Enhancing Products by Embedding Agents: Adding an Agent to a Robot for Monitoring, Maintenance and Disaster Prevention

Leo van Moergestel[1(✉)], Erik Puik[1], Daniël Telgen[1], Hendrik Folmer[1], Matthijs Grünbauer[1], Robbert Proost[1], Hielke Veringa[1], and John-Jules Meyer[2]

[1] HU Utrecht University of Applied Sciences, Utrecht, The Netherlands
leo.vanmoergestel@hu.nl
http://www.hu.nl
[2] Utrecht University, Utrecht, The Netherlands

Abstract. Monitoring of computer networks, complex technical systems like aeroplanes is common practice. In this article the use of a monitoring agent in an arbitrary product is discussed. The product itself could be any product with sufficient hardware capabilities. The focus is on the product enhancement by adding an embedded agent. This so-called product agent can represent the product in the internet of things and it can also be a member of a multiagent system. In this way exchange of parts and subsystems is possible. The possibilities and advantages of this concept are discussed as well as a more elaborate example of the implementation in an experimental discovery robot.

Keywords: Agents · Monitoring agent · Product life cycle

1 Introduction

Agent technology for agile manufacturing was the starting point of this research. In this research the concept of a product agent was introduced. Every product to be made starts as a software entity or agent that is programmed to meet its goal: the production of a single product. To be able to reach its goal this agent knows what should be done to create the product. This entity is called a product agent and it guides the product along the production cells to be used for manufacturing and it will collect all kinds of important manufacturing data during the production process. When the product is finished, this agent has all the manufacturing details and this agent is still available for further use containing valuable information about the product. The next step in this approach is to investigate and study the roles of this product agent in the other phases of the life cycle of the product.

In this paper we study the implementation of a product agent that has not been used to create the product itself, but this agent is created for a specific phase in the life cycle of the product. First the use and roles of agents in all phases of

J. Filipe and A. Fred (Eds.): ICAART 2013, CCIS 449, pp. 115–130, 2014.
DOI: 10.1007/978-3-662-44440-5_7

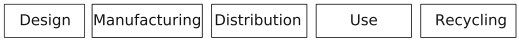

Fig. 1. Life cycle of a product.

the life cycle are discussed as well as how agents could be implemented for these different phases. Next we focus on the use phase. The case study for our product agent in the use phase is based on a discovery robot. This robot is also introduced and globally described. After this description of the product, the embedding of the product agent is discussed and some results of the implementation of the product agent in this complex system are shown.

2 Role of Agents in the Lifecycle of a Product

In Fig. 1 the life cycle of an arbitrary product is shown. After the design, the product is manufactured in the production phase, next the product is distributed. A very important phase is the use of the product and finally the product should be recycled. In all these phases, the product agent can play a role that will be globally described in the next sections

2.1 Design and Production

In our view the design of a product will be greatly influenced by the individual end-user requirements. This means that cost-effective small scale manufacturing will become more and more important. In [1,2] a manufacturing system based on a grid of cheap and versatile production units called equiplets is described. This grid is capable of agile multiparallel production. In this model every single product is guided through the production environment by the already introduced product agent. This agent is responsible for the manufacturing of the product as well as for collecting relevant production information of this product. This is normally a function of the so-called Manufacturing Execution System (MES) [3]. The result is that every product has its own production journal in contrast to batch production using a MES that generates one journal for a whole batch of products. In Fig. 2 the agent based manufacturing is depicted. In this figure the product agent is hopping from equiplet to equiplet to guide the product along the production machines or equiplets and monitor success or failure of the production steps [4]. To make a smooth transition from design to production possible, the product agent is designed as a co-design for the product. Because of the fact that the same equiplets that are used in the production phase are also used in the product design phase, a short time-to-market can be realised. Though this is all based on our own special production environment, we expect this approach to be useful in other production environments as well.

The concept of using agents for production is not new. Among others a multiagent-based production system is also developed by Jennings and Bussmann [5]. This system focuses on reliability and minimizing downtime in a production line. This approach is used in the production of cylinder-heads in car manufacturing. The roles of the agents in this production system differ from our approach.

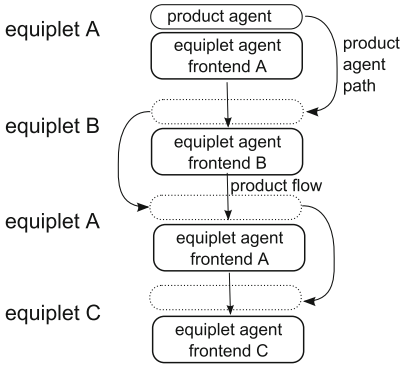

Fig. 2. Product agent and equiplet agents during production.

2.2 Distribution

Product agents can negotiate with logistic systems to reach their final destination. Logistic applications based on multi agents systems already exist [6]. Information of product handling and external conditions, like temperature, shocks etcetera can be measured by cheap wireless sensors and collected by the guidance agent during the transport or after arrival at the destination. The handling and external conditions during transport can be important during product use, especially for product quality, maintenance and repair.

2.3 Use

The role of the product agent during the use of the product could focus on several topics. The first question one should ask is: who will benefit from these agents, i.e. who are the stakeholders. In a win situation both the end-user as well as the manufacturer could benefit from the information. If a product is a potential hazard (in case of misuse) for the environment, the environment could also be a winner if the agent is capable of minimizing the effects of misuse or even prevent it.

Collecting Information. A product agent can log information about the use of the product as well as the use of the subsystems of the product. Testing the health of the product and its subsystems can also be done by the agent. These actions should be transparent for the end-user. If a product needs resources like fuel or electric power, the agent can advise about this. An agent can suggest a product to wait for operation until the cost of electric power is low i.e. during the night.

Maintenance and Repair. Based on the logging information about the product use and the use of the subsystems, an agent can suggest maintenance and repair or replacement of parts. Repairing a product is easier if information about

its construction is available. Also the use of a product or the information about transport circumstances during distribution can give a clue for repair. An agent can also identify a broken or malfunctioning part or subsystem. This could be achieved by continuous monitoring, monitoring at certain intervals or a power-on self test (POST).

An important aspect of complex modern products is the issue of updates or callbacks in case of a lately discovered manufacturing problem or flaw. In the worst situation, a product should be revised at a service center or the manufacturing site. Information about updates or callbacks can be send to the product agent that can alert the end-user in case it discovers that it fits the callback or update criteria. This is a better solution for a callback than globally advertising the problem and alert all users of a certain product when only a subgroup is involved.

Miscellaneous. Use of product agents could result in transparency of the status of a product after maintenance by a third party. The agent can report to the end-user what happened during repair so there is a possibility to check claimed repairs. Of course the agent should be isolated from the system during repair to prevent tampering with it. Recovery, tracking and tracing in case of theft or loss are also possible by using this technique. When the end-user wants to replace a certain device by a new one, the product agent can give advice about the properties the replacing device should have, based on what the product agent has learned during the use phase.

2.4 Recycling

Complex products will have a lot of working subsystems at the moment the end-user decides it has come to the end of its life cycle. This is normally the case when a certain part or subsystem is broken. The other remaining parts or subsystems of the product are still functional, because in a lot of complex products the mean times between failure (MTBF) of the subsystems are quite different. The product agent is aware of these subsystems or components and depending on the economical value and the remaining expected lifetime these components can be reused. This could be an important aspect of 'green manufacturing'. An important issue here is that designers should also take in account the phase of destruction or recycling. Disassembly and reuse of subsystems should be a feature of a product for this approach to be successful.

The product agent can reveal where rare or expensive material is situated in the product so this material can be recovered and recycled. This way the product agent can contribute to the concept of zero waste. *Zero waste is just what it sounds like - producing, consuming, and recycling products without throwing anything away* [7].

3 Product Types

This approach of having an agent for a product could be used on different kind of products, but one should investigate if the final product has intelligence and hardware to communicate with the agent. Some products have this by nature (computers, cell-phones); for other products (cars, machinery, domestic appliances) it should be a small investment. An important aspect will be the possibility to connect to certain subsystems for monitoring important events. If temperature is an important item for the product agent, connection to a temperature sensor or at least a place where this temperature data is available is a must. If this connection is not available, a temperature measurement system should be added to the agent.

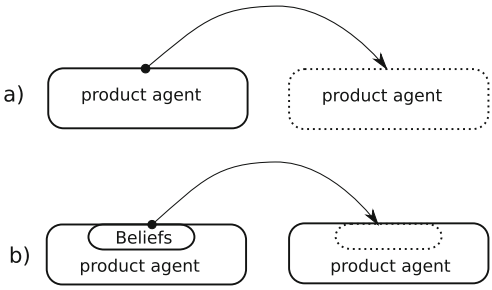

Fig. 3. Mobile agent versus moving data.

3.1 Where Do These Agents Reside?

A product agent should stay alive or at least the information the agent has collected and the knowledge the agent has learned should be available under all circumstances. To accomplish this, two solutions are available. The agent can be a mobile agent moving from platform x to platform y as depicted in Fig. 3a. The other solution requires moving data (beliefs of the agent) from one agent to a newly created agent as shown in Fig. 3b. In our case both agents should be product agents.

The second solution is much easier to implement because of the fact that only transport of data is required, while in the case of moving agents, the whole executable should be adapted to the new situation. Another advantage of the second approach is that a product agent can be added in any phase of the life cycle. This is also what has been done for this specific research. A product agent was added to a system in the use phase. The biggest challenge for implementing the approach of a product agent or guidance agent will be in the use phase. This is where the product is under control of the end-user and not as during the production under control of the manufacturer. In the latter case an agent-based infrastructure can be implemented for the production system or production line. The same is true for transport and even disassembly of the product. In case

of the use phase, the agents should reside in a system that is connected to the product, but should be available at the moment the product itself is broken. This is comparable to the case of the so-called black box in aeroplanes. There are several possibilities, depending on the type of product:

- The agent runs on its own separate hardware that is closely tied to the product;
- The agent runs on the hardware of the product but stores information on a special place on the product itself. This information can be recovered after breakdown;
- The agent runs on the hardware of the product but stores information on a remote system;
- The agent runs on a remote system that has a continuous connection;
- The agent runs remote on a system using a 'connect when necessary' approach.

The last two options require a stub or entry point for the remote agent to make contact with the product system. The connection with the environment could be established by wired or wireless sensors or sensor networks as well as computer subsystems in the product. Interaction with humans in the environment could be established by a messaging system or human computer interface (HCI).

4 Discovery Robot

This section gives details of the a discovery robot that was built by our research group. To investigate the implementation of the product agent during the use phase, the product agent was embedded in this complex technical system. To understand the details of the product agent implementation, it is important to have a global understanding of the construction and working of the discovery robot. In this section we present a short overview of the robot capabilities, the architecture, the software and an example of a result produced by the robot system.

4.1 Robot Capabilities

The robot that will be used as a platform for the product agent is capable of mapping a room with objects by using a laser scanner. The robot can move by itself using the map that has been created by the laser-scan. It is possible to direct the robot to a certain point in its map. The robot is also capable to avoid newly introduced obstacles and other moving objects. This robot is used as a system that will be enhanced by a product agent.

4.2 Architecture

Figure 4 shows a picture of the hardware of the robot. Two motors are connected to two wheels. Two swivelling wheels are added to keep the platform in balance. Attached to the platform is the laser scanner, printed circuit boards, a WiFi

transceiver, a camera and a set of ultrasonic sensors placed in a circle at the edges of the platform. These ultrasonic sensors are not yet used at this stage. A block diagram of the robot is depicted in Fig. 5. An important aspect is shown in this figure. An external computer is part of the system. This computer is used to do the heavy calculation to generate the map information, to display the map in real-time and to plan the path the robot has to follow. A wireless Ethernet connection (WiFi) connects the robot with this external system.

4.3 Software

The software for this robot is based on ROS. ROS is an acronym for Robot Operating System [8]. ROS is not really an operating system but it is middle-ware specially designed for robot control and it runs on Linux. In ROS a process is called a node. These nodes can communicate by a publish and subscribe mechanism. In ROS this communication mechanism is called a topic. Figure 6 shows the relation between two nodes and one topic.

A node that produces data can publish this in one or more topics. Other nodes interested in these data can subscribe to one or more topics. TCP/IP is used to actually carry out the communication. This platform has been chosen for the following reasons:

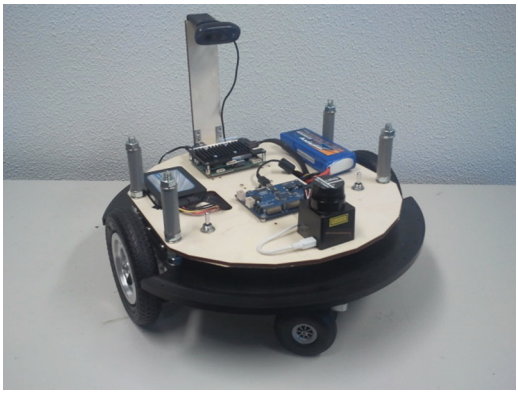

Fig. 4. Discovery robot.

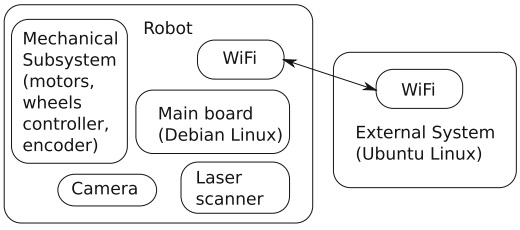

Fig. 5. Block diagram of the robot.

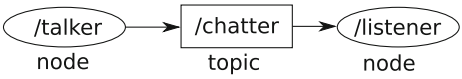

Fig. 6. Two nodes connected by a topic.

- open source, so easy to adapt, compliant with a lot of open source tools;
- wide support by an active community;
- huge amount of modules already available;
- nodes that are parts of ROS can live on several different platforms, assumed that a TCP/IP connection is available.

The mapping is done using SLAM. SLAM stands for Simultaneous Localisation And Mapping [9]. This module was already available in ROS and fitted well to the on board laser scanner.

4.4 Results

The results of a mapping in progress are displayed in Fig. 7. Here the robot mapped the corridors in a rather big building with three wings. The corridors are plotted as a light grey shape. The length of the longest corridor in this map is about 50 m. In this stage, the robot is not yet autonomous, but is controlled by a human operator that uses the external system and the on-board camera to guide the robot during the mapping. When the map is completed, the robot is capable to navigate autonomously to a given point in the map, even if new or moving obstacles are introduced in the mapped environment.

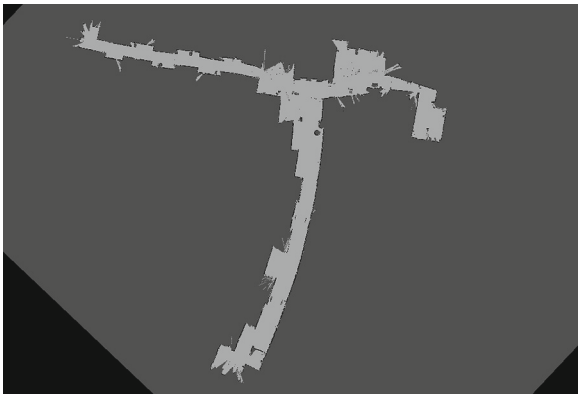

Fig. 7. 2D mapping of a building.

5 Embedded Product Agent

This section describes the product agent and also shows some results of its functioning.

5.1 Functional Requirements

The product agent that is added to the robot has the following requirements:

- monitoring status of the system or subsystems;
- monitoring health of the system or subsystems. The difference between health and status will be explained in the next subsection;
- react only in case of emergency;
- the robot should operate without the agent;
- making useful data available to the outside world, like construction details, materials used and its localisation in the robot.

5.2 Implementation

The first step in implementing this robot is to make an overview of information available in the system. Different types of information are considered:

1. status: is data available and of interest to the product agent and/or the end-user;
2. health: has to do with the condition of components that have mechanical parts or deteriorate during use;
3. alarm: an internal condition that could result in a troublesome situation or disaster;
4. additional information: this is the information that was conceived in earlier phases of the life-cycle.

Because the ROS environment is already available, it seems a natural choice to use this environment to implement the agent. The agent consists of ROS-nodes, ROS topics and some other subsystems. In Fig. 8 the internal modules of the agent are shown. All parts surrounded by an ellipse are ROS nodes. The rectangles represent topics. For human interfacing a small web server is included. This server is capable to serve static pages, containing technical data about the robot as well as dynamic pages containing data collected during use. Figure 8 shows the internal parts of the product agent and Fig. 9 shows the product agent

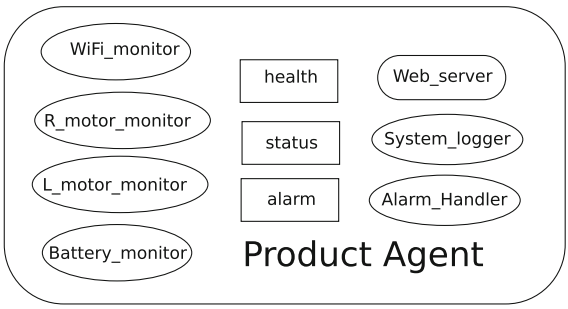

Fig. 8. Architecture of the product agent.

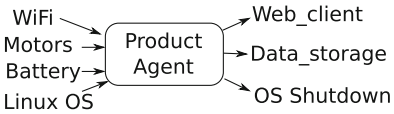

Fig. 9. The product agent and its environment.

in its the environment. The product agent interacts with its environment. The agent gets its information from the robot and its operating system. The agent will log this information and can also display information on a web browser (web client) by using the aforementioned webserver. A shutdown can be performed in case of a certain alarm condition.

5.3 Monitoring Status

The monitoring function is an important aspect of the product agent. In our prototype a selection of possibilities was made. A node will monitor the use of the motors and this will be available to subscribers of the health topic. The status topic is comparable to the health topic, but here information is made available that is not a result of the wear and tear of for example mechanical parts or of the de-charging of the battery, but is a result of measurements of interesting data like the strength of the WiFi signal. There is one topic that can trigger a node that will issue a system shut-down. This topic is called the alarm topic. Apart from these nodes, the agent can also retrieve its information directly from the Linux environment. Commands are available to get the CPU-load and memory usage. The pseudo filesystem `/proc` offers also a wealth of technical information that can be useful for the product agent. Examples of what can be retrieved from the product agent are plots displayed in the following two figures. Figure 10 shows a picture of the strength of the WiFi signal. The robot first moved away from the wifi access point and then returned towards the wifi access point again. The plotted data show a global decreasing and again an increasing trend but there are also strong fluctuations. These fluctuations are normal and due to all kinds of reflections and interference that occur in an indoor environment. In Fig. 11 the load of the processor is plotted. This curve is quite smooth and shows that the available processor power is adequate to operate the robot platform.

5.4 Monitoring Health

In the robot there are two candidates for monitoring the health. The motors and the battery. The battery should be monitored because of the fact that, like almost all rechargeable batteries, it can be re-charged and de-charged a finite amount of times and information of its remaining charge is valuable information to the end-user that operates the robot. In Fig. 12 the status of the battery is plotted during 90 min of operation of the robot. A steady decrease is shown as might be expected.

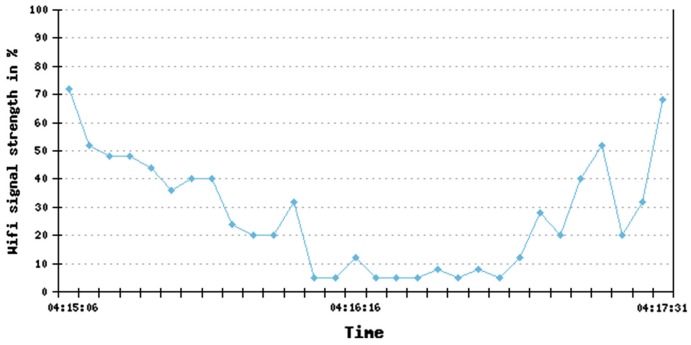

Fig. 10. Strength of the WiFi signal.

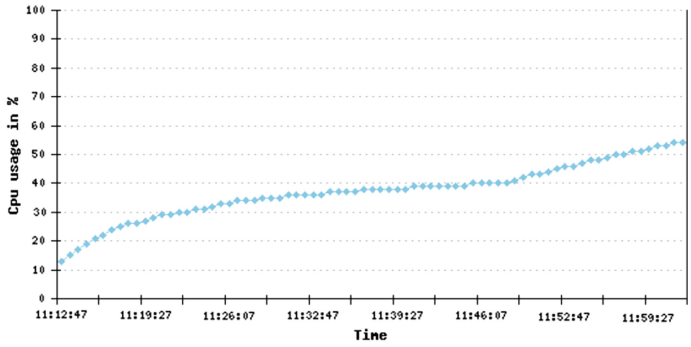

Fig. 11. CPU load.

5.5 Alarm Conditions

In this section an alarm condition will be described. The fact that the type of battery that is used in the robot should never be completely discharged gives rise to such an alarm condition. When the charge capacity drops below 10 % a system shut-down action should be triggered. By shutting down the system, the discharge of the battery will stop, thus preventing the loss of a rather expensive component. To implement this feature an Analog to Digital Converter (ADC) should be available to check the status of the battery.

5.6 Extra Functionality

The extra functionality that is offered by our implementation is embedded documentation and a mapping of materials and components that are of interest during the recycle phase. The information is offered using the same web-interface as was used in the monitor section previously discussed. The documentation is comparable to printed documentation that could be bundled with any device.

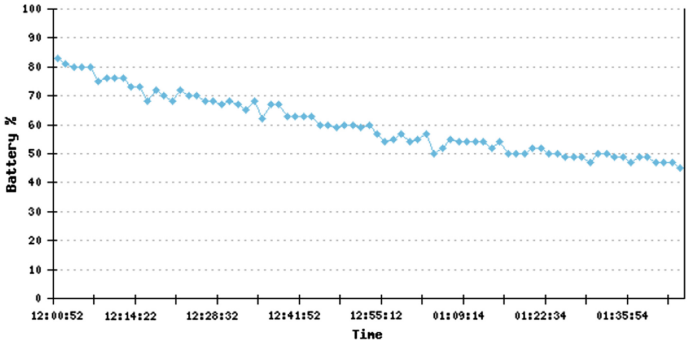

Fig. 12. Charge status of the battery.

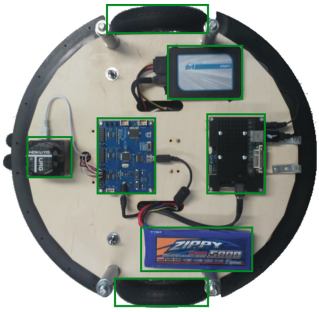

Fig. 13. Discovery robot subsystems.

This includes a user manual, a technical manual and a maintenance manual including a trouble shooting section.

In Fig. 13 a webpage is displayed showing the subsystems of the robot. This allows the user to select a subsystem to get more detailed information about that specific subsystem. Important information for the recycle phase is also offered using the web interface. Two different approaches are implemented. Using the web interface from Fig. 13, one could point at any part of the robot and receive information about the 'ingredients'. An example is given in Fig. 14. This is the result of selecting the wheels of the web page shown in Fig. 13. As a response the information about the material available in these parts is displayed. The materials as well as other relevant information is shown in Fig. 14.

Another approach is presented in Fig. 15. A list of interesting materials is presented and by clicking on an item, the subsystems containing this material are highlighted as shown in Fig. 15, where the subsystems containing gold are shown. These examples only show the interface designed for human users. The information is also available in a machine readable form using XML.

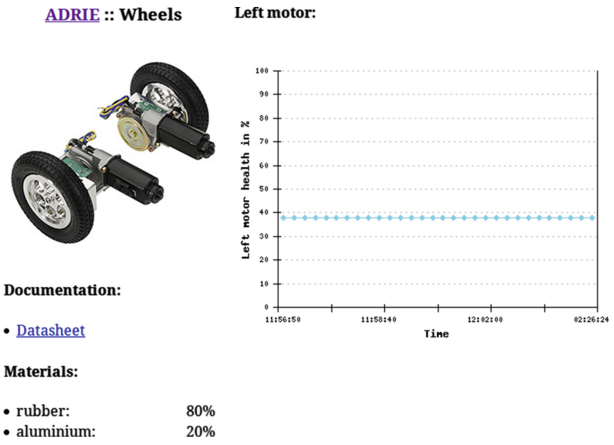

Fig. 14. Motor and wheel subsystem.

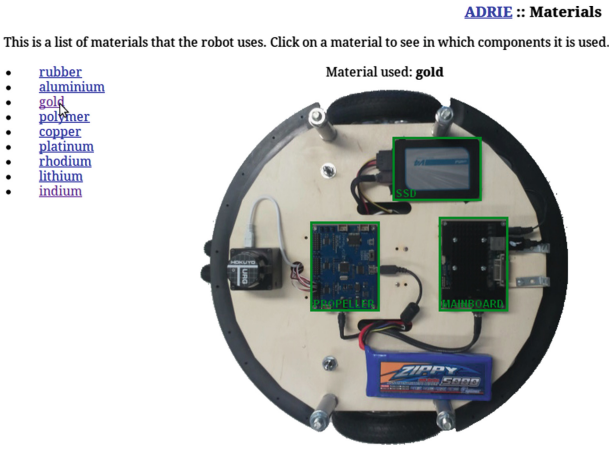

Fig. 15. Where is the gold hidden?

6 Related Work

The work on ROS played a very important role in this research. By using ROS we had a stable and well developed platform for our robot. The use of proven modules prevented reinventing solutions to already solved problems. The work on discovery robots is huge, [10,11] show some developments focussing on multiagent and swarm solutions. Agents for distribution, logistic applications and product manufacturing already exist [12]. In most situations agents represent human operators or negotiators. Jennings and Bussmann introduce the concept of a product agent, in their terms workpiece agents, during the production. These agents do not however perform individual product logging. The use of a product

is also studied by observing and/or interviewing end-users [13,14]. Some software applications do connect with their originating company to report the use by end-users.

Several proposals and implementations of including monitoring and documentation within the product itself are made and implemented. Burgess [15,16] describes Cfengine that uses agent technology in monitoring computer systems and ICT network infrastructure. In Cfengine, agents will monitor the status and health of software parts of a complex network infrastructure. These agents are developed and introduced in the use phase of this infrastructure and focus on the condition of the software subsystems. In our approach this monitoring function for hardware and software is the role of the product agent but that role has been played already by an agent during the manufacturing phase where valuable information that can be useful to the end-user has been collected. Actually this product agent in the use phase is not necessarily the same software entity that played the role of product agent during production, but the belief base of the product agent is kept intact and handed over to a new incarnation of the product agent.

In [17] an integrated diagnostic architecture for autonomous underwater vehicles is described. In this work the focus is on an intelligent system for system diagnostics. The architecture uses a variety of domain dependent diagnostic tools (rulebase, model-based methods) and domain independent tools (correlator, topology analyzer, watcher) to first detect and then diagnose the location of faults. This work could be used and combined with the model present in the current paper, because the artificial intelligence based techniques can applied in the product agent. Our work expands the idea of diagnosis and related data to the whole lifecycle of a product. By using this same agent again in the final phase of the life-cycle, component reuse and smart disassembly is a very important aspect when it comes to recycling of rare or expensive building material. The status of the quality of used sub-parts is available from and presented by the product agent.

In [18] the concept of the 'Internet of Things' is explained by the first user of the term 'Internet of Things'. The main idea of this concept is that the content of Internet is not only built and used by humans and therefore largely depending on humans, but the content will also be built by things connected to the Internet that are programmed to do so. The work presented in the current paper shows a possible technique to implement this concept of the 'Internet of Things'.

7 Discussion

In this paper the focus was on implementing a product agent in a complex product. This product was a discovery robot but could have been any other technical system. For every system the requirements for a product agent should be specified. However some global specifications are applicable for every system. The choices for monitoring subsystems made in this research were limited only to a few due to the fact that a proof of concept was the goal of this research.

The actions the agent can perform are in this case displaying and storing system status and system health status as well as system design and technical data. The agent is not influencing the robot itself however one alarm condition is implemented resulting in a system shut-down. It is not a difficult task to expand the capabilities of the agent. The robot itself will be further developed. For the product agent a wide variety of future enhancements is possible, especially when product agents of a certain type of product are united in a multiagent system:

- A model that builds a failure overview of subsystems. This way an accurate insight in the reliability of subsystems and components can be obtained. This model only works if a huge amount of product agent are participating.
- On behalf of the end-user, a product agent can report component failure and suggest or order replacement parts.
- An interesting model to implement the previous feature could be a marketplace in cyberspace where product agents can negotiate with other product agents about exchanging parts.

In all these enhancements special attention should be paid to security and the protection of privacy of the end-users of product agent enhanced systems. An important aspect is the fact that the agent should store its information at a safe place in case the robot hardware will fail. In our case this is the remote system where the agent has the possibility to store important data.

8 Conclusions

Product agents can play an important role in every part of the life cycle of a product. An important property of these agents is that they should have no direct impact on the product or system they are living in. However useful information should be collected and in case of disaster, these agents should keep a log of the events leading to the disaster.

Product agents can be a virtual digital equivalent of a product and this concept will be an enabling technology in implementing the internet of things.

The concept presented here is a natural evolution of the concept of using agents during production. However in case of products made by production technology not based on agent technology, a product agent can be added afterwards, as described in our case study. The information that could have been collected during design and production is added afterwards and will play a role in the recycle phase or maintenance during use phase.

The ROS platform proved to a very good platform to implement the product agent. This is because of the fact that the data-communication infrastructure between nodes is already implemented in a way that helps a lot in both the design and the implementation of the product agent.

References

1. Puik, E., van Moergestel, L.: Agile multi-parallel micro manufacturing using a grid of equiplets. In: Ratchev, S. (ed.) IPAS 2010. IFIP AICT, vol. 315, pp. 271–282. Springer, Heidelberg (2010)

2. van Moergestel, L., Meyer, J., Puik, E., Telgen, D.: Simulation of multiagent-based agile manufacturing. In: CMD 2010 Proceedings, pp. 23–27 (2010)
3. Kletti, J.: Manufacturing Execution System - MES. Springer, Heidelberg (2007)
4. van Moergestel, L., Meyer, J., Puik, E., Telgen, D.: Decentralized autonomous-agent-based infrastructure for agile multiparallel manufacturing. In: ISADS 2011 Proceedings, pp. 281–288 (2011)
5. Bussmann, S., Jennings, N., Wooldridge, M.: Multiagent Systems for Manufacturing Control. Springer, Heidelberg (2004)
6. Burmeister, B., Haddadi, A., Matylis, G.: Application of multi-agent systems in traffic and transportation. IEEE Proc. Softw. Eng. **144**(1), 51–60 (1997)
7. Gunther, M.: The end of garbage. Fortune **155**, 158–166 (2007)
8. Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Eheeler, R., Ng, A.: Ros: an open source robot operating system. In: Open-Source Software Workshop of the International Conference on Robotics and Automation (ICRA) (2009)
9. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping (slam): part i the essential algorithms. Robot. Autom. Mag. **13**(2), 99–110 (2006)
10. Wnuk, K., Fulkerson, B., Sudol, J.: A scalable architecture for multi agent vision based robot scavenging. In: American Association for Artificial Intelligence (2006)
11. Blazovics, L., Varga, C., Csorba, K., Fehér, M., Forstner, B., Charaf, H.: Vision based area discovery with swarm robots. In: Second Eastern European Regional Conference on the Engineering of Computer Based Systems, ecbs-eerc, pp. 149–150 (2011)
12. Paolucci, M., Sacile, R.: Agent-Based Manufacturing and Control Systems: New Agile Manufacturing Solutions for Achieving Peak Performance. CRC Press, Boca Raton (2005)
13. Nielsen, J., Levy, J.: Measuring usability: preference vs. performance. Commun. ACM **37**, 66–75 (1994)
14. Nielsen, J., Mack, R.: Usability Inspection Methods. Wiley, New York (1994)
15. Burgess, M.: Cfengine as a component of computer immune-systems. In: Proceedings of the Norwegian Informatics Conference (1998)
16. Burgess, M., Hagerud, H., Straumnes, S., Reitan, T.: Measuring system normality. ACM Trans. Comput. Syst. (TOCS) **20**(2), 125–160 (2002)
17. Hamilton, K., Lane, D., Brown, K., Taylor, J.: An integrated diagnostic architecture for autonomous underwater vehicles. J. Field Robot. **24**, 497–526 (2007)
18. Ashton, K.: That 'the internet of things' thing. RFID J. (2009)