

Estructopoly

Arturo Díaz, David Montaña, and Samuel Hernández

ITC Department, ICESI University

09687: Algorithms and Data Structures

Johnatan Garzón Montesdeoca

Estructopoly: Engineering Method

Estructopoly is a board game based on the exchange and sale of real estate (normally, inspired by the names of the streets of a certain city), the objective of the game is to win the most money and property while achieving your opponent's go bankrupt. This is achieved by acquiring properties, constructing houses and buildings, and collecting rent for your properties.

Players take turns moving their respective tokens clockwise around a board, based on the dice score, and they land on properties that they can buy from an imaginary bank. If the properties in which they fall already have owners, the owners can charge to pass through their property or whoever falls will be able to buy them, in case of advancing by chance or communal ark the properties cannot be bought. Next up, an engineering approach to solve the problem using the Engineering Method.

Context of the Problem

As engineers we have a computer in our backpack most of the time and we don't have space to carry games with us to spend time with our co-workers, so we plan to develop a new version of monopoly for the desktop and thus be able to pass the time.

Development of the Solution

Based on the description of the engineering method given in the book Introduction to Engineering by Paul Wright, the following flowchart was drawn, and will be followed according to the steps shown in it during the development of the solution.

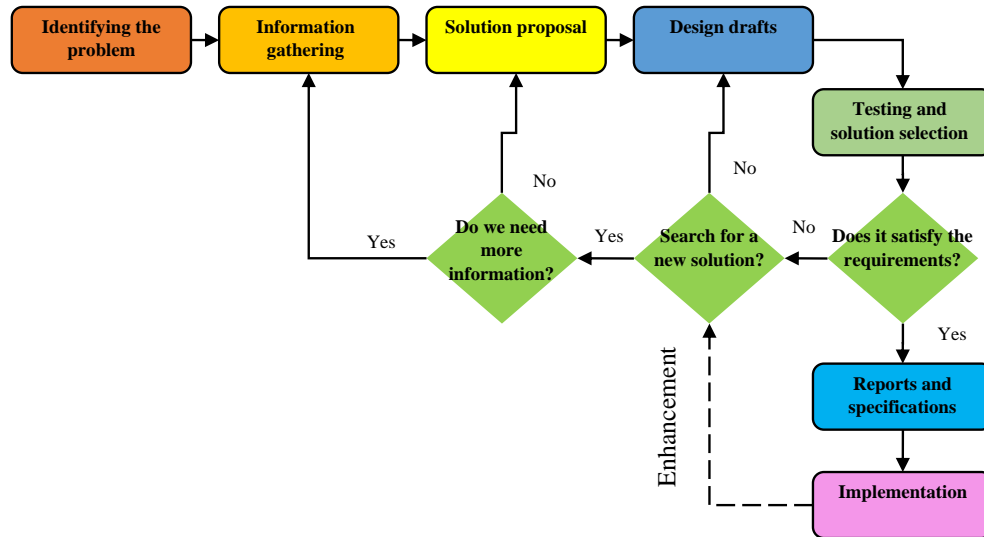


Figure 1. Flowchart representing The Engineering Method proposed by Paul Wright.

The steps shown in figure 1 are elaborated in detail following up.

Identifying the Problem

Identifying symptoms and necessities

- **Go (Forward):** It is the starting square of the game. Every time a player passes through this box, he is entitled to receive a salary of \$ 200, except when you are on your way to Jail.
- **Communal Ark or Chance:** The player must draw a card from the corresponding pack and follow the instructions indicated on it.
- **Properties:** When a player reaches a space of a property, he can acquire it, if it does not have an owner. If the property has an owner, he must pay the corresponding rent. Properties include: land, railways, and services such as electricity and water.
- **Taxes:** The player must pay the bank 10% of his total assets or an estimate of \$ 200.

- **The Jail:** A player goes to the jail space when he lands on the space that indicates "go to jail", picks up the card that indicates it or gets three doubles in a row when rolling the dice. To get out of jail, he must get a double on one of the next three turns, use the "Get Out of Jail" card, or pay a \$ 50 fine before his next turn.
- **Free parking:** rest area.

Definition of the Problem

We require a comfortable and friendly user interface, in addition that the game runs smoothly to generate a good user experience, with the condition of respecting the traditional rules of the game

Information Gathering

Given the technological context of the solutions to be proposed, the following terms must be defined prior to anything:

Software: Software is every application program and operative system that allow the computer can run smart tasks, directing the physical components or hardware with instructions and data through several different kinds of programs.

Simulation Software: A simulation software has the objective of facilitating or automating the modelling process for a real-world phenomenon, using mathematical formulas through programming. At its core, it is a program that allows the user to see what will happen after doing a specific action or set of actions, without having to do it in the real world.

Graphical User Interface (GUI): A graphical user interface (GUI by its English acronym) is a program or environment that manages the interaction with the user basing itself on visual relations such as icons, menus, or pointers.

Now, Players start in turn, the player who starts the game is randomly decided before the game. Each player starts the game with \$ 1500. To buy avenues, it is not necessary to go around two times but only one. In turn, as is customary, you start by rolling the dice and your piece is advanced clockwise around the board, the corresponding number of spaces.

If a player falls into the Chance or Communal Ark space, he picks up the top card of the corresponding deck and follows the instructions written on it (these cards must be face down before starting the game). Once the card is drawn, it is placed at the bottom. If the player lands on an ownerless property, be it a street, a railroad, or a utility, he may purchase it unless he is in the first round of the game, the property for the purchase price on the list. If he decides not to make this purchase, the property is available to another player who has the money to buy it. If the property already owns an owner, you must pay the owner a certain rent, the price depends on whether the property is part of a complex or its level of development. If a player rolls the same value with the 2 dice (roll doubles), he rolls again after completing his turn. If the player rolls 3 doubles in a row, he must go to jail. In the event that in the first round you go to jail, that lap will not count, you will have to finish turning and completing it.

If a player is in jail, he does not take a normal turn, he can pay \$ 50 to be released from jail, or try to roll a double five on the dice. If a player does not roll a double five to get out of jail, he gives up his turn. If he tries this 3 times he will have to pay the \$ 50 penalty per obligation to be released. While the player is in jail, he can still sell property and buildings, and collect rents. If a player rolls a double five, he can immediately move according to the amount indicated by the dice, but cannot do so a second time after being released from jail. During a player's turn, that player can also choose to develop properties, if he possesses all the properties of the color group. The development involves the construction of houses or hotels on the properties, of certain

amounts of money that is paid to the bank and is tracked on the board by adding houses and hotels to the plaza. Development must be uniform across a monopoly (group of properties of the same color), such that a second home cannot be built on one monopoly property until the others have a home.

Although houses and hotels cannot be built on railways or utilities, the rent increases if a player has multiple railways or both utilities.

Properties can also be mortgaged, although all developments made on the monopoly must be sold before the property is mortgaged or marketed. The bank also pays the player every time the established price falls on his property. The player receives money from the bank for each mortgaged property, which must be returned with interest to withdraw the mortgage. Houses and hotels can be sold back to the bank at half price. Property cannot be given to another player.

When a player declares bankruptcy, he is eliminated from the game, if he cannot pay what he owes. If the bankrupt player owes the bank, he must transfer all of his property to the bank. If the debt is to another player, all properties must be given to the opponent, but the new owner has to pay the bank to withdraw the mortgage for any property received. The winner is the player left after everyone else files for bankruptcy.

Solution Proposal

3D, UNITY, FXML

Proposal 1: A board object is serialized with all the information needed for the game, such as wild cards, properties, total money; In this way, the program can be loaded quickly and easily so as not to make the user experience a bad thing, it is proposed to create 2 main screens, 1. a screen

with a button to select play and another to see the score of past games , 2. another screen where the game board is and all the necessary information to be able to play.

Proposal 2: It is proposed that the information of the game is housed in a csv file so that the user can easily configure the structure of the game and be able to add new cards, be they wild cards, properties and modify the initial values of each player.

It is said that it is more efficient to develop only an fxml screen where the game board and all the necessary information for the game are already located.

Proposal 3: It is proposed to put together the proposals described in points 1 and 2 to make a more complete and diverse game in different aspects, in addition to adding an option to print in a pdf all the events that happened during the game

Solution Selection

It was found that the first proposal provides the user with few tools to personalize their gaming experience and power, but it was found that it is the proposal that best distributes the game screens to make the game simpler and not load a single screen with all the information.

The second option was found to offer users better customization options, but its repertoire of screens is very ineffective because it relies on loading a screen with all the game information and options.

It was defined to develop the game considering proposal number 3, since it is the most complete and provides the user with a broader repertoire of options so that they have the best gaming experience, since they will be able to customize all aspects of the game to your liking, create different rules




Design Drafts





Figure 2. Main View of the software

FIBA Stats

PLAYER REGISTRATION

Name 
 Team 
 Number Is Active ☐ Yes ☐ No
 Points Turn Over
 Offensive BPM Rebound
 Defensive BPM Usage


 SELECT IMAGE

Figure 3. Player registration View

FIBA Stats

LIST OF PLAYERS

Search...



Name	Team	Number	Is Active	Points
Pepo	FIB	12	true	12.0

Double click on a player to select it

Figure 4. List of Players

FIBA Stats

TEAMS REGISTRATION

Name _____ Country _____

Search by Name _____

Name	Country
FIB	COL
ICS	MEX

< >

Double click on a team to select it

Figure 5. Team registration view

Model Class Diagram

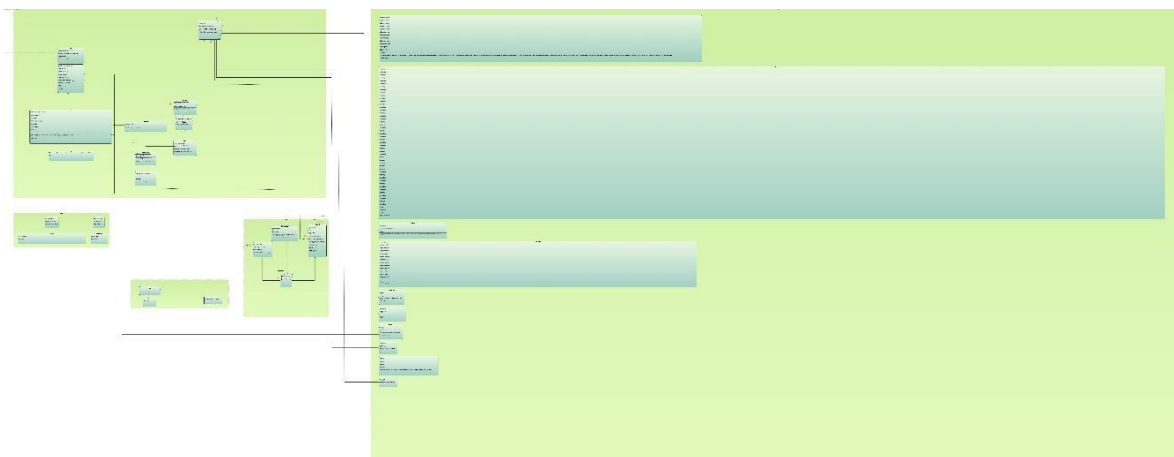
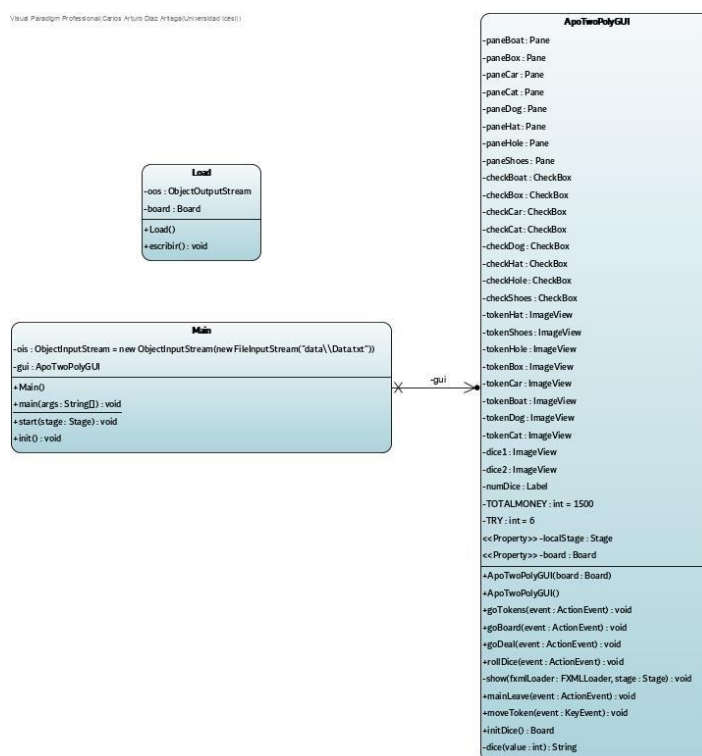
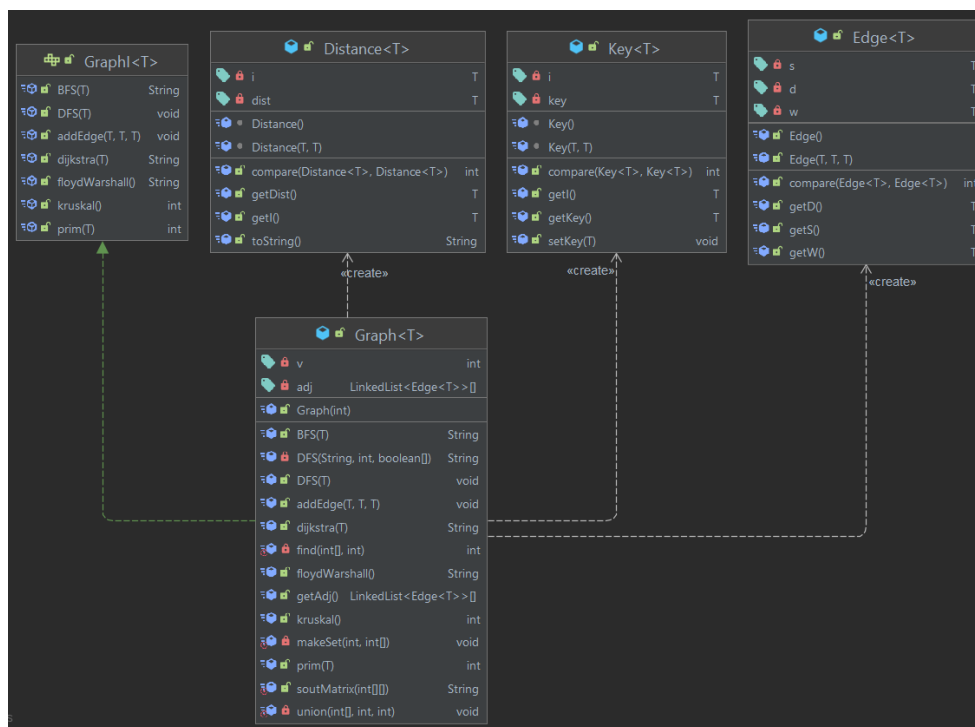


Diagrama de Clase del UI

Visual Paradigm Professional Carlos Arturo Diaz Ariaga(Universidad Icesi)



Graph diagram



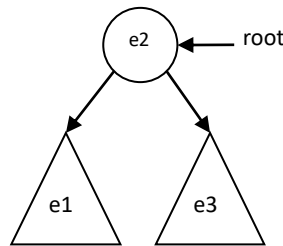
Test graph diagram



Abstract Data Types

This project requires the usage of some abstract data structures to properly model the problem it attempts to solve. Namely, Binary Search Trees (BST), Self-Balancing Trees (AVL), and Red Black Trees (RBT), all of which will be shortly characterized below, taking into account AVLs and RBTs inherit from BSTs.

Binary Search Tree (BST) ADT



$$BST = \langle \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle, root \rangle$$

$$\{inv: \forall n (e_{n-1} \wedge key[e_{n-1}] \leq key[e_n]) \vee (e_{n+1} \wedge key[e_{n+1}] \geq key[e_n])\}$$

$$BST \rightarrow BST$$

$$search: BST \times Key \rightarrow Element$$

$$insert: BST \times Element \rightarrow BST$$

$$delete: BST \times Element \rightarrow BST$$

$$min: BST \rightarrow Element$$

$$max: BST \rightarrow Element$$

$$successor: BST \times Element \rightarrow Element$$

$$predecessor: BST \times Element \rightarrow Element$$

BST

Builds an empty Tree

$$\{pre: null\}$$

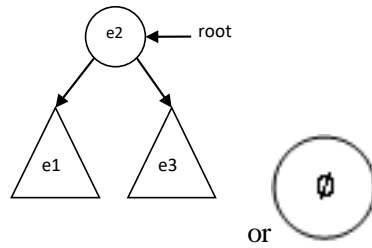
$$\{post: BST \ t = \emptyset\}$$



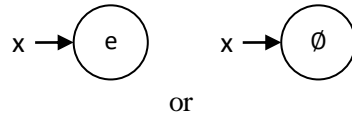
search

Retrieves element e from tree t

$$\{pre: (BST \ t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle \wedge e) \vee (t = \emptyset \wedge e)\}$$



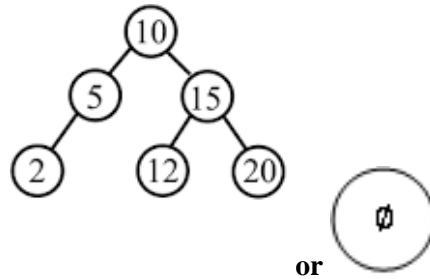
$\{post: [(t = t \wedge x = e) \wedge (e \ni t)] \vee (x = \emptyset)\}$



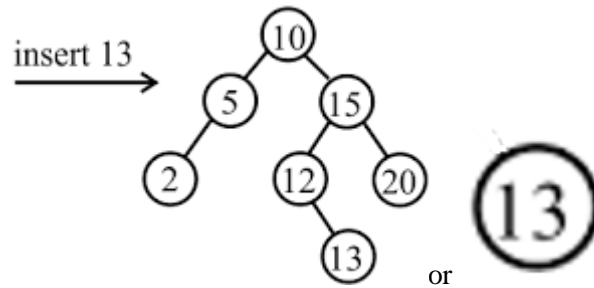
insert

Adds element e to tree t .

$\{pre: (BST\ t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle \wedge e) \vee (t = \emptyset \wedge e)\}$



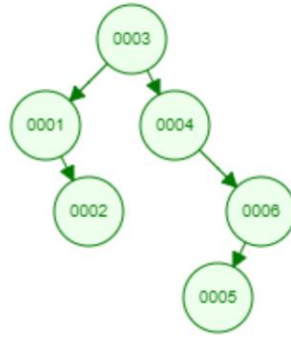
$\{post: BST\ t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1}, e \rangle \vee t = \langle e \rangle\}$



delete

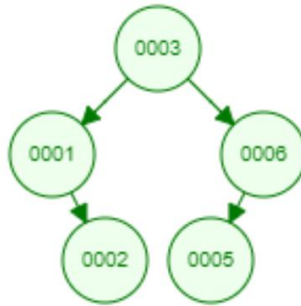
Removes element e from BST t .

$\{pre: BST\ t \neq \emptyset\ i.e.\ t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle\}$



$\{post: BST\ t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle\}$

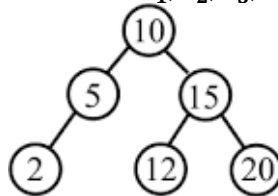
Delete 4



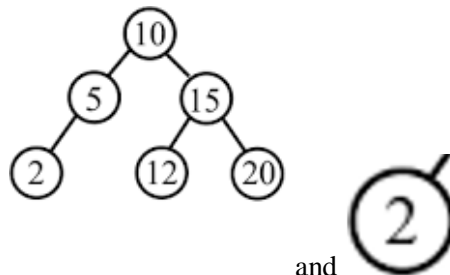
min

Fetches minimum element e from tree t .

$\{pre: BST\ t \neq \emptyset \text{ i.e. } t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle\}$



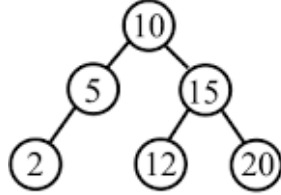
$\{post: BST\ t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle \wedge e\}$



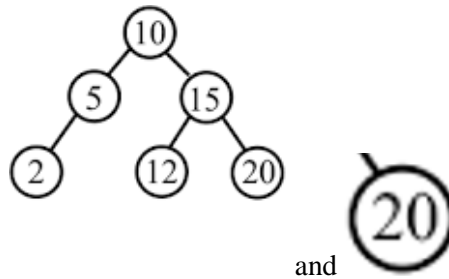
max

Fetches maximum element e from tree t .

$\{pre: BST\ t \neq \emptyset\ i.e.\ t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle\}$



$\{post: BST\ t = \langle e_1, e_2, e_3, \dots, e_{n-1}, e_n, e_{n+1} \rangle \wedge e\}$



Bibliography

- FIBA Central Board. (2021, March 26). *Document Library*. Retrieved September 29, 2021, from FIBA.basketball: <http://www.fiba.basketball/documents>
- FiveThirtyEight. (2021, August 30). *2021-22 NBA Player Projections*. Retrieved from FiftyThirtyEight: <https://projects.fivethirtyeight.com/2022-nba-player-projections/>
- Hagness, M. (2021). *The Most Important Stats To Track For Your Basketball Team*. Retrieved from Basketball Breakthrough: <https://www.breakthroughbasketball.com/stats/how-we-use-stats-Hagness.html>
- Leadoff Digital. (2015). *NBA Dictionary*. Retrieved from Sporting Charts: <https://www.sportingcharts.com/NBA/dictionary/>
- Sports Reference LLC. (n.d.). *Glossary*. Retrieved from Basketball-Reference.com - Basketball Statistics and History: <https://www.basketball-reference.com/about/glossary.html>