CS-205 Hibbeler
Joshua Dickerson, Anders Melen, Katie Smith, Dana Desautels
02/25/2013 - University of Vermont

Rat-A-Tat-Cat Design Specifications

Table Of Contents

- Purpose
- Definitions, Acronyms, & Abbreviations
- Introduction
- Project Road Map
- Requirements
- Gameplay & Implementation

Purpose

The purpose of this document is to provide a general framework by which our team will create a full-featured, web-based clone of the popular card game Rat-A-Tat-Cat. Using a combination of static HTML/CSS and embedded Java applets, out user interface will provide a rich experience for users of all ages. Game-play data will be stored in a publicly available database from which a multitude of charts will be populated, and statistical conclusions can be drawn.

Definitions, Acronyms, & abbreviations

Turn - An individual's own turn within a round of game-play.
Round - A complete game, in which one player has knocked and scores tallied. Games can be multiple rounds.

Introduction

Rat-A-Tat-Cat is an award winning card game, produced by Gamewright Inc., for children and adults of all ages. The objective is simple; get the lowest scoring hand and "knock" to end the round before your opponents beat you to it. This implementation of Rat-A-Tat-Cat will strive to maintain all of the game rules as prescribed, using GUIs and a detailed tutorial. The GUIs will be intuitive, and draw from a color palette based closely off the colors used in the card game. A tutorial will be available for new players, and also serve as a refresher for more experienced players. The project will post game-time statistics to a public web-server where it will be stored in a relational database.

A PHP script running on the server will periodically refresh dynamically generated figures and charts with up-to-date data culled from the database, that highlight statistical points of interest related to AI and Human user performance. The AI will have multiple settings for adjusting difficulty, in particular memory capability and a risk threshold for switching cards. Users will be prompted with a login screen, in which they may enter their names, and have their scores saved. Statistics will be drawn from the

comparison of these scores, as more players are entered. The scores, after each game, will be displayed on a leader-board which will show the top player's scores, and individual score.

Road Map

Deliverable I: *Specifications* Due Wednesday, Feb 27th, 9:35 am
(Specification)

- Receive final project outline from J. Hibbeler
- Introductory meeting
- Play game and hash out game flow
- Develop design-spec documentation
- Decide on which technologies to implement
- Implement version control & ensure all members are tied in

Deliverable II: *Test Results* Due Wednesday, Apr 3rd, 9:35 am
(Development and Verification)

- Define component pieces
- Define coding / syntax standards
- Define version-control standards (commits and pushes)
- Define testing standards
- Prioritize component development
- Define strict interfaces between components
- Delegate workload and define deadlines for priority components
- Individual and Group coding sessions (sprint phase)
- Individual testing
- Group testing

Deliverable III: *Final Product* Due Friday, Apr 22nd, 9:35 am
(Product Packaging)

- Conduct usability testing, and make needed changes
- Test product on a multitude of platforms
- Disseminate presentation tasks
- Prepare presentation
- Practice presentation
- Deliver product to class

Specific Requirements

User Interface

GUI :: Displayed in Browser
- Instructions
- Game-play (must show card graphics)
- Game statistics
- Leader-board

AI Play

Granular user control of AI player difficulty level.
- AI's play memory
- AI's risk tolerance

Version Control

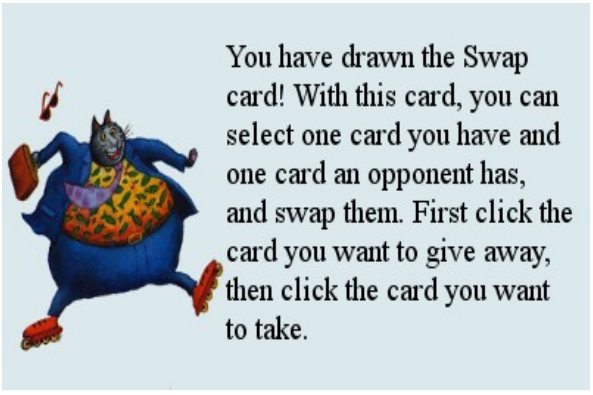
The project will employ a GIT repository stored at https://github.com/JoshuaDickerson/cs205-Final_Project. The client is encouraged to create a free GIThub (http://github.com) account in order to view real-time project progress.

Database

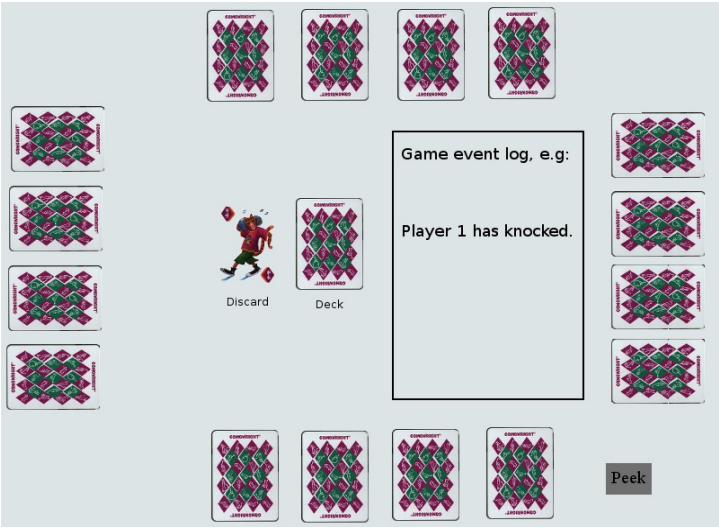
The project will post game-time statistics to a public web-server where it will be stored in a relational database. A PHP script running on the server will periodically refresh dynamically generated figures and charts with up-to-date data culled from the database, that highlight statistical points of interest related to AI and Human user performance._

Gameplay & Implementation

Upon opening the game, the UI queries the user for initialization data such as the number of human and computer players, the names of the human players, and the AI difficulty. The user is then prompted to decide whether the game should proceed in normal mode or a tutorial demo mode. If the demo mode is selected, the user is taken through a tutorial round of the game. The round will consist of tool-tips for what to click, hints for suggestions on strategy, and the entire round is played with all cards up in order to remove any ambiguity and allow the player to see and understand the process.



If the normal mode was selected at setup, then the user will see no hints or tool-tips, and their (and others') cards will be dealt face down per the manufacturer's rules.



Once the game begins, each player will be shown the two outside cards of their hand. They may not, after the initial phase, look at those two cards again, unless they use a peek power card.

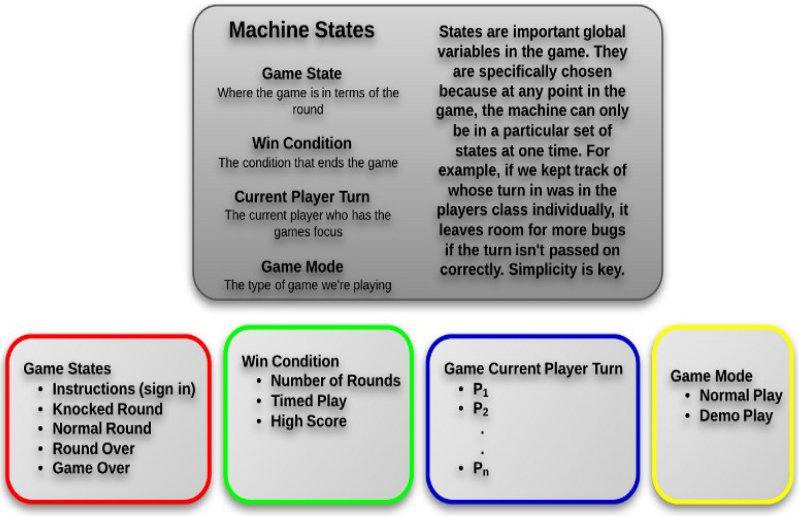
At the beginning of each set of new turns, the system will check for a knock state transition accordingly. If the the game is not in knocked state, the game proceeds normally, and the current player is then prompted with an indicator of their turn. At this point, they then must make a decision whether to draw a card from the deck, or the discard pile.

If the player chooses to select a card from the discard pile, they must replace that same card with one in their hand. They may not pick up if it is a power card in the discard pile.

Otherwise, the player must then draw a card from the deck. The program will display this card to the player, who then must decide whether to replace a card in their hand, or to discard it.

For either choice of drawing from the deck or discard pile, the player is then given a choice whether to knock. If the player chooses not to knock, then the turn completes and the next player's turn begins. If the player does however knock, it flags the turn as knocked. When the knocked flag is set, then the game proceeds to finish one last set of turns for every player until reaching the player who knocked, upon which the round will end.

At the end of the game, all the cards for each player will be shown. Then, the score will be tallied for each player, and a winner will be determined for the round. Two things will then subsequently occur. A leader-board displaying the scores of the top players as well as rank, name, and difficulty will be shown to the user. After, the game will return back to the beginning setup, to start a new game if desired. Simultaneously, data will be sent to the server, and statistics such as name, rank, and score will be stored in the database. A PHP script will then analyze the data and create charts and statistics on the player and the game.



Each stage of the game will be implemented as one of finite number of states, and a player's progression through each state will be logged and stored in a database as JSON, which can then be used to generate game replay and all means of statistical analysis. Each state will loop while waiting for user input and eventual transition. Figures 1.3 – 1.5 provide example states and their transitions.

Fig 1.2 – Game flow

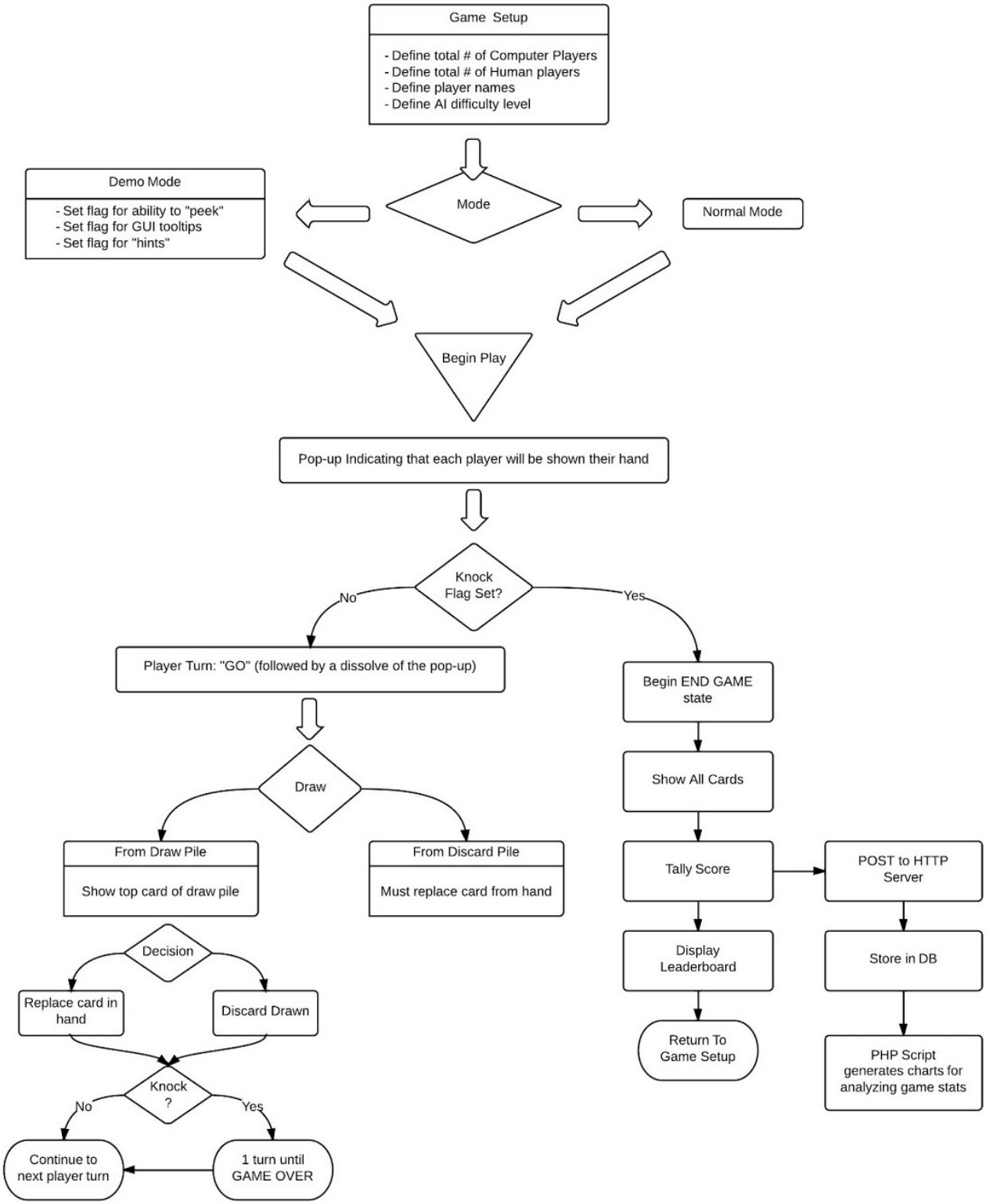


Fig 1.3 Game states ex1

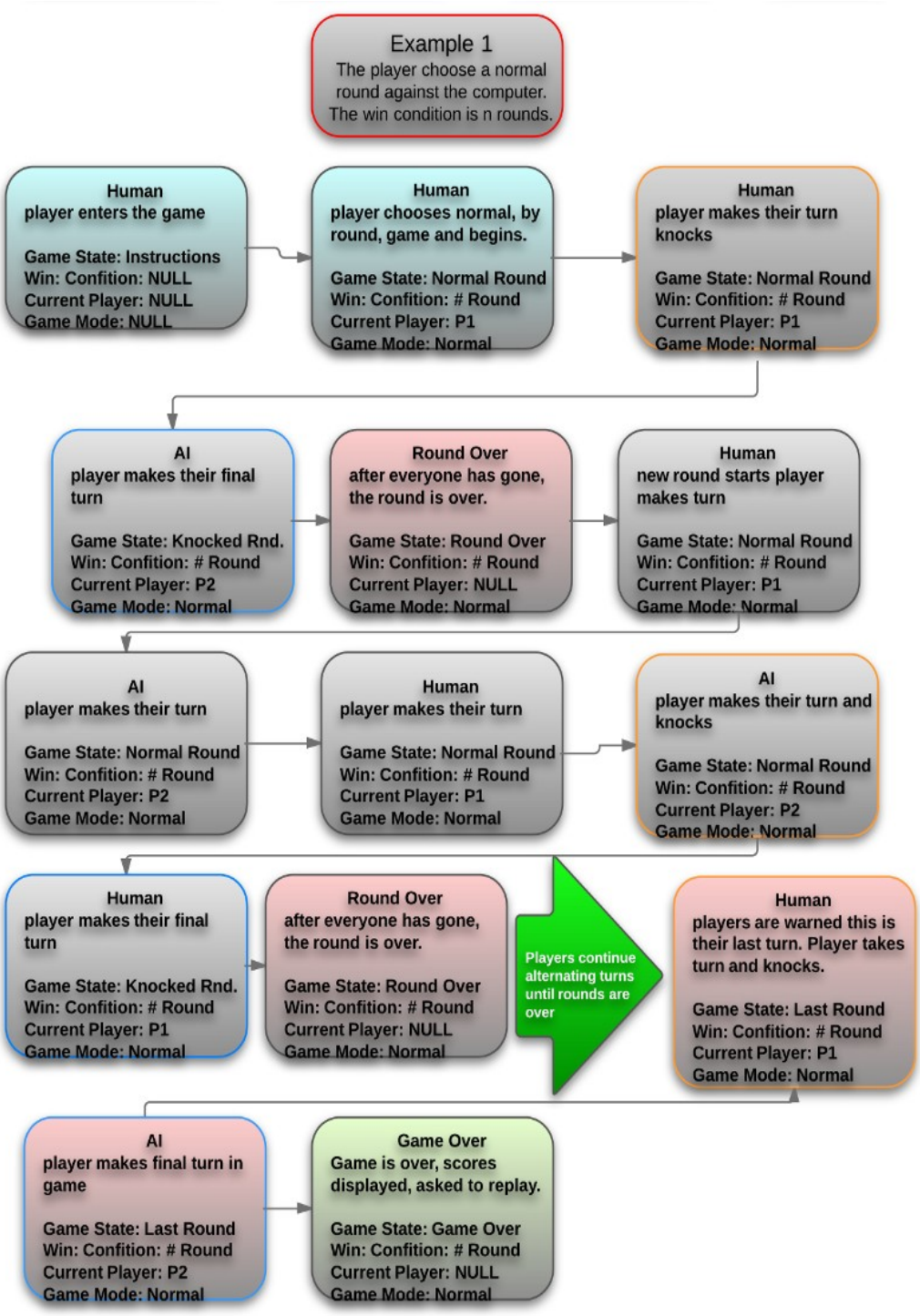


Fig 1.4 Game states ex2

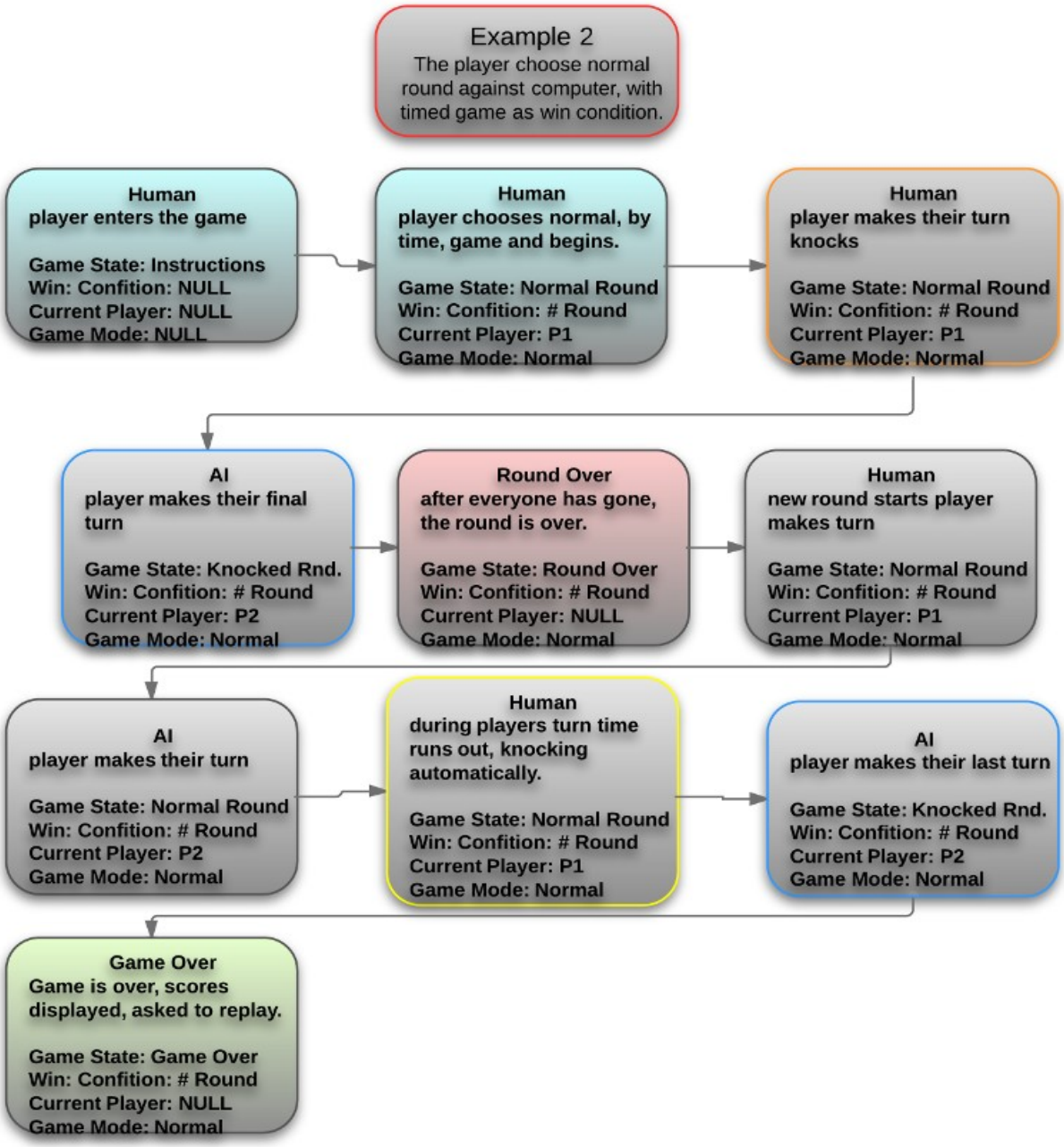
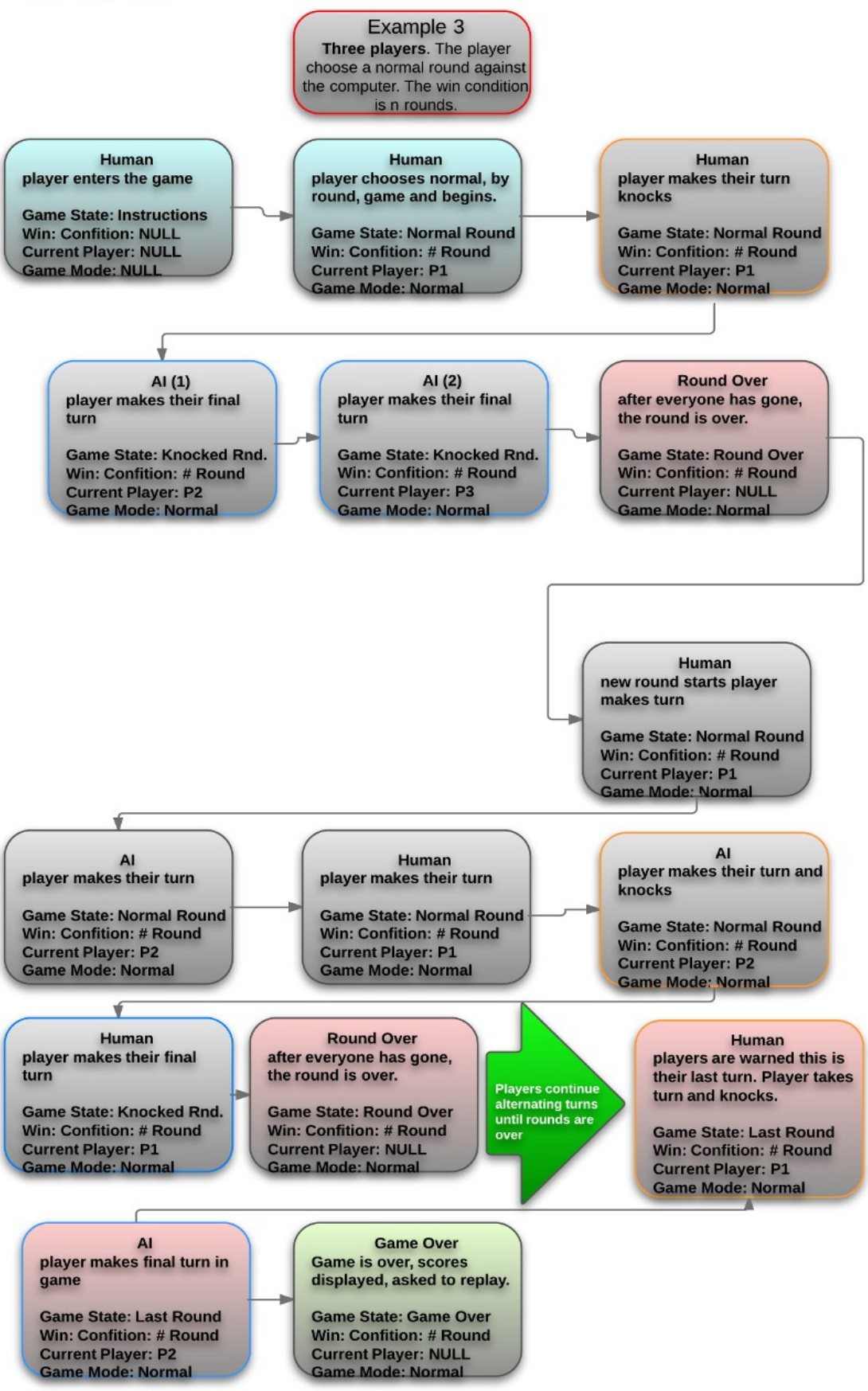



Fig 1.5 – Game states ex3