Alexander Tripp

CIS 588 / Winter 2015

2015, April 24

Design Document

Once A Pawn

Contents

Design History

2015-02-03 – Initial version

2015-03-25 – Beta Release

2015-04-24 – Gold Release

Section I – Game Overview

Game Concept: *Once A Pawn* is a turn-based strategy game with chess-like gameplay. The player's army is represented by chess pieces that move and capture by rules similar to chess, but players must also build cities and collect resources, which can be used to raise a larger army.

Feature Set: "3X" gameplay, resource management, tactical strategy, empire building. Unlike a traditional 4X game, there is no exploration – the entire board is always visible.

Genre: Turn-based Strategy

Target Audience: Strategy gamers

Game Flow Summary: The player begins with only a single city and a few pieces. Initially, they must focus on collecting enough resources to build more cities and a larger army. As their position solidifies, the focus shifts to attacking the opponent and capturing their pieces and cities.

Look and Feel: The game takes place on a large chessboard with added features – mountains, rivers, and forests. The visual style is highly stylized, with deliberately low-poly models and caricatured shapes for a cartoon effect. All pieces are monochromatic; the board has light and dark gray tiles while the playable pieces are blue (for the human player) and red (for the AI opponent).

Project Scope:

- Each match is entirely self-contained on a 32×32 chessboard. Matches are not linked to each other, so there is no progression from one match to the next.
- There are five terrain types: open ground, forest, water, mountains, and bridges.
- The player can build three different buildings (cities, mines, and lumber camps) and six pieces (pawn, bishop, rook, knight, queen, and king). Each piece has moves similar to its abilities in chess, slightly modified to match the game environment.

Production Team: Tripp Industries

Section II – Gameplay and Mechanics

Gameplay

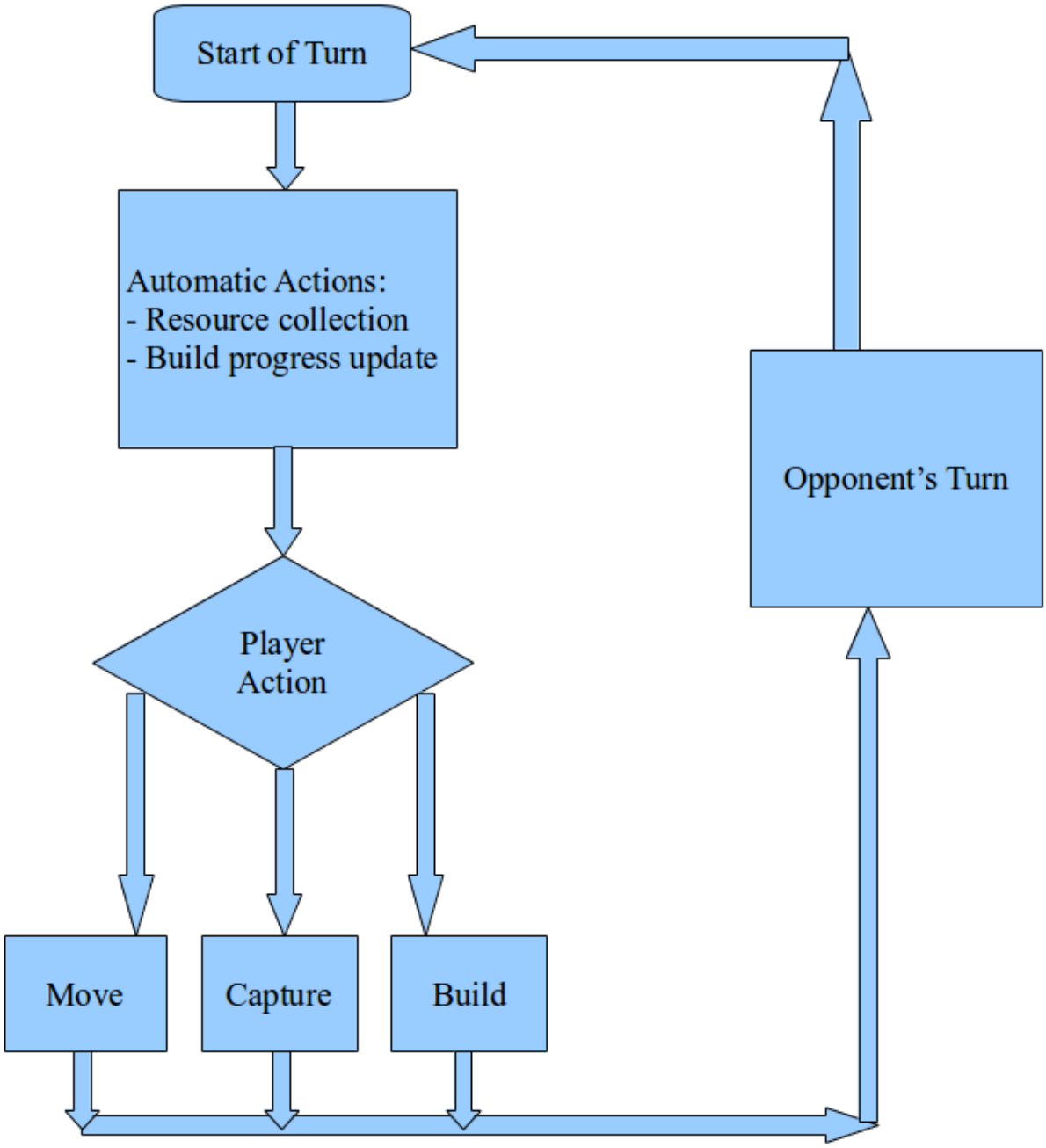
Objectives: The player begins with one King piece, and the objective is to capture the opponent's King. Minor objectives along the way include:

- Purchasing structures to collect resources
- Creating new pieces to expand the army – Kings may not be created.
- Capturing opposing pieces (which removes them from the game) and buildings (which turns them to the capturing player's control).

Mechanics

Classifications: "Pieces" refers to all player-controlled objects on the board. These are divided into "units" (which can move) and "buildings" (which cannot). The units in *Once A Pawn* are the Pawn, Bishop, Rook, Queen, and King. The buildings are City, Lumber Camp, and Mine.

Turn Sequence: During each turn, each player may take one action. This can be moving a unit, capturing an opposing piece, creating a new building, or creating a new unit.



Movement: All pieces have the same movement properties for capturing as they do for moving. They may only move across open ground – water, mountains, and forests are impassable. Pieces cannot move through other pieces – they may capture opposing pieces but cannot move past friendly ones. They may occupy the same space as a building or city, however.

- Pawns may move up to two spaces per turn, either orthogonally or diagonally. They can also build buildings, thought they must remain in place until the building is completed.
- Rooks may move orthogonally an unlimited number of spaces (until they reach an obstacle).
- Bishops may move diagonally an unlimited number of spaces (until they reach an obstacle), or orthogonally into an adjacent square only. They may not do both in the same turn.
- Queens may move orthogonally or diagonally an unlimited number of spaces (until they reach an obstacle).
- Kings may move orthogonally or diagonally one square per turn.

Buildings: Cities may be placed anywhere on the board (on open ground only).

In the case of mines and lumber camps, the resource production of the building depends on the number of mountain / forest squares respectively within a radius of 3 from the building. A resource building may not be built close enough to another of the same type that their radii would overlap (i.e. a lumber camp cannot be built within 6 tiles of another lumber camp).

Economy:

- Wood and Metal are collected by Lumber Camps and Mines, respectively. Both resources are used to purchase new pieces and buildings. Lumber camps and mines automatically collect resources each turn, as described above.
- All pieces and buildings take resources to create. Weaker pieces such as the pawn will are cheap to create, whereas the queen is very expensive. Each build action takes one turn to complete.

Piece Actions: Units may move and capture, and in the case of the pawn, build.

Cities can create new units if the player has sufficient resources.

Lumber Camps and Mines cannot take any actions, but collect resources each turn automatically.

Capturing: Capturing works as in chess – moving a piece onto the same square as an opponent's piece removes it from the game. Cities and buildings may be captured as well, but they are not removed. Instead, they immediately change allegiance to the player that captured them.

Section III – Story, Setting, and Character

Narrative

An evil wizard has turned everyone in the kingdom into chess pieces! You must build up your forces to defeat his chess army and restore the land.

Game World

The world in which the game takes place would usually be a fairly ordinary fantasy world, but within the game it appears as an elaborate chess set. The board has black and white squares like a regular chessboard but also features water, mountains, forests, bridges, cities, and other buildings.

Characters

The only specific character (the evil wizard) does not directly appear in the game, as he is represented by his army of chess pieces. All other characters have been transformed into chess pieces, and do not have personalities within the scope of the game.

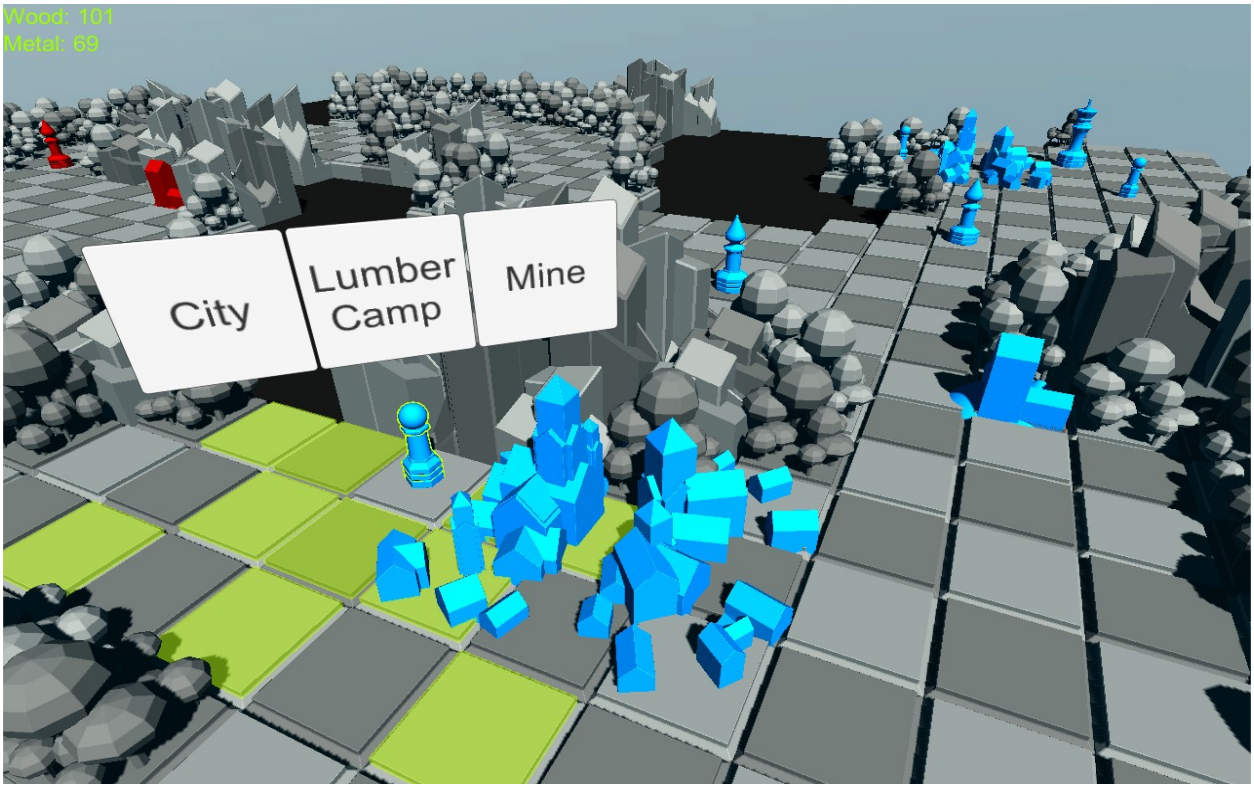
Section IV – Match Setup

The board is larger than a regular chessboard, with 32x32 squares. It has half-turn rotational symmetry, so the players start from an equal footing (the only difference is the order in which they start).

The blue player (controlled by a human) makes the first move, like the white player in chess.

Section V – Interface

Visual System



HUD: The HUD displays the player's current resource stockpiles. Unlike in many strategy games, there is no minimap. Instead, the player can zoom out indefinitely to view the entire board at once.

When a piece that can create other pieces (such as the City) is selected, a HUD will appear above the piece, displaying all build options and the prices of each item.

If the player attempts to create a piece they cannot afford, or in an invalid location, a message will appear to alert them that this is not allowed.

Camera: The game uses an angled top-down viewpoint. The player may move the camera across the map, zoom in and out, and rotate the camera both vertically and horizontally.

Control System

The game is operated using a standard mouse. Left click selects a piece or building and issues a command, Middle click and drag rotates the camera, and Right click and drag pans the camera. The scroll wheel zooms in and out.

Audio

Music: The game features ambient background music. The music is "Saint Paul's Cathedral" composed by Jamie Hargrove and arranged & recorded by Aaron Elkiss and Monique Rio. Used with permission.

Section VI – Artificial Intelligence

Opponent AI

As *Once A Pawn* is a single-player game, the opponent is controlled by AI. Each turn, the AI takes stock of the current situation and chooses an appropriate move.

The AI decision-making process works as follows:

1. The engine refreshes the available moves for all pieces (both players) on the board. During this process, it also creates maps of which tiles each player can attack.
2. The AI controller analyzes each move individually, and assigns each a priority based on a number of factors: For example, moves such as capturing an opponent's piece or moving out of the line of fire are prioritized above normal moves, while moving into the line of fire is less favored.
3. Each move is given a very small random adjustment. The purpose of this is to prevent the AI from acting the same way every game – essentially, when presented with multiple equivalent moves, it will choose randomly from among them rather than always taking the first one encountered.
4. The controller adds all moves to a sorted list, with the priority as the key. Lower values indicate a higher priority.
5. After all moves have been added, the controller chooses the first move in the list and executes it.

Support AI

Collision Detection: Collision detection is handled directly by the Unity engine. It is used for detecting mouseover and click events on pieces and tiles. Other than this, it is not used for anything.

Section VII – Technical

Target Hardware & Software: The game is designed for a typical home desktop or laptop computer running Linux or Windows. It requires a mouse and monitor, and recommends a keyboard and speakers.

Development Hardware: The game is developed on PCs running Windows 7. All development machines have a keyboard, mouse, monitor, and speakers; some have two monitors.

Game Engine: The game uses the Unity 4 engine (version 4.6.2f1). All coding is done in C# through Unity's built-in scripting ability.

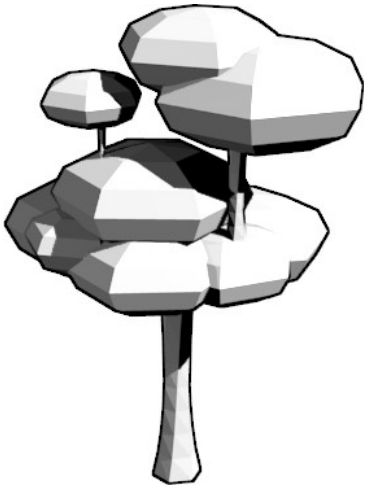
Development Software: In addition to Unity, several supplemental pieces of software are needed during development.

- Models are created using Blender 3D.
- Maps are created using GIMP.

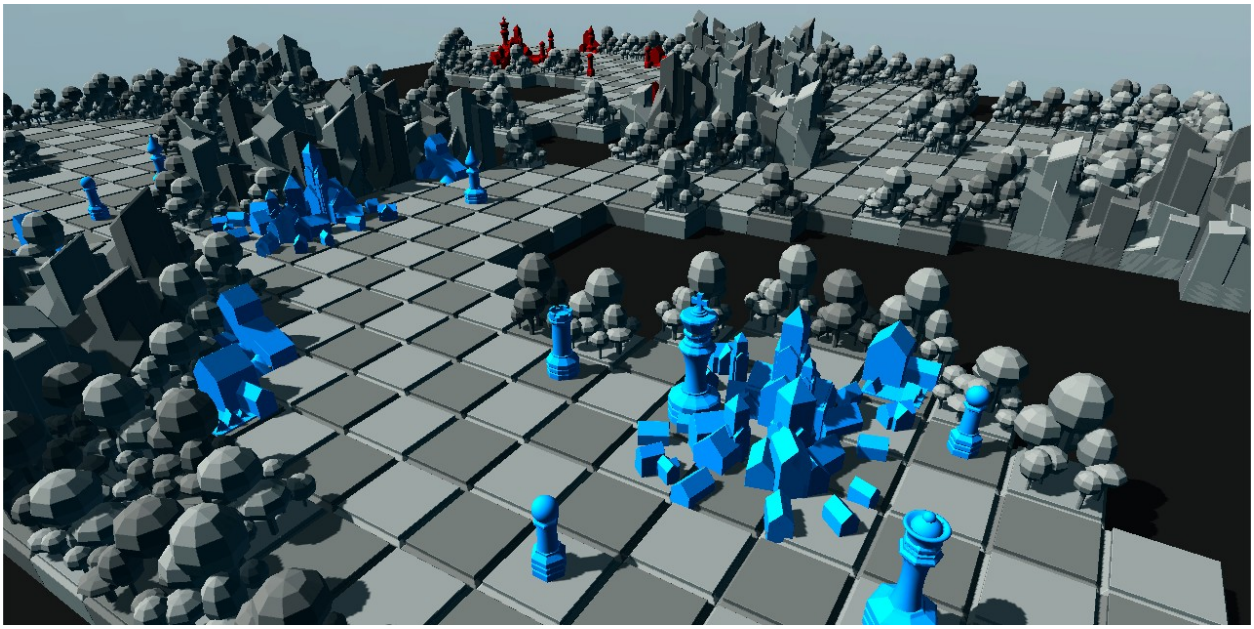
All supplemental programs are either free to use, or are already owned by the production team.

Section VIII – Game Art

Concept Art



In-game Screenshots



Section IX – Management

Schedule

2015-02-18 – Alpha release complete

2015-03-25 – Beta release complete

2015-04-15 – Marketing Presentation

2015-04-24 – Project Fair, release candidate complete

Budget

The total financial budget for Once A Pawn is $0.00. As Tripp Industries consists of a single individual, the current time budget for the game is one hour of work per day (on average). This is consistent with the amount of work that has produced both the Alpha and Beta releases so far.

Risk Analysis

Risk	Probability	Severity
Unsuitable game engine	Occurred	Extreme
Security compromised on production machine	Occurred	High
Conflict with paid employment	High	High
Conflict with other game design projects	High	Low
Illness or injury preventing work	Low	High
Loss of Data	Low	Extreme

Many of these risks are addressed by following a set but flexible schedule. This allows development time to fit in around Tripp Industries' somewhat unpredictable income source of substitute teaching.

Unfortunately, due to limitations in the Unity Editor (and the fact that Tripp Industries does not own a Mac), the game must be developed in Windows, which is known to have security flaws. During development, the production machine fell to a malware attack, requiring a complete reinstallation of the operating system. Fortunately, no files were damaged and development continued as normal after the incident was resolved.

Section X – Testing

Strategy

Initial testing was done through a "test immediately" model – as soon as a new feature is implemented, it is immediately tested. All other main features are tested as well to ensure that they are still working properly after the addition of new code.

In the late stages of development, more rigorous testing of the game has taken place. In this phase, the testing is less focused on single features and more on the overall functionality of the game.

Test Results

Numerous tests have been run in-game to check the functionality of various features.

Test	Outcome	Status
Move pieces around a lot to test proper functioning of moves.	Success	Working
Create new units to test resource checking	Success	Working
Create new buildings to test resource and location checking	Success	Working
Create new Mines and Lumber Camps to test resource production increases	Success	Working
Threaten opposing pieces (but not capture) to test AI	Partial Success – Units will move out of the way most of the time	**Known Bug**
Place units in danger to test AI attacking	Partial Success – The unit will be captured unless a more valuable unit is being threatened.	**Known Bug**
Create new pieces to test board entrance	Success, with minor graphical bug	**Known Bug**
Capture enemy pieces to test piece deletion	Success	Working
Capture enemy buildings to test switching a piece to a different player	Success	Working
Wait for AI to build and create new pieces	Partial Success – AI can create new buildings, but does not necessarily place them in sensible locations; AI is not capable of planning ahead.	**Known Bug**
Capture enemy King to test game over condition	Success	Working

Known Bugs

- When threatening opposing pieces, they may not successfully move out of the line of fire. This is because the tiles behind the threatened piece are not accessible to the attacker, so they are marked as safe. Detecting them as unsafe requires looking ahead by one move, which the AI engine currently does not support.
- The AI does not always choose the best option when presented with multiple captures or threats. For example, if a Queen is threatening the King but is under fire from another piece, the AI may choose to move the King out of danger rather than capture the Queen.
- When new units or buildings are created, they may “flicker” in their final position before rising out of the board properly. This is caused by the fact that the rendering engine and game engine run in different threads, so the piece may be rendered before it can be properly positioned.
- When placing new Lumber Camps or Mines, the AI may not place them near forests or mountains.