Adaptive Multi-agent System for a Washing Machine Production Line

Nelson Rodrigues[1,2], Arnaldo Pereira[1], Paulo Leitao[1,2]

[1] Polytechnic Institute of Bragança, Campus Sta Apolonia, Apartado 1134, 5301-857 Bragança, Portugal, {nrodrigues, arnaldo, pleitao}@ipb.pt
[2] Artificial Intelligence and Computer Science Laboratory, R. Campo Alegre 102, 4169-007 Porto, Portugal

Abstract. This paper describes the implementation of a multi-agent system in a real industrial washing machine production line aiming to integrate process and quality control, allowing the establishment of feedback control loops to support adaptation facing condition changes. For this purpose, the agent-based solution was implemented using the JADE framework, being the shared knowledge structured using a proper ontology, edited and validated in Protégé and posteriorly integrated in the multi-agent system. The solution was intensively tested using historical real production data and it is now being installed in the real production line. The preliminary results confirm the initial expectations in terms of improvement of process performance and product quality.

Keywords: Multi-agent systems, Manufacturing control, Self-adaptation.

1 Introduction

A current trend in manufacturing domain is the development of adaptive production systems, facing the emergent requirements imposed by global markets demanding high quality customized products at reduced prices, to overcome existing process limitations and enable new manufacturing and processing methods. This challenge, part of the vision for the factory of the future, is also referred in the strategic research agenda made by the Manufuture European Technology Platform [1], which points out the need for enabling technologies, oriented to flexible and intelligent processes that contribute for the achievement of more modular, flexible, re-configurable and responsiveness manufacturing systems.

Multi-Agent Systems (MAS) [2] are a suitable approach to solve these challenges by providing an alternative way to engineer manufacturing control systems, based on the decentralization of the control functions over a set of distributed, autonomous and cooperative entities, the agents. They differ from the conventional centralized, rigid approaches due to their inherent capabilities to adapt to emergence without external intervention [3]. In spite of the promising perspective of using MAS solutions to address the challenge of achieving adaptive production systems, only few industrial applications were reported in the literature, e.g. the application in a factory plant of Daimler Chrysler [4] and some experiences from Rockwell Automation [5] and Schneider Electric [6] (see [7] and [8] for a deeply analysis). Several reasons were

identified for this weak adoption by industry, being probably the most important one the convincement of the benefits of using agents running in industry, showing the maturity, flexibility and robustness of the technology.

Under the scope of the GRACE (InteGration of pRocess and quAlity Control using multi-agEnt technology) project (www.grace-project.org), a collaborative MAS solution was developed to operate in an industrial production line, integrating process and quality control, at local and global levels. This approach is aligned with the described trend to build modular, intelligent and distributed control systems, to introduce adaptation facing unexpected deviations and failures, namely in terms of production conditions, product fluctuations and production/process deviations. An important aspect in this work is the achievement of product quality and production performance benefiting from MAS principles, even in a rigid production structure. In fact, MAS are usually useful when the production structure allows alternative ways to re-route the production but in this work the benefits are in terms of product quality and production performance by adapting the process and quality control parameters.

The objective of this paper is to describe the implementation of the GRACE multi-agent system for a production line producing washing machines, and the posterior deployment into real operation. An important contribution of this work is to demonstrate the effective applicability of multi-agent systems in real industrial scenarios, contributing for a wider adoption of this technology by industry.

This paper is organized as follows. Section 2 presents the industrial problem to be addressed and the basic principles of the GRACE multi-agent system architecture. Section 3 describes the implementation of the multi-agent system solution and Section 4 describes how the ontology, supporting the shared knowledge representation, was designed and integrated in the multi-agent system. Section 5 overviews the deployment of the control system into real operation and discusses some preliminary results. Finally, Section 6 rounds up the paper with the conclusions.

2 GRACE Multi-agent System

The GRACE efforts focus the development of an agent-based system that integrates process and quality control for a production line producing washing machines.

2.1 Description of the Problem

The problem addressed in this work considers a washing machine production line, owned by Whirlpool and located in Naples, Italy (Fig. 1 shows a simplified vision of the line). The production line is composed of several machines arranged in a sequential order, which are linked together by conveyors. Each station performs a single operation in the product being produced, which can be of different types: processing (e.g. bearing insertion or pulley screwing), quality control (e.g. visual inspection or vibration analysis) or manual (e.g. cable and electronics assembly). Along the line, the quality control stations run proper inspection programs, which results are compiled for posterior analysis. The product instances enter the line with a specific process plan that takes into consideration the materials variables (e.g. type of

the rear tub) and the operation parameters (e.g. thickness of welding process) according to the type of washing machine to be manufactured.

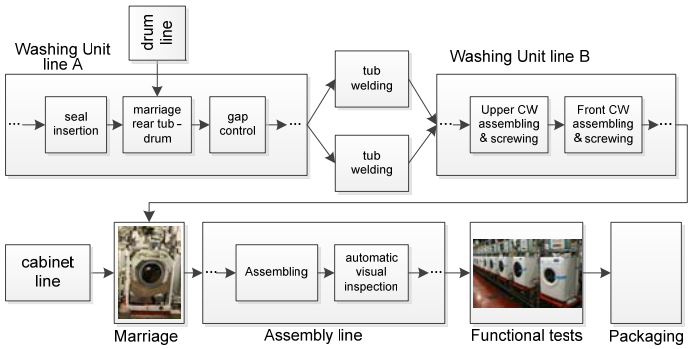

Fig. 1. Layout of the production line case study.

The objective to be achieved in this work is the integration of the process and quality control levels, creating feedback control loops that will allow the adaptation of production parameters. Note that the objective is not centred in the re-configurability of the production line (that is fixed and rigid), but instead to introduce adaptation to improve the product quality and the process performance, e.g. reducing the production time, correcting earlier the deviations or quality problems, skipping unnecessary tests along the line and customizing the final product.

An important assumption in this work is to maintain the low-level control, which already uses state-of-the-art industrial control based on Programmable Logic Controllers (PLCs) running IEC 61131-3 control programs, and introduce the multi-agent system solution at a higher control level to provide intelligence and adaptation.

2.2 GRACE Multi-agent System Architecture

The proposed multi-agent architecture involves a society of distributed, autonomous and cooperative agents representing the components of the production line, to operate at the factory-level. The agents act autonomously on behalf of these components, namely resources or products, introducing intelligence and adaptation, and cooperating to achieve the global production objectives. Several types of agents were identified according to the process to control and to their specialization [9]:

- *Product Type Agents* (PTA), representing the catalogue of products that can be produced in the production line (i.e., different washing machines models). They contain the product and process models.
- Product *Agents* (PA), managing the production of product instances launched in the production line (e.g., washing machines and drums).
- *Resource Agents* (RA), representing the physical equipment disposed along the production line and responsible to manage the execution of their process/quality control operations.
- *Interdependent Meta Agents* (IMA), representing supervision entities that implement global supervisory control and optimized planning.

In the definition of the RAs, several specializations were considered, namely Machine Agents (MA), associated to processing machines, such as robots or screwing stations, and Quality Control Agents (QCA), associated to quality control stations. In these components, two layers were considered:

- A low-level layer, representing the physical machine or testing station. This level may comprise PLCs running IEC 61131-3 control programs.
- A high-level layer, i.e. the agent itself, to perform intelligent management, control and adaptation functions.

The resulting system emerges from the cooperation among distributed, autonomous agents, coordinating their actions along the production line according to the product dependencies and the production plan, as illustrated in Fig. 2.

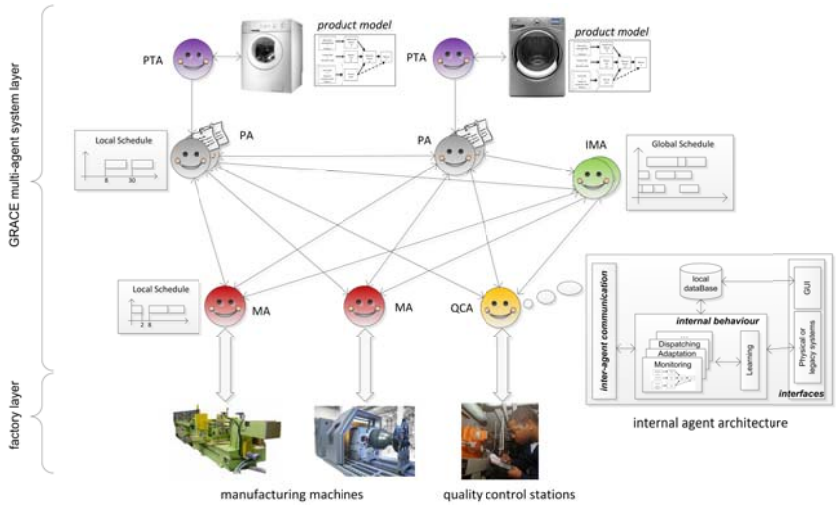

Fig. 2. Multi-agent System Architecture for the Production Line.

This interaction among agents is designed to enhance collaboration among intelligent agents providing a sound perspective to achieve:

- Dynamic and run-time adaptation to respond to condition changes, such as unplanned fluctuations of process/product parameters, at local and global level.
- Supervision and control schemes at factory level which maximize the efficiency of production and product quality, through feed-back control loops based on a continuous flow of information among agents.

For this purpose, individual agents exhibit a structure of behaviours according to their individual roles, enhanced by proper adaptation mechanisms, focusing local and global self-adaptation (see [10] for more details).

3 Implementation of the GRACE Multi-agent System

The development of multi-agent system solutions is strongly simplified if a proper agent development platform is used, taking advantage of the useful services it

provides, such as registry and management services. In this work, the Java Agent Development Framework (JADE) [11] was used, since it better responds to several requirements, namely being an open source platform and compliant with Foundation for Intelligent Physical Agents (FIPA) specifications, providing low programming effort and features to support the management of agent-based solutions and delivering an easy integration with other tools, namely the Java Expert System Shell (JESS).

Briefly, JADE is a Java based architecture that uses the Remote Method Invocation (RMI) to support the creation of distributed Java based applications. JADE provides a framework containing several agents that support the management of agent-based applications, namely the Agent Management System (AMS) and Remote Management Agent (RMA).

3.1 Implementation of Individual GRACE Agents

The GRACE multi-agent system comprises four types of agents, the PTA, PA, RA and IMA. Each one of these GRACE agent types is a simple Java class that extends the *Agent* class provided by the JADE framework, inheriting basic functionalities, such as registration services and capabilities to send/receive agent communication language (ACL) messages [11]. These functionalities were extended with features that represent the specific behaviour of the agent, as detailed in [9].

Each GRACE agent is developed using multi-threaded programming constructs, over the concept of the JADE's behaviour, allowing the execution of several actions in parallel. The execution cycle of a GRACE agent is briefly illustrated in Fig. 3.

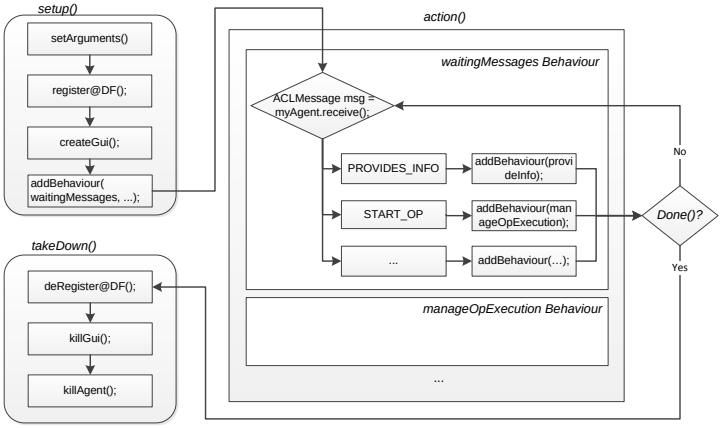

Fig. 3. Execution life-cycle of a GRACE's agent.

When the agent is created, the first method to be executed is the *setup()* method that should be fulfilled with the actions to be performed during the agent's start-up. In the GRACE's agents, the initialization procedure is responsible to register the agents' skills in the Directory Facilitator (DF), connect to the local database and create the GUI component. At the end of the *setup()* method, some behaviours are triggered, passing the control to the *action()* method that implements the code related to the

desired behaviours. The behaviours launched in the *setup()* method and those posteriorly invoked within these behaviours are provided in a software package in the form of Java classes.

The communication between distributed agents is asynchronous and done over Ethernet network using the TCP/IP protocol. The messages exchanged by the agents are encoded using the FIPA-ACL communication language, being their content formatted according to the FIPA-SL0 language. The meaning of the message content is standardized according to the GRACE ontology.

Since the behaviour of the agent is mainly driven by the messages received from other agents, a cyclic behaviour called *WaitingMessages* is launched in the *setup()* method. This behaviour is a Java class that is waiting for the arrival of messages, using the *block()* method to block the behaviour until a certain time elapses or a message arrives, and the *receive()* method to extract the incoming message. The arrival of a message triggers a set of actions related to decode the message and select the proper behaviours to be performed. As an example, if the received message has a PROVIDES_INFO identifier, a new behaviour called *ProvidesInfo()* is triggered. Note that after triggering the action related to the received message, the *WaitingMessages* behaviour remains continuously waiting for incoming messages.

At the end of the GRACE agent life-cycle, two methods are invoked: i) *done()* that tests if the behaviours included in the action method are finished, and ii) *takeDown()* that performs the last actions of the agent life-cycle, e.g. deregister from the DF.

3.2 Integration of Adaptation Functions

The functions providing the adaptation capabilities were embedded in the agents according to their roles. For this purpose, these functions were aggregated in software packages according to the agents where they will be hosted, being the functions that are shared by more than one agent type grouped in one separate package.

At the PA level, the *configuresOperationParameters()* functions adjust the operation parameters to be executed by the station, considering the data related to previous executed operations. In the case of testing operations, it can involve the selection of inspection algorithms and parameters; in case of processing operations, the selection of components, programs and parameters. Two examples are the customization of the testing plan to be performed by the functional tests station and the adjustment of the parameters to be written in the machine's controller board.

At the RA level, the *adjustParameters()* set of functions adjusts the processing or testing parameters, just before to trigger the execution of the operation, according to the local knowledge of the agent and the information gathered from previous quality control operations. As an example, QCA may use the feedback deriving by the past visual inspections to optimize its parameters and adapt itself to the environmental conditions (e.g. adapting the exposure time of the camera for image acquisition).

At the IMA level, global optimization procedures were deployed to elaborate optimization over identified meaningful correlations, supporting the adaptation of global policies for the system. As an example, the *trendAnalysis()* function performs a short analysis of the collected data to detect deviations or patterns in the data (e.g. a degradation process towards the defect) and generate warnings for other agents.

3.3 Integration with Legacy Systems

An important issue is the integration of the multi-agent system in a computational ecosystem, which is already running in the production line at different control levels. In this work, the integration of legacy systems will be exemplified with the integration of the quality control applications. As referred, in the GRACE multi-agent architecture each quality control station, which can be a physical equipment or an operator, has associated a QCA agent that is responsible to introduce intelligence and adaptation in the execution of the inspection operations.

In this work, several quality control stations are developed in LabView™ [12], imposing specific constraints to be integrated with the Java applications, i.e. the agents. In order to enable the communication, it was necessary to choose among different technological approaches. The best approach is to use a kind of service-oriented approach, where the services encapsulate the application's functionalities. These services can be provided (and announced in a service registry) by the Java application (i.e. the agent) or the LabView™ application (i.e. the quality control station). Since the integration is focused in two pre-defined applications (which doesn't require the need to discover the available services in the distributed system), only a basic layer is implemented, using TCP/IP sockets to interconnect the two applications. This approach allows the easy interconnection of the two applications running in different machines, achieving a very portable solution.

3.4 User Interfaces

Graphical User Interfaces (GUIs) are one way to provide a user interface supporting the management and monitoring of the system. Each type of agent provides a different GUI, since each one handles a particular set of information and allows different types of interactions with the users. In spite of providing different information, the GUIs follow a common template of menus, customized according to the agent's particularities.

The use of a Java based framework to develop the multi-agent system, offers the possibility of using *Swing*, a well-established toolkit to implement GUIs for desktop applications. Each type of agent in the GRACE system has its GUI implemented as an extension of the *javax.swing.JFrame* component. The GUI displays the local information stored in the agent's database, which feed the appropriate graphical components that are relevant to the users. As example, the IMA's GUI provides a global perspective of the entire production system, illustrating what is being produced (actual status) and what was produced (historical data).

4 Ontology for the Shared Knowledge Representation

Ontologies plays a crucial role in the GRACE multi-agent system to enable a common understanding among the agents when they are communicating, namely to understand the messages at the syntactic level (to extract the content correctly) and at the semantic level (to acquire the exchanged knowledge). For this purpose, a proper

ontology was designed to represent the knowledge associated to the domain of washing machine production lines, formalizing the concepts, the predicates (relation between the concepts), the terms (attributes of each concept), and the meaning of each term. The ontology schema (see [13] for more details) was edited and validated using the Protégé framework (protege.stanford.edu/), which is an ontology editor and knowledge-based framework.

The integration of the GRACE ontology in the GRACE multi-agent system requires the derivation of the ontological terms from the interfaces and the generation of a set of Java classes. For this purpose, two different approaches can be considered: a manual or an automatic one. In the first, the concepts are derived manually, which is a long, very difficult and time consuming task. The second approach uses tools that automatically translate the ontological concepts into the Java classes.

In this work, the *OntologyBeanGenerator* plug-in was used, which allows to automatically generate the Java classes from the Protégé tool, following the FIPA specifications. The main generated class represents the vocabulary and main ontological objects (i.e. concepts and predicates) defined in the ontology. The second group of generated Java classes specify the structure and semantics of each ontological object defined in the ontology. The methods defined in each individual class, e.g. the getters and setters, allow to handle the data related to the object.

At the end, some hand-made corrections in the exported Java classes are required due to syntax errors introduced by the plug-in during the automatic generation process. However, the use of this plug-in accelerates the integration process of the ontology, being a good approach to manipulate, integrate and reuse ontologies.

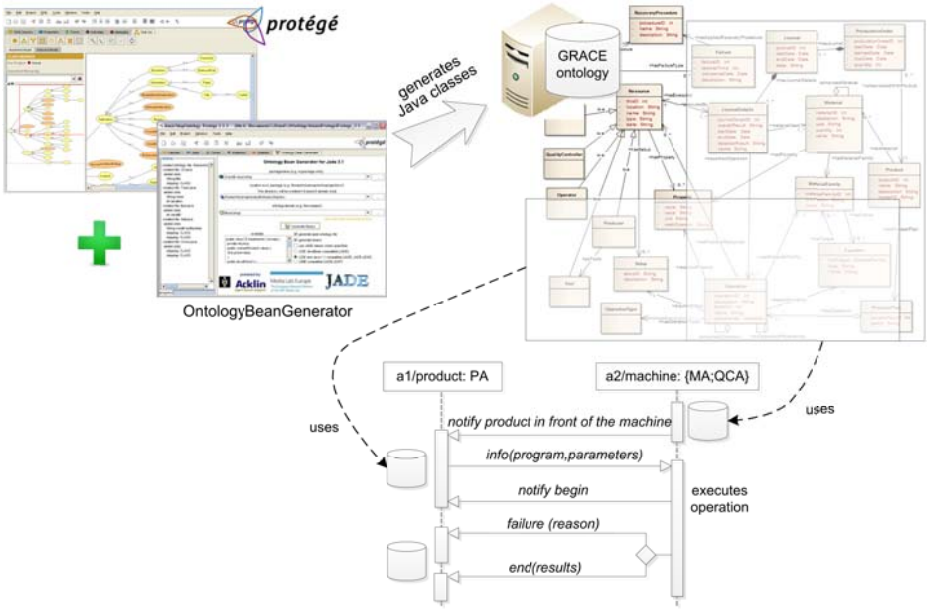

Fig. 4. Agents using ontologies to exchange knowledge.

At this stage, all classes needed for the ontological model are created, and ready to be used by the GRACE agents (after registering the ontology). As illustrated in Fig. 4, the agents use the same ontology (but different fragments of the ontology) to express the shared knowledge that is exchanged through the message exchange according to proper interaction patterns following the FIPA protocols.

A local database is used by each agent to store the data handled by the agent, according to the ontology used to represent the GRACE knowledge. To potentiate the solution portability, a SQLite database (http://www.sqlite.org/) was implemented using the SQLite JDBC Driver (https://bitbucket.org/xerial/sqlite-jdbc).

5 Deployment in the Production Line

The developed GRACE multi-agent system was initially debugged by using the unit test approach through the JUnit framework [14] to test each agent's function individually. After this testing phase, the system is ready to be deployed in the production line. This section discusses the agentification of the production line and the analysis of the results from the multi-agent system operation.

5.1 Agentification of the Production Line

The launching into operation of the multi-agent system requires two important actions: launching the JADE platform and launching instantiations of the agent classes developed for the specific agent-based solution, in this case for the Whirlpool's production line producing washing machines.

For this purpose, the structure of the GRACE agent classes described in the previous sections (i.e. PTA, PA, RA and IMA) is instantiated according to the production line needs. The set of agents launched to handle the washing machine production line are distributed by several computers, running in Windows 7 (64-bit) operating system with an Intel(R) Xeon (R) CPU W3565 processor @ 3.20 GHz. In terms of resources, 17 RA agents are launched, being the QCAs installed in the computers where the LabView™ applications for the associated quality control station are running. In another side, 9 PTA agents are launched corresponding to 9 different washing machine models, and 1 IMA is deployed to supervise the production line activities. The number of PAs running in the system is variable, depending on the number of products that are being produced in the production line, but in a stable production flow more than 200 PAs are simultaneously running.

The instances of the agent classes need the customization of their behaviours. For example, each one of the stations disposed along the line will be associated to instances of the RA agent, but each one has its particularities and it is necessary to reflect them in the generic structure of the RA agent. For this purpose, each agent has associated a XML file, describing the particularities and skills of the station. The following example illustrates the XML for the bearing insertion station.

```
<machine>
     <name> Bearing_insertion </name>
     <type> processing </type>
```

```
        <id> 5100 </id>
        <line> WU_line_A </line>
        <port> n.a. </port>
        <sqlserver> 159.154.64.164 </sqlserver>
        …
</machine>
```

These XML files are read when the agents initiate their life-cycle, within the *setup()* method, loading the agent profile with the customized parameters.

5.2 Analysis of Preliminary Results

The GRACE multi-agent system was deployed in the real production line. In the first phase, the operation of the multi-agent system was tested using real historic production data (avoiding the need to connect to the physical devices in the factory plant). This stage was crucial to identify and correct possible bugs before proceeding to the deployment into the on-line production of the factory plant. For this purpose, virtual resources were created to emulate the functioning of the physical devices using the production data stored in the production database. Since the database is managed by a Microsoft SQL Server application, the Java Database Connectivity (JDBC) API (Application Programming Interface) is used to establish the connection between the agents and the SQL-based database.

The intensive operation of the GRACE multi-agent system (running non-stop during slots of 1 week), showed, in a first instance, the correctness of the agent-based system, namely its stability and robustness, which allowed to conclude that the multi-agent system is sound and ready to be used in the real production line. Fig. 5 illustrates the multi-agent system running in practice, being possible to verify that the IMA agent had executed a trend analysis and generate a warning to the agent that may be responsible for the detected deviation.

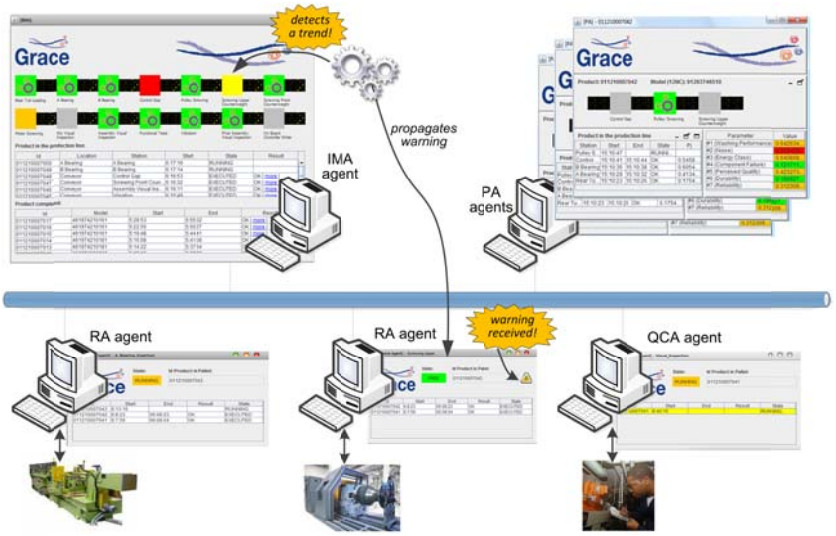

Fig. 5. GRACE multi-agent system working in practice.

In terms of quantitative results, besides the improvement of the product quality, it was noticed a reduction of the production time. Particularly, the implementation of adaptation mechanisms in the selection of the functional tests, by removing unnecessary tests, adjusting others or customizing the messages to the operators, allows an increase of:

- Productivity, since the inspection time is reduced (the average reduction of 1 minute over the default 6 minutes corresponds to an increase of approximately 20% in the production line productivity [15]).
- Product quality, since most effective quality control procedures are performed.

The proposed multi-agent system is now being installed in the real production line and the very preliminary results confirm the initial expectations in terms of improvement of process performance and product quality. In fact, after running the GRACE multi-agent system in practice, the average time to execute the functional tests was reduced, but more important was the customization of the messages displayed to the operator that allowed performing the inspection with better accuracy for particular situations. Additionally, the adaptation of operations parameters according to the production history and current environmental conditions, e.g. the customization of the on-board controller, allows a significant improvement of the product quality, since using the proposed multi-agent system, each washing machine is customized according to its production process historic.

The success of the multi-agent system deployment in a real industrial application is a crucial step to prove the applicability and merits of GRACE multi-agent system for production lines, and also the benefits of agent-based control approaches in industry.

6 Conclusions and Future Work

This paper describes the implementation of a multi-agent system in a production line producing washing machines, aiming to integrate the process and quality control and to provide adaptation capabilities to the system operation.

This agent-based system was implemented using the JADE framework, which provides an integrated environment for the development of such systems. The skeleton of the several GRACE agents were implemented, namely the behaviours' structure of each agent, the ontology schema for the knowledge representation, the interaction patterns supported by FIPA protocols, and the integration with legacy systems, particularly the LabView™ applications running in quality control stations. Several GUIs were also implemented to support an easy interaction with the users. Local and global adaptation functions were embedded in the several developed agents, aiming to provide adaptation and optimization based on the integration of the quality and process control.

The multi-agent system was intensively tested using historical real production data, aiming to test and correct the detected mistakes and bugs during the development process. The intensive tests showed the correctness of the system, namely in terms of adaptation, robustness and scalability. At the moment, the GRACE multi-agent system is being deployed in the factory plant and future work is devoted to the

complete commissioning of the agent-based solution in the factory plant and the continuously monitoring of the system operation to extract the achieved benefits.

Acknowledgments. This work has been financed (or partly financed) by the EU Commission, within the research contract GRACE coordinated by Univ. Politecnica delle Marche and having partners SINTEF, AEA srl, Instituto Politecnico de Bragança, Whirlpool Europe srl, Siemens AG.

References

1. European Commission: MANUFUTURE, Strategic Research Agenda: Assuring the Future of Manufacturing in Europe, Report of the High-level Group, Brussels (2006).
2. Ferber, J.: Multi-Agent Systems, An Introduction to Distributed Artificial Intelligence. Addison-Wesley (1999).
3. Wooldridge, M.: An Introduction to Multi-Agent Systems, John Wiley & Sons (2002).
4. Schild, K., Bussmann, S.: Self-Organization in Manufacturing Operations, Communications of the ACM, 50(12), 74--79 (2007).
5. Vrba, P., Tichý, P., Mařík, V., Hall, K., Staron, R., Maturana, F., Kadera, P.: Rockwell Automation's Holonic and Multi-agent Control Systems Compendium, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 41(1), 14--30 (2011).
6. Schoop, R., Neubert, R., Colombo, A W: A Multiagent-based Distributed Control Platform for Industrial Flexible Production Systems, Proceedings of the IEEE Int. Annual Conference on Industrial Electronics (IECON´2001), vol. 1, 279--284 (2001).
7. Pechoucek, M., Marik, V.: Industrial Deployment of Multi-agent Technologies: Review and Selected Case Studies, Autonomous Agents and Multi-agent Systems, 17(13), 397--431 (2008).
8. Leitão, P.: Agent-based Distributed Manufacturing Control: A State-of-the-art Survey, Engineering Applications of Artificial Intelligence, 22(7), 979--991 (2009).
9. Leitão, P., Rodrigues, N.: Multi-agent system for on-demand production integrating production and quality control, Holonic and Multi-Agent Systems for Manufacturing, V. Marik, P. Vrba, and P. Leitao (eds.), Lecture Notes in Computer Science, vol. 6867, 84--93, Springer Berlin / Heidelberg (2011).
10. Leitão, P., Rodrigues, N.: Modelling and Validating the Multi-agent System Behaviour for a Washing Machine Production Line, Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE´12), Hangzhou, China, 28-31 May, 1203--1208 (2012).
11. Bellifemine, F., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE, Wiley (2007).
12. Johnson, G.W.: LabVIEW Graphical Programming: Practical Applications in Instrumentation and Control, McGraw-Hill School Education Group, 2nd ed. (1997).
13. Leitão, P. Rodrigues, N. Turrin, C., Pagani, A., Petrali, P.: GRACE Ontology Integrating Process and Quality Control, Proceedings of the 38th Annual Conference of the IEEE Industrial Electronics Society (IECON'12), Montreal, Canada, 4328--4333, (2012).
14. Beck, K. and Gamma, E.: Test Infected: Programmers Love Writing Tests, Java Report, 3(7), 37--50 (1998).
15. Fernandes, A.: Simulação de Linha de Produção usando a Plataforma ARENA, MSc Thesis in Informatics Engineering, Polytechnic Institute of Bragança, Portugal (2012).