Not Catan SRS 3
(Revision of
Not Catan SRS 2)

Not Catan Software Requirements Specifications

Approved 28 November 2017

CPSC 4175 Group Delta:

David Grant
Joseph Kim
Bradley Laney

Contents

Section 1
Introduction

1.1 Purpose
This Software Requirements Specification (SRS) document outlines and provides details to the developers and instructor of the CPSC 4175 (Software Engineering) course at Columbus State University. Specifically of the Not Catan project being developed. This project and document is an exercise of adhering to the principles and processes being taught in the course as a proof of concept.

1.2 Scope
The product that this SRS belongs to is called Not Catan. The software product will be implemented so that it functions as closely to the popular board game called Settlers of Catan. This includes features such as a structured turn, multiple players, and a multitude of interactions with the different game pieces adding to the experience of the users of the software product.

1.3 Definitions

1.3.1 BUILD (BUILDING)
You may build on your turn after you have rolled for Resource production and finished trading. To build, you must deduct the specified combinations of Resources (see the Building Costs Key).

Your Resources will be automatically deducted for the item you try to build. You can build as many items and buy as many development cards as you desire—as long as you have enough Resources to “pay” for them and they are still available in the supply.

Each player has a supply of 15 roads, 5 settlements, and 4 cities. If you build a city, return the settlement to your supply. Roads and cities, however, remain on the board until the end of the game once they are built. Your turn is over after “building,” and the next player to your continues the game.

1.3.2 BUILDING COSTS KEY
The Building Costs Key is a graphic displayed on the GUI to show what can be built and which Resources are required. When you pay the building costs, your Resources will be deducted automatically. You can build settlements and roads, upgrade settlements to cities, and buy Development Cards.

1.3.3 CITIES
You cannot build a city. You can only upgrade an existing settlement to a city. When upgrading a settlement to a city the Resources are deducted, the settlement is returned to your supply, and the settlement is replaced with a city on the same intersection.

Each city is worth 2 victory points. You receive double Resource production from the adjacent terrain hexes whenever those numbers are rolled.

When you build a city, the upgraded settlement piece becomes available again. You can build more settlements later.

1.3.4 COAST
When a terrain hex borders on the sea (i.e., the blue framing the Catan hexes), it is called a “coast.” You can build a road along a coast. You can build settlements and upgrade settlements to cities on intersections that border on the sea. Since a site on the coast borders only 1 or 2 terrain hexes, however, coastal settlements generate smaller Resource yields. Still, coastal sites often lie on harbors, which allow you to use maritime trade to trade Resources at more favorable rates.

1.3.5 COMBINED TRADE/BUILD PHASE
The separation of the trade and build phases was introduced to make the sequence easier to learn for beginners.

After rolling for Resource production, you can trade and build in any order. Naturally you can trade, build, trade again and build again. You can even use a harbor on the same turn you build a settlement there. This will also speed up the game.

1.3.6 DESERT
The desert is the only terrain hex that does not produce Resources. The robber is native to the desert, and they starts the game there. A settlement or a city built adjacent to the desert yields fewer Resources than than those built next to one of the other terrain types.

1.3.7 DEVELOPMENT CARDS
There are 3 different kinds of Development Cards:

1.3.7.1 Knights,

1.3.7.2 Progress, and

1.3.7.3 Victory Points.

When you buy a Development Card your Development Cards remain hidden until you play them. This keeps the other players in the dark. You cannot trade or give away Development Cards. You may only play 1 Development Card during your turn — either 1 Knight Card or 1 Progress Card. You can play the card at any time, even before you roll the dice. You may not, however, play a card that you bought during the same turn.

Exception: If you buy a card and it is a Victory Point Card that brings you to 10 points, you may immediately reveal this card (and all other VP cards) and win the game.

You only reveal Victory Point Cards when the game is over—once you or an opponent reaches 10+ victory points and declares victory.

1.3.8 DISTANCE RULE
You may only build a settlement on an open intersection and only if none of the 3 adjacent intersections contains a settlement or city.

1.3.9 DOMESTIC TRADE
On your turn, you may trade Resources with the other players (after rolling for Resource production). You and the other players negotiate the terms of your trades—such as which Resources will be exchanged. You may trade as many times as you can, using single or multiple cards, but you may not give away cards (i.e., “trade” 0 cards for 1 or more cards).

Important: While it is your turn you must be a part of all trades, the other players may not trade amongst themselves.

Example: It is Pete’s turn. He needs one brick to build a road. He has 2 lumber and 3 ore. Pete asks aloud, “Who will give me 1 brick for 1 ore?” Beth answers, “If you give me 3 ore, I’ll give you a brick.” Cooper interjects, “I’ll give you 1 brick if you give me 1 lumber and 1 ore.” Pete accepts Cooper's offer and trades a lumber and an ore for a brick. Note Beth may not trade with Cooper, since it is Pete’s turn

1.3.10 ENDING THE GAME
If you have—or reach—10 victory points on your turn, the game ends immediately and you win! You can only win during your turn. If somehow you find you have 10 victory points during another player’s turn, you must wait until your next turn to claim victory.

Example: Jhinuk has 2 settlements (2 points), the Longest Road special (2 points), 2 cities (4 points), and 2 Victory Point Cards (2 points). She reveals her 2 Victory Point Cards, giving her the 10 points needed to win. She surprises her opponents and claims victory!

1.3.11 GAME PLAY
Here is a summary of the the game sequence, plus some more specific entries where you can find details:

1.3.11.1.1 Lay out the game board: Set-up, Variable

1.3.11.1.2 Initial set-up: Set-up Phase

1.3.11.1.3 Play.

The starting player begins the game. The other players follow in clockwise order. On your turn, you complete these 3 phases in order:

1.3.11.2.1 Roll for Resource Production (the roll applies to all players).

1.3.11.2.2 Trade.

1.3.11.2.3 Build.

You may play 1 Development Card any time during your turn.

Then the next player's turn begins.

1.3.12 HARBORS
Harbors allow you to trade Resources more favorably. In order to control a harbor, you must build a settlement on a coastal intersection which borders the harbor. See also “Maritime Trade”.

1.3.13 INTERSECTIONS
Intersections are the points where 3 hexes meet (Aka vertices). You may only build settlements on intersections.

The influence (for Resource yields) of settlements and cities extends into the 3 adjacent terrain hexes that form the intersection.

1.3.14 KNIGHT CARDS
See “Soldier Cards.”

1.3.15 LARGEST ARMY
If you are the first player to play 3 Knight Cards, you receive this Special Token, which is worth 2 victory points. The "Largest Army" token will be placed by your player icon. If another player plays more Knight Cards than you have, he immediately takes the Special Token. The 2 victory points likewise count for the new owner.

1.3.16 LONGEST ROAD
If you are the first player to build a continuous road of at least 5 individual road pieces, you take this Special Token and it is placed by your player icon. This token is worth 2 victory points.

Note: If your road network branches, you may only count the single longest branch for purposes of the longest road.

If you hold the “Longest Road” token and another player builds a longer road, he immediately acquires your “Longest Road” token. He also acquires the 2 bonus victory points.

(Since you also lose the 2 victory points, it is a 4 point swing!)

You can break an opponent’s road by building a settlement on an open intersection along their road!

Set the “Longest Road” token aside if—after a longest road is broken—several players tie for the new longest road or no one has a 5+ segment road. The “Longest Road” token comes into play again when only 1 player has the longest road (of at least 5 road pieces).

1.3.17 MARITIME TRADE
On your turn, you can trade Resources using maritime trade during the trade phase even without involving another player.

The most basic (and unfavorable) exchange rate is 4:1. You may trade 4 identical Resources to the supply in exchange for the (1) Resource of your choice. You do not need a harbor (settlement at a harbor location) to trade at 4:1, so when nobody wants to trade…

Example: Benny returns 4 ore to the supply and takes 1 lumber in exchange. Normally, he should first try a more favorable trade with the other players (domestic trade).

If you have built a settlement or city at a harbor location, you can trade more effectively. There are 2 different kinds of harbor locations:

1.3.17.1 Generic Harbor (3:1): Here you may exchange 3 identical Resources for any one other Resource during your trade phase.

Example: Olivia, the red player, has built a settlement at a generic harbor. She can, for instance, exchange 3 lumber for 1 wool.

1.3.17.2 Special Harbor (2:1): There is but 1 special harbor for each type of Resource (with the same symbol). So, it is important to build on the type of special harbor you can use fairly frequently. (Look at your Resource production.)

The exchange rate of 2:1 only applies to the Resource shown on the harbor location. A special harbor does not permit you to trade any other Resource type at a more favorable rate (not even 3:1)!

Example: Nick, the orange player, built a settlement at the ore special harbor. Nick may exchange 2 ore for any 1 other Resource. He can also trade 4 ore for any 2 other Resources. If he traded 4 wool instead of 4 ore, he would get only 1 Resource in return.

1.3.18 NUMBER TOKENS
The 18 number tokens are marked with the numerals "2" through "12." There is only one "2" and one "12." There is no "7."

The more often a number is rolled, the more often each associated hex produces Resources.

The small letters on the top of the number markers are important during the setup phase (see Set-up Phase).

1.3.19 PATHS
Paths are defined as the edges where two hexes meet. Paths run along the border of two terrain hexes or between a land hex and the sea. Only one road can be built on any path. Each path leads to an intersection where 3 hexes meet.

1.3.20 PROGRESS CARDS
Progress Cards are a type of Development Card. There are 2 each of 3 varieties:

1.3.20.1 Road Building: If you play this card, you may immediately place 2 free roads on the board (according to normal building rules).

1.3.20.2 Year of Plenty: If you play this card you may immediately take any 2 Resources from the supply. You may use these Resources to build in the same turn.

1.3.20.3 Monopoly: If you play this card, you must name 1 type of Resource. All the other players must give you all of the Resources of this type that they have in their hands.

If an opponent does not have a Resource of the specified type, they do not have to give you anything. You may play only 1 Development Card during your turn.

1.3.21 RESOURCES
There are 5 different types of Resources:

1.3.21.1 grain (from fields),

1.3.21.2 brick (from hills),

1.3.21.3 ore (from mountains),

1.3.21.4 lumber (from forest),

1.3.21.5 and wool (from pasture).

You receive these Resources as income from the Resource production of these hexes. Resource production is determined by the dice roll at the beginning of each turn. You receive your income for each terrain hex adjacent to your settlements or cities every time the production number on the hex is rolled (exception: see Robber).

1.3.22 RESOURCE PRODUCTION
On your turn, you must roll the dice for the turn's Resource Production. The number rolled determines which hexes produce Resources. Each number appears twice—except for "2" and "12," which only appear once.

All players who have settlements or cities on the hexes indicated by the roll receive the yields (Resources) of those hexes. Each settlement produces 1 Resource; each city 2 Resource.

Example: Loren, the blue player, rolls a "4". Her settlement "A" borders a pasture marked by the number "4", so she is given a wool Resource. If settlement "A" had been a city, she would have* *received 2 wool Resources. Bridget owns the red settlement “B” that borders on 2 hexes with the number “4”: mountains and pasture. Bridget is given 1 ore and 1 wool resources from the supply.

It is possible that during the game there will not be enough Resources in the bank to supply all of the yields. If there are not enough Resources to give every player all the production they earn, then no player receives any of that Resource that turn. Production of other types of Resources is not affected.

1.3.23 RESOURCE TRADE
In the second phase of your turn, you may trade with the other players. The other players may not trade amongst themselves, only with the player whose turn it is. There are 2 different kinds of trade:

1.3.23.1 domestic trade and

1.3.23.2 maritime trade.

1.3.24 ROADS
The roads connect your settlements and cities. You build roads on paths. You cannot build new settlements without also building roads. Roads provide victory points only if you hold the Longest Road Special Card. Only 1 road may be built on each path. You can build roads along the coast.

Example: Liam, the red player, would like to build a road. He may build (place) his road on any of the paths marked with arrows. Each of these paths connects to either Liam’s road or his settlement.

1.3.25 ROBBER
The robber begins the game in the desert. It is moved only by rolling a “7” or playing a Knight Card.

If the robber is moved to any other terrain hex, he prevents that hex from producing Resources.

Players with settlements and/or cities adjacent to the target terrain hex receive no Resources from this hex as long as the robber is in the hex.

1.3.26 ROLLING A “7” AND ACTIVATING THE ROBBER
If you roll a “7” for Resource production, none of the players receive Resources. Instead:

1.3.26.1 Any player with more than 7 Resources (i.e., 8 or more), must choose and deduct half of them to the supply. If you hold an odd number of Resources, round down (e.g., if you have 9 Resources, you choose and deduct 4).

Example: Alex rolls a “7”. He has only 6 Resources. Larry has 8 Resources and Will has 11. Larry must choose and deduct 4 Resources and Will 5 (rounding down).

1.3.26.2 Then you (the player who rolled the “7”) must move the robber to the number token of any other terrain hex (or to the desert hex). This blocks the Resource production of this hex, until the robber moves to another number token.

1.3.27.3 After deduction occurs, you also steal 1 Resource at random from a player who has a settlement or city adjacent to this new hex. If there are 2 or more players with buildings there, you may choose from which one to steal. The robber must be moved. You may not choose to leave the robber in the same hex. After moving the robber, your turn continues with the trade phase.

See also Knights and Soldiers.

1.3.28 SETTLEMENTS
A settlement is worth 1 victory point. Settlements are built on intersections (where 3 hexes meet). You share in all of the Resource production of each terrain hex adjacent to your settlements. You must meet 2 conditions when building a settlement:

1.3.28.1 Your settlement must always connect to 1 or more of your own roads.

1.3.28.2 You must observe the Distance Rule.

Note: If you have built all 5 of your settlements, you must upgrade 1 of your settlements to a city before you can build another settlement. You will then have the settlement in your supply, so you can build another settlement.

1.3.29 SET-UP PHASE

1.3.29.1 Begin the “set-up phase” after you build the game map. Everyone chooses a color and is given the corresponding game pieces:

1.3.29.1.1 5 settlements;

1.3.29.1.2 4 cities;

1.3.29.1.3 15 roads;

1.3.29.2 Shuffle the Development Cards and place them face down beside the Resource supply.

1.3.29.3 The 2 Special Tokens are not used until a player qualifies for them.

1.3.29.4 Robber is placed in the desert.

1.3.29.5 The set-up phase has 2 rounds. Each player builds 1 road and 1 settlement per round.

1.3.29.6 Round One

Dice is rolled for each player. The player who has the highest roll is the starting player and begins.

The starting player places a settlement on an open intersection of his choice. He places a road adjacent to this settlement.

The other players then follow clockwise. Everyone places 1 settlement and 1 adjoining road.

Important: When placing all settlements, the Distance Rule always applies!

1.3.29.7 Round Two

Once all players have built their first settlements, the player who went last in the first round begins round two: he builds his second settlement and its adjacent road.

Note: After he builds, the other players follow counterclockwise, so the starting player in Round One places his second settlement last.

The second settlement can be placed on any open intersection, as long as the Distance Rule is observed. It doesn’t have to connect to the first settlement. The second road must attach to the second settlement (pointing in any of the 3 directions).

Each player receives his starting Resources immediately after building his second settlement; for each terrain hex adjacent to this second settlement, he is given a corresponding Resource from the supply.

Note: The starting player (the last to place his second settlement) begins the game. You can find helpful tips about the set-up phase under “Tactics.”

1.3.30 SOLDIER (KNIGHT) CARDS
When you play a “Knight” Development Card during your turn, you must immediately move the robber.

The Knight Card is placed face up in front of you.

You must move the robber away from his current spot and onto the number token of any other terrain hex.

You then steal 1 Resource from a player who has a settlement or a city adjacent to the robber. If there are 2 or more such players, you may choose your victim.

The player you elect to rob has 1 Resource removed at random. If he has no Resources, you get nothing! (However, you can always ask players about the number of Resources they hold.)

If you are the first player to have 3 Knight Cards face up in front of you, you take the “Largest Army” Special Token. This Special Token is worth 2 victory points.

If another player has more Knight Cards than you, he takes the Special Token, and the 2 victory points that go with it.

1.3.31 TACTICS
Since you can play The Not Catan with a variable map, the tactical considerations of each game are different. There are, nevertheless, some common points you should consider:

> **1.3.31.1** Brick and lumber are the most important Resources at the beginning of the game. You need both to build roads and settlements. You should try to place at least 1 of your first settlements on a good forest or hills hex.
>
> **1.3.31.2** Do not underestimate the value of harbors. For instance, a player with settlements or cities on productive fields should try to build a settlement on the “grain” harbor.
>
> **1.3.31.3** Leave enough room to expand when placing your first 2 settlements. Look at your opponents’ sites and roads before making a placement. Beware of getting surrounded! If you plan to build toward a harbor, the middle of the island may be a tricky place for a starting settlement, for it can easily be cut off from the coast.
>
> **1.3.31.4** The more you trade, the better your chances of victory. Even if it is not your turn, you should offer trades to the current player!

1.3.32 TRADE
After you roll for resource production, you may trade with other players (domestic trade) or with the supply (maritime trade).

If you decide not to trade during your turn, no one can trade.

You may trade with another player between your turns, but only if it is their turn and they elect to trade with you. You cannot trade with the bank during another player’s turn. You may not give away Resources.

You may trade as long as you have Resources.

You may not trade Development Cards. You may not trade like resources (e.g., 2 wool for 1 wool).

1.3.33 VICTORY POINT CARDS
Victory Point Cards are Development Cards, so they can be “bought.” These Development Cards represent important cultural achievements, represented by certain buildings.

Each Victory Point Card is worth 1 victory point.

You only reveal your Victory Point Cards when you or someone else wins the game! Keep Victory Point Cards hidden until you have 10 points during your turn and you can declare victory. (You should also reveal them if someone else wins.)

1.3.34 VICTORY POINTS
The first player to reach (be at) 10 victory points on his turn wins the game.

Players acquire victory points (VPs) for the following:

> **1.3.34.1** 1 Settlement = 1 VP
>
> **1.3.35.2** 1 City = 2 VPs
>
> **1.3.35.3** Longest Road Special Token = 2 VPs
>
> **1.3.35.4** Largest Army Special Token = 2 VPs
>
> **1.3.35.5** Victory Point Card = 1 VP

Since each player begins with 2 settlements, each player begins the game with 2 victory points. Therefore, you only need 8 more victory points to win the game!

1.4 References

This document was created and detailed in conjunction with the following publications.

The Settlers of Catan Game Rules & Almanac[1]

Turn Overview: “The Settlers of Catan”[2]

1

1.5 Overview

The Not Catan software requirements specification outlines the details of the functionality of the project being worked on by Bradley Laney, Joseph Kim, and David Grant. These are students at the institution of Columbus State University that are developing this product as a proof of concept for following the software engineering principles that are being taught in CPSC 4175 (Software Engineering). The project entails the use of networking principles, an object-oriented implementation of a turn structured game called Settlers of Catan, and a functional graphical user interface that represents the real-world interaction for the game and its players. The process of development attempted to be used is the spiral-iterative development process.

[1] The Settlers of Catan Game Rules & Almanac can be found online http://www.catan.com/en/download/?SoC_rv_Rules_091907.pdf

[2] Turn Overview: “The Settlers of Catan” can be found online https://github.com/stoksc/Settlers-of-definitely-not-Catan/blob/master/Design/settlers_of_catan_turn_overview.pdf

Section 2
Overall Description

2.1 Product Perspective

2.1.1 User Interfaces
A user of Not Catan will be prompted to select profile or register upon opening the application in a window with two button graphics, Select Profile and Register New Profile.

If the Register New Profile button is pressed it will ask for a name to register a new profile and afterwards should continue to the Select Profile screen which should now display all previous profiles and the new profile.

On the Select Profile screen there should be buttons for each profile that has been previously registered and stored on the device Not Catan is being run.

Upon selecting a profile from the Select Profile screen, the window should proceed to a screen that has two buttons on it, Connect to a Game and Host a Game.

If the Host a Game button is pressed, the window should change out the Connect to a Game button that displays the IP address that others can connect to if the Connect to a Game button was selected instead.

If the Connect to a Game button is pressed, the window should change out the Host a Game button with a text box to input an IP address of an already known host that has already selected the Host a Game button.

After a connection is made between players and the host, all Not Catan's user windows will screen change to the screen that appropriately represents the start up game state that all users can interact with in order to experience the functionality of the gameplay of the software product.

2.1.2 Hardware Interfaces
The Not Catan software interacts with the keyboard and mouse user interface devices managed by the OS of the computer that it is being ran as well as the standard hardware connections with the appropriate network ports.

2.1.3 Software Interfaces
There is no significant different software systems that are interacting with each other besides the software application itself and the underlying operating systems.

2.1.4 Communications Interfaces
There is no significant differences between the systems besides the devices they may be running on, but the communication portions of the software application over a network will be handled by the underlying operating systems and messages passed over a network socket communication protocol based on the software application functionality.

2.1.5 Memory Constraints
The memory constraints to run the Not Catan have not been decided upon yet.

2.1.6 Operations
The user has various modes of operations that it can initiate to interact with the Not Catan software application. The software is constantly interactive with the user when the game has started.

2.1.6.1 When it is not the user's turn in game, the user's operations will only look as if they are being attempted rather than actually executing any interactions besides the user moving the graphical representations.

2.1.6.2 When it is the user's turn in game, the user's operations will actively attempt to execute, causing the logical checks, client network communications, and consequences of said operation of the GUI by the user.

2.2 Product Functions

With the Not Catan software product, each user will be able to play a game with three other players over a network.

Users will have to either register a new profile or select an existing one then select either to host or connect to a game. If a host has already been decided upon the host will have their IP address to connect to displayed to give to other players. If a user would like to just connect to a game that already has a host ready that is waiting to connect to the other players, the users connecting to a game will have to type in the IP address that the host will have on display.

Once all connections are made, the application itself will determine player order for turns and begin the game from the default setup. Each user will be able to interact with the game board and manipulate the objects that are displayed but unless it is that user's turn, the actual action does not execute or make any checks, reverting to the previous rendition of the game board that is represented by the graphical user interface in the software application.

The software application then plays out based on the set game rules and the actions of all users during their turns; attempting to complete the objective of the game using the interactions and functions at the user's disposal by gaining 10 victory points.

2.3 User Characteristics

Users of Not Catan will be of two varieties. The first is the normal user using the software and connecting to a game. The only difference between this type of user and the next is the host user that decides to host the game.

2.3.1 Normal User

Normal Users will only have the Graphical User Interface and the Client Control parts of the software application running on their computers.

2.3.2 Host User

Host Users are the same as the Normal Users, however they additionally have the Host Control and Game Engine parts of the software application running on their computers.

2.4 Constraints

The software application is constrained by the need for a network to be able to connect and play games with other players. This constraint also implies that the host user must have a stable connection to the network as well as a suitably more powerful computer in order to run the other required parts of the software application to run the game.

2.5 Assumptions and Dependencies

An assumption about the Not Catan software application is that it is going to be used on computers that have enough resources to run the application.

Another assumption is that the computers that the software will be ran on will be on a Local Area Network that allows them to connect to each other through local ports.

Section 3
Specific Requirements

3.1 External Interfaces

3.1.1 User Interfaces

3.1.1.1 Graphical User Interface

3.1.4 Communication Interfaces

3.1.4.1 Network Socket Communication Protocol

3.2 Classes/Objects

3.2.1 Vertex

3.2.1.1 Attributes

3.2.1.1.1 self.edge_arr = [self.e1, self.e2, self.e3]
3.2.1.1.2 self.adj_vertices = set (( ))
3.2.1.1.3 self.tile_arr = [self.t1, self.t2, self.t3]
3.2.1.1.4 self.settlement
3.2.1.1.5 self.city

3.2.2 Edge

3.2.2.1 Attributes

3.2.2.1.1 self.edge_arr = [self.e1, self.e2, self.e3, self.e4]
3.2.2.1.2 self.tile_arr = [self.t1, self.t2]
3.2.2.1.3 self.vertex_arr = [self.v1, self.v2]
3.2.2.1.4 self.road

3.2.3 Tile

3.2.3.1 Attributes

3.2.3.1.1 self.tile_arr = [self.t1, self.t2, self.t3, self.t4, self.t5, self.t6]
3.2.3.1.2 self.vertex_arr = [self.v1, self.v2, self.v3, self.v4, self.v5, self.v6]
3.2.3.1.3 self.edge_arr = [self.e1, self.e2, self.e3, self.e4, self.e5, self.e6]
3.2.3.1.4 self.type
3.2.3.1.5 self.value

3.2.4 Board

3.2.4.1 Attributes

3.2.4.1 self.tile_array =
[[Tile.Tile ( ) for _ in range (3)],
[Tile.Tile ( ) for _ in range (4)],
[Tile.Tile ( ) for _ in range (5)],
[Tile.Tile ( ) for _ in range (4)],
[Tile.Tile ( ) for _ in range (3)]]

3.2.4.2 Functions

3.2.4.2.1 connect_tiles (none)

3.2.4.2.2 add_edges (none)

3.2.4.2.3 add_vertices (none)

3.2.4.2.4 connect_edges_to_edges (none)

3.2.4.2.5 connect_vertices_to_vertices (none)

3.2.4.2.6 __repr__ (none)

3.2.4.2.7 is_valid_coordinate (x, y)

3.2.5 Inventory

3.2.5.1 Attributes

3.2.5.1.1 Player Resources

3.2.5.1.1.1 self.brick = 0
3.2.5.1.1.2 self.lumber = 0
3.2.5.1.1.3 self.wool = 0
3.2.5.1.1.4 self.grain = 0
3.2.5.1.1.5 self.ore = 0

3.2.5.1.2 Player Assets

3.2.5.1.2.1 self.roads = [ ]
3.2.5.1.2.2 self.settlements = [ ]
3.2.5.1.2.3 self.cities = [ ]
3.2.5.1.2.4 self.dev_cards = [ ]

3.2.5.2 Functions

3.2.5.2.1 add_road (road)

3.2.5.2.2 add_settlement (settlement)

3.2.5.2.3 add_city (city)

3.2.5.2.4 add_dev_card (dev_card)

3.2.5.2.5 has_road (none)

3.2.5.2.5 has_settlement (none)

3.2.5.2.5 has_city (none)

3.2.5.2.5 has_dev_card (none)

3.2.6 Player

3.2.6.1 Attributes

3.2.6.1.1 self.netid
3.2.6.1.2 self.name
3.2.6.1.3 self.color
3.2.6.1.4 self.inventory

3.2.6.2 Functions

3.2.6.2.1 longest_road (none)

3.2.6.2.2 road_length (road, visited_roads)

3.2.7 Game State

3.2.7.1 Attributes

3.2.7.1.1 self.board
3.2.7.1.2 self.player_array
3.2.7.1.3 self.invalid_vertices_to_build_array = set (( ))
3.2.7.1.4 self.current_player
3.2.7.1.5 self.current_player_number
3.2.7.1.6 self.robber_tile

3.2.7.2 Functions

3.2.7.2.1 initial_setup (none)

3.2.7.2.2 setup_helper (settlement1, settlement2, road1, road2, player)

3.2.7.2.3 add_invalid_vertices_to_build (vertex)

3.2.7.2.4 vertex_check (vertex)

3.2.7.2.5 has_connected_road (vertex)

3.2.7.2.6 not_on_opp_sett (edge)

3.2.7.2.7 road_has_connected_road (edge)

3.2.8 Game Engine

3.2.8.1 Attributes

3.2.8.1.1 self.game_state

3.2.8.2 Functions

3.2.8.2.1 build (build_info)
The Game_Engine uses this method in order to determine which action, if said action is valid, and to complete the instantiation of the object.
Args:
build_info (BuildInfo): Object with relevant information to dynamically determine what and where the object to instantiate in the game_state.
Returns:
True (bool): Returns a True boolean value if the game_state was modified and the item determined from the passed info was built.
False (bool): Returns a False boolean value if the game_state was not modified for any reason.

3.2.8.2.2 check_player_inventory (build_type)
The Game_Engine uses this method in order to check if the current_player has the appropriate attributes in order to proceed with the build ( ) method that invoked this method.
Args:
build_type (str): A string argument that is passed to this method to determine which object the build ( ) method is to try and instantiate and check whether or not the current_player has the appropriate attributes to actually instantiate the type of object that the build() method is trying to instantiate.
Returns:
True (bool): Returns a True boolean value if the current_player has the appropriate attributes to instantiate the type of object the build() method is attempting to instantiate.

False (bool): Returns a False boolean value if the current_player does not have the appropriate attributes to instantiate the type of object the build() method is attempting to instantiate.

3.2.8.2.3 check_location (object_to_build_on)
The Game_Engine uses this method in order to check if the object_to_build_on value is a valid object to proceed with the build ( ) method that initially invoked this method.
Args:
object_to_build_on (object): An object argument that is passed to this method to determine whether or not the object is a valid value that in order to proceed with the build ( ) method that invoked this method.
Returns:
True (bool): Returns a True boolean value if the object_to_build_on value is valid in order to proceed with the build ( ) method that invoked this method.
False (bool): Returns a False boolean value if the object_to_build_on value is not a valid value in order to proceed with the build ( ) method that invoked this method.

3.2.8.2.4 build_item (object_to_build_on)
The Game_Engine uses this method in order to instantiate the object that the build ( ) method has determined as the valid object to instantiate after previous checks, check_location ( ) and player_inventory ( ) methods, have returned a True boolean value.
Args:
object_to_build_on (object): An object argument that is passed to this method to determine which object that the build ( ) method, that invoked this method, is attempting to instantiate.
Returns:
True (bool) Returns a True Boolean value once the object has been instantiated by this method at the appropriate location and added to the appropriate player's inventory.

3.2.8.2.5 next_player (none)

3.2.8.2.6 dice_roll (none)

3.2.8.2.7 update_vps (none)

3.2.9 Build Info

3.2.9.1 Attributes

3.2.9.1.1 self.row
3.2.9.1.2 self.column
3.2.9.1.3 self.edge
3.2.9.1.4 self.vertex
3.2.9.1.5 self.dev_card

3.2.10 Host Control

3.2.10.1 Attributes

3.2.10.1.1 self.sock
3.2.10.1.2 self.clients
3.2.10.1.3 self.requests

3.2.10.2 Functions

3.2.10.2.1 get_conns (none)

3.2.10.2.2 rec_data (client)

3.2.10.2.3 send_data (client)

3.2.10.2.4 close (none)

3.2.11 Host

3.2.11.1 Attributes

3.2.11.1.1 host
3.2.11.1.2 player_addrs
3.2.11.1.3 player_array
3.2.11.1.4 netid
3.2.11.1.5 player1
3.2.11.1.6 player2
3.2.11.2.7 game_engine
3.2.11.2.8 roll

3.2.11.2 Functions

3.2.11.2.1 broadcast_message (message)

3.2.11.2.2 broadcast_resources (none)

3.2.11.2.3 broadcast_vps (none)

3.2.11.2.4 make_build_info (request)

3.2.12 Client Control

3.2.12.1 Attributes

3.2.12.1.1 self.sock
3.2.12.1.2 serv_addr
3.2.12.1.3 self.requests

3.2.12.2 Functions

3.2.12.2.1 send_build_obj (msg)

3.2.12.2.2 rec_data (none)

3.2.12.2.3 close (none)

3.2.13 Graphical User Interface

3.2.13.1 Attributes

3.2.13.2 Functions

3.2.14 Constants

3.2.14.1 MAX_ROADS = 15
3.2.14.2 MAX_SETTLEMENTS = 5
3.2.14.3 MAX_CITIES = 4
3.2.14.4 NUMBER_OF_CLIENTS = 2

3.3 Performance Requirements

3.3.1 Static Numerical Requirements

3.3.2 Dynamic Numerical Requirements

3.4 Design Constraints

3.3.1 Standards Compliance

3.5 Software System Attributes

3.5.1 Reliability

3.5.2 Availability

3.5.3 Security

3.5.4 Maintainability

3.5.5 Portability

3.6 Additional Comments

Section 4
SRS Change Management Process

4.1 Change, Edit, Addition, or Deduction Entry Procedure

4.1.1 Dedicated Location for Entry Log

4.1.1.1 Section 5 of this SRS document.

4.1.2 Content

4.1.2.1 Content of the change/edit, addition, or deduction should be located in the body of the SRS and the dependencies should also be changed to suit the content and its details.

4.1.3 Entry Format

4.1.3.1 Date	mm/dd/yy
4.1.3.2 Location	Section X._._ to Section Y._._
4.1.3.3 Type	Change/Edit | Addition | Deduction (brief description)
4.1.3.4 Dependencies	Dependencies Resolved Y | N \ Location
4.1.3.5 Author	Last, First
4.1.3.6 Verified	Verified Y | N \ Last, First \ Date

Section 5
SRS Document and Change Approvals

Entry 5.1

5.1.1 11/13/17
5.1.2 Section 1 to Section 5
5.1.3 Addition (Iteration 2)
5.1.4 Dependencies Resolved: Y \ Section 1 to Section 5
5.1.5 Grant, David
5.1.6 Verified: Y \ Laney, Bradley \ 11/18/17

Entry 5.2

5.2.1 11/18/17
5.2.2 Section 1 to Section 2
5.3.3 Change/Edit (Settlers of Definitely Not Katan to Not Katan)
5.4.4 Dependencies Resolved: Y \ Section 1 to Section 2
5.5.5 Grant, David
5.6.6 Verified: Y \ Laney, Bradley \ 11/18/17

Entry 5.3

5.3.1 11/26/17
5.3.2 Section 1 to Section 2
5.3.3 Change/Edit (Katan to Catan)
5.3.4 Dependencies Resolved: Y \ Section 1 to Section 2
5.3.5 Grant, David
5.3.6 Verified: Y \ Laney, Bradley | Kim, Joseph \ 11/28/17

Entry 5.4

5.4.1 11/27/17
5.4.2 Section 3
5.4.3 Change/Edit | Addition | Deduction (Iteration 3: Edited a multitude of entries to bring document up to date with current code, readability, and some commenting/details)
5.4.4 Dependencies Resolved: Y \ Section 3
5.4.5 Grant, David
5.4.6 Verified: Y \ Laney, Bradley | Kim, Joseph \ 11/28/17

Entry 5.5

5.5.1 11/28/17
5.5.2 Title Page, Contents Page
5.5.3 Change/Edit | Addition (Changed appropriate information on Title Page, added details to Contents Page)
5.5.4 Dependencies Resolved: Y
5.5.5 Grant, David
5.5.6 Verified: Y \ Laney, Bradley | Kim, Joseph \ 11/28/17

Entry 5.6

5.6.1 11/28/17
5.6.2 Section 3
5.6.3 Change/Edit | Addition | Deduction (Added Host class, and edited various class functions that were added and deducted)
5.6.4 Dependencies Resolved: Y
5.6.5 Grant, David
5.6.6 Verified: Y \ Laney, Bradley | Kim, Joseph \ 11/28/17