Scorekeeping with Smart Phones
Mobile Solution for Outdoor Team Sports Tournaments

Hartti Suomela

Finnish Flying Disc Association
SLU 00093, Finland
hartti@csli.stanford.edu

Asmo Soinio

Finnish Flying Disc Association
SLU 00093, Finland
asmo@kortis.to

Abstract

This paper studies the use of smart phones for keeping score of games and timing those games in a sports tournament that is organized outdoors. The paper first describes the general challenges faced in designing such systems. Then we examine the issues faced during design, implementation, deployment, end-user training, and use of a scorekeeping system for a large-scale team sports tournament, in which multiple games are played simultaneously. Our implementation shows that smart phones can be effectively used for scorekeeping in such tournaments. These kinds of solutions are comparatively easy to develop and, if adequately designed, the volunteers can easily learn to use such a system.

1 Introduction

Scorekeeping and timing of games in professional team sports tournaments, leagues, and separate single games is usually handled by hired and trained personnel operating sophisticated computer applications or dedicated scorekeeping equipment. However there are plenty of team sports events, where such facilities and equipment for scorekeeping activities are not available. At those events also the personnel is usually comprised of untrained volunteers, and they would need extra training to use such equipment.

This lack of personnel and equipment is especially true in amateur sports events, in events targeted for juniors, and in events for less-known sports like ultimate Frisbee[1]. Additionally organizing an outdoor event for such sport creates further challenges compared to running an indoor event with controlled climate and most of the infrastructure (e.g. electricity, possibly wired networking) in place.

This paper describes the implementation and deployment of a comparatively low-cost system, which allows untrained volunteers to keep time and score for matches in a large-scale, multi-site tournament. First we study the challenges and requirements for keeping score of games in an outdoor tournament in general and provide an overview of related research. Thereafter we briefly visit the rules of Ultimate and describe the prior scorekeeping methods and systems for this sport. The design, implementation, testing, and deployment of our system for World Ultimate Championships held in 2004 in Turku, Finland (WUGC2004) is described in detail hereafter. In the final chapter we discuss how the system could be developed further to for example

[1] From here on, we will refer to ultimate Frisbee with the term Ultimate. Frisbee is a registered trademark of WHAM-O Inc.

accommodate also other sports with somewhat different rules and different scorekeeping requirements.

2 Score- and Timekeeping Systems for Team Sports

There are a number of requirements for a scorekeeping system for team sports tournaments.[2] First of all the system should be easy to learn and easy to use, as there is usually only limited time available to train the volunteers to use the system. Secondly, the system has to be reliable and robust as there is not usually enough personnel or additional devices available to have a parallel backup system running for all of the games. The system should have means to gather the required data from the game (for example scoring time and scorer information)[3] and to help the scorekeeper to keep track of available time-outs and to control the time-out lengths, etc. If real-time score reporting on a Web site is required, the scorekeeping devices should also send scoring updates to the tournament back-end system. A nice additional feature would be to have the scorekeeping device to control a scoreboard display on the fields for the enjoyment of on-site spectators.

The scorekeeping systems for outdoor tournaments face additional challenges. The scorekeeping devices operated on the fields have to be weatherproof or at least there should be means to easily protect the devices from rain. Tables and chairs might not be available, so the scorekeeper has to be able to operate the device with one hand only, and sometimes even while standing.

When smart phones or other hand-held devices such as PDAs (Personal Digital Assistant) are used for scorekeeping, there is a choice of input methods available. Some of them are better suited for scorekeeping systems. This paper studies keypad and menu-based UI in more detail, but also other input methods would be possible. In short, voice recognition could be used, as the required vocabulary to control the UI is limited – the vocabulary would consist of a handful of event nouns (e.g. ‘timeout’, ‘score’ / ‘goal’, ‘assist’, ‘substitution’), the names of the two teams (or ‘visitor’ / ‘home’), and a list of player names. However the background noise typical of most sport events would most likely cause problems even if highly a directional microphone were used. Pen input would allow the application designer to create a screen layout with on-screen buttons, which could be matched to events and players. However, pointing those small buttons with the stylus could be challenging in real-time situations, given the lack of haptic feedback and high precision requirements. Additionally, the stylus could be lost easily.

From the infrastructure viewpoint, there is seldom access to electricity on the fields, so the devices need to be battery-operated. Preferably the battery should last for one day. Alternative solution is to provide backup batteries for the field personnel or to set up charging stations near the fields. Also the fields usually do not have wired access to the Internet to access the tournament back-end system. Therefore the devices need to have some kind of wireless connectivity, being it cellular data access such as General Packet Radio Service (GPRS), or wireless broadband access such as WiFi. As the cellular access can be costly, an additional challenge for the tournament organizers is to negotiate an agreement with local carriers for cheaper data access or to find other financial means to handle the data transfer costs.

[2] Besides a score- and timekeeping subsystem, a tournament needs to have a back-end system to schedule the games, to collect and to tabulate the scores from multiple simultaneous games, and to display the final scores and team standings for example on a Web site. In this paper we do not study the back-end system as we concentrate on the scorekeeping subsystem.
[3] The data gathering requirements vary from sport to sport. In some sports the official box score includes also other data than just the scoring information.

2.1 Related Research and Available Scorekeeping Systems

The score- and timekeeping applications and systems have not been widely studied from the academic perspective. The usability approach in studying these systems is practically non-existent. For baseball, a simple scorekeeping device for disabled people, with LCD-screen and a couple of input buttons to update the score has been described (Mierzejewski & Enderle, 2000). This stand-alone device does not have any access to a back-end system to allow it's use in tournaments, nor does it include timekeeping functionality. Barstow (Barstow, 1999) has studied online baseball statistics system, but mainly from a system architecture viewpoint. On the other hand, there is an abundance of PC-based scorekeeping software applications for various team sports, including baseball, basketball, football, ice hockey, and soccer.[4] Some of these applications are basically digital versions of traditional paper score sheets and the user experience is not optimal. There is also a selection of scorekeeping applications for hand-held devices.[5] Such systems are more flexible for outdoor deployments than systems utilizing laptops.

There is also a selection of usability studies of mobile devices providing additional event-related information to the audience (Olsson & Nilsson, 2001) and facilitating sports event audience participation, see for example (Daley & Gabriel, 2004).

3 Case Study: Ultimate

Ultimate is a team sport played by two seven-player squads with a plastic flying disc on a field similar to football. The object of the game is to score by catching a pass from a teammate in the opponent's end zone, for which a team is awarded with one point. A player must stop running while in possession of the disc, only pivoting and passing to any of one's teammates is allowed.

Games are usually played to a point cap (usually 15-17 points / goals). In tournaments game-play is limited with an additional time cap (usually 90-100 minutes) to keep up with the schedule. If the time cap is reached before either of the teams has reached the point cap, a multi-step time cap procedure is used to find out to how many points still need to be scored (in general "leading score" + 2). As this procedure can appear complicated to an inexperienced scorekeeper, a scorekeeping system should assist the scorekeeper in calculating the correct point cap.

Besides the time cap, timekeeping responsibilities generally include limiting halftime length, timeout lengths, and the time between points (after one goal has been scored and before the play for the next point has started). The scorekeeper should also be aware of how many time-outs each team currently have available.

3.1 Available Scorekeeping Systems for Ultimate

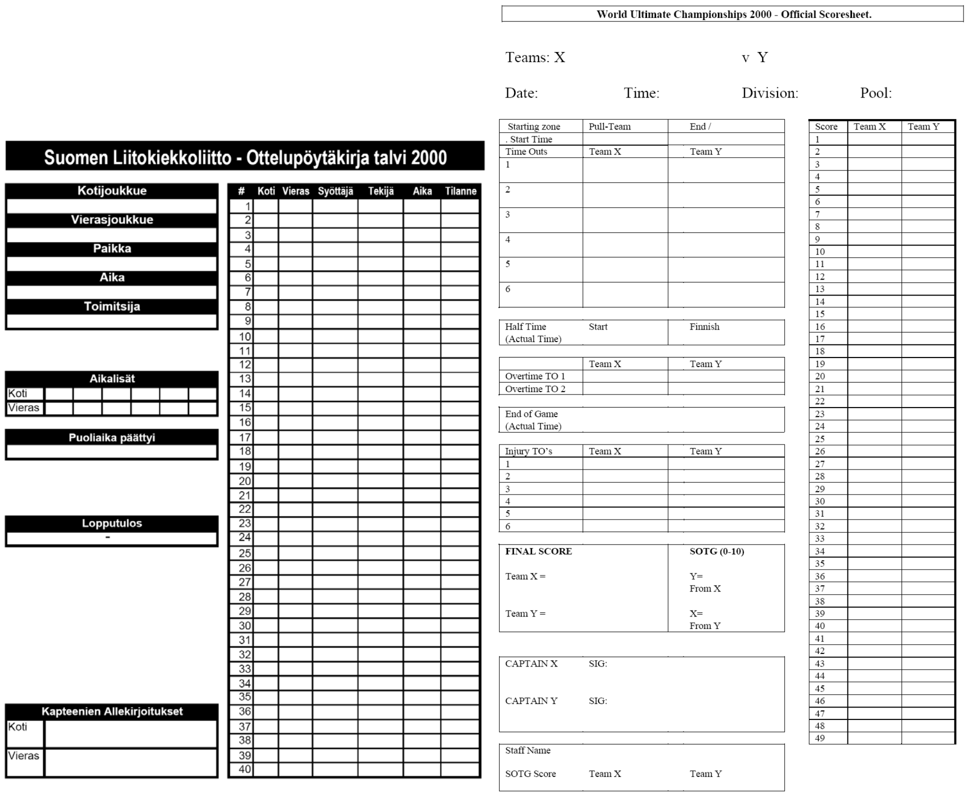
The Ultimate community in general is quite relaxed what comes to the personal scoring information and even to the exact final scores. It is common that in tournaments only win-loss record is kept although the final scores of all games are usually observed to be able to solve tiebreakers. In many tournaments there are no separate scorekeepers for the games. The games are timed with a central horn and the players themselves are responsible for keeping the score.

[4] For example in NCAA basketball the official score is kept and the box scores are produced with The Stat Crew System available at http://www.statcrew.com/

[5] For example Digital Scout provides solutions for various team sports. See http://www.digitalscout.com/

However there are a growing number of tournaments, which provide somewhat delayed scoring information for the games on the Internet, to serve also those people who were not able to make it to the tournament themselves. Also some tournament organizers have started to provide individual scoring information (goals and assists only). Such record keeping requires that each game has a non-playing scorekeeper.

For example the Finnish Flying Disc Association (FFDA) has been using paper score sheets since early 80s to keep track of individual scoring information. Usually the "home team" is responsible for keeping the score and reporting the score to the league organizers. FFDA has also an online system called 'Pelikone', into which scorekeepers could update the information from the paper score sheets[6]. Also the World Flying Disc Federation (WFDF) has used paper score sheets in the official WFDF sanctioned tournaments, like World Ultimate Championships and World Ultimate Club Championships. Two examples of these score sheets are provided in Figure 1.



Suomen Liitokiekkoliitto - Ottelupöytäkirja talvi 2000

Kotijoukkue
Vierasjoukkue
Paikka
Aika
Toimitsija

Aikalisät						
Koti						
Vieras						

Puoliaika päättyi

Lopputulos
-

Kapteenien Allekirjoitukset	
Koti	
Vieras	

#	Koti	Vieras	Syöttäjä	Tekijä	Aika	Tilanne
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						
28						
29						
30						
31						
32						
33						
34						
35						
36						
37						
38						
39						
40						

World Ultimate Championships 2000 - Official Scoresheet.

Teams: X v Y

Date: Time: Division: Pool:

Starting zone	Pull-Team	End /
. Start Time		
Time Outs	Team X	Team Y
1		
2		
3		
4		
5		
6		

Half Time (Actual Time)	Start	Finnish

	Team X	Team Y
Overtime TO 1		
Overtime TO 2		

End of Game (Actual Time)	

Injury TO's	Team X	Team Y
1		
2		
3		
4		
5		
6		

FINAL SCORE	**SOTG (0-10)**
Team X =	Y= From X
Team Y =	X= From Y

CAPTAIN X	SIG:
CAPTAIN Y	SIG:

Staff Name		
SOTG Score	Team X	Team Y

Score	Team X	Team Y
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		
25		
26		
27		
28		
29		
30		
31		
32		
33		
34		
35		
36		
37		
38		
39		
40		
41		
42		
43		
44		
45		
46		
47		
48		
49		

Figure 1: Examples of score sheets for Ultimate. The official FFDA score sheet is on the left. The other one is the official score sheet for a WFDF tournament.

Obviously these paper-based systems cannot alone take care of timekeeping, for which the scorekeepers need to have a stopwatch. An alternative method, as described above, is to use a central horn to observe the time cap. In this case the timeouts and halftime are not properly timed.

[6] Pelikone (English translation: Game engine) was developed by Pasi Tuulosniemi in 1998. Web access to the system is available at http://www.liitokiekkoliitto.fi/pelikone

Score-O-Matic[7] is a computerized scoring system, which initially accepted scoring information for soccer and Ultimate games from any phone through key dial tones. Anyone could register to the system to provide scoring information and the system displayed real-time score of the games on the Web. More recently the development team has added WAP (Wireless Application Protocol) interface to the system, and it has been already piloted in a couple of North American tournaments. The WAP interface causes some delays in the UI and does not therefore provide an optimal user experience. However the pace of an Ultimate game is such that these delays are acceptable. Additionally very few people in North America realize that their phone has a WAP browser, so it has been quite hard for the Score-O-Matic team to find volunteers to keep scores in tournaments (Kerr, 2005).

There are also some more advanced statistical systems for Ultimate. These systems, such as the UltiStats application for Palm OS devices[8] and a paper-based notational sports analysis system called RUFUS[9], are more targeted towards keeping specialized and more detailed statistics of a game than providing scoring information for a tournament.

4 StatKeeper: Scorekeeping and Timekeeping for Ultimate

In this section we describe the design, implementation, and deployment of our system called StatKeeper for World Ultimate Championships tournament. The application was designed for Nokia Series 60 smart phones, as the tournament got a number of such devices as a donation. The tournament organizers were also able to negotiate an affordable deal for data transmission charges with a local carrier, which further solidified smart phones as our choice of development platform.

Mainly due to the strict development timetable, Java (J2ME MIDlet) was chosen as a development language instead of native Symbian C++, which has a steep learning curve. This limited our choices to some degree during the development especially on the UI side (e.g. soft key mapping).

One major advantage of MIDlet development is the availability of very good and free-of-charge development and testing tools. For this project, two software development kits were evaluated. Mainly because the emulator startup time in the Nokia Developer Suite for J2ME 2.1 was about one minute, which made the testing really slow, Sun's J2ME Wireless Toolkit, version 2.1 was used for development and testing. The drawback of this solution was that the emulator did not really resemble real devices. For detailed UI testing in real environment the application was sent to the real target device over Bluetooth.

In parallel to the scorekeeping application development a back-end system for scheduling the games, keeping track of rosters, collecting the scores, and displaying the scores on a Web site was developed by a 3rd party company (Fiare Oy). As there is a messaging interface between these two systems, the development efforts needed to be coordinated carefully and therefore a requirements specification along the messaging interface specification was created (Suomela, 2004). Many of the requirements listed in this document have been already presented in the earlier chapters of this paper.

[7] Score-O-Matic is developed by a team lead by Charles Kerr. Web access to the system is available at http://www.scoreomatic.com/

[8] The source code and other UltiStats documentation is available at http://sourceforge.net/projects/ultistats/

[9] This system was used in late 90s in U.S. for pass and possession level analysis of Ultimate.

4.1 User Experience and UI Design

In general, the process and the actions of score- and timekeeping define a simple structure for scorekeeping applications. First of all, the user has to command the application to observe the game one has been appointed to. Then the application waits until the game is started. After game start the application keeps track of time and waits for input from the scorekeeper (mainly timeout taken, goal scored), which will lead to specified actions. As the application returns to the main "game-on" state, a report to the tournament back-end server is sent in the background.

Alternatively, based on the game clock or on the score of the game the application itself suggests actions to the scorekeeper (time for half-time, game has ended, etc.). After the end of the game the scorekeeper collects approvals from the team captains and a game summary gets sent to the back-end system. This simplified process is depicted in Figure 2. The same structure was used as basis for the design of the StatKeeper application.

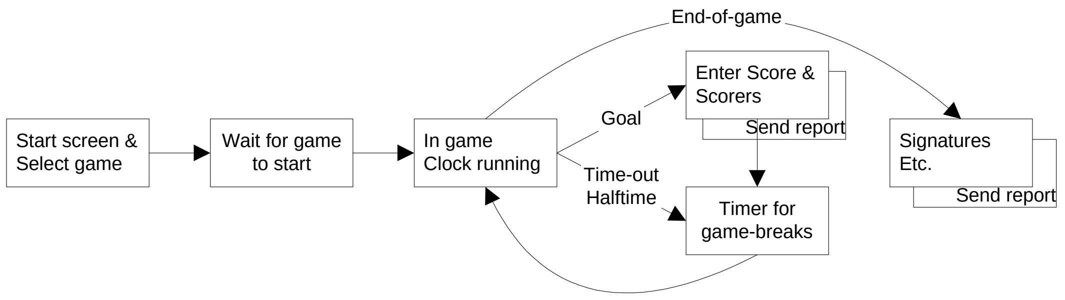
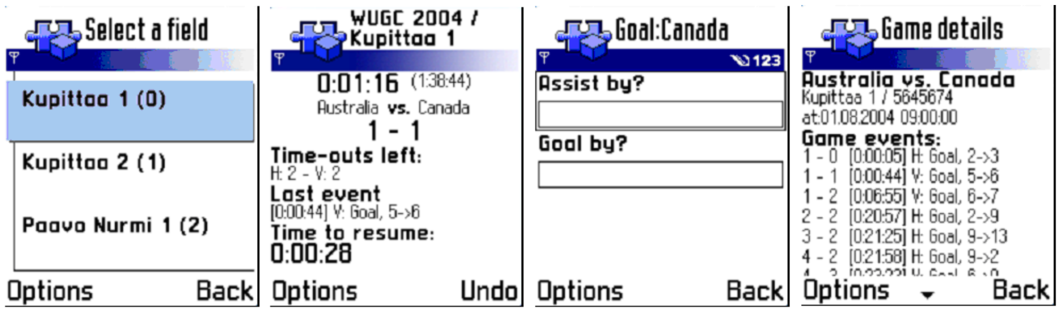


Figure 2: A simplified process view on scorekeeping used in the StatKeeper application.

As the volunteers for the tournament were from many age groups and from different parts of the world, their background with cellular phones varied a lot – some had had only a little experience with cellular phones, and even less with smart phones. Also training time prior the tournament was limited, and even if a user manual was provided to the volunteers, it was most unlikely that anyone of the volunteers would have had time to read the manual. As the volunteers were envisioned to work sometimes under a lot of stress, the system needed to be as easy to use as possible. The volunteers should not need to configure anything, all the messaging between the application and the back-end server had to be hidden, and as much of the game logic as possible (e.g. timeout lengths, point caps, time caps) was to be implemented into the system.

The usability requirements dictated the choice of the UI style. Using numbers as shortcuts to specific actions would have been too hard for the novice users to learn, and therefore this approach was discarded. Using the menus operated with the soft keys of Series 60 phones was seen much easier for the volunteers to learn. Initially the plan was to map the left soft key to one team and the right key to the other team. Each of the menus would have had selections for timeouts and goals. This would have been natural from the scorekeeper's viewpoint, but this mapping was discarded because of implementation problems and because the mapping of the right soft key run against the general Series 60 application guidelines. As following the standard was seen to be important, we finally agreed on implementing a menu structure along the Series 60 guidelines, where the left soft key was reserved for various actions and the right soft key was mainly used for reverting the previous action and for exiting from the application.

Considerable time was also spent on deciding what to show to the user on each screen. It was agreed that better to err on showing too much than too few items to the user, even if the screen would end a little cluttered. Figure 3 provides four screenshots of the StatKeeper application. Note that in the second screenshot the letters 'H' and 'V' stand for home team and visiting team respectively. This order for the team scores is typical in Europe, whereas in U.S. the score for visiting team is usually displayed first.



```
Select a field
Kupittaa 1 (0)
Kupittaa 2 (1)
Paavo Nurmi 1 (2)
Options        Back
```

```
WUGC 2004 /
Kupittaa 1
0:01:16 (1:38:44)
Australia vs. Canada
1 - 1
Time-outs left:
H: 2 - V: 2
Last event
[0:00:44] V: Goal, 5->6
Time to resume:
0:00:28
Options        Undo
```

```
Goal:Canada
123
Assist by?
Goal by?
Options        Back
```

```
Game details
Australia vs. Canada
Kupittaa 1 / 5645674
at:01.08.2004 09:00:00
Game events:
1 - 0  [0:00:05] H: Goal, 2->3
1 - 1  [0:00:44] V: Goal, 5->6
1 - 2  [0:06:55] V: Goal, 6->7
2 - 2  [0:20:57] H: Goal, 2->9
3 - 2  [0:21:25] H: Goal, 9->13
4 - 2  [0:21:58] H: Goal, 9->2
Options        Back
```

Figure 3: Screenshots of StatKeeper. In the first screen the user selects the appointed field and game. The main screen displays relevant game information. Scorers are entered via a dedicated form. The fourth screen presents a detailed view of a game displayed after the end of the game for verification purposes.

4.2 Software Design

Due to the strict timetable and the quite good understanding of the application's structure, there was little pure architecture and design work done during the project – application development started almost immediately. The overall design of the application was around the UI – each visible UI state, mostly a MIDlet UI form, corresponds to an application state, and all the actions that change the UI state go through the same method, which handles state changes. Other actions are handled within each state. Some drawbacks due to this simple design were found as the application grew: The in-game state became really big – even though not so complex – and the method for changing between states also became really long.

Prior to the start of the game, changes between many UI states required communication. This was implemented with a common interface for sending data and getting a callback function called with the response once it was available. The data for such states was then filled in the callback function and at that time the actions made active.

The only state in the application that differs from the logic that only UI actions can cause actions is the in-game state. Two separate threads were used, one to update the game timer and the clock time, and another to send the game events (e.g. goals, timeouts taken) generated so far to the server. The design of having one single separate thread for handling all the communication during a game was selected mostly for clearness – the logic is extremely simple and easier to monitor and handle, compared to creating separate threads for communication each time a game event occurs. This thread does not signal any errors to the UI during a game, it just retries to send the data every two minutes. This logic was selected not to disturb the user during an eventually hectic game situation. Only after each game, when the final score is sent, the user is told about possible communication problems, and the game data is stored in the phone for later handling.

One crucial design point of view for the whole application was robustness for example in the case that there are application errors or the phone's battery dies. To protect against such cases the game data is stored in the phone's non-volatile memory as well as sent to the server each time after an event in the game: At the beginning of the game the actual starting time is also stored. With this information available, it is possible to fully recover the application without any problems if the battery dies or the application goes to an unrecoverable state. The application can similarly recover from situations, in which the non-volatile memory of the phone is erased.[10] All events inputted so far are stored on the server, and the only thing lost would be the game clock and events that happened while the phone was rebooting.

The UI of the application currently in English only, but it was designed and implemented so that it is easy to add new languages. All texts are accessed via a function interface using a logical id that is basically the index in an array of all texts. The lifetime of some graphical items made it hard to change the language on the fly, therefore multiple language interfaces were not implemented.

4.3 Implementation Details

The UI of the application was implemented using the MIDP Form system. It provides an easy and fast way to create applications with standard widgets (e.g. text fields, text areas, radio buttons). It has however quite a few limitations: There is no interface to define colors on the widgets, so the application is black and white (except for the title logo). There is also no interface to listen to direct key presses – all functionality must be handled via abstract commands, which in practice are mapped to the soft keys and the "Options" list.

The most severe limitation, which is a Series 60 MIDP implementation specific feature, is the fact that an "Exit"-command is added to each form. Therefore it is too easy to accidentally quit the application, especially by an inexperienced user in a hectic situation. One way around this would be to use the MIDP Canvas-system, which provides more freedom to the UI design, but at the same time it makes the implementation harder. Alternatively one could create a native Symbian application or develop the application on a completely different platform.

Among positive findings is that the Series 60 screen size is big enough to hold the needed information during the game. This could further improved with a better graphical design, as the UI of the current application was not fine-tuned much. Also the usability of the system and the understandability of the platform's logic (the soft keys and the "Options" list) were clearly good enough judging by how successful even the inexperienced volunteers were with StatKeeper.

4.4 Pre-tournament Testing

Because of the tight time schedule, and especially as the back-end server implementation was completed only a few days before the tournament, it was necessary to be able to test the application without the actual server. For this reason an emulation of server events was implemented in the communication part of the application. This emulation gives the correct XML server responses for each possible request the application makes, so that everything down to XML parsing can be tested.

[10] This would require some really fatal application error or severe problem in the phone's internal software.

Testing with the built-in server emulation gave no clue about how long it would take to send and receive the messages. This information was not extremely important, as during the game the application communicates in the background. However, some data was needed. For this purpose a really simple HTTP server emulation was implemented using PHP. The script simply matches the client's requests to filenames of XML files that contain the correct server response. Testing with this server was enough to proof that pre-game functionality, selecting the field and the game, was in fact quite fast, even though it requires communication at each state change.

Even with these emulations, testing the full system was of course a necessity. Some parts of the communication were tested while the server was still under implementation, but a few test runs with the full system were done just some days before the tournament started. Because of the accurately specified XML interface, there were only a few bugs related to communication. The most valuable output from testing was to help making the UI as logical as possible, and this was possible even with the in-phone server emulation.

4.5 Tournament Usage and Feedback

The group of 30 volunteer pairs was very heterogeneous. They had different skill levels in using phones and in scorekeeping. To make sure that all the scorekeepers were capable of using StatKeeper, a short training session was organized and a short user manual was provided for them. The training was organized so that each pair got a mobile phone, and they were given step-by-step instructions on how to handle the application. Many scorekeepers were able to use the application without any instructions, but some inexperienced phone users were slower to learn, as they had never used a complex mobile phone application before. The fact that all of the scorekeepers were able to use the application with only a little training verified our UI choices and instructions to be acceptable. The training session was also the first ever testing that the server was able to handle multiple simultaneous clients.

The application succeeded in providing results for almost all of the games (392 out of 395) although the application crashed for some unidentified reasons more often than that. The built-in recovery mechanism succeeded in allowing recovery in most of the cases. The missing results were collected from the paper score sheets, which were kept as a backup for StatKeeper application. The instability of the application and platform was noted also in the feedback gathered from the volunteers using a paper form (28 % response rate). For some reason the application often slowed down if it was used for many games in a row. Restarting the system solved this problem. The overall slowness of the application on the Series 60 platform[11] was not perceived to be problematic, except in a few exceptionally fast situations.

For the tournament organizers the system was a great success, and provided more detailed information than what typically has been available in these kinds of tournaments. Unfortunately, because of the limited functionality of the tournament Web server, the scores weren't shown on Web in real-time although the data was available on the server side.

5 Discussion

The StatKeeper application proves that implementing a low-cost and highly usable scorekeeping solution is possible using smart phones as a platform. The authors see this kind of system

[11] The application runs much faster for example on a Series 40 UI phones such as Nokia 6230.

benefiting many volunteer run tournaments like international junior soccer tournaments (e.g. Hesa Cup with 800 teams) or little league baseball tournaments. Some changes would be needed because of different rules and statistical requirements. In slow scoring games such as soccer and even Ultimate, a WAP-based solution would be sufficient too, if the network coverage is good.

Based on the feedback and experiences during this project some future improvements are planned. The UI should be re-implemented using MIDP Canvas instead of the MIDP Form system, which allows more freedom in the UI design. The most important improvement achieved with this is that the "Exit"-command would not be on every form. Besides, with MIDP Canvas the UI design can be redone to improve the visibility of the important details – different fonts and colors can be used to make the main screen easier to interpret. Also current clock time and battery level should be available on the main screen, because that information is worthwhile to the scorekeeper.

There is one application logic error in the application: If a goal is scored and the discussion on the field takes too long (as can happen when a new point cap is introduced because of reaching the time cap), the phone might start the next point automatically after one minute, thus changing the point cap incorrectly. This was encountered once during the tournament.

In many tournaments and especially in series, which contain many tournaments, the player roster for any team may change. Therefore roster-editing capabilities should be added to the application. The scorekeeper could show the current roster to the team captains and the application should have functionality to add, delete, and edit entries if needed. In the current system player names were actually sent to the mobile phone, but they were not shown to the scorekeeper. Another useful feature would be the ability to send a stored game via Bluetooth to the server.

6 Acknowledgement

The authors are thankful to Pasi Tuulosniemi and Juha Jalovaara from FFDA as well as to Mikko Kaihlavirta, Hannu Kytölä, and others from Fiare Oy for their efforts in the project.

References

Barstow, David. (1999). Baseball seasons and dog years. Proceedings of the 21st international conference on Software engineering. Pages: 535 - 542

Daley, W. Gabriel, I. (2004). System for audience participation in event scoring at the 2004 Olympic Games. Extended abstracts of the 2004 conference on Human factors and computing systems. Pages: 1685 - 1689

Kerr, Charles. (2005). Personal email communication.

Mierzejewski, G. Enderle, J. D. (2000). The electronic baseball scorer. Bioengineering Conference, 2000. Proceedings of the IEEE 26th Annual Northeast. 161-162.

Olsson, Daniel. Nilsson, Andreas. (2001). MEP: a media event platform. Mobile Networks and Applications. Volume 7, Issue 3 (June 2002). Pages: 235 - 244

Suomela, Hartti. (2004). StatKeeper midlet – Keeping statistics of an ultimate game: A document of client requirements http://www.ultimateplayer.org/statkeeper/StatKeeperReqDoc_0_62.pdf