****

**Object Oriented Software Engineering Project**

**Design Report**

**CS 319 Project: RISK: LOTR**

**Group 1J**

* **Miraç Vuslat Başaran**
* **Hazal Buruk**
* **Elena Çina**
* **Doğa Zeynep Germen**

**Intructor: Bora Güngören**

**Table of Contents**

1. **Introduction**
   1. **Purpose of The System**
   2. **Design Goals**
2. **Software Architecture**

**2.1 Subsystem Decomposition**

**2.2 Hardware/Software Mapping**

**2.3 Persistent Data Management**

**2.4 Access Control and Security**

**2.5 Boundary Conditions**

1. **Subsystem Services**

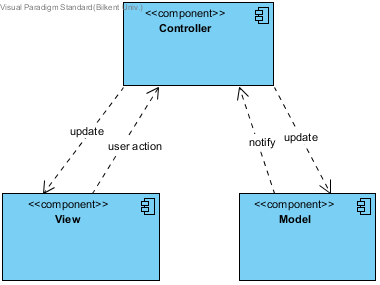
**2. Software Architecture**

This section includes a subsystem decomposition of our software project to make it understandable and easy to implement. Also, which hardware/software tools are needed and the database management is explained here. Finally, access control and security issues as well as boundary conditions are examined in their respective subsections.

**2.1 Subsystem Decomposition**

RISK: LOTR, being a graphical strategy game, has a couple of interfaces to display the status of the game and to take user input as well as a complex logic that runs behind the scenes. To capitalize on this interface-logic separation, the game is designed on MVC (Model-View-Controller) pattern. This will help minimize coupling between main subsystems and maximize cohesion within those subsystems.

The “View” subsystem deals with handling user interfaces and getting user input. “Model” subsystem handles the data, rules and the logic of the application. Finally, “Controller” subsystem gets input from the other subsystems and updates them.



Below, the MVC architecture is expanded to show the subsystems of these three components and their interactions.

