Enrichment of Raw Sensor Data to Enable High-Level Queries*

Kenneth Conroy[1] and Mark Roantree[2]

[1] CLARITY: Centre for Sensor Web Technologies, Dublin City University
kconroy,mark@computing.dcu.ie,
[2] Interoperable Systems Group, Dublin City University

Abstract. Sensor networks are increasingly used across various application domains. Their usage has the advantage of automated, often continuous, monitoring of activities and events. Ubiquitous sensor networks detect location of people and objects and their movement. In our research, we employ a ubiquitous sensor network to track the movement of players in a tennis match. By doing so, our goal is to create a detailed analysis of how the match progressed, recording points scored, games and sets, and in doing so, greatly reduce the effort of coaches and players who are required to study matches afterwards. The sensor network is highly efficient as it eliminates the need for manual recording of the match. However, it generates raw data that is unusable by domain experts as it contains no frame of reference or context and cannot be analyzed or queried. In this work, we present the UbiQuSE system of data transformers which bridges the gap between raw sensor data and the high-level requirements of domain specialists such as the tennis coach.

1 Introduction

Many new applications employ sensors or networks of sensors to automatically monitor and generate reports and analysis across domains. Increasingly, elite sports men and women are monitored to determine the effects of various exercises on their bodies. In football matches, it is often common to track the distance covered by players by inserting GPS devices in their footwear. In almost every case, sports coaches will watch video recordings of previous matches to look for player faults and determine strengths and weaknesses. The problem with this effort is that it is extremely time-consuming as the coach must search for key moments in the match. In the case of tennis, this problem is exacerbated as matches can be up to five hours in length, and coaches will often have many players in their charge. What these coaches require is an automatic analysis of tournament and practice matches with the possibility to query to retrieve key segments. In more advanced scenarios, they require data mining functionality to analyze matches based on duration of points and games, analysis of defensive against attacking play; and details on games where the player had service. Our

* This work is supported by Science Foundation Ireland under grant 07/CE/I1147

research involves a collaboration with tennis coaches in Ireland to determine if it was possible to capture game analysis automatically and provide some form of query interface for coaches, to facilitate extracting the type of information described above.

In a generic sense, we present a framework and methodology for automated processing of wireless sensor data so that it can be queried using a standard query language. The major benefit of our research is to provide queryable information to knowledge or domain workers using the sensed data. Our test application, developed with Tennis Ireland [2], allows coaches to query this information immediately after the match has finished. The coach uses this data to monitor performance and modify the behavior of the player for subsequent sessions or games.

Our contribution is in the development of a framework and data management layer, with algorithms, to automate the analysis of a ubiquitous sensing environment. Specifically, we record a tennis match using location sensors and provide a queryable analysis of the match using a metadata approach and a sert of data management processors. This research required the development of a series of algorithms to understand and interpret player movement according to the rules of tennis. To test our system, we deployed *Ubisense* [3] on an indoor tennis court and monitored a series of matches, processed the sensed output and provide a query interface to extract end user requirements. Our experiments section shows the levels of accuracy we managed to achieve.

The structure of the paper is as follows: Section 2 provides an overview of the sensing environment, lists the requirements of the end user and describes the difficulties in providing these requirements. Section 3 explains the UbiQuSE architecture and our framework for automatically process wireless sensor data. Section 4 describes domain-specific aspects of our system using the tennis case study. Section 5 outlines the result of experiments. Section 6 presents the related research, and finally, section 7 provides conclusions.

2 Problem Description

In this section, we motivate our work through a detailed description of the problem. This section will provide an overview of the sensing environment, and the requirements of the end user. The aim of this discussion is to illustrate the significant gap between the sensed data and the requirements of knowledge workers in various domains.

2.1 Sensing Environment

In a ubiquitous computing environment we have a space and a selection of basic sensor data. Our space is equipped with a Ubisense setup, where sensors are fitted on all sides of the space. Portable *ubitags* are then held by a participant and the sensors track its movement through the space. The raw data output is primitive, consisting of only three distinct properties: Ubitag ID, timestamp and 3D

	Query
1	Return all serves for Player 1 in Game 1
2	Return the time of each period of play that resulted in a point being scored in Game 1
3	What zone is Player 1 is in at game time 8000ms?
4	What is the duration of the rally prior to the point at 05:02 being scored?
5	What is the average duration of a rally resulting in a point scored in Game 1
6	How many points were scored in Game 2?

Table 1. Coach Queries

location (x,y,z). As a result, raw data may only be queried for basic information and the lack of contextual information results in the data being meaningless to users unfamiliar with the configuration and physical setup. More complex queries cannot be performed without a domain specific application being built to address particular needs.

There is a significant gap between the raw sensor data generated by the ubiquitous environment and the query and analysis needs of the coach/domain specialist. To bridge this gap, a system of both generic and domain-specific layers is required to provide meaning to the data and solve complex queries. This system uses generic functions to first allow basic queries in XPath[14]/XQuery[15], the query languages for XML, and then allow an interaction between structurally enriched sensor data and a domain specific context database. The system consists of three layers, the sensing layer which provides the raw data, the process layer which applies the data processing; and the storage layer where context is provided and enriched files are stored.

To test our system in a real-world scenario, we collaborated with Tennis Ireland, the governing body of tennis in Ireland, working with a national coach to meet their analytical requirements after tennis matches. An indoor court was fitted with the Ubisense sensor network and eight low definition cameras. An experiment was setup to sense the movement in a number of training games where each player was carrying a ubitag. The data is collected on a nearby server. The resulting data files consist of low-level data, specifically a players position in space at a certain time and are streamed in textual format. We will later show that this data is unsuitable for any form of automatic analysis or queries.

2.2 User Requirements

The sport coaches need to query the data for certain events, such as a players serve, a point being scored or the end of a particular game. Some of the queries required by the users - in this case tennis coaches - are shown in table 1. Each query requires information regarding each players position, and analysis of the time spent in such a position.

- In query 1, the coach wants a list of times that a serve was made by Player 1 during Game 1. These times can be mapped to a video of the serve, using

the timestamps, and the coach can review the players technique or decide to ask further queries.
- Query 2 returns the beginning time of a period of play - such as a rally - that resulted in a point being scored. The coach can then review the period of play from serve made to point scored on the corresponding video.
- Query 3 determines which zone a player is in at a certain time.
- Query 4 is a query used to check the length of a rally leading up to a point scored. This kind of information can be useful to the coaches in deciding how tired the player may be and if later actions were effected.
- Query 5 gives the coach a broader view in how points are being scored, quickly in the case of aces or following long rallies.
- Query 6 counts the amount of points scored in Game 1. A small amount of points gives an indication of an easy win by one player.

Using these and other queries, the coach can request a breakdown of the entire match, in terms of its different states (games, points, serves). These queries are not possible when the data is in its raw form, as it lacks the structure and meaning required to identify these events and states. There is a gap between the query requirements and the raw sensor data which we must bridge in order to enable the coach to execute these queries.

This setup illustrates a real-world application where there is a substantial gap between the raw data sensed and the query requirements of the coaches. At present, event detection techniques in both tennis and other sports is primarily tackled by video and audio recognition techniques. These are generally computationally expensive and inaccuracies can develop when players leave the line-of-sight of the camera, as well as poor picture quality and the inclusion of graphics, replays and advertisements on broadcast media. The Ubisense system uses ubitag IDs to ensure no mix up in identification, and the static nature of the environment provides an ideal training environment where complex actions can be detected without noisy data.

3 The UbiQuSE Architecture

In this section we present the *Ubiquitous Query Sensing Environment (UbiQuSE)* architecture which was designed to facilitate both live and offline queries, for both tracking and context purposes. The architecture consists of three broad layers: Sensing, Process and Storage. However, it is the processors of the Process Layer that merit detailed discussion. Figure 1 illustrates the UbiQuSE architecture in the tennis domain.

3.1 Sensing Layer

The sensing layer contains the devices which generate the raw data in a sensing environment. The hardware platform for our current domain application is the Ubisense network. Ubisense generates data based on a participants movement, specifically tracking position in space using triangulation of ultra-wide band (UWB) radio waves. When a participant carrying a ubitag moves, Ubisense detects the movement and the timestamp and location of the ubitag, in (x,y,z) coordinates, are transmitted. Essentially, this is the hardware layer for UbiQuSE and provides the raw sensor data.

3.2 Process Layer

The process layer contains the main processing algorithms and reduces the gap between user query requirements and raw sensor data. There are three processors which when used together will transform the data into an XML format, which is then used by XQuery to express end user queries. The *Structural Transformation (STP)* and *Contextual Transformation Processors (CTP)* are generic and remain unchanged across application scenarios. A separate *Domain Knowledge Enrichment Processor (DKEP)* is required as UbiQuSE is used in different environments. Figure 2 provides a simplified version of the typical enrichment process.

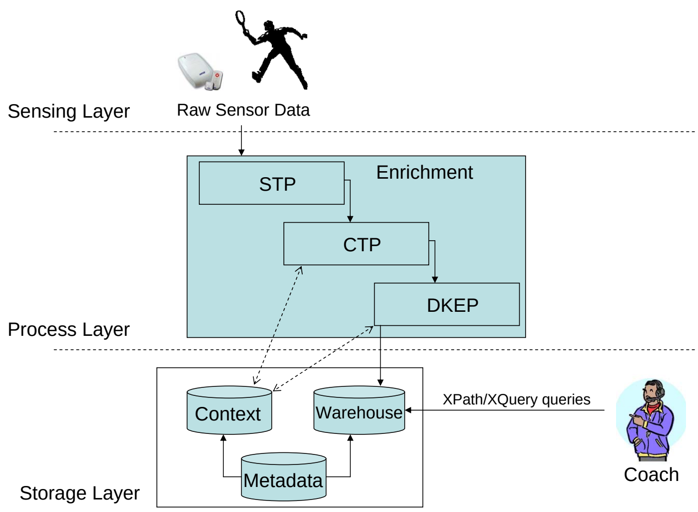

Fig. 1. UbiQuSE in the Tennis Domain.

ST Processor: Structural Transformation The role of the Structural Transformation (ST) Processor is to convert the sensor data from an unstructured stream of data to a structurally enriched XML format. The input for this processor is the raw data stream in text format. The ST Processor is the first step in providing structure and meaning to the raw sensor data. The basic conversion wraps the Ubisense sensor data into <x>,<y>, <z>, <player>, and <timestamp>

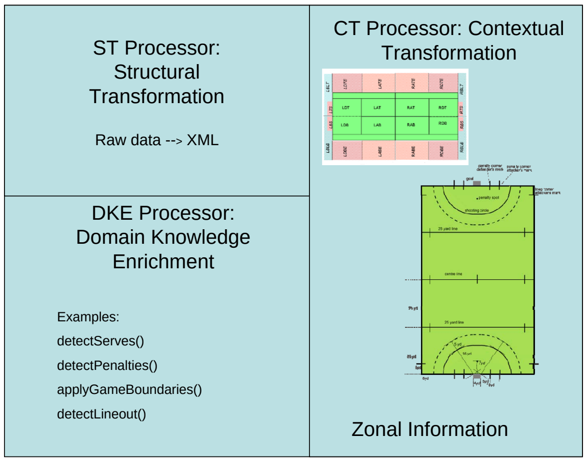

Fig. 2. Sensor data enrichment.

tags which are generic to any Ubisense environment. The enrichment is illustrated in figure 3. Very basic queries, limited to the exact position of a player in space can be determined using XQuery following enrichment. The range of (x,y,z) coordinates will differ based on the size of the application space where Ubisense is setup.

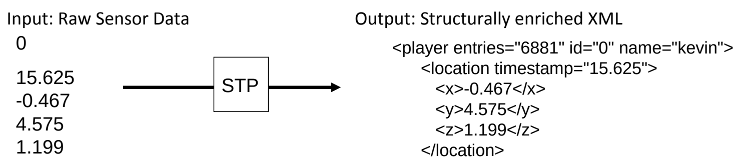

Fig. 3. Structural Transformation

CT Processor: Contextual Transformation Our vision for ubiquitous computing is that all sensed data is associated with a specific *Zone* (as part of a Smart Space). Each activity may have a series of *States* and in the case of a tennis match, these states will be Game Number, Set Number etc. The combination of these Zones and States provide a powerful enrichment to basic sensed data as it permits us to make certain assumptions with varying degrees of confidence. Furthermore, with our metamodel approach, the application of Zone and State information is performed in a generic fashion, standard to any activity that required Zones and/or States.

The Contextual Transformation (CT) processor provides contextual enrichment for sensed data. The input to this processor is the XML stream shown in

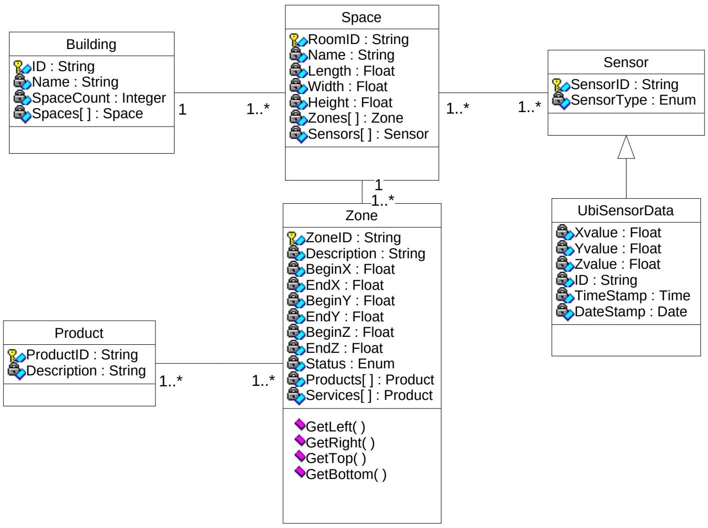

Fig. 4. SmartSpace Metamodel

Fig. 3. This processor is not domain specific but the UbiQuSE framework can still add Zone and State information using the metamodel approach illustrated in Fig. 4. The Zones within the Space are setup as described later in Fig. 6. As a result, we can associate every sensed location with a specific Zone and automatically detect the player's movement for the entire match. This process takes place irrespective of activity or application domain. Both the Zone and State elements have templates to suit application scenarios so that the system remains unchanged as it moves from domain to domain.

The zones are contained within newly created `<zone>` tags, which appear as children to each `<location>` element in the sensor data file. In our tennis case study, the context specific zonal tags include `<special_zone>` and `<side>`.

A further generic operation - `addStates` - is carried out during this stage. This function adds domain tags relevant to future enrichment of the data. The exact tags created depends on the domain, in the case of tennis `<game>`, `<serve>`, `<point>`, `<hit>`, `<receive>`, `<duration>`, `<change_side>` are added. Our metabase contains different State constructs that are added without being populated. The domain specific rules for the content of these domain tags is computed by the Domain Knowledge Enrichment processor. An example of the output file from CTP for tennis is shown in figure 5.

DKE Processor: Domain Knowledge Enrichment The role of the Domain Knowledge Enrichment (DKE) processor is to apply the rules of the particular domain (or sport) to the sensor data to allow the user the ability to return the required query results. The input to this processor is the output from the

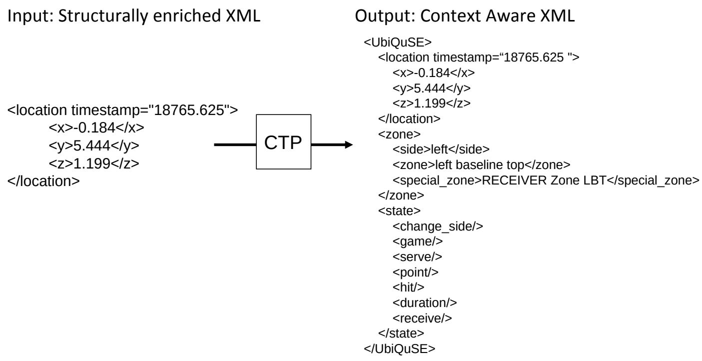

Fig. 5. Contextual Transformation

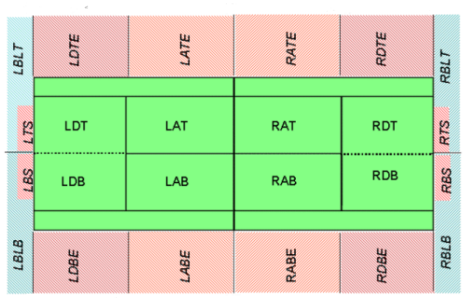

Fig. 6. Tennis court divided into 24 zones.

Contextual Transformation Processor. The output is a fully enriched XML file on which XPath/XQuery expressions can be used to detect complex domain specific states and events. It provides the final step in providing a complete break down of the domain into its constituent states. As this processor changes from domain to domain, we focus a detailed discussion of the tennis match case study in the following section.

3.3 Storage Layer

The storage layer contains both the context repository and the final XML output following enrichment, available for querying. The context repository is accessed throughout processing in order to identify the space being used, its corresponding zonal information and the domain specific rules or expectancies of the scenario being recorded.

4 Tennis Match Case Study

In tennis, the zones originate from the actual structure of the tennis court, as illustrated figure 6, and first introduced in previous works [4] [5]. The game itself has a rigid structure: when a player is serving, the receiving player is usually beyond the baseline on the opposite side of the court. However in practice, the receiver can be closer to the net, or within the baseline, which requires flexibility in our zoning rules. To improve accuracy, the boundaries of the special zones(`<special_zone>`) were altered and a new receiver zone was defined. By zoning the court, we can use the rules of tennis to define a rule set to deduce probable actions during the match.

We applied the rules of tennis [1] to our system and as required by the domain specialists, the complete breakdown of a tennis match was automatically computed based solely on the Ubisense data. Serves, points and games are the key aspects of tennis that were identified by our system. The functions used are `detectServes`, `applyPointBoundaries` and `applyGameBoundaries`, the logic for identifying each of these is explained below. The resulting enriched file is illustrated in figure 7.

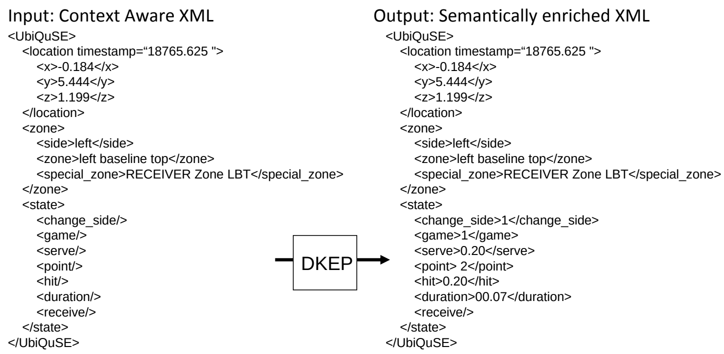

Fig. 7. Domain Knowledge Enrichment Processor

1. `detectServes` We know that if we can detect when a new player is serving, then the match has changed state (a new game or possibly set has commenced). Accuracy in detecting this change in serve is crucial to providing correct results. Thus, the special zones described in Phase 2 were creasted. The previous version of our system suffered from false-positives and manual annotation had to be used to remove such instances. Our current system removes all manual steps by altering overlapping zones close to the serve position, and refining the duration required to flag a serve. The algorithm consists of examining the position of both players position for a specific time period. A lack of consecutive serve event detections by the same player indicates a false positive which is then dropped.

2. `applyPointBoundaries` - Point scoring is identified by checking multiple instances of serves by the same player. As stated in the rules of tennis [1], players who serve from one side of the baseline must serve from the other side following a point being scored (by either player), providing the point does not also result in the end of a game - in which case players switch service. We apply our knowledge of the rules to the data to find points scored. The duration of a point and the time between each point can be calculated by examining the timestamps of the current and preceding points.
3. `applyGameBoundaries` - The game boundaries are based on which side of the court a player stands as well as when each player switches serve. The rules state that every odd-numbered game (1,3,5...) of a set is followed by a change of side and all games are followed by a change of service (from one player to the other).

5 Experimental Data

After the sensor data is processed by the UbiQuSE processors, data in the XML database is sufficiently transformed to enable querying using the XQuery language. The queries listed in table 1 are expressed in XQuery and presented in table 2. In fact, the enrichment of the XML requires simple query expressions to return previously complex requirements. Figure 8 shows after which phase of enrichment each query becomes possible following the addition of the relevant tags and values.

	Query
1	let $c := collection('db/ubisense/trainingFeb10') return $c//player[@id=1] /UbiQuSE/state[game=1]/hit
2	let $c := collection('db/ubisense/trainingFeb10') return $c//state[game=1]/point
3	let $c := collection('db/ubisense/trainingFeb10') return $c//player[@id=1] /UbiQuSE/location[@timestamp=8000]/zone
4	let $c:= collection('db/ubisense/trainingFeb10') return $c//player[@id=1] /UbiQuSE/location[@timestamp=302000]/duration
5	let $c:=collection('db/ubisense/trainingFeb10') return fn:avg($c//player[@id=1] /UbiQuSE/state/duration)
6	let $c:=collection('db/ubisense/trainingFeb10') return fn:count($c//player /UbiQuSE/state[game=2]/point)

Table 2. XQuery implementations of required queries.

Table 3 shows the output of queries 1,2,5 and 6 as these can be easily displayed. The results for queries 1 and 2 are time values in mm:ss format that the event requested occurred (serves/periods of play resulting in a point). The time returned for query 2 is the time of the serve leading to the point, so the coach can watch the period of play from serve to point scored. The result of query 5 shows that the average rally leading to a point was 9 seconds long, and the result for query 6 shows there were 8 points scored in game 2.

Query	Result
1	0.10, 0.25, 0.33, 0.53, 1.03, 1.23, 1.47, 1.75, 2.09, 2.34, 2.51, 3.00, 3.18, 3.27, 3.36, 3.45
2	0.10, 0.33, 1.03, 1.23, 1.57, 2.09, 2.34, 3.00, 3.27, 3.45
5	0.09
6	8

Table 3. Sample Query Output

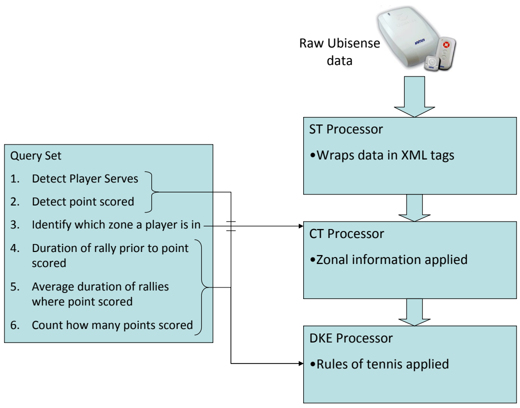

Fig. 8. 3-Stage Enrichment Process.

5.1 Verification of Enrichment

As the data generated by the Ubitags is of varying degrees of accuracy, a player may be incorrectly tagged as being in an adjacent zone. As a result, we incorporated safeguards by slightly extending the boundaries of a serve zone, beyond where a legal serve can be performed. We also implemented a system where outliers are recognized and eliminated. This can help capture instances where the server is almost on the baseline when serving, and as the device is sometimes carried in the pocket it may be registered as being slightly beyond the baseline. In this section, we report on the accuracy of our results as our algorithms make many assumptions based on player location and movement, combined with the semantics of the sport of tennis.

One aspect of the system verification is the accuracy of the Ubisense system itself, prior to using the data to form as the basis for event detection. In order to identify the accuracy of the system, we took a sample set and examined player positions during a number of games. In our setup, inaccuracy of Ubisense is confined to a maximum error of 15cm. The accuracy is best in the centre of the court, greater inaccuracies occur on the periphery of the Ubisense space. Other researchers have shown inaccuracies of up to 48cm on the periphery of their setup [7]. The fringes of the Ubisense range on the tennis court are rarely in use by a player during game time. Retrieving stray balls is usually the condition where a player would be in such a position. We have taken Ubisense inaccuracy into account when devising our algorithms.

The focus of our experiments was on determining the accuracy of the UbiQuSE algorithms in the Domain Knowledge Enrichment processor which detects the key events in a tennis match. In particular, we examine how the system performs in detecting player serves, points scored, and game boundaries. Our experimental data consisted of a number of best-of-5-set training matches between two elite tennis players. Each player carried a ubitag in their pocket and the game was played in a competitive manner. We randomly selected one of the sets for experimental evaluation, used UbiQuSE to enrich the raw data, and visually inspected the video recording to time the exact occurrence for all serves, points and game boundaries. We compared the manual recording of events with that produced by UbiQuSE, with results shown in Table 4.

	Serves	Points	Games
Total	72	44	6
Detected	70(97%)	43(98%)	6(100%)
Missed	2(3%)	1(2%)	0
False-Positives	9(12.5%)	0	0

Table 4. Accuracy Experimental Results

- Out of a total of 72 serves made in the set, 70 were correctly detected by our system, representing 97% of the total. Apart from the 2 undetected serves, we had 9 false positives where we believed serves to be taking place. In previous work [4], where 16 of 87 serves were missed by the system (due to their close proximity to each other), the amount of false positives is lower. The improvements were achieved by refining the special zones for both server and receiver and also changing the time constraints that form part of the algorithm for `detectServes`. Additionally, end-users reported a preference for serve detection over the low number of false positives as these can easily be ignored.
- Regarding point detection, a total of 44 points were scored in the set we examined. All but one of these were detected by `applyPointBoundaries`.

As a result 98% of points are correctly identified by the system. Point detection is in part based on the `detectServes` algorithm and poorly performing algorithm affects the detection of points. The nine incorrectly flagged serves did not have an effect on point detection, however, the point that was not detected was directly related to one of the serves not detected by `detectServes`.
– Game detection is also built on the previous algorithms for point and serve detection and can be effected by poor accuracy. In our experiments, UbiQuSE determined all six games in the set were correctly identified by our system.

6 Related Research

A general model to represent semantics of a smart space based on lower-level location contexts is provided by [6]. Several sensor types, including Ubisense, generate sensor data which is combined with a time factor and a conditional confidence value factor - computed using Bayes Theorem - to identify valid contexts for that situation. Context is described using an RDF triple. Like UbiQuSE, space is limited to where the sensors (Ubisense) can get a signal. However, we have built a model where we can change zonal boundaries depending on a changing domain, or apply to another Ubisense setup. Our system achieves a high level of accuracy using only one sensor type, relying on powerful location and domain dependant rules.

A formal structured format for representing contextual information of a smart space environment is outlined by [8]. They use RDF to define context ontologies, and provide a generic querying mechanism to infer high-level contexts based on rules. Querying is performed using RDF Data Query Language. While this approach does define an infrastructure for smart spaces and represents contexts as easily interpreted semantic markups from which higher-level contexts can be inferred, the context is generally provided by external sensing sources. This is in contrast to our approach where we take previously meaningless raw data from one source (Ubisense) and transform it structurally and semantically in order to allow the detection of complex actions and events. Evaluation of this technique is limited to computational performance rather than accuracy, as the inferences are not complex domain specific actions

In [9], a multiple-sensor approach is described, consisting of wearable sensors including cameras to interpret human actions. Raw data is segmented into meaningful clusters, and then applied with primitive context. A composite layer results in a higher-level context describing several defined interactions of the wearer. It is a real-time system which is implemented in applications such as interactions with robots, and video highlights. Unlike UbiQuSE, a standard query language is not possible and the algorithms involved do not appear to make use of a defined context repository.

The identification of events in sports, and in tennis in particular are mainly based on video analysis [10][11][12]. Approaches make use of audiovisual data to extract relevant features of a video clip. These features when combined with the semantic concepts and game structure, can identify the required events. The performance of such systems varies greatly, ace detection in [10] has a precision accuracy of 65%. In [11] this increases to 93%, however, a net approach in the same system has an accuracy of only 50%. Our system has an accuracy of 100% when detecting net approaches - by detecting players entering the offensive zones. Video recognition can suffer from poor camera angles, poor conditions and is usually very expensive computationally to perform. Our system is relatively lightweight, and offers a powerful querying facility, using XQuery to allow the detection of complex events. Our experimental evaluation has shown a high level of accuracy in detecting complex actions, and in breaking the tennis match into its individual serve, game, point states.

7 Conclusions

In this paper, we presented the UbiQuSE system which provides a data management layer for sensor networks. It contains two generic processors, the Structural Transformation and Contextual Transformation processors, that are generic and operate on any sensor network that captures location and movement within a smart space. These transformers take the raw sensor data and provide both structure and basic semantic enrichment as they place each player movement within some context. The third processor, the Domain Knowledge Enrichment processor, differs across domains, and is used to populate generic data structures and provide higher levels of enrichment. Our goal in creating UbiQuSE was to provide the missing data management layer that bridges the gap between raw data and the high-level query expressions of specialist users. To demonstrate the effectiveness of UbiQuSE, we took a query set specified by the domain specialist (in this case a tennis coach) and expressed them using XQuery, the standard XML language. In our experiment, these expressions generated the required results after the sensor data was passed through UbiQuSE. Our experiments focused on the accuracy of our algorithms in detecting key events in the sensed data and results are reported and discussed in this paper.

Future work is focused both on improving the accuracy of our event detection algorithms and widening the scope of the coaches requirements to include doubles matches and the complexities of the tie-break in tennis. Current algorithms will not detect all possibilities at present. We will also move to deploy UbiQuSE to monitoring a field hockey game where one of our collaborators has installed the UbiSense system around the playing area. While it will require the development of only one processor, this will provide an interesting challenge as it increases the number of players within the sensor network.

References

1. Rules of Tennis 2009. http://www.tennisireland.ie/userfiles/File/Database /IO_38417_original.pdf
2. Tennis Ireland. http://www.tennisireland.ie

3. Ubisense English Site. http://www.ubisense.net/
4. Shaeib A., Conroy K., Roantree M. Extracting tennis statistics from wireless sensing environments, in *Proceedings of the Sixth international Workshop on Data Management For Sensor Networks*, DMSN 2009.
5. Technical Report. http://www.computing.dcu.ie/ĩsg/ publications/ISG-09-02.pdf
6. Ye J., McKeever S., Coyle L., Neely S., Dobson S. Resolving uncertainty in context integration and abstraction: context integration and abstraction, in *Proceedings of the 5th international conference on Pervasive services*, ICPS 2008.
7. Coyle L., Ye J., Loureiro E., Knox S.,Dobson S., Nixon P. A Proposed Approach to Evaluate the Accuracy of Tag-based Location Systems in *Workshop on Ubiquitous Systems Evaluation*, USE 2007.
8. Wang X., Dong J. S., Chin C., Hettiarachchi S., Zhang D., Semantic Space: An Infrastructure for Smart Spaces/ in *IEEE Pervasive Computing, vol. 3, no. 3, pp. 32-39*, July-Sept. 2004
9. Takahashi, M., Ito, S., Sumi, Y., Tsuchikawa, M., Kogure, K., Mase, K., and Nishida, T. A layered interpretation of human interactions captured by ubiquitous sensors. in *Proceedings of the the 1st ACM Workshop on Continuous Archival and Retrieval of Personal Experiences*, CARPE 2004.
10. Tien M., Wang Y., Chou C., Hsieh K., Chu W., Wu J. Event detection in tennis matches based on video data mining in *Proceedings of the 2008 IEEE International Conference on Multimedia and Expo*, ICME 2008.
11. Tien, M., Lin, Y., and Wu, J. 2009. Sports wizard: sports video browsing based on semantic concepts and game structure. in *Proceedings of the Seventeen ACM international Conference on Multimedia*, MM 2009.
12. O'Connaire, C., Kelly P., Connaghan D., O'Connor N. TennisSense: A Platform for Extracting Semantic Information from Multi-camera Tennis Data in *Proceedings of the 16th International Conference on Digital Signal Processing*, DSP 2009.
13. http://exist.sourceforge.net/
14. http://www.w3.org/TR/xpath20/
15. http://www.w3.org/TR/xquery/