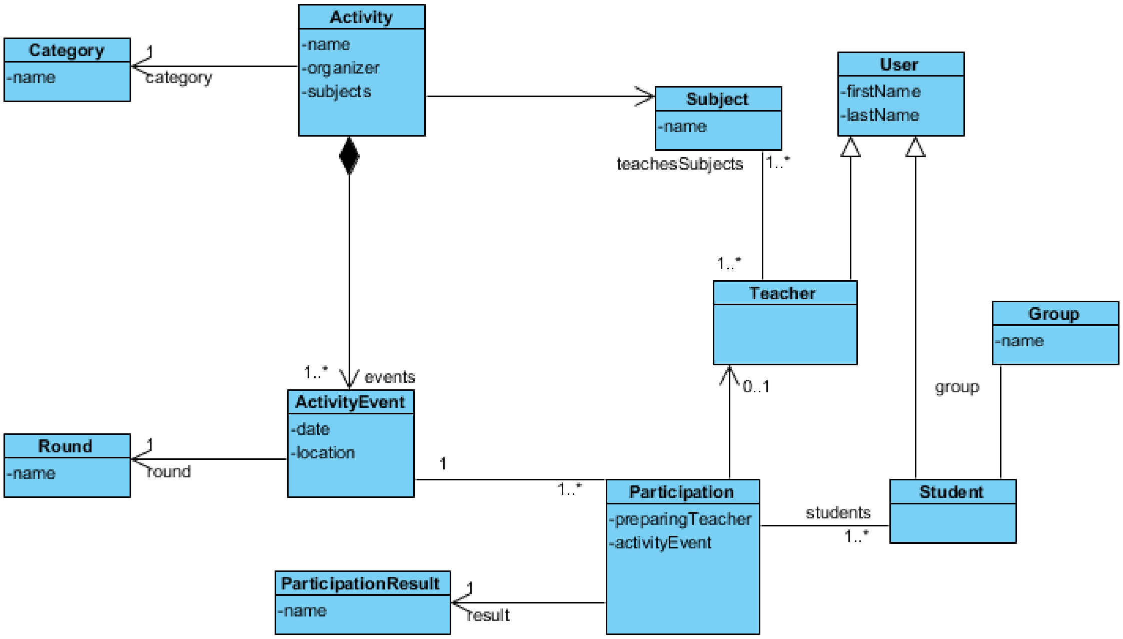
Istvan Csaszar
30431

Student performance tracker
Analysis and Design Document

I. Elaboration – Iteration 1.1

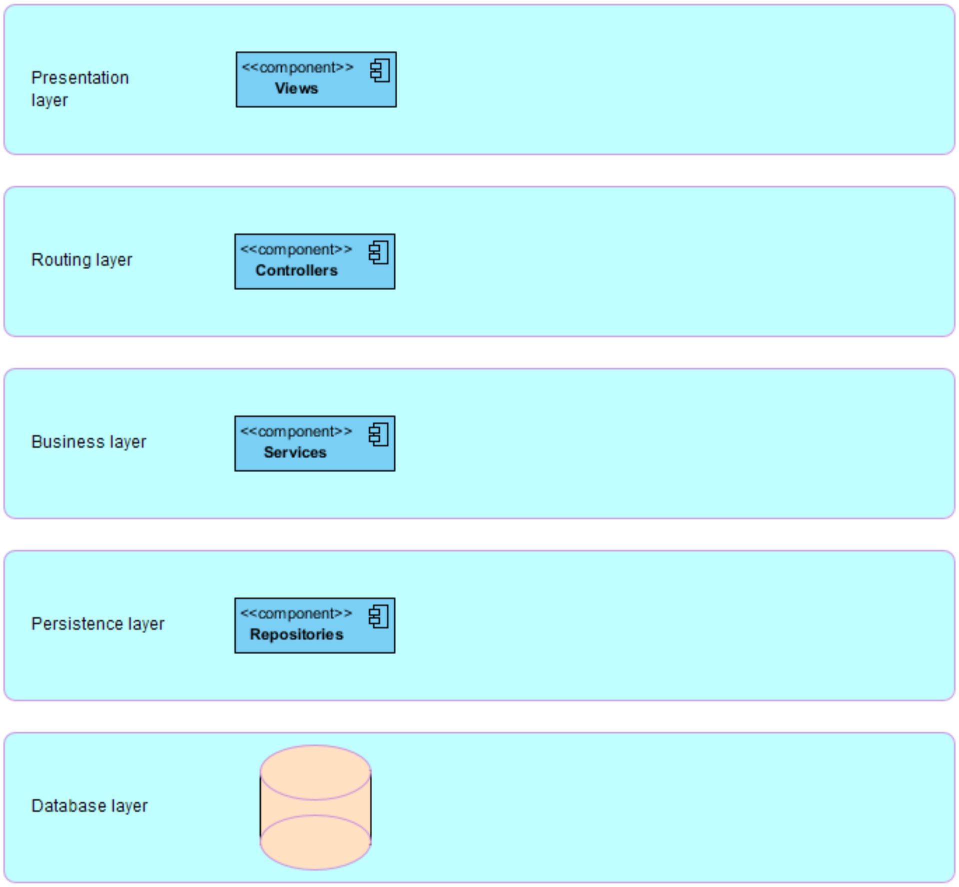
1. Domain Model

The system records student participations in contests and activities. Each contest has a name, an organizer and belongs to a given category and can be about some subjects. A contest has one or more rounds, each round has location, a date when it took place. Students participate (individually or in teams) in one or more rounds of a contest, where they obtain results, students may be prepared for the round by a teacher.
Participations earn students and teachers points, depending on the category, the round and the placement (result).

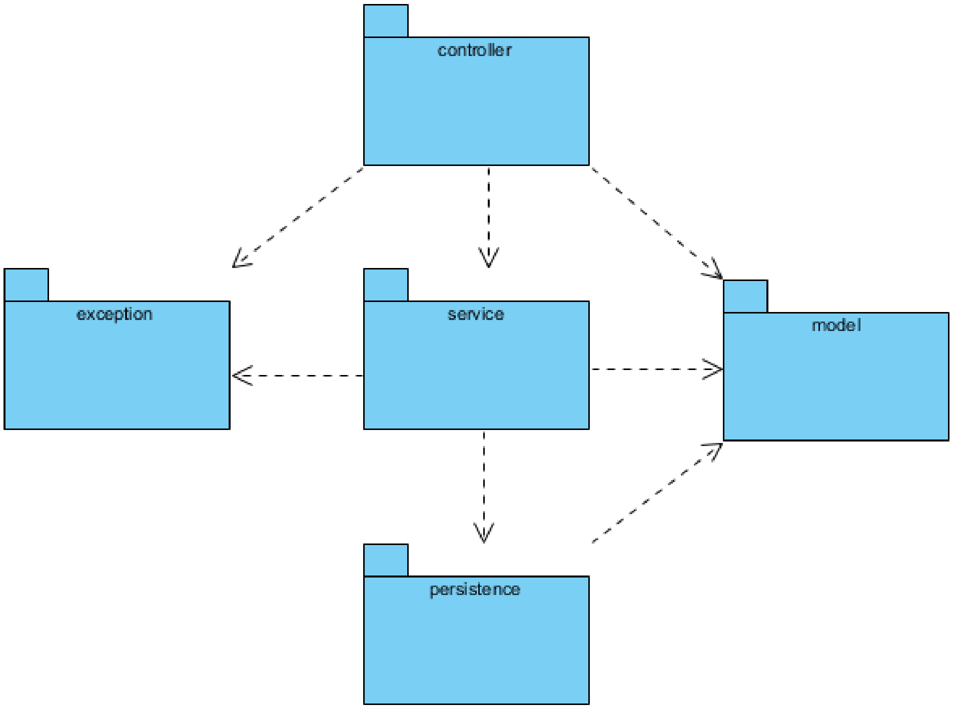


2. Architectural Design

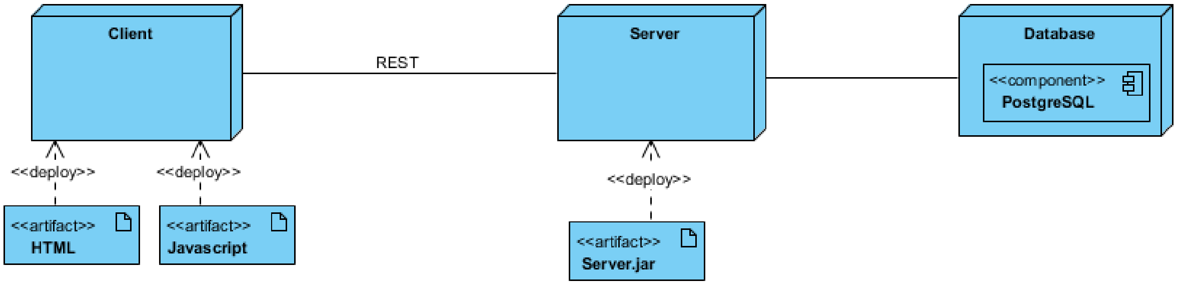
2.1 Conceptual Architecture



2.2 Package Design



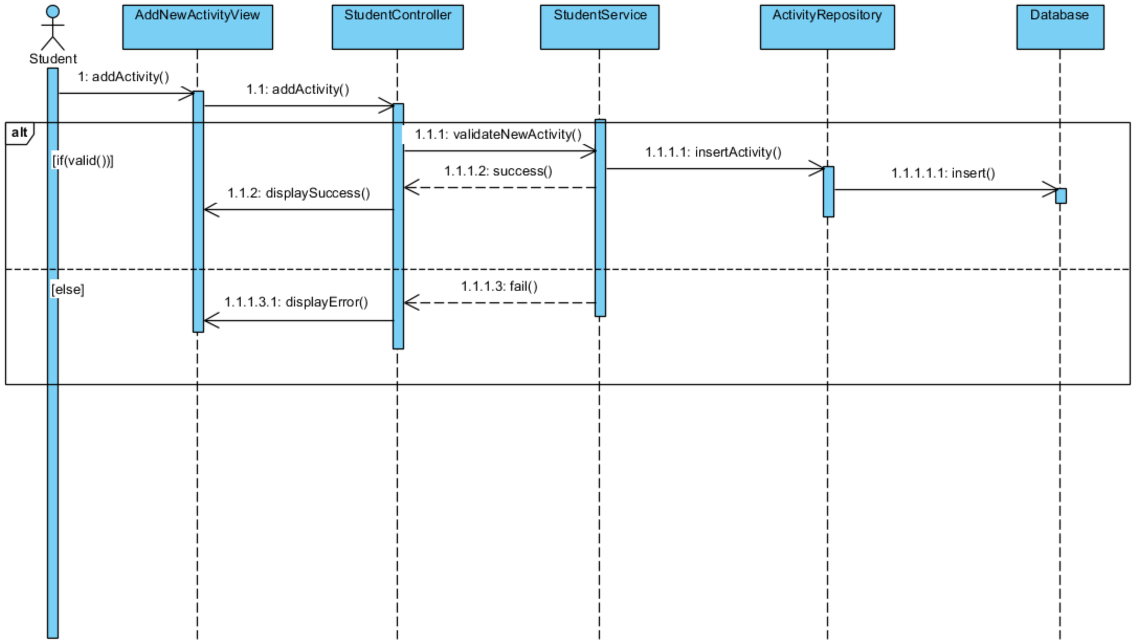
2.3 Component and Deployment Diagrams

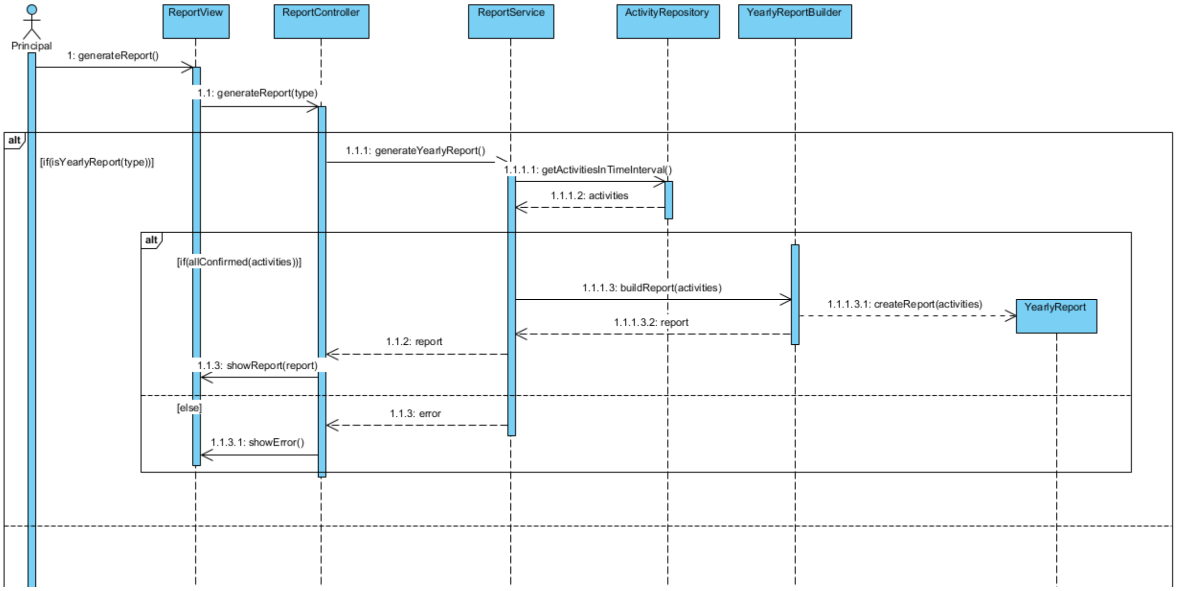


II. Elaboration – Iteration 1.2

1. Design Model

1.1 Dynamic Behavior

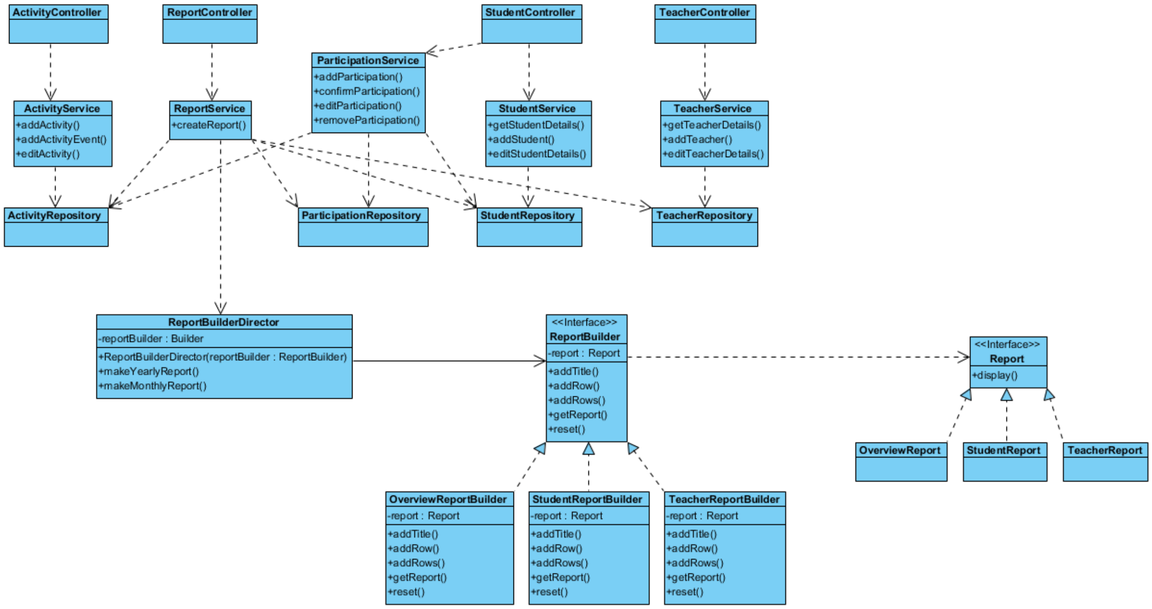



Principal
ReportView
ReportController
ReportService
ActivityRepository
YearlyReportBuilder
1: generateReport()
1.1: generateReport(type)
alt
[if(isYearlyReport(type))]
1.1.1: generateYearlyReport()
1.1.1.1: getActivitiesInTimeInterval()
1.1.1.2: activities
alt
[if(allConfirmed(activities))]
1.1.1.3: buildReport(activities)
1.1.1.3.1: createReport(activities)
YearlyReport
1.1.1.3.2: report
1.1.2: report
1.1.3: showReport(report)
[else]
1.1.3: error
1.1.3.1: showError()

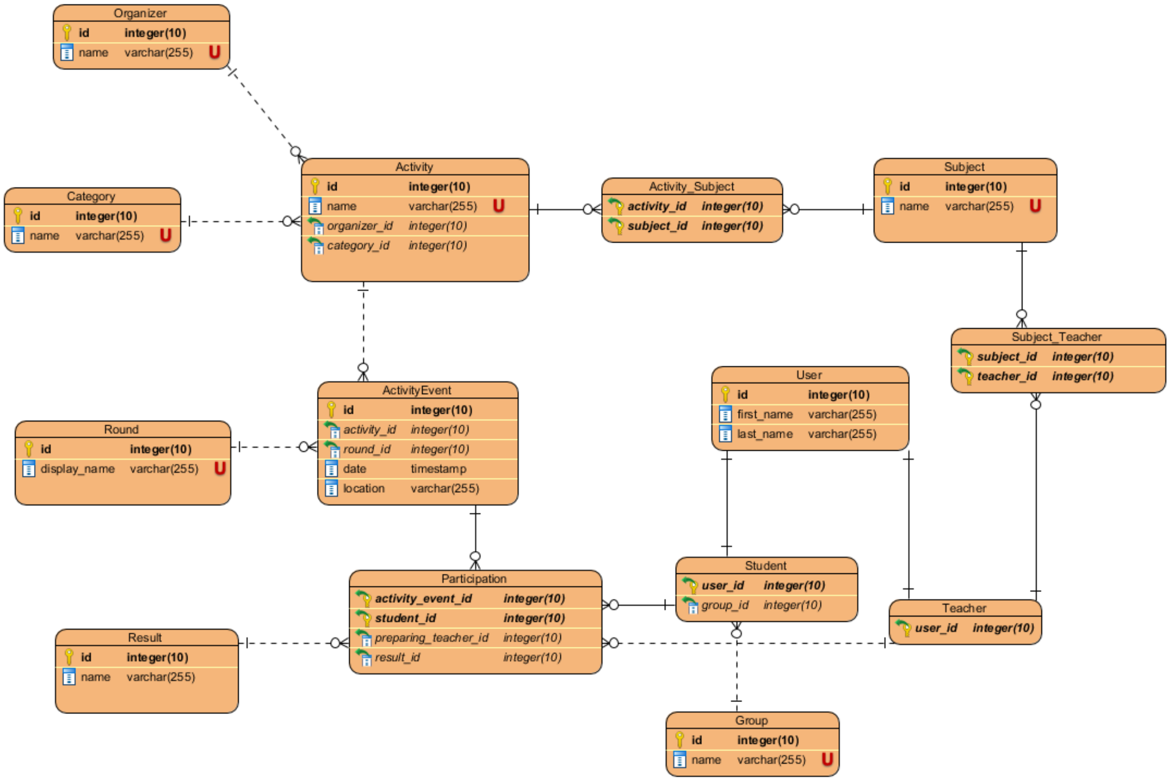
1.2 Class Design

Patterns:

- Value objects for Category, Round, Subject, Organizer, Group, ParticipationResult
- Builder pattern for the reports - reports can differ by the time interval they cover and the fields (columns) they include, and can take many parameters to build, which is a perfect fit for the builder pattern with a director.



2. Data Model



3. Test Strategy

3.1 Unit tests

Testing for the service layer, with one test per method, like:

- Adding a new entity (Student, Teacher, Activity, Participation) is successful
- Awarded points are correctly calculated in the reports
- Data for existing entities is returned correctly
- Handling common invalid input

3.2 Integration tests

Tests for the main use cases, like:

- The personal report of a student is correctly generated
- A teacher can confirm a student activity
- The application responsible can add/remove users.

III. Elaboration – Iteration 2

1. Architectural Design Refinement

[Refine the architectural design: conceptual architecture, package design (consider package design principles), component and deployment diagrams. Motivate the changes that have been made.]

2. Design Model Refinement

[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]

IV. Construction and Transition

1. System Testing

[Describe how you applied integration testing and present the associated test case scenarios.]

2. Future improvements

[Present future improvements for the system]