PRODIGY (working title)

ABSTRACT

// MISSING //

INTRODUCTION

// MISSING //

LITERATURE REVIEW

**The Need**

|  |  |  |  |
| --- | --- | --- | --- |
| Index | Idea / principle | Topic / Section | Articles |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**E-Learning**

E-Learning Motivation

“The e-learning concept is very appealing. In particular, it allows a learner to study at their own pace, at times of their own choosing, and in an order that they prefer (Sloman, 2001). Delivery can also be cheaper, making e-learning more cost-effective overall (KnowITall, 2007), (Kiffmeyer, 2004).” (McGinnis, Bustard, Black, & Charles, 2008)

Unfulfilled Potential Of current Solutions

“Despite this potential, and some initial successes, e-learning systems do not yet have the impact that many believe is possible [4]. Moreover, the gap seems to be increasing because of the greater expectations of the current generation who have grown up with modern technology—the ‘Net-

Generation’ [5-7].” (McGinnis, Bustard, Black, & Charles, 2008)

Problems with current E-Learning systems

“In general, however, there are weaknesses in the implementation of e-learning systems. In particular, there has been insufficient attention paid to achieving compelling content and meeting the general needs and expectations of learners.” (McGinnis, Bustard, Black, & Charles, 2008)

Lack of Compelling Content

“Shneiderman & Kearsley argue that students must be meaningfully engaged in the learning material for effective learning to occur [24]. Engaging learners long enough to see them through to the end of a course has become one of the most significant problems faced by e-learning developers [8, 9]. This lack of engagement in electronic learning content can be attributed to three main issues: interaction, challenge and context.” (McGinnis, Bustard, Black, & Charles, 2008)

Expanding the three main issues mentioned above

*“Interaction*. It is generally agreed that interactivity is a critical factor in the design of e-learning systems [9, 25, 26]. Such interaction directly affects the learner’s overall experience and provides motivation to continue in the learning process. Studies researching the effectiveness of e-learning systems highlight the need for immediate feedback, clear short-term goals and better ‘flow’ in moving through the content [9, 26, 27]. The inherent fixed structure of many elearning systems often fails to provide adequate mechanisms to support interactivity between the user and the system. In many current cases the only interaction required by the learner is to click on the ‘next’ button to step through the material presented.

*Challenge*. Learners have indicated that unchallenging learning material fails to stimulate them, making the experience unattractive and discouraging progression [5, 28- 30]. As a result, many are reluctant to repeat this experience. Burns suggests that effective learning takes place when there is tension between the learner’s base knowledge and the gap between the knowledge or skill to be learned [31]. Such tension fosters a sense of curiosity and/or challenge. Motivation can be further enhanced by incorporating clear short-term goals and providing suitable feedback to encourage the learner to continue. Short-term goals help the learner break down a large task into smaller achievable chunks whilst the feedback gained through interaction helps the learner reflect on the learning process and lets them see the consequences of their actions [23].

*Context*. Current e-learning design often fails to situate the learner within the context of their course of study and provide them with a sense of orientation [26]. Students have stressed the importance of being able to appreciate the significance of their current progress in relation to the overall goal of the learning material and how their choices may have affected their progress [9].” (McGinnis, Bustard, Black, & Charles, 2008)

New Learner Needs

“Oblinger et al argue that the new learner has different needs that have to be addressed if e-learning is to be successful [9, 18]. The three main demands from the new learner are: (i) empowerment, (ii) social identity, and (iii) an authentic learning experience.

*Empowerment.* The new learner expects to be in control of their learning experience while in a supportive, collaborative environment. Thus e-learning systems should promote selfdirected learning. Unfortunately, many e-learning systems have a linear structure with a single path through the learning material. While this design is cost-effective, the lack of choice reduces control of the learning experience. Research suggests that having such control is more motivating [9, 31]. A suitable balance is required, however, as self directed study requires high self efficacy and vulnerable learners often lack the intrinsic motivation to manage their personal learning experience effectively [5, 7, 9, 29, 34].

*Social Identity*. Although current e-learning systems allow learners to move at their own pace it isolates them from their peers participating in the same learning process. This inhibits the learning achieved through social interaction and collaboration, with some learners feeling ‘lost’ [9]. Research indicates that a sense of belonging to a social group improves motivation and effective learning overall [9, 35].

*Authentic Learning Experience*. Reviews of the effectiveness of e-learning suggest content should be

challenging and stimulating. Learners expect the material to be linked to prior knowledge and be relevant to their everyday lives and careers. In short, the new learner is seeking an “authentic learning experience” [26, 36-40]. Also, most LMSs (Learning Management Systems) adopt a granular approach to

learning which is structured around small learning tasks. These support reusability but the decontextualised nature of these tasks is in conflict with the learner’s need for a contextualized experience [35, 41]. Generally, learners are more engaged when they are participating in activities that they can relate directly to prior knowledge and make connections between what they are learning and the real world [40, 42]. If such links are missing, learners are less inclined to participate in the learning process and may see it as pointless [9, 24]. For the new generation who are used to customizing

their environment [9] there needs to be flexibility in the order and way in which material is studied. Research indicates that such personalisation helps learners take ownership of the learning process and promotes engagement [43].” (McGinnis, Bustard, Black, & Charles, 2008)

**Games**

Games and E-Learning designer comparison

“The primary goal of a game designer is to create a game that players find appealing whereas e-learning system designers tend to focus on educational content, leave the student to find the necessary motivation to study that content [8].” (McGinnis, Bustard, Black, & Charles, 2008)

Characteristics of Games (long)

“Goals help maintain engagement and provide motivation for the player to gradually progress through a game. Without clear and compelling goals players easily become bored as they need to know how their actions are directly related to their progression towards the overall goal of winning the game [47].” (McGinnis, Bustard, Black, & Charles, 2008)

Quest Game Progress (if we choose to implement it)

Flow (insert references from Csikszentmihalyi)

“In his book “*Flow, The Psychology of Optimal Experience*” Csikszentmihalyi [49] explains flow as an “optimal experience, which includes a sense of exhilaration along with a deep sense of enjoyment”. Flow theory “proposes that clear goals, achievable challenges and accurate feedback are required to achieve a state of flow in an activity” [49]. Csikszentmihalyi describes how a player begins with a low set

of skills which is matched with a low challenge within the game. As the player progresses, developing skills, the difficulty level of the challenges encountered also increases. If 126 the difficulty increases too steeply, the player will become anxious and discouraged. Alternatively, if the challenge is too low, the player may lose interest and disengage. Csikszentmihalyi describes the path between these two extremes as the “Flow Channel”. However, games with a uniform intensity gradient also become boring. Introducing variety on the slope produces periods of accelerated difficulty mixed with periods of lesser difficulty. This allows the player time to recover from the more intense parts of the game and digest the immediate feedback obtained from them. This classic games structure (Figure 4) shows a curving difficulty progression within a series of convexities, which help engage and immerse the player by using a balance of challenge and skill levels. At the same time the alternating difficulty level gives the player a chance to review old skills and learn new ones without stepping outside the flow channel.” (McGinnis, Bustard, Black, & Charles, 2008)

**Computer Game Enhanced E-Learning Systems**

User Centered Design

“Current e-learning systems, which tend to focus on the management and delivery of content often, overlook the importance of the user within the process. Lessons can be learned here from game design where there is considerable attention paid to the needs and expectation of players.” (McGinnis, Bustard, Black, & Charles, 2008)

* User Progression Relative To Peers

“At a more detailed level, a learner also wants to know how well they are progressing in relation to their peers and the expectations of their teachers.” (McGinnis, Bustard, Black, & Charles, 2008)

Quest Game Progress (if we choose to implement it) – applied in E-Learning Systems

English

Games and E-Learning

English, Games and E-Learning

GAME DESIGN

**Game Goals**

Principle: Provide an obvious goal. (Malone T. W., 1980)

The goal at each level is to complete a number of questions without losing all hit points (without “dying”).

Principle: Players must be able to tell whether they are getting closer to the goal. (Malone T. W., 1980)

The amount of questions completed and those remaining are always visible to the player. The player is indicated as he gets closer to the goal, and specifically when at a point where he is about to accomplish the goal (the last question for example).

**Uncertain Outcome**

Principle: Variable difficulty level (Malone T. W., 1980)

Levels in a section are tagged by a difficulty level. There are three mutually exclusive difficulty levels:

1. Elementary
2. Intermediate
3. Advanced

We consider two ways of determining the difficulty levels:

1. Automatically by the system (Malone T. W., 1980). Depends on the score the user received at the section. Once a threshold is exceeded, a new difficulty level is available, which introduces new levels at that difficulty level.
2. Chosen by the player (Malone T. W., 1980). Two alternatives are considered
   1. The player may select the difficulty level he wishes to play the section at. This case is mostly relevant if the progression through difficulty levels must be done linearly (first accomplish section at Elementary, then at Intermediate, etc.).
   2. The following diagram illustrates the second approach (based on Game Convexities as described in (McGinnis, Bustard, Black, & Charles, 2008)).

All levels are ordered linearly and tagged with one or more difficulty levels. The section may be completed at each difficulty level that appears in at least one node. To get access to higher level material, the player must first complete all levels at the prior difficulty level. We note however that access to more difficult material may be productive and challenging to the player, and this is something we should consider. If this is the case, accomplishing a previous node may unlock all content at one difficulty level higher until the next node with the current difficulty level is met.

Another issue regards the following: should a node at some difficulty level be unlocked only when previous same difficulty level nodes have been accomplished?

Example:

Starting difficulty level = Elementary

Entry node = level 1

Exit node = level 6

Complete level 1

Unlock level 2.

If ( Level 2 )

Complete Level 2

Complete Level 3

Unlock level 5.

If ( level 5)

Complete level 5

Complete level 6

End section.

Starting difficulty level = Intermediate

Entry node = level 2

Exit node = level 5

If (level 2 incomplete)

Complete level 2

If (level 5 incomplete)

Complete level 5

End section

Level 1

Elementary

Level 3

Elementary

Level 5

Intermediate

Level 6

Elementary

Level 4

Advanced

Level 2

Intermediate

Level 7

Advanced

Principle: Multiple Level Goals (Malone T. W., 1980).

The following table illustrates the concept of level progression and how we derive the level goals.

|  |  |  |
| --- | --- | --- |
| Question Order of Appearance | Points Needed to Reach Question | Power |
|  |  |  |
|  |  |  |
| .  .  . | | |
|  |  |  |

At any stage of appearance, the player is required to accumulate a number of points in order to progress to the next question (). Every is lower bounded by the previous question’s value, and upper bounded by the sum of the previous question’s value, and the total amount of points the question is worth. For consistency’s sake - .

We can now determine a level goal at the number of question to reach. In case we use three level goals:

Level Goal 1 – Reach Question 15 (complete 14 questions)

Level Goal 2 – Reach Question 23 (complete 22 questions)

Level Goal 3 – Complete All

Special attention should be given the information available to the player when using this kind of level design. In particular, the information relating to how the user is doing, what he needs to do next, where is he located in relative to a level goal, and what moves will result in him being unable to accomplish a level goal.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Q | Hit Points | Hit point Threshold To Reach Question | Min Hit Points to Pass | Acc. Hit Points | Answer Parts | Size |
| 1 | 5 | 0 | 4 | 5 | 1, 1, 2, 1 | 4 |
| 2 | 11 | 4 | 9 | 16 | 2, 2, 2, 2, 2, 1 | 6 |
| 3 | 8 | 13 | 8 | 24 | 1, 3, 1, 1, 1, 1 | 6 |
| 4 | 10 | 21 | 7 | 34 | 2, 4, 2, 1, 1 | 5 |

Principle: Players must be able to know how they are doing at each moment and how close they are getting to a goal.

**Game Play**

Hit Points

Every answer part of question is assigned some integer number that is larger than one. This number is called a Hit Point. It serves a few purposes:

1. To increase relative significance of parts in an answer.
2. In case of the answer part entity: To maintain the number of mistakes allowed per answer part.

A question’s hit points are calculated as the sum of the parts of it’s answer.

Reward System

* Hit Points

Hit points is an integer number in the range [0,100] that is used to keep track of the amount of damage entities may endure before being incapacitated. This occurs when the hit point level reaches zero. At one hundred, the entity is perfectly undamaged. Incapacitated Question entities are considered *complete,* while when a Character loses all of its hit points, the game ends with or without accomplishing a goal.

* Level Hit Points Multiplier

This is a non-negative Real in the range (1, INFINITY]. The multiplier is used to grant more points at more advanced levels in a section.

* Question Hit Points

There are a total of 100 points (percent) to earn at every stage. Every question is assigned an amount of hit points [0,100].

* Power Ups
* Combo
* Score

The Score keeping mechanism is fairly straight forward.

* Automatically varying in-game difficulty

**Multiplayer Gameplay (superset of single player mode)**

A game instance (round) happens in a single level. At least one player is required to start a game.

The players are divided into two teams, each with a maximum limit of players. The purpose of the teams is to gain control over all questions in a level. This is a team effort, which also incorporates some vs. elements.

The game ends when all questions have been completed, or when a team loses all its combined hit points.

The game starts with the teams at different challenges, and the players scattered randomly across different questions within the starting challenge.

The choices available to a question are hidden until the question is engaged. This forces the player to first focus on the question at hand, and not the available choices. It also requires a certain strategy of question engagement, meaning that players must first engage the questions they are most confident about.

Once a player engages a question, the available choices are discovered. The question is now occupied by the engaging player, meaning that no other player may submit answers to that question. Players are however able to see the answers that are submitted by others.

Hit points are lost if a player submits an incorrect answer.

Every challenge in a quest (question) is supplemented with a set of possible answers. This is the main interaction. A player clicks on an answer and receives feedback. Sound shall be used to enhance the experience, indicating a correct answer or a mistake.

Engage Question Use-case

User is at an incomplete question

User clicks the engage button.

The event details are propagated to the server. The Question is added to the client and locked from other clients.

The choices and solutions are returned.

The user is at index=0 of the answer. He clicks a choice. The following happens:

1. At client side, all the answers that contain a matching choice at current answer index are retrieved.
2. If no answers match
   1. Show choice in red.
   2. Update screen with new game state.
   3. Send event details to server.
   4. Update screen with game state from server (happens when game states differ).
3. Else one or more answers are found
   1. Show selected choice in green and add to the formed choice sequence.
   2. From the matching answers, if one is found such that: answer index == answer.size-1
      1. Complete answer submitted.
      2. Repeat 2.b – 2.d.
4. Repeat all until all answers are discovered.

Features

Glossary

MAX\_CHOICES = #maximum allowed unique choices in a question = 8

MIN\_CHOICES = #minimum allowed unique choices in a question = 1

MAX\_ANSWER\_SIZE = #maximum allowed sequence of choices that makes an answer.

MAX\_DUMMIES = #maximum allowed dummy choices in a question = MAX\_CHOICES – 1

MIN\_DUMMIES = #minimum allowed unique dummies in a question = 0

DUMMIESq = #dummy choices in question q

REAL\_CHOICESi = #correct choices that can be submitted at location i of an answer

CHOICE\_SCENARIO = an empty sequence of x choices, together with the available choices.

DUMMY\_HIT\_PROBABILITYi = #dummies / #reals

HIT\_POINTS\_QUESTION = 100

SOLUTION\_SIZE = #answers left in question (>0)

COMBO = #consecutively submitted correct choices in a question

HIT\_POINTS\_MULTIPLIER = COMBO \* some normalizing factor

HIT\_POINTS\_CORRECT\_ANSWER = HIT\_POINTS \_QUESTION / #answers in question

HIT\_POINTS\_ANSWER =

QUESTION\_STREAK = #question eliminated in a row without dying.

HIT\_POINTS\_ASSIST =

The purpose and goal of each level is to accumulate as much intelligence point as possible. This is done through completing questions. Every level carries a number that acts as a hit point multiplier. This is to create an illusion of higher difficulty

Intelligence Power

This is the game score.

Every answer in a question is assigned an amount of hit points. The number of hit points granted for submitting a correct answer is calculated as a fraction of the total number of answers in a question.

The players start with an intelligence level at zero.

When an answer is correctly submitted, the player‘s intelligence level increments by the answers hit point amount, and the question loses the same amount.

When an incorrect answer is submitted, the player loses the same amount of hit points from his health.

Combo is an intelligence multiplier that takes affect when a correct answer is submitted. It starts counting when two consecutive choices have been correctly submitted, and increments with each correctly selected choice.

It rewards players for submitting a long answer; those that are made of a number of consecutive choices.

XP (Intelligence Level)

Player is granted points after every move which results in a positive outcome. For submitting a correct answer, player is granted an amount of points equal (or multiplied by some constant factor) to the weight of the missing piece. For the correct use of any item, the player is rewarded a constant amount of bonus points. Optionally, each item may have its own XP scheme, where the points relevant to the used item are increased, thus allowing upgrades.

Elimination

The ability to eliminate pieces that are irrelevant to the current challenge. This is done by first toggling the eliminate button on (or two finger touch if touch is implemented), and then clicking on an answer to eliminate. Once an answer is eliminated it does not appear as an available answer in the challenge any more - the shield stripe is shrunk. No feedback is received when eliminating an answer but additional points are granted at challenge end for correct use of ability.

Elimination is required to start a combo (?)

Submit

Submitting an answer is done by clicking on a shield stripe. When a correct answer is submitted, the UI indicates it to the user – the shield stripe is shrunk, combo increases if possible, points are rewarded and a written feedback is shown ,such as “good” or “great”.

Combo (Correct Answer Streak)

Combo is an intelligence multiplier that takes affect when a correct answer is submitted. It starts counting when two consecutive choices have been correctly submitted, and increments with each correctly selected choice.

It rewards players for submitting a long answer; those that are made of a number of consecutive choices.

Answer Streak

Shield Power

At the end of each challenge, the shield is charged based on the amount of correctly used stripes. The shield remains charge throughout all challenges in a level. It enables the use of Special Items.

The charge is based on the amount of expanded stripes at the beginning of each challenge. Every shrunk stripe (resulting from correct use only, either by submission or elimination) charges the shield with some amount. Combo should affect the shield charge.

Magic Items (Special Abilities)

The player can use special items (e.g. health or help) where each use of an item results in either a positive or negative outcome. An item must first be charged before it can be used (this can be achieved by either the combo mechanism, or using some sort of “fuel” to charge items up before or during gameplay.).

Magic Items are charged using the shield power.

// BEGIN IRRELEVANT

Once an item is selected and charged, the outcome depends on the result of the next answer submission. If the submitted answer was correct, the positive outcome of the item is in effect. If the answer was incorrect, the negative outcome is in effect.

// END IRRELEVANT

Some of the items are: (perhaps divide to different characters)

1. Heal: Increases player health by some amount. Very hard to charge.
2. Time Extension: Stops clock count for X seconds, where X is determined as a constant fraction of the initial clock time. Relevant only for timed quests.
3. Undo: provides undo operations of the last submitted piece. Cannot be used after last piece is submitted.
4. Armor – protects player from health decrease if wrong answer is submitted.
5. Wild Card – a “Joker” answer. Always correct, but the real answer is not revealed.
6. Automatic Elimination: automatically eliminates one (or possibly more) piece that is irrelevant to the current missing piece.
7. Reveal – Marks all correct choices of a solution in sequence. Starts with the first choice, if the player clicks on it, the next correct choice is marked. Currently the most powerful item.

Achievement Levels (Intelligence)

An achievement level is a condition that is evaluated during a quest, which concerns either the percentage of challenges to conquer, or that of pieces to be submitted correctly. There are three (number should not be hardcoded) achievement levels in which a character may complete a puzzle. A quest master must determine values for all three conditions when creating a quest.

**A quest is completed if during play, at least one of the achievement levels is reached.**

For example:

1. 60% = Easy
2. 80% = Normal
3. 100%+ = Hard

Additional constraints may be imposed in case of timed quests.

Solution Granting

Once a quest has been completed, the character is granted the correct answers (solution) which were submitted by him. Once a solution is in possession, it is available for review purposes at the library (a place where characters may meet, and review passed quests and their solutions).

World Progression

The world is an ordered list of nodes. Characters progress from one node to the next after accomplishing all assignments at a node. A node contains one or more quests, and assignments impose conditions on how to accomplish quests.

Solution Ranking Algorithm

The algorithm acknowledges the order of answers in submitted solutions, calculates the amount of discrepancies between

Let be the current challenge, and be the set of all existing solutions in the challenge.

We define as a vector of strings constructed from the available answers in .

Let be a predefined solution to , where is an integer in the range , and as the solution submitted by the user , where .

We refer to an item in a solution vector as , where is an integer in the range .

Finally, we define to be an item with an empty string.

The goal of the algorithm is to remove items from so that eventually it shall contain a single solution closest to .

The algorithm returns an integer vector ,where , and which holds the following flags:

1. : indicating a correct answer at index j.
2. : indicating an incorrect answer at index j.

We define the following functions:

1. : retains all solutions that contain the item .
2. : retains all solutions that equal the length of the second argument.
3. :

ANALYSIS AND DESIGN

Requirements

User tasks:

1. Create the list of subjects.
2. Create a number of multiple choice exercises (group of questions) for each subject.
3. Assign introductory and summary text for each exercise.
4. Create questions for each exercise, with one or more answers to each question.
5. Assign a feedback text to various possible answers for a question.
6. Create and add textual theoretical background to be accessible for reference by the student at each subject.
7. Access student specific progress information:
   1. Average score in each completed level
   2. Number of questions completed successfully
   3. Number of questions not completed
   4. Average number of mistakes per question (also by subject)
   5. Time spent in subjects
8. Support the following two ways of submitting answers to a question:
   1. By building a complete answer from the choices in a ordered fashion, starting from the first missing part and until the answer is complete.
   2. By selecting the missing place in the answer to which the user wishes to submit a choice to.
9. Control the progression of students through the syllabus by setting a minimal passing criterion in each level.
10. Allow student to undo a submission in certain scenarios.
11. Consider support for multiplayer gaming.
12. Mark words that appear in the question headline as Bold, Italic, and Underscored.
13. Provide multiple answers to the same question.
14. Provide a template for an answer, from which words are missing.
15. Provide answer that are made of multiple parts, where the order of the parts doesn’t matter.
16. Each level playable at different difficulty levels.
17. Define multiple levels of goals in each level:
    1. Example: By the amount of questions that must be completed / intelligence points gained.
18. Allow use of special game clues.

Use Cases

UC3: Engage Questions in Game

Actor: Student

User selects a question to answer. He presses the engage button. The questions choices appear to the user. The buttons in the choices are expanded and populated with text. Up to 8 unique choices can appear at once. If the question holds more than the maximum, the choices update after the first few have been chosen.

At all times, the on screen choices are taken from the current location in the sequence and onwards, with regards to the answers corresponding the selected sequence. For example, if the question contains one answer <A, B, C, D, E>, the choices shown at the beginning (location at index zero, matching letter A) are {A,B,C,D,E}. At the second location, index 1, the choices are {B,C,D,E} and so on.

Select Choice

The basic logic of selecting a choice.

Repeat process until one of the following happens:

1. A choice sequence has been completed and matches an answer.
2. User has falsely selected choices more than the allowed number.

User clicks on a choice.

If the choice is correct

Append choice to on screen sequence

Increment combo count by 1

Increment number of correct choices by 1

Decrease 1 hit point from question

Increase user hit points by 1

Else

Append the token to the choice sequence but mark as a mistake

Reset user combo to default value

The question does not lose a hit point

If the number of hit points left in the question is lower than the amount required for achieving the minimal goal

Disengage question and lock from user

If the number of available choices is greater than 2

Show

Complete description of a single player game

A player signs in the system. He clicks the button that says ‘start’. Because the player is new, the first section of the game is shown, together with the number of levels. The latest accessible level is presented (which in this case is the first), and its details are shown:

* The introduction
* The instructions
* The difficulty level
* The goal levels that the level can be accomplished at: A goal level sets the minimal amount of intelligence points a user must accomplish in order to reach this level’s goals. For every goal that is reached, the user receives an amount of bonus points and some label, such as “GrammarKing” or perhaps something more specific to the section he is at, such as “The Count of Counting” in the “countables” section. (these may also be unlocked when playing in multiplayer mode)

The player clicks the “Play” button, which starts the game. The level data is loaded at the server and client, and once complete, the game starts at the first question.

The player now sees on screen the following:

* The headline of the question
* The number of intelligence points accumulated in the level so far (starting at zero)
* The

Server Side Development

(*inspired by sourceforge,net/apps/trac/svbg/wiki/Server-side%20Development*)

Design Rationale

High Priority

* There should be no UI necessary to test and develop server-side code.
* The server implements all game logic. Client inputs should be validated.
* The server should know the expected task for each client at each point.
  + Which player’s turn is it? reject input from other users (multiplayer)
  + Which task is expected by that player? Reject invalid input.
  + Provide end-user compatible error messages.

Lower Priority

* The server should know the data needs to be sent to a client. The data should be minimized, e.g. by sending incremental data.
* It should be possible to replay a complete game and/or have an overview of what happened in which turn after the game is finished.
* The server should minimize its own state.
* The server should be able to restore its state form persistent storage.
* The game logic should be implemented in a readable, extensible way. A state machine? Rules?

Nice To Have

* Undo/Redo functionality.
* An overview of how much time each player requires.

Client Server Communication Protocol

Single Player

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Initiator | Key | Value | Destination Process | Result | Notes |
| Client | Play Game | Level Number | Load data, send data to client, start game when client signals that data is loaded and ready |  | Request to start a new game |
| **Client** | **Move To Question Number** | **Question Number** | **Update player location** | **Notify all (multiplayer)** |  |
| Client | Engage Question |  | Validate that question is not previously engaged. Assign Question as occupied by user. | Notify all  (multiplayer) |  |
| Client | Select Choice | Choice Token | Validate selected Token, place token at question answer, at the current user location | Progress location by 1 if possible |  |
| Client | Disengage Question |  | Validate that user is able to disengage. | Notify all |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Multi Player

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Initiator | Key | Value | Destination Process | Return | Notes |
| Client | Join Game | Level Number | Load data, send data to client, start game when client signals that data is loaded and ready |  | Request to start a new game |
| **Client** | **Move To Question Number** | **Question Number** | **Update player location** | **Notify all (multiplayer)** |  |
| Client | Engage Question |  | Validate that question is not previously engaged. Assign Question as occupied by user. | Notify all  (multiplayer) |  |
| Client | Select Choice | Choice Token | Validate selected Token, place token at question answer, at the current user location |  |  |
| Client | Disengage Question |  | Validate that user is able to disengage. | Notify all |  |
| Client | Use Armor |  | Validate action |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

DATA DICTIONARY

LITERATURE GLOSSARY