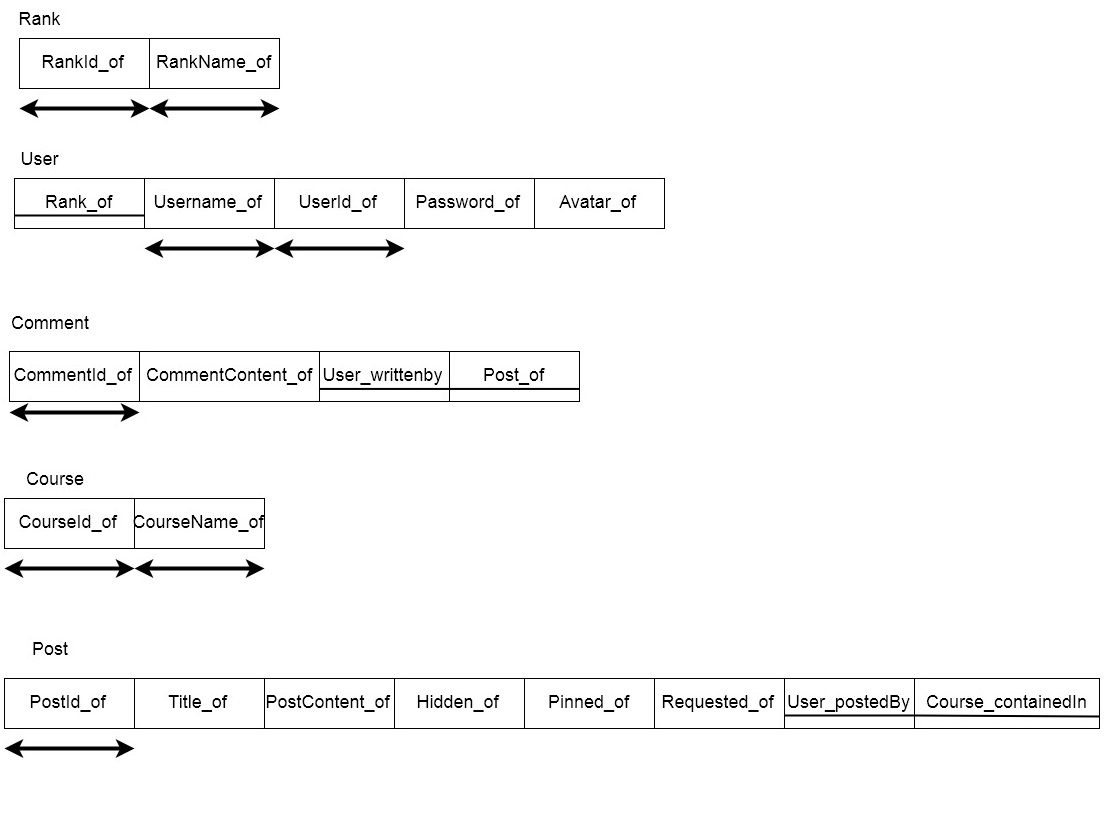
We need to pass 7 steps of RM mapping into a database model.

Verifying RMR completeness

1. All LOT are RMR.
2. “Rank” is RMR since it has a unique and total simple reference by “Rank name”.
3. “User” is RMR since it has a unique and total simple reference by “User name”.
4. “Comment” is RMR since it has a unique and total simple reference by “CommentId”.
5. “Course” is RMR since it has a unique and total simple reference by “CourseId”.
6. “Post” is RMR since it has a unique and total simple reference by “PostId”.

Grouping around non-subtype entities

Following tables are obtained:

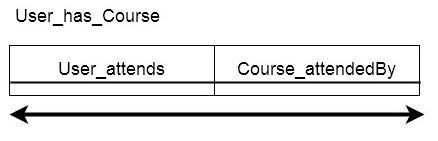


Group around subtypes

We do not have any subtype in the ORM.

Map many-to-many

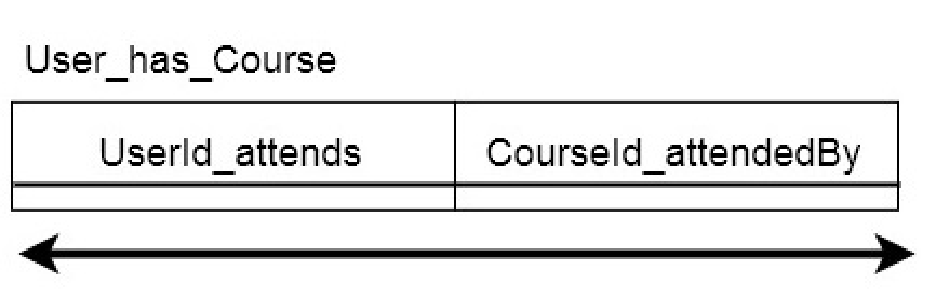
We only have one such relation



Make lexical

One primary key was selected from each of the table candidate keys.

In User\_has\_Course the corresponding keys were substituted.



Result

