Concurrent Poker Player

Milestone Two

The Parallel Poker Team
Mark Jenne
Ian Roberts
Bennie Waters
Sarah Jabon

October 20th, 2009

Instuctor:
Sriram Mohan
Rose-Hulman Institute of Technology

Revision History

Date	Revision	Comment
10-03-09	1.0	Initial draft
10-04-09	1.1	Addressed formatting issues
10-09-09	1.2	Addressed Drew's comments
10-20-09	1.3	Addressed Sriram's comments
11-09-09	1.4	Addressed Boby's comments

Signatures

Mark Jenne	
Ian Roberts	
Bennie Waters	
Sarah Jabon	

Contents

Part I
Executive Summary

The parallel poker player project involves designing and developing an application that exhibits the benefits of parallel computing [1] over sequential computing [2]. The application will showcase this through a demonstration of different computer-based players involved in a Texas Hold 'Em poker game [12]. Numerous key elements discussed in this document:

- Background Information - Essential information from Milestone One
- Use Cases - Descriptions of how the user will interact with the system
- Storyboard - Images and a description that show one possible user's experience with the system
- Supplementary Specifications - List of design constraints including restrictions of design choices, conditions imposed on system development, and other regulations
- Data Flow Diagrams - Diagrams that describe the flow of data through the system

The poker application will be a visually stimulating demonstration of the differences between parallel [1] and sequential computing [2].

Part II
Introduction

Milestone Two is the second document of the poker application's requirements and specifications. The contents of this document - use cases and functional requirement descriptions - draw upon the core elements of Milestone One [3], such as user needs and system features. The use cases and supplementary specifications in this document will help the team develop test cases. Based on the test cases, the team will develop an interaction prototype, which will be completed by November 2009. The design of the actual application will be determined by the use cases, supplementary specifications, and the interactive prototype. Once the design is established, the team will create a prototype. This prototype will be fully completed by February 2010. Finally, both the prototype and the incorporated design will be used to create the complete system. The system will be complete by the end of the academic year in May 2010. The objectives of this document are to concisely summarize user interaction and functionality requirements of the poker application. With this core information, it will be possible to develop aspects of the project in fuller detail in the future.

Part III
Background Information

This section includes excerpts from Milestone One that contribute to this document.

1 System Users and Their Needs

There are three main types of users that will work with the poker application:

1. A Microsoft [6] associate that is on the parallel computing [1] team
2. A Microsoft [6] associate that is unfamiliar with the parallel computing [1] team
3. A potential Microsoft [6] customer that is interested in learning about Microsoft's [6] products

All three users have different backgrounds and needs. Microsoft [6] associates that are familiar with parallel computing [1] could be developers, testers, or project managers. They would expect to see a clear demonstration of the key differences between parallel [1] and sequential computing [2]. Since they are familiar with the efficiency of parallel computing [1], they would desire the application to accurately display the benefits over sequential computing [2]. They would also assess the application for its consumer appeal. Their definition of success would be seeing an application that could convince potential customers that parallel computing [1] is applicable to their work and that they should use the Concurrency Development Platform (CDP) API [9]. Therefore, they would expect the application to be visually appealing.

Table 3.1 - User Needs: Potentials of Parallel Computing

Element	Description
The problem of...	customers not realizing the potentials of both the CDP API [9] and parallel computing [1] as a whole
Affects...	potential Microsoft [6] customers, current Microsoft [6] customers, Microsoft [6] shareholders, Microsoft [6] associates, and competing software companies
And results in...	potential customers not using Microsoft's [6] API [9]. If they do not see how effective Microsoft's [6] tools are, they have no reason to use them. If customers are not buying Microsoft's [6] products, then the shareholders will be affected by lower stock prices. Also, Microsoft [6] associates will be affected by how much revenue is coming into their company. Finally, competing software companies may receive more business that would have gone to Microsoft [6].
Benefits of a solution include...	Microsoft [6] acquiring customers if the application impresses them. The poker application is a great way to showcase how efficient parallel computing [1] is. Microsoft [6] will also be showing customers the API [9]. If the company gains more customers, then associates and shareholders of Microsoft [6] will benefit. Finally, competing companies may lose business to Microsoft [6].

The Microsoft [6] associates on the CDP team would also want to improve the API [9], which involves receiving feedback from new users. The poker application provides a great opportunity for programmers with no experience in parallel computing [1] to provide suggestions and comments.

Table 3.2 - User Needs: Improving Microsoft's [6] API

Element	Description
The problem of...	Microsoft [6] needing to improve their API [9]
Affects...	potential Microsoft [6] customers, current Microsoft [6] customers, and competing software companies
And results in...	unhappy Microsoft [6] customers. If Microsoft [6] does not develop their API [9] to the greatest extent possible, their customers may choose competitors' products.
Benefits of a solution include...	Microsoft [6] improving their API [9] with a good understanding of what needs to be changed. The solution is to have programmers who are inexperienced with parallel computing [1] test out the API [9] and provide feedback about it. The poker application will be a great opportunity to acquire this kind of feedback.

The Microsoft [6] associates that are unfamiliar with the CDP team will have a different background. They may be in any department at Microsoft [6], so there is no guarantee that they would be experienced with programming at all. They would be excited to see the capabilities of parallel computing [1] and learn about the CDP's projects. Their idea of success would be seeing an application that explains what the CDP team does. The presentation of the poker application could improve collaboration between departments at Microsoft [6].

Table 3.3 - User Needs: Collaboration between Departments

Element	Description
The problem of...	Microsoft [6] associates in other departments not knowing about the parallel computing [1] division
Affects...	current Microsoft [6] associates
And results in...	less collaboration between departments. If Microsoft [6] associates in other departments do not know what the parallel computing [1] team is working on, they cannot collaborate or make suggestions for extensions of their work.
Benefits of a solution include...	Microsoft improving communication between departments. The Microsoft [6] associates will see an application that represents the CDP team's work, so they may have a better understanding of what they do. This poker application should be flashy enough to catch any associate's attention - even if they were unfamiliar with parallel computing [1].

The potential Microsoft [6] customers that are interested in learning about Microsoft products would usually understand the basics of parallel computing [1]. Their deliverables would vary depending on the type of development team with which they work. However, they could potentially benefit from using parallel computing [1] instead of sequential computing [2]. The customers' definition of success will be seeing that the poker application uses parallel computing [1] to solve a real-world problem in a more efficient way. Seeing an application that was created using the CDP API [9] will help them see how relevant Microsoft [6] products are to their work.

Table 3.4 - User Needs: Customers' Programming Efficiency

Element	**Description**
The problem of...	Microsoft's [6] potential customers not using their computing power up to the full potential
Affects...	Microsoft's [6] potential customers, the programmers that work for them, and the people who buy the customer's products
And results in...	less efficient programming techniques, which leads to companies producing less efficient products
Benefits of a solution include...	customers realizing how to use Microsoft's [6] API [9] to improve the efficiency of their systems. They would learn this by interacting with the poker application and looking at the code behind it.

All users have specific needs that will be addressed with the poker application. The CDP team's needs will be met by the application clearly showcasing the benefits of parallel computing [1]. Also, their need for feedback on the API [9] will be met by poker application's team members providing comments and suggestions in the development period. The solution for the needs of both Microsoft [6] associates who are not on the CDP team and potential Microsoft customers is the visually appealing poker application itself. Hence, all user needs will be met.

2 System Features

The following section describes all system features that have been approved at this time. Refer to the feature scale in the appendix for information regarding the attribute values for each feature.

2.1 System Features List

Feature Number 1	**Feature** Interactive Graphical Interface	
Status Approved	**Priority** Medium	**Effort** Medium
Risk Low	**Stability** High	**Target Release** First release
Assigned To Benjamin Waters	**Rationale** To provide an attractive and interactive interface for users who are utilizing the system. The priority is medium because it is not required for the system, but it would be beneficial to the user for demonstration purposes. The user interface features were designated to Benjamin due to his development experience of front-end systems.	

Feature Number 2	**Feature** Computational Performance Display	
Status Approved	**Priority** Critical	**Effort** High
Risk Low	**Stability** High	**Target Release** First release
Assigned To Benjamin Waters	**Rationale** Constantly provides performance statistics for the analysis of the parallel computing [1], which will reinforce the superiority of parallel computing [1] to the viewer. This feature has a priority of critical because it is necessary to showcase the superiority of parallel computing [1].	

Feature Number 3	**Feature** Parallel Computing [1] Based Player	
Status Approved	**Priority** Critical	**Effort** High
Risk Low	**Stability** High	**Target Release** First release
Assigned To Ian Roberts	**Rationale** Essential for the demonstration of the parallel computing [1] technology in comparison to traditional sequential computing [2]. This feature has a priority of critical because it is a core element of the system. Ian has some experience in working with parallel computing [1] and so will be in charge of the application's usage of and connection to the parallel computing [1] API [9].	

Feature Number 4	**Feature** Sequential Computing [2] Based Player	
Status Approved	**Priority** Critical	**Effort** High
Risk Low	**Stability** High	**Target Release** First release
Assigned To Ian Roberts	**Rationale** Essential for performance comparison between parallel [1] and sequential computing [2] in the application. This feature has a priority of critical because it is a core element of the system. Ian will also be in charge of the sequential computing [2] player to keep it consistent with the parallel computing [1] player.	

Feature Number 5	**Feature** Support for Human Player	
Status Approved	**Priority** Medium	**Effort** Medium
Risk Low	**Stability** High	**Target Release** Second release
Assigned To Ian Roberts	**Rationale** User can interact with the application and play the game with the automated players. The priority is medium because it is not required for the system, but it would be beneficial to the user for demonstration purposes. The target release is the second release; the application will first be developed with only two computer-based players.	

Feature Number	**Feature**	
6	Artificial Intelligence [10] Engine Driving Computer-player Decisions	
Status	**Priority**	**Effort**
Approved	Critical	High
Risk	**Stability**	**Target Release**
High	Medium	First release
Assigned To	**Rationale**	
Mark Jenne	The automated players need to demonstrate basic analysis and reasoning skills and be able to make informed decisions. The artificial intelligence [10] engine poses a high risk. If the engine is developed and not yielding intended or appropriate results in the decision-making process, the structure may need to be reconsidered with a different reasoning system. Since all of the actions of the automated players rely on this feature of the system, it is of critical importance that it be developed properly despite the level of risk present. The AI [10] engine is also the least stable as it may require alterations or redesign if it is not yielding desirable results. Mark has experience in AI [10] development and so will be in charge of the development of and implementation of the AI [10] engine.	

Feature Number	**Feature**	
7	Support for Two or More Automated Players	
Status	**Priority**	**Effort**
Approved	Critical	High
Risk	**Stability**	**Target Release**
Medium	Medium	First release
Assigned To	**Rationale**	
Mark Jenne	The system, particularly the AI [10] engine should be able to support two automated players and preferably more. This feature has a priority of critical because it is a core intended function of the system.	

Feature Number	**Feature**	
8	No-Limit Texas Hold 'Em Poker [12]	
Status	**Priority**	**Effort**
Approved	Critical	High
Risk	**Stability**	**Target Release**
Low	High	First release
Assigned To	**Rationale**	
Sarah Jabon	Game structure of the application which offers a sandbox environment to demonstrate the parallel computing [1]. All the rules of Texas Hold 'Em poker [12] will be implemented. This feature has a priority of critical because it is necessary to satisfy core requirements of the system as specified by the client.	

Feature Number	**Feature**	
9	Ability to Select Type of Player Types	
Status	**Priority**	**Effort**
Approved	High	High
Risk	**Stability**	**Target Release**
Low	High	First release
Assigned To	**Rationale**	
Sarah Jabon	Gives the user the ability to designate automated players using sequential [2] or parallel computing [1]. The priority is high because it is not required for the system, but it is important for the user to interact with the application.	

Feature Number	**Feature**	
10	Ongoing Summary of API [9] Feedback	
Status	**Priority**	**Effort**
Approved	Critical	High
Risk	**Stability**	**Target Release**
Low	High	First release
Assigned To	**Rationale**	
All team members	Provides the CDP team with feedback that they can use to improve the API [9]. The priority of this feature is critical because it is one of our client's core requirements.	

Part IV
Use Interaction

3 Use Cases

Use cases describe the flow of events when a user interacts with the system.

3.1 Use Case #1: Starting the Game

Name	Starting the Game
Description	Describes how the user starts the simulated Texas Hold 'Em [12] game
Actors	A potential Microsoft [6] customer or a Microsoft [6] associate who is showcasing the application
Pre-Conditions	1. User has opened application 2. User has specified player options
Basic Flow	1. User clicks "Start" button (alternate flow 1 possible) 2. System begins to simulate automated poker game (alternate flow 2 possible)
Alternate Flows	1. User chooses to exit program (a) User clicks "Exit" button (b) System exits 2. System does not respond (a) User restarts system
Post-Conditions	The user watches the simulation take place, or the user is unable to watch the simulation because of a system failure.
Other Stakeholders	Microsoft's [6] CDP team and other people watching the computer screen for the demonstration

Systems/Subsystems The computer's hardware

Special Requirements

1. The system should not allow the user to start a simulation while a simulation is already in progress
2. The system should notify the user if there is a failure when it starts the game

3.2 Use Case #2: Setting Player Options

Name Setting Player Options

Description Describes how a user sets the options for the simulation's players

Actors A potential Microsoft [6] customer or a Microsoft [6] associate who is showcasing the application

Pre-Conditions

1. The application has initialized properly
2. Parameter menus are functioning properly

Basic Flow

1. User selects "Select Player Types" button (alternate flow 1 possible)
2. User selects the number of players
3. System prompts user to select "Parallel Player" or "Sequential Player" for each computer-based player
4. User selects the type of each computer-based player (alternate flows 1 and 4 possible)
5. User selects "OK" button to indicate a finalized selection (alternate flows 1, 3, and 4 possible)
6. System updates game and changes labels by the players appropriately (alternate flow 2 possible)
7. System returns to main game screen

Alternate Flows

1. User chooses to exit program
   (a) User clicks "Exit" button
   (b) System exits
2. System does not respond
   (a) User restarts system
3. User neglects to select one or more types of players
   (a) System displays message to user, stating that the types of players must be selected in order to continue
   (b) System returns to parameter selection menu
4. User cancels the change to player types
   (a) User clicks "Cancel" button
   (b) System returns to main game screen

Post-Conditions The types of players have changed to the user's preferences, or the types of players remain unchanged.

Other Stakeholders Microsoft's [6] CDP team and other people watching the demonstration

Systems/Subsystems The computer's hardware

Special Requirements

1. The system should not allow the user to change the types of players while a simulation is in progress
2. The system should notify the users of the player types they have selected

3.3 Use Cases and Feature Mapping

Each use case incorporates the corresponding features listed in the table.

Use Case	Related Features
#1	1, 8
#2	1, 3, 4, 7, 9

4 Storyboard

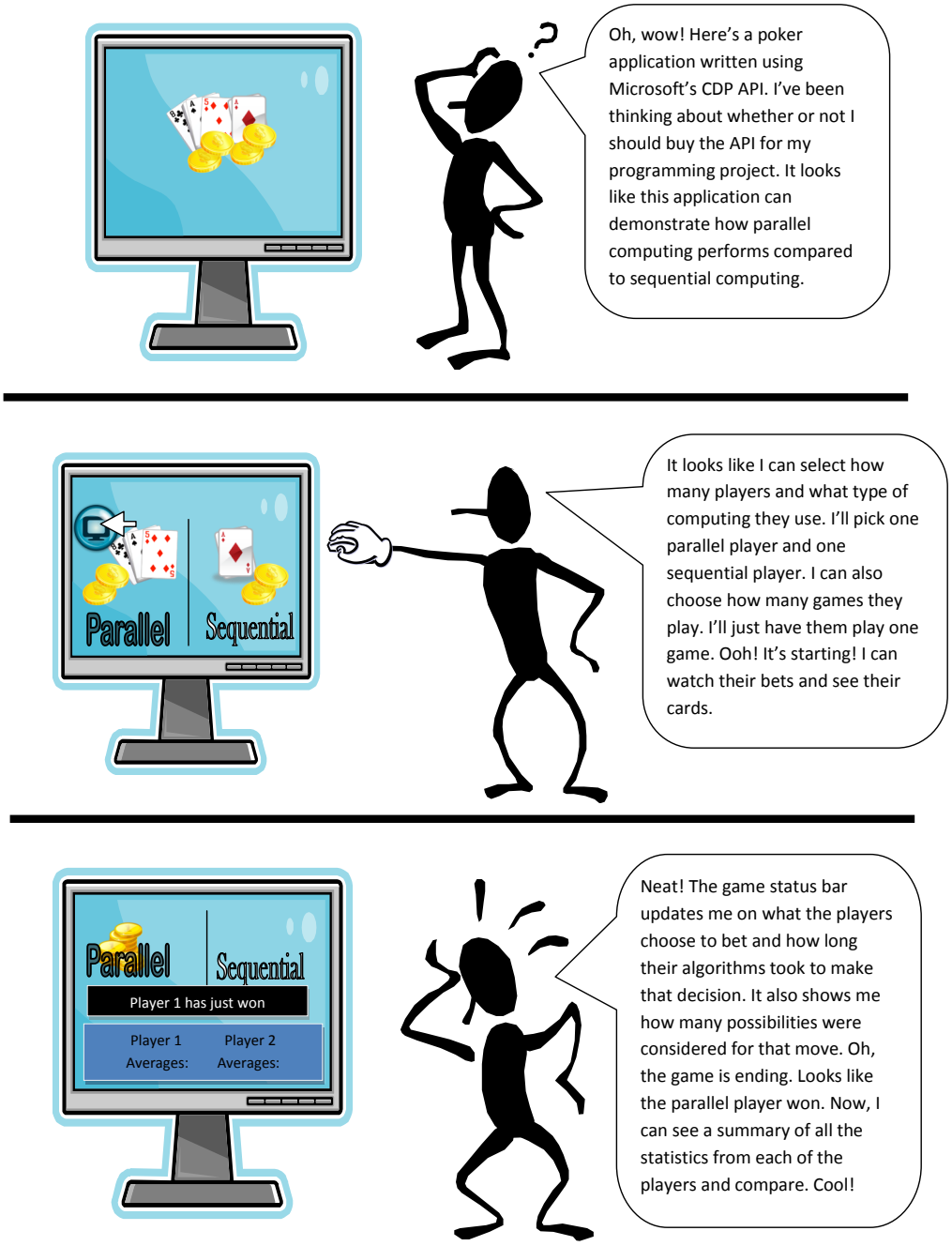

Figure 1: Storyboard

Paul was a game developer. His team was creating a brand new video game, and they in the planning process of the project. Paul had heard about parallel programming a year ago, and he thought it would be the way of programming in the future. He had taken a very basic class on parallel computing [1], but he was still learning about it. He decided to go to an event where Microsoft [6] was showcasing their new API [9]. He was selected to interact with the demonstration. Here is how his interaction went:

- Paul first read a brief description about the neural network that the players used to compute the best move. He understood from the description that the efficiency of each player would be measured by how long the player took to make a move.
- Paul then selected how many players there were going to be in the game. He chose two.
- Paul then selected which type of computing each player would use - either parallel or sequential. Paul selected the first player as parallel and the second player as sequential.
- Paul clicked the "Start" button.
- Paul watched as the players played numerous rounds of Texas Hold 'Em poker [12]. A game status bar at the bottom of the screen displayed the last action of either player and data about how long the move choice take. For instance, the game status displayed the message: "Player 1 chose to raise $50. It took Player 1 0.521 seconds to make this move."
- Paul watched the game status bar update throughout the game while he saw animations of the players' chips going into the pot.
- When the simulation ended, Paul saw summary statistics on the bottom of the screen. The parallel player's average move time was significantly less than that off the sequential player, so Paul concluded that the parallel computing [1] player was more efficient than the sequential computing [2] one.

After seeing this demonstration, Paul decided that he would talk to his development team about using parallel computing [1] in the development process for their new video game. He was very excited about Microsoft's [6] new API [9] and would definitely buy it if his team decided to use parallel computing [1].

Part V
Supplementary Specifications

5 Supplementary Specifications

The following is a comprehensive list of all system requirements. They are compiled from the features listed prior and should be sufficient to capture all requirements to which the system will be constrained.

5.1 Usability Requirements

- The system must allow the user to start the poker simulation using six clicks or fewer
- The system should use no language other than English
- The system will use standard representations of chip value and card type in Texas Hold 'Em [12] for easy recognition
- The system shall look and feel like standard Poker applications that can be found online, such as at `http://casino-games.pogo.com/games/no-limit-texas-holdem-poker#` and `http://www.pokerstars.com/`
- The system shall be simple enough that it takes a user no longer than two minutes to learn to initiate the game
- The system should display statistics about the players clearly on the screen so the user can view them easily

5.2 Functional Requirements

- The system shall follow the standard rules of Texas Hold 'Em [12] as specified at `http://www.pokerlistings.com/poker-rules-texas-holdem`
- The system must display the current game state
- The system must display a record of actions taken by the players
- The system should show animations for bets and card actions
- The system should display the cards graphically
- The system must allow the user to select the number of players
- The system must allow the user to select if each player is parallel [1] or sequential [2]
- The system will allow the user to select the number of hands to simulate
- The system must display the actions of each computer player as they occur, such as "Player 1 has folded."

5.3 Performance Requirements

- The system will complete approximately one hand per four seconds
- The system will run for a specified number of hands then stop to display statistics in the form of average runtime of each player.
- The system must not intentionally make the sequential [2] player inefficient
- The system must use an AI [10] algorithm that functions well in a parallel [1] environment

5.4 Reliability Requirements

- The system must be stable for as many rounds are required to win a game
- The system may experience no more than one error per 100 executions

5.5 Supportability Requirements

- The system must support at minimum two players

5.6 Hardware and Software Interfaces

- The system must display the differences in runtime for parallel [1] players and sequential [2] players
- The system must have a visually appealing graphical user interface

5.7 Documentation, Installation, Legal, and Licensing Requirements

- The team members must document their experiences with the CDP API [9] The system's use of .NET [4] code is under a license from Microsoft [6] for Visual Studio [7]
- The system must have all design considerations and actions documented
- The system need not include a user manual. However, it is recommended that the system have one
- the IP of the application shall be the sole property of the Rose-Hulman students developing it.

5.8 Design Constraints

- The system must implement the API's [9] provided by Microsoft's [6] .NET [4]
- The system must run on a Windows [8] platform

5.9 Training Requirements

- The system should require no training to use
- The system should have a "Help" option at the user's disposal

5.10 Deliverables

- Milestone Documents One to Five
- Final deliverable, which includes updated versions of all Milestone Documents, client comments, and lessons
- Individual engineering journals
- Reports with feedback on the API [9]

Part VI
Data Flow

6 Data Flow Diagrams

The data flow diagrams in this section describe how different data flows throughout the system. Several of these diagrams have been altered to reflect changes in system structure. This set of data flow diagrams currently represents the final system to be completed by the end of this academic year. All intended functionality is represented here with the exception of the potential for the user to be able to play along with the automated players.

6.1 Context Flow Diagram

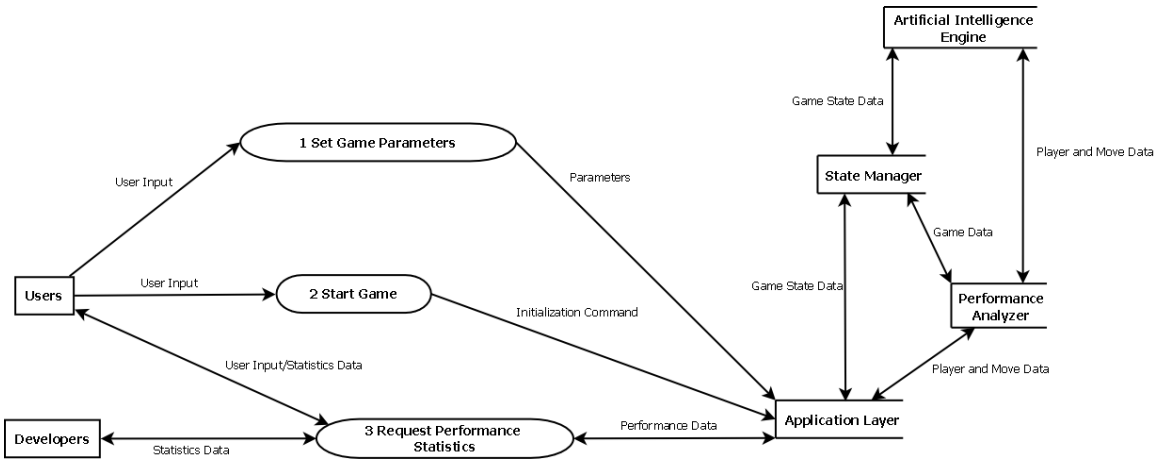

Figure 2: Context Flow Diagram

This diagram is an overview of the flow of data between the users and the back end components governing the system. The user specifies game and player parameters and then initializes the system. The application plays through the desired number of hands and provides feedback in the form of performance statistics during game play. The user can also request performance feedback for the game as a whole once it has completed. The developers will use the application to collect data about all the games, not just the current game. Since their interaction is different than the interaction of the user, they are specified separately.

6.2 Level 0: 1. Set Game Parameters

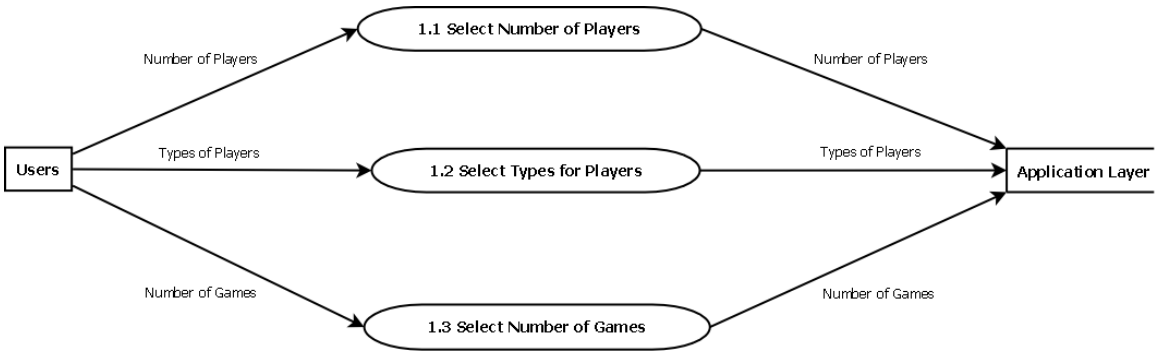

Figure 3: 1. Set Game Parameters

These three processes handle user input parameters for the creation of the game state. The user will be able to specify the number of players to play in the games, the type of each player, and the number of hands to be played. This data is passed to the application layer for game initialization.

6.3 Level 1: 1.1 Select Number of Players

The user selects the number of players. The user must indicate at least two players to play the game. This sub-process is not expanded as all it requires is the user to specify the number of players either by manually typing an integer or selecting one of them provided in the form of buttons.

6.4 Level 1: 1.2 Select Types for Players

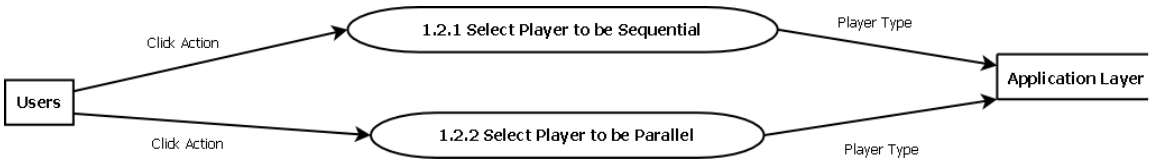

Figure 4: 1.2 Select Type of Players

The user selects the type for each automated player. The user has the choice of making each player either parallel [1] or sequential [2]. Performance statistics will be maintained for both types of players, even in the case of all sequential players.

6.5 Level 1: 1.3 Select Number of Games

The user selects the number of hands to be played. This sub-process is not expanded because it only requires the user to input an integer value representing the number of games to be played by the automated players.

6.6 Level 0: 2. Start Game

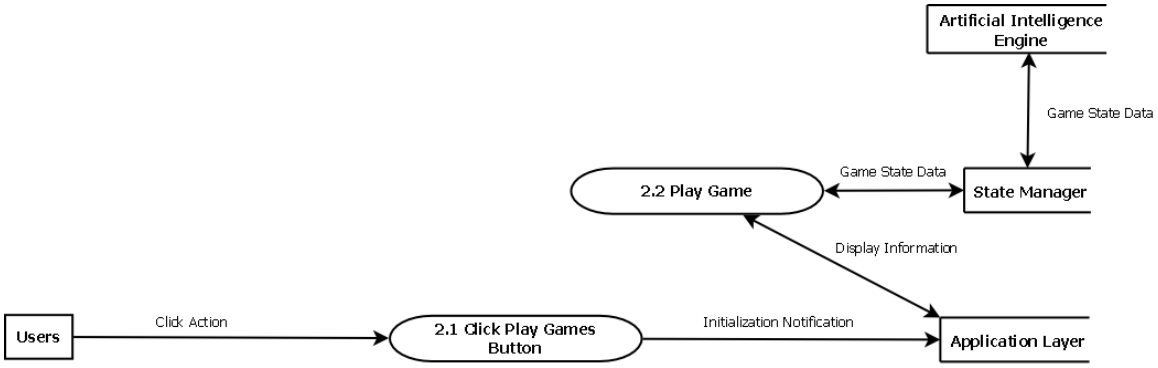

Figure 5: 2. Start Game

This diagram provides a basic overview of the user starting the application once the game parameters have been set as well as of what the system does once the user has selected start. The user is not actively involved in the playing of the game.

6.7 Level 1: 2.1 Click Play Games Button

The user will simply click a button that will cause the application to play through the designated number of poker hands with the given player parameters. This sub-process is not expanded because it only involves the user clicking a button to start the game.

6.8 Level 1: 2.2 Play Game

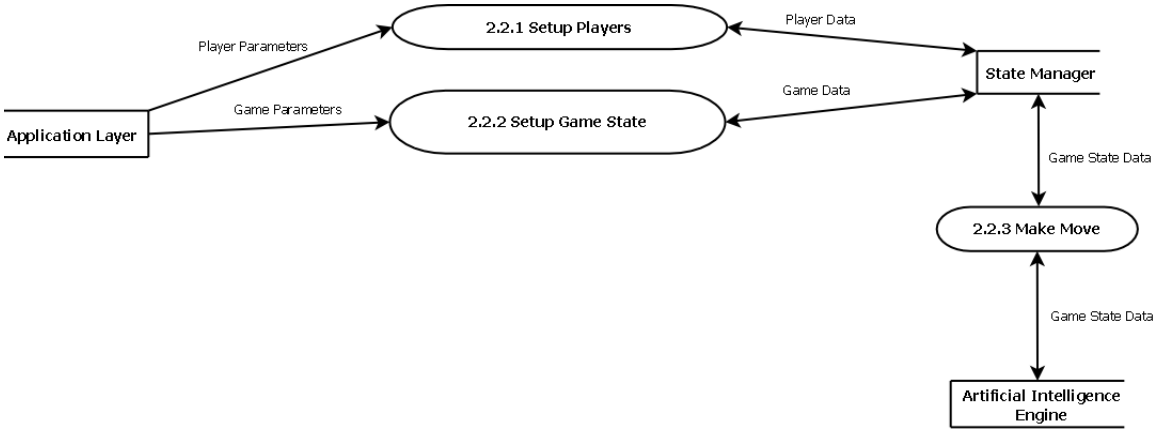

Figure 6: 2.2 Play Game

This data flow diagram reveals how the application layer, state manager, and artificial intelligence [10] engine work together within the system to initialize the game state and play through the hands.

6.9 Level 1: 2.2.1 Set Up Players

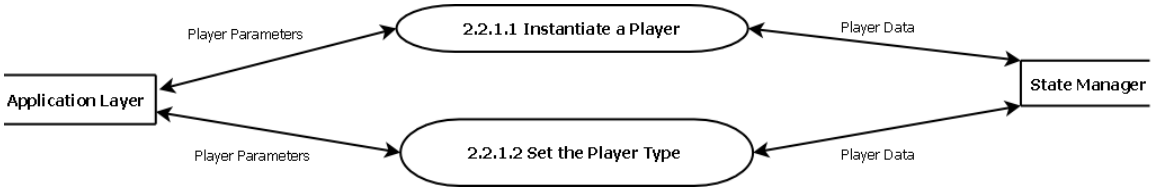

Figure 7: 2.2.1 Set Up Players

The application layer takes the player parameters input by the user and uses them to instantiate the desired number of players and give each player the user-selected type. The application layer communicates this data with the state manager component of the system.

6.10 Level 1: 2.2.1.1 Instantiate a Player

The application layer will create player instances to meet the desired number of players as specified by the user. This sub-process is not expanded because the instantiation is code functionality that will be handled through object management within the software.

6.11 Level 1: 2.2.1.1 Set the Player Type

The application layer will set each player to be sequential [2] or parallel [1] based on the types the user has selected for the players. This sub-process is not expanded because the software component governing player creation will handle setting the player type.

6.12 Level 2: 2.2.2 Set Up Game State

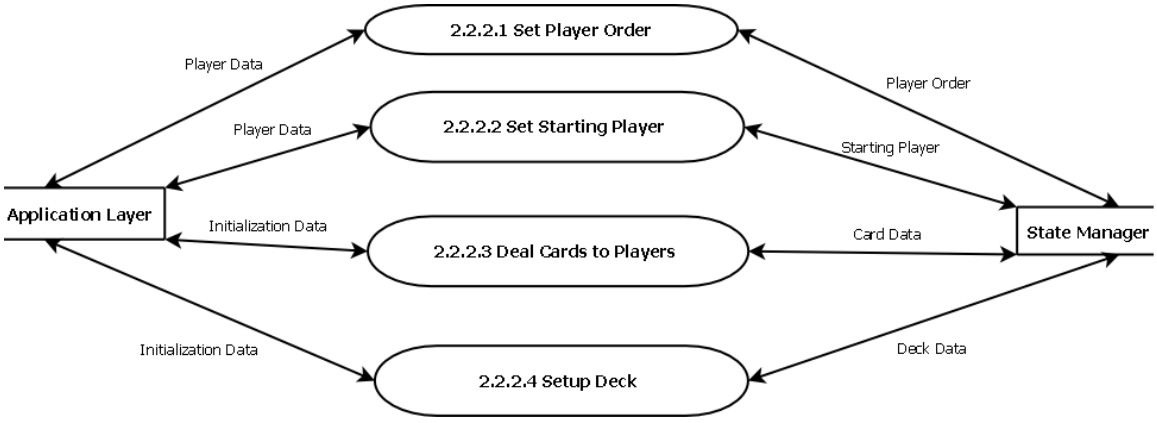

Figure 8: 2.2.2 Set Up Game State

The application layer communicates user input as well as derived parameters to the state manager in order to initialize the overall state of the game. Each of these sub-processes represents simply getting and setting data and do not need to be expanded.

6.13 Level 2: 2.2.3 Make Move

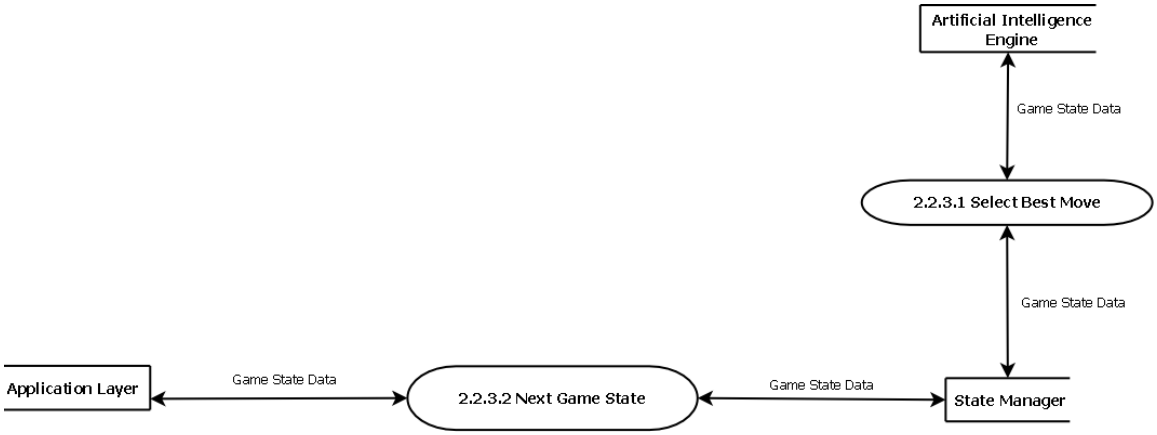

Figure 9: 2.2.3 Make Move

The artificial intelligence [10] engine will govern the reasoning exhibited by and decisions made by each automated player. This component will pull all necessary information from the state manager and then select a move to be executed. The artificial intelligence engine communicates the decision with the state manager.

6.14 Level 2: 2.2.3.1 Select Move

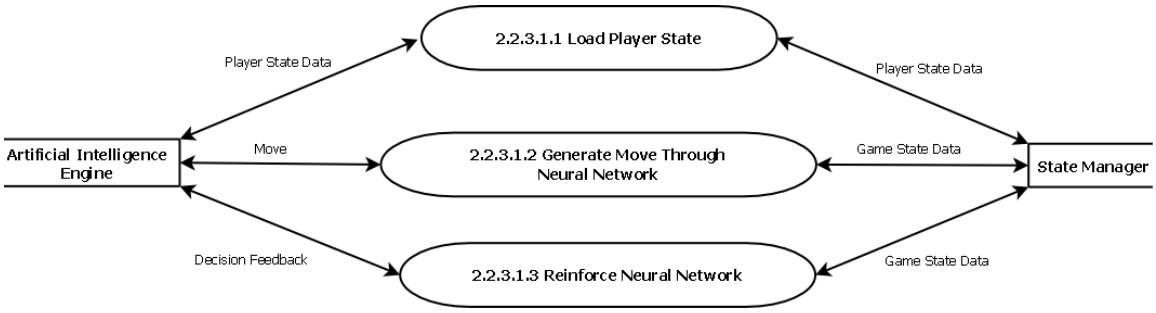

Figure 10: 2.2.3.1 Select Move

The processes shown here form the prominent function of the artificial intelligence [10] engine. Here an artificial neural network [11] generates a move to be made, followed by reinforcement being applied to the connection strengths in the network.

6.15 Level 0: 3. Request Performance Statistics

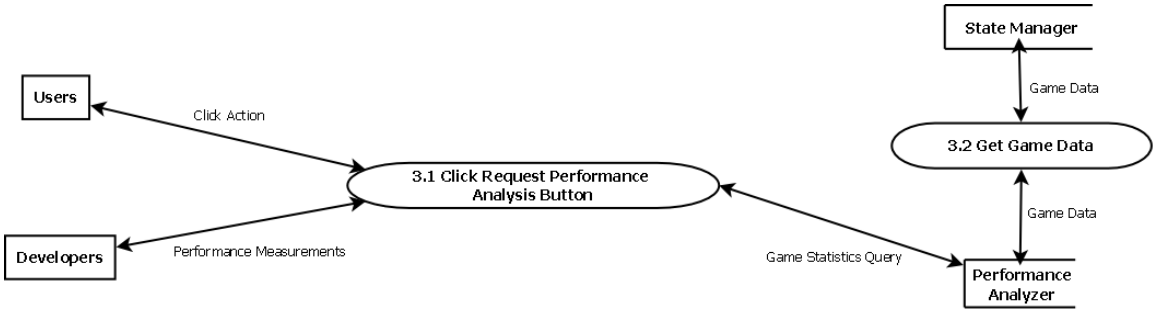

Figure 11: 3. Request Performance Statistics

This process governs the user requesting performance statistics for the game just played. Performance measurements will include data on the number of hands won by each player, the hand each player had and the amount bet, and the average time taken per move for each player.

6.16 Level 1: 3.2 Request Game Data for Performance Statistics

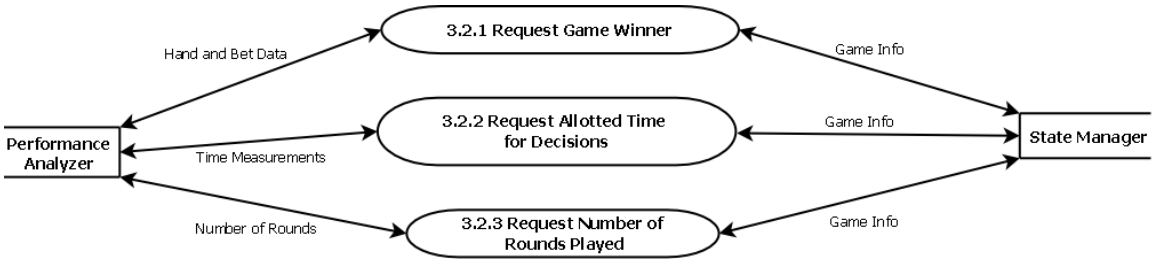

Figure 12: 3.2 Request Game Data for Performance Statistics

The performance analyzer will request state data and allotted time for each decision made in the poker game. This data will be communicated with the application layer in order to provide the user with performance statistics for the game just played.

Part VII
Appendix

7 System Features Scale Key

Status - The status of a feature pertains to the progress or current state of the feature during the planning and design stages of the system specified by the project.

Priority - Priority here refers to the importance of the development of a feature based on its significance to the overall system. Of the features listed, their priorities range from medium to critical levels. The critical features are labeled as such because they are necessary for the proper functionality intended of the application as specified by the client.

Effort - The effort attribute of a feature describes the amount of time and resources that need to be designated to the development of that feature based on the level of priority and overall importance to the structure of the system. The critical features compose the base functionality of the entire application and represent the goal of the project as a whole. All of the high priority features are integral to the proper functioning of the application as well, but are not quite of dire importance.

Risk - This attribute represents the likelihood that a feature will cause unintended results, such as cost overruns or delays. A low risk represents a lack of potential for undesirable events.

Stability - The stability of a feature reflects the probability that the feature will change or the team's understanding of the feature will change throughout the development of the feature. The stability of a feature relates to the risk that feature presents. If a feature causes unintended results, it will very likely be unstable and need to be redesigned.

Target Release - This records the intended version of the product or system in which the feature will first appear and usually reflects the priority of and effort delegated to that particular feature.

Assigned To - Refers to whom within the development team will be responsible for the implementation of the feature. Given that we have four members on this project team, each with capabilities maybe not seen in the others, the features must be distributed for development to the most capable of individuals.

Rationale - The reason or reasons for intended implementation of a feature can be required by the system, mandated by the client, or just a feature decided upon to better the user experience in using the system.

8 Glossary

- C# - an object-oriented language that can be used to create a wide variety of applications, services, and tools. [5]
- Microsoft Corporation - A multinational corporation that has five business segments: client, server and tools, online services business, Microsoft business division, and entertainment and devices division. [6]
- .NET Framework - A platform build by Microsoft that provides a common set of APIs and a consistent programming model. [4]
- Parallel Computing - using multiple resources simultaneously to solve a computation problem. [1]
- Sequential Computing - solving a computational problem with sequential processing. [2]
- Texas Hold 'Em Poker - A popular card game with multiple players. Players can use any combination of five community cards in combination with their own cards to create the best hand according to numerous rules. [12]
- Visual Studio - an integrated development environment produced by Microsoft. [7]
- Artificial Neural Network - an adaptive artificial intelligence system that can model decisions by propagating information loaded into input nodes through the network, resulting in a decision at the output nodes. [11]

References

[1] Blaise Barney. What is Parallel Computing?, 2009. https://computing.llnl.gov/tutorials/parallel_comp/#Whatis.

[2] Barney Blaise. Overview: Sequential Programming, 1995. http://www.hku.hk/cc/sp2/workshop/html/parallel-intro/ParallelIntro.html#sequential%20programming.

[3] S. Jabon, M. Jenne, I. Roberts, and B. Waters. Milestone One - The Parallel Poker Team, 2009.

[4] Microsoft. .NET Framework Overview, 2008. http://www.microsoft.com/net/Overview.aspx.

[5] Microsoft. Getting Started with Visual C#, 2009. http://msdn.microsoft.com/en-us/vcsharp/dd919145.aspx.

[6] Microsoft. Microsoft, 2009. http://www.microsoft.com/en/us/default.aspx.

[7] Microsoft. Microsoft Visual Studio, 2009. http://www.microsoft.com/visualstudio/en-us/default.mspx.

[8] Microsoft. Windows, 2009. http://www.microsoft.com/WINDOWS/.

[9] Wikipedia. Application Programming Interface, 2009. http://en.wikipedia.org/wiki/Application_programming_interface.

[10] Wikipedia. Artificial Intelligence, 2009. http://en.wikipedia.org/wiki/Artificial_intelligence.

[11] Wikipedia. Artificial Neural Networks, 2009. http://en.wikipedia.org/wiki/Artificial_neural_network.

[12] Wikipedia. Texas Hold 'Em, 2009. http://en.wikipedia.org/wiki/Texas_hold_%27em.

Index