Sistemas e Tecnologias de Informação

Atas da 8ª Conferência Ibérica de Sistemas e Tecnologias de Informação
Lisboa, Portugal
19 a 22 de junho de 2013

Vol. I – Artigos

Editores
Álvaro Rocha
Luís Paulo Reis
Manuel Pérez Cota
Marco Painho
Miguel Castro Neto

Artigos da Conferência
Artigos dos Workshops

SGaMePlay 2013 - Third Iberian Workshop on Serious Games and Meaningful Play
WISA 2013 - Fifth Workshop on Intelligent Systems and Applications
TICAMES 2013 - First Workshop on Information and Communication Technology in Higher Education: Learning Mathematics
A Field, Tracking and Video Editor Tool for a Football Resource Planner

Pedro Rodrigues
Institute of Engineering
University of the Algarve, 8005-139 Faro, Portugal
pedroarodrigues@gmail.com

Pedro Cardoso(1,3) and J.M.F. Rodrigues(1,2,3)
(1)Institute of Engineering,
(2)Vision Laboratory and (3)LARSyS
University of the Algarve, 8005-139 Faro, Portugal
{pcardoso, jrodrig}@ualg.pt

Abstract— The current Football Resource Planning (FRP) systems integrate both internal and external information management across the entire professional football organization, including features to gather automated information from the teams, not only on the competition part but also on the preparation. In this paper, we present three of those tools from the Footdata’s web/cloud FRP system: Field Editor (FE), Tracking Editor (TE) and Video Editor (VE). The FE provides the coach with a platform where he can represent the tactical movements for his team within an animated context. On the other hand, the TE allows the interaction with the Footdata tracking system. Finally, the VE tool offers the possibility for a coach to watch, cut and analyze football videos.

Keywords: information system, web tools, football resource planning, open source technologies, HTML5.

I. INTRODUCTION

Football is a sport where everyone that is involved with it (e.g., players, staff and agents) make all the efforts aiming for perfection. Not only the players need to show their skills on the pitch but also the coach, and the remaining staff, need to have their own tools so that they can perform at higher levels. Granting both parts a multi-functional information system (IS), with the objective of minimizing the adverse effects from the most critical and sensitive points of football, is then crucial. A match analysis, for example, can generate a huge amount of data. Consequently, it is very important to have a way to process that data providing only the most important information to the coach, as soon as possible. Another example is based on the observation that nowadays most of the football teams have players and staff from different countries in their roster, which can be an obstacle to the correct understanding between them all.

Footdata [1] is a project in development by Inesting, S.A., the University of the Algarve and the football coach Domingos Paciência. The goal is to build a new multi/web-platform product for football, based on information technologies and communication systems, which integrates two fundamental components of the football world: i) a social network, which provides the typical features, and ii) the professional component, which can be considered as a FRP, featuring a system for acquisition and processing of information to meet all the football management needs. The latter component includes an automated platform to gather information from the teams. This platform will be based on a system that will process live images acquired on-site, or images gathered from recorded or broadcasted matches. The processing of this information goes beyond the traditional statistical data compilations, extracted from automated observation of the teams actions. One of the objectives is to add features which will allow the analysis of the football match structure, allowing rationalization and optimization of the team’s actions regarding the occupation of spaces. However, the main goal is to automatically collect information and on-the-fly alert the technical staff about specific events. All the above should be presented in a web (browser) environment, accessible from a personal computer or a mobile device (smartphone or tablet). All the application structure is defined and supervised by the football coach Domingos Paciência.

In this paper, we present the proof of concept of three different tools for the Footdata project: Field Editor (FE), Tracking Editor (TE) and Video Editor (VE). The main contribution is the conjugation of a set of cross-platform, cross-browser web technologies that makes possible the access to an online FRP as well as the foundation for such implementation. There are a few commercial systems that partially integrate some of the components of this project (e.g., Kizanaro® [2], Tracab® [3], Prozone® [4], Amisco® [5], and SportVU® [6]). We can also find in the literature examples that explore some aspects of the project, like video technology for coaching [7] and computerized video analysis of football [8]. Even so, none of those integrate all the technologies and goals of the Footdata project. Please refer to Section II for a more comprehensive review of the literature and a discussion about the above mentioned tools.

The rest of this paper is organized as follows. The structure of the Field, Tracking and Video Editor tools is presented and discussed in Section II; Section III describes the implementation of the Field Editor tool; and Section IV approaches the implementation of the Tracking and Video tools. We conclude, in Section V, with a small discussion and future work.

II. THE STRUCTURE OF THE FIELD, TRACKING AND VIDEO EDITOR TOOLS

The Footdata project integrates a large number of tools. Those tools are all browser based due to the imposed project specifications [1]. Here we present in more detail three editors, all directly or indirectly related with the video features. For a
better understanding of the exposed concepts, the developer’s version design examples are presented (please refer to [1] for more details about the Footdata interface). The first interface is a Field Editor (FE) tool which targets the following points: (1) provide the coach with a web platform where he can easily represent players movements to be applied in the training sessions and in matches; (2) provide an animation feature to help understanding the coach intentions regarding a certain play or movement; (3) provide a way to extract data from a represented movement so it can be processed and analyzed later by matching the tracking obtained from a game. The second interface is a Tracking Editor (TE) tool that accomplishes the following: (1) help setting the Footdata tracking system by providing essential information, like the limits of the pitch, so that the tracking system can enhance its performance; (2) assist the Footdata tracking system by correcting, whenever necessary, the players numbers (this can occur after a camera zoom or a large concentration of players in a particular spot); (3) provide an alert feature which gives key information regarding important events. The third interface is the Video Editor (VE) tool based on the following objectives: (1) provide the staff with a way to easily analyze and get important statistical information from a certain match with the help of a video editing and manipulation tool; (2) provide an interface to make presentations; (3) provide tools for drawing over the video images as well as writing and player dragging.

There are many football applications with similar functionality to the one we desire for the Field Editor tool. Examples are: Kizanaro Playmaker® [2], CoachFX® [9], Easy Animation® [10], eSpor Software Soccer eAssist® [11], Tactics Manager Software® [12], and TacticalPad® [13]. Some of the most common limitations found in the majority of the mentioned applications are: (1) the lack of elements and options when the coach needs to represent different plays as close as possible to reality; (2) some of those applications are not web applications; (3) some applications are unable to run in all platforms including mobile; and (4) a part of them are not as intuitive as they should be.

Regarding the video tool subject, we can name Kizanaro Studio® [2], Prozone Playback® [4] and Sportstec SportsCode © [14] as the most used ones. Again, part of these technologies are not web applications and don’t run in the wanted platforms (personal computers and mobile systems). Moreover, some of them don’t use the capabilities offered by the Computational Vision domain to provide interaction between the user and the video he is analyzing.

Figure 1 shows the Footdata structure scheme [1], indicating the applicable modules for the FE (represented in blue), for the TE (represented in green), and for the VE (represented in red). Figure 2 resumes some of the features of the three interfaces discussed in this paper. We must refer that these three modules are only a part of the available tools in the Footdata project.

As already mentioned, the three editors will be delivered as web interfaces structured in HTML5, styled with Cascading Style Sheets (particularly CSS3, which is the latest standard for CSS) and its behavior controlled by JavaScript. In fact, JavaScript probably has the most important role, since it is the technology that implements most of the interactive features of the web pages that we will be developing. JavaScript turned out to become the default browser side scripting language, despite the bad reputation it had for almost a decade, since it was created in 1995 [15]. Due to their importance in the implementation of the presented interfaces, there are some JavaScript libraries that we have to take into account at this point, namely: jQuery [27], jQuery User Interface (UI) [27] and jQuery Mobile [28] [29]. The first one can be seen as an extremely valuable library since it makes programming JavaScript much easier. On the other hand, jQueryUI provides a set of UI elements and widgets, built on top of the jQuery library. Finally, jQuery Mobile was specifically designed to help developers in creating applications that can be accessible from smartphones and tablets, without putting aside desktop devices. 

Fig. 1 – Footdata structure scheme [1] indicating the applicable modules for the Field Editor (represented in blue), for the Tracking Editor (represented in green) and for the Video Editor (represented in red).

Fig. 2 – Summary of the most relevant (web) interfaces features.
The technologies mentioned above are the foundation for the development of each interface. However, it is obvious that one interface might need some other technologies. So, in the next sub-sections we will explore each interface individually and occasionally point out particular technologies needed only for that interface.

III. FIELD EDITOR TOOL

Until recently, graphic rendering on the web was only possible using third party plugins such as Flash or Silverlight. With the development of HTML5, some functionalities can be achieved without the need of any plugins, not only for the graphics matter itself but also for the audio and video, as we will see later. The two major technologies in the web graphics branch are the HTML5 Canvas and the Scalable Vector Graphics (SVG). According to [16], SVG enables rendering in high resolution at any level of magnification due to its vector nature, unlike Canvas where the basic elements are pixels, and therefore it does not differentiate between the objects in the graphic, which means that every time we want to make a change, the whole graphic must be redrawn (for more details see [17] and [18]).

While in the process of evaluating the technologies, with the objective of selecting which one to use for the elaboration of the presented FE tool, we found us between a dilemma which was to use one of the mentioned technologies separately or as a conjugation of both. From our point of view, this application must be: (a) dynamic, i.e., we want to make animations with the shapes we draw; (b) it must also be user-friendly, and (c) functional, i.e., it has to provide an easy way to add and edit shapes and some functions like drag, resize, recolor, rotate, etc. Also important was to have fast and smooth animations despite the number of shapes. Therefore we reached the conclusion of using Canvas, but still keep the door opened for a possible connection to SVG whenever needed. The steps so that a beginner in Canvas can understand the basics, but also more complex features such as data visualization, game development and 3D modeling, are provided in [19] (see also [20] for Canvas/SVG integration).

For such a complex tool (the FE tool) to be implemented in Canvas, it was decided to use an appropriate JavaScript framework. The use of JavaScript frameworks and libraries alongside with Canvas is motivated by the native low-level API. In other words, if our goal is to draw simple shapes then there will be no problem but, if we want to implement many animations and complex shapes, it will get harder and harder. Another reason is based on the fact described earlier: HTML5 Canvas is a pixel-based technology, therefore, the shapes we might draw in the future will be just seen as colored pixels and not as objects we can interact with. With some frameworks we can get an object model structure to work with those shapes.

Nowadays, there are many powerful frameworks and useful libraries so that the developers can take full advantage from the HTML5 Canvas. We can list some of them: EaselJS [21], Fabric.js [22], KineticJS [23], and Paper.js [24]. At this point, we decided to use KineticJS, not only because it offers a suitable structure with a stage, layers and shapes, allowing us to group different shapes and perform particular actions to that group, but also because KineticJS is one of the few which supports mobile events (e.g., touchstart, touchmove, touchend, tap, and dbltap). KineticJS also supports SVG paths which can turn out to be very important later when integrating with the other Footdata modules. Moreover, KineticJS fulfills another important request which is the handling of a large number of concurrent shapes without losing performance (e.g., see [34]).

That said, the next point to look at is the implementation of the major football features with custom JavaScript code. The base structure we architected using KineticJS is schematized in Fig. 3. As we can see, the main stage will have two different layers: an objects layer and a tooltips layer. The first one aims to be the main layer for all the shapes within our FE tool. On the other hand, the tooltips layer will be the one responsible for a structure to show the information regarding a certain shape (e.g., id, name), by displaying the individual tooltip near its position. In the objects layer, we will have a KineticJS group for each player, putting together its circular representation, its number and its movement arrow. The movement arrow is also a group which includes the arrow line and the arrow triangle shapes. Every other item, such as the ball or a football training cone that doesn’t need the change of properties (e.g., color), are treated as Portable Network Graphics (PNG) icons.

![Fig. 3 – KineticJS layer and shape structure for a football context.](image)

This structure might suffer a few tweaks in the future, but until this date it is the one that we found to be the most appropriate in the football point of view. For example, each football coach has his own methods and symbology, which makes important that the default ones, provided by the system, are understandable by the majority of the coaches but they should also be as customizable as possible.

An example of the FE tool is shown in Fig. 4 (top row), where we can see a set of players (circular shapes) from two distinct teams. In this case, the objective is to represent the statistics of the red team and the statistics of the blue team. By double-clicking in each player we can edit its number, name and color. As we can see in the figure, it is possible to represent a tactical movement regardless of its complexity. Still in Fig. 4, but in the bottom row, it is shown in field sections a very basic animation: the red and the blue players move according to the
arrows, and the ball goes into the goal. For future work in this interface, we pretend to deal with more complex animations, which will be based in each player movement arrow (as represented in Fig. 4 - top) with a certain start time and duration and the coding of specific football events. This last point will be extremely important because those coded football events will be compared to the analog ones that are happening in a football match, i.e., a comparison between the virtual/optimal events (statistical data gathered from the FE tool) and the real events (statistical data gathered from the tracking system) will take place and the coach will be notified if there is a significant difference between both.

![Image](image.png)

**Fig. 4** – In the top, the Footdata Field Editor tool, in the bottom an example of a very basic animation shown in field sections: the red and the blue players move according to the arrows and the ball goes into the goal.

IV. VIDEO AND TRACKING EDITOR TOOL

Before deepen the implementation of the next two tools, we notice that the Video and Tracking Editors are in the same section due to the fact that both use the HTML5 video feature.

When talking about video on the web, we can point out two major phases: before HTML5 and after HTML5 arising. Before HTML5 and similarly to what happened to the graphic component, playing audio and video content was only possible using third party plugins, such as Adobe Flash. Web developers could only include video in a web page using the `<object>` and the `<embed>` elements. Nowadays, HTML5 has an element called `<video>` that allows us to display video directly in a web page without using any other plugins. Despite HTML5 Video tag being supported by the most recent browsers, we have to take into consideration that legacy browsers might not support it. Internet Explorer 8, for example, is one of the browsers that does not support the HTML5 Video tag and in this particular case it would be useful to have a fallback to Flash (see [25], [26] and [30]), so that we could minimize the losses to those users. It is true that some functionality will be lost, since the main assumption was to use the HTML5 Video, however those users would still have a playable video.

After overcoming this problem, one can notice that not every video format can be played in all browsers. In other words, we have to take into account the video and audio formats supported, which differs from browser to browser. From our analysis, we concluded that we need to have a combination of two different formats so that we can cover the great majority of the current platforms and browsers. According to [26], if we need to setup a web site from scratch then the best choice is to use MP4 (video codec: H.264 / audio codec: AAC) and WebM (video codec: VP8 / audio codec: Vorbis), since WebM tools are quickly improving. It is useful to access [26] for more details about video and audio codecs as well as how to convert videos to the specified formats (MP4/M4A, OGG/OGV and WebM).

Once again, JavaScript will play an important role in order to get more functionality from the HTML5 Video. Some developers might find useful to build their own controls for the video (you can find some examples of custom video players in [25], [26] and [30]), because the native API provided is a bit rudimentary. Also, this gives the chance to design the controls with CSS3. Either way, if your objective is just to place a simple video player in your web page regardless of the design then it fits properly.

If you still want a more complex video player but somehow you are not able to code it, there are some important JavaScript libraries worth to consider: Modernizr [30], HTML5media [30], MediaElement.js [30], Video.js [30], jPlayer [31] and Popcorn.js [32]. This last one deserves special emphasis, because it allows the media to control some elements of the web pages. For example, Popcorn.js supports a large set of features from the web such as Facebook, Twitter, Wikipedia and more, allowing an integration of those services in our website controlled by media. In a football platform, we thought a coach might need to take notes when watching a football video and assign them to appear in later visualizations, in a specific time of the video and they would stay visible for a certain amount of time. Popcorn.js provides the solution with the Popcorn Footnote.

In Fig. 5 top, we can see the VE tool, with a video player coded and customized by us with a few more functions than a generic web video player such as forward, backward, restart and speed. It is also possible to capture specific moments of the video by drawing the current video frame into a HTML5 Canvas which can be very useful for a coach to save a certain movement and/or draw something on it. The most important improvements we want for this tool in a near future are the possibility of cutting and saving automatically small portions of a video, called football compact, and add more functions to the video player, similar to the ones we can find in the offline software regarding the video editing domain.

Turning our attention to the TE tool (Fig. 5 - bottom) it is important to mention that the Footdata tracking system offers the possibility to work in two ways: track the players directly on-site from a camera positioned on the stadium stands and track the players from a previously recorded match or from a television broadcast (detailed information about how the
functions. A simple video player will only allow play and pause functions. Another important point is the initial settlement of the tracking system, e.g., it will be extremely advantageous to provide the pitch limits to the tracking system, by clicking in the video. In this case, the data, i.e., the points coordinates and frame time will be passed to the tracking system. Again, like in the field editor tool, the structure of the video and tracking tool might suffer a few tweaks in the future.

V. CONCLUSION

The three interfaces we presented in this paper will help to bridge the gap between the football coaches and the complex multi-platform information system for football discussed in [1]. Although the interfaces are implemented in multiple languages, the proof of concept of the three interfaces in Portuguese is depicted here (refer to [1] for an initial graphical design of the FRP). There is no doubt that HTML5 provides a competing open standard in the multimedia domain with the Canvas and Video elements. JavaScript can expand our horizons in developing a cross-browser, cross-platform, interactive application.

It is quite difficult to compare this work with similar publications. The main reason is the lack of publications in this area. Nevertheless, there are some works that are worth mentioning, like [7], [8], [16], [28], and [29]. None of these publications, including the commercial products mentioned in Section I and II, present all the features of our tools. For example, some of the features that are mentioned by our consultants as missing in the majority of the above projects, are the elements and options that are needed to represent different plays as close as possible to reality.

As future work, we intend to continue developing the mentioned interfaces so that they can fulfill the requirements of a continuously demanding world that is football. This leads us to another important fact, which is the communication between the TE interface and the tracking system (implemented in C++). We are studying the best approach and so far we plan to use websockets (more information about the HTML5 WebSocket and real-time web can be found in [30] and [33]), respectively).

ACKNOWLEDGMENT

This work was partly supported by the Portuguese Foundation for Science and Technology (FCT), project PEst-OE/EEI/UI0009/2011 and project Footdata QREN I&D&T, n° 23119. We also thank to project leader Inesting, S.A. [www.inesting.com], our colleague Carlos Gomes for all the advices and football inputs, and the consultant football coach Domingos Paciência.

REFERENCES

[27] jQuery Community Experts, jQuery Cookbook, O’Reilly, 2010
Fig. 5 – Footdata Video Editor tool in the top, and in the bottom the Tracking Editor.