TEAM KEPLER

IAN JOHNSON

ZACHARY GARCIA

DEVON MILLER

LOUIS MILLER

PROJECT OVERVIEW

- Our project is a point-and-click/text-based adventure game built in Twine, a game building tool that utilizes HTML, CSS, and JavaScript macros (built-in and our own source code).
- The game is structured using “passages” which contain either source code (variable initialization, audio/image caching, JS functions, etc.) or individual story pieces which can be traversed using links and can handle logic, variable updating, or other function calls to control the direction the user moves through the story.
- Progression through the game is driven by a blackboard system (similar to that of Firewatch) where all information about game state is held and then passages will execute differently based on the current state of the game and update the blackboard as necessary.

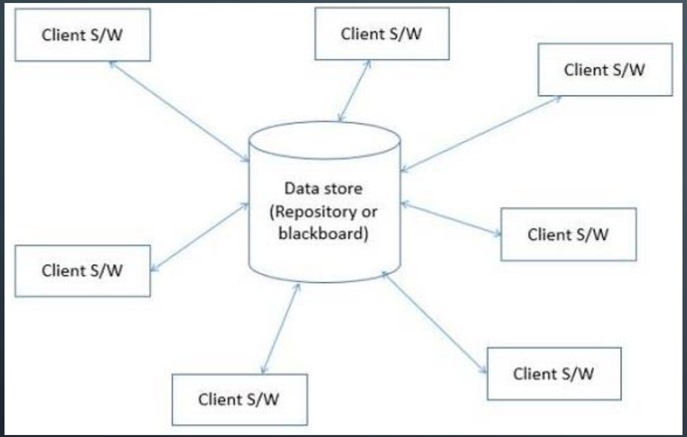
KEY ARCHITECTURAL DRIVERS

- Because progression through our game is based on user choice in relation to the current state of the game, we needed to implement an architecture that allows us to easily access the current game state information but would also fit into the structure of our development environment.
- Twine's structure limits us in what architectures can be implemented so we had to choose something that would fully function in Twine but also allow us to execute game logic and drive core game mechanics without extraneous changes to our overall code structure.

CHOOSING AN ARCHITECTURAL STYLE

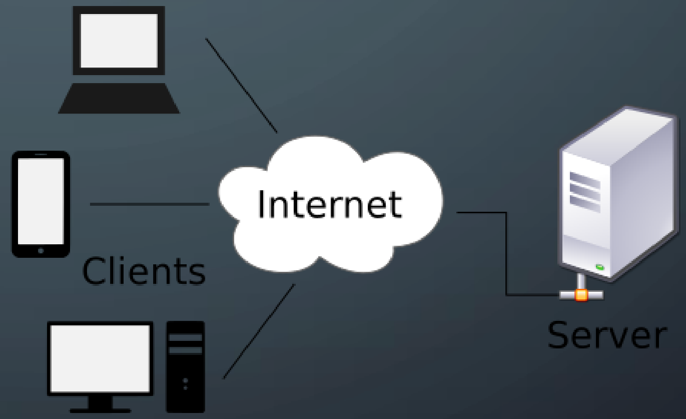
The two architectural styles we considered for our project were repository and client-server.

Repository: Because we are using a "blackboard" system, similar to the one we saw for Firewatch, to implement our features.



https://www.tutorialspoint.com/software_architecture_design/data_centered_architecture.htm

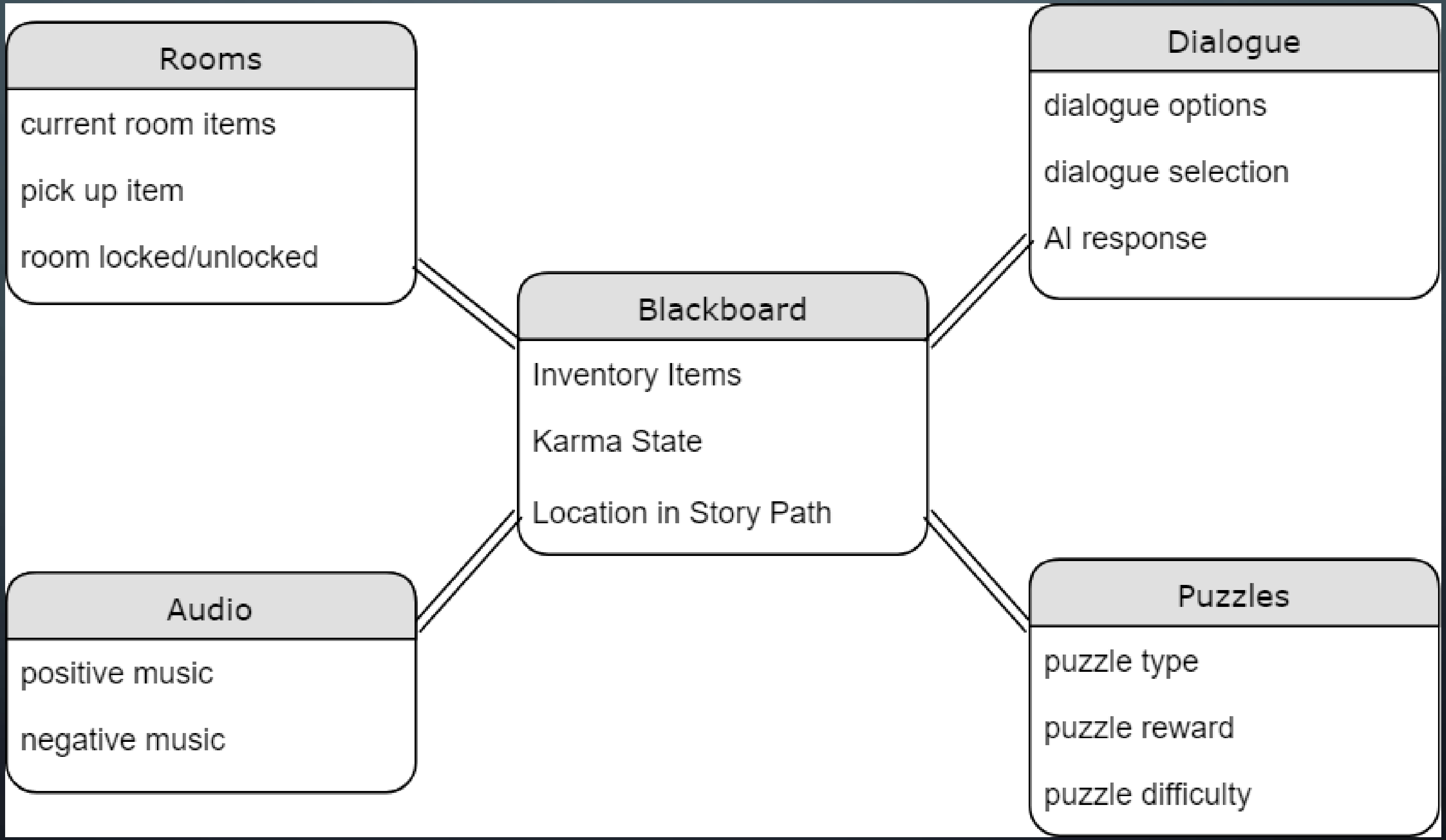
Client-server: Because the way we are deploying our game is on a web server, which is exactly the client-server model structure.



https://en.wikipedia.org/wiki/Client%E2%80%93server_model

We eventually decided that repository was our primary architectural style because that is the architectural model that we are using to actually develop the game and implement new features as opposed to just the deployment scheme of the project.

ARCHITECTURE DIAGRAM



CONCLUSION

- Primary project architecture is a <u>repository</u>-based blackboard or storyboard which contains information about the game state.
- The secondary architecture is <u>client-server</u> because the project is deployed and accessed through a website.
- Issues and Risks
  - Repository-based storyboarding approach using Twine forces us to use only HTML, CSS, and JavaScript with limited outside libraries.
  - Web-based game is dependent upon network connectivity (though we are looking into implementing a downloadable, offline version of the game)

THANK YOU!

ANY QUESTIONS, COMMENTS, OR RECOMMENDATIONS?