Minecraft

Project for Object Oriented Software Engineering course

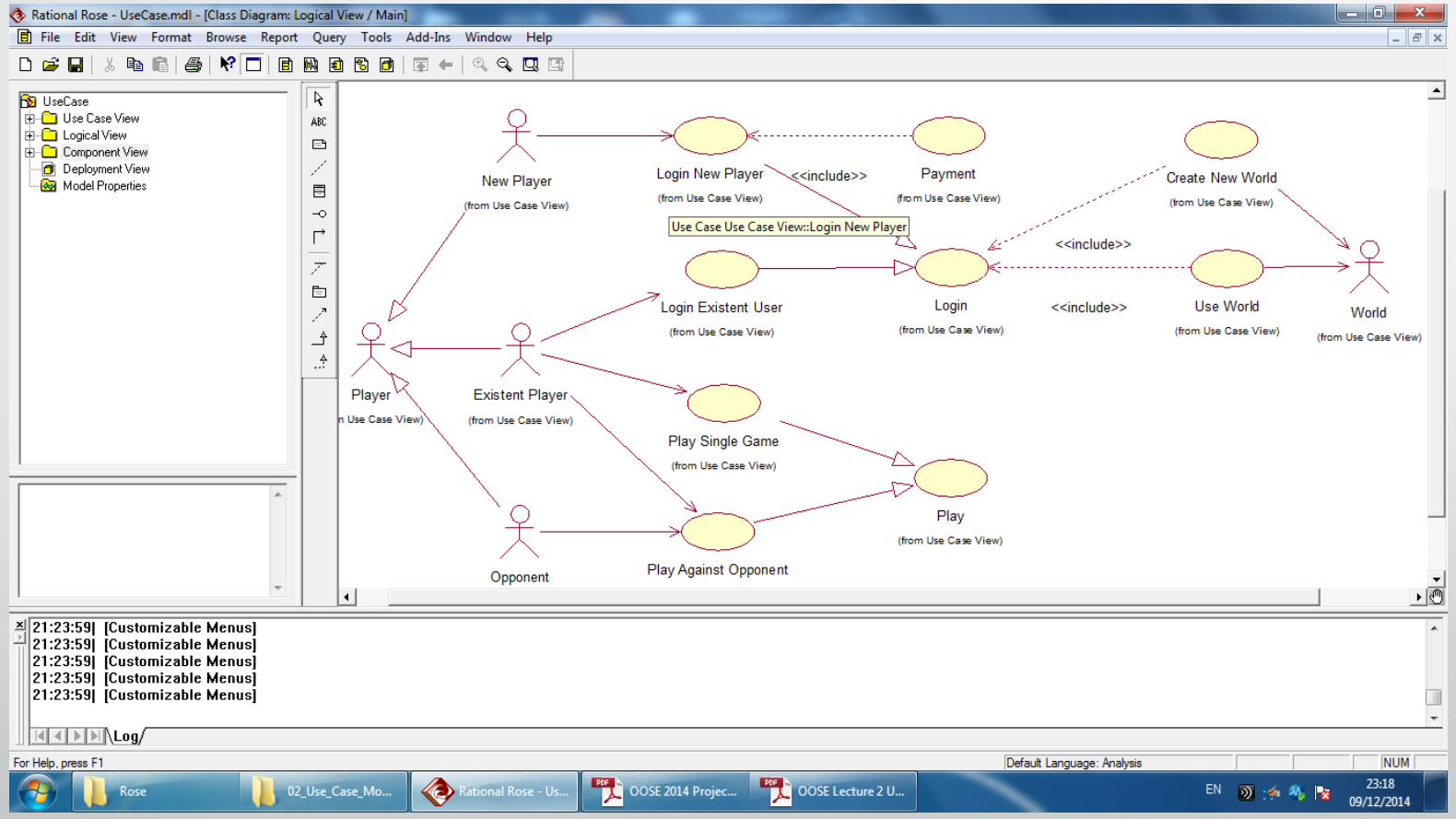
Index of Deliverables

- 1. Identify the **actors**.
- 2. Construct a **Use Case Model**.
- 3. Describe in **detail** any use case from the use case model. The use case must contain an alternate flow.
- 4. Create a **conceptual class diagram** of the chosen use case. The conceptual class diagram should demonstrate the use of attributes, relationships, navigability, association class, multiplicity and composition.
- 5. Create a **glossary** that lists and defines all the terms that require clarification .
- 6. Draw **a System Sequence diagram** from the conceptual model.
- 7. Develop a **Contract** for any system operation in the system sequence diagram.
- 8. Draw a **Collaboration diagram** based on the above contract. The collaboration diagram should demonstrate the use of design patterns.
- 9. Draw a **Component diagram** for the system.
- 10. Draw a **Deployment diagram** for the system.
- 11. Presentation (how well does the package of models look?).
- 12. Use of **Rational Rose**.

1. Identify the actors

- Player
  - Existent Player
  - New Player
  - Opponent
- World

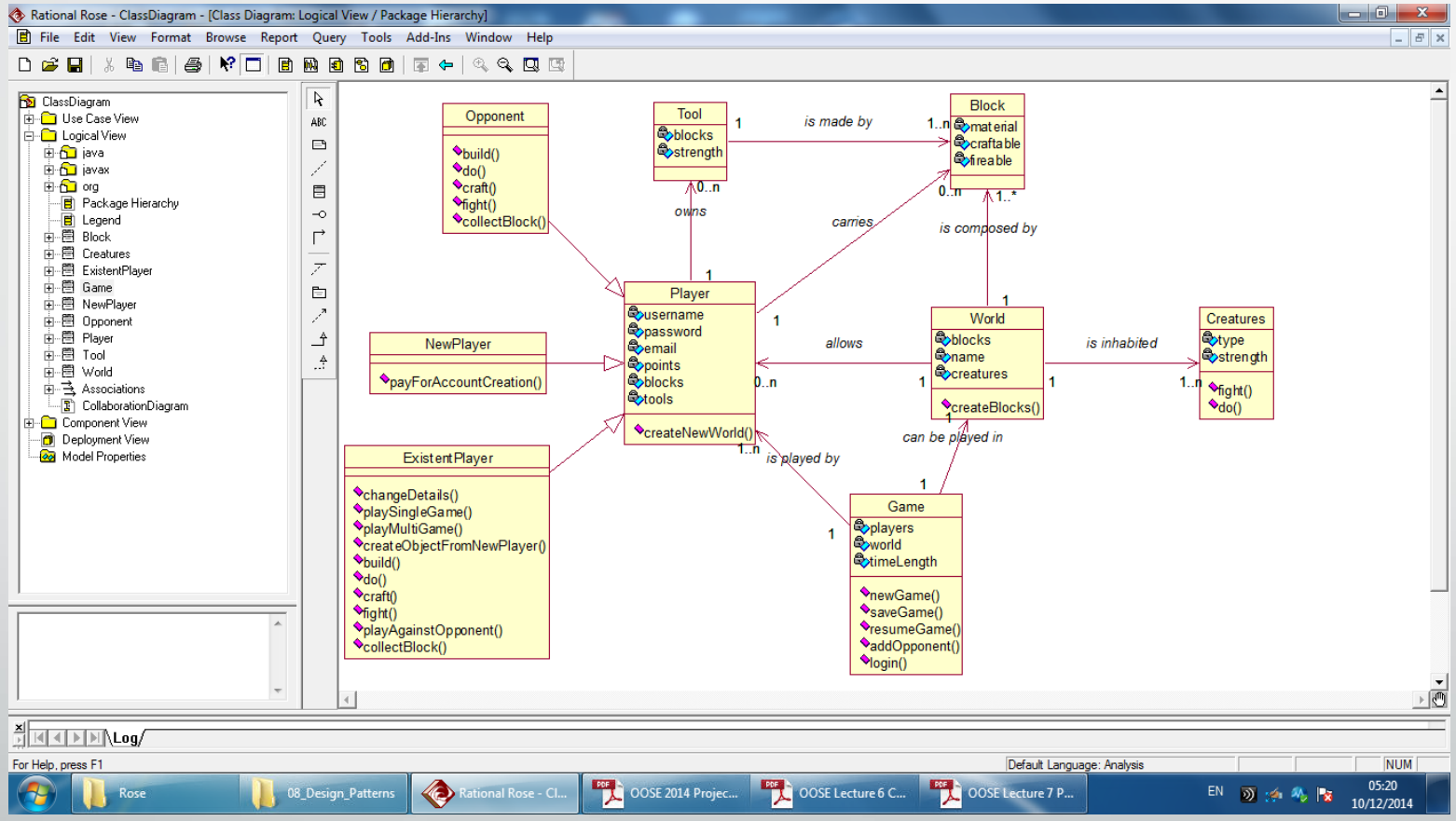
2. Use case diagram



3. Use case description

- **Login New Player**: It is used for the creation and login of a new player.
- **Login Existent User**: It is used for an already created user login.
- **Login**: Login use case.
- **Payment**: Included use case in case of creation of a new *Player*.
- **Use World**: It is used in case the player want to use a *World* already created.
- **Create new World**: It is used when the user wants to create a new *World*.
- **Play**: Play Minecraft.
- **Play Single Game**: It is used for a single player game. Extends *Play*.
- **Play Against Opponent**: It is used for multi player game.

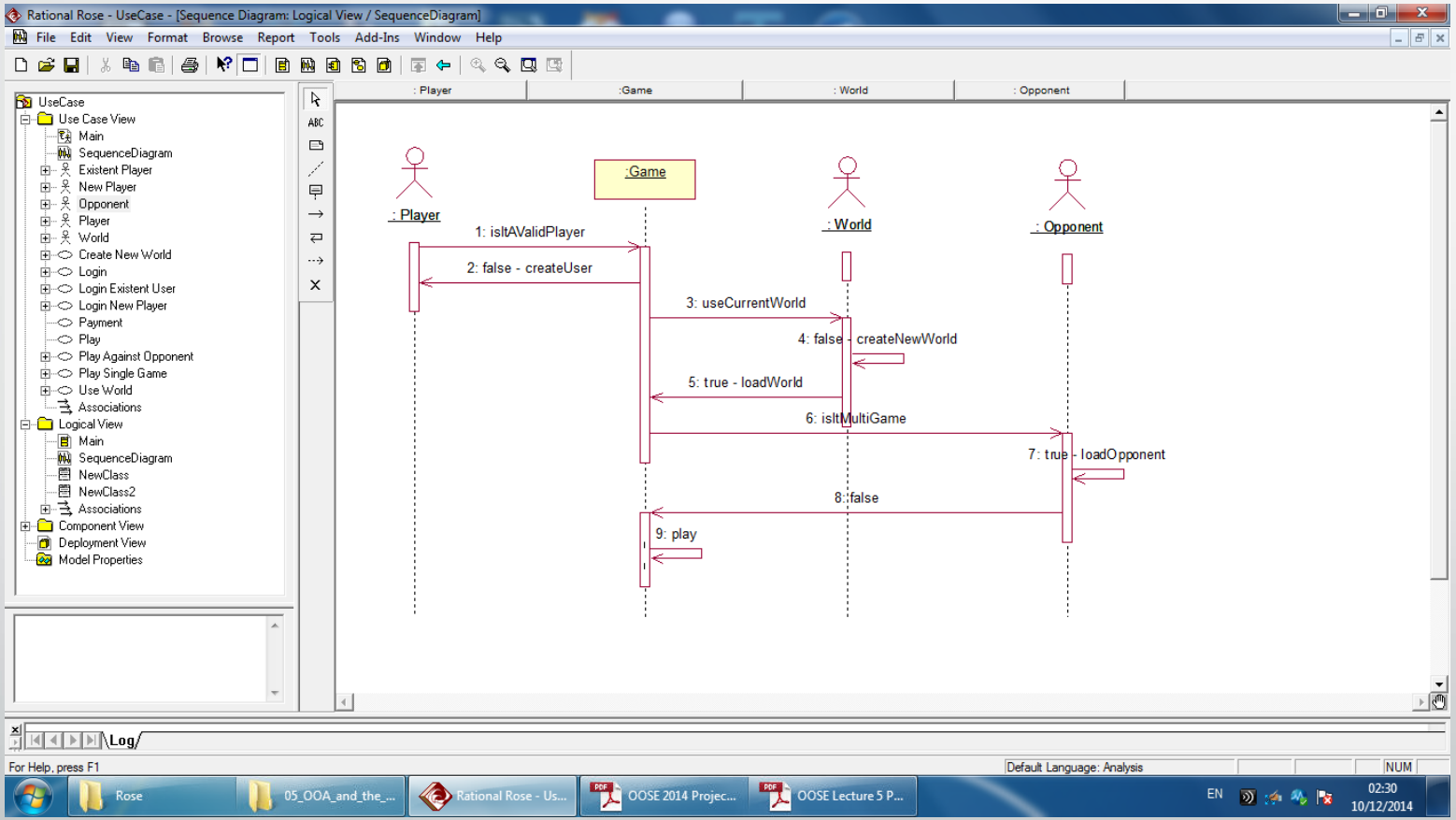
4. Conceptual class diagram



5. Glossary

- **Block**: Atomic unit of the game. Everything is made by Block.
- **Creature:** all the inhabitants of a world.
- **Game**: The game. It stores all the info related to the world, players etc.
- **Opponent**: Another character owned by a human player.
- **Player**: It is the character owned by human player.
- **Tool**: An artefact crafted by a player.
- **World**: Represent the game universe. Everything happens in a World.

6. System sequence diagram



7. Contract

Term	Description
Name	isItAValidPlayer
Responsibilities	Checks if the player is already registered or not.
Type	Game
Pre-conditions	Player username passed.
Post-conditions	• If the user is valid the player can load the game. • If the user wants to create an account will be redirect to the *create player* form.

Term	Description
Name	createUser
Responsibilities	Contains the logic to add the details of the players fee and payments details.
Type	Game
Pre-conditions	None.
Post-conditions	• If the user is valid the player can begin. • If the user is already present the error(s) will e highlighted. • If the payment details are wrong the error(s) will be highlighted.

7. Contract

Term	Description
Name	useCurrentWorld
Responsibilities	Playing in one of the world already created.
Type	Game
Pre-conditions	World(s) already created.
Post-conditions	• Provide a list of the worlds already created.

Term	Description
Name	createNewWorld
Responsibilities	Create a world.
Type	Game
Pre-conditions	A new world is created.
Post-conditions	• If the user is valid the player can begin. • If the user is already present the error(s) will e highlighted. • If the payment details are wrong the error(s) will be highlighted.

7. Contract

Term	Description
Name	loadWorld
Responsibilities	Load data of a created world.
Type	Game
Pre-conditions	World already created.
Post-conditions	• Game settings for the selected world.

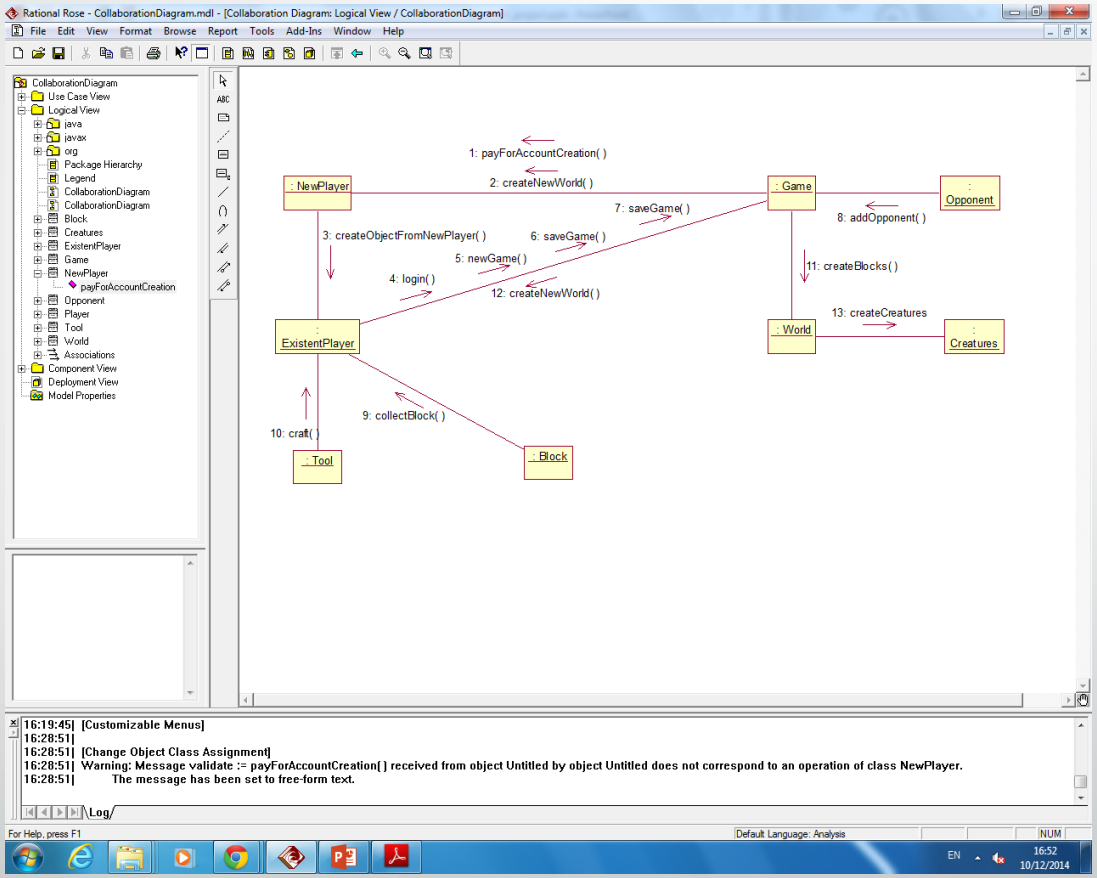
Term	Description
Name	isMultiGame
Responsibilities	Checks if is a single multi player game.
Type	Game
Pre-conditions	None
Post-conditions	• List of the possible online opponents.

7. Contract

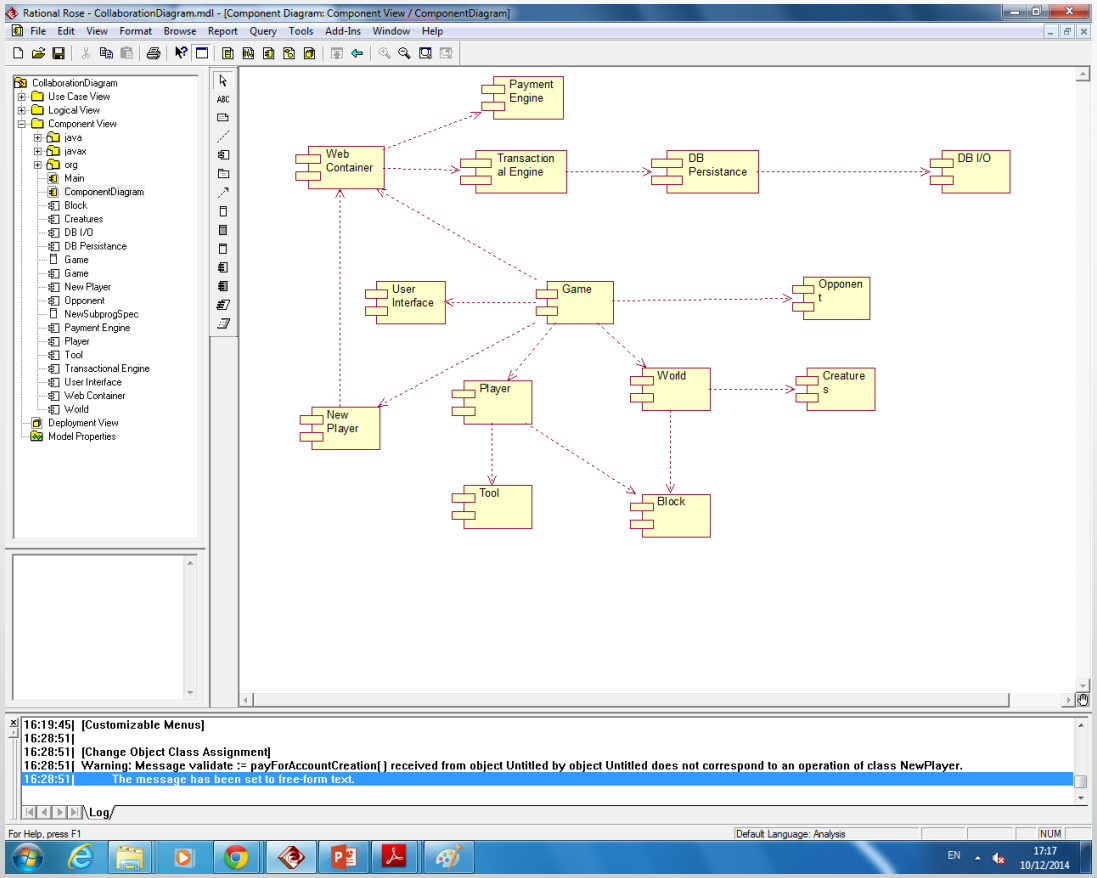
Term	Description
Name	loadOpponent
Responsibilities	Load into the world and the game the data of the opponent chosen.
Type	Game
Pre-conditions	• At least one online player. • Multi player game selected.
Post-conditions	• Player data loaded into the world.

Term	Description
Name	Play
Responsibilities	Manage all the game possibilities like collect block, build, craft, do actions, world and players management.
Type	Game
Pre-conditions	All the data should be correctly loaded.
Post-conditions	• Game, players, world data changed.

8. Collaboration diagram



9. Component diagram



10. Deployment diagram