Golf Tournament Management System

for Three Rivers Golf Association

A Manuscript

Submitted to

the Department of Computer Science

and the Faculty of the

University of Wisconsin-La Crosse

La Crosse, Wisconsin

by

Kevin J. Drewiske

in Partial Fulfillment of the

Requirements for the Degree of

Master of Software Engineering

December, 2010

Golf Tournament Management System

for Three Rivers Golf Association

By Kevin J. Drewiske

We recommend acceptance of this manuscript in partial fulfillment of this candidate's requirements for the degree of Master of Software Engineering in Computer Science. The candidate has completed the oral examination requirement of the capstone project for the degree.

_____ _____

Dr. Kasi Periyasamy — Date
Examination Committee Chairperson

_____ _____

Dr. Kenny Hunt — Date
Examination Committee Member

_____ _____

Dr. David Riley — Date
Examination Committee Member

ABSTRACT

DREWISKE, KEVIN, J., "Golf Tournament Management System for Three Rivers Golf Association", Master of Software Engineering, 10 2010, (Dr. Kasi Periyasamy).

The Three Rivers Golf Association was created in 1996 to create opportunities for competition within the greater La Crosse area that would provide a competitive atmosphere for a broad range of golfers. The information about the competition and tournaments is managed using Microsoft Excel, Microsoft Access, and two custom Visual Basic applications. None of the applications are tightly integrated to provide seamless data management. Additionally, some required functionalities are not available to the primary user, including the ability to determine season point standings or determine championship eligibility.

This document describes the development of a software system designed to assist the primary user to fulfill their duties while streamlining the entire process. The application will allow the user to maintain member information, record registrations, and manage tournament pairings and results. It will additionally determine season point standings and championship eligibility. Additional features to be added in the future include the ability to synchronize the application data with the association's website and track financial information. The addition of these features will further the single point of management the application is intended to fulfill.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my project advisor Dr. Kasi Periyasamy for his continued patience throughout the project. Secondly, I would like to thank Margaret “Squib” Olson and Jeremiah Shimshak of the Three Rivers Golf Association who may not have always fully understood the overall goal of the project, but continued to patiently answer questions regarding the process involved with executing tournaments. I would also like to express my thanks to Tom Steele, Three Rivers Technologies, for his time to assist while I learned the .NET environment. Finally, I wish to thank my parents and brothers for their encouragement and prodding during the tenure of this project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

GLOSSARY

Course / Slope Rating

Two individual values contained on all scorecards which are used to identify the relative difficulty of a course and used to calculate a player's index.

ESC – Equitable Stroke Control

Maximum number of strokes a player can take on a single hole for purposes of handicap adjustment to avoid the effects of a disaster hole. For high handicap individuals (course handicap greater than 40), this value is 10 while low handicap golfers will be limited to two strokes above par.

Gross Score

Total number of strokes taken by a player, including penalties.

Handicap

A numerical measure of an amateur golfer's playing ability calculated with a specific arithmetic formula that approximates the number of strokes above or below par a player should be. It is used to calculated a net score from the gross score, thereby allowing players of different proficiency to play against each other on somewhat equal terms. (1)

Net Score

Gross score adjusted for course handicap.

Pairings

Defined starting order for all players in a tournament including the starting time and the start order.

Starting Order

Predefined order divisions are organized when creating pairings, thereby ensuring that the player divisions are not always at the beginning or end of the tournament field.

TRGA

Three Rivers Golf Association, a regional association of golfers who compete in a series of tournaments hosted at multiple locations.

TRGA Index

A handicap specific to the TRGA using an adjusted formula using a player's five best score differentials during the past ten rounds he/she played.

1 Introduction

The Three Rivers Golf Association currently executes eight regular season tournaments along with a two-day championship. Each of these tournaments requires data management for membership, pairings, and results. The data is managed using Excel spreadsheets. In addition, a third-party application for handicapping and Microsoft Access are also used for further calculations to determine season point standings and championship eligibility. The collection of information is ultimately stored in multiple files in different formats, is managed by multiple individuals, and requires additional work to merge season point totals with registrations to setup late season pairings. Additionally, all pairings, results, and point standings are published on the association website by a separate process.

This project will create a single application to combine all necessary functionalities to manage membership, tournament registration and processing of results. It will allow data to flow seamlessly from one function to the next without any need to copy or retype that data. It will provide basic decision making to set up tournament pairings, calculate earnings and points as part of tournament results, and recalculate the player's handicap all with minimal effort and added data validation. Having these functions built-in will additionally remove the need to maintain and license a third-party application or engage another individual to assist with these functions. Moreover, it will return control to a single individual to eliminate the timing dependence that often occurs towards the end of the season when data must be merged to produce pairings.

At least one commercial package is known to exist that could provide similar functionality and would have been available immediately. It does however come with a cost that the association desired to avoid, if similar functionality can be provided which may be more closely suited to their exact needs. The association

is also aware that custom software may not be as readily supported as a commercial application.

1.1 Current Management Process

The core purpose of data management for the TRGA is to execute tournaments. As such, the first step in the entire process is to receive a paper form from a player indicating which tournaments they desire to participate. If the player is a first-time participant, the player's address, telephone, and basic golf information is first entered in a Players spreadsheet. An entry is then created in a second spreadsheet used to track tournament entries. This spreadsheet repeats name, home course, and member status columns from the first spreadsheet and contains columns for each tournament. The tournament registration columns are then filled in with an amount the player has paid for that particular tournament. The cost varies for players who are members of the course compared to non-members.

Pairings

After the registration deadline has closed, several days before the tournament date, the registration entries are used to create pairings. The tournament registration sheet is first sorted by the tournament column to group all players who registered for a tournament. The player's name, flight and index are then copied to a new spreadsheet for that tournament's pairings, where the data is resorted by flight and index. The entries in the Men's Flighted division are further divided into five subdivisions to form a total of eight flights used to organize the starting order of players.

The eight divisions are organized based upon the tournaments predefined starting order. Four different starting orders have been defined which order divisions in a manner that allows all players the opportunity to start earlier or later in the day. Such ordering ensures that the same players are never placed always

at the beginning with early starting times or at the end where the length of a round may become undesirable.

After changing the order of the individual divisions to reflect the defined starting order, the player starting order within the flight is then determined using one of three methods: (A) by random selection, (B) using player index, or (C) using player points. Player indexes are copied from the tournament entry sheet, while player points require external merging with data managed in a Microsoft Access database.

Having finally sorted all of the flights and players, players are grouped into sets of three, unless the total number of players is not a multiple of three. An adjustment is made by starting the flight pairings with one or two groups of two players. Finally, starting times for each group are assigned, typically starting at 7:30 or 8:00 A.M., depending upon the total number of players in the tournament. Successive pairing times are set nine minutes after the preceding group.

Before pairings are considered complete, a review of all players' grouping and their starting times are examined. This is done to separate players where it is desired not to play together, to honor requests from players to carpool or to obtain slightly better starting times with respect to their other commitments. These changes are made on a case-by-case basis. These pairing are then provided to the course for their portion of the tournament setup, to the website manager for posting, and to local media for inclusion in the distribution.

Results

After completion, the pairings are then copied to a fourth spreadsheet to finalize the results. Prior to the tournament, the spreadsheet can be setup to contain the points that will be assigned for each place finish, assuming no ties. These points have been established to be 18, 15, 12, 10, 8, 7, 6, 5, 4, 3, 2, and 1. A lookup spreadsheet is used to determine both the division purse and individual player payouts. The user enters the number of players in the flight, the cost of a plaque, and the corporate contribution for the division as formula inputs to

calculate the division purse. The total purse is then multiplied by the percentage shown in the column corresponding to the number of players in the flight to identify the payout for a specific place. An example of this spreadsheet is shown in Figure 13.

As the player scorecards are completed and posted, their scores are entered into the spreadsheet in the "front-back-total" format. If a player does not complete their round or has been disqualified, the total score column will be replaced with a value signifying why a score has not been posted. After all scores have been posted for a division, the results are then sorted to determine the lowest score. In the event the lowest score is shared by multiple individuals, a playoff will occur to identify the sole winner. If three or more players have tied with the same lowest score, the final results will reflect on champion with the remaining players tied for second.

The scores and results are then examined to adjust point totals and payout amounts to evenly divide those values between the players and places in which the tie exists. For example, if a three-way tie exists for second place, each player will receive 12.33 points, and the fifth place individual will receive eight points. After all flight results have been entered, the completed results are provided to the web manager for posting and to the media for distribution.

Tournament Finalization

Before the tournament can be considered complete, several additional steps need to be created. The point results are imported into a Microsoft Access database containing all the scores for the season. A custom Visual Basic application is used to find the top four point values for each player. The player's season point total is the sum of those four values. These point values are then posted on the association website.

The scorecards are also reviewed to determine the Equitable Stroke Control (ESC) value for each player. This step is taken to eliminate the effect of disaster holes and incorrectly affect the player's handicap. The ESC scores are then

entered into a third-party application used to manage the players' indexes. Each score must be entered into the software twice, as the TRGA uses a modified version of the USGA Handicap Calculation in which the best five of ten scores are used, rather than then best ten of twenty. This is done to ensure indexes are more reflective of the player's ability when a typical season only involves ten rounds.

After all scores are entered into the third-party application, the application recalculates the indexes for each player. A report is then generated, which is used to update the indexes contained on the tournament registration spreadsheet. These updated values are then used when setting up flights for the next tournament.

New players, identified as not having participated in the minimum three tournaments necessary for index calculation will be placed in a flight based upon their current USGA index. A player's USGA index may also be used at the discretion of the tournament committee.

At the conclusion of the regular season, a two-day championship tournament is played. In order to be eligible for the tournament, players must have participated in a minimum of four tournaments and either have an automatic qualification from participating in all events or by winning one or more tournaments. This lookup is done using a second Visual Basic application against the same Microsoft Access file.

The pairings for the first day of the championship are setup in the same process as a regular tournament. The scores for day one are entered into the spreadsheet and are used to create the pairings for the next day, with players starting in order of their finish, starting with the highest score.

Results are processed in a manner similar to a regular event, with the exception that earned points are doubled from their normal value and the points are added to the total calculated regular season point standings.

1.2 Automated Application Process

The previous section explained the details involved in the data management process for the Golf Tournament system. The current process uses several different files in different formats. It is therefore evident to consolidate all the files and the databases into one single application. For example, the usage of spreadsheets to maintain data and management tournaments is a repetitive process that can be automated and better organized to increase the efficiency, effectiveness and accuracy of these processes. In creating an application to automate this process, the core actions performed by the user will remain generally the same. One major benefit is that no data will be duplicated, and the necessary data will seamlessly flow from one area to the next, precluding the user's need to perform any copy operations.

The application will remain focused on the player data. The user will either add or select an existing player from a searchable list, which will open a new window showing all details of the player. The user will be able to review all current information and update if necessary. The user will also be able to review and update registration information for a particular tournament.

A tournament setup dialog has been created for the user to define and review information associated with executing the tournament results. This setup dialog contains a spot to record the total corporate contribution for the purse, the cost of award plaques, as well as an opportunity to review course rating information necessary to calculate net scores.

After all tournament entries have been entered, the user will start a multi-step process to create pairings for the selected tournament. The first screen will allow the user to verify and change the selection of the index to be used for the tournament, if there is a need to override the pre-selected item. The next step will present the user with a predetermined flight breakdown. The user will have the ability to change the number of players in the flight for evenness and equability in the event the split between player indexes wasn't the best choice. After entering

the desired starting time and pairing setup method, the application will automatically assign tee times for all groups. Lastly, the user has the ability to change the order of players within their assigned flight, before finalizing the pairings.

On event day, the user will use a single dialog to process all result. The dialog shows one flight at a time, in which the user will enter a score for the front and back. The application will then calculate the gross score and if applicable, the net score. In the event the user did not satisfactorily complete the event, the user will select a value from a combo box in a Miscellaneous Score column to define the reason. Upon entering all scores, the user will initiate an action to calculate results. Prior to result determination, the application will validate all entered data to ensure (1) all necessary fields have been completed and (2) two or more players do not share the same lowest score.

After validation of the data, the application will determine the total purse for the particular division, as it is based upon both the total number of players in the tournament, but also the individual flight. Having calculated the total purse, the application will then look up the appropriate percentage payouts for each place. It will then examine each result, assigning the appropriate placement, points earned, and percentage of the purse. This information will be shown on the screen and provided in a report. Finally, the application will automatically recalculate the player's season point totals using the newly earned points.

Finally, after all results have been posted, the user will use the final dialog to enter in the ESC score. The dialog will default to the player's gross score, requiring the user to change the information, only if it an adjustment has been made. The final step the user takes is to confirm all changes, which causes the application to recalculate all player indexes, which will be used when setting up pairings for the next tournament.

Handling of the championship will occur in multiple parts, though very similar to the regular season tournaments. The logic of the existing Visual Basic

application will be moved internal to the application and modified to produce a report which will show both players that are eligible as well as alternates. This same list will be available to quickly register players for the championship, preventing the user from editing each player individually.

The user will enter results using the same dialog as the regular season. It will be aware if it's day one or day two of the tournament and adjust accordingly to either create pairings for the next day, or process the results. Finally, it will also update points appropriate to continue using the same season point standing report to show the final season point standings used to determine player of the year.

2 Software Development Process

A software development process is a sequence of well-planned activities. There are many types of processes describing the tasks and activities that must occur during the process, from the gathering of requirements to implementation and testing, before deployment and maintenance. Each defined process attempts to achieve a goal to identify a repeatable, predictable process which improves productivity and quality (2).

The Waterfall Model requires each phase of the project to be completed prior to moving to the next phase. This model discourages revisiting the previous phases, therefore making it great for projects where the requirements are clearly defined and known at the beginning of the project. Projects where the requirements are less defined may be more suited to an interactive or agile model. A variety of models are available, all based upon smaller iterations allowing a customer to review the current state and refine requirements before moving on to the next phase in the project.

The golf management system was developed using the iterative model. This model was chosen, as it allowed the application to be developed in smaller chunks associated with key areas of functionality – membership, registration, pairing creation, results, etc. In addition to limiting the focus to a smaller area, it was also used to divide the testing and validation process into smaller parts which again would help flush out issues before they were carried forward.

2.1 Requirements Gathering

The TRGA Management application was developed as an automation project replacing a manual process. In order to automate the process, the developer had to (1) understand the existing workflow from the end-user perspective, including rules and criteria of the domain and (2) more clearly understand the data as it pertains to the domain.

For this project, the sponsor served as the primary application domain resource. Personal involvement with the association served as a secondary resource, leaning on information obtained from assistance provided in previous seasons.

The first step was to meet with the user to walk through their process of registering a player for a tournament, creating tournament pairings, executing and calculating tournament results, before finally updating the system to prepare for the next tournament. At that time, existing data files were provided by the sponsor for additional review. Upon that discussion and further review of the process, the functionalities were identified and grouped into four primary categories: (1) member management, (2) tournament execution, (3) core data maintenance, and (4) championship management. These functionalities are described in Table 1 through Table 4.

A second phase to create high level requirements and a design was then completed prior to product implementation using an incremental development process along with agile concepts. Throughout the iterative development process, communication with the sponsor continued to obtain and refine requirements.

	Name	Description
1	Add / Update Member	Create or update basic member information
2	Define Home Course	Manage course(s) a player designates as their home course. Players typically only have one designed home course, but in some instances have two when they are members at multiple courses.
3	Tournament Registration	Add or remove registration for upcoming tournaments.
	Tournament History	Review previous registration from current and prior years.

Table 1 - Member Management

	Name	Description
1	Tournament Setup	Define or update information for a specific tournament including purse and course/slope rating.

2	Create Pairings	Define pairings for a single tournament
3	Enter Results	Enter player scores upon the completion of their round, and calculate results (place, points, earnings)
4	Adjust Handicap	Allow user to update players gross score for ESC used to calculate handicap.

Table 2 - Tournament Execution

	Name	Description
1	Tournament Maintenance	Add or edit tournaments
2	Division Maintenance	Add or edit player divisions
3	Course Maintenance	Add or edit courses hosting tournaments and identified as home courses by players
4	Starting Order Maintenance	Add or editing starting order designations

Table 3 – Core Data Maintenance

	Name	Description
1	Season Point Calculation	Calculate season point total from top four earned point values
2	Championship Eligibility	Identify player eligibility for championship

Table 4 - Championship Functions

During the discovery process, a review of all data stores was performed to identify a data model and corresponding database schema. It was quickly determined none of the three existing systems contained any data which could be used to link between the systems.

The analysis did examine:

- What are the primary objects (tables)?
- Do any primary keys exist in the current data?
- How are the primary objects associated with each other (foreign keys or cross-references)?
- What can and cannot be normalized?

The analysis revealed that while no true relations exist in the current data, relationships did exist, but would require some data manipulation. The relationships will be able to be created, but will likely require temporary tables and transformations in order to migrate the existing data into the new schema.

2.2 Functional Requirements

The following functional requirements were identified to develop the application with functionality sufficient to manage golf tournaments. They have been broken into their respective sections for organization.

Administrative Data

- Course
  - Add Course
  - Edit Course
- Division
  - Add Division
  - Edit Division
- Starting Order
  - Add Starting Order
  - Edit Starting Order
- Tournament Schedule
  - Add Season
  - Add Tournament
  - Edit Tournament

Player Management

- Player
  - Add new player
  - Edit existing player
  - Search for player by Name and Division
- Player Course
  - Add new player course
  - Change existing player course
  - Delete player course
- Tournament Registration

- Register player for tournament
- Change player division for tournament
- Unregister player from tournament
- View prior registrations
- Register player after pairing creation
- Remove player after pairing creation

Tournament Execution

- Setup
  - Define Corporate Contribution and Plaque Cost
  - Define individual flight purse adjustments
- Pairings
  - Review and identify player index to use
  - Adjust flight entry splits
  - Define pairing method, starting time and interval
  - Adjust player order within flight
  - Print / export pairings
- Results
  - Enter Front, Back or Miscellaneous Score
    - Calculate Gross Score
    - Calculate Net Score (if specified)
      - Calculate Player Handicap
  - Verify results
    - All results entered
    - No ties for first place
      - Playoff winner identified with checkbox
  - Calculate Results
    - Points: 18, 15, 12, 10, 8, 7, 6, 5, 4, 3, 2, 1
    - Earnings
      - Flight Purse

$$\left(\frac{Corporate\ Contribution}{Total\ Players} \times Players\ in\ Flight\right) + (Per\ Player\ Contribution \times Players\ in\ Flight) - Plaque\ Cost$$

      - Player Earnings (see Figure 13)

$$Total\ Flight\ Purse \times Placement\ Percentage$$

    - Print / export results
    - Recalculate Season Point totals
- Post Event Cleanup
  - Apply Equitable Stroke Control (ESC) to scores
  - Recalculate Player Index using TRGA index formula based on USGA index formula

Reports

- Pairings
- Results, By Flight
- Results, Entire Field
- Season Point Standings
- Championship Eligibility

Other

- Championship Eligibility determination
    - Field limited to 120 players
    - Completed minimum of 4 events
    - Automatic qualifier
        - Won one or more events
        - Participated in all regular season events
    - Points qualification
        - Fill remaining field using remaining eligible players in descending order of season points

Once the requirements were identified, analyzed and confirmed with the sponsor, the developer proceeded with the application design.

3 Design

The application will be used by a single user who will perform a distinct set of tasks which are generally disconnected from each other. Each task has a particular timeframe with respect to the overall season or a single tournament. A set of use cases was identified starting with an examination of the processes the user performs when managing the data necessary for tournament execution. Four core areas were identified – administration, player, tournament, and championship. These areas were further subdivided into smaller components as shown the high-level use case diagram in Figure 1, with each of the display use cases being able to broken into smaller use cases.

The use case for creating pairings is shown in Figure 2. When the process is initiated, the first step requires data to be retrieved from the database. Wizard-like functionality will then guide the user through the multi-step pairing process before the user examines the final information before the data is output and stored in the database for use during the remainder of the tournament.

The majority of the administration data is likely to be a one-time setup with minimal changes thereafter. In the spring, the tournament schedule will be defined and likely will not be changed until the following season. Individual components within the Manage Administrative Data use case include defining courses, defining divisions, defining starting orders, and defining the tournament schedule.

Throughout the season, player and registration data will continually be accessed as players register for upcoming tournaments. At that time, changes may be made to their personal and golf information.

Tournament specific functionality will be executed in a pipe-line fashion process – create pairings, enter results, and update handicap before repeating for the next tournament. There is only one order in which the majority of a tournament can be processed as later steps are dependent upon earlier steps.

Towards the end of the regular season, the user will generate a report which determines championship eligibility. The report results are not official until the

final regular season event is completed. However, it is possible to use the report prior to the end of the regular season to get a preview of likely eligible players.

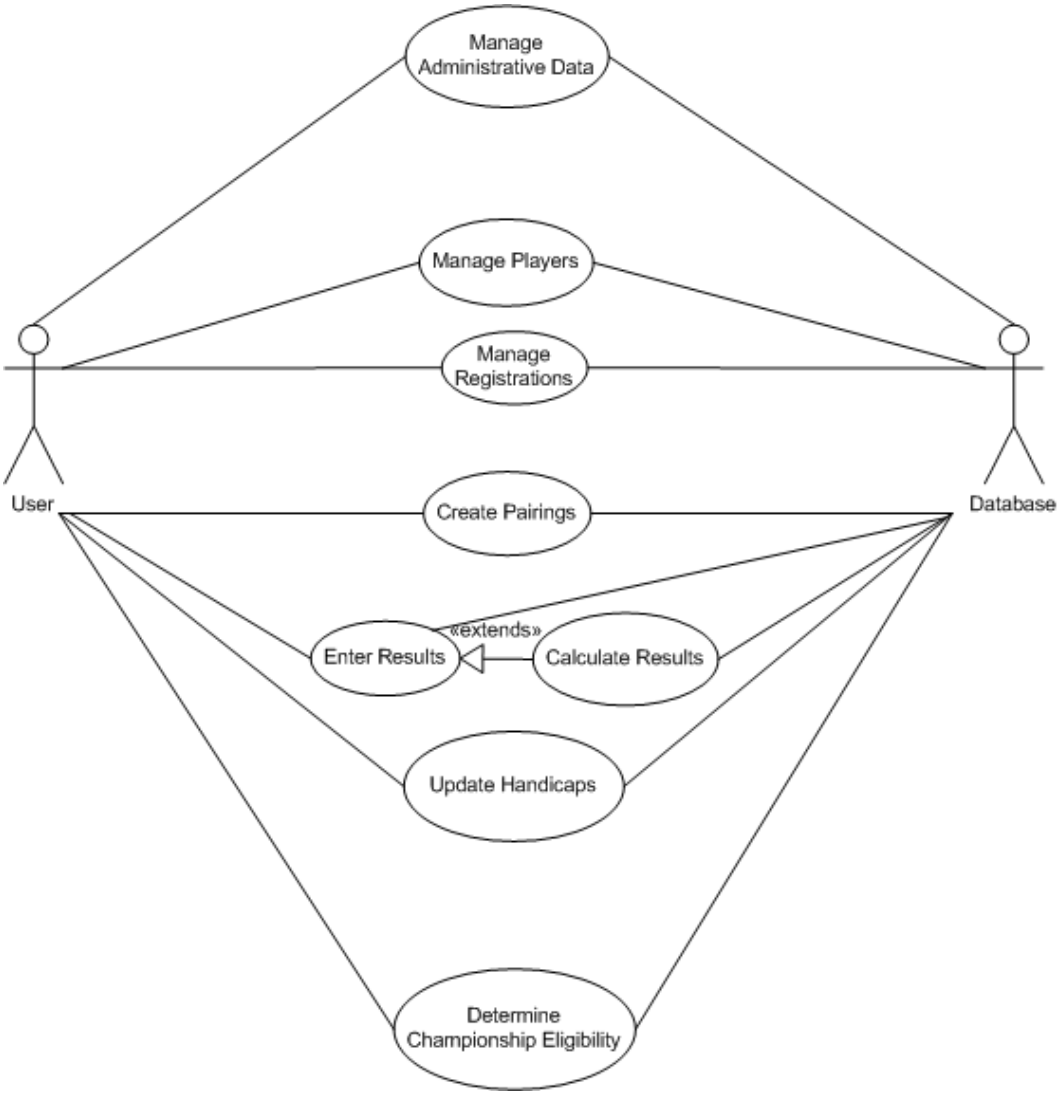

Figure 1 – High-Level Use Cases for the TRGA management system

Figure 2 - Create Pairings Use Case

The class architecture for the application is based very tightly on the database schema where all application data is stored. Several association classes were created to manage the relationships contained in cross-reference tables in the database, while several additional associations were defined which link two specific objects. An associated class, PairingSetup, was created to manage additional data properties normally stored elsewhere in the data, notably the player's USGA index and current TRGA index.

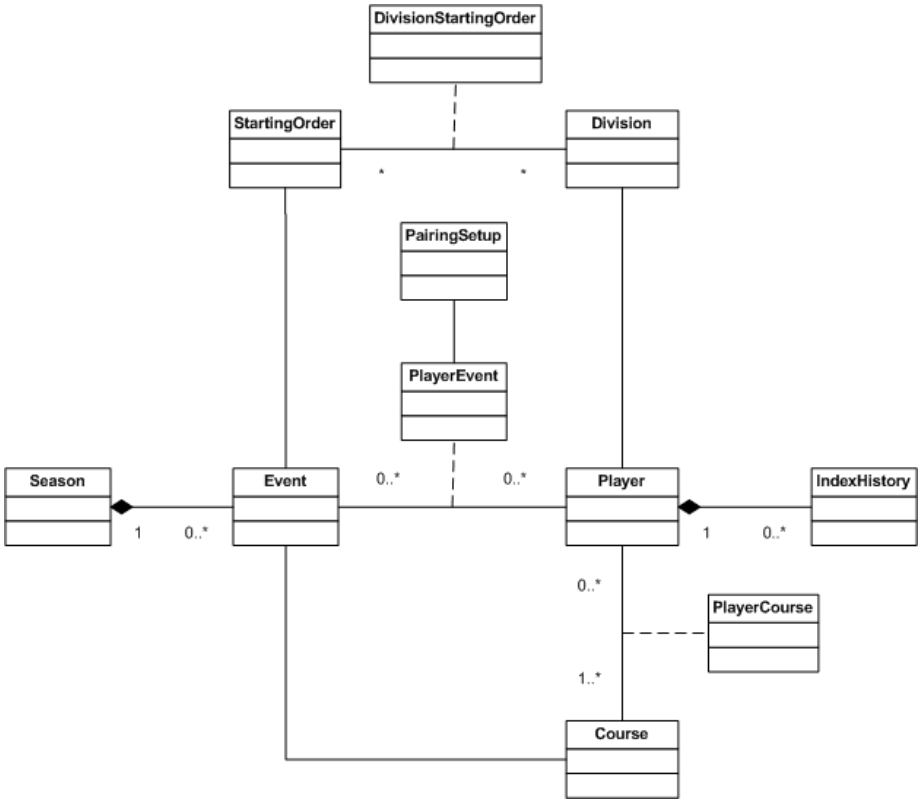

Figure 3 - Class Diagram for the TRGA management system

The entire application is designed to manage player entries in a tournament. Each step in the tournament process is conditional on a prior step being completed. The sequence diagram shown in Figure 4 **Error! Reference source not found.**describes the process flow for creating pairings upon the completion of registration entries. Additional sequences exist within tournament processing with less complexity. Since the other sequence diagrams are very similar, they are not included in the report.

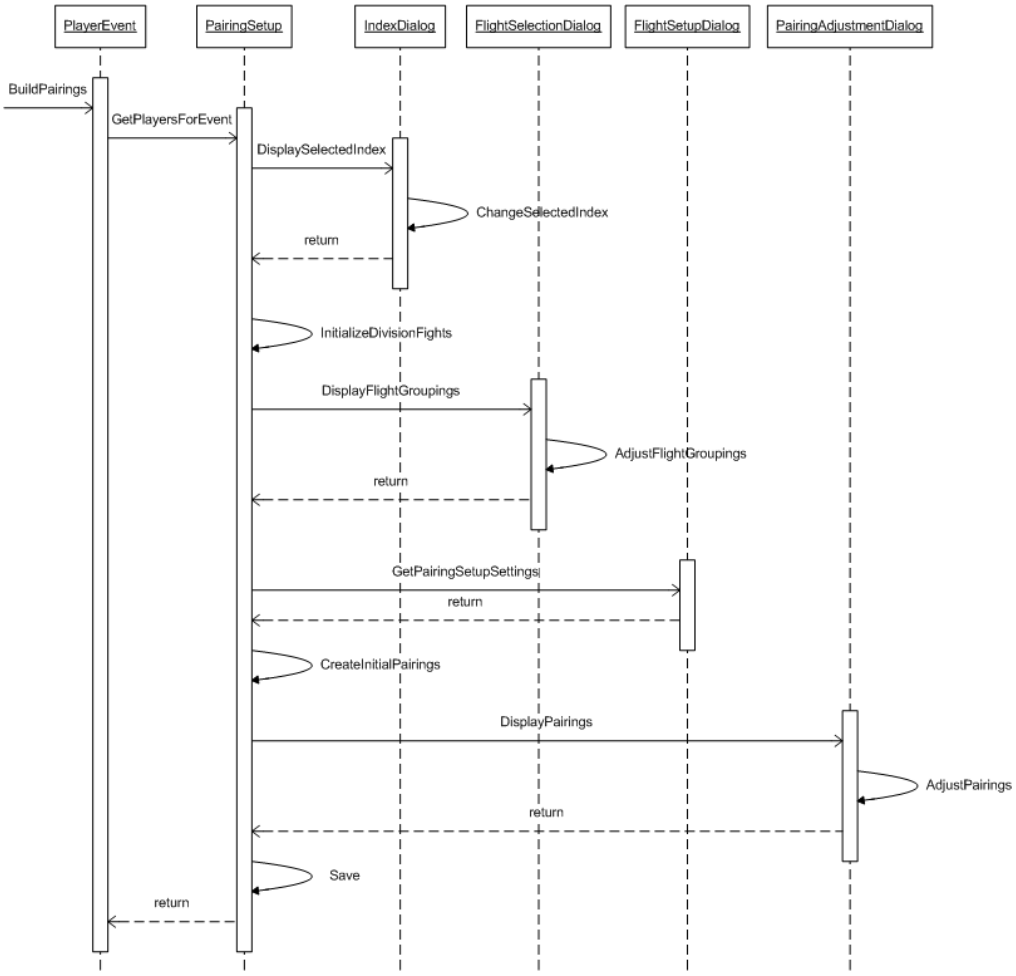

Figure 4 - Sequence Diagram for Pairing Creation

3.1 Database Design

The design of the database was a critical component of the application design. In the original analysis of the existing data, several core data objects were identified. These objects included (1) Player, (2) Event, (3) Course, (4) Division, and (5) Starting Order. Additional objects where then identified where many-to-many relationships would exist to link the data including (6) Player Event, (7) Player Course, and (8) Division Starting Order. Lastly two additional objects where defined to organize (9) Seasons and maintain history of (10) player handicaps without needing to recalculate each time a value was requested.

Having previously identified the core and secondary objects and their relationships, the design process mostly consisted in determining which properties would reside in each object. All data was normalized as much as possible, which does have a direct impact in situations where a course or player name was changed.

A decision was made not to create domain lookup tables for gender or scoring methods (gross or net). It was felt that the additional overhead required to provide self-documenting data was unnecessary and the comments in code, data, and schemas could provide sufficient identification.

Upon creating all columns necessary to import existing data and provide key values for relations, additional columns were identified that were essential for application operations. Slope and course ratings were added for the three different sets of tees used (Women's, Men's, and Championship) which would be used to calculate a player's course handicap used for Net scoring, and to calculate a scoring index differential necessary to update a players index.

The Player Event table was created to handle everything associated with a player's participation in a tournament. Initial thoughts were to separate it into separate tables for pairings and results, but since these two tables would essentially have a one-to-one relationship, the separation idea was eliminated. The lone consequence of this decision is that pairing and results will always contain the same players, even if a player was originally registered and did not participate, or an unregistered player did participate. During development, it was determined that a column for starting order would be necessary to retain the proper order of players within a pairing regardless of database default ordering. Additionally, the result place field was added for future enhancement of the website to show the player's placement for a specific tournament.

Finally, a table was created to retain the history of a player's index. At the current time, the sponsor of the application is only interested in the most current calculated value. The value could have been included in the Player Event table,

but the value may not change for each event, as an index is determined from the best five differentials out of the last ten. The resulting schema designed is shown in Figure 5.

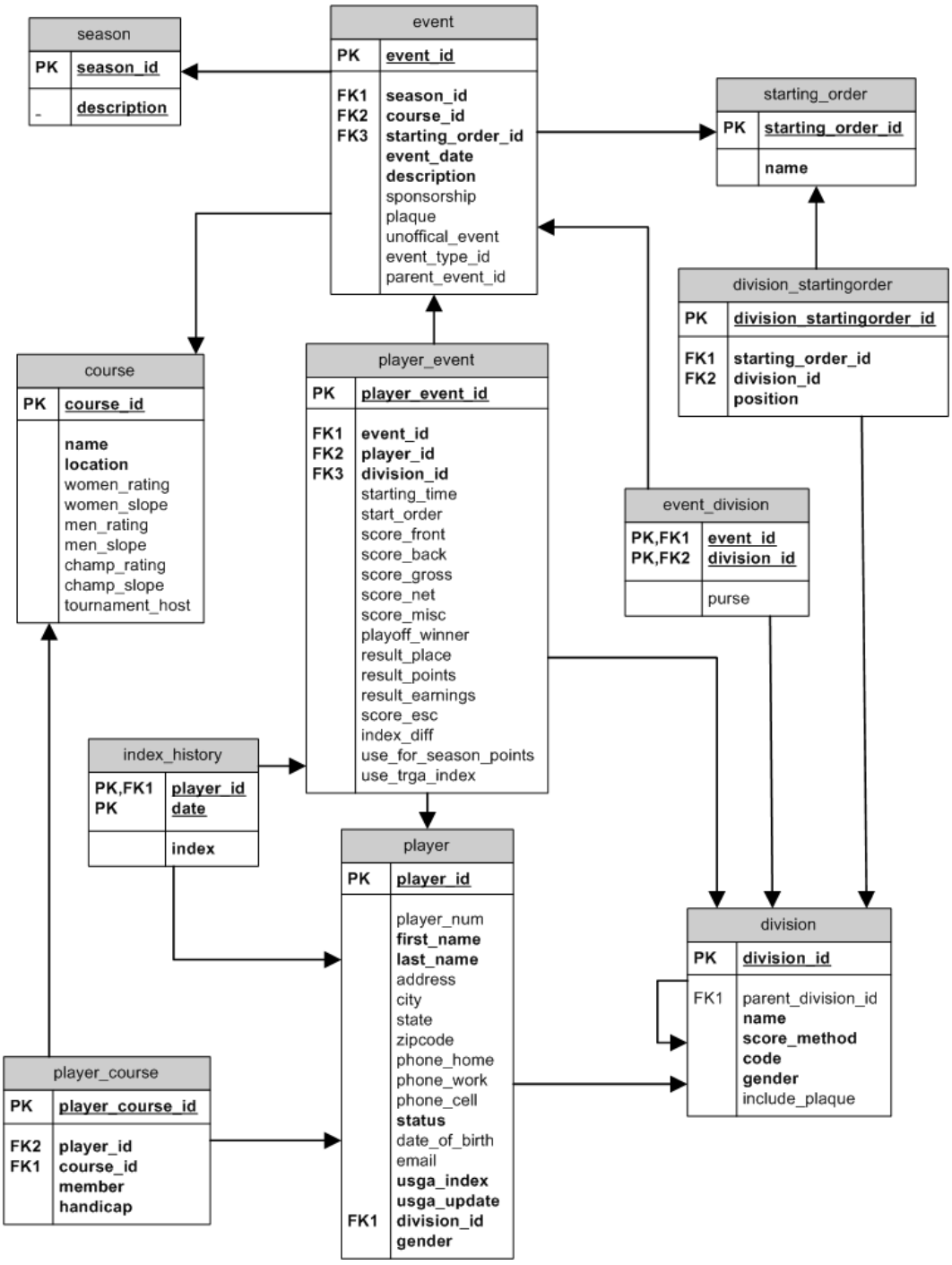

Figure 5 - Database Diagram of TRGA Management application

4 Implementation

Development of the application was completed using an iterative prototyping model. The database was first constructed, as many design decisions were based upon the structure of the database. Work on the application was next started with a focus on the general architecture of the application, which also included the initial administration data. Having core administration data available, the next phase was to add player management. From these initial phases, there was sufficient data in the application to create actual data to implement the tournament process starting with pairings, before moving onto results and post-tournament processes. The final step in creating the application resulted in adding functionality to execute the unique rules specific for the championship.

Using the prototyping process allowed the application to be built in a logical sequence with the needed functionality. Additionally, functionality that had been previously completed was used and tested while working on the next set of functionality. It provided an opportunity to validate the usability of the functionality.

4.1 Data Migration

Application development started with an empty database and populated data using the interfaces provided to manage administrative data and player data. Initially this worked well as it was thought that real data could be used as examples for test cases. However, over time, the volume of data necessary to be entered to properly handle the execution of tournaments became immense. Each tournament is comprised of a field generally between 90 and 140 players, with the total number of players per season being nearly 200. Additionally, a key component of the application is to calculate player indexes after each tournament. This value requires knowledge of the last ten tournaments a player has participated, which could require multiple seasons of data to be entered.

Therefore, it became necessary to import existing data. Having electronic records for results dating back to 2002, it was decided that the initial import of history would revert to 2002; prior history may be imported in the future.

Having three distinct, but related storage systems proved to be a challenging task. It was originally intended that a migration would be performed at the beginning of the project and would be re-executed at the end to get the most current information imported. However, the lack of consistent key values changed the direction to keep the original import and load all new data using the application - both to test the application using real data and also to help refine usability.

Since SQL Server Compact Edition doesn't support usage of the Import and Export Wizards provided in the SQL Server Management Studio (SSMS), an Express Edition database was created to perform all initial loads and data cleansing operations.

Using the Import Wizard, all data was loaded into temporary tables. Basic tables were loaded using a simple SQL statement to insert all records without transformation. Tables such as player information required more complex SQL statements to parse compound fields into their individual components. Examples of these compound fields include name and the non-street portion of an address.

Having populated core tables, the next set of data consisting of player scoring history was imported. Since scoring data originated in the MS Access database, it didn't have a perfect correlation to the player data already imported. Therefore, even after writing a query which would parse the name, there were still instances where the name didn't exactly match. These records were continually processed until the query matched all records, at which time the data was sufficiently matched to load into the final table. The loading of result data achieved the primary goal of providing scoring history since 2002, but did not serve to populate any pairing history. This data was not necessary and since it is not conveniently available for import, may never be populated.

The final major piece of data necessary to load was the ESC scores from the handicap system. Since it was a third-party application, the name was the only value that could possibly be matched, though it also wasn't consistent with the previous tables. Following a similar process as score history, the handicap data was merged with the existing player result data.

After all data was imported, several additional statements were executed to initialize data that was essential to the operation of the application, including player gender, default division, and score differential. The score differential itself isn't valuable, but is the primary component of the index calculation process.

Finally, after all initial data has been imported and cleansing was deemed complete, the Express database was exported to a SQLCE database using SQL Server to SQL Server Compact Edition Database Copy (3). Further refinements to the data were performed using SSMS.

The primary challenge encountered during the data migration process was simply trying to mesh data together which did not contain a consistent key field. Therefore, it was necessary to manipulate name fields to create a key. Despite using multiple methods, errors still occurred which were not discovered until later in the process. Additionally, data errors and omissions within the original source were only identified through the testing of other processes.

4.2 Technologies Used

The TRGA Golf Management application will be used by a single user at golf courses that do not have reliable internet connections. Since it is imperative that scores and results be processed immediately upon the conclusion of each flight, it was essential that the application be a thick client. The computer used currently runs the Windows Vista operating system, allowing for all current Microsoft technologies to be used.

The application was created as a Windows Forms application using .NET Framework 3.5. Microsoft SQL Server Compact Edition (SQLCE) was chosen

for the data store of this particular application. SQLCE was chosen over the Express Edition since it did not require an additional installation and the additional features provided by Express Edition, namely stored procedure support, were not deemed necessary to the application.

Reporting for the application is handled using Microsoft .NET's RDLC reporting technology. Like Crystal Reports, the format can be defined using Visual Studio and can be viewed using controls provided by .NET. In addition to showing the report in the application prior to printing, it can also be exported to Adobe's PDF format or exported to a Microsoft Excel file.

4.3 Application Prototype 1 – Architecture & Administration

Having chosen to create the application as a Windows Forms Application, it was decided that the user interface for the application would be centered on a single dialog and would not use a menu system to reach the various components. Rather, a tab control would be the primary navigation between the three major areas, with additional links used to access the remaining functionality. The resulting dialog used for system administration is shown in Figure 6. After the initial configuration of data, it is expected that this screen will be minimally used beyond the annual setup of tournaments, as divisions and starting orders are generally fixed and most courses within the region have already been added.

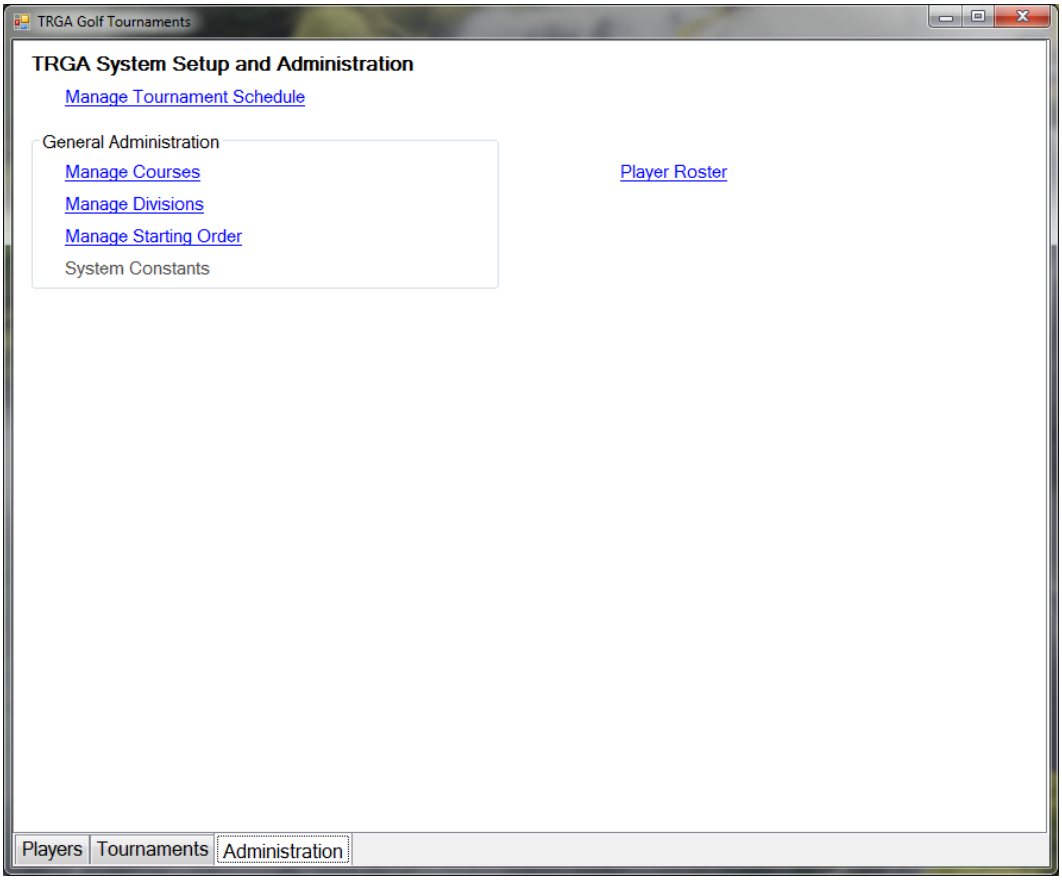

Figure 6 - Application Data Administration

The Event Management dialog (Figure 7) is the primary administrative dialog in the application, as events are the only administrative data that will be updated each season. The modal dialog is mapped to a Binding Data Source, which neatly combines the DataGrid showing a summary of all tournaments with the detail shown on the right side. Changing the selected row automatically updates the individual detail properties. Additionally, the combo boxes for course and flight order are populated using Binding Data Sources.

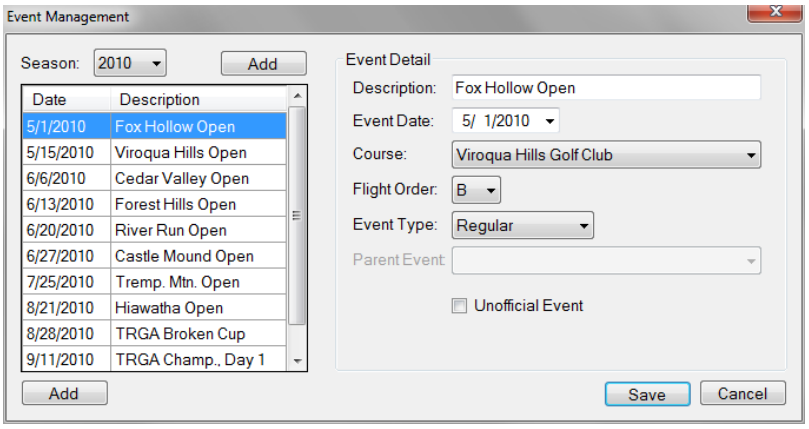

Figure 7 - Event Management

The remaining administrative dialogs, shown in the Appendix, are all developed in a similar manner using a modal dialog and Binding Data Sources. The Starting Order Setup (Figure 16) is somewhat unique, as it is the only dialog that does not contain any controls that allow the user to change values. A new starting order is created by clicking the Add button, which displays a new dialog prompt the user for the starting order label. The application will then populate the right-side list in flight-level order. The flights can then be reordered by dragging and dropping a specific row. The application will also check to ensure the number of flights contained in the right side match the total number of divisions stored in the system. If the number does not match, the application will prompt the user to reset the flight list before continuing.

Since the development of the TRGA application was the developer's first significant experience using Windows Forms technologies, each control seemed to provide its own challenge, some of which were easy to overcome, while others were more challenging. For instance, once a ComboBox received focus, the focus could not be moved to any other control. This problem was ultimately overcome by changing the properties used to associate the bound item with the appropriate control property.

Rather than employing buttons to manage up-down navigation when reordering divisions within a starting order set, it was decided to create a drag-and-drop method within a data grid. This process was undertaken in part because of its planned usage for pairing adjustments later on in the application development. In addition to learning more about the behavior of Binding Sources and associated actions, it also provided an opportunity to work with custom drawing actions and a solid understanding on the differences between moving a row up as compared to down.

4.4 Application Prototype 2 – Player Management

The second prototype focused on managing players using two different screens. The first screen is a tab on the main application listing all players. The grid can be sorted by clicking on the column headers and filtered by name and division. The individual player dialog is displayed when adding a new player, or editing an existing player when selecting a player in the grid. A screenshot of the player management is shown in Figure 8.

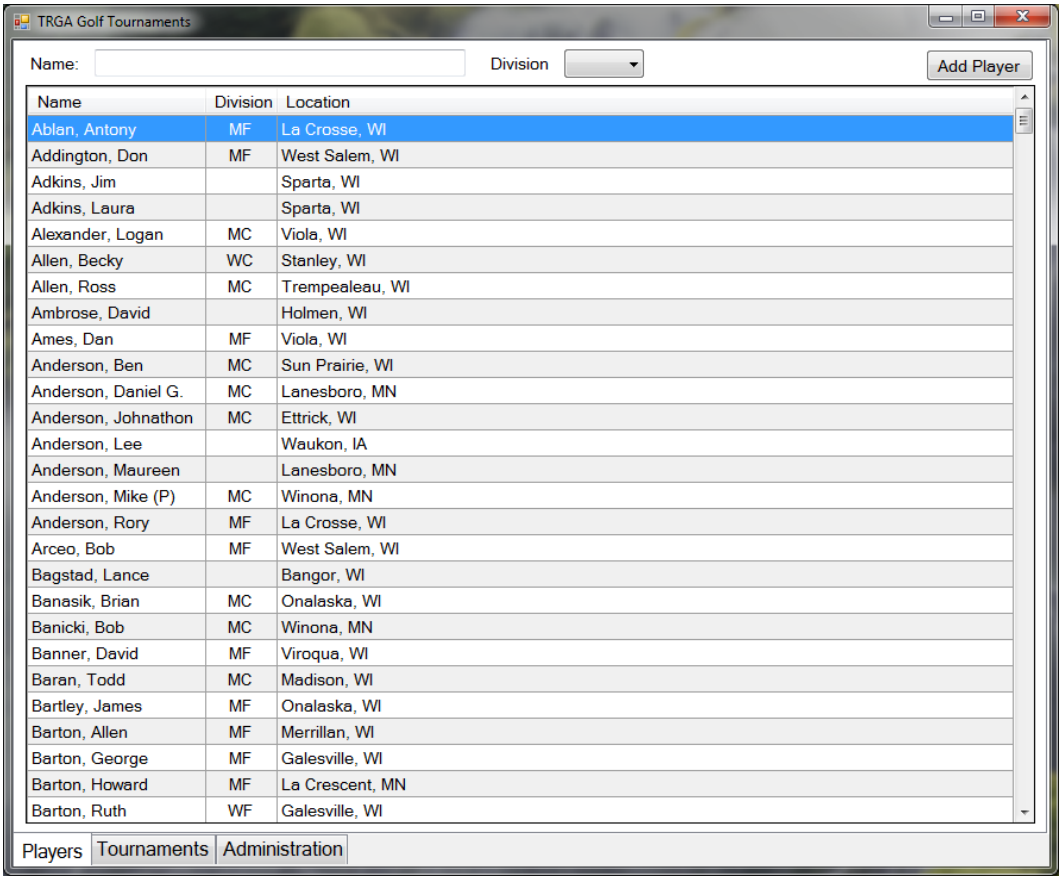

Figure 8 - Player Management

All information for an individual player is managed through a single dialog with three tabs. The tabs separate data by its usage – general contact information, player association data, and finally tournament data. As with prior dialogs, the data is primarily mapped to the form fields using a binding source. The home course and tournament listings are populated using additional association data, and the TRGA index data is populated in a separate lookup from an additional table.

The tournament tab is used for both tournament registration and a review of prior registrations. If the event has occurred in the past, the event will only display if the player participated and will show the division in which the player competed. If the event will occur in the future, the row will be editable, allowing the user to

check a box indicating the player has registered for that particular tournament. The division will default to the division specified on the Player tab, but can be overridden by selecting a different value from the combo box. If the user first specifies a division, the checkbox will automatically be selected.

If changes are made to the registration status after pairings have already been completed, but prior to the event date, the user will be prompted prior to the changes being made. If the player is being removed from a tournament, the registration will simply be removed. A player being added to tournament after pairing setup will cause the user to specify the division, starting time and starting position of the player being added. While not an uncommon occurrence, the changes are infrequent and could be handled in a basic, user-driver manner, as changes made in the manual system never required updates to the previously posted pairings.

The form layout and controls did not create any problems, but the process of cancelling data changes while using a Binding Source control was problematic as the order of operations used when calling cancel methods on both the Binding Source and data source were critical. Functionality to identify if changes had been made without being saved also required a bit more work, as changes were occasionally identified, but could not be traced to any particular item.

4.5 Application Prototype 3 – Tournament

The third prototype comprised the largest portion of the application, which also is the key reason for the application. Upon the completion of this prototype, the application was able to execute a single, regular season tournament. The first component to manage tournaments was to create an interface that would allow each tournament piece to be accessed. A decision was made to create a collapsible panel for each tournament, containing a summary of registration counts and

navigation links. Additionally, the headers are colored depending upon the type of tournament and current date with respect to the event date.

The first action, Tournament Setup, does not need to be completed first, but is required prior to result calculations. The dialog recaps the current event details, as specified in the Event Administration dialog. It is critical the user verifies the course and slope ratings for the course, as the values are used for both calculating net scores as well as updating the player indexes. It additionally adds properties for the general purse and cost of a plaque, which are used to determine the purse for each division when calculating results. If individual flights have adjustments to their purse, an additional dialog can be used to specify flight purse overrides. The values entered are in addition to the standard formula.

The pairing process has been designed as a series of dialogs to guide the user through a wizard-type process. The first step is to review all players who were identified to have registered for the tournament. The display shows both USGA and TRGA indexes for the player, and attempts to choose which index should be used. The user has the ability to change the selected index simply by clicking the cell of the appropriate index.

After the indexes have been selected, the application will sort all players by index within their respective division after identifying which divisions need to be split into flights – notably Men's Flighted. Upon identification of flights associated with the parent division, it will automatically divide the players equally across the subdivisions. The user will be shown the resulting breakdown to manually adjust the breaks, ensuring the divisions, while balanced, also keep players of equal ability grouped together. Since players are always separated into flights using their index, the user only has the ability to move the first player to the previous flight or the last player to the next flight.

Having establishing the players in each division, the user is then prompted for the tournament pairing parameters. These values consist of the first tee time, the interval between flights, and finally the method to use for organizing pairings.

Three different methods are available, allowing for a random assignment, by index (grouping like-ability golfers), and points (grouping players competing against each other in the season standing). These methods are used at distinct periods during the season, but have been left for the user to select. The application will then calculate the total number of pairings required for the entire field, adjusting accordingly if flights cannot be evenly broken into threesomes, thereby requiring one or two twosomes to be created. The application will finally set up pairings automatically using the defined values.

The last step for the user is to display the pairings established by the application. Using a drag and drop method similar to that used for the starting order, the user will be able to move players around within the flight. The application will then adjust the starting time and order of all players between the old and new location of the moved individual. When the user has determined the order is acceptable, a report will be displayed that can be printed or exported and the pairings are stored into the database.

On tournament day, the user will only need to use a single dialog to enter results. The user will enter the front and back nine values, which will automatically sum to determine the gross score. The miscellaneous column will only be used if a player did not post a score and rather will receive a notation for having withdrawn (WD), not starting (DNS), not finishing (DNF), or disqualified (DQ). If the division being entered has been defined to use Net Scoring, an additional Net Score column will be displayed and the value will automatically be calculated using the players course handicap and gross score. The course handicap calculation is dependent upon the course rating entered earlier.

After all scores have been entered, the user will click the Finalize button. Prior to calculating any result, the application will first verify all scores have been entered for all players and that two or more players do not share the lowest score. If two or more players are tied with the lowest score, the user must select the checkbox of the player winning the playoff and declared the winner.

When the flight result data has been validated, the application will then determine the flight results. The first step is to calculate the total purse for the flight. The purse is determined by taking the total corporate purse contribution and dividing among the total number of players in the tournament. This per-player value is then multiplied by the number of players in the flight, providing the total corporate contribution for the flight. This value is further adjusted by adding a set number of dollars for each player in the flight and any additional flight contribution adjustment before removing any costs for an award plaque.

Having determined the total flight purse, the application then orders the scores from lowest to highest and obtains two look-up arrays containing the percent payout and points for each respective place. It then iterates through the player results assigning earnings and points for each player. If two or more players finish with the same score, the earnings and points for the respective places are summed and divided equally among those individuals. For example, if there was a two-way tie for second place, the dollars and points to be earned by the second and third place finishers would be averaged and given to both players with the next highest score being awarded the dollars and points for fourth place. The process ends by displaying a report which can be printed.

The final step in executing a tournament starts with a dialog to enter ESC scores. The dialog will default the ESC score to the gross score, requiring the user to only update values for players that have scoring corrections for index updates. The index differential will immediately recalculate, providing the user with some feedback of the respective change. The application will verify the ESC score entered is not greater than the gross score. After all ESC scores have been updated, the user will click Finalize to initiate the process to recalculate player indexes which will be used when setting up the next tournament.

Score adjustments are not required to be done prior to the next tournament, but failure to update data will result in previously calculated indexes being used until

the values have been updated. Finalization of ESC scores is the final step in executing a single tournament.

The challenges associated with this prototype all were related to data, both existing and original. On multiple occasions, players were placed into the wrong division due to their index not matching the value which had been originally used. Extensive examination of the original data and new calculations often occurred with mixed results. In several instances, the flight assignment was manually changed with the hope that the data would eventually self-correct in the future.

4.6 Application Prototype 4 – Championship Management

The final tournament of the season is a two-day championship that requires a few tweaks to the standard processing of a tournament. Unlike regular season events, players must qualify for the tournament and the field is capped at 120-players with slots assigned to divisions based upon the percentage of total registrations within that division. Therefore, the first step in determining eligibility is to calculate the number of entries for each division.

Having determined the number of entries for each division, the players results are then processed to identify all players having participated in a minimum of four tournaments. From these players, any player who has won one or more tournaments or has participated in all tournaments is automatically qualified. The remaining slots are then filled with the remaining players in descending order of season point standings. Any players having played in the minimum number of tournaments but did not earn enough points to qualify by points will be listed as alternates in the event a qualifier elects not to participate.

A special registration dialog was created specifically for the championship which simply lists the eligible players and a checkbox. This method was chosen over the regular registration method, as a large number of registrations are processed in a short period of time and it was felt that opening individual player dialogs would be extremely cumbersome.

The pairing setup process for the first day of the championship functions exactly like it does for a regular tournament. The user will adjust the index to use, adjust the flight breakdowns, and manually adjust starting times as necessary. The user will also enter scores as they would for a normal tournament, however when finalizing the first day's flight entries, the application will not verify a single lowest score or calculation results, rather it will create pairings for the second day.

Pairing creation on the second day is a shortened process as the indices have already been defined and players are already separated into their flights. The application will use the same starting times as the first day, reordering players in reverse order of their first day score. The user will only be allowed to override the player order before finalizing the second day pairings.

The user perspective for handing results for the second day will remain virtually the same. Two additional columns will be included to show the first day total and the two-day total score. After entering all scores, the finalize action will validate the data, ensuring the two-day total does not contain a tie for first place. Result calculation will again follow the same pattern, however it will adjust to reflect double points for each place – first place will earn 36 points, rather than the standard 18. The corporate purse contribution is to be entered as one total, and therefore no additional steps are necessary.

As with regular tournaments, the ESC scores will also need to be updated at the conclusion of the tournament. It is expected that scores will not be entered until after both days have been completed, however, even if the first day scores are entered immediately, they will have no impact on the player flights for the second day.

Having previously completed the majority the functionality necessary to implement championship management, the challenge of significance in this prototype was creating a user interface to easily register players for two events in a single step, as the data model ultimately treated the multiple day event as a series of single day events.

4.7 Testing

Rather than waiting until the entire application had been developed, the application was tested throughout the development process. The administrative data areas were initially verified by smoke testing. Since most of the interfaces were simple in nature, it was relatively quick to test the functionality to ensure new records were added to the database while existing records were updated. The cancel process was tested to ensure a new record was not created and changes were not saved.

The division starting order process was a bit more involved to test to ensure that reordering of divisions, both in moving a division up or down, would correctly update the numeric values maintaining the proper ordering. This testing was done using a combination of step-through debugging as part of the development process and later by querying the stored data.

Player management testing was initially done in a similar manner as done on administration. Using dummy players, a majority of the components were tested for correct functionality. As development progressed to player registration and tournament execution, and it was determined that actual data was needed, additional situations were encountered involving records that didn't have all required fields provided during the import process. These errors indicated that additional changes to be made to the application.

Testing of tournament execution was performed nearly exclusively using data for the 2009 and 2010 seasons. Having completed the 2009 season, the data was considered to be correct and would provide a method to test both user interface functionality as well as application logic. It was initially expected that some errors for player placement within the appropriate division may not be exact, but as more tournaments were processed, the accuracy improved to become a perfect match. However, this proved to be challenging due to data import and data integrity errors from older data. The 2009 data also contained errors itself, such as

incorrect summing of gross scores, transposed entries of ESC scores, and failure to calculate tied place results.

Some of these discrepancies were immediately ignored, while others were significantly researched in the attempt to understand the problem origination. Research occasionally uncovered situations where a particular player had not been sufficiently imported, a problem easily resolved. Another instance showed a player having an erroneous index entry which effected their handicap and net score calculation. Since this entry was in error and could not be replicated in the application, the only solution was to manually override the application data each time the value was updated until a number of tournaments had been played to remove the erroneous score from the calculation.

The final test for each tournament was made by comparing the results, calculation for dollars and points against those listed in the official results. If the total scores, points and dollars matched the results, the tournament was considered to have passed.

4.8 Requirement Changes

Several changes to the originally understood requirements have been made since the application development began. These changes are primarily a result of lower participation in recent years and an effort to put additional funds into the individual flight purses.

The first requirement change required the addition of a property to identify a tournament as unofficial, in which no points would be earned towards the season point standings. This requirement change was based upon an in-season bylaw change by the association, and had never been considered prior to the tournament for which it was added.

The women's divisions decided in recent seasons that they would prefer to forgo receiving a plaque as the tournament champion in favor of additional money in their relatively small purse. Therefore, it was necessary to provide a method to

remove a plaque from an individual division and adjust the purse calculation formula.

Later it was decided that the women's divisions should receive a plaque for the division champions, but in order to maintain the purse, the association would contribute some money towards the purchase price of the plaque. At the same time, the standard formula for the number of players in Men's Championship resulted in a total purse lower than the desired amount. This resulted in a small amount of the total purse being moved from the general purse to the Men's Championship purse. To accommodate these two requests, a feature was added that allowed per-flight contributions to be specified in addition to the general purse.

It was also determined that player registration changes between pairing creation and tournament day were more frequent than originally understood. It was originally believed these situations occurred once or twice per season, but in reality, they occurred on average of one or two times per tournament. The addition of a confirmation dialog box quickly resolved the inability to remove a registered player. A new dialog was added to add a new player into the appropriate location. The dialog, while simple, should be sufficient for release one of the applications as the user is familiar with the data and rules necessary to insert a player into the appropriate division.

5 Limitations

The primary limitation of the application is that it was designed exclusively for the TRGA, their policies and encountered situations during the last two seasons. Any requirement or encounter situation that was not identified during the development and validation process, or a change in the association policy will require an application enhancement.

Additionally, the application was not designed for flexibility to support the removal of certain core data, specifically starting order or divisions. This decision was primarily based upon the association history in which the number of divisions has not grown or reduced as the number of players varies. If this functionality does become necessary, the application can likely be updated with minimal modifications to support the necessary requirements.

6 Continuing Work

The application performs all necessary tasks to execute the tournament season. It does not provide any financial tracking to eliminate all needs for external data tracking. This financial tracking would provide the means to know which players have not submitted the appropriate funds, or more likely, have an unused credit.

Additionally, the project was originally conceived with the intention that the data from the application would synchronize with the association website, eliminating any manual management of the website data. Initially, it is expected that the application would only push data to the website, but a second step may retrieve data from the website containing tournament registration and payments.

Both financial tracking and website integration projects will likely be undertaken at a later date prior to the start of the next season to finish the initial vision that this application was intended to meet.

The association also contains a junior division to support and grow the junior golf community. Their tournaments, while similar in nature to the regular association, have very different rules regarding pairing setup and results. There may be an opportunity to include juniors within the existing application, or create a separate application using significant portions of this application.

7 Conclusion

The TRGA Golf Management application is an automation application, designed for a single user to more efficiently manage the execution of golf tournaments. It automates many of the tedious tasks and centralizes all data and functions into a single location, which should save the user significant time in the "between tournament day" data management tasks.

The application was developed using Microsoft's .NET Framework and SQL Server Compact Edition database. As a single user application, it can easily be moved to a different computer and have its data backed-up without any complex processes or tools. Reports generated by the application can be displayed onscreen, printed, or exported to either Excel or PDF formats.

8 Bibliography

[1] http://en.wikipedia.org/wiki/Handicap_(golf), *Handicap (golf)*, Wikimedia Foundation, Inc., 2010.

[2] http://en.wikipedia.org/wiki/Software_development_process, *Software development process*, Wikimedia Foundation, Inc., 2010.

[3] http://johnnycantcode.com/page/SQL-Server-to-SQL-Server-Compact-Edition-Database-Copy.aspx, *SQL Server to SQL Server Compact Edition Database* Copy, Scott "Johnny" Smith, 2009.

[4] Thomas A. Pender, *UML Weekend Crash Course*, Wiley Publishing, Inc., 2002.

[5] Institute of Electrical and Electronics Engineers, Inc., New York, NY, USA. *IEEE Recommended Practice for Software Requirements Specifications*, 1998. IEEE Standard 830-1998.

[6] http://msdn.microsoft.com/en-us/library/system.windows.forms.datagridviewcomboboxeditingcontrol.aspx, *DataGridViewComboBoxEditingControl Class*, Microsoft Corporation, 2009.

[7] http://msdn.microsoft.com/en-us/library/system.windows.forms.datagridview.rowenter.aspx, *DataGridView.RowEnter Event*, Microsoft Corporation, 2009.

[8] http://social.msdn.microsoft.com/forums/en-US/winformsdatacontrols/thread/a44622c0-74e1-463b-97b9-27b87513747e/#page:2, *Windows Forms Data Controls and Data Binding FAQ: #22 How do I perform drag and drop reorder of rows*, Zhi-Xin Ye, 2009.

[9] http://gotreportviewer.com/, *ReportViewer Tutorial*, Unknown, 2009.

9 Appendix

9.1 Sample Spreadsheet Data

	B	C	G	H	J	K	L	M
1	No.	Name	Address	City, ST Zip	Phone	Course	Member	Status
2	1	[redacted]	[redacted]	[redacted]	[redacted]	Coulee Golf Bowl	N	Amateur
3	2	[redacted]	[redacted]	[redacted]	[redacted]	River Run	Y	Amateur
4	3	[redacted]	[redacted]	[redacted]	[redacted]	Castle Mound	N	Amateur
5	4	[redacted]	[redacted]	[redacted]	[redacted]	LaCrosse CC	Y	Amateur
6	5	[redacted]	[redacted]	[redacted]	[redacted]	La Crosse CC	Y	Amateur
7	6	[redacted]	[redacted]	[redacted]	[redacted]	La Crosse CC	Y	Professional
8	7	[redacted]	[redacted]	[redacted]	[redacted]	Forest Hills	Y	Amateur
9	8	[redacted]	[redacted]	[redacted]	[redacted]	Cedar Creek	Y	Amateur
10	9	[redacted]	[redacted]	[redacted]	[redacted]	Forest Hills	Y	Amateur
11	10	[redacted]	[redacted]	[redacted]	[redacted]	Castle Mound	Y	Amateur
12	11	[redacted]	[redacted]	[redacted]	[redacted]	Trempealeau Mountain	Y	Amateur
13	12	[redacted]	[redacted]	[redacted]	[redacted]	Walnut Grove Golf Course	N	Amateur
14	13	[redacted]	[redacted]	[redacted]	[redacted]	Cedar Creek	Y	Amateur
15	14	[redacted]	[redacted]	[redacted]	[redacted]	Castle Mound	Y	Amateur
16	15	[redacted]	[redacted]	[redacted]	[redacted]	Castle Mound	N	Amateur

Figure 9 - Membership Roster

	A	B	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Player	TRGA Member10	Course	Mbr	Flt	Index	FXHL	VQGC	CVGC	FSHL	RRGC	DCM	TMGC	HGC	TRGA1	TRGA2
2	[redacted]	Y	Cedar Creek	Y	MC	3.8				$45.00	$45.00	$45.00		$45.00		
3	[redacted]	Y	Hiawatha	Y	MC	5.9								$30.00		
4	[redacted]	Y	Hiawatha	Y	MF	11.7	$45.00	$45.00	$45.00			$45.00		$30.00		
5	[redacted]	Y	Castle Mou	Y	MC	1.3					$45.00	$30.00	$45.00	$45.00		
6	[redacted]	Y	Hiawatha	Y	MF	9.1				$45.00	$45.00	$45.00		$30.00		
7	[redacted]	Y	Hiawatha	Y	MF	12.5	$45.00		$45.00					$30.00		
8	[redacted]	Y	Hiawatha	Y	MF	13.4	$45.00	$45.00	$45.00	$45.00	$45.00	$45.00		$30.00		
9	[redacted]	Y	Hiawatha	Y	MF	15.9	$45.00	$45.00	$45.00		$45.00			$30.00		
10	[redacted]	Y	Hiawatha	Y	MF	19.2		$45.00	$45.00		$45.00	$45.00	$45.00	$30.00	$45.00	$45.00
11	[redacted]	Y	Hiawatha	Y	WF	25.6								$30.00		
12	[redacted]	Y	River Run	Y	MC	0.6			$45.00		$30.00	$45.00	$45.00	$45.00		
13	[redacted]	Y	River Run	Y	MC	1.0	$45.00	$45.00	$45.00	$45.00	$30.00	$45.00	$45.00	$45.00		
14	[redacted]	Y	Viroqua Hil	Y	MC	1.2		$30.00			$45.00	$45.00	$45.00	$45.00	$45.00	$45.00
15	[redacted]	Y	Castle Mou	Y	MC	1.4	$45.00	$45.00	$45.00	$45.00	$45.00	$30.00	$45.00	$45.00		

Figure 10 - Tournament Entries

	A	B	C
1		TRGA Fox Hollow Open	
2		Fox Hollow Golf Course	
3		*at La Crosse, WI*	
4		Saturday's Pairings	
5			
6		Women's Championship Flight	
7		8:00 a.m.	[redacted]
8			[redacted]
9		8:08 a.m.	[redacted]
10			[redacted]
11			
12		Women's Flighted	
13		8:16 a.m.	[redacted]
14			[redacted]
15		8:24 a.m.	[redacted]
16			[redacted]
17			[redacted]
18		8:33 a.m.	[redacted]
19			[redacted]
20			[redacted]

Figure 11 - Tournament Pairings

	A	B	C	D	E	F	G	H
1		TRGA Fox Hollow Open						
2		Fox Hollow Golf Course						
3		*at La Crosse, WI*						
4		Saturday's Results						
5								
6		Women's Championship Flight			Score		Points	Earnings
7			[redacted]	41--42--	83		18.00	$57.00
8			[redacted]	40-44--	84		15.00	$31.00
9			[redacted]	43-43--	86		12.00	
10			[redacted]	46-43--	89		10.00	
11								
12		Women's Flighted			Score	Net	Points	Earnings
13			[redacted]	46-44--	90	67	18.00	$70.00
14			[redacted]	42-45--	87	71	15.00	$52.00
15			[redacted]	45-47--	92	77	12.00	$35.00
16			[redacted]	48-53--	101	78	10.00	$17.00
17			[redacted]	48-52--	100	79	8.00	
18			[redacted]	49-52--	101	82	7.00	
19			[redacted]	70-70--	140	110	6.00	
20			[redacted]			DQ		

Figure 12 - Tournament Results

	A	B	C	D	M	N
1	*Tournament:*		TRGA Championships 2008			
2	*Flight:*		Men's Championship Flight			
3						
4	Entries/Flight	"1-3	"4-6	"7-11		
5	Places Paid				Purse/Flight	Payout
6	1	100.00%	65.00%	40.00%	$187.50	$75
7	2		35.00%	30.00%	$187.50	$56
8	3			20.00%	$187.50	$38
9	4			10.00%	$187.50	$19
26		100.00%	100.00%	100.00%		$188
27						
28	Formula for Certificate/Cash Calculations:					
29	Entries/Flight	7				
30		X				
31	Purse/Player	$15.00				
32		-				
33	Plaque:	$17.50				
34		+				
35	Corp. Contrib	$100.00				
36	Total Payout:	$187.50				

Figure 13 - Purse Calculation

9.2 Application Screens

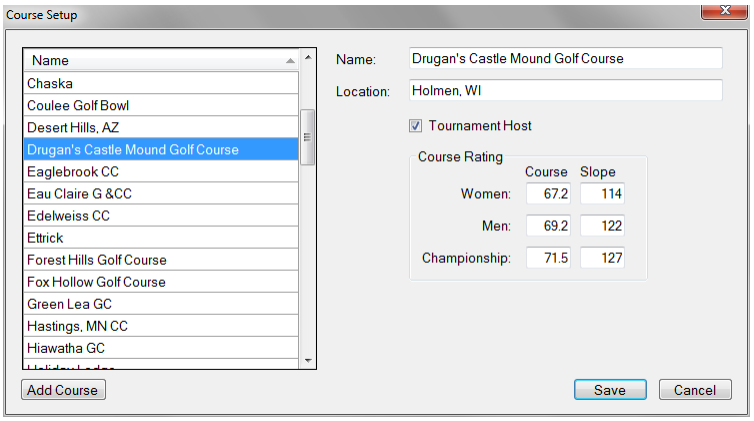

Figure 14 - Course Setup Administration

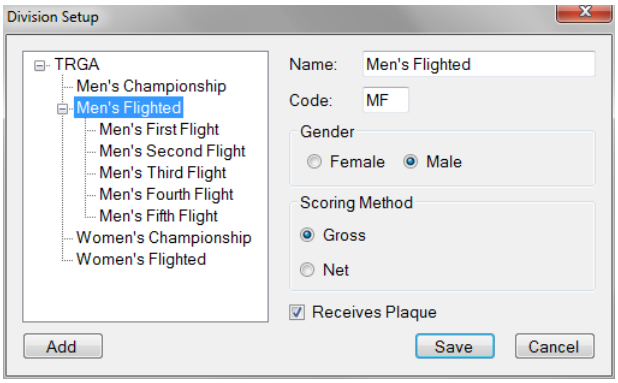

Figure 15 - Division Setup Administration

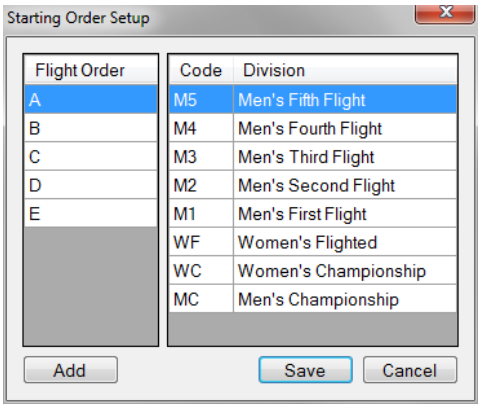

Figure 16 - Starting Order Administration

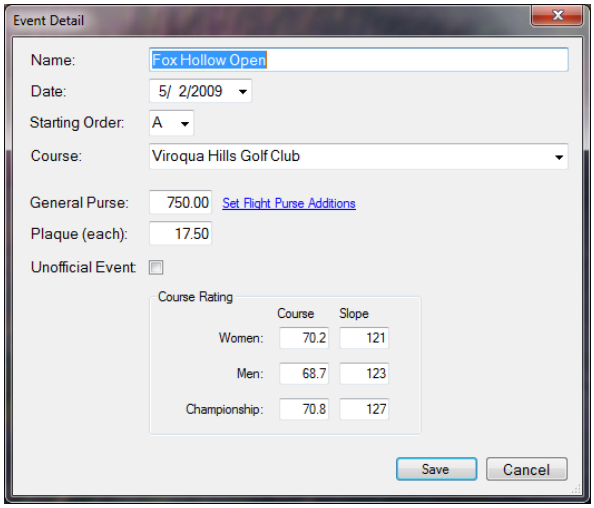

Figure 17 - Event Setup