***Homework 2 – Softwaretechnik***

**Task 1**: Process Modeling with Activity Diagrams

Steps

1. *Lecturer Logs into System*
   1. Action: “Lecturer enters credentials”
   2. Control Flow: Transition to “Open course page”
2. *Open Course Page*
   1. Action: “Lecturer opens course page”
   2. Control Flow: Transition to “Enter lecture times and exercise times”
3. *Enter Lecture and Exercise Times*
   1. Actions: “Lecturer enters lecture/exercise times”
   2. Decision: System performs quick validation check
      1. Path: **If valid**, continue
      2. Path: **if invalid** (conflict detected), alternative flow (adjustment)
4. *Confirm Changes*
   1. Action: “Lecturer confirms changes”
5. *System Checks and Reports Conflicts:*
   1. Action: “System generates conflict report”
   2. Decision: Conflict detected?
      1. Path: If **no conflicts**, continue to submission
      2. Path: If **conflict exist**, alternative flow (adjustment)
6. *Adjust Times (Alternative Flow)*
   1. Action: “Lecturer adjusts times and resubmits”
   2. Control Flow: Returns to “System checks and reports conflicts”
7. *Submit Course*
   1. Action: “Lecturer submits course details”
   2. Control Flow: System updates courses availability
8. *Ende*
   1. Final Node. Course available for students to enroll

Start

Conflicts? Times?

Exercise Group Distribution

Open Course Distribution

Open Course Page

Open Group- Page

Times

YES

Adjust Time

YES

Times

Course Available

Adjust Time (YES)  
Yes Adjust Time   
- No

NO

Adjust Times (---) (NO)

Adjust Course Page

SUBMITE

Course Available

**Task 2:** State-Machine Modeling

1. *Initial State: The student is not enrolled in the course*
2. *Transition to Enrolled*
   1. When the student clicks “enroll” they enter enrolled state but aren’t yet assigned
3. *Assignment Process*
   1. Assigned: If the assignment has no issue, the student is assigned directly
   2. In Conflict: If there are conflicts, the student moves to an in-conflict state that requires coordination between the lecturer and student
4. *Resolution of Conflict*
   1. Conflict Unresolved: If unresolved, the student is terminated from the course
   2. Conflict Resolved: The students move to the assigned state
5. *Termination*
   1. Drop Out: If the student drops out, they return to the not enrolled state
   2. Course Distribution Closed: If the lecturer closes the course distribution, all students are terminated
6. *End Conditions*
   1. A student can be in the not assigned, assigned or in conflict state, which are all considered enrolled

Not Enrolled  
(Student not   
Assigned)

Assigned  
(Conflict Resolved),  
(Not Assigned)

Not Enrolled  
 (Not Assigned)

Assigned in Conflict  
 (Terminated)

**Task 3:** System Decomposition and Architecture

1. A Layered Architecture Pattern is most suitable for this Exercise Group Distribution (EDG), since it organizes the application into distinct, horizontal layers.  
   Each layer is responsible for specific aspects of the system’s functionality, has a defined role and only interacts with the layers immediately above or below it.   
   This architecture is common in complex systems, because it promotes modularity, maintainability and separation of concerns.  
     
   In this EDG system, we can break down the functionality into three main layers:
2. Presentation Layer: Manages user interactions (Frontend)
3. Application Layer: Manages core business logic and application processing
4. Data Source Layer: Manages external data sources and databases (KLIPS)

Each layer has well-defined responsibilities, which helps keep the system organized and makes it easier to maintain, test and update over time.

For this task, the Layered Architecture Pattern structures a diagram well, which includes:

1. Frontend connected to Communication and Course Management
2. Communication and Course Management as accessible components for the client
3. Course Management connected to KLIPS for course data
4. Course Distribution using Course Management for course and user data, and Communication for notifications
5. Interfaces and Portes where applicable
6. .

**Frontend**

Display

Component

**Exercise Group Distribution**

Course Management

Course Component

User Client

Course Management

Frontend

Directly Available

Display Component

Course Management

**KLIPS**

Course Data

Database

Frontend

Directly Available

Display Component

Course Management

Course Data

Courses

Course Distribution

Course Data

Course Component

Course Component

Course Component

Course Data

Course Migration

Requires a provided interface

User Distribution

Course Distribution