Application of Service-Oriented Context-Aware Architecture to Laundry Management System

Ufuk CELIKKAN, Kaan KURTEL

Izmir University of Economics
Faculty of Engineering, Department of Software Engineering
Sakarya Cad. No: 156, 35330, Balcova, Izmir, Turkey.
{ufuk.celikkan, kaan.kurtel}@ieu.edu.tr

Abstract: The laundry cleaning on an industrial scale has become a highly automated process, carried out by machines using sensors generating very detailed data. Data from these sensors allow the precise control of the laundry operation, often remotely. The combination of new machines with information technology has created a more efficient and cost effective process, enabled by software based on extensible architecture. The operations of a laundry exhibit all the properties of a context-aware system. This paper extends the authors' earlier work on context-aware laundry management architecture emphasizing business processes. In this study, we present architectural details that will aid implementers of the system in their choice of tools, techniques and technologies. In particular, a detailed account of laundry context information is presented. The proposed architectural framework used in the solution is able to monitor and autonomously manage laundry operations. The system employs a layered architecture which allows the separation of system components such as data capture, data processing and business services. An inference and a rule engine help to orchestrate system activities and transmit system status to interested parties through an interface. Tight coupling of system components are eliminated by web services.

Keywords: Context-aware systems, laundry management system, software architecture, web services.

1. Introduction

An essential part of many industries, clean and hygienic laundry services help businesses to improve quality, control costs, and provide customer satisfaction. Industrial laundry operations vary in size and scope, and are common in the hospitality, leisure, and healthcare industries. Other industries that use laundry services can be listed as sea travel for leisure and military purposes, fire departments, airlines, food industries, sport centres, schools, and apartments [9].

Laundry services exhibit similar characteristics of businesses operating in complex, heterogeneous and dynamic environments. One of the primary purposes of a laundry management system (LMS) is to organize and monitor different laundry processes at various levels, to ensure accurate and correct laundry activities that increase in productivity, quality, customer loyalty, and satisfaction.

This article presents an architectural framework to demonstrate how features and services of an industrial laundry operation can be implemented using a novel context-aware software approach. This solution contains a detailed presentation of laundry operations, its context model, interactions between the architectural components and the rule engine, and depicts the workflow engine orchestration.

The context consists of the various parameters that affect the operation of the laundry, andmust therefore be controlled and monitored.

The laundry operations in hospitals and hotels are very different. Therefore, even though the context in either setting may contain same data items (i.e. pH value, temperature), different data values are assigned to context items due to differing constraint; thus, we may have several contexts in the same application domain, rather than a single one. For instance hospital and hotel laundries present different contexts, even though they both belong to same laundry domain. Naturally, context content also varies from one domain to another [7]. For instance, for the laundry domain, a context may consists of location, collection time, item weight, identification tags, washer temperature and detergent type; in contrast, for the automotive domain, a context may consists of speed of a car, oil level, and cooler level. Context information can further be classified as being primary or secondary context type to better characterize a situation.

A context-aware system exploits the information provided in the context to deliver services to users or to other entities in the system [7]. Our proposal for a context-aware system possesses the features put forward by Dey et al. in [8] and described in following terms “information and services are presented to a user, a service is executed, and context is linked to information for later retrieval”. Together these aspects provide a powerful way of building scalable systems and dynamic adaptable business processes in a distributed environment.

This paper is an extended version of our work published in [5], expanding the scope by presenting a more detailed analysis of the laundry operations, laundry context model and context interpretation. This paper significantly advances the discussion on rule engine operations and workflow engine orchestration.

The structured of our paper is as follows: Background research is presented in Section 2. An outline of typical laundry operations is given in Section 3. The software architecture and proposed laundry management system is depicted in Section 4. A practical scenario from the health sector is presented in Section 5. Finally, we present the conclusions.

2. Related Works

A Context-aware system adapts and adjusts itself without needing an explicit user intervention in response to changes in its context. The concept of decomposition guided many researchers in their development of the context-aware architectural models. The main goal of using decomposition is to construct a layered architecture, augmented with encapsulation techniques so that context acquisition is separated from its use. Functionally separating layers from each other allows developers to hide lower layer details from higher layers.

An important consequence of using layered architectural approaches is that they easily lend themselves to sharing and re-use of software and hardware components.

Business logic functions, user interface, and data acquisition can easily be separated from each other in a layered approach. One of the conceptual models for context-aware applications proposed by Alisto et al. modularizes the system using five layers: physical, data, semantic, inference, and application [2]. Despite this complexity, however, for those familiar with Model-View-Controller (MVC) architectural pattern, it is relatively easy to map these five layers to the layered abstraction of MVC. Rehman et al. [20] has proposed architecture based on the MVC pattern with a special emphasis on user-application interaction. Another equally important topic in context aware systems is suggesting a representation and storage for context data, in order to simplify the processing and transfer of it as much as possible by the applications. Therefore, successful development and maintenance of context-aware applications depends on selecting a flexible and easy-to-use context model.

Several resources provide valuable information, and a clear vision of the context modelling approaches [19, 21]. Among these, ontology based models possess attributes such as simplicity, flexibility, and extensibility which make the model expressive and generic. Jung et al. [12] proposed an ontology-based context model that specifies concepts, and relations between them in terms of context.

Context-aware architectures have been applied to a wide spectrum of application domains, among which the mobile computing segment having the largest share. Detailed reviews by Baldauf et al. [3] and Hong et al. discuss several context-aware systems [11]. Context-aware applications surveyed in these studies include building smart home and hospital environments [16], class room and tourism [1]. Additionally, Hong et al study lists several other applications in the areas of information and decision support systems, mobile systems and web services. Truong et al. studied context-aware web service [24] and Miraoui et al. [15] surveyed pervasive context-aware architectures. Celikkan and Kurtel [6] and Merezeanu et al. [14] proposed platforms which separate context data retrieval and storage from the business logic. The PCAD platform [6] by Celikkan and Kurtel presents platform architecture for context aware application development, enabling context aware applications to be built rapidly using services provided by the platform. The platform manages the common work such as context acquisition and storage, therefore enabling the application to focus on the domain specific logic. Merezeanu et al. [14] integrated Wireless Sensor Networks, IoT and Cloud Computing technologies in a framework built as a layered architecture to provide context-aware sensing, computing and communication capabilities to industrial applications.

The work reported in [5] proposed a context aware system to manage laundry business processes. Many studies in the literature focus on particular aspects of laundry system implementations without addressing the end-to-end laundry processes issue.

For example Lu et al. [13] proposed a service-oriented software system architecture using washable RFID tags. Their study concentrated on tracking items between the factory and laundry shop but did not consider the processes taking place in the factory. Tajima et al. [23] proposed a context aware hanging system for laundry items using hangers which automatically detect the status of clothes, and inform users via multiple media. Van et al. [25] proposed an RFID based LMS consisting of RFID tags and also wireless RFID hangers. Noor et al. in [17] designed and developed proposed a "smart basket" system in order to optimize the resources used during laundry activities.

3. Laundry Operations

In this section, we describe the operations of a laundry from a business processes perspective in order to reveal the fundamental business functions and services. The Laundry Management System (LMS) technical architecture is presented in Section 4.

The core laundry business processes typically contains four major activities: 1) Collection and Transportation, 2) Administration/Back Office operations, 3) Scheduling, and 4) Cleaning, shown in Figure 1.

A typical LMS workflow is shown in Figure 1, in which important steps are marked by numbers in black circles. Some steps are performed manually. The workflow starts with an order from a customer❶, followed by collection and transportation of unwashed laundry❷.Next is the cleaning process, which includes automated operations performed by different machines. Upon completion of the cleaning process, items are first sent to packaging❸ then to customer service❹, which is responsible for on time delivery and calculation of the bill. Finally, the cleaned laundry is returned back to the customer❺. At every step of the operation, a cost control process ❻ ensures the costs are minimized.

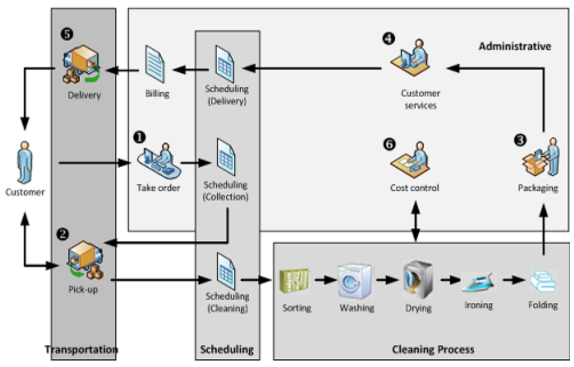


Figure 1. The workflow of LMS [5]

Collection and Transportation: Collection and Transportation process includes dirty laundry collection, transportation to factory, and the return to the customer to an agreed schedule. Since certain items need to be treated according to their condition, attention must be paid during collection. For instance, it is important that linens and bedding used in surgical operations are collected and treated separately from other laundry at every stage.

Administrative: Effective management of administrative activities is vital to maintain a healthy and sustainable financial position. Order taking, billing, inventory and cost control measures, pricing, customer service, and shipment are among many activities which have to be monitored and managed to effectively control a company's financial affairs. For example, costs are influenced by the choice correct type and amount of detergent used during cleaning and identifying optimal delivery routes with respect to time and distance during transportation.

Scheduling: Collection, delivery, and cleaning are the operations included in Scheduling. The effectiveness of these operations contributes to customer satisfaction and allows a company to realize its business goals.

Cleaning: The core process of a laundry business is the cleaning process which is a highly complex operation requiring a high degree of control and monitoring. Performing the cleaning process efficiently depends on using different rules and parameters, determined by the industry. Activities in the cleaning process are carried out by sophisticated machines or by humans, or in some cases, a combination. There may be optional services like repairing and dry cleaning, which may happen before or after the main cleaning process, and are not considered part of the core activities. Innovative and well-designed IT services enhance the accurate and timely functioning of the whole operation.

During the cleaning process, the machinery generates context data through its sensors. Certain context data must be processed and acted on in real time since it directly affects the cleaning process. Table 1 gives a list of these factors of laundry process data, which include, the amount and type of cleaning agent, and ingredients, water temperature and cycle time.

Any inappropriate value in these key parameters will increase cost and decrease cleaning effectiveness. The management of these factors needs a knowledge-based information infrastructure and a technical architecture.

4. Technical Architecture

A robust and extensible technical architecture design separates the complex business logic from the other components, using design principles such as abstraction, encapsulation and layering. This is particularly important for a context-aware system such as the proposed LMS, where hardware and software components work together. By encapsulating functionality into layers, and providing a well-described service points, tight coupling of the functions are eliminated, components become more cohesive, and a higher degree of reusability is ensured. Coupling and cohesion are important indicators of flexible, reliable and maintainable software components. For example, the sensor software is dependent on the sensor hardware. This dependency must be separated from the rest of the system using abstraction. Context information collected by the sensors is consumed by business logic, which can be implemented using different techniques, while the business rules can be managed through a user interface, therefore, the functions of core business logic and the user interface must also be abstracted away.

Table 1. Factors that affect the cleaning process

Factor	Explanation
Length of Cycle	
Too Short	The linen will not be cleaned.
Too Long	Risk of clothes becoming dirtier since soil will be redeposited. Unnecessary wearing away of clothes.
Temperature of Water	
Too high	Damage to linen.
Inadequate	Chemicals will not work.
Level of Water	Laundry agents are rendered ineffective due to ,incorrect 'dip' levels altering the laundry agent concentration.
Amount of Detergent	
Too little	Incomplete cleaning process.
Too much	Detergent residuals after rinse cycle
Type of cleaning agent [4]	
Water softeners	A chemical agent that remove calcium in water.
Bleach	Remove fruit, vegetable and food coloring stains.
Enzymes	Remove recalcitrant stains such as proteins and fats.
Load	
Overloaded	These causes articles to be tightly packed, which results in inadequate friction, therefore deeply embedded soil is not removed.
Under loaded	Enough centrifugal action but inadequate friction.
Hydro Extraction time	
Not at all	Silk should not be hydro-extracted.
Too Short	Increased drying time hindering operation of finishing equipment.
Too Long	Unnecessary use of resources.

Service-oriented computing (SOC) augmented with context awareness sets the foundation for developing an agile laundry management system that merges business processes with the information technology in which distributed components cooperates among them using a service-based approach [5]. At the core of the SOC paradigm lays the Service-Oriented Architecture (SOA) that employs web services as autonomous and platform-independent computational entities providing services to other applications or services. SOA uses Internet-based standards, such as Simple Object Access Protocol (SOAP) for message interchange between two services, and Web Service Definition Language (WSDL) for service interface definition [18].

4.1 Software System Architecture

LMS architecture is based on a layered model. The model partitions the system into functionally disjoint but cohesive layers, each of which provides its services through a well-defined public interface. Layers interact with others according to a strict order, using the interface. A layered model also has the advantage of being reliable, serviceable and maintainable. Figure 2 depicts our layered model, and the purpose of each layer is explained below.

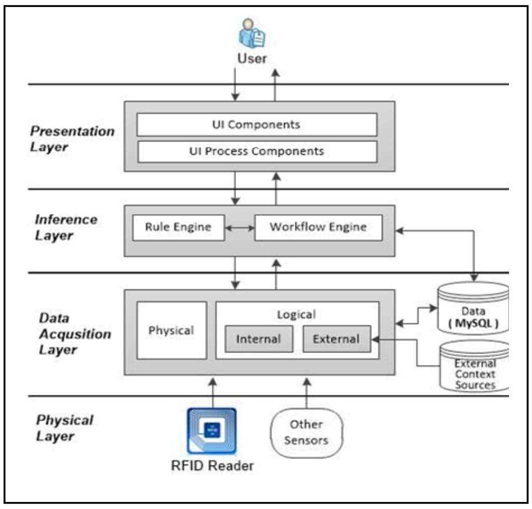


Figure 2. Context-Aware architecture for the LMS

Physical layer - This layer provides an abstraction of physical and virtual sensors [3] connected to the system through some communication mechanism (PLC, IP-based). Physical sensors capture a snapshot of the state of the cleaning process and machine status. Raw sensor data is accessed through drivers and passed to the data-acquisition layer for further formatting and processing. Virtual sensors include software which gives the impression that soft data, such as GPS position or real-time camera stream, is coming from a sensor. They use programming Interfaces or web services to receive data from external sources. In many situations, context data becomes more meaningful and relevant when it is aggregated. Instead of using a single data source, context aggregation uses several physical and virtual sensors to generate combined data, which leads to a better and more accurate interpretation of the state of the system.

Data Acquisition Layer – The primary responsibility of data acquisition layer is context data storage in some repository (e.g. database). Some examples of context data are washer temperature, water pH value, and load weight. These values, obtained through sensors, are collectively used to determine the specific wash program for each type of linen. The application data, user profile and the rules are also stored and managed by this layer. The application data represent business oriented operational data, such as user profiles and business rules. In addition to obtaining data from sensors directly, this layer is able to retrieve and aggregate context data from external sources through the virtual sensor abstraction using, for example, web services, and store it in its database. As an example, the external context aware application development platform PCAD [6] provides an interface to make its context data available to those requesting it.

In this case, PCAD will be the context data provider to LMS, and LMS will be context processor. This integration can be even made tighter by eliminating context data storage from LMS and transferring the data storage responsibility to PCAD. However, LMS is still Responsible for maintaining business related operational data on its own storage.

Inference Layer - Inference layer plays an important role within the system by coordinating the processes through via its workflow engine, and by making inferences and reasoning using the information stored in the database.

Presentation Layer – Presentation layer contains usability components and navigational elements enabling the users to interact with the system using services the layer.

4.2 Context Model

Choosing a flexible model that allows an easy definition and storage of data is even more important than accurately specifying contents of the laundry context when implementing a context-aware application. As indicated earlier, laundry operations are a series of processes including physical, machine executed processes and logical, backend office operations. Context data is obtained from both these sources using physical and virtual sensors which transfer pertinent information and furnish valuable services to the user. Thus the correct operation of the laundry system is achieved by reacting to the data acquired from these sensors. This is indeed the main idea as stated in the context-aware definition given by [7, 8].

A clear understanding of the context types is an essential aspect of the system design and development in a context aware application. Dey et al. [8] used a two-tier approach, classifying context types as primary or secondary, where primary context types are deemed more important than the secondary. Primary context types, as reported in [8], are as follows: *"location (where), identity (who), time (when) and activity (what)"*. Context information of secondary type is attributes of one of the primary types, and therefore each secondary type can be indexed to a primary type. We shall follow the same taxonomy in our context modelling. Table 2 lists the primary context types for the laundry domain and the questions that capture the context data from user's environment [3, 8]. Table 3 provides a list of attributes that are included in context information for a washing machine, and the sources of these attribute values.

To better illustrate the context model, and the contents of a context, the washing machine's cleaning process can be taken as an example. Garments are made from different kinds of material with diverse fibre types. Consequently, a range of detergents are manufactured to protect the different fabric types. In addition, an extra challenge may be caused by the detergent used and drying conditions, which may cause allergic reactions in the users. The type and amount of detergent used is naturally part of the context. Because of the damage or shrinkage caused during the washing cycle, it is essential to track the number of times each item is washed, particularly in the hotel industry, so that articles such as towels, bedding and linen can be discarded at the appropriate time. This information reduces sorting time, increases efficiency, and decreases labour costs. Items must also be tracked en-route from customer to factory shop and back to the customer.

Therefore, the "id" of a garment and number of times it has been cleaned is also part of the context. RFID tags are not only an effective solution to tracking, but also significantly improve sorting process accuracy and reduce the time needed. Using an RFID reader, it is possible to count the number of shirts in a basket, or to find clothes that have been misplaced according to size.

Automatic machines, such as washers and dryers, are equipped microprocessors to control the washing operation and sensors which continuously produce data either synchronously on user request, or asynchronously in response to a change in condition. These sensors give raw data, such as temperature, pH value and hardness of the water, which make up part of the context data. Sources other than the cleaning and drying machines generate context data, such as weight and time which also become part of the context data. Because of the increasing complexity of the laundry process, the accurate specification and efficient processing of context data is essential for setting the optimum conditions in a laundry management system. Table 4 shows optimal values for some of the context information.

Gruber in [10] and Struder et al. in [22] has the following definition *"ontology is a formal explicit specification of a conceptualization where conceptualization an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon"*. Formal reasoning techniques can be used as ontology based modelling is a formal representation. This type of modelling possesses several of the desired properties: simple to use for the developers, extensible, generic and rich enough to allow maximum context description [19, 21]. Extensibility is of particular importance for a domain such as the laundry management system, because it is important that the addition of new context elements is simple. OWL has been chosen to represent our context. OWL is intended to be used when information is processed by applications rather than presented to the humans. The below OWL example demonstrates the relationship between the water and its pH value.

```
<rdf:RDF
  xmlns:rdf=http://www.w3.org/1999/02/22-
rdf-syntax-ns#

xmlns:rdfs=http://www.w3.org/2000/01/rdf-
schema#
xmlns:owl="http://www.w3.org/2002/07/owl#"
>

  <owl:Ontology rdf:about="">
    <owl:versionInfo>
      $Id$
    </owl:versionInfo>
  </owl:Ontology>

  <owl:Class rdf:ID="Water">
    <rdfs:label>Water</rdfs:label>
    <rdfs:comment>
        Water used in Laundry Process.
     </rdfs:comment>
  </owl:Class>

  <owl:DatatypeProperty rdf:ID="PHValue">
   <rdfs:range
rdf:resource="&xsd;double"/>
   <rdfs:domain rdf:resource="#Water"/>
  </owl:DatatypeProperty>
</rdf:RDF>
```

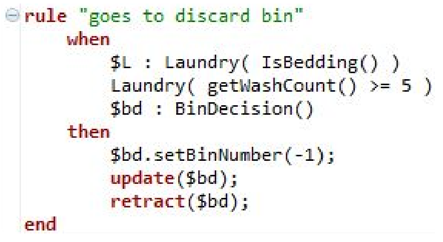
4.3 Context Interpretation and Inference

A LMS is a composite application with many subsystems and sub-processes, each of them supplying information to the system. Logistics, finance, administration and cleaning are the major subsystems. The processes in a laundry system are classified as information and business. While an information process is responsible for creating tasks that manage the information flow, a business process describes the laundry activities and uses business rules to control business process operations. Business rules are expressions of an organization's constraints, and create a set of guidelines for the business processes. Workflow engine drives both the information and business processes, and oversees the progress of the transaction as the operation moves from process to process. The workflow engine is capable of accessing services of the rules engine through a web service, and the rules engine uses the data obtained from context sources to drive the workflow engine.

4.4 Rule Engine as a Web Service

One of the components of the inference layer is the rule engine responsible for executing the rules controlling the business logic. The information and business processes are driven by the business rules using the services of the rule engine.

rule: TimesWashed
when:
  The number of times item is washed >= 5
then:
  Do not sort the item
  Add item into discard bin



```
rule "goes to discard bin"
    when
        $L : Laundry( IsBedding() )
        Laundry( getWashCount() >= 5 )
        $bd : BinDecision()
    then
        $bd.setBinNumber(-1);
        update($bd);
        retract($bd);
end
```

Figure 3. A rule example and its implementation in Drool

An example rule is shown in Figure 3. Despite the close relationship between information/ business processes and business rules, these two are decoupled.

One result of decoupling rules and rules engine from the architecture via web services is plugability: now it is possible to replace the representation of the business rules and the engine executing them without affecting the business processes. When web service model is used, the rule engine assumes the role of a web service server; the business process assumes the role of a client. The business process will

```
<!-- Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.6 in
JDK 6.-->
<definitions xmlns:soap=http://schemas.xmlsoap.org/wsdl/soap/
xmlns:tns=http://wsServer.ieu.com/
xmlns:xsd="http://www.w3.org/2001/XMLSchema"xmlns=http://schemas.xmlsoap.org/wsdl/
targetNamespace=http://wsServer.ieu.com/ name="GetContextService">
<types>
  <xsd:schema> <xsd:import namespace=http://wsServer.ieu.com/
                     schemaLocation="http://localhost:8080/wsContextServer?xsd=1"/>
  </xsd:schema>
</types>
<message name="getPHValue">
  <part name="parameters" element="tns:getPHValue"/>
</message>
<message name="getPHValueResponse">
  <part name="parameters" element="tns:getPHValueResponse"/>
</message>
<portType name="GetContext">
   <operation name="getPHValue">
         <input message="tns:getPHValue"/>
                          <output message="tns:getPHValueResponse"/>
   </operation>
</portType>
<binding name="GetContextPortBinding" type="tns:GetContext">
   <soap:binding transport=http://schemas.xmlsoap.org/soap/http style="document"/>
   <operation name="getPHValue">
        <soap:operation soapAction=""/>
        <input> <soap:body use="literal"/> </input>
        <output> <soap:body use="literal"/> </output>
   </operation>
</binding>
   <service name="GetContextService">
        <port name="GetContextPort" binding="tns:GetContextPortBinding">
            <soap:address location="http://localhost:8080/wsContextServer"/>
        </port>
    </service>
</definitions>
```

interact with the rules engine using solicit-response port type. Modification of the software, the port and the operation calls are all possible in the web service model without changing client business process. Workflow invokes a rule on the rules engine via the web service, possibly providing data as well. The rule engine performs the computations, comparisons and controls using this data and returns a response upon which the workflow engine changes the flow.

The workflow engine can also choose a path without the involvement of the rule engine. It invokes a web service to request a piece ofcontext data, such as the pH value of the water, and determines the most appropriate path. The following example shows the WDSL of a web service to retrieve the pH value of the water.

4.5 Workflow Engine

The workflow engine manages the complex business processes composed of tasks and sub tasks. It orchestrates LMS processes by sequencing, scheduling routes and monitoring these tasks. The workflow engine handles the location of a garment or pH value of water using a web service.

A workflow specifies how the available tasks are utilized in response to events. One can query the state of a process, and if necessary, to alter the flow, via a graphical user interface front end to the workflow engine. Rule engine and workflow engine cooperatively work together when changing a workflow path or starting or terminating a task. This cooperation processes (i.e. workflow businesses). An example workflow for the sorting sub process given in Business Process Model and Notation 2.0 is shown in Figure 4, while Figure 5 demonstrates how the workflow engine interacts with subsystems using web services.

Interaction with the external context sources are done via the interfaces advertised by the external data source. Some common interfaces used to exchange data are Web services and XML. PCAD [6] provides an API via a library, and an HTTP request-response mechanism to provide interoperability with other systems.

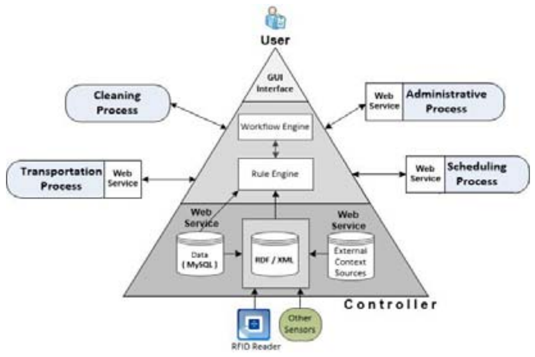


Figure 5. System-wide process view of the LMS

5. A Practical Scenario

In this section, in order to illustrate the functioning of the system, we present a context-aware scenario in which a patient named Mr. George who is under surveillance in a hospital's Infectious Diseases Unit tested positive for H5N1 bird flu virus. *"Mr. George's situation is critical and there is great risk of spreading the virus to others. His octors are very careful and want to be sure that his personal clothes, beddings and laundries are disinfected properly duringhis stay in hospital. Additional precautions are needed while handling this case, prompting the hospital cleaning service to make extra effort to fulfil the hygiene requirements for the patient and the hospital. When the service collects his clothes, beddings and towels from his room, they ensure that the laundry process is performed following strict guidelines."*

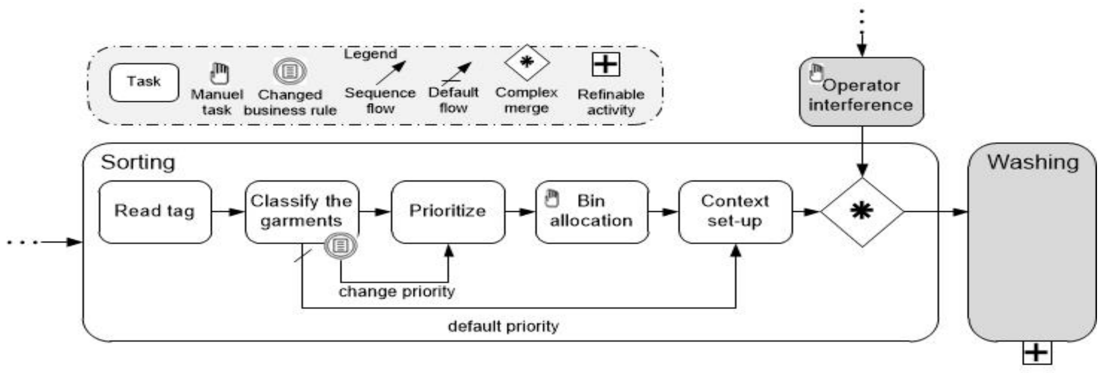


Figure 4. A workflow example

The scenario above demonstrates an application of context-aware LMS described in this paper. It exhibits several pieces of context information that must be considered by the system. The hospital bedding used by the patient must be tracked using an RFID tag attached to Mr. George's laundry items. RFID tag informs the LMS that the items belong to Mr. George and consequently, allowing LMS determine how parameters need to be set during the laundry process. Due to the infectious nature of the disease, the items must be washed separately at above a certain *temperature,* using a special *cleaning agent,* and dried with a special dryer within a *temperature range*. At the same time, the hospital must control the costs incurred due to the special handling of the items. A context-aware laundry system can monitor and alter the cleaning process using the context information acquired for Mr. George, which, in this scenario, consists of item *ids* (RFID tag), *detergent type*, *temperature*, *drying conditions* (inside a dryer), *time*, and *cost*.

The flexibility of the system allows the addition of other context information. As an example, the hospital policy may dictate that in highly infectious cases, the items must be discarded after 5 washes or that such items must not be used in other hospital departments. These types of requirements can easily be incorporated by adding other attributes into the context, namely the *number of times an item has been washed*. The architecture presented in the paper allows the monitoring of the process from start to finish allowing the operators to intervene if necessary. The system components interact with each other using web services, providing a constant flow of information among sub-processes in a standard way when needed.

6. Conclusion

A Laundry Management System solution based on the principles of context-aware and service-oriented architectures is proposed in study.Two major processes define LMS: business and information, each of which can be decomposed into sub processes. Business processes are driven by business rules and workflows, whereas information processes are driven by information technology. We proposed a layered architecture that is modular, extensible, and cohesive. It also uses web services to provide the highest degree of decoupling of system components and the agility needed in the current IT environment. Web services allow business processes to be decoupled from business rules, and context data providers to be decoupled from the rules engine enabling remote and local context acquisition. Ontology based context modelling is used, as this is the most expressive model. The proposed system facilitates the convenient integration of information technology infrastructure with an industrial cleaning system for hotels, hospitals or other businesses using cleaning services. The service-oriented context-aware architecture presented here offers seamless integration of the laundry management system with the partners. This architecture makes the integration and unification of various departments within a laundry business very straightforward.

Acknowledgments

The authors thank to the Scientific and Technological Research Council of Turkey - TÜBİTAK for their support on the 3001-Starting R&D Projects Funding Program for the project of "Architecture, Design and Implementation of Context Aware Service Platform", Grant No: 114E938.

REFERENCES

1. Abowd, G. D., et al. (1997). Cyberguide: A mobile context-aware tour guide. *Wireless Networks*. 3(5), 421-433.
2. Ailisto, H., Alahuhta, P., Haataja, V., Kylloenen, V. & Lindholm, M. (2002). Structuring context aware applications: Five-layer model and example case. In the *Ubicomp Workshop on Concepts and Models for, Ubiquitous Computing*.
3. Baldauf M., Dustdar S. & Rosenberg F. (2007). A survey on context-aware system. *International Journal of Ad Hoc and Ubiquitous Computing,* 2(4), 263-277.
4. Broze, G. (1999). *Detergents: Technical and Practical Challenges*. Handbook of Detergents. Part A: Properties (Surfactant Science), Guy Broze (ed.).
5. Celikkan, U.&Kurtel, K. (2013). A Context-Aware Architecture for the Management of Laundry Business Processes. *14th European Conf. on Knowledge Management - ECKM 2013*, Kaunas, Lithuania, 159-166.

6. Celikkan U. & Kurtel K. (2015). A Platform for Context-Aware Application Development: PCAD. *Federated Conf. on Computer Science and Information Systems (FedCSIS)*, 5, 1481-1488.

7. Dey, A. K. (2001). Understanding and using context. *Personal Ubiquitous Computing,* 5(1), 4-7.

8. Dey, A. K. & Abowd, G., D. (2000). Towards a Better Understanding of Context and Context-Awareness. *Proc. of the Workshop on the What, Who, Where, When and How of Context-Awareness*, ACM Press, New York.

9. Girbau Group, (available at http://www.girbau.com/ accessed on 22.03.2017).

10. Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2), 199-221.

11. Hong, J. Y., Suh, E. H. & Kim, S. J. (2009). Context-aware systems: A literature review and classification. *Expert Systems with Applications*, 36(4), 8509-8522.

12. Jung H., Yoo S. & Park S. (2012). Context Modelling Using Semantic Web Technologies, *Studies in Informatics and Control*, 21(2), 173-180.

13. Lu, Y. & Yu, H. (2010). A Flexible Architecture for RFID Based Laundry Management Systems. *Wireless Communications Networking and Mobile Computing*, 6th Int. Conference, 1-4.

14. Merezeanu, D., Vasilescu, G., & Dobrescu, R. (2016). Context-aware Control Platform for Sensor Network Integration in IoT and Cloud. *Studies in Informatics and Control*, 25(4), 489-498.

15. Miraoui, M., Tadj, C. & Amar, B. C. (2008). Architectural Survey of Context-Aware Systems in Pervasive Computing Environment. *Ubiquitous Computing and Communication Journal*. 3(3), 68-76.

16. Munõz, M. A., Gonzalez, V. M., Rodriguez, M. & Fa vela, J. (2003). Supporting context-aware collaboration in a hospital: an ethnographic informed design. *Proc. of Workshop on Artificial Intelligence, Information Access, and Mobile Computing*, 330–334.

17. Noor, M. Z. H. et al. (2012). Design and development of smart basket system for resource optimization. *Control and System Graduate Research Colloquium (ICSGRC)*, 338-342.

18. Papazoglou M. P., Traverso P., Dustdar S. & Leymann F. (2008). Service-Oriented Computing: A Research Roadmap. *International Journal of Cooperative Information System*, 17(2), 223-255.

19. Perttunen, M., Riekki, J. & Lassila, O. (2009). Context representation and reasoning in pervasive computing: a review. *Int. Journal of Multimedia Ubiquitous Engineering*, 4(1), 1–28.

20. Rehman, K., Stajano, F. & Coulouris, G. (2007). An Architecture for interactive context-aware applications. *Pervasive Computing*. 6(1), 73-80.

21. Strang, T. & Linnhoff-Popien, C. (2004). A Context Modeling Survey. *First Int. Workshop on Advanced Context Modelling, Reasoning and Management at UbiComp.*

22. Studer, R., Benjamins, V. R. & Fensel, D. (1998). Knowledge engineering: Principles and methods. *Data Knowledge Engineering*, 25(1-2), 161-197.

23. Tajima, N., Tsukada, K. & Siio, I. (2011). AwareHanger: Context-aware hanger for detecting the status of laundry. *Pervasive 2011*, San Francisco, CA, USA.

24. Truong H-L. & Dustdar, S. (2009). A survey on context-aware web service systems. *Int. J. of Web Information Systems*. 5(1), 5–31.

25. Van, N. T. et al. (2012). An implementation of Laundry Management System based on RFID hanger and wireless sensor network. *Ubiquitous and Future Networks 2012 (ICUFN)*, 490-493.