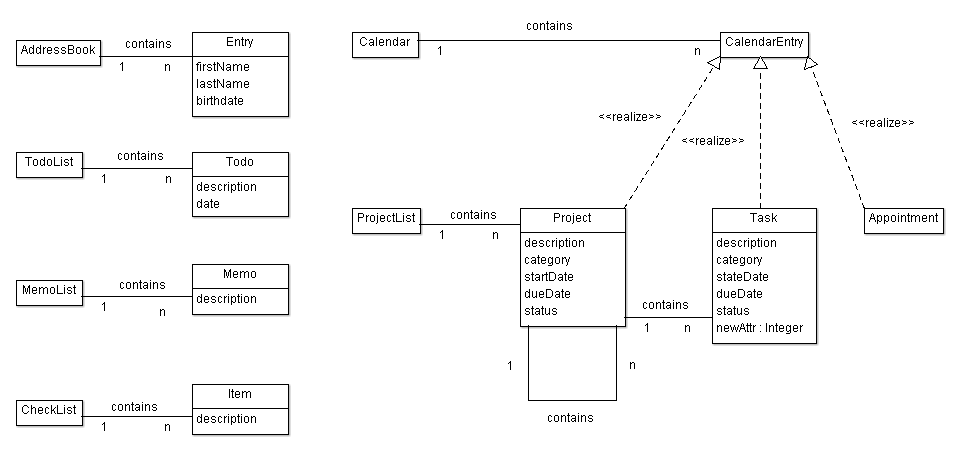
BORG CALENDAR – M3

Student Name and Number

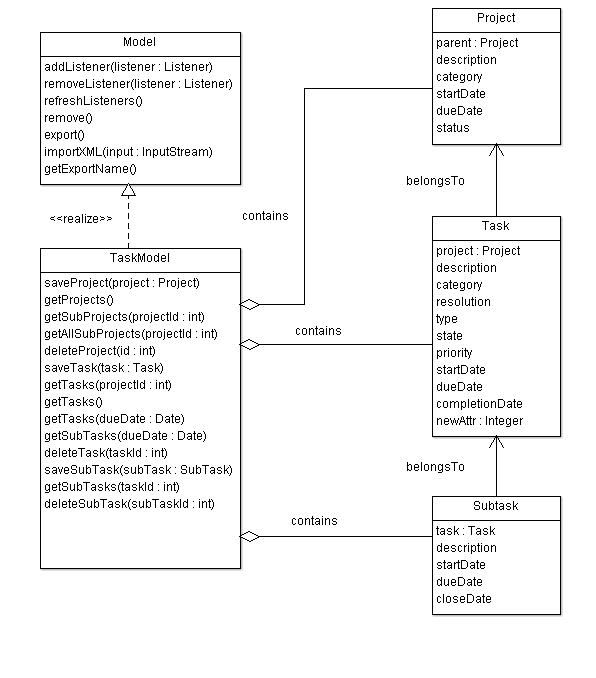
Hamid Shahrestani (9729747)  
Anjaneyulu Bodepudi (5973775)   
Manouchehr Azizi (5232287)   
Hermann Sonfack (5986052)   
Viet Hung Nguyen (9816240)

# Class Diagram of Actual System

**Conceptual Diagram**



**Actual Class Diagram**

****

We show here only five classes relate to project management features:

* Model is an abstract class for all model classes. Each model class is a singleton class which provides functionalities to access/manipulate entities for a feature. For example: task model provides functionalities to access/manipulate entities for project management feature; includes: Porject, Task, SubTask.
* TaskModel is a realization class of Model and is used to access/manipulate projects, tasks and subtasks.
* Project is a data class to hold data of a project. It provides no functionality relate to a project, for example, get sub projects, get tasks, etc.
* Task is a data class to hold data of a task.
* SubTask is a data class to hold data of sub task.

**Compare Conceptual Diagram & Actual Class Diagram**

* TaskModel in some degrees acts as ProjectList in conceptual diagram. But they are not totally the same: TaskModel not only contains projects, but also tasks and sub tasks.
* In both conceptual diagram & actual class diagram, a project may contain other projects. But in conceptual diagram, a task cannot contain other tasks. Actual class diagram has sub task class and a task may contain sub tasks.

**TaskModel class**

public class TaskModel extends Model implements … {

public void saveProject(Project p) throws Exception {}

public Collection<Project> getProjects() throws Exception {}

public Collection<Project> getSubProjects(int projectid) throws Exception {}

public Collection<Project> getAllSubProjects(int projectid){}

public void deleteProject(int id) throws Exception {}

public void savetask(Task task) throws Exception {}

public Collection<Task> getTasks() throws Exception {}

public Collection<Task> getTasks(int projectid) throws Exception {}

public Collection<Task> get\_tasks(Date d) {}

public void saveSubTask(Subtask s) throws Exception {}

public Collection<Subtask> getSubTasks() throws Exception {}

public void deleteSubTask(int id) throws Exception {}

}

**Project class**

public class Project extends KeyedEntity<Project> implements CalendarEntity {

private Date StartDate;

private Date DueDate;

private String Description;

private String Category;

private String Status;

private Integer Parent;

public void setStartDate(Date value) {

StartDate = value;

}

public Date getStartDate() {

return StartDate;

}

// … other getters/setters

}

# Code Smell and Possible Refactoring

Look at TaskModel class we can see that it’s quite long: there is Large Class code smell here.

Refactoring to fix this code smell in TaskModel:

* daysBetween(Date start, Date dd), daysLeft(Date dd) is not the responsibilities of TaskModel and can be moved out of it.
* …