Torchlight Games Project Specification

Arcana

Project Description

Game Description

Arcana is a four-player maze game with magical elements and theming.

Three players are randomly placed in the middle of a procedurally generated maze and they have a limited time to find the exit. The fourth player is the “Dungeon Master,” who can place traps, change the layout of the maze, and perform other actions to stop the players from escaping the maze. Inside of the maze, there are various magical items that the players can find to help them defeat enemies and get past the traps and obstacles.

If we have the time, we would also like to implement a boss fight that the escapee’s have to fight once they escape the maze. This boss would be controlled by the dungeon master, and would incentivize players to stay in the maze longer to try and find better items before escaping, at the risk of potentially not making it out of the maze.

Game Objective

Players: The objective of the players is to escape the maze within the given time frame, while avoiding or overcoming the traps set by the dungeon master.

Dungeon Master: The Dungeon Master’s objective is to prevent the three players from escaping the maze under the time limit, using any traps or obstacles to their disposal.

Win Condition: If at least one player leaves the maze, the 3 escaping players win; otherwise, the Dungeon Master wins.

Interesting/Unique Aspects

Our game allows cooperation amongst 3 players to escape the maze from a single dungeon master. The players will work together to escape from the traps and monsters placed within the maze along with the distractions from the dungeon master. The randomness of the maze and enemies will allow enjoyable repeated plays of the game.

The role of the Dungeon Master offers a unique way for all players to interact with the maze, and has strategic elements for when and how to most effectively use the traps to block the other

players. The Dungeon Master has a top-down view of the maze and can place traps / change the maze layout to stop / delay the players from escaping.

When players are eliminated by the Dungeon Master, they respawn at a starting position within the maze and lose their previous progress towards escaping.

Features

Minimum Viable Product	- A single pregenerated maze layout w/ placed traps - 2D movement for players with proper collision - Basic traps (arrows, lava pit, etc.) - Dungeon Master with a single action
Must Have	- Several pre-designed mazes - Player jump - Basic modeling of the player / walls - Basic magical spells / items for the players to find - Simple (magic-based) enemy combat - Basic sound effects (footsteps, traps, enemies)
Nice to Have	- Procedural Generation of the maze - More complex player actions (climb, roll, sprint) - Physics based traps - More complex items/spells - Complex AI enemies (warren bear my belovèd, pathfinding) - Dungeon master with enemy manipulation / apply debuffs (almost RTS-y) - Text Communication - Better game music / sound effects
Only if Ahead of Schedule	- Added rooms for the maze (procedural generation) - Boss Fight - Regular or Proximity voice chat - More enemy types / AI - Character classes / specializations

Group Management

Major Roles / Decision Making

We chose not to specify any major roles for team management. For decisions, we will use our group meetings to discuss potential options or solutions and then come to a consensus together, whether by vote or mutual agreement.

Communication

- **Discord:** Main team communication (Using a dedicated Discord server for our group)

- **GitHub:** Project management / version control
- **Google Drive:** Notes / image sharing / design documents

Meeting Scheduling / Project Tracking

Weekly Meetings:

- Wednesdays 7:30pm: group meeting
- Tues/Thurs 9:30am - 11:00am: in-person programming / debugging sessions / meeting with professor/TA
- Sunday night: asynchronous/virtual status reports
    - If people are available, we will join a quick Discord call. If not, then we will leave Discord messages summarizing our statuses.
- Other subteams will schedule times to work together as needed.

Project Status Tracking

We will track the progress of various tasks and features using the **GitHub Project Board**. In our weekly meetings, we will review the tasks that were completed throughout the week, and then adjust future planning accordingly. This will allow us to incrementally track the progress of the game and provide flexibility for us to adjust the tasks as necessary.

Weekly Status Reports

All group members / subteams will create their own progress reports, and then each week these reports will be consolidated into a general group report. The group reports will be created by one person according to the following schedule:

- Week 2: Anthony
- Week 3: Ted
- Week 4: Tyler
- Week 5: Coco
- Week 6: Gil
- Week 7: David
- Week 8: Edward
- Week 9: Anthony
- Week 10: Gil

Project Development

Development Roles

Gameplay/Physics	Graphics/Client	Networking	Art
Ted Park Gil Keidar	David Min Anthony Tarbinian	Tyler Lentz Edward Jin	Jiawen (Coco) Wang

Development Tools

- **Programming Language:** C++
- **Build System:** CMake
- **Graphics:** OpenGL
- **Modeling / Design:** Blender, Procreate, Photoshop

Testing

- Unit Tests for individual modules / algorithms
    - Run Unit Tests on Github Actions CI
- Playtesting internally with group members
- Playtesting with friends/people not in the class/team

Documentation

- Internal Documentation:
    - We will explore using Doxygen to auto generate documentation files based on comments in the code. If this works out, we will upload the documentation to our group website.
- External Player Documentation:
    - General user tutorial / gameplay overview
        - Systems: maze, traps, win conditions
        - Escapers: movement, combat, items
        - Dungeon Master: camera movement, trap activation
    - More in-depth wiki for all items / enemies / features

Project Schedule

Our project will roughly follow the schedule as defined below:

Week	General	Gameplay	Graphics	Networking	Art	Design	Etc
2		Set up GameState skeleton	Moving wireframe cube	Echo server setup / Design basic protocol	Art Style inspo / research	Begin maze / trap design	Set up Repo structure / project management
3	Graphics Client/server integration	Basic player movement, Basic Enemy AI	Integrate w/ server	Event handling system	Character Sprites & Basic modeling	Items / combat / enemies	
4		Trap design/implementation, DM logic	Asset integration / texturing	Network handling for multiple players	Modeling	Finalize first maze	Basic sound effects
5	Finalize server / finish MVP	Advanced movement / combat	UI / menu integration		Finish basic assets/models		
6		Enemy AI / pathfinding	Camera logic for player / DM	Optimize protocol	UI Icons & Animation	Create more maps / map chunks	
7		Map generation	Animation				Better sound effects + game music
8	No new Architectural Changes		Shaders, particle effects				Create player documentation/tutorial
9	Playtesting / bugfixing / adding new content						
10	Finalize testing / demo	Grind the game	Grind the game	Grind the game	Grind the game	Grind the game	Create demo 'script'

Project Milestones

- **Graphics Client-Server Integration**
    - What it means:
        - One machine can run the server, one machine can run the client
        - The client is able to “connect” to a server. Depending on different user inputs, the server will tell the client to render various models

- **MVP**
    - What it means: All features defined as MVP will be implemented
    - How we know we have implemented it
        - multiple players can join a server and try and escape the maze in some time limit, with the dungeon master able to take actions to stop them
- **No New Architectural Changes**
    - What it means: all of the base systems & features of the game are complete
    - How we know we have implemented it:
        - We can solely focus on adding content to the game, and using the existing systems we have already developed in new and unique ways