Course project

Design and implement the game Mancala (AI and network support optional). Use the objects first design method. This means, start with scenarios, derive user stories, create object diagrams, sort into usecases, derive class diagrams, use object game or note-pad test to define methods, implement methods. Iterate, refactor, use/apply design patterns, refine documents form the previous steps. Implement at least a hot seat version of Mancala (using two players). Implement a history function for keeping scores of specific matches.
Deadline: November 30
Delivery: git-repository + project report as pdf
Repository should include:

- At least 30 user stories
- At least 40 object diagrams
- Class diagram (as Fujaba ctr or pdf)
- At least 15 test cases in story boards (or a compilation of pre- and postconditions with object diagrams and the methods which will be called in between and reference to the corresponding test methods)
- Project report: guideline to the project, historic and management information, lots of documentation, documented design decisions (dynamic documentation)
- User manual, documented code and documented diagrams or references to documents (static documentation)
- Material for project presentation
- Version history

Presentation time 10 minutes. If you take longer, points in “demo & presentation” will be deducted. Realization programming language and gui can be freely chosen – however object oriented design must be possible and easily visible. We suggest using Fujaba, Java, and Swing. Make sure to practice your presentation. Presentation will be on December 7th. Make sure there are pdfs, pngs, or text versions available of documents in proprietary formats.

Course project grading table

Participation on presentation day: 1pt
Total points: 36 (7*5+1) + optionally 5 extra points for the use of Fujaba

Criteria	**Standards**				
	Poor (1pt)	**Lacking (2pt)**	**Pass (3pt)**	**Good (4pt)**	**Outstanding (5pt)**
System Design (1) Intial design [5 pts]	Userstories/ scenarios, object diagrams, usecases, class diagram, architectural overview are incomplete and faulty. This means any of these diagrams are missing or missdesigned (i.e. classes instead of objects), too small, or have a wrong level of abstraction (not concrete or abstract enough)		Userstories/ scenarios, object diagrams, usecases, class diagram, architectural overview are provided but they are incomplete or the design violates design principles. (>=15 User stories, >=20 Object diagrams)		Userstories/ scenarios, object diagrams, usecases, class diagram, architectural overview are provided and sound, interesting, and complete. (>=30 User stories, >=40 Object diagrams)

Criteria	Standards				
	Poor (1pt)	**Lacking (2pt)**	**Pass (3pt)**	**Good (4pt)**	**Outstanding (5pt)**
System Design (2) [5 pts]	Application of some Design Patterns visible, but wrongly used and not used often enough (however, also don't "over"-engineer).		Design patterns used, mostly correctly, some mistakes in application, some more design patterns could have been used.		Perfect usage of design patterns in the right places, well applied. MVC was used accurately.
Implemented Tests [5 pts]	Tests are present, but fail or don't match userstories		Tests work, mainly match user stories, but some important corner cases are missing. (At least 10 tests.)		All tests are present and match the userstories. (At least 15 tests.)
Functionality & Correctness [5 pts]	The system runs but is trivial in functionality or full of bugs.		Provides a reasonable level of functionality and relatively few bugs.		Provides an outstanding level of functionality and no obvious defects.
User Manual, Code documentation [5 pts]	Documentation is present but some parts are not covered, incomplete, or wrong. Cross references are incomplete.		User manual and documentation for all code and models is present, but part of the documentation is incomplete or faulty. Cross references between all artifacts (for example: object diagrams to userstories, scenarios to user stories, usecases to user stories) are present.		All parts of the code and model are documented well. Commits in the repository are well documented, too. Cross references are present. The User manual can be understood by a reasonable knowledgeable person not familiar with this system.
Report [5 pts]	Report is clearly incomplete (for example no report guide, no management description, or no decision guide). It is not clear that the objects first method was used to design the system.		Report has most or all of the required elements. It provides a rough guide through the repository but lacks certain areas. Also the part of participating people is not clearly visible. It is visible that the objects first method was used.		Report has all the elements and it is well-presented. It provides a good guide through the repository and explains all design decisions made. It also explains who managed/carried out which part and how problems were solved and coordination worked. It is evident how the objects firsts design method shaped the project.
Demo & presentation [5 pts]	The demo did not work properly, presentation was very poor		The demo worked but oral presentation was poor or answers to questions were not satisfactory		The demo fully worked and the oral presentation was clear and convincing
Fujaba Usage (extra) [5 pts]	Fujaba was correctly used for Designing class diagram and Storyboards fro tests		Fujaba was additionally correctly used for activity design, some activities are not done elegantly (not very object based).		Fujaba was used for all steps of the design and also to generate the model (in MVC) code. All Model-tests are realized with Fujaba Storyboards. Activities for all Model-transformations are well designed and functional.