

Circus Tycoon

Alejandro Lopez-Lago

Miles O'Connell

Josh Siegel

Lucy Vasserman

September 13, 2009

1 High Concept Statement

The player in Circus Tycoon is the owner and ringmaster of a traveling circus business. The player starts out in a small town with a few dollars, a map of the town, and a couple circus attractions. The ultimate goal is to maximize profitability of the circus. To meet this goal, the player uses given maps to travel around local towns or cities while buying items for the circus that will increase attraction (and therefore ticket sales). As the player progresses through the game and saves up money, he will also have the opportunity to move the circus to more profitable cities on the global map. The player will need keen map reading skills and a sharp mind to efficiently travel around cities to find circus attractions and complete mini puzzles and challenges.

Featuring...

- Fantastical setting
- Interactive maps and route planning
- Economic incentives and feedback

2 Competitive Analysis

Geo Challenge

Geo Challenge is a game on Facebook that contains several geography-related minigames. Three of them deal with map literacy: one was a simple “identify the country from its profile” minigame while the other two required players to click on the location of a city/monument on an unmarked map.

Pros	Cons	Comparison
<ul style="list-style-type: none">* Slightly addicting gameplay due to the ease at improving your score and surpassing your friend’s score.* Educates players about the locations of cities and monuments on a map (without giving them country labels).* User interface is simple and easy to learn (mostly point-and-click).	<ul style="list-style-type: none">* The full game costs money* For the find the city/monument game, the resolution of the map is too small (after a while you just click somewhere on the country since you can’t really position your mouse to where it should be).	<ul style="list-style-type: none">* Geo Challenge only uses world maps while our game will cover several types.* We include the circus, travel, and fantasy aspects adding depth to our game that is lacking here.* They game resides in Facebook while ours will likely be a standalone web-app or regular application.

Where in the World is Carmen Sandiego?

In this famous game, players travel to various locations around the world picking up clues to catch a villain. The game has definitely withstood the test of time, beginning in 1986 and maintaining popularity through the 1990s.

Pros	Cons	Comparison
<ul style="list-style-type: none">* Mystery solving story is very fun.* Problem solving and story line is appealing to both girls and boys - a rarity in children’s games.* Successful in teaching geography by showing maps noting names of countries and cities frequently and having appropriate time intervals pass during travel between locations.	<ul style="list-style-type: none">* Advertises teaching cultural information, but information given is very limited and often stereotypical.* Mystery solving and time limit discourages exploration. Students are looking to find only enough information to continue the game.	<ul style="list-style-type: none">* WitWiCS does not attempt to teach map and navigation skills as we will.* We plan to encourage more autonomy and personal initiative by giving students more options of where to go and how to get there and the tools to determine these things.* Mini-games and circus tasks distributed throughout cities in our game will encourage exploration and therefore more learning about local maps and culture, as opposed to the timed mystery solving.

Maps.com

This site has a set of games that are typical of the map-based educational games available today. These games all follow the same format and are focused on clicking on desired locations on unlabeled maps. They occasionally include trivia about cultural and geographic information. Variations between games are come from map scale and difficulty, and so these are considered together.

Pros	Cons	Comparison
<ul style="list-style-type: none">* These games truly encourage geographical knowledge.* Definite room for learning and improvement with correct answers given when players are incorrect.	<ul style="list-style-type: none">* These games are boring.	<ul style="list-style-type: none">* These games do not teach only geography, not map reading skills. We hope to teach students how to get themselves to use maps to travel, not just how to read them.* Circus theme and fantasy setting will make our game more engaging.

Sim City (series)

Sim City is a city management simulation game in which the player is responsible for building and maintaining a city as the “mayor” of that city. The player is presented with a map (grid) of the city and given tools to build residential units, commercial units, etc. In the default mode of play, there is no given objective to the game so the player is free to do as he wants as the “mayor.” In some modes of play, the player can choose a scenario in which they have to accomplish an objective like reducing traffic by building a mass transit system.

Pros	Cons	Comparison
<ul style="list-style-type: none">* The player learns about transportation systems and city planning in a real time, reactive environment* The player is given tools to use and alter the map while building around the given topography	<ul style="list-style-type: none">* There is no clear goal to the game* The player is only presented with one static map for the duration of the game* The maps are usually of fictional cities	<ul style="list-style-type: none">* Building a successful circus will be very similar to building a successful city.* Focus is on building on top of a map, not on navigating using a map. The Sim City games require players to read and understand traffic systems, mass transportation and other geographical challenges, but does not explicitly teach them or use them as a focal point.

Grand Theft Auto (series)

In Grand Theft Auto (henceforth “GTA”), the player takes on the role of an outlaw in a large metropolitan area. The player’s goal is to rise through the ranks of organized crime by completing a set of missions across the city. In between and during missions, the player travels around the city making use of primarily stolen cars but also public transportation and aircraft when available. To travel between and during missions, the user must read a map of the city to navigate to destinations.

Pros	Cons	Comparison
<ul style="list-style-type: none">* The player uses a map to navigate the city via various modes of transportation* The gameplay is exciting because it encourages acts that would be illegal and taboo in everyday life	<ul style="list-style-type: none">* The criminal nature of the gameplay makes it impossible to use in a classroom setting* The missions are in no way educational and may even encourage criminal activity	<ul style="list-style-type: none">* The structure of GTA (map-based fantasy world) is similar to what we are trying to achieve.* The main difference is that we are trying to build an educational game while GTA aims to build an entertaining game with no true educational value.* The current versions of GTA are much too sophisticated for our time frame.

3 Game Overview

Game Design

- Introduce students to map reading skills at a reasonable pace
- Use world, local, and representative maps to navigate around the fantasy world and perform in new locations
- Integrate map usage to mimic real-world usage
 - Map screen will require user to choose between various maps
 - Once map is chosen users will be required to plan route and mode of transportation interactively on screen
- Give users feedback through ticket price, audience size, and overall profits
 - Ticket price and audience size increase with number and quality of circus attractions
 - Audience size depends on city and circus running time in that city (spend too long in one city and audiences will get bored)
 - Higher quality circus attractions will cost more

Rational

- Students will learn how to choose the appropriate map for a given task through the choices required on the map screen
- Students will learn how to plan routes from looking at a map through the interactive route planning
- Fantasy world and circus theme will be fun for Middler schoolers
- Maps of fantasy world will teach map skills in a new and different way, without the overtly educational (and perhaps boring) associations of common national and world maps
- Economic aspect will give users immediate feedback and provide obvious goals (increase profits)
- Short feedback loops will keep users engaged and fit into 30 minute class periods
- Savable games will allow students to play one game over multiple sessions, making incentives for success greater

Key Requirements

- Variety of interactive maps (world, city, public transportation) (high)
- Short feedback loops - 30 minutes of gameplay yields significant progress (high)
- Interactive route planning (high)
- Fantasy world (high)
- Compatible with PC (high)
- Feedback through tickets price, audience size, and profits (high)
- Players interpret map symbols (medium)
- Minimal computer skills required for play (medium)
- Games must be savable (medium)
- Coordinate systems for maps (low)
- Zoomable maps (low)
- Mini-games at various locations (low)
- Small map skill tasks called “quests” interspersed throughout the game (low)

4 Architecture

The major components of the game fall into three categories. These are Art, Data, and Code.

4.1 Art

The art in the game will consist of image files, the format of which will likely be determined by storage space and programming language compatibility. Art will be loaded and used by the game to display nearly all game content.

- Backgrounds will be included for all map types and locations.
- Map features (roads, towns, etc.) will not be included in the backgrounds as these need to be interactive.
- Foregrounds will be included for various types of locations and can be overlain on backgrounds (e.g. an airport foreground will be overlaid on a background denoting the town or region, such as mountains, a cityscape, etc.)
- Art will be included for each interactive game elements (such as map features and circus attractions), and will be overlaid on the static background and foreground art.

4.2 Data

No map information, location data, etc. should be encoded into the code elements. Instead, all such information should be encoded into the “game data”. This can be as simple as a text file which code can read, or as complex as an encrypted archive. What is important is that any changes or balancing to map or world layout, game flow, etc. should be made in this game data.

- Game data will include a file or archive which stores player progress, which is the only component of game data which should be modified by the code.
- Game data will include map and world data such as locations of cities, roads, etc.
- The data will also specify the art components needed by each area or location.
- Circus elements specify a related image, name, worth, and other required data.
- Game flow will be stored in data. This includes locations of circus attractions, mini-games, and quests, etc.

4.3 Code

The game concept is relatively novel, and as such several aspects of the game prevent the use of existing code frameworks. The unique requirements for map navigation, etc., as well as the proposed layout of the game, do not exist in currently available toolkits or engines. The code components are the backbone of the game design. The code will be broken up into several categories.

- Resource Loading code will be responsible for loading images and game information from the art and game data files. It will then manage these resources for use by the rest of the game code.
- Display code will request the appropriate data and images, and be responsible for displaying this on screen.
- Input code will handle user input, both accepting and interpreting user input, as well as allowing for direct input sanitization.
- Action handling code will be responsible for knowing and enforcing game “rules” thereby specifying available actions, and handling the results of user actions.

Each game screen will require code elements of each of the above four types. For example, the “map” screen will be rendered by a display code element, which receives images and map data from resource loading elements, and which must have an input element accept input from users clicking on the displayed imagery.

Finally, minigames, such as those triggered when attempting to obtain a circus element, must be coded. These can be largely separate from the rest of the game code - they need only be activated, and when finished, must convey to the game code whether the player was successful.

5 Project Plan and Risk and Feasibility Analysis

5.1 Art

Art can be obtained in many ways. First and foremost, we should look for free art readily available on the web. If this is unavailable, art may potentially be obtained through similarly free photography, which might be adapted to the game ‘style’ through methods such as photoshop (photoshop filters can create primitive ‘cartoony’ effects).

Tasks and Time Required

- Obtaining, creating art: 1-2 weeks, 2-3 people, early in development
- Modifying, fixing, resizing art: 1-2 weeks, 2-3 people, at the end of development

Major Issues

- Availability of art content on the web. Art creation may take longer if significant photoshop or tracing is required.

Special Skills

- Both teams - potentially, photoshop experience, artistic talent, artistic friends

5.2 Game Data

The most important aspect of the game data is a standard data format for all game data types. This must be agreed upon by all team members responsible for resource loading and game data creation. This process is also dependent upon the overall design of the game. Game flow, number of map types, etc. must all be designed before it can be decided how they might be encoded.

Once these specifications are obtained, the format for game data can be agreed on. Input and confirmation will be required from a large percentage of those working on the code as well as game data.

Additionally, the world must be constructed. A world map should be designed by the game data team, and then individual aspects of the world, such as towns, can be designed by smaller groups. When format is determined, and the world designed, creation of actual map data may begin.

First the world must be encoded into the game data. With the world already designed, complete with roads, subways, geographic entities, etc., this should be a simple but time-consuming task. This task may be parallelizable, arising from the fact that this is a very simple rote transcription task - converting a graphical

map into an encoding the game can understand. As such, separate regions may be encoded independently before recombining the entire map, and greater manpower may significantly shorten time required. It is assumed that these human resources are not available however, so the current time estimate reflects this task mostly unparallelized.

In the meantime, another team should be responsible for designing the flow of the game, such as tasks and circus elements. This can only start after the world is designed, since the team members need to know where to place elements, etc. Designing the tasks and element locations should be relatively simple, but again time consuming to provide for sufficient content.

Save data will be the responsibility of the coding team.

Tasks and Time Required

- Formatting specification: 1 week, 2-3 people, very early development
- World design: 2 weeks, 4-6 people, mid development
- World encoding: 2 weeks, 3 people, late development
- Game flow and Element encoding: 2 weeks, 3 people, late development

Major Issues

- Conflicts, or bad initial formatting specification, may require a complete rewrite of all game data, which would likely sink the project or require significantly reduced game content. The specification of the format of game data should ensure that it is clear to use and able to express all features needed. Optimally, this specification will be extensible and flexible in case of changing or forgotten requirements.

Special Skills

- Latitude/Longitude knowledge
- Map designers must be able to balance expansive scope, many map scales, so that the map is interesting for students at all scales (sufficient small-scale content in towns, etc).

5.3 Code

Code can be done in any of several different languages, although an object-oriented language such as Java or C++ may be most desirable for this application.

Architecture should be done by the entire coding team. After that, a ‘moderator’ component, which should marshal different sub-components as necessary should be constructed.

The code team (8-10 people) should split into four groups. The first team will be in charge of the map components. They will likely need to implement loading and displaying maps and then navigation on various map types.

The second team will be in charge of “location navigation” - displaying and acting within a location, such as an airport. They will implement loading and displaying the location and interactive elements. Afterwards they will focus on input, ensuring that users can interact with the characters and attractions in the town.

The third team will be in charge of several tasks. The first will be action validation, where the actions currently available to the user, are managed and actions the user takes are verified and acted upon. Then this team will be responsible for the “circus” page - where the circus is displayed as an onscreen collection of all attractions currently owned. Finally, this team will implement various animations (or the display of appropriate non-animated images). For example, when the user attempts to make an illegal action, the proper failure image should be displayed (such as attempting to drive into the ocean), and all failure and success states should have at least an appropriate message, if not image, attached.

The final team will be responsible for mini-games. These mini-games should be exceedingly simple, and separated into fun, simple, irrelevant mini-games (dodge falling objects) and educational mini-games (given a map and a key, click on the airport). The simplicity of these games is required so that the team can spend as little as a week on each game, giving the opportunity for up to 4 separate mini-games to be implemented. As the mini-games are a low priority requirement, this group will also help with any overflow tasks from the other code groups.

Tasks and Time Required

- Architecture: 1 week, 8 people, beginning of development
- Moderator: 1 week, 2-3 people, after architecture
- Team one - Maps: 4 weeks, 3 people
 - Loading and displaying: 1 week
 - Navigation: 3 weeks

- Team two - Location display/interaction: 3-4 weeks, 2 people
 - Loading and displaying: 1 week
 - Input: 2-3 weeks
- Action handling, etc: 4 weeks, 3 people
 - Validation: 2 weeks
 - Circus page: 1 week
 - Animations and messages: 1 week
- Mini-games: 4 weeks, 2 people

Major Issues

- Time - deadlines may get very tight, depending on how smoothly programming goes, given the very small sub-teams. Therefore, it will be especially important for “nice, not absolutely necessary” components to be finished last, and for a simple working (limited functionality) version of all components to be each team’s primary goal. This will allow slow components or delays to have a lesser effect on final deadlines. The group working on mini-games will be prepared to help where ever extra resources are needed.
- Interactions between software components may not be able to be tested until late in the development process. Bugs, therefore, may cause large setbacks. Again, emphasis should be placed on “working, testable, limited functionality” versions being created as soon as possible.

Special Skills

- Some members from each team - I/O
- Some members from each team - UI design
- Map navigation team - user input sanitization, potentially pathfinding
- All teams - graphics

5.4 A Potential Five Week Schedule

Week 1:

- Code team: Architecture
- Data team: Data formatting specification
- Art team: begin finding/creating art

Week 2:

- Code team: split up into groups, begin completing all 4 subtasks
- Code team: 2-3 people should take time here to write the moderator component
- Data team: World Design
- Art team: finding/creating art

Week 3:

- Code team: continue on subtasks. Emphasis on displaying imagery and UI so that images can be tested
- Data team: World Design
- Art team: art integration, modification, fixing

Week 4:

- Code team: continue on subtasks
- Data team: encoding map data, designing/encoding game flow data
- Art team: art integration, modification, fixing

Week 5:

- Code team: finish subtasks
- Data team: finish encoding map/game data
- Art team: Assist other teams in finishing tasks

Will it work?

This is a 5-week schedule, allowing for time to do user testing, etc., as well as allowing for deadline slippage. Obviously, this is a very tight schedule. Certain components may need to be dropped, and the schedule may have to be modified to more quickly address the more critical components to allow for changing of scope later on. Additionally, there is room in the schedule to extend certain elements, especially those not critical to the release of the final product. Number of minigames could be reduced, and game data could be simplified, reducing the amount of game content.