A SCADA oriented middleware for RFID technology

Ismael Abad Cardiel *, Ruben Heradio Gil, Carlos Cerrada Somolinos, Jose Cerrada Somolinos

Dept. of Software Engineering and Computer Systems, Universidad Nacional de Educación a Distancia, Juan del Rosal 16, E-28040 Madrid, Spain

ARTICLE INFO

ABSTRACT

Radio Frequency IDentification (RFID) has emerged as the new technology paradigm for acquisition and information management. RFID can be used to improve significantly the efficiency of business processes by providing the capability of automatic identification and data capture. This technology introduces new challenges on data and process information management in current systems. RFID data are time-dependent and dynamically changing. In addition, data carry implicit semantics. The homogeneous data processing of such implicit semantics allows us to propose RFID middleware as a WHO–WHEN–WHERE data problem. This paper presents DEPCAS, a new middleware for RFID information based on the SCADA architecture for control systems. An application of DEPCAS is the resolution of heterogeneous situations, which solves the WHAT or context-aware to apply the auto identification data received from RFID systems in business applications.

© 2012 Elsevier Ltd. All rights reserved.

1. Introduction

Due to the ubiquity of Radio Frequency IDentification (RFID) deployments and low cost tagging (Jung, 2011), a growing amount of data are being generated from multiple sources that need to be processed and managed. So, new questions are arising, such as how to encompass current and future identification technology at the same time and how this information can be combined with existing applications (Floerkemeier, Roduner, & Lampe, 2007; Sheng, Li, & Zeadally, 2008; Su, Chu, Prabhu, & Gadh, 2008). The large amount of information generated in the RFID startup projects requires to know how and when the information is going to be summarized (Lee & Chan, 2009; Ustundag, Kilin, & Cevikcan, 2010).

Many of the non-finished RFID pilot projects have been focused in the RFID data instead of being focused on the business related events (Bornhövd, Lin, Haller, & Schaper, 2004). Raw RFID data have little value until they are transformed into a form suitable for application-level interaction (Curtin, Kauffman, & Riggins, 2007). Moreover, data have implicit meanings and associated relationships with other RFID data (e.g., data about containment, position, aggregation...). So, RFID applications must make appropriate inferences. For instance, observations of RFID tags associated to several items and a RFID tag associated to a container in a certain time interval can imply that the items are packed in the container.

In addition, the wide spread use of RFID technology involves a huge migration problem. There are thousands of applications using identification through codes, bar codes, plates, badges, 2D codes,

operator identifications, etc. (He et al., 2011) that in a mid-time will change to RFID auto identification (Brewer, Sloan, & Landers, 1999). From the application that can be found in any little market, to the large and more complex information system of a big and global company, the change from identification to auto identification will arrive Lin, Chen, and Hsu (2011) and Lee and Lee (2011).

The state of the art in RFID integration in existing and real systems provide a research prospect to propose a general middleware architecture that processes and consolidates RFID information solving the former issues and helping efficiently to migrate from identification to auto identification systems (Kefalakis, Leontiadis, Soldatos, & Donsez, 2009). This paper proposes a new middleware architecture for RFID called DEPCAS (Data EPC[1] Acquisition System), which integrates existing proposals for RFID middleware and it is based on the Supervisory Control And Data Acquisition (SCADA) architecture for control systems (Zafer & Aydogmus, 2009). Key concepts supported by DEPCAS are:

1. *Converting RFID information to business information*: RFID applications receive basic and simple data source inputs as time-stamp, tag code or reader dock (reader plus antenna) (Hsu & Yuan, 2011). This raw information should be translated to business relevant information using abstract process that can be applied in heterogeneous RFID installations.
2. *Distributing RFID information to business processes*: the RFID processed information need to be defined and accessible for external applications in a general way. The format and the exchange mechanisms must be transparent and general for

---

* Corresponding author.
E-mail addresses: iabad@issi.uned.es (I.A. Cardiel), rheradio@issi.uned.es (R.H. Gil), ccerrada@issi.uned.es (C.C. Somolinos), jcerrada@issi.uned.es (J.C. Somolinos).

[1] EPC stands for Electronic Product Code.

any kind of applications. The exchange flow between existing running programs and RFID acquisition system is bidirectional. It is necessary to solve how to send the necessary business information data to the RFID acquisition system and the opposite flow from RFID middleware to external entities.

3. *Filtering information*: the computer systems that manage a big quantity of input data must solve a filter policy. Filter process could be as simple as eliminating all unknown information or as complex as operations that manages every time-stamp and any network structure to ensure the data quality. In the RFID acquisition systems the raw tag readings are filtered according to a previous algorithm that helps the back end application to receive only relevant information.

4. *Triggering business process from RFID process and vice versa*: it is a common situation to solve that RFID acquisition system can receive any kind of input that triggers external activities. And in the opposite way, external input can modify RFID acquisition process. In any direction, RFID acquisition must solve the event trigger process.

5. *Managing RFID devices*: this requirement deals with an abstraction of physical communication and protocol exchange between reader devices and the acquisition platform. It is necessary to manage also the topological reader network, i.e., how readers are distributed or centralized.

6. *Consolidating RFID information*: the general RFID process generates relevant information that has to be consolidated. This permanent information is the result of specific process according to the homogeneous process. The homogeneous process creates detailed information about how, where, and why RFID information is being used in a productive process.

The remainder of this paper is structured as follows. Section 2 summarizes related work. Section 3 describes DEPCAS. Section 4 outlines our current implementation of DEPCAS. Section 5 presents a scenario where DEPCAS is used. Finally, Section 6 presents the conclusions of our work.

2. Related work

The need of RFID middleware is gradually becoming a basic question for non-trivial RFID installations. This is particularly true in heterogeneous environments, including multiple readers, applications instances, supply chain management, complex process and sophisticated business semantics (Cheung, Cheung, & Kwok, 2012; Kim, Kim, Hong, & Kwon, 2008; Ko, Kwak, Cho, & Kim, 2011; Lee, Ho, Ho, & Lau, 2011).

RFID middleware is indispensable for these main reasons:

1. The need to filter RFID data in order to avoid redundant or erroneous information that is not needed to business applications, while at the same time optimizing resources.
2. The need to process and interface RFID data from origin (tag reads) in heterogeneous deployments (multi-tagging and multi-readers systems) without resorting to business integration logic.
3. The flexibility to integrate RFID systems to support auto-identification in different applications and process models.

There are several surveys in the literature that propose system taxonomies and compilations about RFID middleware (Bo, Jin-Ta, Ping, Jun-Bo, & Zhen-Hua, 2006; Huang, Le, & Sharma, 2007). The RFID middleware market is dominated by three tendencies:

1. Specialized companies which offer RFID middleware. These companies can be classified into product oriented and specific RFID middleware producer. If they are reader/antenna, or general

hardware producer, then they include specific RFID middleware to support the hardware. If we speak about a general middleware producer it is quite common they have included an specific plugin to acquire RFID data. There are new companies that have born with the specific commercial target to create, distribute and install RFID middleware. We can mention in this group:

(a) WinRFID from UCLA-WINMEC Research Lab (Prabhu, Su, Ramamurthy, cheng Chu, & Gadh, 2005) is a research result to solve RFID features like scalability and administration, event and data intelligent process and dispatching.

(b) OAT Foundation Suite[2] is a RFID middleware platform that includes two-way process integration between enterprise applications and RFID acquisition systems.

(c) DETEGO/You-R OPEN[3] is the RFID middleware from RF-iT Solutions based on three main functions: business integration, data management and device management.

(d) TrueVUE RFID Platform[4] from VueTechnology is a complete software infrastructure to manager RFID deployment. It contains several modules to support different points of view in item level management.

2. The second group is the one of giants' software vendors such as IBM, Microsoft, ORACLE/Sun Microsystems, or SAP. Every of these huge companies have dedicated in the last decade important resources to include RFID acquisition inside their own products. Between the RFID middleware we can mention:

(a) RFID Java Middleware from ORACLE/Sun Microsystems.[5] With ORACLE and Sun Microsystems company fusion, the RFID middleware from both companies was renamed using the ORACLE reference. Now the product name is ORACLE Sensor Edge Server based on ORACLE Fusion Middleware.

(b) SAP Solutions for Auto-ID and Item serialization[6] is a set of components to include RFID technologies in SAP management tool. The solution offers three major components: SAP Auto-ID infrastructure, SAP object event repository and SAP event manager.

(c) IBM WebSphere RFID[7] is the IBM middleware to implement RFID acquisition. It consists of EdgeControllers and the WebSphere RFID Premises Servers which includes WebSphere Application Server and DB2 Workgroup server.

(d) RFID Anywhere[8] from Sybase iAnywhere is a software infrastructure that simplifies the development, deployment, configuration and management task for RFID networks.

3. The last group of RFID middleware producers is the open source initiatives. These proposals are research results that take origin in university groups or consortiums. There are several Open Source Initiatives to solve an RFID middleware:

(a) CUHK RFID Middleware[9] from MobiTEC Technologies Centre of the Chinese University of Hong Kong.

(b) ASPIRE RFID Middleware[10] is an EU funded project to deliver an Advanced Sensors and lightweight Programmable middleware for Innovative RFID Enterprise applications as an open source middleware.

---

[2] http://www.oatsystems.com.

[3] See "Plug and Trace. Excellent RFID Solutions" available at http://www.enso-detego.com.

[4] http://tycoretailsolutions.com.

[5] See "Oracle Application Server Wireless Developer's Guide" available at http://docs.oracle.com.

[6] See "SAP Research Report 2009/2010. From creativity to value" available at http://research-report-20092010.research-events.com/.

[7] See "IBM WebSphere Premises Server" available at http://www.ibm.com.

[8] See "Getting The Most From Mobile RFID With Middleware" available at http://www.sybase.com.

[9] See "CUHK RFID Middleware – System Design Document" available at http://mobitec.i.e.cuhk.edu.hk.

[10] http://wiki.aspire.ow2.org.

(c) RIFIDI Edge Server[11] is a software architecture proposed by the RFID Research Center in the University of Arkansas.

(d) FOSSTRAK (Floerkemeier et al., 2007) (called ACCADA) is a middleware implementation sponsored by Auto-ID Labs oriented to produce a free and open source software for track and trace.

SCADA and RFID middleware systems share a set of commonalities about structure, functions and characteristics. From an acquisition point of view, the physical low level is a set of devices acquiring data: we have called sensors and RTU's in SCADA world and we call antenna/edge/reader in RFID world. The communication networks to integrate the data capture system with the master processing system. This network communication could be defined with a variety of technologies and topologies. And the last element: the server data processor. The functions in SCADA and RFID middleware systems are similar. Main question of both of them is to acquire data: SCADA data acquisition could be expressed in terms of discrete inputs (equipment is on or off, trip-wire alarms, power failures, etc.) and analog values (level in tanks, voltage levels, temperature, or any other factor, etc.). RFID middleware data acquisition system is a three-tuple: the RFID code or the WHO, the time-stamp or the WHEN, and the reading point or the WHERE.

If we analyze the characteristics in the two kind of acquisition systems, there are a set of common points between SCADA software systems and RFID acquisition system:

1. *Hiding hardware*: SCADA software system hides the specific Remote Terminal Units (RTUs) hardware. It is a common situation that SCADAs manage a set of RTUs from different companies or with different structure. This same situation could be found in RFID middleware with a deployment of an RFID infrastructure. At our days we can find a set of various RFID readers' trades like Alien, Intermec, Feig, Siemens, Symbol, Sick, etc.

2. *Hiding communication between devices and master software*: data transport from acquiring devices to master software can be made using diverse communication infrastructures and different protocols. There are all kind of SCADA communication infrastructures, from the simplest serial wired structure to the most complex satellite structure. Additionally, any advance in telecommunication field will be used if there was any opportunity. The same situation may be repeated in RFID systems. The present readers have limited options to communicate to master software using Ethernet network or serial communications. But there are prototypes using optical fiber or wireless communications.

3. *Quantity Information*: one of the more significant common points between SCADA and RFID acquisition software is that they receive a relevant quantity of dynamic information. SCADAs can receive data with milliseconds of resolution from a big quantity of data signal to monitor a complete installation. The same situation may be produced inside a RFID monitoring installation. This data should be received and processed to obtain relevant information.

4. *From fine-grain information to relevant information*: both systems acquire data characterized from a very fine grain. SCADA basic data are status signal, analog value, and may be rate counters. RFID acquire system receive data about auto identification with a code. The two type of systems have the possibility to manage additional basic information like time-stamp, read quality, read origin, etc. All this individual information has very little meanings and it has a very low meaning relevance. The relevant information is obtained when it is processed with others data to obtain significant information.

5. *Real time process*: SCADA systems are typical real time systems. Data is acquired and processed to obtain the relevant alarms or calculation that give information to the operators. Of course, there are differences about the meaning of real time depending on specific situations. It is not the same to speak about real time in electrical SCADA systems that in water SCADA. Time scale it is not comparable between electrical processes with water processes systems. The same situation it is reproduced in RFID acquisition system. Auto identification could be time critical as the process where it has been using.

6. *Consolidating information*: the SCADA and RFID middleware process data is defined to obtain significant information not only real time alarms or events. This processed data is structured, stored, and summarized to be available for other systems. All this operations allow filtering managing the information that is received in the system.

7. *External interfaces*: the consolidated information should be offered to others systems. Modern SCADA software systems manage external structured information to define and configure the own process. This information has in-out components. SCADA receives information about configuration, Geographic Information System (GISs), maintaining systems or any internal warehouse. SCADA offers information about the process results to any back-end company software. This situation is repeated in RFID middleware environments.

3. DEPCAS

Fig. 1 depicts the overall DEPCAS architecture. The scheme proposed here is based on the architecture of modern SCADA (Supervisory Control and Data Acquisition) systems. The basic issues in DEPCAS are:

1. Producing RFID processed data.
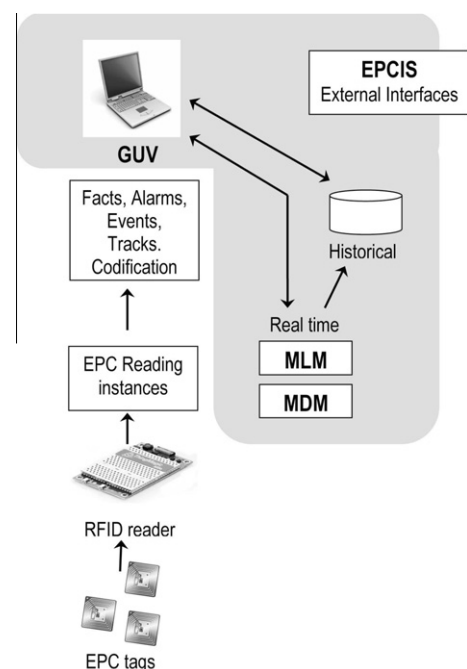2. Hiding heterogeneous RFID deployment systems with an homogeneous layer approach.

Fig. 1. Architecture of DEPCAS.

3. Translating business (async or sync) needs to RFID systems.
4. Providing management capabilities in RFID middleware environment.

To solve these main targets the structure of the acquisition system that we propose is organized in four great layers or subsystems:

1. Middleware Device Manager (MDM). It has the following basic functions. First, the communication management with one or more devices: RFID readers. Second, implementing communication protocol with the readers and event processor implementation (ALE: Application Level Event used in the EPCglobal standard[12]). And third, topological configurations support defined by the acquisition equipment: antennas, edge readers and readers.
2. Middleware Logic Manager (MLM). It supports the process to obtain RFID summarized data from auto-identification raw acquired data. The logical process to resolve will depend on each particular scenario (Chow, Choy, Lee, & Lau, 2006). This process will generate information of a permanent nature which is the result of the process in every scenario instance. The basic scenarios on the proposed work are related with: monitoring and tracking operations, aggregation of information items to generate new information, sorting of items, format raw RFID data conversion, and state machine delivery information. The concept of scenario is equivalent to a logical process that generally can be applied in all situations where information is used for self-identification. At each stage information is received, processed, summarized and accumulates, and generate and consolidate the data according to an specific logic.
3. Graphical User Viewer (GUV). It is the mechanism that DEPCAS supplies to help in the installation and management of a RFID system from the engineering point of view. This subsystem supports two basic operations: the supervisory capability to the RFID system and certain data manageability (configuration and acquisition data).
4. Information system exchange (EPCIS: EPC Information Services). EPCIS services should be the gateway of information between the data provided by the MDM and different MLMs to external business. These services will enable both receiving and sending information from the DEPCAS system to other systems.

Following subsections describe in detail MDM, MLM, GUV and EPCIS.

3.1. MDM

One of the main purposes of RFID middleware is to manage and hide the broad range of device readers existing in acquisition RFID networks. There are two basic solutions to override this question. The first one is to define a specification that should be accomplished for every RFID reader inside a network. The second solution is to define an abstraction layer that can translate proprietary protocols and specific characteristics from a vendor to a general solution. MDM is the specific DEPCAS layer to support all these management operations. The study of different RFID Device Management components allows obtaining a set of features that are included to solve it in this kind of systems. Among these features we include:

1. *Device Agnostic*: the device management should include a specific function to hide the alternatives of existing devices related with: physical parametrization, communication protocol, communication options, etc.
2. *Configuration and parametrization of RFID devices*: an agnostic approach to existing devices sets the options to manage from an homogeneous point of view the configuration and parametrization of every equipment in the RFID acquisition network.
3. *Topological RFID devices*: the RFID device management can include some network organization that give additional capabilities to the acquisition network as: redundant point of reading, input characteristics, out characteristic, parallel reading devices, etc.
4. *Start-up devices*: the set of operations to startup and configure an RFID equipment inside a network are known as starting up protocols. Some of the existing RFID middleware support a starting up protocol giving instructions about how to include RFID reader in the network.
5. *Statistical and Report RFID devices data*: the deployments of RFID reader require to introduce maintenance operations in the network. The statistical and reporting information are essential to optimize these operations.
6. *Monitoring devices in live operation*: some existing RFID networks can manage critical operations thus the RFID middleware supports a supervisory management about working devices.
7. *Diagnosis of devices in live operation*: the RFID middleware includes operations like device signals, examining logs, updating software devices, etc.
8. *LLRP implementation*: the Low Level Reader Protocol[13] is the standard proposal from EPCGlobal Network to specify the interface between RFID readers and any software client.
9. *OSGi Technology*: the increase of software complexity in heterogeneous hardware environment has promote the development of frameworks to support the continuous changes. OSGi framework is a module and service platform that implements a complete and dynamic model. Any framework that implements OSGi provides an environment for modularization of application in small pieces tightly-coupled, dynamic loadable and configuration files to declare external dependencies. The OSGi Architecture is one of the most common specification framework used in our days. The OSGi specification enable components to hide their implementation from others components while communicating through services.

These features and functions define the operation of the middleware device manager in RFID middleware. For instance, Device Agnostic Feature is provided when middleware infrastructure acts as an integration point for RFID readers, printers, and other infrastructure devices, such as any sensors and actuators. RFID reader and printer manufacturers typically provide complex configuration options, proprietary interface languages, and communication protocols. If RFID device infrastructure eliminates the need for RFID integrators to be aware of these complexities and provides a device agnostic interface with which supported devices can be configured hiding the complexity. Other features are related to scope and technological implementation of middleware device manager.

Table 1 compares the features supported by DEPCAS MDM to the related work summarized in Section 2. The main goals of MDM in DEPCAS are:

1. Managing the topology RFID reader's network.
2. Agnostic management of different RFID vendors.
3. Homogeneous process of different RFID tag reads.

Table 1
DEPCAS MDM support compared to related word.

	FOSSTRACK	RIFIDI	ASPIRE	ORACLE	IBM	DETEGO	OAT RFID Plat.	DEPCAS
Device Agnostic	Yes/No	No	Yes	Yes	Yes	Yes	Yes	Yes[a]
Device Configuration	Yes	Yes	Yes	No	No	Yes	Yes	Yes
Topological Configuration	No	No	Yes	Yes	No	Yes	Yes	Yes[b]
Needs Programming Starting-up Device	Yes	Yes	Yes	No	No	No	No	Yes/No[c]
Statistical Data from Device	Yes	No	Yes	No	No	Yes	Yes	Yes[d]
Monitoring Data from Device	No	No	No	Yes	Yes	Yes	Yes	Yes[e]
Diagnosis capabilities	Yes	No	Yes	No	Yes	No	Yes	Yes[f]
LLRP Implementation	Yes	Yes	Yes	No	No	No	No	No
OSGi Technologies	No	Yes	Yes	No	No	No	No	Yes[g]

[a] using MARC sentences to configure and connect RFID devices.
[b] using RRTL.
[c] depending on the API of the chosen reader.
[d] statistical operations are included in MARC sentences.
[e] MARC sentences include data quality evaluation if it is possible to obtain information from readers.
[f] there are MARC sentences to produce device diagnosis and there are operations included in RRTL about topological diagnosis.
[g] to process the parallel RFID readers.

4. Supporting LLRP and ALE specification process.
5. Providing configurable RFID network.
6. Hiding operational environment.
7. Minimum tag management operations.
8. Monitoring and controlling RFID readers.

To support these objectives, the MDM definition includes the concept of Minimum Abstract Reader Commands (MARC) and the RFID Reader Topology Language (RRTL). The MARC allows hiding the particularities of every RFID reader vendor through a set of operations that are mapped to specific vendor API. The RRTL includes the description of RFID topologies including reader control and management. The MARC is the minimum set of instructions that an RFID reader must supply to support the operational instructions inside a DEPCAS network. The set of MARC is closed and give an independent layer of RFID management. The implementation of MARC is related to RFID reader API and gives an independent hardware layer. The RRTL is a language that includes primitives to define DEPCAS network. The sentences of this language include the necessary semantic to explain how the readers are organized inside the acquisition network.

3.2. MLM

Current RFID middleware systems capture raw readings from readers, filter and aggregate the data, and finally yield cleansed RFID raw events. However, RFID raw events only provide low-level information such as EPC (Eun-Jun & Yoon, 2012), reader identification and time-stamp, which are not directly related to business processes. Therefore, the ability to quickly transform these raw events into business events useful for decision-making becomes a central issue for realizing the maximum value from RFID technology in business.

The key aspects to take in count in processing RFID information are:

1. *Process data*: the RFID raw process is more than a basic filtering, it is a cleansing, consolidation and summarization set of operations that have as objective to create and consolidate useful RFID data for business.
2. *Alarm*: one of the purposes of RFID middleware is to make difference between relevant and not relevant data in acquisition. The alarms are the first processed data that are generated inside the middleware. RFID middleware's can generate two types of alarms: infrastructure alarming and process alarming.
3. *Event*: the RFID acquisition system should digest the meaningful events. The alarms are also event from this point of view but not only. The RFID middleware can define other relevant situations that are interesting for external application or to support the system exploitation.
4. *Filter*: the filter RFID process refers to the removal of certain tag read events based on any discriminator parameter as tag code repetition in a short period or the same tag code in two different readers.
5. *Consolidate*: the RFID middleware process will create a new RFID data that should be consolidated and maintained inside any database systems. This consolidation opens the way to share and to transmit to external applications the cooked RFID information.

Table 2
DEPCAS MLM support compared to related word.

	ORACLE	IBM	FOSSTRACK	RIFIDI	TrueVUE	DETEGO	DEPCAS
Alarm	No	Yes	No	Yes	Yes	Yes	Yes
Event	No	Yes	Yes	Yes	Yes	Yes	Yes
Filter Process	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Aggregation Process	No	Yes	No	Yes	No	Yes	Yes/No[a]
Configurable Data Process	No	No	No	No	Yes	Yes	Yes[b]
Consolidate Data	Yes	Yes	No	Yes	Yes	Yes	Yes[c]
Specific Business Modules	No	No	No	Yes	No	Yes	No[d]
Supply Chain Process	Yes	Yes	Yes	Yes	Yes	Yes	Yes[e]
EPC Network Support	Yes	Yes	Yes	Yes	Yes	Yes	Yes[f]

[a] depending on every scenario process.
[b] scenario configuration.
[c] scenario process results.
[d] scenario configuration applies to every business case.
[e] using the scenarios according to the supply chain process definition.
[f] using the scenario process.

Table 2 compares the features supported by DEPCAS MLM to the related work summarized in Section 2. Existing RFID middleware includes a basic RFID data process. This basic process is related with event process and filter reads in the RFID middleware. Some of the existing middleware also propose an additional operation, called aggregation or summary reads. The basic idea under this operation is to reduce the tag reads using any criteria. For instance, aggregating operation can detect input/output read, selecting just the first and the last read when a tag appear under read range. Other approach is to count the tag reads and accumulate it. Other RFID include specific business modules. These modules perform business explicit process from the raw acquisition systems and offers direct connectors with the back end application.

The main target in the conception of DEPCAS is to think about the use of homogenous RFID middleware that can be introduced in working systems that already use identification (and want to change to use auto identification). For instance, we are habituated to use the bar code in many circumstances in daily life, like shopping, luggage, personal cards, etc. All these heterogeneous systems that use identification in a half term will transform into systems that use the auto identification. The mechanism that solves this homogenization in DEPCAS is the scenario concept, which is equivalent to a logical process with a general application that it is possible to be used in a set of situations where the information takes advantage of auto identification. In each scenario there are RFID data that is received, processed, summarized and accumulated, according to their own logic.

The basic defined scenarios in DEPCAS are:

1. *Tracking of items*: this scenario corresponds to an application in which it is wanted to make the tracking of some object in a chain that is supervised with RFID readers. The objects when happening through each point generate the identification events that are registered from an origin to a destiny. The different events will depend on the parameters that are desired to check: time, distance, speed, step/no step, complete route, etc. Section 5 details this scenario.
2. *Items classification*: this scenario consists of a classification of elements after an identification of an object. The read event from an object generates the classification according to certain criteria. The application of this scenario would be applicable to an automatic robotically classification system. or to a system of shipments classification.
3. *Aggregation of items*: this scenario would allow adding auto identified objects to generate a new information registry. This scenario can be applied in those cases in which a product is develop by parts that are added, like for example the creation of baskets of gifts, pallets, etc.
4. *Format conversion*: this scenario would allow generating information from auto identification codes. For instance, it could be applied for systems in which hybrid identifications are received and wanted to make uniform. This scenario allows to turn formats EPC to other formats of particular objects like the ISBN, ISSN, etc.
5. *Execution of machine of states*: the scenarios that represent state machines would allow associating information of auto identification to the identified object states. For instance, it could be used to make the track of a rent of vehicles or the management of loans in a library.

3.3. GUV

GUV is a combination of configurable components, that graphically reflect the dynamic state of the simple components (tags, readers...) or complex ones (routes, adders, state machines...)

obtaining different types of information from data servers. The creation of a specific application will be carried out by the grouping of these components, properly configured like a client data or on a web node for an Internet browser.

The study of different RFID GUVs allows obtaining a set of characteristics that are implemented to solve the user interface in the RFID middleware. Among these features we include:

1. *Web Client*: the graphical component can be supported as lightweight software using web client applications.
2. *Collection of dynamic information*: GUV should include dynamic graphical components to show the status and quality of represented data.
3. *Managing the database infrastructure of RFID*: the database management tool is a graphical component that simplifies the database operations (i.e., search, create, delete or update) about RFID infrastructure (e.g., readers, tags...).
4. *Managing the process database*: the database management tool is a graphical component that simplifies the database operations (i.e., search, create, delete or update) about RFID process configuration.
5. *Lists of information*: the list graphical resource presents the database information like a summary where the user can choose the rows and the columns shown for an specific table.
6. *Editing and management of graphical elements*: the option to edit and manage the graphical resources allows building specific user environment depending on installation or user authorization policy.
7. *Management and configuration of alarms*: alarms are one of the most important resources on RFID middleware. GUV includes several mechanisms to present the alarming information: tables, summaries, flashing and sound signals, etc.
8. *User management and privileges*: these kinds of options allow defining graphical user organization and policy to show/not to show data to the users.

Table 3 compares the features supported by DEPCAS GUV to the related work summarized in Section 2. The main objectives of GUV are:

1. Including a single common environment and information management in DEPCAS middleware. This environment is common to all management information system configuration data regardless of being related to infrastructure, the processing logic sets, or the EPCIS modules.
2. Managing a common graphical object to support information lists. These lists can be used like dynamic queries with real time information or static queries to show database information.
3. Supporting graphical displaying information about a particular object in real time. You may include in new graphics another graphical object with real time attributes.
4. Representing alarm system that can be configured. This treatment includes alarm management levels and the realization of automated operations
5. Defining the process and system events for the acquisition of auto identification data.
6. Managing graphical objects that are used and which allows their reuse in different graphical interfaces.

3.4. EPCIS

The EPC Information Service provides a specification framework to solve the external RFID middleware applications to capture, secure and access RFID data. The original EPC network architecture was defined in Harrison et al. (2003) like an "omniscient entity" to provide relevant information required from business application

Table 3
DEPCAS GUV support compared to related word.

	ORACLE	IBM	Sybase	TrueVue	FOSSTRAK	WinRFID	RIFIDI	DEPCAS
Web Client	Yes	Yes	No	No	Yes	No	No	No
Dynamic Information	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes[a]
Configurable Dynamic Information	No	Yes	Yes	Yes	No	Yes	Yes	Yes[b]
RFID Database Management	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes[c]
RFID Process Configuration	No	Yes	Yes	Yes	No	Yes	No	Yes [c]
Lists	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes[d]
Graphical Edition	No	No	Yes	Yes	No	No	Yes	Yes [b]
Alarms	No	Yes	Yes	Yes	No	Yes	No	Yes[e]
Events	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes[f]
User Management	Yes	Yes	No	No	Yes	No	No	Yes[g]

[a] using graphical connection to real time DEPCAS database and DEPCAS repository.
[b] using ediGUV to create and compound graphics.
[c] using a uniform Database Management Tool (DMT).
[d] dynamic and static graphical representation.
[e] like an special dynamic list and connected to the real time database.
[f] like an special static list and connected to the repository database.
[g] using user and user groups to define operational capabilities.

to RFID middleware. The main four functions included in EPCS in this initial proposal were: providing a long term repository of EPC event data, supplying an access point to higher-level information obtained from EPC process acquisition, solving the storage and access to detailed business information associated to EPC process acquisition, and providing transparent on-demand access to data attribute held in others systems. EPC data was interchanged using the Physical Markup Language (PML) (Floerkemeier, Anarkat, Osinski, & Harrison, 2003), an XML schema for representing EPC-related data including auto identification information (e.g., codes, readers, antennas, time-stamp. . .) and business information (e.g., size, quantities, width. . .).

These functions were so broad that a general confusion about how to implement them happened. These pending questions are got pass in the definite EPCIS specification definition (EPCglobal, 2007) including an standard interface to allow EPC data interchange. This standard interface is characterized by: managing only historical data (real-time EPC data is decoupled from EPCIS), operating with cooked EPC related information (raw EPC data is processed in low EPC Network layers), and dealing only with enterprise systems environment. All this specific functions are solved using a three layer interface: the abstract data model layer, the data definition layer and the service layer. The abstract data model is a closed and defined interface that proposes a specification of general requirements for creating data definitions in the data definition layer. This data definition is called core event type. This structural organization in the interface allow to use standard vocabulary elements used by inter organizations and user specific vocabulary elements defined and meaning inside intra organizations.

Despite the objective of all these implementations is the same: to connect business with RFID middleware, there are a set of characteristics that differ between specific solutions. This analysis introduces the following features:

1. *Standard compliance.* What is the compliance degree in the existing implementation according to EPC Network standard?
2. *EPCIS permanent data model implementation.* What is the existing repository implementation supported by EPCIS solutions.
3. *Security EPCIS control.* There is a large range of security control levels in EPCIS data, from general user access to low granular data access.
4. *Metadata management.* The business process defines process data and specific attributes that are introduced in the EPCIS repository.

5. *Supply chain support.* The EPCIS implementation has specific elements to solve supply chain questions.
6. *Reports.* The EPCIS includes report facility to data access using XML exporters.

Table 4 compares the features supported by DEPCAS EPCIS o the related work summarized in Section 2.

The main functionalities of EPCIS in DEPCAS are:

1. Publish/subscription manager.
2. Permanent historical data information.
3. Transfer in/out data between DEPCAS and business applications.
4. Security distribution of DEPCAS information.
5. XML data mapping.
6. Specific application connector with DEPCAS.

4. A DEPCAS prototype

DEPCAS basic implementation includes a prototype module for all its components.

4.1. MDM implementation

The DEPCAS MDM core architecture prototype has been developed using TCP and HTTP for transporting reader protocol messages, while the message content can be either XML or Text. In addition, it supports synchronous and asynchronous messaging (through the reader's protocol Notification Channels mechanisms). The MDM MARC sentences implementation includes the following operations:

1. `setAntennaSequence`: Set an order sequence to read from multiple antennas installed in one reader
2. `setAutoFalseOutput`: Order to autonomous function mode to ignore tag reads
3. `setAutoFalsePause`: Order to autonomous function mode to pause working
4. `setAutoMode`: Start/Stop autonomous reader functions
5. `setAutoStartTrigger`: Activate the trigger sending functions in reader autonomous function
6. `setAutoStartTimer`: Activate periodic timer reader function in reader autonomous function
7. `setAutoStopTimer`: Reset periodic timer reader function in reader autonomous function

Table 4
DEPCAS EPCIS support compared to related word.

	ORACLE	IBM	WinRFID	FOSSTRACK	DEPCAS
Standard Compliance	Partial	Total	Partial	Partial	No
Data model	Oracle PostgreSQL	Oracle DB2	Oracle SQLServer	MySQL	Yes[a]
WebServer	Jini	WebSphere	Unknown	Apache	Yes[b]
Security	WebServer security	Propietary security	Remote objects	Authentication and authorization for query operations	No
Metadata	Using data in repository	Specific metadata model	XML framework	No support for new vocabulary type	Yes[c]
Supply chain support	Yes	Yes	Specific adaptor	Yes	Yes[d]
Reports	Not included	Business Intelligence Reporting Tool	Not included	Not included	Not included
EPCIS events	Yes	Events and alerts	Rule manager exceptions	Yes	Yes[e]

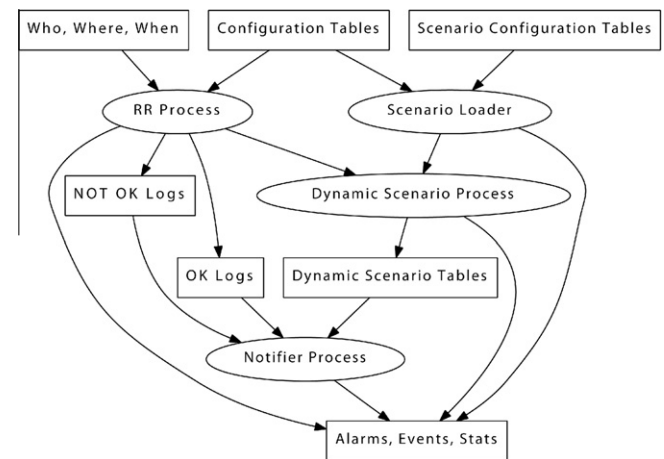[a] to solve the relation between business and MLM scenario.
[b] as a publish/subscription server to the DEPCAS repository.
[c] metadata captures the relationships between the RFID product-process data.
[d] as a middleware connector.
[e] to manage the external connection and to supervise the way connections are working.

8. setAutoStopTrigger: Reset the trigger sending functions in reader autonomous function
9. setAutoTrueOutput: Order to autonomous function mode to process tag reads
10. setAutoTruePause: Order to autonomous function mode to start reading process
11. setConnectTCP: Establish a TCP connection
12. setNotifyAddress: Init a listening TCP port to instruct the reader to send notification messages
13. setNotifyFormat: Configure the format message to be used by the reader
14. setNotifyTime: Establish the period that the reader notifier should use
15. setPassword: Identify the password reader access
16. setRun: Start to execute a configuration reader mode
17. setTagType: Establish the expected tag type property
18. setTime: Synchronize reader time
19. setUsername: Identify the user reader access

The MDM prototype includes MARC implementation for RFID reader ALIEN8780, FEIG2000, SKYWRAEM2 and we are working in MARC implementation for ThingMagic Mercury 4, Intermec IF61 and Impinj Speedway R420.

4.2. MLM implementation

The general process in any of MLM configuration follows the general flow presented in Fig. 2. MLM uses two main groups of configuration tables: the scenario configuration tables and the static configuration tables. The scenario configuration tables include data structure about the specific scenario that is processed. For instance, in the tracking scenario (see Section 5) the most relevant table is the "routes" table that includes information about the itinerary that items are going to be monitoring. The static configuration tables for MLM includes data about the general elements processed. For instance, what codes are going to be supervised (the who's that DEPCAS is going to process) or the reader points in the systems (the where's that DEPCAS logic manages). The MLM initial trigger is always a RR (relevant read) that always include a who (RFID code), a where (RFID reader point) or a when (RFID read time-stamp). RRs are processed one by one or can be processed by set of data by the RR Process. These processes generate data logs information from the processing result of incoming new RR and static configuration. For instance, if there is a RR with a who (RFID code)

Fig. 2. General process of DEPCAS MLM.

that is not inside the static configuration table the process generate a NOT_OK log and create the necessary data inside alarm, event or statistical tables.

The scenario loader initializes the configuration parameter of each scenario. For instance, in the state machine scenario process the initial operation starts every state machine defined to the initial state, producing the ignorance of every read associated to other states. The Dynamic Scenario Tables define proprietary data structures for every particular scenario. These tables are managed by the dynamic scenario process and they are scenario dependent. In the aggregation scenario, there are several scenario dependence process, for example the compositions table receives the items that are made up of the complete aggregation. When the items are processed they area accumulated to an specific aggregation. The notifier process communicates with the Business Event Process (BEP) to trigger it the events that are configured in every scenario.

4.3. GUV implementation

Based on the concepts of high-level design have been shown we have developed a GUV prototype environment that allows building graphical environment for a specific domain. The prototype has a graphic editing system that allows creation, modification and deletion of graphical elements. This environment, called ediGUV

defines a core container referred as DEPCASGUV in which may include other edited elements. The prototype operates two types of graphic elements: the basic elements that are provided directly by the window SDKs and the compound DEPCAS elements that are the result of basic elements compositions for a specific function in DEPCAS middleware. The basic graphic elements in the actual DEPCAS version are:

1. Windows
2. Buttons
3. Edit Fields
4. Text Fields
5. Check boxes
6. Combo boxes

DEPCAS compound graphic elements are:

1. Panel database (Add-Delete-Modify Data)
2. List of dynamic database
3. List of static database
4. RFID/EPC Label
5. RFID Reader Point
6. Dynamic graph (Route/Forecast)

4.4. EPCIS implementation

EPCIS is implemented using the following four subsystems:

1. *Publish/Subscription Manager* supports two kinds of relationships: relationships that are simple data representations of some actual relationship that exists between business atomic data and DEPCAS data, and relationships between tables that have their own properties. Data representations of relationships can contain a Web service reference to access the relationship Web service. The EPCIS implementation defines interfaces to query a manageable resource about the relationships it participates in directly, such as a containment relationship. The Publish/Subscription manager also defines an interface for a service that offers relationship information for many manageable resources enabling a relationship registry. Manageable resources do not have to be aware that they are participating in relationships.
2. *Converter Manager* implements a global repository that captures the relationships between the RFID product-process data, that we call scenario in DEPCAS terminology, and existing business data. The converter manager layer offers the possibility to control the level of aggregation by customized combination of several built-in filters as well as those developed by specific environments. Hence a wide range of variation is at hand. Attaching meta-data from backend-systems to DEPCAS scenarios requires connecting specific existing external data to DEPCAS scenarios tables.

3. *Feeder EPCIS interface* works using a Java Message Service (JMS) to exchange data from DEPCAS real time processing scenarios to EPCIS process. This capturing interface distributes the information to the consolidate database and to synchronous ECPIS connectors. The message structure is defined using an XML schema. This schema structures the general data field of any message according to a specific scenario. For example, the JMSType for a tracking scenario includes: general position, specific place, building, area, reader used, department, auto id read. This message is instantiated as showed in Fig. 3.
4. *Query EPCIS server* supports the DEPCAS servers/clients connections. It solves the different data interchange (HTTP, web services or XML) between DEPCAS and external business applications. In the same way, EPCIS server is designed as a platform with a uniform query and update interface to applications, while the actual implementation solves the details and data binding to existing databases and information systems. EPCIS server supports simultaneous binding to multiple databases and information systems from multiple vendors, as well as EPCIS as a managed application service. In order to accommodate various types of data relationships such as routes, predictions, reader points, associations with particular transactions, EPCIS server is implemented as a modular framework, with a very lightweight core functionality, which merely described which "profiles" or relationship types are implemented on any particular instantiation providing an EPCIS interface.

5. An example of DEPCAS scenario

The tracking of people or goods with identification mechanisms is repeated in heterogeneous form in diverse situations. The tracking operation abstraction can be considered in terms of a route to monitor and the parameters to process in this route, that we call predictions. The fundamental software function is tracking objects that travel by the defined routes. The process for each case depends on the type of parameter assigned to the route. The different types of routes make the corresponding verifications in each case:

1. BDR (Basic Data Routes): they are fixed routes in which the passage in a certain absolute time is verified.
2. BDTR (Basic Data Time Routes): they are fixed routes in which the passage in a certain relative time relative to the origin is verified.
3. ADR (Alternative Data Routes): they are alternative routes in which one verifies the origin and the destiny in a certain absolute time.
4. ADITR (Alternative Data Intermediates Time Routes): they are alternative routes in which one verifies the alternative origin, destiny and intermediate steps in an absolute time.
5. ADRTR (Alternative Data Relative Time Route): they are alternative routes in which one verifies the origin and the destiny according to a time relative to the origin.

```
Startup time: 21.jan.2008 11:30:28:343
21.jan.2008 11:30:28:343 Queue name: jms/Queue
Consumer Queue Selector: JMSCorrelationID = 'UNED' AND JMSType LIKE 'ISSI_2'
To end program, type Q or q, then <return>

Recv message: 21.jan.2008 11.38.01:531
Message header: 21.jan.2008 11:38:01:437 Producer-63, message order: 3
Data message: UNED, Campus 1, Edificio A,
Zona 3 Reader_A 00 ISSI_2 0000 1111 2222 3333 4444 8391 1981

Recv message:  21.jan.2008 11:38:01:562
Message header: 21.jan.2008 11:38:01:531 Producer-63, message order: 9
Data message: UNED, Campus 1, Edificio A,
Zona 3 Reader_A 00 ISSI_2 0000 1111 2222 3333 4444 7134 2187
```

Fig. 3. Example of EPCIS message.

6. ADRTITR (Alternative Data Relative Time Intermediate Time Routes): they are alternative routes in which one verifies the alternative origin, destiny and intermediate steps in a time relative to each made step.

7. UPUR (Undefined Predictions Undefined Routes): routes of any type for which any type of prediction algorithm is not defined.

The different processes generate the warnings that determine if the readings correspond to the predicted thing or they do not correspond and why. Between the system warnings we can distinguish three groups. The first group are the warnings generated in correct tracking, we denominated `LOG_OK`. A second group is related with a correct tracking in which some type of incidence takes place, like for example not fulfilling a certain part of a route or for those cases in which the route is fulfilled with some time delay error. The third group of warnings are those generated in case of breaks of established routes, an example of this type are the `LOG_LR_NOK` warning that is generated in the case of appearing labels in routes in which they would not have to appear. In order to verify the operation of this type of scenarios, a monitor has been developed that shows all information about the labels tracking.

6. Conclusions

Research on RFID middleware is oriented to find new alternatives to efficiently include auto-identification into productive processes. New topological networks of tag readers and heterogeneous business applications need to be connected through RFID middleware, solving all the established requirements.

In this paper, we have summarized the features that must be supported by any "real" RFID middleware. As a result, we propose the DEPCAS middleware, which starting point is the well known SCADA architecture. After describing the theoretical basis of our approach, i.e., the architecture of DEPCAS, its current implementation and an application scenario have been reported.

Acknowledgements

References

Bo, F., Jin-Ta, L., Ping, Z., Jun-Bo, G., & Zhen-Hua, D. (2006). Study of rfid middleware for distributed large-scale systems. In *Information and Communication Technologies, 2nd ICTTA '06. Vol. 2* (pp. 2754–2759).

Bornhövd, C., Lin, T., Haller, S., & Schaper, J. (2004). Integrating automatic data acquisition with business processes experiences with sap's auto-id infrastructure. In *Proceedings of the 30th international conference on Very large data bases – Vol. 30, VLDB '04, VLDB Endowment* (pp. 1182–1188).

Brewer, A., Sloan, N., & Landers, T. L. (1999). Intelligent tracking in manufacturing. *Journal of Intelligent Manufacturing*, 245–250.

Cheung, C., Cheung, C., & Kwok, S. (2012). A knowledge-based customization system for supply chain integration. *Expert Systems with Applications, 39*(4), 3906–3924.

Chow, H. K., Choy, K. L., Lee, W., & Lau, K. (2006). Design of a RFID case-based resource management system for warehouse operations. *Expert Systems with Applications, 30*(4), 561–576.

Curtin, J., Kauffman, R., & Riggins, F. (2007). Making the most out of rfid technology: A research agenda for the study of the adoption, usage and impact of RFID. *Information Technology and Management, 8*, 87–110.

EPCglobal. (2007). EPC information services (epcis) version 1.0 specification, Technical report.

Eun-Jun & Yoon (2012). Improvement of the securing rfid systems conforming to EPC class 1 generation 2 standard. *Expert Systems with Applications, 39*(1), 1589–1594.

Floerkemeier, C., Anarkat, D., Osinski, T., & Harrison, M. (2003). Pml core specification 1.0, Technical report, Auto-Id Centre Institute For Manufacturing, University Of Cambridge.

Floerkemeier, C., Roduner, C., & Lampe, M. (2007). RFID application development with the accada middleware platform. *IEEE Systems Journal, 1*(2), 82–94.

Harrison, M. (2003). EPC information service-data model and queries, Technical report, Auto-Id Centre Institute For Manufacturing, University Of Cambridge.

He, M., Horng, S.-J., Fan, P., Khan, M. K., Run, R.-S., Lai, J.-L., et al. (2011). A fast RFID tag identification algorithm based on counter and stack. *Expert Systems with Applications, 38*(6), 6829–6838.

Hsu, C.-C., & Yuan, P.-C. (2011). The design and implementation of an intelligent deployment system for RFID readers. *Expert Systems with Applications, 38*(8), 10506–10517.

Huang, X., Le, S., & Sharma, D. (2007). A taxonomy for RFID systems. In *IEEE International conference on signal processing and communication systems, Gold Coast, Australia*.

Jung, J. J. (2011). Ubiquitous conference management system for mobile recommendation services based on mobilizing social networks: A case study of u-conference. *Expert Systems with Applications, 38*(10), 12786–12790.

Kefalakis, N., Leontiadis, N., Soldatos, J., & Donsez, D. (2009). Middleware building blocks for architecting RFID systems. In *MOBILIGHT* (pp. 325–336).

Kim, M.-C., Kim, C. O., Hong, S. R., & Kwon, I.-H. (2008). Forwardbackward analysis of RFID-enabled supply chain using fuzzy cognitive map and genetic algorithm. *Expert Systems with Applications, 35*(3), 1166–1176.

Ko, J. M., Kwak, C., Cho, Y., & Kim, C. O. (2011). Adaptive product tracking in RFID-enabled large-scale supply chain. *Expert Systems with Applications, 38*(3), 1583–1590.

Lee, C., & Chan, T. (2009). Development of RFID-based reverse logistics system. *Expert Systems with Applications, 36*(5), 9299–9307.

Lee, C., Ho, W., Ho, G., & Lau, H. (2011). Design and development of logistics workflow systems for demand management with RFID. *Expert Systems with Applications, 38*(5), 5428–5437.

Lee, Y.-C., & Lee, S.-S. (2011). The valuation of rfid investment using fuzzy real option. *Expert Systems with Applications, 38*(10), 12195–12201.

Lin, C.-S., Chen, L.-S., & Hsu, C.-C. (2011). An innovative approach for rfid product functions development. *Expert Systems with Applications, 38*(12), 15523–15533.

Sheng, Q. Z., Li, X., & Zeadally, S. (2008). Enabling next-generation RFID applications: Solutions and challenges. *Computer, 41*(9), 21–28.

Su, X., Chu, C. -C, Prabhu, B., & Gadh, R. (2008). *The internet of things: From RFID to the next-generation pervasive networked systems (wireless networks and mobile communications)*. Auerbach Publications. Chapter On the creation of Automatic Identification and Data Capture infrastructure via RFID and other technologies..

Prabhu, B. S., Su, X., Ramamurthy, H., cheng Chu, X., & Gadh, R. (2005). WinRFID a middleware for the enablement of radio frequency identification (RFID) based applications. *Mobile, wireless and sensor networks: Technology, applications and future* (pp. 331–336). John Wiley & Sons, Inc.

Ustundag, A., Kilin, M. S., & Cevikcan, E. (2010). Fuzzy rule-based system for the economic analysis of RFID investments. *Expert Systems with Applications, 37*(7), 5300–5306.

Zafer & Aydogmus (2009). Implementation of a fuzzy-based level control using SCADA. *Expert Systems with Applications, 36*(3, Part 2), 6593–6597.