A low-cost AR assistant component architecture for Warehouse Management Systems

Iakovos Stratigakis
Department of Informatics,Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
iakovosds@csd.auth.gr

George Kakarontzas
Department of Digital Systems University of Thessaly 41500, Larissa, Greece
gkakaron@uth.gr

Apostolos Ampatzoglou
Department of Applied Informatics, University of Macedonia, 54636 Thessaloniki, Greece
a.ampatzoglou@uom.edu.gr

Theodoros Amanatidis
Department of Applied Informatics, University of Macedonia, 54636 Thessaloniki, Greece
tamanatidis@uom.edu.gr

Thrasyvoulos Tsiatsos
Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
tsiatsos@csd.auth.gr

Charalampos Avratoglou
Entersoft, 362 Sigrou Av., 17674 Kallithea, Greece
avr@entersoft.gr

Dimitris Folinas
Supply Chain Management, Dept. International Hellenic University 60100 Katerini, Greece
dfolinas@gmail.com

Christina Volioti
Department of Applied Informatics, University of Macedonia 54636, Thessaloniki, Greece
christina.volioti@uom.edu.gr

Ioannis Stamelos
Department of Informatics, Aristotle University of Thessaloniki 54124 Thessaloniki, Greece
stamelos@csd.auth.gr

Alexander Chatzigeorgiou
Department of Applied Informatics, University of Macedonia 54636, Thessaloniki, Greece
achat@uom.gr

ABSTRACT

This work describes a research collaboration between universities and industry with the aim to provide a low-cost prototype based on Augmented Reality technologies, that assists with maintaining correct information in Warehouse Management Systems. The component interacts with the central server of an existing commercial WMS to provide up-to-date information on the actual state of the warehouse. The low-cost requirement restricts the solution to smartphones and other inexpensive equipment readily available, such as drones, as well as mostly Open Source Software. This requirement also introduces several interesting architectural issues that we discuss in this work. A prototype was built for the proposed architecture and several tests were carried out.

CCS CONCEPTS

• **Information systems** → Information Systems Applications; Enterprise Information Systems; Enterprise Applications..

KEYWORDS

Warehouse Management Systems, Augmented Reality, Low-cost architectures

ACM Reference Format:
Iakovos Stratigakis, George Kakarontzas, Apostolos Ampatzoglou, Theodoros Amanatidis, Thrasyvoulos Tsiatsos, Charalampos Avratoglou, Dimitris Folinas, Christina Volioti, Ioannis Stamelos, and Alexander Chatzigeorgiou. 2021. A low-cost AR assistant component architecture for Warehouse Management Systems. In *25th Pan-Hellenic Conference on Informatics (PCI 2021), November 26–28, 2021, Volos, Greece.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3503823.3503854

1 INTRODUCTION

The work discussed in this article takes place in the context of the WMS+AR (Warehouse Management System & Augmented Reality) project, which is co-funded by the EU and Greek national funds and is a collaboration between a software company and two universities in Greece. The project aims at developing a low-cost prototype that will augment the capabilities of an existing commercial WMS, mainly to allow the automated recording of the warehouse's layout, the automatic measuring of the products in the warehouse, and the management of exceptions which are discrepancies between the measurements and the stored data in the WMS database.

More specifically,

- The recording of the warehouse's layout is performed by a smartphone operated by an employee. Alternatively, the same operations can be performed by a suitably equipped drone. At the time of the layout automatic recording, the warehouse can be empty or filled with products.

- The automatic measuring of the products that are in the warehouse at any given time, again can be carried out by an employee using a smartphone or by a drone, based on barcode or size and location if barcode is unrecognizable.
- The notification of these measurements is sent back to the server of the WMS for the validation of the accuracy of the stored state in the WMS database. Exceptions can be of various types: (a) differences in quantity, (b) differences regarding the location of products, or (c) violations of constraints regarding the placement of products.

In the following sections we will first discuss related work in RELATED WORK, then we will provide the architecture of the prototype system we have developed in ARCHITECTURE EVALUATION. In ARCHITECTURE EVALUATION, we will discuss the evaluation that took place. Finally, in CONCLUSIONS, we will provide future directions and conclusions.

2 RELATED WORK

Various previous works have researched warehouse management systems and software architecture and technologies.

In [1] authors discuss architecture variability and feature models to represent the variety of warehouses and related equipment in the architecture of WMSs. Their goal was to shorten delivery times of WMSs. They identified more than 200 features organized in three hierarchical feature layers. This work as ours has an architectural focus for WMSs, but their goal is shorter delivery times and their method is based on variability, whereas our goal is inexpensive end-solution and our method is mostly using commodity hardware and open source software with a suitable architectural solution for the task.

Another study [2] looked at using RFID and XBee technologies as a solution to address human error in warehouse put away operations in real-time. The authors designed three modules (warehouse, racks and forklifts) that can reduce placement errors in real-time. Their approach as ours tries to reduce human errors in warehouses, but our approach differs since it is not concerned specifically with put away operations in real-time but looks also at other issues and takes place asynchronously to other warehouse operations. Also, our approach does not require extensive differences in hardware and operations than currently used in warehouses today and aims to extend an existing WMS.

Several other studies have also looked at the use of RFID technology for warehouse management systems. In [3] the authors looked at the integration of lean production and RFID technology with the aim to improve warehouse management. In [4] the authors carried a case study in the tobacco industry to demonstrate the use of RFID for improving the accuracy of inventory control in plain (i.e. non-automated) warehouses. Another study [5] looked at the challenges in warehouse operations at a branch of Shell Petroleum Development Company (SPDC) and how the use of an RFID-based inventory management system can help addressing various issues including inaccurate stock accounting in the warehouse. Our proposal does not use RFID technology but is based on existing barcodes for storage locations and items in the warehouse as well as in AR architectures capabilities such as image processing for improving the management of a non-automated warehouse.

In [6] the authors propose a conceptual solution for WMS operations based on two-dimensional barcode (QR code) identification. They propose a WMS design for such a system and provide an initial discussion of possible problems and solutions. Compared to our work, this work does not discuss additional functionalities addressing situations when the code is difficult to recognize, and there is no mobile device or commodity smartphones. Also, this work does not provide architectural analysis involving trials of the proposed solution.

In [7] the authors proposed the use of IPS (Indoor Position System) with Bluetooth for various operational aspects of the warehouse, including material management, job control management and location management with the use of tag information. The proposed system requires some new equipment but can be used in an existing warehouse. Although there is some discussion about algorithms, such as location tracking algorithms, the focus is different from our work that aims at pinpointing discrepancies between the real warehouse state and the database of an existing WMS. Also, the technologies used (such as Bluetooth) are different and there are no specific architectural aspects discussed.

Other efforts include, WMS system design using Aspect-Oriented Programming [8], Decision Support Systems (DSS) for the design and management of warehousing systems [9] and proposed architectures for intelligent warehousing operation management systems based on safety monitoring [10].

Concluding, our efforts aim at enhancing an existing WMS with capabilities that allow layout generation, product measuring and handling of discrepancies in an asynchronous manner at non-automated warehouses. Our proposed solution for these tasks using inexpensive commodity hardware and open source software combined with an original architecture, is original, since we could not identify something similar in the related literature.

3 SYSTEM ARCHITECTURE

In this section the architecture overview will be provided and the basic principles of the architecture design along with the affected quality properties will be discussed.

3.1 Architecture overview

In Figure 1, the high-level view of the software architecture is depicted. The components depicted are executed in the warehouse server which is a separate server that communicates with the central WMS server in which the Entersoft WMS is executed. The main components of the AR WMS Services are the following:

- **High Level - AR Actions API**: This component is responsible for the communication with the existing Entersoft WMS server via an API, so that the Entersoft WMS server can provide list of actions to the AR WMS Services subsystem. Actions can include requests for information (e.g. volume of parcel on a shelf etc.), but also instructions such as measurements.

- **Visualization Services**: This component provides a visualization of the warehouse with information provided by the various devices (hand-held and drones). Colors (red, yellow, green) are used to highlight exceptions. Also sizes of the various objects are depicted accurately and a visual representation of the warehouse is provided to the Entersoft WMS client. All the information passed to this component from the other components of the AR WMS system are in JSON format.
- **Manager**: The manager component is responsible for coordinating the other components. It receives action requests from the High-Level AR Actions API component and forwards these requests to the motion controller. Also, schedules actions and keeps track of their state (pending, executing, completed, failed). To improve accuracy, the component accumulates multiple measurements of the same item and provides to the Entersoft WMS server a median of these measurements, thus eliminating minor differences.
- **Motion Controller**: This is the subsystem responsible for controlling the Drones. It provides instructions for performing an action and for returning to their dock position. It also applies a shortest-path algorithm to calculate the route that a drone should follow to complete an action possibly with multiple steps.
- **Motion Device**: This is a device capable of autonomous movement in the warehouse premises, currently a drone.
- **Hand-held Device**: This is a hand-held device operated by a warehouse employee. It fulfills the same purposes as the Motion Device (measurements and visualizations).
- **Visual Processor**: This component receives visuals and 3d models from the devices. It provides some additional functionalities such as barcode recognition and volume calculations. This is an example of a decision with architectural significance since it would be possible to perform such calculations in the devices. However, by delegating them to this component the devices save energy.
- **Dashboard**: The Dashboard communicates with (1) the High-Level AR Actions API (ordering of actions, listing of actions, cancellation of actions, listing of WMS-AR exceptions) and (2) the Visualization Server, via a REST API over HTTP(S).

Next, we will examine in some depth the main components.

3.2 Motion and hand-held devices and related components

These devices are being used in the warehouse to carry out precise measurements of the various products that are stored in the warehouse's storage locations. The hand-held device is being operated by an employee and initially we used Google Tango AR platform enabled phones. This however, changed in later phases of the project. The motion device is currently a drone and it can move autonomously in the warehouse's premises. In addition, these devices are used for the initial generation of the warehouse's layout. In Figure 2, the structure of the Manager component is depicted. The Manager component basically interacts with the High Lever AR Actions API component through the provided interface "ActionsAPIInterface". Through this interface it accepts requests for actions. In addition, it also interacts with the Entersoft WMS through a required interface to provide to the Entersoft WMS the various types of information for the warehouse's observed state. Furthermore, the component interacts with the MotionController through the MotionInterface in order to request actions.
A typical interaction related to the motion controller in general looks like this:

- Floor-plan generation.
- Post processing of floor plan which contains the following steps: (a) creation of a 2d representation array, (b) identification of the number of bays and provision of IDs to them, (c) binding of shelves to bays (user selects bays and shelf types), (d) provision of shelves ids.
- Action receipt for specific shelves and creation of 2 async tasks. The first task is to find the optimal route. The second is to forward that route to the drone.
- On arrival the drone performs scanning for barcodes. It also performs object detection and constructs the output.
- Finally, it interacts with the provided interface of the Visual Processor to provide models of the warehouse for 3D Rendering.

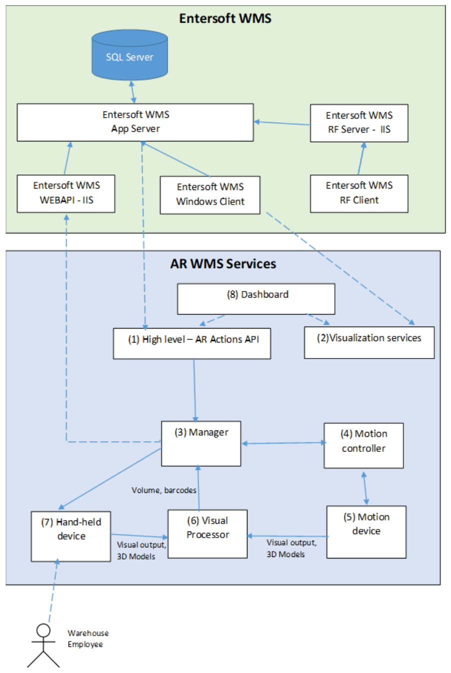

Figure 1: High-level architecture overview

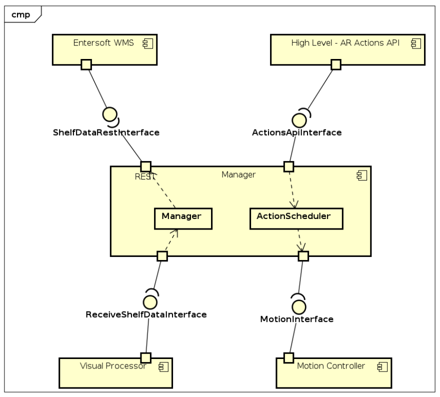

Figure 2: Structural view of the Manager component

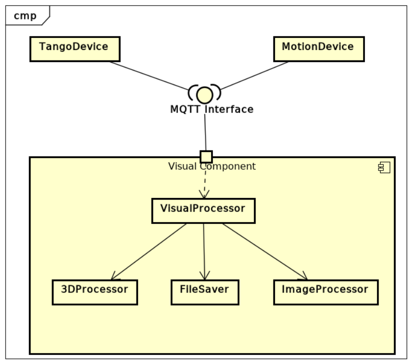

Figure 3: Structural view of the Visual Processor component

In Figure 3 the structural view of Visual Processor component is depicted which communicates with the hand-held (e.g. Tango) device and Motion Device components.

As can be seen from Figure 3, the Visual Processor component will communicate asynchronously with the Tango device and the Motion device, using MQTT (Message Queuing Telemetry Transport), a publish-subscribe-based messaging protocol. We made use here of the messaging architectural pattern for asynchronous communication specifically using MQTT since it is lightweight and suitable for the hand-held and Motion devices.

The related Non-Functional Requirements (NFRs) as well as the main architectural means to satisfy these NFRs, for the hand-held device and the motion devices are the following.

Energy Consumption-Efficiency for mobile devices: Motion devices in the latest prototype version are bare-metal dummy devices in our architecture: The motion device does not perform any computation. It is responsible only for sending the sensor data back to the warehouse server and receiving instructions. In this way, energy consumption is limited only to drone hardware energy requirements.

Optimized algorithms for the measurements process: After calculating the shortest path for a bill of actions, and before the device starts visiting the shelves, battery is checked. The algorithm calculates the maximum distance that the device can travel and respectively the selected N shelves to visit. If N< length(bill_of_actions), the remaining shelves remain in the task queue.

Mapping process: The server using a 3 seconds interval, calculates the distance of the device from its dock. In the case the battery is adequate only for returning back to the dock point, the device returns to the dock. This is also checked before the devices start their route to the next goal.

Energy Consumption-Efficiency for handheld devices: Handheld devices perform what motion devices do, i.e. they simply send sensor data back to the warehouse server. Again, as is the case with the drones, tango devices also consume less energy by 'outsourcing' the computations to the warehouse's server.

Precision: precision concerns barcode recognition, measurements in point-clouds and measurements in images:

- **Barcode recognition:** Under certain conditions the recognition of the barcode will be 100% accurate. These conditions are the following: (a) A certain level of brightness (b) Image analysis should be more than 480p (720x480), & (c) Distance from the barcode (i.e. the size of the barcode compared to the size of the picture should be more than 5%).
- **Measurements in Point-Cloud**: The volume of the containers is calculated. The free volume at a shelf is calculated as the shelf_volume − content_volume. With experimentation we have determined that we can achieve an accuracy of 2 decimal places for these measurements.
- **Measurements in images:** Users can insert photographs of the various items that are in the warehouse. Using feature and keypoint extraction in the provided photos, we can see if a certain photograph taken by a device matches a provided photo. By testing we have specified that a 70% match or more is possible. This is very important since it can provide a fallback pattern when the barcode recognition fails.
- **Speed of Layout Creation:** The time of creation and extraction of the rack/shelf positions, in the case of the hand-held device depends solely on the time required by a carrier to walk through the warehouse and it can be in the area of a few minutes. In the case of the motion device, it is still very fast, but it depends obviously on various factors such the motion devices number, the battery life, the environment complexity etc.

Speed of Measurements:

- **Barcode:** The time to locate and read barcode in a photo is less than 0.4 sec.
- **Point-cloud:** The time for segmentation, volume measurement and distance measurement from the edge of the shelf ranges from 0.5 sec to 3 sec.
- **Image:** The time required for the calculation of features, descriptors and keypoints in an image and matching to with the precalculated features of a dataset image, on a typical PC is around 12 seconds.

The final response time is not cumulative of the above. The response time is equal to the latest time plus a very short time for cross-corelating the image features with the features of the dataset

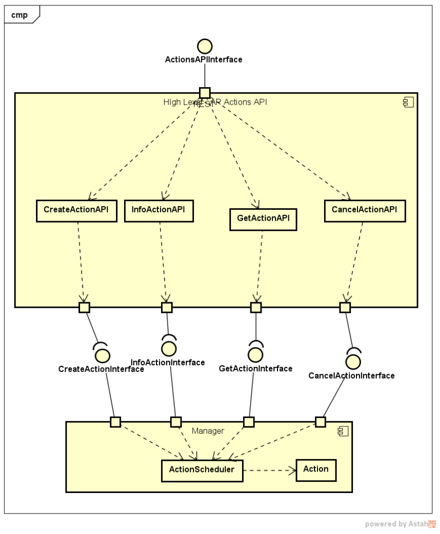

Figure 4: Structural view of the High Level - AR Actions API component

images, as everything runs in parallel using an OpenMP architecture. For the technologies used in the warehouse for the devices and the warehouse server we have used the following programming languages: C/C++, Python, Java, Bash and Lua. To avoid licensing issues and reduce the cost, we have modified sever OSS libraries used in the project including the following: ROS, PCL, FLANN, CUDA, OpenMP, OpenCV, Gazebo (Simulations), RabbitMQ, Boost, Eigen, QHull and Sci-Kit. Regarding the cost of the hardware used, it is the cost of the handheld Tango device, the cost of the Drones and the drones' dock stations as well as the cost of the warehouse server which can be a rather cheap but still powerful computer (~1,5K euros).

3.3 High Level - AR Actions API

The High-Level AR Actions API component serves as the communication channel between the WMS and the WMS&AR System. Particularly, the AR Actions API accepts incoming Actions from the WMS and forwards them to the hand-held devices via a message broker. These actions are incoming HTTP requests through a RESTfull web API and must follow a specific format that is acceptable by the AR Actions API. The API is documented in detail at http://se.dai.uom.gr:3333/swagger/.

In Figure 4 the structural view of High-level AR Actions API is depicted. As can be seen this component will provide an API intended to be used by the Entersoft WMS system, the ActionsAPI interface. Through this provided interface the WMS system will request actions from the WMS&AR Services middleware.

Actions can be Create, Info, Get and Cancel actions.

The related NFRs as well as the main architectural means to satisfy these NFRs, for the high-level AR Actions API component are the following.

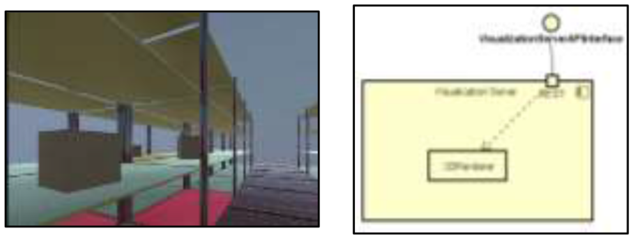

Figure 5: 3D Visualization of the warehouse (left) and UML design (right)

- Portability: To achieve maximum portability, the High Level - AR Actions API was designed to be deployed as a composition of independent Docker containers.
- Security: Every endpoint of the AR Actions API is protected with HTTP Bearer token Authorization. This means that only privileged users that hold valid Bearer Tokens can access the functionality of AR Actions API.
- Usability: To maximize understandability and learnability, every endpoint of the AR Actions API is documented in detail..
- Reliability : To guarantee the processing and execution of every incoming action from the WMS, all actions are persisted into a FIFO queue. Then a dedicated worker processes the actions and broadcasts them to the subscribed hand-held devices to begin their execution.
- Performance: The response time of the AR actions API for an incoming action ranges from 77 to 145 milliseconds. Every incoming action triggers some processes such as authentication and format validation. Finally, a broadcast of the action to the subscribed devices takes place. Due to the lightweight design of the AR Actions API and the high-performance capabilities of NodeJs and MongoDB the response time of the API is quite satisfying.

This component , regarding the cost is free, and the programming languages and technologies used include: TypeScript & JavaScript (NestJs framework), NodeJS, MongoDB, RabbitMQ (open source message broker) and Docker.

3.4 Visualization Service

This service creates the 3D visualization of the warehouse and its storage locations (Figure 5, left). The information of each storage location is updated according to the latest scanning session. Moreover, color annotations are used to signify discrepancies or exceptions between the central WMS and the AR system. The user can navigate into the warehouse and select to view detailed information, such as content, count, volume, package Ids, etc. by clicking on a specific storage location. The Visualization Service component rather minimal structure is depicted in Figure 5 (right). As can be seen the main responsibility of this component is to provide an API Interface to the Entersoft WMS by which a client (e.g. a browser) can ask for the rendering of various locations in the warehouse. The rendered images along with related metadata will be returned to the client.

The related NFRs to the visualization service are mainly performance attributes and more specifically are the following:

- **Speed of 3D Warehouse Creation**: The creation time of the 3D visualization of the warehouse is approximately 2 sec, regardless of the size of the warehouse. Creation time refers to the time that is needed to visualize the warehouse to the user.
- **Rendering speed**: The rendering speed depends on the size of the warehouse, meaning that if a large warehouse needs to be rendered, the time that is needed for the user to navigate to the whole warehouse might involve significant time delays. Therefore, the warehouse is divided into sections and only the specific part of the warehouse in which the user is located is rendered. Additionally, this approach enables the possibility to depict the latest information of each storage location in the 3D visualization of the warehouse.

The programming language that is used for the 3D Visualization service is C# and the software is Unity (free version). Unity must be licensed when used by companies, educational institutions or incorporated entities that had a turnover in excess of US$100,000 in their last fiscal year.

4 ARCHITECTURE EVALUATION

To evaluate the system as well as the architectural design at various stages, several trials took place, after a significant part of the prototype implementation was already available. More specifically 5 trials took place in the period commencing from October 21, 2019 until January 16, 2020[1]. The first four trials took place at a university-owned warehouse with the following characteristics: (a) area: 200 square meters, (b) number of racks: 1, (c) number of shelves: 15, (d) height: ~5-6 meters. The fifth trial took place at a real IKEA warehouse.

The **first trial** took place on October 21, 2019 and the release of the prototype was AR_WMS release 1.0 which was Google Tango AR platform oriented. In this version, all computation was carried out in the tango device and then sent to the AR&WMS server.

The process had the following steps:

- Creation of the warehouse layout in which first the device created an ADF (Area Description File), a binary file created by Tango after it “learned” a space. This allowed the device to detect its position in the future. Then, the layout in JSON format was generated.
- Receipt and display of incoming actions list in the tango device.
- Manual selection of an action from the list.
- Manual Go-To-Spot (an employee had to carry the device).
- Photograph of the area followed by barcode reading and object detection.
- Sending of the result back to the AR&WMS server.
- Repetition again from step 2.

The AR_WMS features included the following: (a) Floor Plan reconstruction, (b) Shelf location extraction (Grid over Floor Plan), (c) Real Time Device Localization, (d) On-Device Barcode Reading, (e) On-Device Object Detection.

[1]A short video of the trials is available at https://youtu.be/GrWJUd-GzTI

The most important conclusions from this trial, were that for power efficiency reasons it would be preferable to move barcode reading and object-detection to the warehouse server. Also, a memory scalability issue was observed since reconstruction speed decreased over time, due to device memory. Regarding usability carrying the device would be an issue.

The **second trial** took place on November 14, 2019 and was based on AR_WMS release 1.2, which was also Tango oriented. In this trial we also included the drone on which we mounted the Tango device. The drone used was a Phab 2 Pro-AR Drone. Regarding the drone we concluded that a drone with a simple camera can send photographs to the backend server where the processing takes place, but it cannot navigate autonomously without some depth information source. Regarding the mounting of the Tango device to the drone several issues were observed, such as (a) drone stability issues, (b) noisy data (cause of the lack of stability), and (c) power inefficiency since the AR Drone flight time without a mounted tango device was 12 minutes, whereas with the mounted Tango device was 5 minutes.

The basic conclusions of this trial, where that (a) we needed to look more carefully in the drone H/W requirements, (b) Use photography-based recognition by adding point-cloud processing for the various objects and (c) proceed to a Tango-agnostic architecture by moving the mapping process to the backend server and use Tango only as an input data source for the point cloud.

The **third trial** took place on December 4, 2019. The AR WMS release was 2.0, which was ROS-oriented (ROS - Robot Operating System). We carried out new mapping procedures and comparisons with the first version of the system.

The basic conclusion was to deploy and integrate ROS Octomap framework to the system, which is an open-source framework used to generate volumetric 3D environment models [11].

The **fourth trial** took place on December 12, 2019, and the system release was AR WMS 2.1 (ROS oriented). We tested and evaluated the segmentation process in point-cloud data (e.g. number of objects, dimensions etc.). Using this approach there is another information source available which will be unaffected by environmental variables such as the light conditions. The basic conclusion was that this approach provides a reliable information source. Therefore, our approach was modified to send back to the server the photograph and point cloud data per action as well, for post-processing. Hence, we added the required information to the result.

The **fifth trial** took place on January 16, 2020 at an IKEA warehouse. The goal of this trial was to proceed to performance testing on a real-scale warehouse. The release of the prototype was again AR WMS 2.1 (ROS oriented). The main conclusion from this trial is that we need to have a PC to play the role of the server in the warehouse since a low-end laptop would not be sufficient. However, a low-cost server around 1,500 euros will suffice.

5 CONCLUSIONS

In this work we discussed an architecture-centric approach for the enhancement of an existing commercial WMS, with additional components. These components aim at improving the correctness of the WMS data as well as the process of the initial warehouse layout insertion in the WMS. We have used exclusively inexpensive hardware and software and used an innovative architecture that we discussed in this work to provide a satisfactory solution regardless of the reduced cost requirement.

Future steps include, more testing in real warehouses. Also, the prototype requires significant effort to become a commercial product, since the project's goal is merely to prove the feasibility of the proposed solution. To this end our efforts have been successful so far.

ACKNOWLEDGMENTS

This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code: T1EDK-03502).

REFERENCES

[1] Fang, M., Leyh, G., Elsner, C., Doerr, J.: Experiences During Extraction of Variability Models for Warehouse Management Systems. In: 2013 20th Asia-Pacific Software Engineering Conference (APSEC), pp. 111-116, IEEE (2013).
[2] Bektaş, B. and Korkmaz, H.: RFID and XBee based automated verification of put-away operation for Warehouse Management Systems. In: 15th International Conference on Intelligent Systems Design and Applications (ISDA), pp. 635-641, IEEE (2015).
[3] Chen, J.C., Cheng, C., Huang, P.B. *et al.* Warehouse management with lean and RFID application: a case study. Int. J. Adv. Manuf. Technol. 69, 531–542 (2013).
[4] Wang, H., Chen, S. and Xie, Y.: An RFID-based digital warehouse management system in the tobacco industry: a case study, International Journal of Production Research, 48:9, 2513-2548 (2010).
[5] Adoga, I. and Valverde R.: An RFID based supply chain inventory management solution for the petroleum development industry: A case study for Shell Nigeria. Journal of Theoretical and Applied Information Technology, Vol. 62, No.1, pp. 199-203 (2014).
[6] Wu, J., Xu, D. and Li, Z.: Research on Key Techniques of Warehouse Management System Based on Two-Dimensional Bar Code. 2013 International Conference on Computer Sciences and Applications, Wuhan, pp. 353-356, IEEE, 2013.
[7] Kim, G.-H., Kwonm, O.-H. and Oh, A.-S.: Design of Warehouse Management System Using IPS under Bluetooth Environment. International Journal of Multimedia and Ubiquitous Engineering, Vol.11, No.6, pp.281-288, (2016).
[8] Cheng, L., Didi, X., Mingyong, L. and Yan, W.: Design and Implement of Warehouse Management System Based on AOP. IEEE International Engineering Management Conference, pp. 243-246, IEEE (2006).
[9] Accorsi, R., Manzini, R. and Maranesi, F.: A decision-support system for the design and management of warehousing systems. Computers in Industry vol. 65, issue 1, pp. 175–186, Elsevier, January 2014.
[10] Liu, J., Yang, H.-Y. and Jiang, T.: Intelligent Warehousing Operation Management System Based on Safety Monitoring. Journal of Applied Sciences, Volume: 13, Issue: 11, Page No.: 1975-1979, 2013.
[11] Hornung, A., Wurm, K.M., Bennewitz, M. *et al.* OctoMap: an efficient probabilistic 3D mapping framework based on octrees. Auton. Robot 34, 189–206, Springer, 2013.