Journal of Applied Computing Research, 2(2):91-97
July-December 2012
 - doi: 10.4013/jacr.2012.22.05

ROS as a middleware to Internet of Things

Vinícius A. Hax, Nelson L. Duarte Filho,
Sílvia S. Da Costa Botelho, Odorico M. Mendizabal
Universidade Federal do Rio Grande. Av. Itália, km 8, Bairro Carreiros, 96203-900, Rio Grande, RS, Brazil
viniciushax@furg.br, dmtnldf@furg.br, silviacb@furg.br, odoricomendizabal@furg.br

Abstract. This paper discusses the feasibility of using the framework Robot Operating System (ROS) as a basis for the development of a middleware in the Internet of Things (IoT) context. The main concepts regarding IoT and ROS are presented and followed by the evaluation criteria. Finally, case studies considering the ROS features and some common assumptions of IoT environments are analyzed.

Key words: IoT, distributed systems, ROS, Internet of Things.

Introduction

This paper presents a feasibility study on the use of Robot Operating System (ROS) (Quigley *et al*., 2009) as a basis for the development of an infrastructure to Internet of Things (IoT) environments. Because of the growing interest in IoT, providing the right infrastructure and development tools makes the development of innovative solutions easier. As IoT is a recent researching topic, fundamental development tools for IoT solutions are not established yet. Thus, the adoption of ROS platform on development of IoT infrastructures becomes very attractive. Advantages of ROS include a high abstraction level to access external hardware, such as sensors and actuators. Because of the modular implementation of ROS, additional functionalities can be easily incorporated to the platform. In this work, evaluation criteria for IoT applications are analyzed as well as the feasibility to use ROS on development of IoT applications.

The present paper extends the results presented in (Hax *et al*., 2012). With respect to the original version, we detailed the challenges and some adaptations to adopt ROS as a middleware for IoT and expanded the related work section.

The remaining of this paper is organized as follows. The section "Internet of Things" introduces IoT and highlights some challenges inherent to this paradigm. ROS is presented in "Robot Operating System" section. Some application criteria related to the IoT context are defined in "ROS Evaluation" section and it is also made the analyses of the benefits of using ROS to cope with these criteria. Main challenges to implement an IoT middleware and some ROS adaptations are presented in the section "ROS Challenges and Adaptations". Related work is presented in next section. Conclusions about this work and future directions are presented in last section.

Internet of things

Recently, a new technological paradigm known as Internet of Things (IoT) has emerged. It is attracting increased attention from academic community as well as the industry (MacManus, 2010) and government agencies (Vermesan *et al*., 2009). Furthermore, IoT was considered one of the 6 civil technologies with higher potential impact in the USA (US National Intelligence Council, 2012).

The Internet of Things can be defined as a self-configurable dynamic global network infrastructure based in standards and protocols in which virtual and real objects have identities, attributes, and personality. Interaction between objects is done through smart interfaces

integrated in a network of information (Vermesan *et al.*, 2009, p. 6). Those objects can interact with other objects or with the environment through a variety of sensors and actuators. The technological impact caused by this computational paradigm comprises a range of applications in different sectors, such as automotive, aerospace industry, smart buildings, healthcare systems, logistic, manufacturing, etc.

In 2005, the Internet Engineering Task Force (IETF) created the IPv6 Over Low Power Wireless Personal network group with the goal of standardizing network protocols for low-power wireless personal area networks. Documents written by this group (Kushalnagar et al., 2007) characterize the equipment that compound those networks, as devices that implement protocol stack with small size data packets, around 81 bytes of data, support the IEEE standard for 16 or 64 bits extended to MAC addresses, uses low power (in the most of the cases power is supplied by batteries) and are low cost devices (usually they are associated to sensors and have low processing capacity and small memories). The network of these devices uses small bandwidth usage: rates of 250, 40 and 20 kbps, are organized in star and mesh topologies and is installed in groups of many devices. The location of devices is unpredictable, once they are installed as they are demanded.

Another way of classifying IoT elements is dividing nodes in two categories (Kosmatos *et al*., 2011). *Reduced functionalities devices*, which have processing and memory limitations and usually does not provide packet routing, or *complex functionalities devices*, which have higher capacity than the others. The latter often acts as routers and provides services to more simple devices or to elements outside the network like an external agent consulting provided services. The reduced functionalities devices are typically RFID devices (Kosmatos *et al*., 2011; Mattern and Floerkemeier, 2010).

The choice of IP (Internet Protocol) in IoT environments becomes natural because of its popularity and its usage established long ago (Mattern and Floerkemeier, 2010). Among its advantages, we would mention the wide adoption by several heterogeneous devices and the existence of specialized tools for monitoring, diagnostic, and configuration. The openness of this protocol is also an important benefit. Some problems existent in IPv4 were solved by IPv6, such as the support for a huge number of devices and the inclusion of self-configurable network mechanisms. These features suggest that the IP protocol is suitable for provision of communication to IoT.

Robot Operating System

ROS platform arose from the need to integrate common solutions employed in the robotic area. Different kinds of robots use different components of hardware, for instance actuators distributed at the engine and at the wheels, or even visual and infra-red sensors (Marder-Eppstein *et al*., 2010; Achtelik *et al*., 2011). The heterogeneity of the components makes the software development process difficult, which traditionally needs to attend each hardware. The knowledge of a variety of specific hardware profiles and their interfaces is rare, making the development of complex systems unfeasible. Thus, ROS was proposed to make the development in robotic area easier (Quigley *et al*., 2009).

One of the criterions in the ROS development was the adoption of peer-to-peer (P2P) communication instead of centralized communication. In scenarios where multiple nodes propagate information collected by sensors to processing nodes, the P2P model usually offers better scalability and performance than centralized communication models.

Although the massive communication in ROS is based in P2P model, the ROS naming service is centralized. A specialized node known as *master* provides a naming service. It is responsible for registering new services, inform which services are available and which peers are responsible for them. After the naming discovering stage, the remainder communication does not depend on the master node.

Figure 1 shows the first steps into communication between ROS nodes. A node called "hokuyo" offers a service "scan". First, this node registers the service "scan" in the naming server (master node) and informs the entry point to the service. Another node called "viewer" asks to the master node how to access the service "scan". As a response, the master node returns the service entry point. On the conclusion of this stage, the node "viewer" contacts directly "hokuyo" through TCP or UDP connection. Thereafter, the master node does not form part of the communication unless another service needs to be discovered.

ROS implementation aims to be compatible to several existent programming languages, given more flexibility to the system developers.

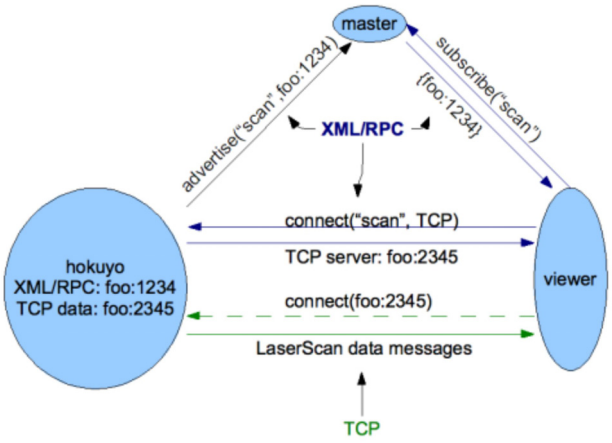

Figure 1. Communication in ROS.

In order to do this, it uses XML-RPC, a stateless, HTTP-based protocol wide available in a variety of programming languages. A common interface definition language is used to describe the data structures independent of the programming language being used.

ROS provides modular development, allowing the composition of small modules to perform more complex tasks. As a consequence, it eases the reuse of code and services from several projects. The ROS itself reuses code from other projects (OpenCV Project, 2012; Gerkey and Hedges, 2012).

ROS Evaluation

To evaluate ROS as a middleware to IoT we need multiple criteria that measure the level of difficulty to create an IoT application with ROS. If the difficulty is low and the criteria are served, ROS will be a good choice to use as development base to a middleware for IoT. IETF (Kim *et al.*, 2012) provides some criteria for IoT applications project. These criteria are used in order to evaluate multiple aspects of IoT applications.

- Installation: How will the devices be installed? Will the installation be manual or made by an automated process?
- Network size: Number of devices per group.
- Energy Supply: The energy of devices is supplied mainly by domestic/industrial standards or by batteries.
- Connection: The connection between nodes will be permanent, periodic or sporadic.
- Communication: Messages exchanged in the network will be through one or many gateways.
- Communication standard: broadcast messages or end-to-end communication.
- Security: What are the security requirements to application? Do the messages need to be encrypted? Who will have access to communication?
- Mobility: Mobile or static devices.
- Quality of Service: Includes the regular QoS challenges and add specific requirements from collective communication. The objective is to minimize communication and resource utilization. May change with the data communication patterns.

In order to clarify the criteria in Kim *et al.* (2012) it is showed the hospital case study, where it a temperature and humidity control for blood bags is needed. These variables must be constantly adjusted from the blood collection until their utilization. This control is done by sensors inside bags, transportation vehicles and rooms where the blood is stored. The installation is made manually following an existing plan. Some changes may occur, but they are infrequent and can be anticipated. The size is medium, and it depends of the hospital size. The node energy is supplied by battery on almost all devices and the connectivity must be permanent because all changes on sensors are important.

The communication is made through multiple nodes to reduce the fault occurrence through redundancies. The communication standard many to one, provides aggregation of information collected by the sensors. The security level is high: patient data is confidential (unauthorized person cannot access the data), the information needs to be available every time, and the data integrity must be guaranteed, *i.e.* data cannot be unduly changed. Once the sensors transmit control information to the actuators and alarms with required speeds, the Quality of Service must be taken into account.

Starting from previous defined criteria and using two hypothetical scenarios, the viability of ROS as IoT middleware will be evaluated.

Scenario 1: Health-care

In this scenario, there is a patient that leaves the hospital but needs constant observation. For cases where the patient lives alone at home, sensors are installed at the patient's

home in order to monitor his/her temperature and heart frequency. The collected information is transmitted to the hospital where the health of the patient can be monitored by specialists. If necessary an alarm echoes and an emergency team can move to the patient house.

The sensors installation procedure is manual and planned to collect data regarding the patient health. The network is small because one single house is monitored. The energy of the nodes is supplied by batteries to simplify the installation and maintenance of broken sensors. The connection is permanent to monitor the patient. A transient connection may imply life risks in some diseases. In case of battery replacement, ROS allows a node to stay offline for a while.

In this context the detection of problems like loss of energy or communication failure are crucial to minimize the impact of these problems. Maybe software solutions can be used to prevent this class of problems. The standard communication with the hospital is through a single connection but more routes of communication can be configured in a environment based on ROS.

The ROS software does not provide security solutions, mainly to guarantee integrity and authenticity of data. A malicious agent can listen the communication or fake his identity sending false data. A ROS-based middleware need to add security layers on API, like data cryptography and identity verification of the agents involved on communication.

Mobility is not a problem inside same network, but if a node changes his network the communication is lost with elements of previous network. An improvement on ROS is needed to permit communication between multiple networks. With this change lists of services and topics can be exchange between each ROS master. The synchronization operation does not exist on ROS because the default implementation uses only one master.

At last, ROS does not provide Quality of Service aspects. Each node is responsible to managing its own communication. New primitives can be implemented to provide QoS on ROS. In this scenario we may create a concentrator agent to compact similar messages and reduce network communication. Message from the same kind of sensor are potentially candidates for compression. To guarantee scalability the concentrator nodes may not be static, but new nodes may begin to act like a concentrator too.

Scenario 2: Agriculture monitor

In a farm, many nodes are manually installed. Each node has sensors do measure many variables: humidity, temperature, soil condition, intensity of sun. Some special nodes are installed to aggregate content from the sensors. The installation is planned by a specialist to try reducing the number of sensors. Network size is big to cover all terrain. The node alimentation is made through battery because there is no traditional energy alimentation around the plantation. ROS give support to full communication between the nodes for data exchange. With a non centralized communication multiple communication fluxes at same time without information bottleneck. ROS allow communication between multiple elements, so the sensors can send information to multiple content aggregators.

Security is not crucial on this application because the farm is distant from great centers. The information can be send from nodes to aggregators and on the inverse way if there is actuator on the farm like irrigation equipment. The mobility is not a problem because the nodes are static. With the great size of terrain, maybe the nodes are not in a single network and the ROS adaptation to multiple networks is required. Traditional Quality of Service is not essential because all sensor data are equally important, but data compression is wished to minimize network communication.

ROS challenges and adaptations

There are many challenges in implementing a middleware for Internet of Things using ROS. Some of them are:

Interoperability: The Internet of Things environment will consist of "billions or trillions of wireless identifiable things" (Sundmaeker *et al.*, 2010, p. 44). In this scenario is easy to see that nodes of IoT will come from multiple vendors and each device will made his own assumptions and have his own objectives. The IPv6 protocol has great chances to be chosen as default network protocol. ROS support IPv4 so a natural evolution is support IPv6 as network protocol. The robotics context where system components come from different vendors made interoperability a natural purpose on ROS implementation. The adaptations of ROS to IoT context will depend on the adoption of ROS by organizations interested in provide

solutions and devices to IoT. The possibilities of devices and demands from the future made this one of the biggest challenges for development of an IoT middleware.

Open standards: The adoption of open standards is a possible consequence of pursue of interoperability. Closed standards and royalties of patents leads do higher prices to devices. The expected number of nodes on IoT requires lower individual prices to large scale adoption. Because of this, the use of open standards and open source licenses reduces the price to IoT solutions. ROS is available as an Open Source Software without any cost per individual use. One of the main advantages of ROS usage as IoT middleware and not other middleware is the community already created around ROS. There are multiple groups working on ROS core improvement and building modules to provide functionalities above ROS.

Scalability: The great number of devices introduces difficulties to produce responses in a reasonable time. At every second thousands of requests can be generated. A special bottleneck is the naming service. The traditional Domain Name System (DNS) is not designed for this volume of requests. There are no studies for ROS performance with high number of nodes, but ROS naming service is centralized in the master node leading to scalibility problems. The availability of ROS source code permits a adaptation to alter ROS to a multi-master approach. With this modification ROS can support a greater number of devices.

Conflict resolution: In Vincent *et al*. (2010) an example of conflict resolution is given. Is supposed a smart house where two persons are trying to adjust the temperature according with his preferences. The proposed solution is use a mediator agent that adjusts a temperature with the average temperature. A ROS middleware must provide facilities to create and manage a mediator agent. These primitives must be embedded in the API.

Dynamic infrastructure: The high number of nodes in IoT makes manual device configuration prohibitive. The devices are supposed to be able to auto-discover services needed and provides interfaces to another devices use his data when desired. ROS is a platform service oriented. The architecture design favours a dynamic infrastructure because each agent can solicit to ROS masters the necessary services that reply with information to access services desired.

Quality of service: This is a feature that is treated as optional in traditional Internet but becomes necessary in IoT. The high volume of communication can prevent development of time critical applications without IoT. Furthermore intermediate agents can provide support to compaction and summarization operations. These features are not implemented in ROS leading to another possible improvement.

Privacy and security: These two aspects are closely related in IoT. The communication between nodes can include exchange of potential private data, like geolocalization, health information, financial data and so on. The dynamic disposition of devices implies that are almost impossible to configure each device to restrict privacy. The privacy level required must be provided by network and by the middleware. ROS does not provide security features that must be implemented in a middleware extension. It is possible and desired to implement features like encryption of messages, node authentication and data integrity.

An adaptation on ROS is possible using a module named Multi-master FKIE (Tiderko, 2013). Using this module it is possible to a ROS master synchronize with another, sharing topics and services provided by it. First a node called "master_discovery", connected on master A, sends a multicast message and if another "master_discovery" is connected on another master called B, it replies with the master B information. After, a node called "master_sync" uses the information from local "master_discovery" and makes a copy of the remote information on local master A. In tests already made, a node connected on master A has send and receive messages to another node connected on master B successfully.Another test was connect nodes on distinct networks. Two separate networks were configured and IPTables was used to route packets between networks. Using Multi-master FKIE on this scenario, two nodes on distinct networks can send and receive messages.

Related work

In Sundmaeker *et al*. (2010) a software stack is presented, ASPIRE, an abstraction to RFID devices and Hydra, a middleware to "networks of distributed wireless and wired devices". Vincent *et al*. (2010) provides a middleware with a different approach to some challenges like data availability and service discovery using statistic models. Caro *et al*. (2009) show another middleware for IoT with emphasis in security. A hybrid approach is

made using RFIDs and smart objects in a single SOA architecture that contains too a social network approach to objects (Kosmatos *et al*., 2011).

There is related work using ROS. Roalter *et al*. (2010) use ROS as base to create an intelligent environment. In Osentoski *et al*. (2011) a Javascript layer is developed above ROS to permit remote access to robotic applications.

Other more theoretical works are found in literature too. Paridel *et al*. (2010) critique solutions of kind publish and subscribe as unique solutions to IoT environment and propose hybrid solutions based in programming languages. Chaqfeh and Mohamed (2012) provide an overview about challenges and compare some recent implementations of middleware. Mattern and Floerkemeier (2010) show a path from our Internet of Computers toward an Internet of Things using existing technologies. Poikolainen (2012) focus on existing standards to IoT.

Conclusion

After evaluate ROS as base to developed a middleware for IoT we concluded that ROS support the installation and alimentation criteria. The direct communication between nodes permits greater network sizes. There is a bottleneck on naming resolution that can be solved with a automatic multi-master implementation. The subscription and publishing architecture allow applications where the node connectivity is limited or periodic. There are no primitives in ROS for multi-cast communication but this can be made on application level. There are not primitives to security and quality of service too. These two features are of great importance in IoT context but the middleware can implement this above ROS. Mobility is partial supported. It is necessary to develop inter-networks communication.

Security can be implemented extending the API and adding functionalities to offer authenticity, through exchange of private keys. Another solution is integrating the IP extension to security, IPSec (Arkko *et al*., 2004) inside ROS. With this alternative, cryptography can be supported on receiving and sending messages. Preliminary tests with existing ROS modules allowed synchronization between two masters even through two distinct networks. However, a necessary feature is implement algorithms to enable automatic multi-master synchronization, to allow multiple instances of masters, avoiding bottleneck of communication. This is a key feature because high number of nodes and messages expected in a IoT environment. The ROS features, like hardware abstraction, communication P2P, multiple programming languages binding, availability of source-code and modular code organization made ROS an attractive alternative as basis to an IoT middleware.

The next steps are implementing algorithms to enable automatic multi-master synchronization and launching new master nodes when necessary. After these implementations, ROS performance in multi-master and single master environment will be measured and compared. With these improvements we can implement IoT environment prototypes and evaluate the proposed solution mainly in scalability factor.

References

ACHTELIK, M.; WEISS, S.; SIEGWART, R. 2011. Onboard IMU and monocular vision based control for MAVs in unknown in-and outdoor environments. *In:* IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), Shangay, 2011. *Proceedings*… Shangay, p. 3056-3063. http://dx.doi.org/10.1109%2FICRA.2011.5980343

ARKKO, J.; DEVARAPALLI, V.; DUPONT, F. 2004. Using ipsec to protect mobile ipv6 signaling between mobile nodes and home agents. Available at: http://www.ietf.org/rfc/rfc3776.txt. Accessed on: March 15, 2013.

CARO, R.; GARRIDO, D.; PLAZA, P.; ROMAN, R.; SANZ, N.; SERRANO, J. 2009. SMEPP: A Secure Middleware For Embedded P2P. *In:* ICT MOBILE SUMMIT, 18, Santander, 2009. *Proceedings*… Santander.

CHAQFEH, M.; MOHAMED, N. 2012. Challenges in middleware solutions for the internet of things. *In*: INTERNATIONAL CONFERENCE ON COLLABORATION TECHNOLOGIES AND SYSTEMS (CTS), Denver, 2012. *Proceedings*… Denver, p. 21-26.

GERKEY, R.E.A. ; HEDGES. 2012. The Player Project. Available at : http://playerstage.sourceforge.net/. Accessed on: March 3, 2013.

HAX, V.A.; BOTELHO, S.; DUARTE FILHO, N.L.; MENDIZABAL. 2012. Estudo da viabilidade do ROS como plataforma para IoT. *In*: ESCOLA REGIONAL DE REDES DE COMPUTADORES, 10, Pelotas, 2012. *Proceedings*... Pelotas.

KIM, E.; KASPAR, D; VASSEUR, JP. 2012. Design and Application Spaces for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs) RFC 6568. Available at: http://tools.ietf.org/html/rfc6568. Accessed on: March 15, 2013.

KOSMATOS, E.; TSELIKAS, N.; BOUCOUVALES, A. 2011. Integrating rfids and smart objects into a unified internet of things architecture. *Advances in Internet of Things*, **1**(1):5-12. http://dx.doi.org/10.4236%2Fait.2011.11002

KUSHALNAGAR, N.; MONTENEGRO, G; SCHUMACHER, C. 2007. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals RFC 4919. Available at: http://datatracker.ietf.org/doc/rfc4919/. Accessed on: July 20, 2012.

MACMANUS, R. 2010. AT&T & Cisco Talk Up Internet of Things. Available at: http://www.readwriteweb.com/archives/verizon_att_cisco_internet_of_things.php. Accessed on: March 5, 2012.

MARDER-EPPSTEIN, E.; BERGER, E.; FOOTE, T.; GERKEY, B.; KONOLIGE, K. 2010. The office marathon: Robust navigation in an indoor office environment. *In*: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), Anchorage, 2010. *Proceedings*… Anchorage, p. 300-307. http://dx.doi.org/10.1109%2FROBOT.2010.5509725

MATTERN, F.; FLOERKEMEIER, C. 2010. From the internet of computers to the internet of things. *In:* K. SACHS; L. PETROV; P. GUERRERO, *From active data management to event-based systems and more*. Berlin, Springer, p. 242-259. http://dx.doi.org/10.1007%2F978-3-642-17226-7_15

OIKOLAINEN, J. 2012. *Internet of things–emergence of standards.* Jyväskylä, Central Finland. Bachelor's Thesis. University of Jyväskylä.

OPENCV PROJECT. 2012. Open Source Computer Vision Library. Available at: http://opencv.willowgarage.com/wiki/. Accessed on: March 15, 2013.

OSENTOSKI, S.; JAY, G.; CRICK, C.; PITZER, B.; DUHADWAY, C.; JENKINS, O. C. 2011. Robots as web services: Reproducible experimentation and application development using rosjs. *In*: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), 2011 Shangay, 2011. *Proceedings*… p. 6078-6083.

PARIDEL, K.; BAINOMUGISHA, E.; VANROMPAY, Y.; BERBERS, Y.; DE MEUTER, W. 2010. 2010. Middleware for the internet of things, design goals and challenges. *Electronic Communications of the EASST*, 28.

QUIGLEY, M.; CONLEY, K.; GERKEY, B.; FAUST, J.; FOOTE, T.; LEIBS, J.; WHEELER, R.; NG, A. 2009. Ros: an open-source robot operating system. *In*: ICRA WORKSHOP ON OPEN SOURCE SOFTWARE, Kobe, 2009. *Proceedings*… Kobe.

ROALTER, L.; KRANZ, M.; MOLLER, A. 2010. A middleware for intelligent environments and the internet of things. *In*: INTERNATIONAL CONFERENCE ON UBIQUITOUS INTELLIGENCE AND COMPUTING, 7, Xian, 2010. *Proceedings*… Xian, p. 267-281. http://dx.doi.org/10.1007%2F978-3-642-16355-5_23

SUNDMAEKER, H.; GUILLEMIN, P.; FRIESS, P.; WOELFFLÉ, S. 2010. *Vision and challenges for realising the internet of things*. Luxembourg, Publications Office of the European Union, 229 p.

TIDERKO, A. 2013. Multi-master FKIE Package. Available at: https://github.com/fkie/multimaster_fkie. Accessed on: March 15, 2013.

US NATIONAL INTELLIGENCE COUNCIL. 2012. Disruptive Civil Technologies - Six Technologies with Potential Impacts on US Interests Out to 2025. Available at: http://www.dni.gov/nic/confreports_disruptive_tech.html. Accessed on: March 7, 2012.

VERMESAN, O.; HARRISON, M.; VOGT, H.; KALABOUKAS, K.; TOMASELLA, M.; WOUTERS, K.; GUSMEROLI, S.; HALLER, S. 2009. *The Internet of Things - Strategic Research Roadmap*. Brussels, European Research Projects on the Internet of Things, CERP-IoT.

VINCENT, H.; ISSAMY, V.; GEORGANTAS, N.; FRANCESQUINI, E.; GOLDMAN, A.; KON, F. 2010. Choreos: scaling choreographies for the internet of the future. *In*: MIDDLEWARE'10 POSTERS AND DEMOS TRACK, 10, Bangalore, 2010. *Proceedings*... Bangalore, p. 10. http://dx.doi.org/10.1145%2F1930028.1930036

Submitted on February 16, 2013
Accepted on April 3, 2013