eStadium: The Mobile Wireless Football Experience

Aaron Ault, James V. Krogmeier
Center for Wireless
Systems and Applications
Purdue University
West Lafayette, IN 47907, USA
Email: {ault,jvk}@purdue.edu

Steven R. Dunlop
Envision Center for
Data Perceptualization
Purdue University
West Lafayette, IN, USA
Email: dunlops@purdue.edu

Edward J. Coyle
School of Electrical and
Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA
Email: ejc@ece.gatech.edu

***Abstract*—It has become clear that the original Mobile Web model of people using their portable devices to surf normal webpages must be refocused on the creation of applications that are tailored for the mobile experience. The eStadium project is our attempt at creating such an application. It provides live "infotainment" such as real-time statistics, instant replay videos, and venue information to Purdue football fans via a public mobile web application and an on-demand video delivery system. In this paper, we discuss the design, implementation, and operation of the eStadium system and the lessons learned from five years of serving real sports fans.**

I. Introduction

Widespread deployment of 3G systems, advanced broadband data services, and the proliferation of a new generation of web-enabled mobile devices were *supposed* to drive an explosion of usage of the Mobile Web. Unfortunately, the availability of these new systems and devices has not convinced users of the world's 3+ billion cell phones to use them for surfing. [1] The primary reason is that users' hopes of replicating their desktop/laptop (DL) surfing experiences on their small mobile devices have been frustrated by the limited input and display capabilities of the devices. The vast majority of web sites that are interesting enough that users visit them regularly via their DL's are thus not easily viewed or used from their phones.

Since a direct translation retaining the entire functionality of the desktop/laptop (DL) experience has not been successful to date, new usage scenarios in which the small size and mobile nature of the device is *essential* to the user's experience must be explored. One unique opportunity to do this is presented by large sporting events. Most fans bring their phones to these events and could use them for on-demand access to game and venue specific information that could improve their enjoyment of the event. This might include live game statistics, real-time video highlights, site specific resources, ordering food from their seats, etc. .... the type of information and services that could not be provided in any other way. However, it remains to identify the information and services of interest to the largest number of users and to develop ways to access them that make them simple and fun to consume on a typical mobile device.

In this paper, we present the eStadium project, which is our attempt to provide a mobile web application that is simple to navigate, easy to maintain, and contains useful, up-to-the-minute content and services for football fans at Purdue's Ross-Ade stadium. We focus on three areas that are critical to achieving that goal: mobile web application design, mobile video delivery, and real-time data collection and dissemination algorithms.

Fig. 1. In 2001, eStadium became the first to deliver on-demand instant replays to mobile devices.

A. History of eStadium

eStadium, which was launched in 2001, is a partnership between Purdue's Center for Wireless Systems and Applications (CWSA), Information Technology at Purdue (ITaP), and Purdue Intercollegiate Athletics. The project provides a unique service to Purdue football fans as well as a real-world research and learning environment – a *living lab* – for CWSA researchers and students. The original eStadium application provided information such as concession stand menus, live statistics, and a utility to find nearby restaurants and hotels, to name a few features. In 2001, eStadium became the first to deliver instant replay videos wirelessly to mobile devices during football games via a WiFi network deployed in the stadium [2] [3]. eStadium continues to improve and expand its repertoire of tools and services by gathering user feedback, following up on users' suggestions for new applications, and and pursuing innovative research initiatives [4].

B. Related Work

While the Mobile Web has not been as successful as analysts have predicted, there has been plenty of activity on the mobile development front. The relatively new *dotMobi* domains [5], the Open Handset Alliance's much-anticipated open source mobile platform Android [6], mobile payment options with Paypal and Google Checkout, and Yahoo's Go with Mobile Widgets [7] are significant efforts to solve the mobile content consumption problem.

The delivery of sports-related information to mobile phones has seen significant increases in usage in 2007. Companies such as ESPN have begun to see more hits on their mobile sites than their PC-based sites [8]. The most prominent source of applications similar to eStadium is Sprint's NFL Mobile [9], which provides summary statistics, news feeds, live video from NFL Network, and same-day video and audio highlights from around the NFL via a custom web-like interface developed by mSpot [10]. Cisco continues its drive toward the "stadium of the future" with the construction of Cisco Field in Oakland [11]. Companies such as BelAir Networks and Nintendo have approached the problem of the game-day experience as well by deploying wireless networks and mobile applications at Dolphin stadium in Miami [12] and Safeco field in Seattle [13].

II. The eStadium Application

A. Application Medium

The first decision that must be made when creating a mobile application is whether to write native applications that run locally on phones or web-based applications.

One benefit of native applications is that the user experience can be made largely consistent across most devices. Additionally, data can be distributed in a customized manner to make better use of precious wireless bandwidth via predictive downloading or multicasting. The downsides of native applications include the programming staff and expertise required to maintain an application's consistency across the enormous variety of devices now available, the complexity of distributing program updates, and liability issues if the program crashes the user's device.

Web-based applications have the benefits of scalability. A single change to a website acts as an instant program upgrade to all users. Also, the system can be developed and maintained by a much smaller group of programmers. Additionally, there is little or no device-based liability since no code is installed on users' phones. The downsides of this approach are the challenges of implementing clever data distribution models like multicasting and accounting for the limitations and quirks of different mobile web browsers.

eStadium has chosen to focus on the web-based deployment method due primarily to personnel constraints. The current application requires only 2 people to operate on game day, and its code base can be maintained and upgraded by a single part-time developer.

B. Usage Models

To understand the problem of the mobile delivery of sporting event information, it is important to first understand typical usage models. These models can be broadly categorized into 3 groups:

- *Second Screen (SS)*: The mobile user's device acts as a secondary screen that supplements the live event, much in the same way that a large video board provides supplementary information to the live action.
- *Post- and Pre-Game Review (PPGR)*: The mobile users gather information about the event prior to its occurrence, and review what happened during the event both as they leave and during the days following the event.
- *Venue Interaction and Information (VII)*: This model includes ordering food and other items, venue navigation, and security announcements and instructions, to name a few. Note that this is the only usage model which is limited primarily to the people physically attending the event.

There are a multitude of successful solutions to PPGR, as this is where industry has focused its attention. The SS and VII models, however, have not received much attention so far. It is largely these two models that eStadium attempts to address. This paper focuses primarily on the SS model; however, the VII model is addressed by eStadium with items like team schedules, stadium maps, and concession stand menus.

C. Application Evolution

The eStadium application began as a C#/ASP.NET/MSSQL mobile web application that provided information about concession stand menus, an index of local hotels and restaurants, a Purdue trivia game, a Buddy Finder to help Purdue alumni find peers at the games, and live game statistics. Within a year, access to on-demand video of most instant replays was added, with each video indexed by the play-by-play text from the live stats feed.

As shown in Figure 2, the eStadium application has evolved significantly over the years due to user feedback, usability analysis, expansion to additional venues, and new mobile guidelines and standards such as dev.mobi's official Mobile Web Developer's Guide [14].

Lessons learned include the following:

- The large graphical banners used initially as headers and footers were found to occupy too much of the vertical screen real-estate for only aesthetic purposes.
- As the site map grew, it was reorganized so that all "Game Day" related links were grouped together under a single menu page for the current game. Unfortunately, in organizing the site this way, fans were forced to perform an extra page load (to get from the main menu to the game day menu) to access the pages they used most of the time.
- The pages made heavy use of HTML tables, which are not supported by all mobile browsers.

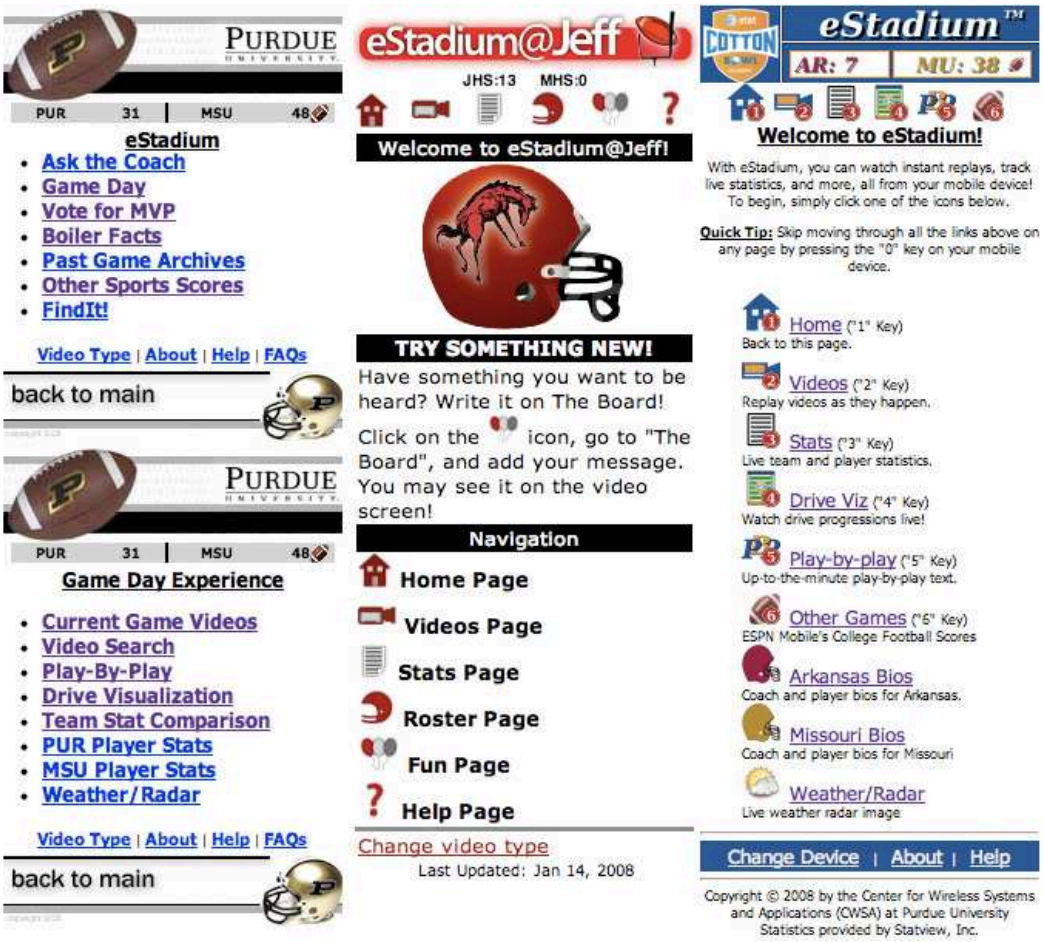

Fig. 2. Evolution of the eStadium application. Top Left: 2007 eStadium mobile homepage. Bottom Left: The result of clicking on the "Game Day" link from the 2007 eStadium mobile homepage. Middle: the eStadium@Jeff mobile homepage, released in the 2007 season. Left: eStadium Cotton Bowl mobile homepage, released in January of 2008.

- Due to content ownership issues that prohibit the distribution of game video outside the stadium until midnight of the day of the game, there was no way of reliably providing video to cell phones without risking a breach of copyright. As a result, only users on the in-stadium WiFi network were allowed to view replay videos. This was not much of a problem initially because 3G cellular services were not available at that time, so very few phones were capable of viewing the video.
- The site relied heavily on text-based rather than icon-based links. Text-based links were found to be difficult to click on in touch-screen based browsers such as the iPhone due to their small vertical size.

Our first attempts at combating some of these problems came in the 2007 season. First, due to recent 3G cellular deployments around the stadium, it became crucial that we support delivery of video to cell phones. The first step in accomplishing that goal was the introduction of a per-game, 4-digit access code that was shown on the video board twice per game, ensuring that only people physically present in the stadium see the code. If the IP address of the device requesting a video is not from the internal WiFi network, then the application checks for a valid access code before providing access to videos. The access code is stored in a PHP session for that device, which expires after the game is complete.

The eStadium application originally used a Windows infrastructure to capture and distribute videos. This setup, unfortunately, only supported delivery of video to Windows Mobile devices. We got an opportunity to prototype a completely new video capture and delivery system via a high school outreach program that we started in the summer of 2007. The high school program, named eStadium@Jeff, was based on teaching local students from Lafayette Jefferson High School to develop and run their own, from-scratch eStadium system.

The creation of a completely new system from the ground up enabled us to move away from Windows to a Mac/Linux-based environment. Given that there is no single standard for delivering video to cell phones, we decided to support as many methods as possible. We capture the game video live, and then use Final Cut Studio 2 to export the video to 3 different formats: large MPEG-4 for PC's, small MPEG-4 for devices with MPEG-4 support, and a small 3GP for devices with 3GP

support. Each of these formats can be delivered via RTSP stream from an Open Darwin Streaming Server (ODSS) or via HTTP download from Apache. This provides 6 overall video delivery options.

We then implemented a device detection algorithm that determines if the connecting device is a PC or a mobile device, and then attempts to determine the appropriate video delivery type based on the connection string. This choice is stored in the PHP session for that device, and controls which types of links are presented on the video pages. An important component of the video delivery subsystem is the ability of the user to change their device type should the auto-detection fail. This change in device type replaces the original decision in the session, and all the information about the change is stored in a database table for future analysis in improving the auto-detection algorithm.

We were also aware of the need to improve site navigation via "iconification;" we wanted to move away from text-based links to icon-based links. You can see in the center figure of Figure 2 the resulting site. A small header of 6 icons appear on every page, requiring fewer page loads and providing easier navigation in a smaller space.

The most recent evolution was a result of an opportunity to run the eStadium system at the January 2008 Cotton Bowl. While we did not have access to a live video feed to create replay videos, we were provided with a live-updating stats feed. Several additional site design modifications were made at this time to help solve the remaining issues.

- All graphics on the site are now drawn entirely from scratch using PHP scripts and the GD library. This enables them to scale like vector graphics for larger PC-based pages. It also enables important features of the site appearance, such as color schemes and logos, to be dependent on a database rather than requiring manual intervention to run at different venues.
- To guarantee that the live score display showed up properly on all mobile phones while simultaneously reducing the amount of vertical space on each page, the scoreboard was added as a part of the header image, which is regenerated automatically each time the score changes.
- Perhaps the most important change was the addition of HTML access keys. Each of the icons in the top header are assigned an access key from 1 to 6, which allows users to easily access the most commonly-used pages on the site with a single keypress from any other page. To remind users which icons are assigned to which access keys, the access key is shown on a red circle in the lower, right-hand corner of each icon. Since the icons are all generated by scripts, the access key assignments can be easily changed as icons are shifted and added or removed.

One unfortunate side-effect of having a nice navigation header at the top of every page is that most mobile browsers will make you navigate through all of the links before you can access the content of the page. To solve this problem, the PHP function that displays the header for every page also creates an empty node link on every page immediately underneath the header and assigns it an access key of zero. In this manner, frequent users of the site can avoid the annoyance of wasting clicks on pages by simply pressing zero to navigate to the start of the content that interests them.

Finally, the site map was pared down to remove links that were rarely used, and to merge other links into the design of other pages so as to minimize the number of "menu" pages and maximize the number of content pages. For instance, there was previously a menu of links for team comparison, home player, and away player statistics. This menu was removed, and the statistics icon link goes directly to the team comparison page, which has links for each team's player statistics at the top.

One additional problem with previous usage of the system was the fact that the data which populated the pages was constantly changing, but there are no auto-refresh features in the site because the wireless bandwidth is too precious to waste on largely unnecessary page reloads. However, now that the live-updating pages of the site have the same access keys on every page, it is simply a matter of pressing a single key to refresh any live page.

III. Important System Issues

A. System Infrastructure

The backend infrastructure of the system resembles that of Figure 3. It is equipped with 20 Cisco Aironet 1200 Access Points covering 4 floors of the pavilion as well as the stands. There is also a Vivato long-range 802.11 base station covering the "tailgating" areas north of the stadium.

Videos are captured live with Final Cut Studio 2, converted to several formats, and transferred automatically to the Apache/ODSS server. Users access links to the videos and other content through the Apache/MySQL server. Videos are then distributed on demand from the Apache/ODSS server as an HTTP-based download or an RTSP stream. Statistics are imported automatically from standard XML stats files provided by the official statisticians at each game. The play-by-play text pulled from the XML stats files is then used to label the action in each video.

B. Mobile Web Application Design

Design considerations for the front-end of the eStadium system were covered in Section II-C.

1) Coding Architecture: A key component of any website is the framework chosen for development. eStadium uses a custom PHP-based framework designed to maximize modularity, reusability, maintainability, and development speed. The framework is based on the common 3-tiered architecture of Database, Logic, and Presentation layers.

The database layer is largely an object-relational mapping protocol in which database *Entity* and *Query* objects are automatically written by PHP scripts which parse the CREATE TABLE statements for the database structure. The only manual database coding required by the programmer is to write functions to find unique lists of ID numbers.

The logic layer is housed in the index file for each application directory, and is responsible for the in-depth processing of input data and creating output data for each page. This

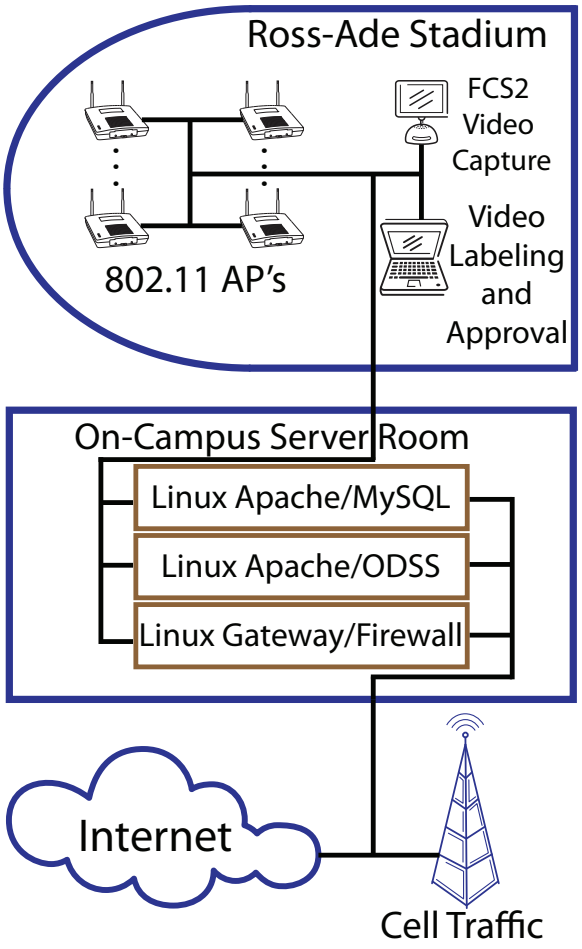

Fig. 3. eStadium infrastructure layout. "FCS2" stands for "Final Cut Studio 2," and ODSS stands for "Open Darwin Streaming Server."

data is then passed to the presentation layer's *template* files by simply using global variables and including the template files at the bottom of the logic files. The template files are then responsible for sending all HTML code to the user's browser.

C. Mobile Video Delivery

1) Device detection: Device detection is a key goal in any mobile web application. Given the non-standard support for delivering video to mobile devices, device detection is doubly important for any mobile video-based websites.

There are two components to device detection: device type identification (DTI) and device capability discovery (DCD). DTI is usually performed by analyzing the HTTP user agent string sent by the device. It can also utilize a technique which uses Javascript's ability to return the screen dimensions on an automatic page refresh. Unfortunately, many mobile devices do not have Javascript enabled, causing most mobile users to see the "You could not be redirected" screen rather than the homepage of the site.

DCD is more complex because, while it may be clear that a connection string containing "blackberry" is obviously from a Blackberry device, you must have access to detailed reviews or, in many cases, physical access to all types of Blackberry's in order to determine if a particular video delivery method works on that device. User agent string/device capability databases such as the WURFL project [15] are currently the best methods for solving both device detection problems.

For the particular problem of delivering eStadium video replays, we are thus far limited in identifying a particular device to those devices for which we have had physical access. We then assign a video delivery method based on our experience. If an un-handled device type connects to the site, it defaults to an HTTP download of the small MPEG-4 video, which seems to be the most widely supported of the formats. As discussed in Section II-C, it is vital in any web application that depends on auto-detection to allow users to manually override any incorrect identifications that the system may provide. These manual interventions by users can be recorded and used later as a free database of capability information. We are planning to experiment with other solutions such as WURFL for more a more reliable detection algorithm.

2) Video Delivery: Since the eStadium application is web-driven, delivering different video types is simply a matter of constructing video links whose href field starts with "http://" or "rtsp://", and contains the path to the proper encoding. The video server itself is able to provide both streaming and download capability, and therefore does not need complicated message passing from the web application.

D. Real-time Data Collection and Production

The live production component of the eStadium site has undergone many improvements to provide the best user experience with minimal errors and simple maintenance. The set of backend importing scripts and administrative web pages are really the core of the eStadium application.

1) Live Statistics Capture: The first component of the live data collection is the automated importing of live statistics. The official statisticians for each game copy a standardized XML file to a directory on the eStadium web server immediately when the file changes using SFTP. When our server detects a change in the XML file, it calls a PHP script that parses all the information and mirrors the state of the XML file in the database.

While that process sounds relatively simple, in reality it suffers from several difficult problems: race conditions, incorrect statistics, corrupted files, and play-by-plays that are later corrected after already being associated with a video. To solve the problem of race conditions and corrupted files, the importing script first reads the entire XML file into memory and checks for the ending XML tag. If the ending tag is found, it is assumed the file is complete and not corrupted. This file is then written to the disk as the last known good copy.

The copy file is then parsed into arrays of various types of Entity database objects. Each of the arrays are then sorted according to various object comparison functions. The equivalent arrays are also pulled from the database to compare with the new data. A fast algorithm was written to mirror the XML data

in the database by exploiting the fact that both the database and XML arrays are sorted according to the same sorting function.

The problem of play-by-plays changing in the XML file after already being associated with a video is difficult because there is no means for uniquely identifying plays in the XML file. Therefore, plays are simply marked *inactive* in the database rather than being deleted when they no longer exist verbatim in the XML file. Additionally, when a play is associated with a video, both the playid primary key and the play text for that play are stored with the video. In this way, since the person associating the videos already made the determination that the play text accurately described the replay video, no further intervention is necessary when plays are changed.

2) Video Capture and Labeling: Videos are captured from an analog video feed which is identical to the feed that is displayed on the Dactronics video board. This feed is captured using Final Cut Studio 2, and exported to the 3 encodings discussed in Section II-C. The video files are organized into folders according to quarter, and named as sequentially incrementing integers. To support multiple video angles for the same play, the filenames use the same incremented integer followed by an "_N", where N is the angle number. To prevent lockups due to network glitches, the files are exported to a local directory on the video capture computer. This computer runs a simple bash script to watch the local directory for changes, and then uses the rsync utility to copy the new video files to the video server.

The video server is simultaneously running a script that watches its main video directory for changes. When it detects that a new video has appeared, it runs a PHP script to mirror the directory listing in the remote web server's database.

An administrative web page shows a listing of all new videos along with a drop-down list of possible play-by-play text that can be associated with that video. A person watches this page and associates the videos when the proper play-by-play text arrives. Given that it can take several minutes for the official play-by-play text to be decided upon by the statisticians, the option to "Approve" the video is provided that will allow the video to show up for users immediately with the pre-determined play text "Recent play." In this way, the time to the web for most plays is less than 30 seconds from the end of the play.

IV. Future Work

The possibilities for the eStadium system are consistently broadening. Current research for future applications includes using wireless sensor networks to track crowd density to provide fans with information such as line lengths and congested pedestrian areas. Combined with crowd density estimation, novel tracking devices are being developed for emergency personnel to create a next generation safety, security, and disaster response system. There are a multitude of user studies yet to be completed. Also, better bandwidth usage via methods such as predictive caching, multi-casting, and cross-layer network optimizations will further improve the system's scalability. We are also currently expanding the application to support additional sports beyond football.

V. Conclusion

The goal of the eStadium project is primarily to provide a convenient mobile web application that enhances the live game experience for attendees of Purdue home football games. This application provides information such as live statistics and on-demand instant replay videos delivered to any mobile device in the stadium. Many obstacles have been overcome to develop a maintainable, adaptable, reliable system which requires minimal personnel to operate. eStadium continues to innovate with many exciting new projects planned for the future.

Acknowledgment

This work has been supported by research grants and gifts from Cisco, HP, Intel, Motorola, Raytheon, and Verizon. The authors would also like to thank many people in ITaP and Purdue Intercollegiate Athletics, and the students on the eStadium VIP team, for their many contributions to and support of this project.

References

[1] M. Fitzgerald. (2007, Nov.) Mobile web: So close yet so far. [Online]. Available: $http://www.nytimes.com/2007/11/25/technology/25proto.html$

[2] X. Zhong, H.-H. chan, T.-J. Rogers, C. Rosenberg, and E. J. Coyle, "The development and estadium testbeds for the research and development of wireless services for large-scale sports venues," in *2nd International IEEE/Create-Net Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities: TridentCom 2006*, Barcelona, Spain, Mar. 2006.

[3] R. J. Glotzbach, E. Coyle, and N. S. Bingham, "e-stadium: Wireless football infotainment applications," in *Proc. ACM SIGGRAPH 2005 Annual Conference and Exposition*, Los Angeles, CA, Jul. 2005.

[4] X. Zhong and E. Coyle, "estadium: A wireless 'living lab' for safety and infotainment applications," in *ChinaCom 2006*, Beijing, China, Oct. 2006.

[5] ICANN and mTLD Top Level Domain Ltd. (2007) .mobi registry agreement. [Online]. Available: $http://www.icann.org/tlds/agreements/mobi/registry-agmt-mobi-01jan07.htm$

[6] T. O. H. Alliance. (2008) Android - an open handset alliance project. [Online]. Available: $http://code.google.com/android/$

[7] Yahoo.com. (2008) Yahoo mobile's go 3.0. [Online]. Available: $http://mobile.yahoo.com$

[8] A. Z. Cuneo. (2008, January) More football fans hit espn's mobile site than its pc pages. [Online]. Available: $http://www.adage.com/digital/article?article_id=122885$

[9] (2008) Nfl mobile. [Online]. Available: $http://www.nfl.com/mobile$

[10] (2008) mspot. [Online]. Available: $http://www.mspot.com/$

[11] A. Press. (2006, November) Cisco pitching high-tech ballpark to mlb's oakland a's. [Online]. Available: $http://www.foxnews.com/story/0,2933,228682,00.html$

[12] G. Torres. (2006, November) Dolphin stadium scores with world's largest wireless pos system powered by belair networks. [Online]. Available: $http://www.dolphinstadium.com/content/pressrelease.aspx?id=60$

[13] T. Booth. (2007, July) Nintendo tests ds lite as fan network at safeco field. [Online]. Available: $http://seattletimes.nwsource.com/$

[14] Dev.Mobi. (2007) Mobile web developer's guide. [Online]. Available: $http://www.dev.mobi$

[15] (2008) Wurfl: The wireless universal resource file. [Online]. Available: $http://wurfl.sourceforge.net$