Towards a Networked Robot Architecture for Distributed Task Execution and Knowledge Exchange

Moritz Tenorth, Koji Kamei, Satoru Satake, Takahiro Miyashita and Norihiro Hagita

Intelligent Robotics and Communications Laboratories
ATR, Kyoto, Japan
{tenorth,kamei,satoru,miyasita,hagita}@atr.jp

Abstract. In this paper, we discuss how networked robot architectures can facilitate the realization, deployment, management and adaptation of distributed robotic applications. Our aim is to modularize applications by factoring out environment-, task-, domain-, and robot-specific knowledge components and representing them explicitly in a formal knowledge base that is shared between the robots. Robot control decisions can then be formulated in terms of inference tasks that are evaluated based on this knowledge during task execution. The explicit and modular knowledge representation allows operators with different areas of expertise to adapt the respective parts of the knowledge independently. We realized this concept by integrating knowledge representation methods of the RoboEarth project with the distributed task execution capabilities of the Ubiquitous Network Robot Platform.

1 Introduction

More and more service robots are being developed to perform tasks like delivering items in hospitals [1], preparing simple meals [2], or interacting with customers in a shopping mall [3] or a convenience store [4, 5]. These robots are to be deployed in larger numbers in different environments, and all of them will eventually face the open-world challenge: Though all shops or all hospital rooms share common characteristics, each is slightly different, and each will bring up novel situations that have not been accounted for at design time.

Our goal is to develop a system to effectively manage large numbers of robots in different locations which perform similar tasks in similar, but different, environments. In particular, we focus on the knowledge aspect of the problem: How can the knowledge required for performing these tasks efficiently be represented? How can it be re-used in similar situations by similar robots? How can robots deal with a lack of knowledge that would be required for solving a novel problem? Ideally, such a system should support simple deployment, should have low operational costs, and should be adaptable to specific environments as well as

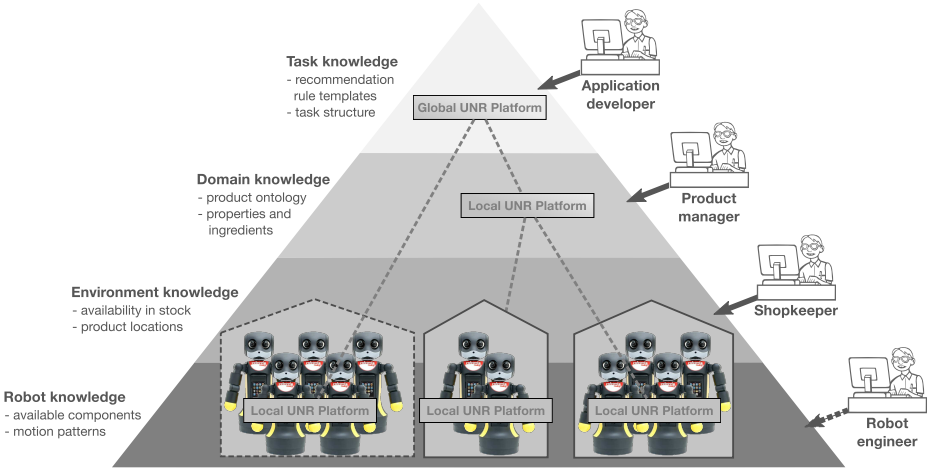

Fig. 1. The modular structure of the proposed system allows different groups of operators to focus on their area of expertise: Robot programmers implement the basic functionality, application developers combine these modules to useful services, while shop operators provide environment-specific information needed to execute the applications.

to different domains (e.g. different kinds of shops like a grocery shop or hardware store), and to changes in the task (e.g. special recommendation of seasonal products).

A decision to be made is the choice of the robot's level of autonomy. Completely autonomous robots could operate with minimal supervision and without a need for human operators. However, the robots will only be able to handle those situations that have been accounted for at development time. If they are asked about something they do not know, for example the ingredients of a product, they cannot answer this question if this piece of information has not been put into their knowledge base at manufacturing time. Updating the robots requires a robotics engineer to drive to their locations and perform the changes in cooperation with a domain expert that knows e.g. the products and their spatial arrangement in a supermarket. A tele-operated solution, on the other hand, would require a dedicated operator per robot, which would often be too expensive, and would require that operator to have both comprehensive domain knowledge and knowledge about the robot's environment configuration. We therefore aim at a combination of both: In normal circumstances, the robots operate autonomously, but can ask a human operator if they encounter problems.

In many cases, however, there is no single omniscient operator, but different experts are needed to contribute their specific kinds of knowledge: The spatial arrangement of products in the market can be provided by the shopkeeper. An area manager can contribute knowledge about the product types and their properties, e.g. their price or ingredients. These product properties will be the same for many shops, so this knowledge should be shared among them. Product recommendation patterns are often independent of the actual product types and

can therefore be implemented by an application developer for a large number of robots at once. Updates to the robot's low-level controllers, e.g. to program new motions, require a robotics engineer. Figure 1 visualizes the different levels of knowledge and the group of people that can provide it to the system.

To facilitate maintenance, it would be desirable to let these different stakeholders update their part of the robot's knowledge base independently: If the product arrangement changes, the shopkeeper should be able to update it without calling a robotics engineer. If new recommendation patterns are developed, they should be deployed without having to ask the shopkeeper for help. We would thus like to factorize these independent areas of knowledge, and would further like to realize a distributed system that allows operators to *remotely* provide their expertise.

In this paper, we present our approach to realizing such a distributed platform which combines distributed task execution and supervision methods with knowledge-sharing capabilities. To allow the adaptation of knowledge without changing the robot's control program, we use explicit knowledge representation mechanisms. We realize the system using the Ubiquitous Network Robot Platform (UNR-PF [6]), which provides distributed task execution and coordination facilities as well as abstraction from the robot's hardware and low-level capabilities, and the RoboEarth system [7], a web-based knowledge base which contributes techniques for representing and exchanging knowledge between robots.

The domain-specific, task-specific and environment-specific parts of the robot's knowledge are represented in a common knowledge base, but in a modular fashion, and separate from the program code of the robot's control program. This way, they can easily be adapted (e.g. when the arrangement of products changed), exchanged (e.g. to deploy an application in a different environment), and shared via the RoboEarth platform. The knowledge-sharing capabilities allow re-using knowledge in different setups (e.g. the same product ontology in many stores) and remotely deploying updates. The abstract component interface of the UNR-PF allows to specify generic task descriptions that can be re-used on heterogeneous robots with similar capabilities.

2 Related work

With the rise of the concept of "Cloud computing", which aims at off-loading components that have high demands in terms of computation or storage to remote servers which can provide the functionality in a more efficient way, the term "cloud robotics" was coined for applications of these concepts to robotics [8]. Since then, multiple systems have been proposed that address different aspects of the cloud robotics vision: Some of them focus on remote sensor data processing, implementing computationally expensive algorithms in the cloud [9]. The "PR2 Remote Lab" investigates robot teleoperation and remote control via the Internet using a Web browser [10]. The Ubiquitous Network Robot Platform (UNR-PF) deals with distributed task execution and supervision [6] on multiple robots and sensorized devices at different locations. The RoboEarth

project develops a web-based knowledge base through which robots can share information they have obtained [7]. In addition, there are attempts to make existing web- and cloud-based resources available to robots. While robot-specific applications will first need to be established and filled with content, many applications originally developed for humans do already provide information that can be useful for robots [11]. Examples are cloud-based object recognition systems like Google Goggles[1], on-line image and object model repositories like the 3D Warehouse [12], and product information from shopping websites as well as task instructions and cooking recipes [13].

With the current system, we develop a knowledge-based framework that combines the aspects of distributed task execution, cloud-based information exchange, and shared-autonomy tele-operation which were previously addressed only separately. Unlike other systems, we focus on the topics of providing knowledge to the robots and sharing it among them in order to give them the ability to act autonomously most of the time. With regard to our example scenario, we build upon prior work on establishing a ubiquitous-sensing infrastructure in a convenience store and using this infrastructure for guiding customers [4] and for recommending products [5].

3 Scenario

In this paper, we consider a convenience store as example, which is equipped with a ubiquitous sensing infrastructure and with robots that interact with customers. The robots can help customers find the products they are looking for and recommend alternatives if these are not available. This mock-up convenience store is equipped with laser scanners for tracking customers and with RFID tag readers for detecting if objects have been picked up [14]. As part of this scenario, we will explore how semantic representations can lead to greater flexibility in customer interaction, and how the proposed distributed infrastructure helps to create, deploy and maintain robot applications. Figure 2 shows the definition of a recommendation task and explains how the applications described in this section contribute to it.

Semantic representations for product recommendation: Using semantic information like an ontology of products and their properties, a robot can flexibly answer questions about these products. The hierarchical structure of the ontology provides the robot with information which products belong to a category like *Food* or *Stationery*. Based on the represented product properties, it can answer questions about ingredients the customer may be allergic against. By computing which products are close in the ontology, it can find semantically similar alternatives if a product is not available any more.

Spatial knowledge for guiding customers: A semantic map contains instances of the abstract product types and the locations where they can be found in the

[1] `http://youtu.be/FxXBUp-4800`

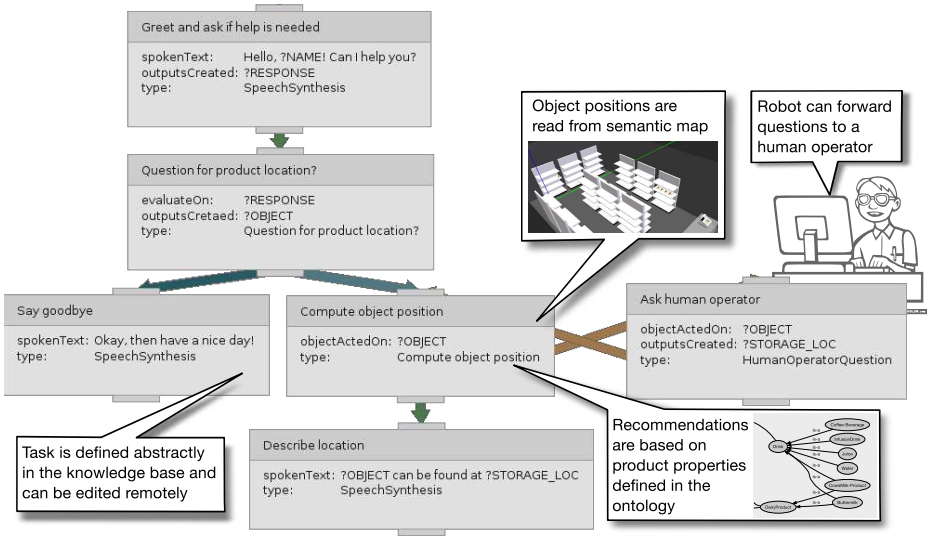

Fig. 2. Example task specification for recommending a product. The call-out boxes indicate where the proposed system contributes to the task execution.

environment. The spatial information enables the robot to point towards objects by computing the bearing from its own position. The map can be coupled with sensors in the environment, for example RFID tag readers, that update the information in the map and provide the robot with an up-to-date view of which products are still available. A graphical editor enables operators without robotics expertise to update the environment model if the shop layout has changed.

Human operator as fall-back knowledge source: While the robots operate autonomously most of the time, there may be situations in which they cannot answer a customer's question. In this case, they can forward the question to a human operator that can update the robot's knowledge, for example by adding or removing products or by changing their properties. The human and the robot share the same environment model that is distributed via the RoboEarth platform; when the human updates the map in RoboEarth, it is directly available to the robot as well. The required interaction with the human can be included into the task specification as a special action that is triggered if a question cannot be answered.

Remote adaptation and deployment of updated task specifications: In some cases, the task specification itself needs to be updated, for instance to change the robot's recommendation behavior or to fix flaws in the task definition. This adaptation can be done in a centralized fashion since the tasks the robots perform are themselves described in the knowledge base and shared via RoboEarth. The operator can adapt the specifications remotely using a graphical editor interface

and upload the updated version to the RoboEarth knowledge base, thereby making it available to all robots in the system.

4 System overview

The system presented in this paper combines the distributed task execution methods provided by the UNR-PF with the knowledge-sharing techniques of RoboEarth. Figure 3 gives an overview of its main components. The UNR-PF is used as communication middle-ware and abstraction layer between the hardware components and service application. A special execution engine can interpret task specification shared via RoboEarth and execute them on the platform, acting as a generic service application that can be parameterized with different task descriptions. The execution interacts with the system's knowledge base to resolve abstract specifications in the task descriptions to concrete parameters that are needed for executing the actions. A graphical interface for a human operator facilitates the inspection, modification and creation of knowledge.

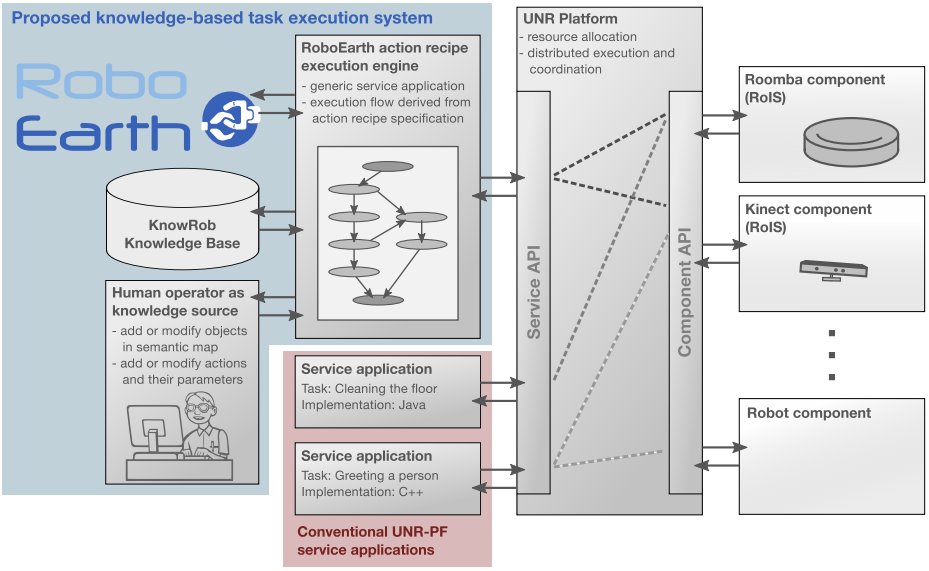

Fig. 3. Structure of the proposed system. Other than common task-specific service applications, the generic execution engine can be parameterized with knowledge-based task specifications. Control decisions are defined in terms of queries to the knowledge base that are answered based on the robot's background knowledge and belief state. A human operator can be asked to provide missing information.

4.1 The Ubiquitous Network Robot Platform (UNR-PF)

The UNR-PF acts as an interface layer between hardware- and software components on the one and service applications on the other side. The components

implement well-defined interfaces (e.g. for a *PersonIdentification* or a *Reaction*) and offer this functionality to the platform via the component API. Service applications compose useful applications out of these basic building blocks. Since all dependencies are defined in terms of the abstract interfaces, service applications are agnostic of how the functionality is provided or which components provide it. The *MovingPlatform* interface, for example, could be implemented using a wheeled or a legged platform, which both provide the functionality of moving the robot. Before executing a command, a service application requests components that have the required properties from the platform and, if suitable components are available, sends commands to these components via the service API or waits for events generated by them. The UNR-PF draws upon different standardized platforms and representations: The component interface of the UNR-PF is based on the RoIS standard [15]. Spatial information is encoded following the Robotic Localization Service (RLS) standard [16] and the CityGML language [17].

4.2 The RoboEarth system

RoboEarth aims at building a "World Wide Web for Robots", a web-based Wikipedia-like platform for sharing knowledge about actions, objects, and environments between robots. All pieces of information stored in the web-based are annotated with their requirements in terms of robot capabilities which are checked when downloading them. Using these requirement specifications, a robot can determine whether it will be able to use this information or if additional information is required. The client-side reasoning system is realized using the KnowRob knowledge processing system which is used for storing knowledge, performing reasoning, and offering a query interface to the execution engine.

In a demonstration by the RoboEarth project, it was shown that the techniques enable robots to autonomously download information required for performing simple mobile manipulation tasks like serving a drink to a human. In this example, both the task specification, an environment map, the models for recognizing the involved objects, and combined geometric and semantic object models required for interacting with articulated objects have been downloaded. The task has been performed by two heterogeneous mobile manipulation platforms in two different environments [18]. The RoboEarth project uses the Robot Operating System (ROS, [19]) as robot middle-ware and software distribution platform and the Cognitive Robot Abstract Machine (CRAM [20]) framework for its execution engine [21]. For the system proposed in this paper, we have created an executing engine that works in the context of the UNR-PF.

The formal language used for encoding the knowledge stored in the RoboEarth system is defined using the Web Ontology Language (OWL, [22]). The RoboEarth ontology defines the concepts and properties that are available for knowledge representation. It is derived from the KnowRob ontology [23] which, by itself, is partly derived from the OpenCyc ontology [24] that emerged as a quasi-standard in robot knowledge representation. By re-using these existing ontologies, we en-

sure compatibility with parallel developments in this research area and can incorporate findings of other projects more easily.

4.3 Knowledge-based task execution

The central component of the proposed system is an execution engine that can interpret action recipes defined in the RoboEarth language and execute them on the UNR-PF. Action recipes are composed of action classes, for example *LocalizingAPerson*, which are linked to the corresponding components in the UNR-PF. Before execution, the recipes are loaded and the system checks if all required components are available, then generates a state machine from the task description in the recipe, and starts the execution by calling the respective UNR-PF components as specified in the recipe.

Each action in the recipe is transformed into one state in the state machine that first requests and binds all required components using the UNR-PF, then calls the respective commands, and finally returns their results. Depending on the results, the execution transitions to the next regular state or to a specified error state. Perceptual components, e.g. for person localization or identification, are interfaced using "perception actions" that wait for events generated by the components. The recipe can specify when these actions are to return, for example once the first event message has been received, or the first n messages, or after a specified condition evaluates to *true* (e.g. once a specific person has been recognized). These conditions are described using OWL restrictions that are evaluated on the robot's knowledge base to check whether the condition is fulfilled.

To account for the heterogeneous nature of actions and their respective requirements in terms of information interchange with other actions, we developed a flexible information passing scheme using the local knowledge base. All information gained by executing an action is represented in the local knowledge base using the RoboEarth language, including the results of sensing actions, the outcome of manipulation actions, and information about which actions have been performed with which parameters. This approach closely links the knowledge-based task instructions to the robot's belief state and allows to reason about and integrate different sources of information like the instructions, the object ontology, and the semantic environment map. It also facilitates the exchange of information via the RoboEarth platform since all information in the system is already explicitly represented in this language.

4.4 Interactive knowledge editor interface

We combine the execution engine with a graphical user interface that allows human operators to extend and correct the robot's knowledge base, to add new objects to a map or new actions to a task, and to start the task execution on the remote robot. The UNR-PF thereby serves as distributed platform for the run-time coordination between the operator, the task-level controller, and the different (robot) components that perform the task. RoboEarth complements

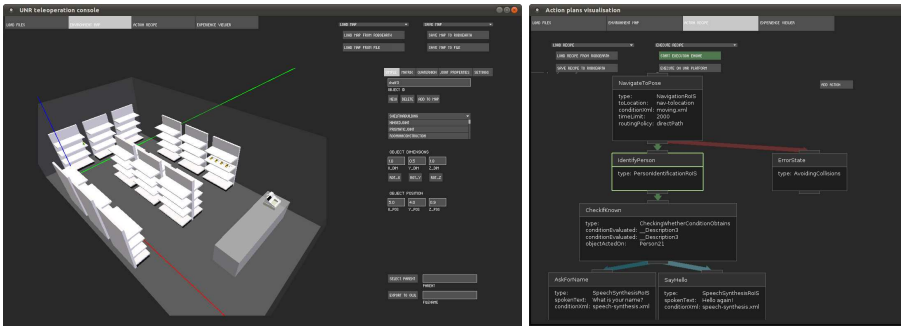

Fig. 4. Left: Semantic map visualization and editor. Right: Editor for defining the task structure and interactive execution interface.

this by providing the robot and the human operator with a shared knowledge base about the task to be executed and the environment the robot(s) operate(s) in.

The two main components of the user interface are an editor for semantic environment maps and one for robot task specifications. Both editors can either load specifications from the robot's local knowledge base or import them from RoboEarth. The map editor can visualize and edit environment maps that describe the spatial arrangement of objects around the robot (Figure 4 left). In these maps, each object is described as an instance of an object class at a specified 6D-pose in space. This instance inherits all properties specified for its object class, and can be further annotated with additional properties. Using the editor, objects can be added to, deleted from and moved around in the map. The task editor can be used to create, visualize and edit task descriptions by adding or removing actions, changing their properties, and specifying transitions between actions. The user can specify in detail how the task shall react in nominal and error cases using conditions for action transitions (indicated by the differently colored arrows in Figure 4 (right)). Green arrows indicate a transition in case of successful execution, dark red ones are in case of an error, light and dark blue transitions are chosen depending on the outcome of a decision node, etc. During task execution, the action editor doubles as supervision interface: The currently executed action is highlighted, action parameters can be inspected, and the task execution can be started and aborted.

4.5 Integration of Human Operator as Knowledge Source

We are planning to integrate the interactive editors with the task executive in such a way that a robot can call a human operator during task execution and ask for information that it found to be missing. The communication will be realized using the UNR-PF by adding the human operator interface as a component that accepts commands to acquire information. If the service application (which is supervising the task execution) notices a problem, it calls the command for asking the operator for help. The human can then download the relevant information

from the RoboEarth knowledge base, investigate the problem, update the information, upload it to RoboEarth again, and notify the service application that the information is available.

The ability to ask for help raises the questions *when to ask* and *what to ask for*. We consider two main cases in which interaction with an operator seems necessary: when a query for information unexpectedly gave no results (e.g. when a question asked by a customer cannot be answered) or when an action failed or produced inappropriate behavior. The former case can be handled in the context of an ongoing task by transitioning to an "interaction state" that sends a support query to the operator, blocks until an answer has been received, and returns to the same action to try again. This interaction scheme is indicated in the right part of Figure 2. The latter is a more exceptional case that often requires changing the task definition itself. It may be detected by errors thrown by the UNR-PF, by checking if the outcome of an action is as expected (e.g. a customer that does not move away on even if the robot considers a dialog to be complete), or if customers or the shopkeeper complain about the robot's behavior. In these cases, the task specification needs to be updated and the task needs to be restarted.

5 Discussion and Conclusions

In this paper, we discussed how a system for knowledge-enabled distributed task execution can be built on top of existing, common platforms in order to increase modularity, flexibility, and adaptability of robot applications. By consequently separating generic functionality from environment-, domain- or task-specific knowledge, we intend to achieve a high degree of reusability as well as improved adaptability to novel situations. Different kinds of knowledge are explicitly represented and can be edited independently by the respective domain experts. Control decisions are formulated as inference tasks that are evaluated on the robot's knowledge during execution.

Our approach involves abstraction along multiple dimensions: Abstraction from the robot hardware and the execution context is achieved by the components, services, and remote execution capabilities of the UNR-PF. Abstraction from the environment is obtained by encapsulating all environment-related knowledge in a semantic map that combines spatial and semantic information about objects. Abstraction from the executed tasks is realized by a generic execution engine that can perform arbitrary tasks defined as combinations of basic functionality building blocks. Abstraction from the application domain can be achieved by parameterizing generic functionality with an abstract domain ontology. We expect this abstraction to increase re-usability of components in different tasks and environments since individual parts of the knowledge base can be exchanged independently. The same task description for serving a drink, for example, should work in different hospital rooms, kitchens or offices as long as an appropriate environment model is available.

At the time of paper submission, the interactive editor interfaces as well as the interface to the UNR platform have been implemented. We are currently in the process of integrating the system with the infrastructure in the "Ubiquitous Market" experiment space and will present first results during the workshop. Once the infrastructure has been set up, we will investigate research issues such as the generation of informative questions and appropriate ways of presenting the state of the environment to a remote operator in order to convey the current situation.

6 Acknowledgments

This work was supported in part by the Ministry of Internal Affairs and Communications, Japan, and by the EU FP7 project *RoboEarth* (grant number 248942). The authors would like to thank Chandraprakash Sharma and Jonas Furrer for their help with using the UNR Platform.

References

1. Ozkil, A., Fan, Z., Dawids, S., Aanes, H., Kristensen, J., Christensen, K.: Service robots for hospitals: A case study of transportation tasks in a hospital. In: IEEE International Conference on Automation and Logistics (ICAL), IEEE (2009) 289–294
2. Beetz, M., Jain, D., Mösenlechner, L., Tenorth, M., Kunze, L., Blodow, N., Pangercic, D.: Cognition-enabled autonomous robot control for the realization of home chore task intelligence. Proceedings of the IEEE **100** (2012) 2454–2471
3. Shiomi, M., Kanda, T., Glas, D., Satake, S., Ishiguro, H., Hagita, N.: Field trial of networked social robots in a shopping mall. In: Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, IEEE (2009) 2846–2853
4. Kamei, K., Ikeda, T., Shiomi, M., Kidokoro, H., Utsumi, A., Shinozawa, K., Miyashita, T., Hagita, N.: Cooperative customer navigation between robots outside and inside a retail shop – an implementation on the ubiquitous market platform. Annals of Telecommunications **67** (2012) 329–340 10.1007/s12243-012-0310-2.
5. Kidokoro, H., Kamei, K., Shinozawa, K., Miyashita, T., Hagita, N.: You stopped by there? i recommend this: changing customer behaviors with robots. In: Proceedings of the 13th International Conference on Ubiquitous Computing, ACM (2011) 569–570
6. Kamei, K., Nishio, S., Hagita, N., Sato, M.: Cloud networked robotics. IEEE Network Magazine **26** (2012) 28 –34
7. Waibel, M., Beetz, M., D'Andrea, R., Janssen, R., Tenorth, M., Civera, J., Elfring, J., Gálvez-López, D., Häussermann, K., Montiel, J., Perzylo, A., Schießle, B., Zweigle, O., van de Molengraft, R.: RoboEarth - A World Wide Web for Robots. Robotics & Automation Magazine **18** (2011) 69–82
8. Guizzo, E.: Robots with their heads in the clouds. Spectrum, IEEE **48** (2011) 16–18
9. Arumugam, R., Enti, V., Bingbing, L., Xiaojun, W., Baskaran, K., Kong, F., Kumar, A., Meng, K., Kit, G.: DAvinCi: A cloud computing framework for service robots. In: IEEE International Conference on Robotics and Automation (ICRA), IEEE (2010) 3084–3089

10. Pitzer, B., Osentoski, S., Jay, G., Crick, C., Jenkins, O.: Pr2 remote lab: An environment for remote development and experimentation. In: IEEE International Conference on Robotics and Automation (ICRA), IEEE (2012) 3200–3205
11. Tenorth, M., Klank, U., Pangercic, D., Beetz, M.: Web-enabled Robots – Robots that Use the Web as an Information Resource. Robotics & Automation Magazine **18** (2011) 58–68
12. Klank, U., Zia, M.Z., Beetz, M.: 3D Model Selection from an Internet Database for Robotic Vision. In: International Conference on Robotics and Automation (ICRA). (2009) 2406–2411
13. Tenorth, M., Nyga, D., Beetz, M.: Understanding and Executing Instructions for Everyday Manipulation Tasks from the World Wide Web. In: IEEE International Conference on Robotics and Automation (ICRA), Anchorage, AK, USA (2010) 1486–1491
14. Kamei, K., Shinozawa, K., Ikeda, T., Utsumi, A., Miyashita, T., Hagita, N.: Recommendation from robots in a real-world retail shop. In: International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction. ICMI-MLMI '10, New York, NY, USA, ACM (2010) 19:1–19:8
15. Sato, M., Hori, T., Nishio, S., Chi, S.Y.: Standardizing framework for robotic services and functions. In: Proceedings of the ICRA Workshop on Robotics Modular Architecture Design and Standardization, IEEE (2011)
16. Nishio, S., Lee, J., Yu, W., Kim, Y., Sakamoto, T., Noda, I., Tsubouchi, T., Doi, M.: Robotic localization service standard for ubiquitous network robots. In: Robotics 2010 Current and Future Challenges. (2010) 381–400
17. OGC: City Geography Markup Language (CityGML) Encoding Standard. Open Geospatial Consortium (2012) http://www.opengis.net/spec/citygml/2.0.
18. Tenorth, M., Beetz, M.: Exchange of action-related information among autonomous robots. In: 12th International Conference on Intelligent Autonomous Systems. (2012)
19. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., Ng, A.: ROS: an open-source Robot Operating System. In: In IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan (2009)
20. Beetz, M., Mösenlechner, L., Tenorth, M.: CRAM – A Cognitive Robot Abstract Machine for Everyday Manipulation in Human Environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan (2010) 1012–1017
21. di Marco, D., Tenorth, M., Häussermann, K., Zweigle, O., Levi, P.: Roboearth action recipe execution. In: 12th International Conference on Intelligent Autonomous Systems. (2012)
22. W3C: OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax. World Wide Web Consortium (2009) http://www.w3.org/TR/2009/REC-owl2-syntax-20091027.
23. Tenorth, M., Beetz, M.: KnowRob – Knowledge Processing for Autonomous Personal Robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. (2009) 4261–4266
24. Lenat, D.: CYC: A large-scale investment in knowledge infrastructure. Communications of the ACM **38** (1995) 33–38