Studies in Systems, Decision and Control 36

Anis Koubaa
Elhadi Shakshuki *Editors*

Robots and Sensor Clouds

Springer

Studies in Systems, Decision and Control

Volume 36

Series editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland
e-mail: kacprzyk@ibspan.waw.pl

About this Series

The series "Studies in Systems, Decision and Control" (SSDC) covers both new developments and advances, as well as the state of the art, in the various areas of broadly perceived systems, decision making and control- quickly, up to date and with a high quality. The intent is to cover the theory, applications, and perspectives on the state of the art and future developments relevant to systems, decision making, control, complex processes and related areas, as embedded in the fields of engineering, computer science, physics, economics, social and life sciences, as well as the paradigms and methodologies behind them. The series contains monographs, textbooks, lecture notes and edited volumes in systems, decision making and control spanning the areas of Cyber-Physical Systems, Autonomous Systems, Sensor Networks, Control Systems, Energy Systems, Automotive Systems, Biological Systems, Vehicular Networking and Connected Vehicles, Aerospace Systems, Automation, Manufacturing, Smart Grids, Nonlinear Systems, Power Systems, Robotics, Social Systems, Economic Systems and other. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution and exposure which enable both a wide and rapid dissemination of research output.

More information about this series at http://www.springer.com/series/13304

Anis Koubaa · Elhadi Shakshuki
Editors

Robots and Sensor Clouds

Springer

Editors
Anis Koubaa
Prince Sultan University
Riyadh
Saudi Arabia

and

ISEP/CISTER Research Unit
Porto
Portugal

Elhadi Shakshuki
Jodrey School of Computer Science
Acadia University
Wolfville, NS
Canada

ISSN 2198-4182 ISSN 2198-4190 (electronic)
Studies in Systems, Decision and Control
ISBN 978-3-319-22167-0 ISBN 978-3-319-22168-7 (eBook)
DOI 10.1007/978-3-319-22168-7

Library of Congress Control Number: 2015947113

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Preface

In the evolving world of wireless technology and computing, there are many levels of technologies being introduced to the web. With the recent development of cloud computing, it is possible for users and machines to utilize and to share several services. The current power of robots, communication, storage, fast progress of wireless techniques, enhanced and different types of sensors, robots and sensor networks are able to take advantage of these services and provide influential solutions.

The book comprises four chapters that address some of the latest research in clouds robotics and sensor clouds.

The first part of the book includes two chapters on cloud robotics. The first chapter introduces a novel resource allocation framework for cloud robotics and proposes a Stackelberg game model and the corresponding task-oriented pricing mechanism for resource allocation. In the second chapter, the authors apply cloud computing for building a cloud-based 3D Point Cloud extractor for stereo images. Their objective is to have a dynamically scalable and applicable to near-real-time scenarios.

The second part of the book includes two chapters on integration of the cloud with the Internet of Things (IoT). The third chapter discusses the importance of the integration of cloud computing with the Internet of Things and presents an architecture for the Cloud of Things. In the fourth chapter, the authors reviewed the main proposed architectures for the Internet of Things, highlighting their adequacy with respect to IoT requirements.

Anis Koubaa
Elhadi Shakshuki

Contents

Part I Cloud Robotics

A Pricing Mechanism for Task Oriented Resource Allocation in Cloud Robotics 3
Lujia Wang, Ming Liu and Max Q.-H. Meng

Study of Communication Issues in Dynamically Scalable Cloud-Based Vision Systems for Mobile Robots 33
Javier Salmerón-García, Pablo Iñigo-Blasco, Fernando Díaz-del-Río and Daniel Cagigas-Muñiz

Part II Cloud for the IoT

Architecting the Internet of Things: State of the Art 55
Mohammed Riyadh Abdmeziem, Djamel Tandjaoui and Imed Romdhani

Cloud of Things: Integration of IoT with Cloud Computing 77
Mohammad Aazam, Eui-Nam Huh, Marc St-Hilaire, Chung-Horng Lung and Ioannis Lambadaris

Part I
Cloud Robotics

A Pricing Mechanism for Task Oriented Resource Allocation in Cloud Robotics

Lujia Wang, Ming Liu and Max Q.-H. Meng

Abstract Cloud robotics is currently driving interests in both academia and industry, especially for systems with limited computation capability. Resource allocation is the fundamental and dominant problem for resource sharing among agents in the cloud robotics system. This chapter introduces a novel resource allocation framework for cloud robotics and proposes a Stackelberg game model and the corresponding task oriented pricing mechanism for resource allocation. Simulation investigates the parameter selection and time cost of the proposed mechanism. Experimental results of co-localization task demonstrate that the proposed mechanism achieve an optimal performance in resource allocation.

Keywords Pricing algorithm · Resource allocation · Cloud robotics

1 Introduction

Nowadays, there is a growing need for service robots in human daily life, and the involved services are more complicated than ever before. For traditional robotic systems, robots have to carry adequate physical processing power and various sensors among other resources to facilitate the completion of various tasks such as visual navigation [38], range-finder-based navigation [41, 43], path planning [14], recog-

L. Wang (✉)
School of Electrical and Electronic Engineering, Nanyang Techonological University, Singapore, Singapore
e-mail: wanglj@ntu.edu.sg

M. Liu
Department of Mechanical and Biomedical Engineering, City University of Hong Kong, Hong Kong, Hong Kong
e-mail: mingliu@cityu.edu.hk

M.Q.-H. Meng
Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong, Hong Kong
e-mail: qhmeng@ee.cuhk.edu.hkk

A. Koubaa and E. Shakshuki (eds.), *Robots and Sensor Clouds*,
Studies in Systems, Decision and Control 36,
DOI 10.1007/978-3-319-22168-7_1

nition [40] and scene analysis [39, 42]. However, developing a practical robot that can cover many services would be extremely expensive and require a long time. It is thus reasonable to combine multiple robots that have limited capabilities, and access variety of information or services. This leads to the so-called paradigm "Cloud Robotics", which combines robot technology with ubiquitous network and cloud-computing infrastructures that link a lot of robots, sensors, portable devices and data centers. Therefore, robots can be remitted from hardware limitations while benefit from plenty of resources and computing capabilities in the cloud. However, resource competition is pervasive in practical applications for networked robotics today. It necessitates the allocation of limited bandwidth as an essential problem to be taken into account for the system design.

The authors of [25] first described a dual-level system architecture for cloud robotics, consisting of a machine-to-machine (M2M) level and a machine-to-cloud (M2C) level. On the M2M level, a team of robots communicates via wireless links such as Local Area Network (LAN) or Mobile Ad-hoc Networks (MANETs). On the M2C level, the infrastructure cloud provides a pool of shared sensor data, computation and storage resources, to be allocated among robotic agents. Considering the aforementioned dual-level architecture, this chapter presents a novel framework of a cloud robotic system. It consists of networked robots and a cloud-computing infrastructure. The latter connects the robots, sensors, portable devices and most importantly a centralized data-center. By adopting such a proxy-based model, all primary data can be retrieved from the cloud and managed by the proxy so that the requirements on hardware for each robot can be minimized. In addition, the proposed pricing resource allocation mechanism is task-oriented, which focuses on completing the necessary task or series of tasks in order to achieve optimized resource allocation.

Briefly speaking, this chapter deals with the resource allocation problem for cloud robotics by using a market-based mechanism. The following major contributions are addressed.

- A novel cloud robotic architecture is proposed based on an asynchronous data flow framework [70] for resource allocation managements among multiple robots. Especially, the architecture of cloud robotics is classified as an inter-cloud formed by robot-to-robot (R2R) and an infrastructure cloud enabled by robot-to-cloud (R2C).
- A Stackelberg game-based [45] resource management mechanism is proposed with consideration of the interaction among robot clients. The mechanism optimization is theoretically proved and implemented as functionalities of admission control, request ranking and resource distributing. Besides, a data buffer is set up on the access proxy for frequently requested data.
- A set of task-oriented Quality-of-service (QoS) criteria are proposed as the primary assessment metric of a co-localization scenario. The QoS's are defined regarding the fact that sophisticated collaborative robotic tasks are usually time sensitive.

The rest of the chapter is organized as follows. In Sect. 2, we discuss the related work in the area of resource allocation and cloud robotics. Section 3 presents our design of a typical cloud robotic system with a resource management middleware.

Afterwards, we define criteria of QoS at the end of the section. In order to solve the inherent conflicts of MSDR, we introduce the theoretical modeling and solution in Sect. 4. In Sect. 5, the parameter investigation and time cost of the proposed mechanism are presented. The experimental setup and results analysis are demonstrated in Sect. 6. At last, Sect. 7 presents the conclusion and future work.

2 Related Work

In this section, current works in the aspect of service-oriented architecture, cloud robotic systems, robotic task allocation and resource allocation mechanisms are reviewed and discussed.

2.1 Service-Oriented Architecture in Cloud Computing

The Service-oriented architecture (SOA) [6] is a widely used framework for cloud computing, where the cloud hosts and clients are synthesized under an elastic architecture. It represents computing in three parallel processes: service development, service publication and application composition using services that have been published. Cloud computing extends the scope of SOA by including the development of platform and infrastructure. So far, it is usually characterized by four paradigms: Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS) and Hardware as a Service (HaaS) [55]. Cloud computing can speed up many computationally intensive robotic and automation systems or applications, such as Simultaneous Localization and Mapping (SLAM) [54], robotic big data analysis, sample-based uncertainty sensing model analysis [32]. Thus, Robot as a Service (RaaS) [10] can also be added to such a scope. Most existing works using this framework are web service-based or database dependent. All major information technology companies and providers, including Google [9], IBM [13], Intel [27], Oracle [20], SAP [46] have adopted and supported this computing paradigm.

Because of the heterogeneous service and data that are discussed in [59], the cloud is usually addressed by a common middleware to get interoperability. Many researchers have worked on the resource management that is constrained in the field of e-commerce and enterprise computing systems, such as Eucalypus of Amazon Elastic Computing Cloud (EC2) [50], OpenNebula [51] and Nimbus [58]. Specifically, for RaaS, SOA can also be introduced. For example, Microsoft Robotics Developer Studio (MRDS) [15] was a vital product in applying SOA to embedded systems [63, 64]. Microsoft also released Visual Programming Language (VPL) in 2006 [57], which marked a milestone in SOA and in robotics. Many robot manufacturers have used VPL as their programming platform, including Coroware, iRobot, Kuka, LEGO NXT Mindstorm, Parallax, Robosoft, Robotics Connection, Whitebox Robotics, MOLMC IntoRobots, etc [15]. In addition, many of them use a web-based platform to configure the infrastructure that processing power, memory capacity, and communication bandwidth.

2.2 Current Cloud Robotic Systems

Although the concept of networked robots can date back to the 1990s [52], cloud robotics is now in better condition for both network and robot to provide an innovated outtake.

A large number of works took advantage of the big data in the cloud to simplify algorithms in robotics. ASORO laboratory in Singapore built a cloud computing infrastructure "DAvinCi" [3] to generate 3D models of environments, which allowed robots to perform SLAM. The Google autonomous driving project indexed maps and images that were collected and updated by satellite, street view, and crowdsourcing to facilitate accurate localization. The "cloud-based robot grasping" [31] used the Google Object Recognition Engine to recognize and grasp common household objects.

Some works provide designs of programming, middleware management framework enable robots sharing of data in the cloud. Google and Willow Garage initialized a software enables an Android phone to control robots based on platforms such as Lego Mindstroms, iRobot create and Vex Pro. MOLMC provides a sophisticated solution to an internet of things using MQTT protocol. The Gostainet [22] is an infrastructure of cloud robotic for executing the speech recognition on humanoid robot Nao [1]. This system is used to improve the interactions with children as part of a research project at a hospital in Italy. Authors of [26] presented a PaaS based cloud engine Rapyuta, which can allocate secure computing environments for robots.

Some works focus on accessing to databases of robotic sensing data such as maps and images. RoboEarth [67] is built for robots to autonomously share descriptions of environments and object models. Microsoft implemented a RaaS [10] platform, which included services for performing functionality, service broker for discovery and publishing, and applications for clients' direct access.

The aforementioned research took advantage of a wide range of online data resources, which is one of the most meaningful fields at the current stage. However, many robotic systems have a very strict assumption, such as *the resource in the cloud is unlimited*. In the matter of fact, most resources in cloud robotics systems are indeed limited [68]. For instance, network bandwidth for transmitting image data, CPU occupancy for parallel computation, as well as available number of hosts (proxy) are limited. Therefore, how to design a module that maximizes the utility of available resources on demand is a quite challenging problem, especially when multiple robots request the same kind of resource or service in an asynchronous manner.

2.3 Robotic Task Allocation Mechanisms

In robotics, tasks are usually regarded as resources to be allocated to robots in a collaborative system. Therefore, multi-robot task allocation (MRTA) problems are widely studied and can be characterized as the following types [21].

- *ST-SR-IA* is a simple optimal assignment problem and can be solved by both the centralized algorithm [8] and the distributed algorithm [37]. Centralized approaches usually can find the optimal faster than distributed approaches, but lead to a higher communication overhead.
- *ST-SR-TA* is an instance of problems when the task information and utility of robots can be predicted with some accuracy. This problem is building a time-extended schedule of tasks for each robot, with the goal of minimizing total weighted cost [65].
- *ST-MR-IA* is a kind of problems that involve tasks that require the combination of multiple robots. It is referred as coalition formation in the multi-agent community and is more difficult than the previously mentioned MRTA problems. The authors of [66] proposed a service-based approach with the principal idea that a robot can ask for services from other robots if the robot cannot execute a task by itself.
- *ST-MR-TA* is a class of problems includes both coalition formation and scheduling. It can be considered as an extension of the ST-MR-IA model with additional scheduling for future allocations. For example, the learning-based probabilistic algorithms [44] and the incremental task allocation algorithms [35] have been proposed.

In [48], the problem is formulated as a set of optimization problems with various objectives:

- *MinMax* (Minimize the maximal cost of nodes): it aims at timely critical missions by finding the shortest mission execution duration, which only concerns the worst node [35].
- *MinSum* (minimize the sum costs of all nodes): it is aimed at optimization of efficiency by minimizing of energy cost. However, it cannot guarantee an optimization on each node, and generally cause some nodes are optimized while some others are not [12].
- *MinAve* (minimize the average cost of all nodes): this objective measures the average time since a task appears in the system until it is completed. It is relevant to the problems where the completion is more important than aggregated global cost [60].

Other objectives are also proposed recently. For instance, minimize the processing time [74], maximize throughput, maximize the utility of the worst node, maximize the sum of individual costs and so on. However, one of the critical challenges in cloud robotic systems is how to optimally manage available resources such as physical robots, and sensor data while considering task constraints. The reasons are two-fold: multi-agent systems are typical complex and distributed, and agents are combined together as an overarching framework for integrated tasks [19]. Therefore, the combined resource allocation should be considered besides robotic task allocation, and current resource allocation mechanisms are reviewed in the next subsection.

2.4 Resource Allocation Mechanisms

In general, resource allocation problems are NP-hard [16, 53], which exist in computation systems, network communications, transportation systems and etc. Resource allocation solutions can be mainly classified into two approaches: one is the optimization-oriented approach which is usually a central planning, and the other is the economic approach which is usually a distributed scheduling.

For the centralized planning, researchers proposed different optimization techniques to minimize the rate of failure and execution time of tasks, or maximize the system utilization and throughput. Colony optimization proposed an algorithm to make an efficient resource assignment for computational jobs being processed, such as Ant colony algorithm [62] which can efficiently solve the resource constrained scheduling problem for mining supply chains. The genetic algorithm is used to solve the optimization problem based on a natural selection process that mimics biological evolution. For example, Rodriguez et al. proposed a particle swarm optimization algorithm for resource providing and scheduling on IaaS cloud to minimize the overall workflow execution cost [56]. Fuzzy logic is a problem-solving control system methodology that lends itself to implementation in various size of systems, such as Fuzzy Clustering Chaotic-based Differential Evolution (FCDE) solved the resource constrained project scheduling problem [11]. Market-based approaches to resource management [2, 28] are characterized by capturing complex interactions among autonomous agents and the system, which suit our problem most. However, most of them have assumptions that are not suitable for practical robotic tasks, such as the boundless communication and computation resources. The limited bandwidth resource should be considered in the real life scenario as presented in [61, 71, 73].

Autonomous negotiation among multiple robots has become a crucial problem in cloud robotic systems when clients query resources in parallel. The key issues of resource allocation for cloud robotics are the uncertain demands for resources such as that in big data mining and computing of robotics, and the large number of unreliable hosts which are physically distributed. Game theory has its advantages in solving this problem since it considers every agent and service provider's profits. For robotic systems, there are also several related works with different structures.

- Centralized approaches: the advantage is that the global knowledge can be used to manage all the available resources optimally while the disadvantage is that time and complexity cost are usually high. For example, the authors of [34] introduced a centralized iterated auction which included three objective functions and six bidding rules for a single task. It firstly demonstrated theoretical guarantees of auction-based methods for such a variety of bidding rules and team objectives. Higuera et al. [24] formulated the task distribution problem as a fair subdivision problem and provided a centralized algorithm to provoke the allocation mechanism for each robot.
- Distributed approaches: these methods are generally low cost since they only use local information, but they cannot achieve the theoretical global optimum. For example, the authors of [65] presented two algorithms for task distribution

problems where multiple tasks can be allocated to a single robot during the negotiation. Later, the authors of [66] presented a distributed market-based algorithm named S + T, which solved a task allocation problem for robot cooperation.

- Combinatorial approaches: allocated resources are a combination of different tasks, rather than a single task in complex systems. In [5, 36], the combinatorial auction was utilized to allocate multiple tasks in a multi-robot system, where robots bid on bundles of targets. They proposed different combinatorial bidding strategies and compared their performances, as well as with single-task auctions. Their computational results indicated that combinatorial auctions generally led to superior team-level performance than single-task auctions.

In general, the above works are based on theoretical analysis and simulation, few real-time robotic scenarios are reported in the real robotic cooperative system. This chapter aims at development of a practical mechanism for real applications.

3 System Architecture of Cloud Robotics

The proposed cloud robotic system is shown in Fig. 1, which includes an R2C network and an R2R network. In the R2C network, an Internet-based cloud infrastructure provides a data center sharing various kinds of sensor data [69]. In the R2R network, a team of robots communicates via wireless links such as LAN or MANETs [72].

3.1 Structure Design

The proposed framework of data retrieval is shown as Fig. 2. It is a host-based network framework which has three main entities involved for supporting Multi-sensor Data

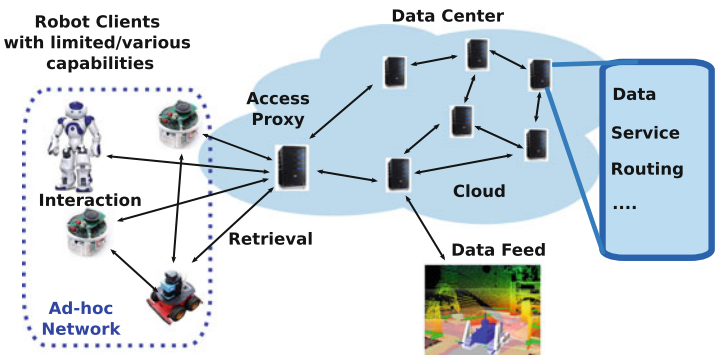


Fig. 1 A architecture of typical cloud robotic systems

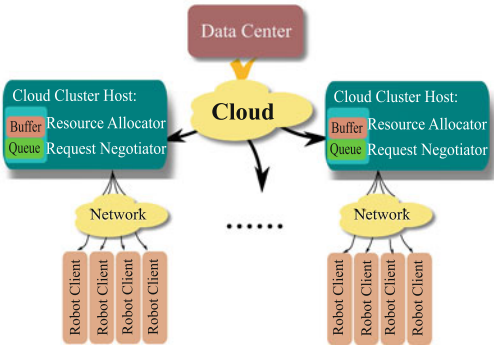


Fig. 2 The data retrieval framework of cloud robotic systems

Retrieval (MSDR) in cloud robotic system, namely, Data Center, Cloud Cluster Host, and Robot Client.

- Data Center (DC): it is a relative database based on PostgreSQL that stores various information, such as point-cloud, images that are established. All data are maintained and shared by any robot client on the network [68]. At the same time, the DC confronts unpredictable parallel requests from the robot clients.
- Cloud Cluster Host (CCH): it is a server that manages a large amount of data retrieval. The CCH consists of two major functionalities: Request Negotiator (RN) and Request Allocator (RA). The RN deals with the interfaces among the robot clients and hosts. It classifies the requests and provides clients with different prices in a pricing scheme. The RA is a function that accesses the cloud for admission control and buffer queue management. The RA distributes resources to robots in term of priority which is derived from the RN.
- Robot Client (RC): at the lowest level of the framework, it is a unit of different kinds of robots with various sensors. RC can be assigned to either an integrated task or separate tasks. Details are introduced in the next subsections.

3.2 Resource Management Framework

In Robotic Operating System (ROS) system, `rosservice` provides task requests and responses among nodes. Although the `actionlib` package provides tools to create servers that execute preemptive long-running goals, it does not support the queue management, especially asynchronous accesses for multiple tasks in the waiting list. Therefore, this is not sufficient for real-time tasks in multi-robot systems. In order to implement the SOA of cloud robotic system, we compared two parallel managements and communication software platforms, Hadoop MapReduce and `twisted` [47]. It is preferable to choose `twisted` as the software platform considering its multi-thread mechanism and compilation of the current database.

Benefits of `twisted` in Parallel Communication `Twisted` [47] is a framework for deploying asynchronous, event-driven and multi-thread supported network system using Python. The obvious advantage is the user-defined structure that can be flexibly applied for various managements. It is composed of the following three primary elements:

- `reactor`
  The `reactor` is the core of the event-driven programming construct in `twisted`. As shown in Fig. 3, it provides a basic interface to a number of services, including network communication, threading, and event dispatching. The application functions can be simply divided into modular and compact parts. It is also easily added specific data query and data retrieval modules for clients and hosts respectively.
- `protocol`
  The `protocol` defines specifications for transmitting and receiving behaviors. Functions of received data and sent data can be constructed following the predefined virtual function names. A `protocol` begins and ends its life with two predefined virtual functions: *connectionMade* and *connectionLost*, which are called whenever a connection is established or dropped, respectively.
- `factory`
  The `factory` is responsible for two tasks: creating new protocols and keeping global configurations and states. The tasks are completed by functions of `buildProtocol` and `management` as shown in Fig. 3. In addition to abstractions of low-level system calls, it also includes a large number of utility functions and classes, which facilitate the establishment of new types of servers. `Twisted` includes the support for popular network protocols, e.g. `SOCKETS`, `HTTP` and `SMTP` etc. It is flexible to define globally visible variables in the `factory`, such as the local data buffer. The host `factory` manages the connections to all the client `reactor` loops. At the same time, it is also in charge of updating the existing relation database, registering with new multiple sensor readings. Details of the framework structure and its assessment are outlined in the following.

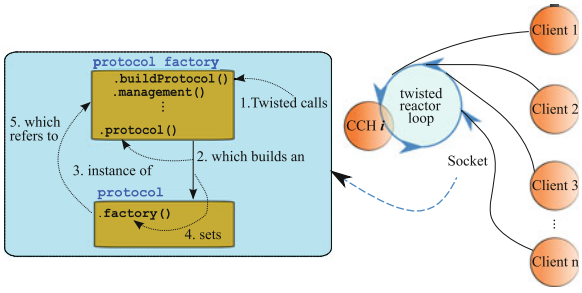


Fig. 3 Protocol creation process in `Twisted` reactor loop between CCH and clients

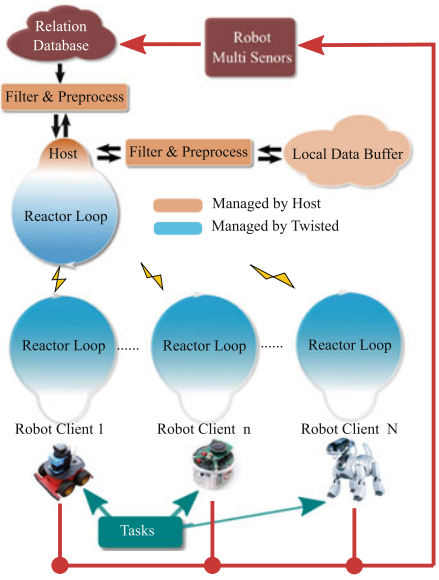


Fig. 4 Data flow of multi-data retrieval and communication in cloud robotic systems

3.3 Data Flow Management

The data flow of multi-data retrieval and communication in our cloud robotic system is shown in Fig. 4. The host and clients are built using the `twisted` framework. The designed program includes a main loop `reactor` and a callback system. This system automatically launches a new thread for each client that attempts to connect the network with a approved address and port. The specific functionalities are defined in the host and clients separately. The major functions in the process are introduced as follows:

- Database Query
  This function is launched and managed only by CCH which retrieves data from DC for RC. It utilizes a standard SQL [18] syntax to retrieve target information from a dynamically updated relation database. The database access may be one bottle-neck in the system, which can be alleviated by the management of the host. To this end, the following two sub-functions is designed to assist the retrieval, namely Filter and Pre-process and Buffer Management and Scheduler.
- Filter and Pre-process
  In the proposed data flow structure, the filter and pre-process blocks stand for general data pre-process. For example, data fusion, feature fusion and decision fusion [17, 23], are the major means to decrease the frequency of database access and reduce data noise.

- Buffer Management
  This function is launched and managed by CCH where a local data buffer is deployed for storage of frequently requested data as depicted in Fig. 4. Because activities of robots are usually regular, the same resource may be queried repetitively. Therefore, the buffered mechanism is built to help alleviate the database access bottle-neck to an extent.
- Scheduler
  Last but not least, the proposed scheduling scheme is launched by CCH that allocates resources for all clients' requests on top of asynchronous communication threads. The asynchronous management based on the `twisted` framework is implemented in CCH to manage all the connections among CCH and robot clients through `reactor` loop in parallel as shown in Fig. 4. Please note that `reactor` loop is a fundamental infrastructure of the `twisted`-based socket, which is used to automate asynchronous data transmission. In addition, the `reactor` loops are running on both CCH and heterogeneous robot clients. The optimization mechanism of resource allocation is modeled as a Stackelberg game. More mechanism details of resource allocation are introduced in the next section.

3.4 Quality of Service Criteria (QoS)

In common sense, bandwidth usage is one of the most important factors to define QoS since the response of most network-based applications is sensitive to it. In cloud robotic systems, instead of taking bandwidth usage as the only criterion, QoS definition can be extended to other aspects regarding the processing or storage capabilities of nodes. The following QoS's are selectively defined as primary criteria to assess the proposed framework.

Definition 3.1: Firm Real-Time Some infrequent deadlines can be missed, which may degrade the system's quality of service. The usefulness of a result is zero after its deadline. In this chapter, FRT is used to measure the performance of such a system. Because FRT is not directly about the delay but about effectivity. It relaxes the certain planned delay and settle with accomplishing the time delay restrictions most of the time.

Definition 3.2: Time of Response (ToR) ToR defines the period that from sending a request to receiving the corresponding response. It is formulated as

$$ToR = T_{\text{Data\_received}} - T_{\text{Request\_sent}}. \tag{1}$$

ToR, including request data transmission period, data matching period and response data transmission period, has been considered because sophisticated collaborating robotic tasks are usually timely sensitive and the long delay of robot's response can

make the task completion meaningless. For instance, cooperative semantic mapping or 3D mapping using several robots is time sensitive. Especially, the data transmission periods are the key factor impacting on the QoS performance.

Definition 3.3: Reliability of Response (RoR) RoR is defined as a success rate of issued data retrievals. Its value is given in percentage and calculated as

$$RoR = \frac{\#_{\text{Succeeded\_Requests}}}{\#_{\text{Total\_Requests}}}. \tag{2}$$

RoR is a key criterion for all services. Typically, in large scale systems, the perception results need to be shared and retrieved with acceptable reliability. The on-board computational capability of the robot clients is usually weak, which implies two inherent requirements as follows.

- The computation and analysis are generally to be off-board from the clients. Therefore, data retrieval from an existing database is inevitable.
- The accuracy of the retrieved data and reliability of the transmission are crucial. Especially, the reliability determines potentials to expand an existing system to a large scale.

QoS criteria advertise performance quality levels of service which are provided by service providers; on the other hand, clients use it to select an optimal candidate data/service, which could in part fulfills the request. Therefore, a well-defined set of QoS's could greatly help the assessment of the quality of a service framework.

4 A Pricing Resource Allocation Mechanism for MSDR

In order to share data and service, the paradigm of cloud robotics enables large numbers of robot clients working in parallel to retrieve multiple data in the cloud. In this section, a system model, MSDR problems, and solutions are proposed as follows.

4.1 System Model

We model the interaction between the cloud service provider and robot client (RC) as a Stackelberg game. The proxy CCH is regarded as the cloud service provider, who sets the price menu for different resource per bandwidth, and RCs respond to the price by presenting a certain amount of requests to the cloud. Suppose that a monopolistic CCH charges RC for usage of a network to maximize profit. Let $N := \{1, \cdots, n\}$ denote the set of RCs, and link of capacity nc accessed by n RCs. For RC i of willingness to pay type ω_i, x_i is usage of bandwidth resource and p_i be the price per unit bandwidth charged by the CCH, then its utility is

$$u_i(t_i, p_i) = \omega_i \cdot \log(1 + t_i) - t_i p_i, \tag{3}$$

where ω_i denotes the willingness to pay of RC i, t_i is the completion time of robot i for resource retrieval. The logarithmic function $\omega_i \log(1 + x_i)$ is verified to ensure a non-trivial and meaningful solution to the Stackelberg game. At the same time, the revenue of CCH is calculated as

$$L(\mathbf{t}, \mathbf{p}) = \sum_{i=1}^{n} t_i p_i, \tag{4}$$

where n is the number of robot clients that are allocated resources.

4.2 Problems and Solutions

The CCH maximizes its revenue by choosing the price p_i and the admitted RC number K for the limited bandwidth,

$$\mathbf{p}^* = \underset{\substack{\mathbf{t}\geq 0 \\ \mathbf{p}\geq 0}}{\operatorname{argmax}} L(\mathbf{t}, \mathbf{p}), \tag{5}$$

RC i determines x_i to maximize its utility and the problem is given by

$$t_i^* = \underset{t_i \geq 0}{\operatorname{argmax}} u_i(t_i, p_i), \tag{6}$$

Constraints are mainly focusing on the deadline of execution time T_0 and the admitted number n as follows

$$\sum_{i=1}^{n} t_i \leq T_0, \quad n = 0, \ldots, N. \tag{7}$$

Remark 1: For each robot client i, the utility function u_i is increasing, strictly concave, and twice continuously differentiable with respect to t_i.

Consider the optimization problem of maximization utility function $u_i : \mathbb{R}^n \rightarrow \mathbb{R}+$, defined in (6), where u_i is twice continuously differentiable in point t_i^*. The *first-order necessary condition* that t_i^* is a local optimum is

$$\frac{\partial u_i(t_i, p_i)}{\partial t_i}|_{t_i = t_i^*} = 0 \tag{8}$$

Therefore, we differentiate the utility function as

$$\begin{aligned}\frac{\partial u_i(t_i, p_i)}{\partial t_i} &= \frac{\partial(\omega_i \cdot \log(1+t_i) - t_i p_i)}{\partial t_i} \\ &= \frac{\omega_i}{1+t_i} - p_i \\ &= 0.\end{aligned} \tag{9}$$

The optimal price of resources queried by robot client i is derived as

$$p_i^* = \frac{\omega_i}{1+t_i^*}. \tag{10}$$

The revenue maximization problem (5) is a non-convex optimization problem with a non-convex objective function. Therefore, we have to convert it into an equivalent convex formulation to solve it. Then the solutions are proposed in the following two steps:

Step 1—Resource allocation: Assuming a fixed admitted RC number K, then plug (10) into (5), the above non-convex optimization problem be easily converted to convex as follows

$$t_i^* = \underset{\substack{\omega_i \geq 0 \\ t_i \geq 0}}{\operatorname{argmax}} \sum_{i=1}^{n} \frac{\omega_i t_i}{1+t_i}. \tag{11}$$

Because the revenue is strictly concave, and the constraint set is convex and compact, this optimization problem admits an optimal solution for the transmission time of robot clients and leads to a unique allocation. Considering the problem in (6) and the constraints in (7), we define the Lagrange function as

$$\Lambda(t_i, p_i, \lambda) = L(t_i, p_i) + \lambda\left(\sum_{i=1}^{n} t_i - T_0\right), \tag{12}$$

where λ is the Lagrange multiplier. By using the Lagrange multiplier technique, we can get the optimal transmission rate that denoted as

$$t_i^* = \sqrt{\frac{\omega_i}{\lambda}} - 1. \tag{13}$$

Step 2—Admission control: Please note the bandwidth constraints (7) must hold equality since the objective is strictly increasing function in t_i. Thus, by plugging the t_i^* into (7), we have

$$\sum_{i=1}^{n}\left(\sqrt{\frac{\omega_i}{\lambda}} - 1\right) = T_0. \tag{14}$$

Assuming that $\omega_1 \geq \omega_2 \geq \cdots \geq \omega_N$, then λ^* must satisfy the above condition (14). For a admitted RC number threshold value K_{th} satisfying

$$\frac{\omega_{K_{th}}}{\lambda^*} > 1 \quad and \quad \frac{\omega_{K_{th}+1}}{\lambda^*} \leq 1, \tag{15}$$

where K_{th} is used for the admission control, so only K_{th} or fewer robot clients can retrieve data. Moreover, we have $\lambda^* = \left(\frac{\sum_{i=1}^{K_{th}} \sqrt{\omega_i}}{T_0 + K_{th}}\right)^2$ derived from (14).

Remark 2: The complexity of **Algorithm 1** is $\mathcal{O}(\mathcal{N})$, which has a linear relationship with the number of robot clients.

Algorithm 1 start to computing λ^* and K_{th} by assuming $K_{th} = N$ and calculate λ. If the condition of (15) is not satisfied, K_{th} is decreased by one and λ is recalculated until it is satisfied. Because $\omega_1 \leq \lambda_1$ and $\lambda_1 = \frac{1}{T_0+1}$, Algorithm 1 always converges and returns the unique value of K_{th} and λ^*.

Algorithm 1: The Revenue Maximization Algorithm

Inputs: ω_i, T_0 and N
Outputs: K_{th}, λ^*, t_i^*, and p_i^*

```
1 BEGIN
2   function Revenue(i, ω_i, T_0, N)
3       k ← N, λ(k) ← (Σ_{i=1}^{k} √ω_i / (T_0+k))^2
4       while ω_k ≤ λ(k) do
5           k ← k − 1, λ(k) ← (Σ_{i=1}^{k} √ω_i / (T_0+k))^2
6       end while
7       K_th ← k, λ* ← λ(k)
8       t_i* = √(ω_i/λ*) − 1
9       p_i* = ω_i/(1+t_i*)
10  return K_th, λ*, t_i*, p_i*
11 END
```

4.3 Scheduling Scheme

All the previous theoretical analysis indicates the proposed scheduling scheme can optimize the MSDR problem. The basic operation of the scheduling scheme is implemented in CCH and comprises the following processes:

- **Admission control**: When a service request is submitted, request negotiator utilizes the proposed admission control mechanism (see **Algorithm 1**) to interpret the request before determining whether to accept or reject it according to the optimal threshold K_{th}. Thus, it ensures that there is no overloading of information, and sufficient requests can be fulfilled successfully. In the `factory`, a threshold is

set for the admitted number of robot clients considering the factors such as ToR and willingness to pay.

- **Request ranking**: The request negotiation is responsible for ranking the admitted requests considering their willingness to pay and time deadline as presented in **Algorithm 2**. Having access to the allocation requests of all robot clients, the CCH can keep tracking current robot clients, and update the ranking list when a new request is registered.
- **Resource distributing**: The admitted requests are responded in accordance with the order in the rank list. In this situation, it optimizes both the utility of each RC and the revenue of CCH. When new requests from robot clients arrive, the resource allocator would response the requests in accordance with the order of the updated rank list.

Algorithm 2: Scheduling Algorithm

Inputs: optimal price p_i^* of request i
Outputs: current_priority_list

```
1 BEGIN
2   function update_priority_list(p_i)
3       current_priority_list.append(p_i)
4   function is_lowest_priority(p_i)
5       current_priority_list.sort(p_i)
6   while i ≤ n_threshold
7       update_priority_list(p_i)
8       is_lowest_priority(p_i)
9   return current_priority_list
10 END
```

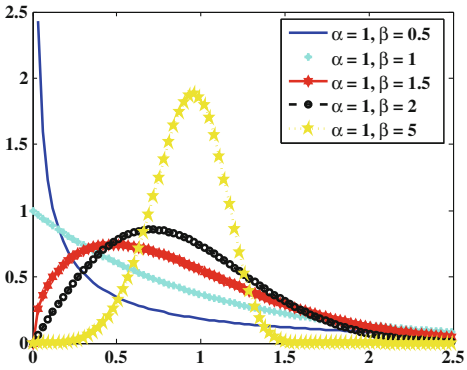
5 Simulation

In this section, the evaluation of mechanism parameter investigation and time cost are presented.

5.1 Parameter Investigation

In the proposed admission control, the K_{th} is determined by the distribution of willingness to pay from robot clients. If there are too many clients with high willingness to pay, the ones with relatively low willingness to pay will not be allocated data. Then the CCH can reduce the K_{th} to fulfill the resource retrieval before the deadline, namely admission control. However, there is no restriction on how to choose willingness to pay as a robot clients. Weibull distribution [49] is chosen because it is a versatile distribution that can represent different kinds of statistical distribution by changing its parameters as shown in Fig. 5. Therefore, Weibull distribution can take on various characteristics based on function as follows:

Fig. 5 Weibull distribution



$$f(x;\alpha,\beta)\begin{cases}\frac{\beta}{\alpha}(\frac{x}{\alpha})^{\beta-1}e^{-(\frac{x}{\alpha})^{\beta}} & x\geq 0,\\ 0 & x<0,\end{cases} \tag{16}$$

where $\alpha \geq 0$ is the scale parameter, and $\beta \geq 0$ is the shape parameter. If the quantity x is the number of clients with a willingness to pay, and the Weibull distribution demonstrates the proportion of the high willingness to pay clients. Then β can be interpreted directly as follows:

- $0 < \beta \leq 1$: $f(x)$ decreases monotonously and is convex as x increases to ∞. Especially, it is an exponential distribution when $\beta = 1$.
- $\beta > 1$: $f(x)$ has a bell-shape, which increases as x increases to the maximum and decreases thereafter. Especially, it is a Rayleigh distribution of mode $\sigma = \frac{\alpha}{\sqrt{2}}$ when $\beta = 2$.

In order to indicate the relationship between willingness to pay and the threshold of admitted number of clients in the proposed admission control, we compare the optimal K_{th} under different distributions of willingness to pay from all clients by tuning three factors: the shape parameter of Weibull distribution β that indicates different distribution of willingness to pay; the *number of clients requested resource* "N"; and the *timeout period* "T_0", which is a required time for a task.

In the simulation, we tested the admission control proposed in Sect. 4 by selecting $\alpha = 1$ and $\beta = \{0.1, 0.5, 1.0, 1.5, 5\}$, the time deadline $T_0 = \{10, 20, 30, 40\}$ and client number $k = \{12, 48, 96, 192\}$ respectively. One hundred runs were carried out on each configuration. In Fig. 6, we can be seen that K_{th} increases as β increases when T_0 is fixed. Especially, the increasing rate of K_{th} when $0 < \beta < 1$ is much larger than the increasing rate when $\beta > 1$. This is because the ratio of clients with high willingness to pay is smaller in the range of $0 < \beta < 1$. Moreover, it also shows that the willingness to pay is a key factor for the designation of the scheduler since it can affect the QoS. Moreover, the above results are references for the evaluation in the next subsection.

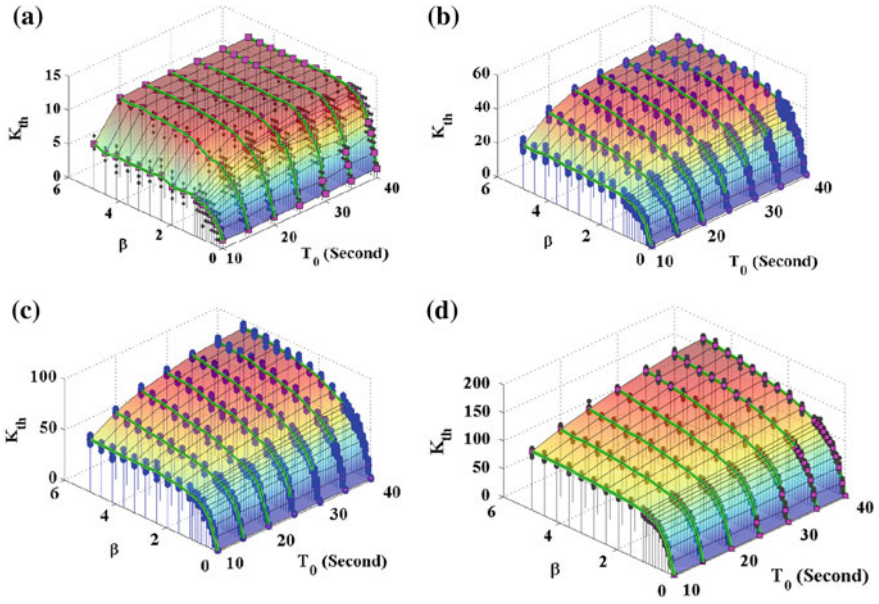
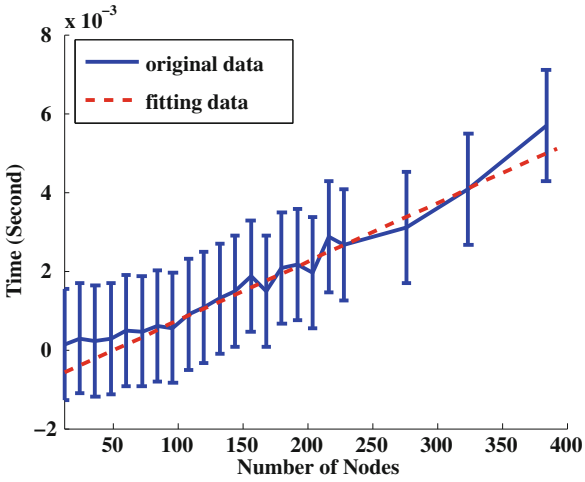


Fig. 6 Comparison of the threshold of admitted user number. The black points are the calculated K_{th} of 100 runs on each configuration, magenta squares are average values from each 100 runs, *green curves* are average values of K_{th} under different T_0, and the colored surface is a regression over all the average values. **a** Comparison of the threshold of admitted number of clients when there are 12 clients in total, **b** Comparison of the threshold of admitted number of clients when there are 48 clients in total, **c** Comparison of the threshold of admitted number of clients when there are 96 clients in total, **d** Comparison of the threshold of admitted number of clients when there are 192 clients in total

5.2 Time Cost of the Scheduling Scheme

The time cost of the scheduling scheme to get the optimal rank, including admission control and request ranking, is directly proportional to the number of nodes as shown in Fig. 7. The time cost is justified when the number of nodes is in the range of [12, 230] with an increment of 12 nodes for each test. In addition, it keeps in tens of milliseconds when the numbers of nodes are 276, 324, and 384. A fitted line is showed in the red dash line. Figure 7 in this document shows the fitted line has a gradient 1.4951×10^5, which means the cost time increase with the number of robot clients.

Fig. 7 Time cost



6 Experiment

This section introduces the experiment design, experimental results and performance discussion.

6.1 Robot Setup

The hardware system is composed of two major categories of robots. The leading robot is set up mainly to work as the database feeder while the others act as consumers of the feeded data.

- Well-equipped leading robot: the leading robot is shown in Fig. 8a. It equips with several sensors like a rotating laser scanner (for 3D point-cloud), an omni-camera (Ladybug™) and an Inertial Measure Unit (IMU) with a GPS module. It can provide an online database with sufficient mapping and localization hints.
- Relatively poorly-equipped follower robot: the follower robot "Epuck" is shown in Fig. 8b. It equips with a Firefly™ camera and a WiFi module. It can request various types of sensor data, for example, the camera can capture 2D bar-codes on the wall in the target environment, then the WiFi module can send it to the host to request the location or the regional map around it.

6.2 Experiment Design

A typical co-localization scenario is shown as Fig. 9. We put up several markers in the environment, which pose can be estimated. The well-equipped robot built a full

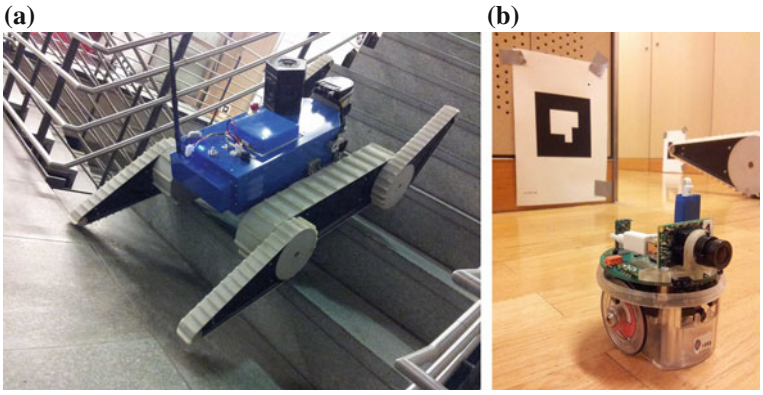


Fig. 8 Robots instances in a typical cloud robotic system, **a** Leading robot: NIFTi, **b** Follower robot: Epuck

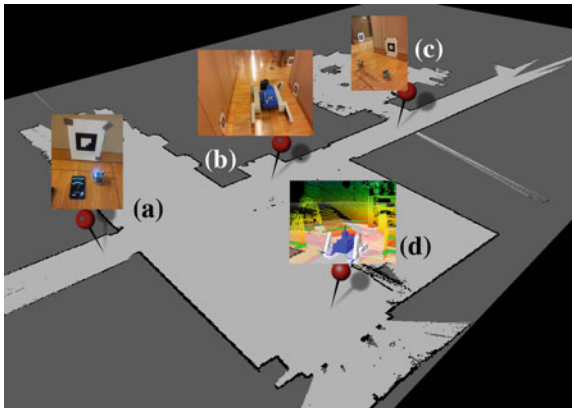


Fig. 9 A map with 3D point cloud of a typical indoor environment for multi-robot co-localization

3D map with marker location registered on it as shown in Fig. 9d. All information on the map were stored in a data center and can be subscribed by follower robots according to the response rank. In the aspect of poorly-equipped robots, they can use their camera to take pictures of AR markers as depicted in Fig. 9a, b. For the specific task, a structure for data flow and resource management is described in Fig. 10. Many services are provided in the cloud such as object detection, node initialization, besides localization as extensions. Robot client can retrieve these services though the structure we introduced in Sect. 3.3.

2D Barcodes Augmented Reality (AR), as shown in Fig. 9, is utilized as 2D barcodes for localization. It provides a package named ARToolKit, which uses computer vision algorithms to calculate the real camera position and orientation relative to physical markers in real time.

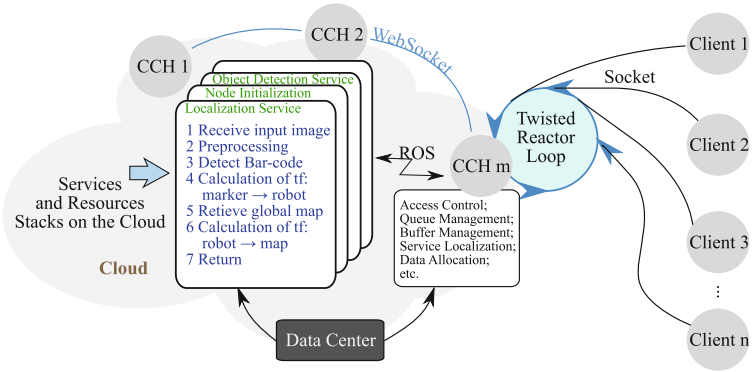


Fig. 10 Data flow and resource management of the proposed system

Marker Pose Registration and Retrieval All this location registration is implemented by ROS since ARToolKit is a package belonged to it. Not all AR markers can be accurately recognized and stable registered, we set a threshold to classify them. Only when the confidence is higher than the preset threshold, the location can be recorded in the database.

Pose Estimation At the beginning, the leading robot builds a 3D map with both images and registered local maps. Then the position of each AR marker is registered on the map by subscribe the related ROS topic, such as transformations. The relative poses are calculated by ARToolKit module.[1] At last, the pose of marker i in the map can be obtained by

$$T^{/map}_{/marker_i} = \underbrace{T^{/map}_{/leading\_robot}}_{\text{SLAM}} \cdot \underbrace{T^{/leading\_robot}_{/camera\_leading\_robot}}_{\text{Sensor Calibration}} \cdot \underbrace{T^{/camera\_leading\_robot}_{/marker_i}}_{\text{ARToolKit}}, \quad (17)$$

where T^a_b is the transformation matrix from frame b to a, namely target frame b in base frame a. Similarly, the pose retrieval for each follower robot j regarding marker i is formulated as

$$T^{/map}_{/robot_j} = \underbrace{T^{/map}_{/marker_i}}_{\text{Data Retreival}} \cdot \underbrace{(T^{/camera\_robot_j}_{/marker_i})^{-1}}_{\text{ARToolKit}} \cdot \underbrace{T^{/camera\_robot_j}_{/robot_j}}_{\text{Sensor Calibration}}. \quad (18)$$

We could see that the data retrieval efficiency will determine the efficiency of the whole co-localization system since it provides a required link. This problem is non-trivial when the system scale is large. Due to the limited bandwidth constraints and constraints of computational ability, the response of such information needs to be negotiated within robot clients and to be managed by the CCH.

[1] www.rog.org/wiki/artoolkit.

Table 1 Matched marker number vs confidence threshold under different WiFi strength

Number of markers	Threshold of confidence	Averaged ratio of matched markers	
		−40 ∼ −50 dB (%)	−50 ∼ −60 dB (%)
30	0.5	84.45	71.24
	0.9	73.28	60.15
100	0.5	80.97	69.22
	0.9	62.64	55.34

Table 2 Measured accuracy results of X-Y-Z offset error

Error marker	Z-offset	X-offset	Y-offset
Standard derivation	32.25884	10.97756	5.197521
Average (mm)	22.87656	–2.36570	0.480358

6.3 Qualitative Results on Localization Behavior

In the online co-localization scenario, all robot clients can be localized in firm real-time by request from the CCH. In this work, localization accuracy is mainly affected by matched marker ratio and marker location registered in the data center. Therefore, as shown in Table 1 we compare the matched marker number with confidence threshold which is pre-set in the `ar_multi.cpp`. For deriving better results, the WiFi strength is recorded when follower robots are requiring location. Moreover, the localization accuracy of ARToolKit is tested in [4], the localization accuracy is evaluated and shown in Table 2. The robot positions are well localized.

6.4 Quantitative Results on Resource Allocation

At first, we compare the *ToR* of continuous requests from 3 robot clients. Figure 11 demonstrates that the *ToR* of requests from all robots in a period. Comparing with the case without mechanism depicted in Fig. 11a, the proposed mechanism managed to reduce the *ToR* according to the priority setup. With the proposed mechanism depicted in Fig. 11b, robot clients received more retrieval data from the cloud. The priority setup for robot 1, 2 and 3 are 1, 3 and 2 respectively. With higher priority, the corresponding requests got a faster response at most of the time duration.

The typical characteristic of cloud robotic is the large number of robot requests in parallel. We compared the *RoR* performance considering the K_{th} and T_0 in the request tasks of 12 clients, which were data retrievals through the Internet. *RoR* of co-localization task depends on the retrieved transformation data registered in the database which can be easily calculated according to (17) and (18). For each request

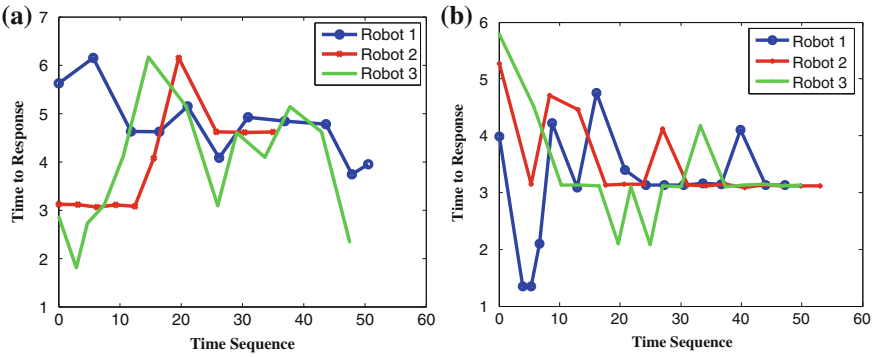


Fig. 11 ToR comparison, **a** Without mechanism, in unit of millisecond, **b** With mechanism, in unit of millisecond

task, it includes 6 independent requests from one client, the package size is $s = 15.625\,\text{Mb} \times 6$, then the ideal transmission time should be $s/(2\,\text{Mb/s}) = 46.875\,\text{s}$. Note that, only partial requests in the buffer queue would get responses from the CCH. It is because the transmission requires time, where the transmission period may be longer than the T_0. In Table 3, we demonstrate the *RoR* among different T_0 and K_{th}. Clients submitted their optimal price of requests, which were determined by their willingness to pay and the desired completing time of the target data retrieval. The results validate that the *RoR* with mechanism performs better, when K_{th} is optimized for each task to respond to their timeout period. In addition, willingness to pay of all clients are uniformly distributed because they have the same requests.

6.5 Discussions

In the online experiment, the proposed system distributed the workload of sensing, localization, computation and communication among a group of robot agents. The regarding characteristics are discussed as follows.

Optimized Constraints of Resource Allocation It greatly reduced the response time for the localization task by deployment of access control, scheduling and protocol management in the proposed cloud robotic system. The optimization is derived from solving the following constraints.

- Data retrieval constraint: in cloud robotic system, a robot can retrieve information from a dynamically updated data center which is built by various types of robots and sensors, therefore it is a heterogeneous structure that needs standard design and regulation to fit it in applications.

Table 3 RoR comparison between with and without mechanism under different timeout period

Threshold K_{th}	Timeout period T_0 (s)							
	10		20		30		40	
	No Mechanism (%)	Mechanism (%)	No Mechanism (%)	Mechanism (%)	No Mechanism (%)	Mechanism (%)	No Mechanism (%)	Mechanism (%)
6	0	0	0	70.61	0.39	100	54.17	100
8	0	0	0	73.83	1.67	100	68.06	100
10	0	0	0	76.39	2.78	100	69.44	100

- Communication constraint: synchronization of data is hard for distributed system, especially when asynchronous tasks are performed [29]. If multi-sensor data are distributed in the network on each client, information sharing among robots is low-efficient without resource allocation.
- Autonomous negotiation constraint: different from research on target tracking [7], automated identification of targets [33], and automated behavior reasoning [30]. Practical autonomous negotiation for resource allocation in firm real-time systems such as cloud robotics is a multi-dimensional problem, therefore both revenues of resource provider and utility of user are considered in this chapter.

Generalization of the Proposed Framework By using such framework, the major generalization can be derived as follows:

- Extension to cloud robotic system with many poorly-equipped robots for information retrieval and communication. It considers of letting robots access a large amount of computational power on demand. The framework sidesteps drawbacks include high computational cost, high configuration, maintenance and update overheads.
- Extension of a more complex hierarchical topology of the system including various types of robots. Especially the negotiation mechanism can be extended according to complex task and environment that it applies.

7 Conclusion

In this chapter, a novel cloud robotic system architecture is developed based on an asynchronous data flow framework. Then the resource allocation problem is formulated as a Stackelberg game and the corresponding solution is proposed. Moreover, the task-oriented QoS criteria are proposed. Afterward, simulations on parameter investigation and time cost are presented. Experiments of co-localization robotic scenarios are implemented for evaluation. Results are discussed and proved the proposed pricing mechanism optimized the resource allocation problem for cloud robotic systems considering physical robots and tasks.

Even if experiments and simulations in the thesis showed promising results, there are still some further work to be developed and extended. Specifically, both the large amount of resource in the cloud and robot clients are heterogeneous. Therefore, a vital issue in this domain is uncertainty about the resource prices, which greatly affect the cost of request retrieval and allocation. As the proposed Stackelberg game mechanism, the optimal price depends on a willingness payment and the corresponding response time. However, the response time is not a prior, which is determined by the willingness payment and a Lagrange multiplier. Instead of depending on the willingness payment, a schedule concerning the price prediction model could be explored to enhance the performance of resource retrieval. Future development of the auction-based mechanisms includes developing and analyzing of optimization

algorithms that are able to tune the variables in the strategy according to the different information conditions. This will then increase the robustness of this resource allocation strategy to make it suitable for all environments and applications.

Acknowledgments This work is supported by RGC GRF Grant CUHK14205914 awarded to Prof. Max Q.-H. Meng; partially supported the Research Grant Council of Hong Kong SAR Government, China, under project No. 16206014 and No. 16212815; National Natural Science Foundation of China No. 6140021318, awarded to Prof. Ming Liu.

References

1. Aldebaran Robotics: Nao robot, http://aldebaran-robotics.com/
2. An, B., Lesser, V.: Characterizing contract-based multiagent resource allocation in networks. IEEE Trans. Syst. Man Cybern. Part B: Cybern. **40**(3), 575–586 (2010)
3. Arumugam, R., Enti, V., Bingbing, L., Xiaojun, W., Baskaran, K., Kong, F.F., Kumar, A., Meng, K.D., Kit, G.W.: DAvinCi: a cloud computing framework for service robots. In: 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 3084–3089, May 2010
4. Berard, B., Petrie, C., Smith, N.: Quadrotor UAV interface and localization design. In: A Major Qualify Project of Missile Defense Agency under Air Force Contract F19628-00-C-0002 (1Apr00–31Mar05) (2010)
5. Berhault, M., Huang, H., Keskinocak, P., Koenig, S., Elmaghraby, W., Griffin, P., Kleywegt, A.: Robot exploration with combinatorial auctions. In: Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), vol. 2, pp. 1957–1962, October 2003
6. Bichier, M., Lin, K.J.: Service-oriented computing. Computer **39**(3), 99–101 (2006)
7. Brooks, A., Williams, S.: Tracking people with networks of heterogeneous sensors. In: Proceedings of the Australasian Conference on Robotics and Automation, pp. 1–7. Citeseer (2003)
8. Burkard, R.E., Dell'Amico, M., Martello, S., et al.: Assignment Problems. Revised Reprint, SIAM (2009)
9. Chang, M., He, J., Castro-Leon, E.: Service-orientation in the computing infrastructure. In: 2nd IEEE International Symposium on Service-Oriented System Engineering, Shanghai, China, pp. 27–33, October 2006
10. Chen, Y., Du, Z., García-Acosta, M.: Robot as a service in cloud computing. In: 2010 5th IEEE International Symposium on Service Oriented System Engineering, pp. 151–158, June 2010
11. Cheng, M.Y., Tran, D.H., Wu, Y.W.: Using a fuzzy clustering chaotic-based differential evolution with serial method to solve resource-constrained project scheduling problems. Autom. Constr. **37**(0), 88–97 (2014), http://www.sciencedirect.com/science/article/pii/S0926580513001593
12. Choi, H.L., Brunet, L., How, J.P.: Consensus-based decentralized auctions for robust task allocation. IEEE Trans. Robot. **25**(4), 912–926 (2009)
13. Clark, J.: Inside "indigo"—infrastructure for web services and connected applications. Microsoft Press (Apr 2005)
14. Colas, F., Mahesh, S., Pomerleau, F., Liu, M., Siegwart, R.: 3d path planning and execution for search and rescue ground robots. In: 2013 IEEE/RSJ InternationalConference on Intelligent Robots and Systems (IROS), pp. 722–727. IEEE (2013)
15. Corporation, M.: Microsoft robotics developer studio, http://www.microsoft.com/Robotics
16. Darmann, A., Pferschy, U., Schauer, J.: Resource allocation with time intervals. Theor. Comput. Sci. **411**(49), 4217–4234 (2010). http://www.sciencedirect.com/science/article/pii/S0304397510004639
17. Dasarathy, B.: Decision Fusion, vol. 1994. IEEE Computer Society Press (1994)

18. Date, C., Darwen, H.: A Guide to the SQL Standard, vol. 3. Addison-Wesley Reading (1987)
19. Dias, M.B., Zlot, R., Kalra, N., Stentz, A.: Market-based multirobot coordination: a survey and analysis. Proc. IEEE **94**(7), 1257–1270 (2006)
20. Heinemann, F.C.: Web Programming with the Sap Web Applications Server. SAP Press (2003)
21. Gerkey, B.P., Mataric, M.J.: A formal analysis and taxonomy of task allocation in multi-robot systems. Int. J. Robot. Res. **23**(9), 939–954 (2004). http://dx.doi.org/10.1177/0278364904045564
22. Gostai Coop.: Gostai, http://www.gostai.com/
23. Hall, D., Llinas, J.: An introduction to multisensor data fusion. Proc. IEEE **85**(1), 6–23 (1997)
24. Higuera, J., Gamboa, C., Dudek, G.: Fair subdivision of multi-robot tasks. In: In: Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA 2013), pp. 2999-3004 (2013)
25. Hu, G., Tay, W.P., Wen, Y.: Cloud robotics: architecture, challenges and applications. Netw. IEEE **26**(3), 21–28 (2012)
26. Hunziker, D., Gajamohan, M., Waibel, M., D'Andrea, R.: Rapyuta: the RoboEarth cloud engine. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), pp. 438–444, May 2013
27. Intel Coop.: Service-oriented enterprise, the technology path to business transformation. (2 Oct 2005). http://www.intel.com/business/bss/technologies/soe/
28. Jalaparti, V., Nguyen, G., Gupta, I., Caesar, M.: Cloud resource allocation games. Technical report, Department of Computer Science (2010)
29. Kaempchen, N., Dietmayer, K.: Data synchronization strategies for multi-sensor fusion. In: Proceedings of the IEEE Conference on Intelligent Transportation Systems. Citeseer (2003)
30. Kam, M., Zhu, X., Kalata, P.: Sensor fusion for mobile robot navigation. Proc. IEEE **85**(1), 108–119 (1997)
31. Kehoe, B., Matsukawa, A., Candido, S., Kuffner, J., Goldberg, K.: Cloud-based robot grasping with the Google object recognition engine. In: IEEE International Conference on Robotics and Automation (ICRA) (2013)
32. Kehoe, B., Warrier, D., Patil, S., Goldberg, K.: Cloud-based grasp analysis and planning for toleranced parts using parallelized Monte Carlo sampling. IEEE Trans. Autom. Sci. Eng. **12**(2) (2015)
33. Klimentjew, D., Hendrich, N., Zhang, J.: Multi sensor fusion of camera and 3d laser range finder for object recognition. In: 2010 IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pp. 236–241. IEEE (2010)
34. Lagoudakis, M.G., Markakis, E., Kempe, D., Keskinocak, P.: Auction-based multi-robot routing. In: Proceedings of the International Conference on Robotics: Science and Systems (ROBOTICS), pp. 343–350 (2005)
35. Lemaire, T., Alami, R., Lacroix, S.: A distributed tasks allocation scheme in multi-UAV context. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, ICRA'04, vol. 4, pp. 3622–3627. IEEE (2004)
36. Lin, L., Zheng, Z.: Combinatorial bids based multi-robot task allocation method. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA 2005), pp. 1145–1150. IEEE (2005)
37. Lingzhi, L., Nilanjan, C., Sycara, K.: Distributed algorithm design for multi-robot task assignment with deadlines for tasks. In: IEEE International Conference on Robotics and Automation, (ICRA 2013), pp. 2992–2998 (2013)
38. Liu, M., Pradalier, C., Pomerleau, F., Siegwart, R.: Scale-only visual homing from an omnidirectional camera. In: 2012 IEEE International Conference on Robotics and Automation (ICRA), pp. 3944–3949. IEEE (2012)
39. Liu, M., Colas, F., Oth, L., Siegwart, R.: Incremental topological segmentation for semi-structured environments using discretized GVG. Auton. Robot. **38**(2), 143–160 (2014)
40. Liu, M., Siegwart, R.: DP-FACT: towards topological mapping and scene recognition with color for omnidirectional camera. In: 2012 IEEE International Conferenceon Robotics and Automation (ICRA), pp. 3503–3508. IEEE (2012)

41. Liu, M., Pradalier, C., Siegwart, R.: Visual homing from scale with an uncalibrated omnidirectional camera. IEEE Trans. Robot. **29**(6), 1353–1365 (2013)
42. Liu, M., Siegwart, R.: Information theory based validation for point-cloud segmentation aided by tensor voting. In: International Conference on Information and Automation (ICIA). IEEE (2013)
43. Liu, M., Siegwart, R.: Navigation on point-clouda—Riemannian metric approach. In: 2014 IEEE International Conference on Robotics and Automation(ICRA), pp. 4088–4093. IEEE (2014)
44. de Lope, J., Maravall, D., Quiñonez, Y.: Response threshold models and stochastic learning automata for self-coordination of heterogeneous multi-task distribution in multi-robot systems. Robot. Auton. Syst. **61**(7), 714–720 (2013)
45. Matti, P.: Nash and Stackelberg solutions in a differential game model of capitalism. J. Econ. Dyn. Control **6**(0), 173–186 (1983), 0165-1889. doi:10.1016/0165-1889(83)90048-9
46. McMurtry, C., Mercuri, M., Watling, N.: Microsoft Windows Communication Foundation: Hands-on. Sams Press (May25 2006)
47. Moshe Zadka, G.L.: The twisted network framework. http://twistedmatrix.com/user/glyph/ipc10/paper.html (2010)
48. Mosteo, A.R., Montano, L.: A survey of multi-robot task allocation. Technical report, Instituto de Investigacin en Ingenierła de Aragn (I3A) (2010)
49. Mudholkar, G.S., Srivastava, D.K., Kollia, G.D.: A generalization of the weibull distribution with application to the analysis of survival data. J. Am. Stat. Assoc. **91**(436), 1575–1583 (1996)
50. Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., Zagorodnov, D.: Eucalyptus: a technical report on an elastic utility computing architecture linking your programs to useful systems (2008)
51. OpenNebula Project: Opennebula.org—the open source toolkit for cloud computing, http://www.opennubula.org/
52. Piaggio, M., Zaccaria, R.: Distributing a robotic system on a network: the ETHNOS approach. Adv. Robot. **11**(8), 743–758 (1998)
53. Rai, A., Bhagwan, R., Guha, S.: Generalized resource allocation for the cloud. In: Proceedings of the 3rd Symposium on Cloud Computing (SOCC). San Jose, CA, October 2012
54. Riazuelo, L., Civera, J., Montiel, J.: C^2 TAM: a cloud framework for cooperative tracking and mapping. Robot. Auton. Syst. **62**(4), 401–413 (2014)
55. Rimal, B.P., Choi, E., Lumb, I.: A taxonomy and survey of cloud computing systems. In: Fifth International Joint Conference on INC, IMS and IDC, 2009. NCM'09. pp. 44–51. IEEE (2009)
56. Rodriguez, M., Buyya, R.: Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. IEEE Trans. Cloud Comput. **2**(2), 222–235 (2014)
57. Steven, C.: Robots, incorporated. IEEE Spectrum (August 2007). http://www.spectrum.ieee.org/aug07/5391
58. Sempolinski, P., Thain, D.: A comparison and critique of eucalyptus, opennebula and nimbus. In: 2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), 30 2010-December 3 2010, pp. 417 –426
59. Sheth, A., Ranabahu, A.: Semantic modeling for cloud computing, part 1. Internet Comput. IEEE **14**(3), 81–83 (2010)
60. Sung, C., Ayanian, N., Rus, D.: Improving the performance of multi-robot systems by task switching. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), pp. 2999–3006. IEEE (2013)
61. Tan, M., Wang, L., Tardioli, D., Liu, M.: A resource allocation strategy in a robotic ad-hoc network. In: 2014 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), pp. 122–127, May 2014
62. Thiruvady, D., Ernst, A., Singh, G.: Parallel ant colony optimization for resource constrained job scheduling. Ann. Oper. Res. 1–18 (2014). http://dx.doi.org/10.1007/s10479-014-1577-7
63. Tsai, W., Huang, Q., Sun, X.: A collaborative service-oriented simulation framework with microsoft robotic studio. In: 41st Annual. Simulation Symposium, ANSS 2008, pp. 263–270, April 2008

64. Tsai, W., Sun, X., Huang, Q., Karatza, H.: An ontology-based collaborative service-oriented simulation framework with microsoft robotics studio. Simul. Model. Pract. Theory **16**(9), 1392–1414 (2008)
65. Viguria, A., Maza, I., Ollero, A.: Set: an algorithm for distributed multirobot task allocation with dynamic negotiation based on task subsets. In: 2007 IEEE International Conference on Robotics and Automation, pp. 3339–3344. IEEE (2007)
66. Viguria, A., Maza, I., Ollero, A.: S+T: an algorithm for distributed multirobot task allocation based on services for improving robot cooperation. In: IEEE International Conference on Robotics and Automation, ICRA 2008, pp. 3163–3168 (2008)
67. Waibel, M., Beetz, M., Civera, J., D'Andrea, R., Elfring, J., Galvez-Lopez, D., Haussermann, K., Janssen, R., Montiel, J., Perzylo, A., Schiessle, B., Tenorth, M., Zweigle, O., van de Molengraft, R.: RoboEarth. IEEE Robot. Autom. Mag. **18**(2), 69–82 (2011)
68. Wang, L., Liu, M., Meng, M.Q.H.: Towards cloud robotic system: a case study of online co-localization for fair resource competence. In: IEEE International Conference on Robotics and Biomimetics (ROBIO 2012), pp. 2132–2137, December 2012
69. Wang, L., Liu, M., Meng, M.Q.H.: Real-time multi-sensor data retrieval for cloud robotic systems. IEEE Trans. Autom. Sci. Eng. **12**(2) (2015)
70. Wang, L., Liu, M., Meng, M.Q.H., Siegwart, R.: Towards real-time multi-sensor information retrieval in cloud robotic system. In: 2012 IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pp. 21–26 (September 2012)
71. Wang, L., Liu, M., Meng, M.H.: An auction-based resource allocation strategy for joint-surveillance using networked multi-robot systems. In: 2013 IEEE International Conference on Information and Automation (ICIA), pp. 424–429, (August 2013)
72. Wang, L., Liu, M., Meng, M.H.: Hierarchical auction-based mechanism for real-time resource retrieval in cloud mobile robotic system. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), June 2014
73. Wang, L., Meng, M.H.: A game theoretical bandwidth allocation mechanism for cloud robotics. In: 2012 10th World Congress on Intelligent Control and Automation (WCICA), pp. 3828–3833, July 2012
74. Zhang, Y., Parker, L.E.: Multi-robot task scheduling. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), pp. 2992–2998. IEEE (2013)

Study of Communication Issues in Dynamically Scalable Cloud-Based Vision Systems for Mobile Robots

Javier Salmerón-García, Pablo Iñigo-Blasco, Fernando Díaz-del-Río and Daniel Cagigas-Muñiz

Abstract Thanks to the advent of technologies like Cloud Computing, the idea of computation offloading of robotic tasks is more than feasible. Therefore, it is possible to use legacy embedded systems for computationally heavy tasks like navigation or artificial vision, hence extending its lifespan. In this chapter we apply Cloud Computing for building a Cloud-Based 3D Point Cloud extractor for stereo images. The objective is to have a dynamically scalable solution (one of Cloud Computing's most important features) and applicable to near real-time scenarios. This last feature brings several challenges that must be addressed: meeting of deadlines, stability, limitation of communication technologies. All those elements will be thoroughly analyzed in this chapter, providing experimental results that prove the efficacy of the solution. At the end of the chapter, a successful use case of the platform is explained: navigation assistance.

Keywords Cloud computing · Computation offloading · Robotics · Dynamic scalability

1 Introduction

Nowadays, new computationally expensive tasks are expected to be performed with relative fluency by robotic platforms. A well-known example is that of artificial vision and higher level tasks arisen from it, such as object detection, recognition and tracking; surveillance, gesture recognition, etc. However, these tasks are so computationally heavy that they may take several seconds in current embedded robot computers. Hence the advantages of using computation offloading (i.e. moving this computing task to an external computer system) are becoming evident in terms of time to finish the task, mobile robot energy saving, amongst others.

J. Salmerón-García (✉) · P. Iñigo-Blasco · F. Díaz-del-Río · D. Cagigas-Muñiz
Escuela Técnica Superior de Ingeniería Informática, University of Seville,
Av. Reina Mercedes S/n, 41012 Sevilla, Spain
e-mail: jsalmeron2@us.es

A. Koubaa and E. Shakshuki (eds.), *Robots and Sensor Clouds*,
Studies in Systems, Decision and Control 36,
DOI 10.1007/978-3-319-22168-7_2

An interesting candidate for the aforementioned external computer system is that of a Cloud infrastructure, thanks to its inherent characteristics [7]: high reliability, larger storage capacity, stable electric power, reutilization of hardware, dynamic scalability and better resource utilization. In particular, the dynamic scalability property is very useful in robotics, as it allows the adaptation of the computing power at runtime (that is, scaling out and back, depending on the needs), and therefore it permits the robot to rapidly adapt in a changing environment. Moreover, another advantage of the Cloud is the instant incorporation of more computation demanding algorithms as they are being implemented.

Apart from computationally heavy tasks, the cloud is being used as a centralized powerful infrastructure for multi-robot cooperative systems that usually works at intermediate levels. This area is intensively studied as new cooperative algorithms are being developed. Examples of these solutions are multi-robot SLAM (simultaneous localization and mapping), map merging (acquired by several robots), networked information repository for robots [23], etc.

As a result, an important number of research papers and projects addressing the use of cloud infrastructures in robotics have been published during the last few years (see Sect. 3). Besides, the term Cloud Robotics has emerged to include this area, which promises a fast development of complex distributed robotics tasks in the forthcoming years.

This chapter addresses the "computation offloading" of an intermediate robot level task: 3-D point cloud (3DPC) extraction from stereo image pairs. In order to do so, a Cloud-based 3DPC extraction platform will be developed. This platform has innumerable applications, such as AI, artificial vision and navigation. The latter is especially interesting, as 3DPCs are one of the most used representation for several navigation tasks, including those of motion planning and obstacle avoidance. Because of that, at the end of the chapter (Sect. 6.5) a navigation use case of our platform will be briefly explained.

In this respect, current embedded computers are able to extract a 3D point cloud and to build a map of the surrounding obstacles in a natural and dynamically changing environment in less than a second, providing that low resolution frames are used. Nevertheless, when an accurate vision is needed, frame resolution must be increased. In addition to this, should the robot be in a fast changing environment, then higher frame rates would be necessary. As a consequence, the limitations of robot embedded hardware will likely arise, and thus sending the stereo image pairs to the cloud can compensate the inherent trade-offs of network communications (explained in Sect. 5.2 in more detail).

In order to exploit the cloud capabilities, the implemented platform must be able to scale out and back, so the robot gets the results faster when more computation power is required. Hence, a dynamically parallel algorithm has been implemented. In this context, the quotient between computation and communication times mainly indicates if the parallelization is to be successful. The developed platform is expected to be applied to near real-time systems as well, which is not without several challenges in terms of meeting deadlines. In this respect, the ratio between local-to-cloud transfer

time (especially in the case of large amounts of data) and computation time in a single node must be taken into account as well, as it indicates the usefulness of cloud offloading.

Section 3 summarizes several related works. Before presenting the developed platform, a thorough analysis of which robotics tasks (especially those that need some soft real time requirements) are candidate for computation offloading is done in Sect. 4. An overall analysis of the implemented solution is depicted in Sect. 5 (together with a time analysis). Experimental results are shown in Sect. 6 to quantify the benefits of the cloud approach for different scenarios. In this last section, we summarize an example application case of the presented platform: a navigation assistant for mobile robots [19]. Finally, conclusions are summarized in Sect. 7.

2 Background

This book chapter covers several areas. The first (and most important) is that of Cloud Computing. In this sense, books like [20] can be helpful for understanding its inherent characteristics. More specifically, this chapter focuses on the idea of computation offloading of High Performance Computing applications. Therefore, so some basic concepts this kind of applications, together with basic concepts on parallelism applied to cloud computing, are crucial to understand this chapter. These concepts are clearly explained in [4].

In addition to this, the developed platform is used for stereo vision, and more specifically in 3D Point Cloud extraction. The readers can find several works in the literature regarding this specific topic, for instance [22]. However, it is not necessary to know how a 3D Point Cloud is obtained from stereo frame pairs (that is, debayering, rectification, amongst others) to understand the contents of this chapter.

Moreover, The presented software solution was developed using the ROS Platform. Basic information about this software, together with beginner tutorials, can be found in their wiki (http://wiki.ros.org/). In [12] there is a thorough outline of current Robotics Software Frameworks (RSF).

Finally, communication issues are covered in this chapter. Therefore, readers can read [21] to get basic knowledge about basic networking concepts and technologies.

3 Related Work

In the last few years works and projects that accomplish high level vision tasks without real time requirements are more and more common [2, 9, 23]. Most of them use the cloud robotics paradigm to offload the robot from high level tasks like those related with visual processing or multirobot cooperation. In our opinion this is a tendency that will burst in the next decade, due to the previous cloud computing advantages.

However, a small group of papers proposes offloading the processing of several parts of the sensor feedback information that are near to real-time. For those operations, a fast and reliable response is needed. In [3] a high resolution SIFT-based object detection is speeded up by transmitting on-board preprocessed image information instead of raw image data to external servers. Here, properties of the cloud computing paradigm are not fully exploited, because the configuration of these external servers is specific to this work.

The idea of Computation Offloading is studied in [16]. These authors present an estimation of the computation and communication times needed for the tasks of recognition and object tracking in order to minimize the total execution time (approaching the real-time constraints). Their analysis permits making offloading decisions for object recognition for different bandwidths, background complexities, and database sizes. In this sense, the method for identifying the optimal balance between a cloud system overhead and performance presented in [8] can be useful. Executing SLAM in the cloud is also studied in [18], where they develop a cloud mapping framework (C2TAM). They combine both computation offloading and collaborative work, as the framework allows fusing the information obtained from several robots. They work with a 640×480 pixel RGBD camera and an average data flow of 1 MB/s, below 3 MB/s, which is the usual wireless bandwidth and hence the mapping is successfully done (moreover, they work with keyframes, reducing the amount of images to send).

In [24] an object-tracking scenario for a 14-DOF industrial dual-arm robot is presented. Standard UDP transport protocol for transmitting large-volume images over an Ethernet network is used. Thanks to the very low sending and cloud image processing times that are achieved, a stabilizing control law can be implemented. Due to the inherent time-varying Ethernet protocol delays, actuation signals incorporate an ingenious hold action.

Finally, the work [1] also asks whether the performance of distributed offloading tasks can be compared with those executed on-board. While the experiment performed here is simple (a visual line follower that guides the robot using a single low resolution camera that points to the floor), this demonstrator gives an idea of the possible scenario for many cloud-based robots of the upcoming future.

Contributions of the chapter: Compared with the described literature, the work presented in this chapter is the first that tries implement a stereo vision platform with focus on both dynamic scalability and near real-time applications. Moreover, a general offloading architecture (applicable to any case) is proposed, used to implement the presented platform. In addition to this, a thorough explanation of the the role of communication technologies, the problems of multi-robot and WiFi AC adds more novelty to our work.

4 Cloud Offloading for Robotics Applications

Figure 1 shows a block diagram of a Cloud-based computation offloading robotics applications. The robot's controller will collect any necessary environment information (using both internal and external sensors) to send the most convenient action to the actuators. Usually internal sensors (e.g. odometric) and simple sensor (e.g. sonars) are easy to be processed on-board, so they provide a fast and reactive feedback to the robot. On the contrary, the robot can include some others more complex or high level sensors (e.g. cameras), whose processing algorithms are more demanding both in computing time and energy.

Because of this, the Cloud-based approach aims that the robot's controller will be freed from heavy computations by offloading. In order to do so, it must send to the Cloud all the sensor information. While the size of high level sensoring is usually big, the rest of sensors suppose a few additional bits; hence all the information can be sent to the Cloud. This will even allow the Cloud to do more involved sensor fusion algorithms without demanding real time constraints (like SLAM). While the Cloud is performing all those high demanding computations, the robot can dedicate its computing power for other (real-time) tasks. Once the Cloud has the processed information ready and tailored to each robot, the robot will receive it and make use of it for whatever operation the robot may require (e.g. vision, AI, trajectory modification, etc.). Cloud offloading provides additional benefits due to the inherent centralization that the Cloud supposes for a distributed robotic team. For example, team collaborative tasks can be more easily and fluently handled in the Cloud as it can manage complete information from all the robots.

Even though its advantages are evident, there are several communication bounds and development issues when offloading robotics tasks. The software architecture (and its components) of a complex robotic system must cater for a variety of characteristics, which distinguish it from other system. The most relevant characteristics of are [12]: Concurrent and distributed architecture, Modularity (several components of high cohesion but low coupling), Robustness and fault tolerance; and real time

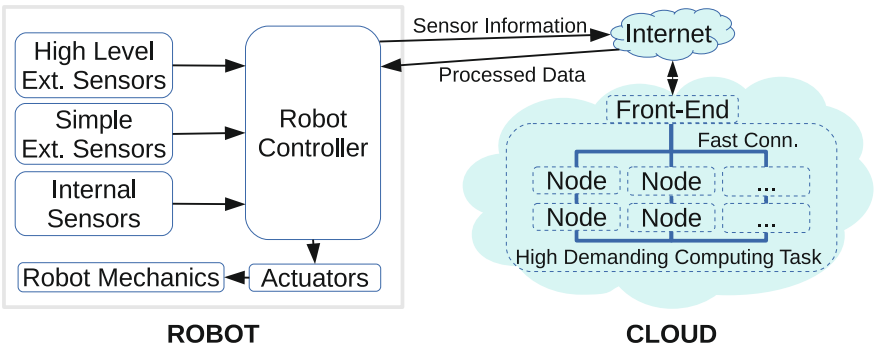


Fig. 1 Block Diagram of a Cloud-based computation offloading system

efficiency. The first two characteristics are primarily benefited by cloud offloading, while the third introduces new challenges (network robustness and fault tolerance appear as a new aspect to considerate). Nevertheless, due that the platform described in this chapter must cope with timing efficiency, communication delays are analyzed in the rest of this section.

As explained in Fig. 1 the robot has to send sensoring information packets to the cloud and wait until it receives the Cloud answer. The communication delays suppose an obligatory inferior bound in the loop controller period. Let BW be the network bandwidth rate and D, the total amount of transmitted and received data. Therefore D/BW must be inferior to the controller deadline. This minimum bound does not suppose for current network technologies a limit, except for the very reactive tasks. For example a WiFi AC networks can deliver until almost 1 Gbps [5]. Even for a demanding control loop of 50 Hz (higher than most mobile robot control loops), this bound would not be exceeded if the transmitted data were less than 0.02 Gbits, because the loop period is 0.02 s.

If images are to be transmitted, an amount of 20 Mbits of data represents 8 raw B/W images of 640×480 pixels (or 32 images for a lower resolution of 320×240 pixels). This suppose that the robot is sending 4 (16 for the lower resolution) stereo frame pairs at each period (without any compression). This is not the common case for a current robot, which is equipped usually with only one stereo camera. Moreover, currently the steady incremental ratio of WiFi networks is more than 40 % per year, which means that only in two years the bandwidth is predicted to duplicate. Hence it can be assured that theoretical bandwidth does not impose a limit in computation offloading. Nevertheless, this may not be the case for latency variability, as our results in Sect. 6.4 demonstrates.

Going further, a quantitative comparison of the times involved in local versus remote computing points out new outcomes. Let IPS the rate of instructions per second that the robot computer can execute [11]. Let us assume that the cloud can speedup an application S times more than the robot, that is, it gets an IPS of $S \cdot IPS$. A high S is expected because of several reasons. Firstly, the cloud is expected to have far more computational resources than a local (usually low power consuming) computer. Likewise, there are big amounts of data parallelism to be exploited when using many sensoring information (image processing, object, voice or face recognition, etc.). Finally these tasks are usually very repetitive. For instance, in image recognition, a pattern has to be compared with thousands of stored patterns. Hence, it can be supposed that S is very big for most sensoring applications. Therefore, for N_I computer instructions local and remote execution times are:

$$t_{local} = \frac{N_I}{IPS}; t_{remote} = \frac{N_I}{(IPS \cdot S)} + \frac{D}{BW};$$

And we can obtain this formula for timing comparison:

$$t_{local} > t_{remote} \text{ if } \frac{N_I}{D} > \frac{IPS}{BW}$$

which indicates whether computation offloading is faster than local computation, and gives us a prospect of which applications are prone to be offloaded. For example, let us compute an estimation of the two members of previous inequality for the Erratic Robot, which CPU runs at a frequency f = 1.4 GHz and has a CPI (Clocks per Instruction) around 2.0 [14]. Hence, if a frame pair is computed by this robot in $t_{exec} = 0.96$ s (see Sect. 6), the number of instructions that are executed [11] results:

$$N_I = \frac{(t_{exec} \cdot f)}{CPI} = 6.72 \cdot 10^8 instructions$$

Besides, transmitted data of this experiment (see Sect. 6) consists mainly in a color 1024×768 frame pair, that is: $D = 1024 \cdot 768 \cdot 3 \cdot 2 \cdot 8 = 3.77 \cdot 10^7 bits$. Hence:

$$\frac{N_I}{D} = 17.8 instr/bit$$

Let us remark that the first term of the inequality is application dependent, which means that high intensive computing tasks will be benefited by cloud computing. On the contrary, the second member is mainly technology dependent. Being $IPS = f/CPI$ [11], the Erratic CPU $IPS = 7.0 \cdot 10^8 instr/s$, but others platforms can achieve a higher *IPS*. Moreover, a 400 Mbps data rate transmission can be easily reached with WiFi IEEE 802.11ac. With these values, offloading is not bounded by time latency, as the second term has a very much lower value (1.75 for actual case) than the first one. As a conclusion, networking bandwidth is crucial for a successful offloading.

Let us finally make some predictions about the tendency of these two terms. If the last decade trend continues, uniprocessor IPS will have an annual growth rate much inferior to that of network technology [11]. This means that cloud offloading is promised to take even more advantage for the next years. With respect to the software development, it seems that the only possibility to speedup embedded CPU *IPS* is by means of more parallelism (and by more efficient tools to develop it). But this, indeed, would be beneficial for the advancement on cloud programming.

To sum up, it can be concluded that those applications with ratio N_I / D bigger than a few units, are currently candidates to be remotely executed. Those where this ratio would be inferior may be successfully offloaded in a few years. This includes many tasks from the top, middle, and even lowest, level of a common layered robot architecture (see Sect. 1). Moreover, independently of this ratio value, there are applications where the size of required (stored) data is huge (see Sect. 1). For them, maintaining local massive storage (in terms of power, failure immunity, backup, weight, etc.) is a hard problem and it is obvious that external offloading is the best solution.

5 3D Point Cloud (3DPC) Extraction Platform

Using the architecture shown in Fig. 1, the offloading of stereo vision tasks has been implemented. As stereo cameras are the high level external sensors, the robot will send a stereo video stream ("Sensor Information" in Fig. 1). The Cloud will extract all the necessary 3D information, sending that processed data (see Fig. 1) back to the robot in the form of 3D Point Clouds (3DPC). The resulting architecture can be seen in Fig. 2. These 3DPCs can be used by the robot to execute, for instance, a navigation algorithm (as explained in Sect. 6.5) or a SLAM algorithm (together with the internal sensing).

As mentioned in Sect. 1, with respect to the precision of the extracted information, the bigger the image resolution is, the more accurate the reconstruction of the surrounding objects will be (extensively demonstrated in the literature [10, 15]). The reason for choosing stereo cameras instead of other simpler sensors (such as those with infrared or ultrasonic technologies) is the completeness of the information they can offer, as well as they can serve for other high level visual tasks (like object detection and recognition, gesture recognition, etc.).

5.1 Software Implementation

In order to convert the image stream sent by the robot to a set of point clouds, the best option is the Point Cloud Library (PCL, http://www.pointclouds.org) combined with the OpenCV Library (www.opencv.org). These large-scale, open source projects for 2D/3D point cloud processing and computer vision, are used by a ROS (Robotics Operating System) library called stereo_image_proc.

This package offers a node that converts two stereo frames to a 3D Point Cloud. In order to do so, the node has an inner pipeline (using ROS nodelets) with several stages. Firstly, a monochrome version of the image is produced (debayer stage).

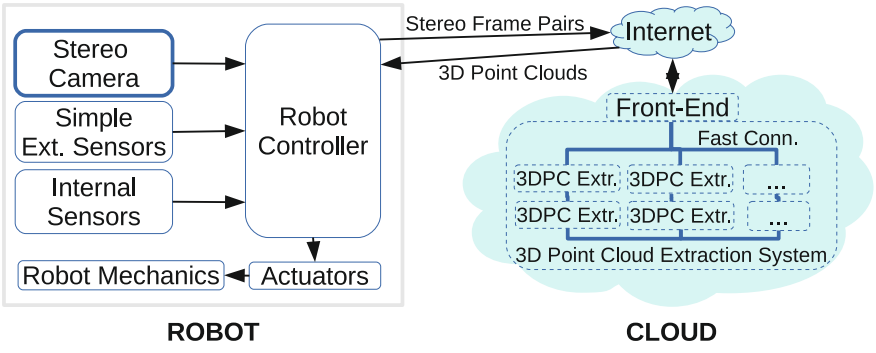


Fig. 2 Block diagram of the 3DPC extraction platform

Secondly, using the stereo camera intrinsic matrices, a rectified version of the image is produced (rectify stage). With the rectified frames, the image matching occurs, obtaining a disparity map (disparity stage). Finally, with this information, the fourth and last step is the 3D point cloud construction (3DPC stage). Due to the very different processing times of the four steps, the minimum time for processing a frame pair will be the maximum of all step times (usually the disparity stage, which lasts most of the whole processing time).

As it can be seen, the aforementioned process cannot be parallelized, as the steps need to be done in order. However, one of the objectives outlined in Sect. 1 is to have a dynamically scalable platform. In order to do so we have exploited the frame pair-level parallelism. Figure 3 depicts the parallel solution. The 3DPC extraction pipeline (stereo_image_proc) is replicated in several virtual instances in the cloud. Therefore, each stereo frame pair will be sent to a different virtual machine in a round-robin fashion. This solution requires an intermediate front-end node, responsible of scattering the stereo stream between the available 3DPC extraction nodes. This way, should the need faster 3DPC extraction times, then more virtual instances could be spun up. However, some extra considerations must be taken into account in order to exploit the parallelism successfully. These considerations are thoroughly explained in Sect. 6.3.

Nevertheless, if we want our system to be dynamically scalable, then it must be able to adapt itself at runtime. Therefore, the buffer node must be able to know how many virtual instances are alive at any moment. This feature is implemented thanks

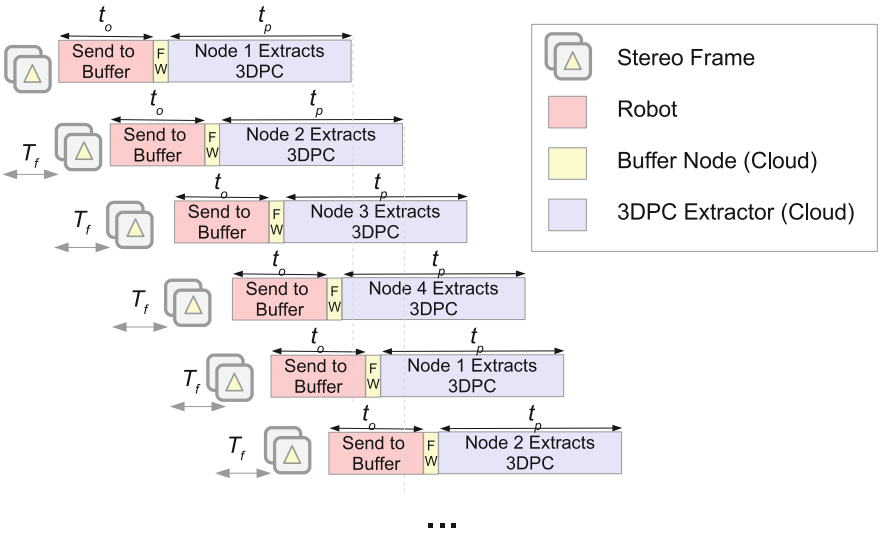


Fig. 3 Stereo frame pipeline process. Four nodes process (in a pipeline fashion) the frame pairs that the front-end node delivers in a round-robin form. T_f is the robot's stereo camera frequency, t_o is the time required to send the frame and t_p is the time required to obtain the 3DPC (see Sect. 5.2 for more information on the involved times). The time required to forward the frame from the buffer node to the 3DPC extractors, thanks to the Gigabit Ethernet connection, is negligible

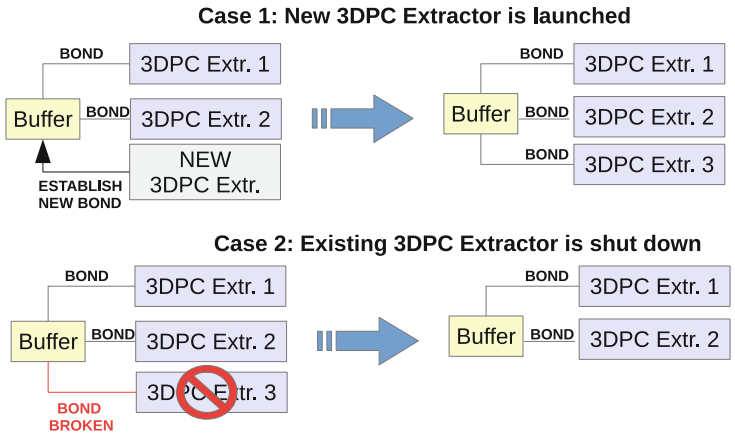


Fig. 4 Dynamic adaptation of the platform when the number of 3DPC extractors changes

to ROS bond library. This library helps to establish a link between the intermediate buffer node and a 3DPC extractor. So, if one of the nodes of the link disappears, the other would be automatically notified. Figure 4 shows the different cases:

- A new 3DPC Extractor is added at runtime: as soon as a 3DPC Extractor node is spun up, it will automatically contact the buffer node and establish a bond between them. The buffer node, with this new link added, will add this new node to the round-robin list.
- One existing 3DPC Extractor is shut down: if this occurs, then the bond would be broken, and the buffer node would be immediately informed. Therefore, the round-robin list would be updated.

When dealing with dynamically scalable solutions, another question arises: When should the platform launch more 3DPC extractors or shut down virtual instances? For vision processing applications a simple Quality-of-Service (QoS) magnitude can serve to determine these actions (with a previous agreement between the robot and the platform). For instance: if the robot has agreed a 3DPC reception frequency of 5 Hz and the cloud is under-providing, then it can scale out to satisfy its demands. The same could be applied for over-providing. The user could change this QoS agreement at any time, and the platform would have to apply it accordingly.

For more technical details of the platform, the code is available with GPL license in GitHub.[1] In addition to this, there are cloud images ready for deployment using the newest cloud technologies: Amazon EC2 virtual machines [2] and Docker containers.[3]

[1] https://github.com/javsalgar/cloud_3dpc_extractor.

[2] The EC2 AMI is ami-893b11b9.

[3] https://registry.hub.docker.com/u/javsalgar/cloud_3dpc_extractor/.

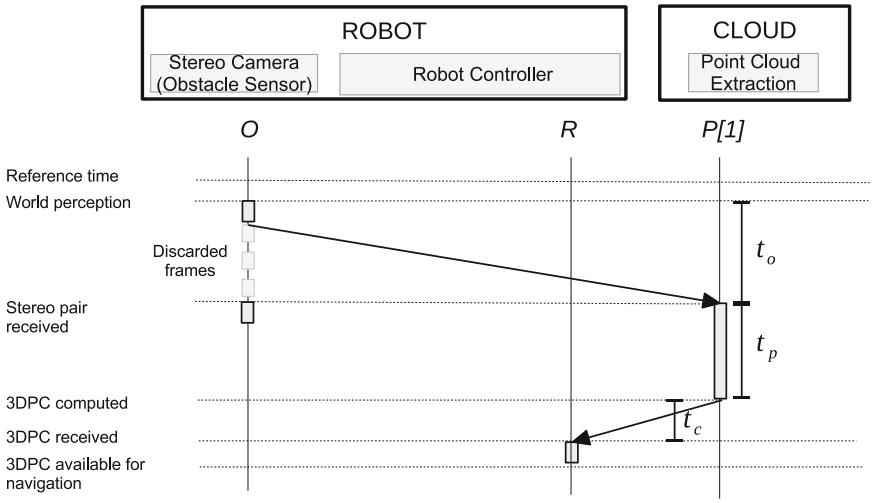


Fig. 5 Time diagram of the point cloud extraction for one virtual node. More virtual nodes suppose that more processes P will be running in parallel with different frames, so a little number of frames would be discarded

5.2 Time Analysis

Processes are running in different machines that are interconnected via standard network protocols, due that the system is running over the Robotic Software Framework ROS [17]. Figure 5 shows a simplified timing diagram for one virtual node (the front-end node is not shown because its delay times are negligible with respect to the other involved times). The two physical systems are shown in the upper part of the figure: the robot and the cloud, each one containing the different logical nodes of the system. As seen in Fig. 2, the robot comprises the stereo camera O and the robot controller R (responsible of tasks such as motor actuation subsystem, motion planner, amongst others). Here, R only contains a reception process that validates the point clouds and do the timing calculation. On the other side, the cloud is running the point cloud extractor P, which can be cloned in several virtual nodes.

Pairs of frames are continuously sent from camera node O to the cloud at a specified frequency. The transmission time from O to P is t_o. Each virtual node receives frames in a round robin fashion (in the figure only one node $P[1]$ is represented for simplicity). $P[1]$ extracts the 3DPC, being t_p the time invested.

This new extracted 3DPC is sent back to the robot R. If a new stereo pair joins the stereo_image_proc inner pipeline (explained in Sect. 5) and enters a stage that is still busy processing a previous frame, this new stereo pair will be discarded. Therefore, when processing times of P (t_p in Fig. 5) are elevated (more than the period of O), some frames are discarded (shown like clear rectangles in Fig. 5) until the point cloud extractors are idle again.

One of the crucial points in the timing analysis is the determination of the number of nodes that the cloud computer dedicates to the image processing (t_p in Fig. 5), in order to assure that the mean time to process a pair of frames is less than the transfer time (t_o in Fig. 5). To get rid of this issue, and due that a scalable cloud computer is available, this number is overestimated, so $t_p/p < t_o$ is always guaranteed (where p is the number of active nodes).

A common issue in distributed systems is the synchronization of the different processes and platforms. For the present experiments a simple solution has been carried out: a ping-pong messaging loop is executed between the cloud and the robot. In spite of non-deterministic TCP protocols, an offset under 0.03 s is always achieved, which is enough for our purposes as total computing latencies (see Sect. 6) are always above 0.2 s. Of course a better synchronization will be reached by using TDMA methods or by the incorporation of an external sync device to each platform.

6 Experimental Results

In this section, a set of experiments is described to do an intensive performance testing for different stereo streams, cloud states and connection technologies (between the robot and the cloud). Our cloud-based solution has been deployed in a private small cluster of 5 physical nodes (1 front-end node and 4 computing nodes). Each node has a AMD Phenom 965 × 4 CPU (with virtual extensions enabled) and 8 GB of RAM. They are all connected using Gigabit Ethernet bandwidth and Openstack Havana is the cloud middleware installed (other well known solutions such as Hadoop were not suitable as we are working with real time systems).

6.1 Scalability of the Platform

The first of the tests is a demonstration that the scalability of our solution is working properly. Thus we use high resolution images (1920 × 1080 pixels) that result into large 3DPC processing times. In order to isolate this experiment from other delaying factors, the test is carried out with offline video images, and the robot is emulated using an Intel Core i7 4750-HQ laptop with 16 GB of RAM. Moreover, the fastest available TCP network (Gigabit) is used to reduce transmission delay overheads. A variable number of frames is sent to the cloud, which processes them and sends a 3DPC back to the emulated robot.

As Gigabit Ethernet is a possible scenario for static robots, this experiment serves also as a reference of the number of virtual nodes needed to extract 3DPC for high resolution images.

Needless to say that, for low resolution images, 3DPC computation is sufficiently fast, so elevated frequencies are obtained for any p (number of virtual computing cloud nodes). The performance test shown in Table 1 measures the time required for

Table 1 Total times to process and receive n point clouds using p 3DPC extractors

Execution time (s)							
p/n	32	64	128	256	512	1024	2048
1	53.97	96.76	173	331	632	1213	2494
2	32.5	48.47	91.71	178	318	629	1218
4	25.38	33.84	55.09	106	186	437	904
6	27.07	36.66	51.48	87.02	171.3	351	635

Resolution of the stereo pairs is 1920×1080

the emulated robot to receive n point clouds processed in p nodes for HD 1080i video stream frames. A significant speedup (ratio between total time for 1 node and for p nodes) is obtained, approaching a sustained average frequency near to 4 frame pairs per second (reached when a high number of frames are processed). In this case, cloud elasticity makes it possible for the robot to change between different computing resources depending on the frequency required by the robot.

6.2 Communication Technology Performance Measures

Once the scalability of the cloud computing solution has been demonstrated, a second experiment is devised to analyze the performance impact of different communication technologies. As stated before, Gigabit Ethernet is a possible scenario for static robots, but the case of mobile robots (where the use of wireless technologies is practically mandatory) must be taken into account as well.

With this in mind, we have tested two wireless technologies: IEEE 802.11n WiFi and IEEE 802.11ac WiFi. The latter, though being still quite recent, can theoretically achieve bandwidths of 768 Mbps (which is close to what Gigabit Ethernet can offer). In this experiment, we have used the on-board computer of the Videre Erratic robot. the stereo camera and the image transmission has been carried out by a real mobile robot (in this case Videre Erratic by LLC). The cloud is configured to have $p = 6$ 3DPC Extractors.

Table 2 compares the performance of the system (in terms of 3DPC reception frequency) using different technologies and frame resolutions. Two facts can be deduced from these results. First of all, for the case of extracting 3DPC for small resolution frames, no performance boost has been found. This is due to two factors: the robot's hardware hardware is powerful enough (for simpler robots, cloud offloading of this process may be beneficial), and insufficient bandwidth of the networks used (better results could have been found for 10 Gbps Ethernet or Infiniband).

However, the robot starts performing worse (due to its hardware limitations) when the quality of the frames is increased. Hence we are able to obtain speed-ups when offloading this demanding computations.

Table 2 Performance measures for different communication technologies

Average Frequency of 3DPC reception (Hz)				
	320 × 240	640 × 480	1024 × 768	1920 × 1080
Gigabit	16.3	6.65	2.22	0.84
WiFi 11n	4.04	2.04	0.29	0.14
WiFi 11ac	4.98	3.02	0.76	0.24
Erratic alone	7.15	2.61	1.01	0.02

Erratic alone means that the Erratic robot is working alone, that is, working as a local stereo vision system

Note that there are performance differences between the Erratic robot and the laptop used in Sect. 6.1. To begin with, the laptop's hardware (both RAM and CPU) is 4 times better than that of Erratic's. Moreover, there are extra factors that affect the overall performance (even though the robot's controller has less to compute because of the offloading), such as frame buffering and sending, peer to peer connection management, 3DPC reception, amongst others.

This experiment (together with the one explained in Sect. 6.4) shows the current limitations of wireless technologies due to the big amount of data to transfer. In order to address this (as explained in Sect. 4) the computation versus communication trade-off must be carefully analyzed for each application case (as done in Sect. 6.5).

6.3 Time Delay Measures

Very delayed data is usually useless for most information processing applications, especially those with near real-time requirements. Taking into account the timing explained in Sect. 5.2, in this third experiment the average latencies to receive the 3DPC of each individual frame are obtained. Each latency is defined here as the time passed since the source stereo frame was actually obtained to the 3DPC reception.

Table 3 shows the latencies obtained (using 1024768 resolution frames) for different communication technologies. The last row shows the same times for Erratic robot computing all the process on its own (no network is used). Once again, these times can serve as a reference to show the viability of cloud computing.

Table 3 Average delay measures for Gigabit Ethernet in the case of 1024 × 768 resolution frames

Delay times (s)				
	t_o	t_p	t_c	Total
Gigabit	0.0704	0.389	0.317	0.7764
WiFi 11n	0.676		2.377	3.442
WiFi 11ac	0.4740		1.61	2.473
Erratic alone	0.0670	0.966	0.166	1.199

Erratic alone means that the Erratic robot is working alone, that is, working as a local stereo vision system

In order to calculate the delay, the average times taken to perform each of the stages explained in Sect. 5.2 (t_o, t_p and t_c) are measured using time stamps at the beginning of every process. The average latencies are calculated by adding the mean runtime of all these stages. In the case of using the cloud, the difference between technologies can be found in the transfer times t_o and t_c, whereas t_p remains unaffected (Fig. 2 clarifies this statement).

As it can be seen, for lower resolutions the robot can outperform the Cloud if wireless technologies are used. However, when increasing the stereo frame resolution, there is a point where the limitations of the embedded hardware start to arise. Therefore, it is worth considering Cloud offloading when bigger resolutions are required.

6.4 Interference Analysis with WiFi AC

The performance of the Cloud itself is not crucial, as we have the premise of "infinite resources". However, as wireless technology is the best choice for mobile robots, an in-depth analysis of interference when increasing the number of robots must be done. It is highly likely that not only one robot, but several are using the cloud at the same time. Hence it is extremely important to study the possible communication quality degradation.

The aim of this experiment is to analyze how WiFi 11ac manages the interferences, and to prove that it is the most suitable technology for mobile robots operation. We will focus only in the transmission of stereo frame pairs (resolution of 320×240) to the Cloud, what renders enough information about interference problems. We are interested in two elements:

- The average transfer time needed to send a stereo frame pair to the Cloud. As we are working with a real-time system, the meeting of certain deadlines is vital. For example, if the robot is transmitting stereo frames at 5 Hz, transfer times lower than $1/5\,\text{Hz} = 0.2\,\text{s}$ are desired in order to meet deadlines.
- The message success rate. When more robots are added, there is the risk that some of the packets that form the message (containing a stereo frame pair) collide and get corrupted. Even though transport-level protocols like TCP allow packet resending, the following scenario could occur: while the Cloud waits for the missing packet (which corresponds to a stereo frame message with timestamp t) to be resent, the same robot had already begun sending packets of a new stereo frame message (that is, a frame with timestamp $t + 1$). Should a packet from a frame with timestamp $t + 1$ arrive, then all the packets from messages with a timestamp lower than $t + 1$ would be automatically discarded (because of its obsolescence). This necessary implies a lower message success ratio.

Table 4 compares the average latency and message success ratio when the number of robots and the message frequency increase. To begin with, note that the packet success rate works exactly as expected. When more robots are added, the number of

Table 4 Performance comparison when adding more robots in the case of 320 × 240 when no 3DPC extraction is done and only delays in stereo frame transmissions are considered

	# Robots	Mean transfer time (s)	Average message success (%)
5 Hz	1	0.117	100.00
	2	0.124	100.00
	4	0.157	94.99
	6	0.147	87.88
10 Hz	1	0.063	100.00
	2	0.086	99.86
	4	0.082	94.99
	6	0.084	87.79

The wireless technology is that of 802.11ac

packet collisions increases, and therefore more messages are lost. Thus, there is a trade-off between number of robots and system stability (e.g., missing environment information can result in a robot crash, in the context of robot navigation). The average transfer time deteriorates until a point where the percentage of message success is low enough. This phenomenon is understood because the mean transfer times are calculated only with those packets that have arrived successfully. That is, those "lucky" packets last little time to complete. Hence, it is not that messages are faster now, but that more packets do never arrive to the Cloud (and therefore their transfer time cannot be properly measured). Therefore, we can assure that there is another trade-off between message success ratio and average transfer time.

With respect to the meeting of deadlines, Fig. 6 shows the Empirical Cumulative Distribution Frequency (ECDF) of delays for the experiment above shown. As it can be seen, adding more robots make it more difficult to meet deadlines (vertical gray dashed line in the figures) because of network interference. This is especially evident in the case of 10 Hz. Therefore, because of the trade-offs previously explained, we can conclude that current wireless technologies are (at the moment) not enough developed for very critical real-time applications when more than one robot in the same wireless cell. Should this be the scenario, then it would be necessary to allow less strict deadlines. The high variability of total latency times that occurs in our experiments can be mitigated by a predictive timing correction of actuation signals [13]. There will be necessary further improvements in 802.11ac MAC layer like TDMA protocols to reduce this latency variance (as mentioned in Sect. 5.2). The use of the Contention Free Period with fixed size packets is an alternative to TDMA protocols. This could guarantee a minimum bandwidth reservation, and therefore we could address the issues explained in this experiment.

6.5 Application Case: Navigation Assistant

This last test summarizes the results for a real task for our cloud vision platform: a navigation assistant for mobile robots. While a teleoperator is driving a mobile robot, the information processed by the 3DPC Extraction Platform helps him/her

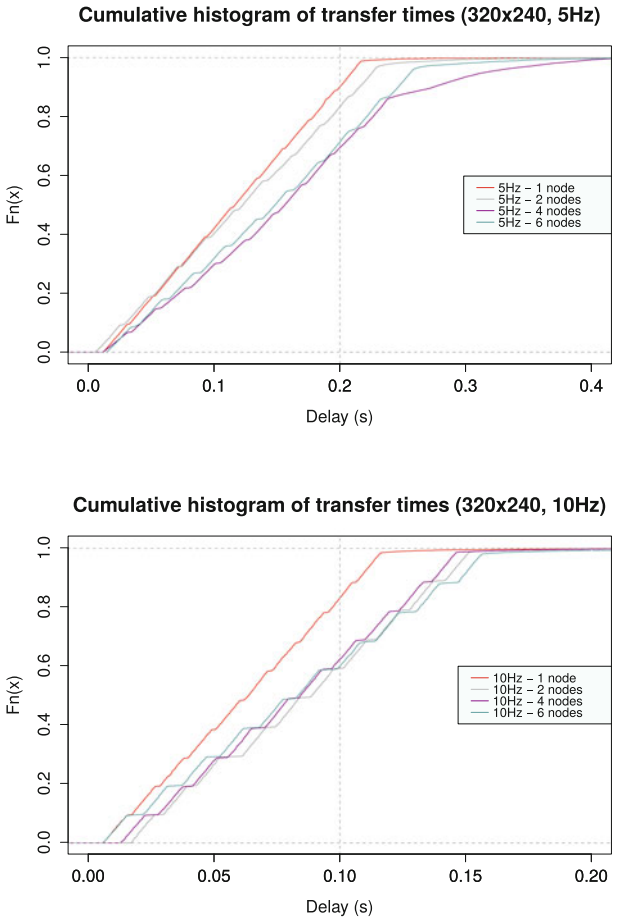


Fig. 6 Empirical cumulative distribution frequency of delays with 5 and 10 Hz

to avoid collisions. Numerous questions arise, as in any real experiment: are really high quality stereo frames necessary to assist in the navigation?, are cloud solution more effective than the on-board one?, do packet latency variability suppose a problem when navigating? If on-board navigation were successful enough for 320×240 stereo frames (which have an adequate 3DPC frequency rate, see Table 2), then Cloud offloading would not be necessary at all. In order to answer these questions, a testing circuit was prepared (see Fig. 7). The Erratic robot was equipped with a stereo camera built from two PSEye cameras and the circuit was completed several times. The ratio of collisions by maneuvers was used as a success magnitude.

The results obtained in Sects. 6.3 show that most of the delays come from t_c, that is, the time required to transfer the 3DPC back to the robot. Thus, we got rid of this communication overhead by moving the navigation assistant to the Cloud as well. Figure 8 shows the diagram of the resulting architecture.

Fig. 7 Testing circuit used in the experiments

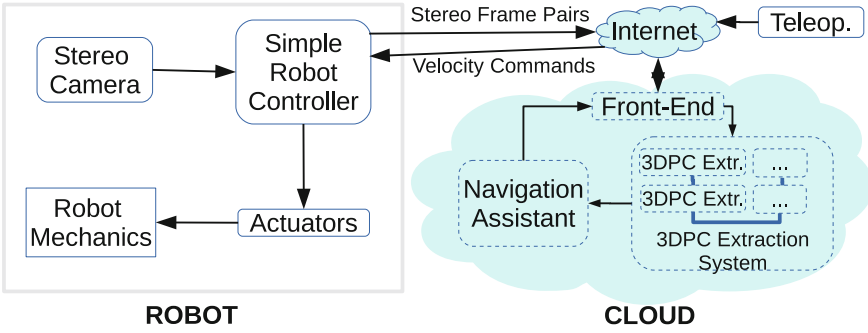


Fig. 8 Overview of the platform applied to the navigation use case

Thanks to this change in the offloading model, we obtained the following results, which solve most of previous questions. First of all, collisions were very frequent (about 50 %) when using 320 × 240 images (for any computing option). Secondly, the number of collisions were considerably reduced (less than 10 %) with higher resolutions (640 × 480 pixels) and using the Cloud. As a conclusion, for the stereo vision algorithm used here, low resolution images are not enough to detect the obstacle information properly, and hence using higher resolution images is justified. Moreover, images with more than 320 × 240 are more difficult for the Erratic robot to process on-board. A demonstration video can be found in [6] and all the details of the experiments and the navigation assistant can be seen in [19].

7 Conclusions and Lessons Learned

The implemented platform (and its experimental results) shows that the cloud-based offloading of heavy visual processing tasks is possible. Several conclusions can be extracted from the experience.

Firstly, the main bottleneck of cloud offloading is due to communication overheads. It is extremely important to mitigate this effect by choosing the correct network technology. Moreover, the non-real time middleware and the inherent non-deterministic of the TCP protocol (available in most Robotics Software Frameworks) introduce a high variability in timing latency. Thus, this drawback should be mitigated by using some kind of predictive correction terms in the loop controller and more deterministic middleware and networks. However, the results obtained by WiFi 11ac are promising, and in future years it may be able to provide bandwidths close to its theoretical 768 Mbps, which may reduce the collision problem that currently appears even for a reduced number of robots (as it has been outlined in Sect. 6.4).

Secondly, there is an inherent trade-off between computation offloading and communication overhead times. Therefore, the platform should be used finding the best balance between those two. In that sense, depending on the use case, it may be worth considering offloading not only the 3DPC extraction, but also other robotics tasks (just like the case shown in Sect. 6.5).

In addition to all this, it has also been demonstrated that if a Cloud Solution is not scalable, it is highly unlikely that good performance results can be achieved, and therefore impossible to meet real-time requirements. As it has been shown in Sect. 6.1, this is an indispensable element to exploit the Cloud's true potential.

As a final conclusion, it can be assured that, despite there are challenges that need to be addressed, the main question has been answered: using the Cloud for offloading can imply better performance results in a robot than using on-board computation (at least for a typical robot, whose hardware is much limited).

Acknowledgments The work shown in this chapter has been supported by the Spanish grant (with support from the European Regional Development Fund) BIOSENSE (TEC2012-37868-C04-02/01) and by Andalusian Regional Excellence Research Project grant (with support from the European Regional Development Fund) MINERVA (P12-TIC-1300). We wish to thank also Prof. D. Cascado for his interesting comments.

References

1. Agostinho, L., Olivi, L., Feliciano, G., Paolieri, F., Rodrigues, D., Cardozo, E., Guimaraes, E.: A cloud computing environment for supporting networked robotics applications. In: 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC), pp. 1110–1116 (2011). doi:10.1109/DASC.2011.181
2. Arumugam, R., Enti, V., Bingbing, L., Xiaojun, W., Baskaran, K., Kong, F.F., Kumar, A., Meng, K.D., Kit, G.W.: DAvinCi: a cloud computing framework for service robots. In: 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 3084–3089 (2010). doi:10.1109/ROBOT.2010.5509469
3. Bistry, H., Zhang, J.: A cloud computing approach to complex robot vision tasks using smart camera systems. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3195–3200 (2010). doi:10.1109/IROS.2010.5653660
4. Buyya, R., Vecchiola, C., Selvi, S.T.: Mastering Cloud Computing Foundations and Applications Programming. Morgan Kaufmann/Elsevier, Amsterdam (2013)

5. Charfi, E., Chaari, L., Kamoun, L.: PHY/MAC enhancements and QoS mechanisms for very high throughput WLANs: a survey. IEEE Commun. Surv. Tutor. **15**(4), 1714–1735 (2013). doi:10.1109/SURV.2013.013013.00084
6. Cloud-based robot navigation assistant using stereo image processing. http://www.rtc.us.es/cloud-based-robot-navigation-assistant-using-stereo-image-processing/ (2014). Accessed 12 Nov 2014
7. Furht, B., Escalante, A. (eds.): Handbook of Cloud Computing. Springer, New York (2010)
8. Gouveia, B.D., Portugal, D., Silva, D.C., Marques, L.: Computation sharing in distributed robotic systems: a case study on SLAM. IEEE Trans. Autom. Sci. Eng. (T-ASE) **12**(2), 410–422 (2015)
9. Guizzo, E.: Robots with their heads in the clouds. IEEE Spectr. **48**(3), 16–18 (2011). doi:10.1109/MSPEC.2011.5719709
10. Handa, A., Newcombe, R.A., Angeli, A., Davison, A.J.: Real-time camera tracking: when is high frame-rate best? In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) Computer Vision ECCV 2012. Lecture Notes in Computer Science, vol. 7578, pp. 222–235. Springer, Berlin (2012)
11. Hennessy, J.L., Patterson, D.A.: Computer Architecture: A Quantitative Approach. Morgan Kaufmann, Waltham (2012)
12. Iñigo-Blasco, P., Diaz-del Rio, F., Romero-Ternero, M.C., Cagigas-Muñiz, D., Vicente-Diaz, S.: Robotics software frameworks for multi-agent robotic systems development. Robot. Auton. Syst. **60**(6), 803–821 (2012). doi:10.1016/j.robot.2012.02.004. http://www.sciencedirect.com/science/article/pii/S0921889012000322
13. Iñigo-Blasco, P., Diaz-del Rio, F., Vicente-Diaz, S., Cagigas-Muñiz, D.: The shared control dynamic window approach for non-holonomic semi-autonomous robots. In: Proceedings of 41st International Symposium on Robotics. ISR/Robotik 2014. Munich (2014)
14. John, B.P.: Effectiveness of SPEC CPU2006 and Multimedia Applications on Intel's Single. Dual and Quad Core Processors. ProQuest (2009)
15. Kytö, M., Nuutinen, M., Pirkko, O.: Method for measuring stereo camera depth accuracy based on stereoscopic vision. In: Proceedings of SPIE/IS&T Electronic Imaging 2011 (2011)
16. Nimmagadda, Y., Kumar, K., Lu, Y.H., Lee, C.S.G.: Real-time moving object recognition and tracking using computation offloading. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2449–2455 (2010). doi:10.1109/IROBlS.2010.5650303
17. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.: ROS: an open-source robot operating system (2009)
18. Riazuelo, L., Civera, J., Montiel, J.M.M.: C2tam: a cloud framework for cooperative tracking and mapping. Robot. Auton. Syst. **62**(4), 401–413 (2014). doi:10.1016/j.robot.2013.11.007
19. Salmeron-Garcia, J., Iñigo Blasco, P., Diaz-del Rio, F., Cagigas-Muniz, D.: A trade-off analysis of a cloud-based robot navigation assistant using stereo image processing. IEEE Trans. Autom. Sci. Eng. (T-ASE) **12**(2), 444–454 (2015)
20. Srinivasan, S.: Cloud Computing Basics. Springer Briefs in Electrical and Computer Engineering. Springer, New York (2014)
21. Stallings, W.: Data and Computer Communications. Always Learning, 10th edn. Pearson, Boston (2014)
22. Szeliski, R.: Computer Vision Algorithms and Applications. Texts in Computer Science. Springer, London (2011)
23. Waibel, M., Beetz, M., Civera, J., D'Andrea, R., Elfring, J., Galvez-Lopez, D., Haussermann, K., Janssen, R., Montiel, J.M.M., Perzylo, A., Schiessle, B., Tenorth, M., Zweigle, O., van de Molengraft, R.: RoboEarth. IEEE Robot. Autom. Mag. **18**(2), 69–82 (2011). doi:10.1109/MRA.2011.941632
24. Wu, H., Lou, L., Chen, C.C., Hirche, S., Kuhnlenz, K.: Cloud-based networked visual servo control. IEEE Trans. Ind. Electron. **60**(2), 554–566 (2013). doi:10.1109/TIE.2012.2186775

Part II
Cloud for the IoT

Architecting the Internet of Things: State of the Art

Mohammed Riyadh Abdmeziem, Djamel Tandjaoui and Imed Romdhani

Abstract Internet of things (IoT) constitutes one of the most important technological development in the last decade. It has the potential to deeply affect our life style. However, its success relies greatly on a well-defined architecture that will provide scalable, dynamic, and secure basement to its deployment. In fact, several challenges stand between the conceptual idea of IoT, and the full deployment of its applications into our daily life. IoT deployment is closely related to the establishment of a standard architecture. This architecture should support future extensions, and covers IoT characteristics such as distributivity, interoperability, and scalability. A well defined, scalable, backward compatible, and secure architecture is required to bring the IoT concept closer to reality. In the literature, several architectures have been proposed. Nevertheless, each architecture brings a share of drawbacks, and fails covering all IoT characteristics. In this chapter, we review the main proposed architectures for the Internet of Things, highlighting their adequacy with respect to IoT requirements. Firstly, we present IoT building blocks. Then, we introduce the high level architecture of IoT before diving into the details of each proposed architecture. In addition, we introduce a classification of the reviewed architectures based on their technical aspects, and their ability to match IoT characteristics. Finally, based on the main shortcomings of the proposed architectures, we conclude with some design ideas for shaping the future IoT.

M.R. Abdmeziem (✉)
LSI, USTHB: University of Sciences and Technology Houari Boumedienne, BP 32,
El Alia Bab Ezzouar, Algiers, Algeria
e-mail: rabdmeziem@usthb.dz

D. Tandjaoui
CERIST: Center for Research on Scientific and Technical Information,
03, Rue des Freres Aissou, Ben Aknoun, Algiers, Algeria
e-mail: dtandjaoui@mail.cerist.dz

I. Romdhani
School of Computing, Edinburgh Napier University, 10, Colinton Road,
Edinburgh EH10 5DT, UK
e-mail: I.Romdhani@napier.ac.uk

A. Koubaa and E. Shakshuki (eds.), *Robots and Sensor Clouds*,
Studies in Systems, Decision and Control 36,
DOI 10.1007/978-3-319-22168-7_3

Keywords Internet of things · Architecture · Middleware · RFID · WSN · Sensor cloud · Robotic cloud

1 Introduction

Internet of Things (IoT) is one of the main communication development in recent years. It makes our everyday objects (e.g. health sensors, industrial equipements, vehicules, clothes, etc.) connected to each other and to the Internet. According to [1], the basic concept behind IoT is the pervasive presence around us of various wireless technologies such as Radio-Frequency IDentification (RFID) tags, sensors, actuators and mobile phones, in which computing and communication systems are seamlessly embedded. Through unique addressing schemes, these objects interact with each other, and cooperate to reach common goals. In fact, this interconnection allows the objects surrounding us to share data, to interact, and to act autonomously on behalf of their users. This prospect opens new doors toward a future, where the real and virtual world merge seamlessly through the massive deployment of embedded devices. These latter enhance dumb objects with computational, communication and storage capabilities. By enabling interactions with and among smart objects, IoT has the potential to add a new dimension in the communication sector. In addition, technology advances coupled with users need will encourage the wide spread deployment of IoT's applications. These applications would deeply affect our corporations, communities, and personal lives. In fact, enabling the objects in our everyday environment to possibly communicate with each other, and process the gathered information will open wide horizons for unpredicted applications [2].

From the perspective of a private use, e-health is one of the most interesting applications. In fact, it provides medical monitoring to millions of elderly and disabled patients while preserving their autonomy and comfort anywhere. For instance, using sensors planted in or around a patient, physiological data is gathered and transmitted to qualified medical staff that can intervene in case of an emergency. At home, energy management could be improved through the control of home equipments such as air conditioners, refrigerators, washing machines, etc. An other illustration of IoT applications in the personal sphere relies on social networking paradigm. Indeed, an interesting development would be using a Twitter like concept. In this concept, various objects in the house can periodically tweet the readings, which can be easily followed from anywhere [3]. From the perspective of business use, environmental monitoring can be achieved by keeping track of the number of occupants, and by managing the utilities within a building. Supply chains could also benefits from the introduction of RFID and NFC (Near Field Communication) devices. As a result, real-time and precise data on the inventory of finished goods could be gathered. In addition, from the perspective of utility services, smart grids are one of the most interesting applications. Using these applications, efficient energy consumption can be achieved through continuous monitoring of electric consumption. Furthermore,

gathered data is used to maintain the load balance within the grid ensuring high quality of service [4].

Several challenges stand between the conceptual idea of IoT and the full deployment of its applications into our daily life. In fact, IoT successful deployment is closely related to the establishment of a standard architecture. This latter should cover IoT characteristics and support future extensions, the same way current Internet architecture achieved during the past forty years. A well defined, scalable, backward compatible, and secure architecture is required to bring the IoT concept closer to reality. In the literature, several architectures have been proposed [5–11]. Nevertheless, each architecture brings a share of drawbacks, and fails covering all IoT characteristics. These characteristics can be summarized as follows:

- **Distributivity**: IoT will likely evolve in a highly distributed environment. In fact, data might be gathered from different sources and processed by several entities in a distributed manner.
- **Interoperability**: Devices from different vendors will have to cooperate in order to achieve common goals. In addition, systems and protocols will have to be designed in a way that allows objects (devices) from different manufacturers to exchange data and work in an interoperable way.
- **Scalability**: In IoT, billions of objects are expected to be part of the network. Thus, systems and applications that run on top of them will have to manage this unprecedent amount of generated data.
- **Resources scarcity**: Both power and computation resources will be highly scarce.
- **Security**: User's feelings of helplessness and being under some unknown external control could seriously hinder IoT's deployment.

In this chapter, we review the main proposed architectures for the Internet of Things, highlighting their adequacy with respect to IoT requirements. We introduce the enabling technologies that are expected to form the building blocks of the IoT in Sect. 2. In Sect. 3, we discuss in detail and classify the different proposed architecture for the IoT gathered into two categories, clean slate architectures and tailored architectures. In Sect. 4, we provide an in-depth analysis of the proposed architectures based on their technical aspect and their ability to match IoT characteristics. Section 5 concludes the chapter.

2 IoT Building Blocks

Instead of emerging as a completely new category of systems, the Internet of Things is likely to rise through an incremental development approach. In fact, in order to reach the physical realm, IoT building blocks will be progressively integrated to the existing Internet. In this section, we focus on the enabling technologies that are expected to form the IoT building blocks. Each technology is briefly introduced, along with its future impact on IoT. The different technologies are classified into three categories.

- The sensing technologies through which the required data is gathered.
- The middleware layer that is in charge of processing and managing the obtained raw data. It provides an abstraction level to users and developers.
- The actuating technologies that represent the physical extension of IoT applications.

As a result, IoT would not only provide a digital support but also a physical one that can directly affect our real world. In the following, we briefly introduce the building blocks of each category.

2.1 Sensing

In the IoT, wireless technologies will play a central role in data harvesting and data communication. In fact, the major part of data traffic between objects will be carried through wireless media. Wireless Sensor Networks (WSN) and radio-frequency identification (RFID) are considered as the two main building blocks of sensing and communication technologies for IoT [2]. Indeed, their ability of sensing the environment and self-organizing into ad hoc networks represent an important feature from the IoT perspective. Nevertheless, these technologies suffer from different constraints (e.g. energy limitation, reliability of wireless medium, security and privacy, etc.). In particular, the scarcity of energy resources available in the embedded devices is a sensitive issue. Consequently, to increase energy efficiency, a number of solutions have been introduced in the literature. For instance, lightweight MAC protocols [12], energy efficient routing protocols [13], and tailored security protocols [14] have been proposed to mitigate the impact of resources scarcity on sensing technologies. Still, their limited autonomy remains a considerable obstacle to their widespread deployment into our daily lives. Besides, the future objects, enhanced with sensing capabilities, are expected to share a set of common characteristics and functionalities. Indeed, these objects will have to properly manage heterogeneity in order to move towards an incremental deployment. In the following, we provide a broad presentation of RFID, WSN, and their integration into the IoT.

RFID technology is considered as an important development in the embedded devices field. In fact, RFID allows the design of tiny microchips (called tags), which can be appended to an object of our daily life. As a result, stored data in these tags can automatically be used to identify and extract useful information from the object. Thus, the tag acts as an electronic barcode.

From a hardware perspective (Fig. 1) an RFID tag is a tiny microchip (e.g. 0.4 mm $\times$ 0.4 mm $\times$ 0.15 mm) attached to an antenna, which is used for both receiving the reader signal and transmitting the tag identity. The tag is manufactured in a package that can be used as an adhesive sticker [15].

Generally, RFID devices are classified into two categories: passive and active. The passive RFID tags are not battery powered. In fact, they use the power of the readers interrogation signal to communicate their data. A lot of applications from

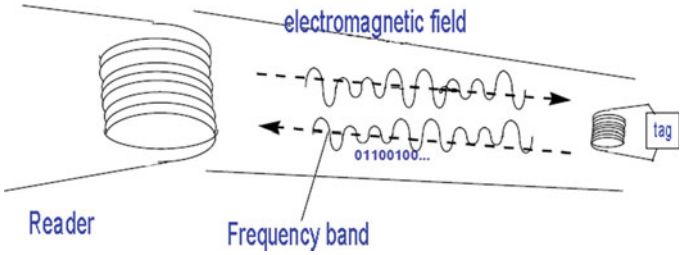


Fig. 1 RFID tag and reader

several fields use this kind of tags. Particularly, in retail, supply chain management, and transportation. They are also used in bank cards and road toll tags as an access control mean. However, the active RFID readers possess their own battery energy, and are able to trigger a communication. Although the radio coverage is more important compared to passive tags, this is obtained at the expense of higher production costs. In fact, one of the most interesting advantage in the use of RFID technology is the limited cost, which would allow a widespread adoption. Among other applications, active RFID tags can be used in port containers for monitoring cargo, robotics in a smart home context, and in hotels to provide automated check-in for customers [16].

Sensor networks on their side will also play a crucial role in the future deployment of IoT. In fact, they can cooperate with RFID systems to better track the status of things (e.g. their location, temperature, movements, etc.). Doing so, WSN are able to augment their awareness of the environment. Hence, they act as a further bridge between the physical and the digital world.

Sensor networks consist of a certain number, which can be very high, of sensing nodes communicating in a wireless multi-hop fashion (Fig. 2). In general, nodes

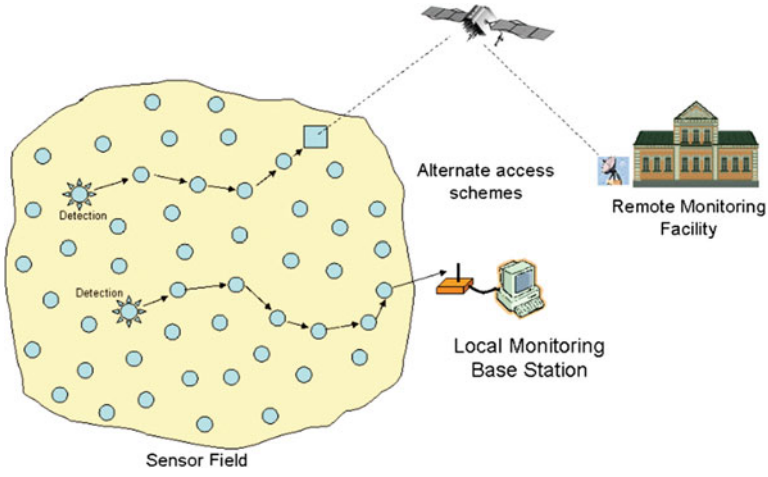


Fig. 2 Wireless sensor network

report their sensing results to a small number of special nodes called sinks (or base stations). A lot of effort has been undertaken by the scientific community on sensor networks. Indeed, many work have addressed several problems at the different layers of the protocol stack. In these works, the main issues concern energy efficiency (which is a limited resource in WSN), scalability (the number of nodes can rise significantly), reliability (the system might be involved in critical applications), and robustness (nodes might be subject to failure) [17].

Integration of sensing technologies into passive RFID tags would bring completely new applications into the IoT context. In fact, sensing RFID systems will allow to build RFID sensor networks, which consist of small RFID-based sensing and computing devices. RFID readers would constitute the sinks of data generated by sensing RFID tags. Moreover, they would provide the power for the different network operations. Efficiently networking tag readers with RFID sensors would allow real-time queries on the physical world. This could lead to better forecasts, new business models, and improved management techniques [18].

2.2 Middleware

The middleware is a software interface between the physical layer (i.e. hardware) and the application one. It provides the required abstraction to hide the heterogeneity and the complexity of the underlying technologies involved in the lower layers. Indeed the middleware is essential to spare both users and developers from the exact knowledge of the heterogeneous set of technologies adopted by the lower layers. It allows the developers to primarily focus on issues related to the designed applications. Hence, it spares these developers losing time and efforts on issues in relation with the management and the utilization of the underlying IoT physical technologies.

The approaches based on service-oriented computing (SOC) could be in charge of playing the middleware role in the context of IoT. A service-oriented architecture (SOA) is a set of communicating services based on standardized interaction models [19]. SOC can be used to manage web services and to make them act like a virtual network. Thus, it adapts the applications to the specific users needs. Besides, Cloud computing [20] is based on a distributed architecture, in which entities are treated in a uniform way and accessed via standard interfaces. Thus, providing a common set of services and an environment for service composition. Actually, combining cloud computing with SOA could provide an efficient middleware for IoT supporting a high level of heterogeneity and flexibility.

The service based approaches lying on a cloud infrastructure open the door toward highly flexible and adaptive middleware for the IoT. For instance, Sensor-Cloud is one of the most interesting design idea to handle the huge amount of sensing devices (see Sect. 2.1), and the unprecedented amount of generated data. In fact, a Sensor-Could infrastructure provides to the end user service instances based on virtual sensors in a automatic way. Actually, the platform offers a virtual feeling to the user as if

these sensors are part of its classical IT resources (e.g. disk storage, CPU, memory, etc.) [10]. The end users do not have to bother with their actual physical location or their actual state. In addition, they do not even have to own the physical sensors. Instead, it is possible to create a set of sensor services to be exploited in different applications for different users through the cloud [21]. Moreover, decoupling the application logic from the embedded devices, and moving it to the cloud will allow developers to provide applications for the heterogeneous devices that will compose the future IoT environment [22].

2.3 Actuating

Internet of Things enhances the dumb objects around us with processing and communication capabilities. Hence, the resulted pervasive applications have the potential to deeply impact our way of life. Indeed the range of domains that might be concerned is impressive. In these domains, solutions might be deployed in both public and private areas. However, bringing to reality the future vision of our societies under the umbrella of IoT can not be achieved by limiting the scope of technology enhancement to cyberspace. In fact, physical support (i.e. actuating) in the real world is definitely required [11].

As an illustration, let us consider an e-health scenario. Indeed, e-health applications are highly promising solutions intending to provide unobtrusive support to frail and elderly people. In particular, these applications might be highly critical in case of a medical emergency. In the following, we present an e-health scenario that highlights the importance of actuating capabilities, in addition to emphasizing the involved IoT building blocks, along with their specific functionalities. Firstly, specialized sensing nodes planted in, or on a patient body are used to collect health-related data (e.g. blood glucose level), plus contextual sensors that gather data such as room temperature and humidity level. Then, gathered data is transmitted to a middleware back-end infrastructure through wireless connexion (e.g. Bluetooth, ZigBee, Wifi). Upon adequate processing, decisions can be made such as alerting medical staff, or family members. To understand the role of actuating devices, we consider the case where a hypoglycemia is detected. If the influence of the system is limited to the digital world, the application would only trigger an alarm. Actually, an hypoglycemia could rapidly engender disastrous consequences to the brain [23]. Thus, a rapid intervention is required. In fact, waiting for emergency teams to arrive might be too late. Consequently, e-health applications have to be enhanced with actuating capabilities through which a decision to provide the patient with sugar (e.g. using an injection) can be executed immediately, probably, saving his life.

Could-Robotics could constitute an ideal candidate to fulfill the role of physical support to IoT applications. In fact, Could-Robotics abstracts robotic functionalities and provides a means for utilizing them. Various equipments and devices that can measure the world or interact with people in both the physical and digital worlds are treated uniformly. Such devices include individual robots, sensors, and smartphones.

These robots are logically gathered to form a cloud of robots by networking. Hence, they realize an integrated system that provides seamless support for daily activities using the available resources on demand [24].

3 The Proposed IoT Architectures and Classification

In this part, we review the proposed architectures in the literature. We start by introducing a high level architecture that is commonly accepted to constitute the basement of the future IoT architecture. Then, we introduce our classification that gathers the approaches into two classes. The first class of approaches is based on existing architectures tailored to the context of IoT. The second one is based on clean slate approaches that propose novel architectures from scratch.

3.1 High Level Architecture

A well defined IoT architecture is still not established. However, a three-layer high level architecture is commonly accepted [25]. This architecture consists of three layers: Perception Layer, Network Layer, and Application layer (Fig. 3). A brief description of each layer is given.

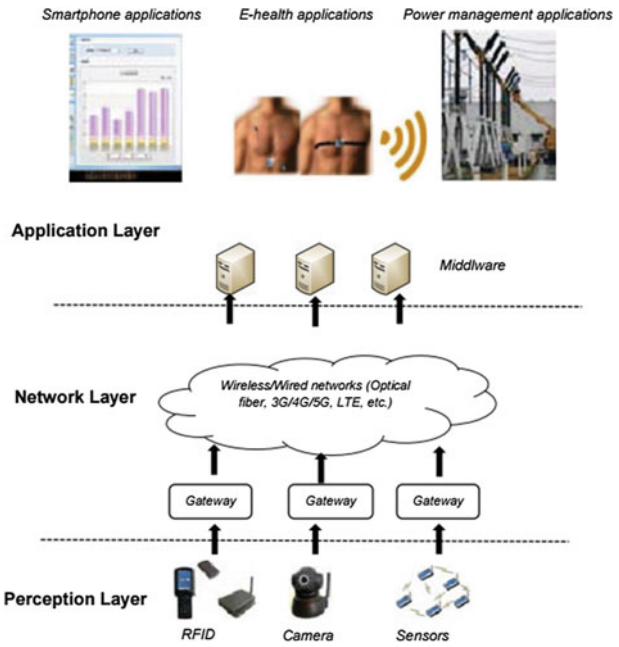


Fig. 3 The three-layer IoT architecture

Perception Layer: The main task of the perception layer is to perceive the physical properties of things around us that are part of the IoT. This process of perception is based on several sensing technologies (e.g. RFID, WSN, GPS, NFC, etc.). In addition, this layer is in charge of converting the information to digital signals, which are more convenient for network transmission. However, some objects might not be perceived directly. Thus, microships will be appended to these objects to enhance them with sensing and even processing capabilities. Indeed, nanotechnologies and embedded intelligence will play a key role in the perception layer. The first one will make chips small enough to be implanted into the objects used in our every day life. The second one will enhance them with processing capabilities that are required by any future applications.

Network Layer: The network layer is responsible for processing the received data from the Perception Layer. In addition, it is in charge of transmitting data to the application layer through various network technologies, such as wireless/wired networks and Local Area Networks (LAN). The main media for transmission include FTTx, 3G/4G, Wifi, bluetooth, Zigbee, UMB, infrared technology, and so on. Huge quantities of data will be carried by the network. Hence, it is crucial to provide a sound middleware to store and process this massive amount of data. To reach this goal, cloud computing is the primary technology in this layer. This technology offers a reliable and dynamic interface through which data could be stored and processed. Indeed, research and development on the processing part is significant for the future development of IoT.

Application Layer: The application layer uses the processed data by the previous Layer. In fact, this layer constitutes the front end of the whole IoT architecture through which IoT potential will be exploited. Moreover, this layer provides the required tools (e.g. actuating devices) for developers to realize the IoT vision. In this vision, the range of possible applications is impressive (e.g. Intelligent transportation, logistics management, identity authentication, location based services, safety, etc.).

To suit IoT specificities, the three-layer architecture provides a high level framework through which different approaches might be implemented. In the following, we present and classify the IoT architectures proposed in the literature, either resulting from public projects, or academic research.

3.2 Tailored Architectures

IETF protocol suite: Given that the protocol suite TCP/IP is recognized as the cornerstone of the current Internet, it is understandable to consider the same protocol stack to be used for IoT deployment [26]. Nevertheless, IoT specificities such as resources scarcity, instable wireless links, and heterogeneity of both traffic and devices, will seriously hinder IP-based protocols deployment in IoT environments. To the end of tailoring the existing TCP/IP architecture to IoT, the Internet Engineering Task Force (IETF) is working on standardizing the corresponding communication protocols for each layer of the communication stack. Namely, IEEE 802.15.4 [27] for

the data link layer, IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) [28] as a lightweight addressing scheme, Routing Protocol for Low Power and Lossy Networks (RPL) [29] as a routing protocol, and Constrained Application Protocol (CoAP) [30] to be adopted in the application layer. In the following, we briefly introduce each protocol.

- *IEEE 802.15.4* is a standard developed by the IEEE 802.15 Personal Area Network (PAN) Working Group. It specifies both physical layer and media access control for wireless constrained devices. Due to its provided features, which aim to be as less resource consuming as possible, several protocols such as WirelessHART [31] and ZigBee are based on the IEEE 802.15.4. In addition, more and more IoT devices are built as IEEE 802.15.4-compliant devices.
- *6LoWPAN* is a standard that aims to transfer IPv6 packets to IEEE 802.15.4 based networks. 6LoWPAN uses IPV6 header compression mechanisms of IPv6 datagrams. Compression mechanisms are motivated by the limited space available in 802.15.4 frames to encapsulate IPv6 packets. 6LoWPAN defines encoding formats for compression based on shared state within contexts. In other words, it takes advantage of the fields that are implicitly known to all nodes in the network or can be deduced from the MAC layer.
- *RPL* is a standardized distance-vector routing protocol designed for constrained IP-based environments. It takes into consideration limitations either in energy power or in computational capabilities of such networks. The protocol organizes a logical representation of the network topology as a Directed Acyclic Graph (DAG). This graph is composed of one or more Destination Oriented DAGs (DODAGs) with one root per DODAG. Each root is typically a border router (BR). This latter establishes an optimum path based on defined routing metrics, which it receives through broadcast messages.
- *CoAP* is an application layer protocol developed by the IETF CoRE Working Group. It is designed for constrained environments. Based on a REST style architecture, the protocol considers the various objects in the network as resources. A Unique Universal Resource Identifier (URI) is assigned to each resource. The protocol uses the corresponding URI to operate the different resources.

SENSEI project: Future networks will be enhanced with ambient intelligence capabilities enabling IoT applications to spread in our environment. To realize this future vision of our communications patterns, heterogeneous wireless sensor and actuator networks have to be integrated into a common framework of global scale. In addition, they have to be made available to services and applications via universal interfaces. The SENSEI project [32] solves the inaccessibility of low-resource end devices by collecting all data from the end devices, and making them available in a centrally accessible database. In fact, it provides necessary network and information management services to enable reliable and accurate context information retrieval and interaction with the physical environment.

The main results of the SENSEI project can be summarized as follows [33]:

- A highly scalable architectural framework with corresponding protocol solutions. These solutions enable easy plug and play integration of a large number of globally distributed devices (i.e. things) into a global system. Doing so, it provides support for network and information management, security, privacy and trust, and accounting.
- An open service interface and corresponding semantic specifications to unify the access to context information and actuation services offered by the system.
- Efficient WSN and actuators solutions consisting of a set of cross-optimised and energy aware protocol stacks.
- Pan European test platform. This platform enables enabling large-scale experimental evaluation of SENSEI results. In addition, it provides a tool for long term evaluation of WSN and actuators integration into IoT.

By adding mechanisms for accounting, security, privacy and trust, SENSEI will enable an open and secure market space for contextawarness and real world interactions.

CASAGRAS project: CASAGRAS is considered as the first view on relevant topics of the IoT (e.g. architecture, features, governance, etc.), which is the result of an international analysis and discussion [33]. CASAGRAS project [5] aims to collect, review and analyze current and emerging proposals and solutions in the IoT. Although CASAGRAS's reference architecture provides the basis for implementing a distributed IoT, the processing is not pushed to the edge of the network, which is in charge of data gathering only. In fact, the logic is located in the Information Management System Layer. The CASAGRAS model includes three layers:

- **Physical Layer**: This layer identifies physical objects, and delivers the sensed data. In order to provide interoperability, objects are organized in networks through the specific Automatic Identification and Capture (AIDC) technology. In fact, an Universal Data Capture Appliance Protocol (UDCAP) is envisioned, whereby each AIDC technology will use its own implementation of UDCAP.
- **Interrogator-Gateway Layer**: It connects object-devices with information management systems.
- **Information Management Systems**: This layer provides the functional platform for supporting applications and services.

Server based approach: In [8], the authors introduce a Server-Based Internet of Things Architecture (SBIOTA). The main idea is to develop protocols, algorithms and services, based on a gateway server. This latter allows networked devices with extremely limited computation and communication capabilities to be part of the IoT in an effective, efficient, and secure way. In the following, we provide a broad overview of the main features of this approach:

- **Physical and link layer connectivity**: It is assumed that each small device is directly connected to a single server, which provides an intelligent gateway function between the device and the Internet.

- **Network layer connectivity**: IP connectivity will be based on IPv6 networking. In this addressing scheme, a gateway will handle any necessary IPv4 to IPv6 translations or tunnelling. By using IPV6, each device will have a dynamically assigned IP address. Because a full IPv6 implementation is costly for small devices, the 6lowPAN [28] protocol for communication on the links between the server and the device will be prefered. The server will also act as a firewall for each device.
- **Transport layer functions**: The two major IP-based transport layer protocols are UDP and TCP. The server will act as an endpoint for these protocols. Since UDP is more lightweight and hence more adapted to the IoT context, the server will communicate with the devices using UDP over 6lowPAN.
- **Application layer functions**: The Internet is moving away from providing access to data and hosts towards providing access to services [34]. In this context, every device will offer a HTTP web-server interface to its functionalities for authorized users. Each of these web-servers will be hosted on the gateway server.

Network virtualization: A solution based on virtual networks is introduced in [35]. According to the authors, current solutions that integrate smart resource-constrained objects into the Internet are mostly gateway-based. Their approach focuses on the objects, both resource-constrained and non-constrained, that need to cooperate. This integration is achieved by integrating the objects into a secured virtual network, named an Internet of Things Virtual Network or IoT-VN.

The authors have categorized the different approaches to expose the services offered by resource-constrained devices into two main categories. The first one is based on using gateways that are in charge of translating between protocols used in the Internet and protocols used in the sensor networks. The second one is based on integrating sensors into the IP-world. This approach allows direct end to end communication between the end sensors.

Both approaches have their advantages and disadvantages, characterized by the degree of openness in accessing the services on the resource-constrained devices. In fact, the use of gateways has certainly many advantages (e.g. high degree of access control, offload heavy computational operations, etc.) at the expense of a reduced flexibility of usage. Besides, IP-enabled sensors allows to overcome some drawbacks of the previous approach, such as providing the possibility of having gateways and sensors from different vendors. However, allowing direct communication between resource-constrained devices, new challenges related to connectivity, scalability and security are introduced. In this context, the authors propose a novel complementary approach.

Based on the fact that in several cases there is no need to expose the data generated by resource-constrained devices to the whole network. In fact, only a limited number of devices are involved. The proposed complementary approach aims to realize a secured and confined environment in which all objects that need to cooperate can communicate in an end-to-end manner. This is achieved by creating a virtual network of all involved devices, including resource-constrained devices.

Inside this virtual network, communication can take place between the networked objects regardless whether they are resource-constrained or not. This is achieved through the use of protocols that take into account the limitations of the most resource-

constrained devices. The authors described how this concept can constitute a valid alternative approach for realizing certain real-life scenarios. To reach such goal, they provide several generic use cases such as partitioning, aggregating multiple sensor networks, and extending a sensor network with non-constrained devices.

3.3 Clean Slate Architectures

BRIDGE project: The EPC Information Services (EPCIS) are used for storage and retrieval of processed information regarding supply-chain events. EPCIS provides a complete decentralized architecture. In fact, they include two separate interfaces, one for query requests and the other one for capture operations. A secure lookup service for locating the different providers of the distributed shares of information is required. Indeed, objects full information in relation with its lifecycle history or its complete supply-chain is spread through the different entities.

To enable RFID and EPC global standard solutions in practice, technical, social, and educational constraints, particularly in the area of security must be overcome. BRIDGE (Building Radio frequency IDentification solutions for the Global Environment) [6] extends the EPC network architecture and focuses on the following aspects [33]:

Network:

- Serial-level lookup service to enable unique item level product information storage and retrieval
- Identification and authentication of tags and readers
- Data management of large amounts of real-time data

Application Software:

- Serial-level inventory management
- Management of large networks of EPC readers
- Models to exploit environmental data (e.g. temperature, humidity, etc.)

Security:

- Security and privacy to prevent illicit use of EPC
- Prevention of cloning and emulation of tags in EPC
- Secure transmission of data between readers and tags

In a nutshell, BRIDGE aims to enable the deployment of EPC global applications in Europe. Its main axis are focused on developing security mechanisms in hardware, software, and business practises.

IDRA approach (direct connectivity): In the future IoT, a tremendous amount of heterogeneous devices (i.e. things) using vendor-specific proprietary network solutions will be connected. As a result, communication will only be possible through the use of gateway nodes, resulting in inefficient use of the wireless medium. In fact, there is no existing architecture yet that:

- Enables optimized communication, at a network and also at a link level, between co-located heterogeneous networks without the use of complex translation gateways;
- Has been implemented and evaluated as a prototype in a large scale experimental setting;
- Is compact enough to fit even on low-resource embedded devices;
- Is fully clean slate, but is also backward compatible with legacy networks.

In order to enable an end to end communication and overcome the use of gateways, the authors in [36] have tailored the IDRA architecture [37] to the context of IoT. This latter was designed specifically to enable connectivity between heterogeneous resource constrained objects. Its main advantages can be summarized as follows:

- IDRA can connect co-located objects directly, without the need for complex translation gateways;
- The architecture is clean slate, but supports backward compatibility with existing deployments;
- Due to its low memory footprint, the architecture can be used in resource-constrained objects.

Based on its characteristics, IDRA architecture aims to provide an approach that fills the gap between the current architectures and the future IoT requirements.

EPC based approach: In [38], the authors present an EPC (Electronic Product Code) based Internet of Things (IoT) architecture. The key concept of this architecture is deploying EPC over heterogeneous networks. It focuses on a ZigBee network as it can collect information about things. In fact, the EPC Network provides certain static information such as names and manufacturers of the objects.

According to the authors, an EPC based architecture requires a minimum set of features, such as uniquely identifying an object and automatic registration into the network. Moreover, it should provide Standard Application Programing Interfaces (APIs) to search, register, observe, and control objects made by different companies. In order to deal with the precedent requirements, the proposed architecture provides two functions. The first one is how to register new objects or devices to a home area network. The second one is how to make objects communicate through the Internet with generic protocols. The proposed EPC architecture uses combination of sensor networks and EPC networks, which provide product information through web services from the manufacturers. This architecture uses UPnP (Universal Plug and Play) protocol to automatically collect the EPC of a new connected object. In addition, ZigBee network system is applied for communication, and XML based web services are used for the application protocol. Genuine HTTP is a heavy protocol particularly for low bandwidth network, such as ZigBee and IEEE 802.15.4. Therefore, CoAP (Constrained Application Protocol) is adopted to support web services over ZigBee network. End to end communication is thus established regardless of the type of the network.

Cloud based approach: In the IoT paradigm, information and communication systems are invisibly embedded in the environment around us. This will result in the generation of huge amount of data, which has to be stored, processed and presented in a seamless, efficient, and easily interpretable way. According to [3], cloud computing is the most recent paradigm to emerge, promising high reliability, scalability, and autonomy. In fact, it provides ubiquitous access, dynamic resource discovery, and composability required for future IoT applications. This platform acts as a receiver of data from the ubiquitous sensors, as a computer to analyze and interpret data, as well as a provider to understand web based visualizations. The Cloud not only reduces costs of deploying ubiquitous applications, but is also highly scalable.

Sensing service providers can join the network and offer their data using a storage cloud, analytic tool developers can provide their software tools, artificial intelligence experts can provide their data mining and machine learning tools, and finally computer graphics designers can offer a variety of visualization tools.

Cloud computing can offer these services to the IoT as infrastructures, platforms, or softwares where the full potential of human creativity can be exploited. The generated data, used tools, and the process of generating complex visualizations are hidden in the background.

Social network approach: The Social Internet of Things (SIoT) architecture is introduced in [9]. The approach establishes a link between social networks and IoT. The main idea is that a large number of individuals tied in a social network can provide far more accurate answers to complex problems than a single individual (even knowledgeable one). In the future, things will be associated to the services they can deliver. Thus, to better implement services within a given social network of objects, a key objective will be to publish information/services, find them, and discover novel resources. This can be achieved by navigating a social network of 'friend' objects instead of relying on typical Internet discovery tools that cannot scale to the trillions of future devices.

Authors in [9], claim that social relationships among humans might be applicable to certain kinds of behaviors of typical objects implementing pervasive applications. There is no doubt that many applications and services should be associated with groups of objects, which will cooperate in order to reach the overall interest of providing services to users (e.g. the same idea is behind the approaches involving the use of swarm intelligence and swarm robotics).

The social architecture relies upon basic kinds of relationships such as the **Parental object relationship (POR)**, which is established among objects belonging to the same production batch, or the **Ownership object relationship (OOR)**, which is based on heterogeneous objects belonging to the same user (e.g. mobile phones, game consoles, etc.). The authors draw attention about the fact that the establishment and the management of such relationships should occur without human intervention. This is not in contrast with a future vision of a fully networked human. This latter is only responsible for setting the rules of the objects and their social interactions. This is a clear paradigm shift from other proposals, where the objects just participate in the human social network built by their owners.

4 Critics and Analysis

The proposed architectures, either the public projects or those introduced by the research community, aim to reduce the gap between the concept of the IoT and its real deployment into our daily lives. We have proposed a classification that gathers the different architectures into two categories. The first one, called the tailored architectures, contains the approaches that propose an evolution of the current Internet to a more suitable network for the IoT such as network virtualization, and server based approach. This category will certainly provide the advantage of backward compatibility with existing architectures. However, several issues remain such as security and resources limitations. The second category includes clean slate architectures such as the IDRA approach and the social network approach. These approaches claim a novel vision of the future IoT that inherently copes with next-generation network challenges. In fact, this provides the benefit of a design, completely dedicated to be tailored to IoT characteristics. Nevertheless, backward compatibility with existing approaches remains a challenge.

Each presented architecture in this paper is summarized in Table 1, along with a brief description and the main shortcomings. Some eventual improvements are also provided. In the following, we propose an analysis of each architecture, highlighting the matching of its characteristics with IoT requirements.

The IETF is focusing its efforts on adapting existing protocols, which have been developed for the classical Internet to the constrained environment of IoT. To this end, the IETF proposes an equivalent of the existing protocols for each layer of the TCP/IP stack, such as 6LoWPAN for IPV6 and CoAP for HTTP. However, although the precedent solutions constitute a sound basement on which further efforts can be made, several challenges should be addressed. For instance, the limited channel capacity of the IEEE 802.15.4 can hinder the scalability and the traffic load of future IoT applications. Moreover, Quality of Service (QoS) support for networks with heterogeneous traffic is still problematic in IEEE 802.15.4 [39]. In addition, several studies such as [40] highlight security breaches in the IETF protocol suite. Thus, the IETF protocol suite has to be strengthen regarding the security aspect, which is considered as a primary concern in the IoT.

SENSEI [32] focuses on equipping the objects with a certain kind of intelligence by embedding processing capabilities into them. The project provides the architecture for connecting heterogeneous objects via the specification of open service interfaces. However, the use of centrally accessible database results in a significant network overhead, and could constitute a single point of failure. Additionally, the SENSEI project is still under development. It needs to reach a mature state before an effective evaluation. CASAGRAS [5] also proposes a vision of the IoT whereby both virtual and physical generic objects are connected through a global infrastructure. The project focuses too much on RFID as the main building block of the IoT while it is likely to have a multitude of integrated technologies forming the future IoT. Like SENSEI, CASAGRAS presents a narrow-waist. Any interaction has to pass through the Management System at the service, or application layer. BRIDGE [6]

Table 1 Summary of the proposed architectures

Architectures	Description	Drawbacks	Potential improvements
IETF protocol suite	Focuses on proposing and adapting standard-based communication protocols for the IoT	Resources limitations, QoS support for heterogeneous traffic, and security issues	Introducing QoS management by handling differently the various classes of traffic. Designing, and integrating built-in standard-based security protocols
SENSEI [32]	EU project aiming to design an architecture for the connectivity of global and heterogeneous sensor and actuator networks via the specification of open service interfaces	The use of centrally accessible data base results in significant network overhead. No ID standards. The SENSEI project is still under development	Adoption of ID standards. The project need to reach a mature state before its results can be evaluated
BRIDGE [6]	The bridge project aims at supporting ambient sensors and sensor enabled RFID tags in the EPC global networks for supply chain monitoring	No extension of the EPC Network standard to deal with sensor data is provided	Extend EPC Network standard with sensor data
CASAGRAS [5]	Focuses on global standards regulatory and other issues concerning RFID and its role in the Internet of Things	Any interaction must pass through an Information Management System at the service or application layer which constitutes the narrow-waist of the architecture. CASAGRAS's focus is too much on RFID only	
IDRA approach [36]	Enabling direct connectivity between heterogeneous objects through a network-service-oriented architecture	Additional processing delay is caused by the different computations in the system	Taking into account the QoS requirements of the packet, additional delay will only be introduced for low-priority traffic

(continued)

Table 1 (continued)

Architectures	Description	Drawbacks	Potential improvements
Server based approach [8]	Communication between networks from different vendors, or between devices that use different network protocols is achieved by connecting each network to a vendor-specific translation gateway	This approach breaks the end to end communication paradigm. It uses the wireless medium inefficiently, and presents a single point of failure	
EPC based approach [38]	Emerging industrial RFID standard architecture. It uses a unique item identification via the Electronic Product Code (EPC)	Does not yet handle sensor data	Extend current standards with sensor data
Cloud approach [3]	Offloads resource intensive tasks to more capable nodes	This approach could be implemented in the network layer to handle processing tasks. It does not solve connectivity challenges	
Social network approach [9]	A parallel is made between the current social networks and a future network of objects	Could only be implemented in the application layer. It does not deal with lower layers issues	
Virtual networks approach [35]	Network virtualization is used to present underlying network layers in a uniform way toward high-level applications	Scalability has not been proven yet and complexity might be an issue on resource-constrained embedded devices	Reduce the complexity of the used techniques and provide tests on huge and scalable networks comparable to the future IoT network scale

aims to research, develop, and implement tools to enable the deployment of Radio Frequency Identification (RFID) and EPC global Network applications. The core of BRIDGE is communication centric. It addresses the problem of handling queries between distributed entities. Nevertheless, the work with sensors does not extend the EPC network standards.

The IDRA [36] architecture proposes a clean slate approach that challenges the layered vision of the current internet architecture. IDRA aims to enable a direct con-

nectivity between heterogeneous objects through a network-service-oriented architecture. However, additional processing might impede an efficient deployment in a resource-constrained environment. The virtualization approach [35] also aims to establish an end-to-end communication between the devices that need to cooperate. In fact, this approach integrates them into a secured virtual network regardless whether the resources are constrained or not. Yet, the scalability has not been proven, and the complexity of the protocols used might be an issue. To provide an end to end communication regardless of the type of the access network, another promising architecture has been presented in the EPC based approach [38]. The main idea is to combine sensor networks with EPC networks, which provide product information through web services from manufacturers. Server based approach [8] proposes a different solution to connect networks from different vendors, or devices that use different protocols. The idea is to use a translation gateway. Nevertheless, this solution breaks the end to end communication principle. In addition, the gateway could represent a single point of failure.

The social network approach [9] introduces an interesting idea by making the parallel between the current social networks and a future network of objects. The goal is to publish, find information, and discover novel resources to better implement the services. Nevertheless, this approach does not deal with the issues of lower layers of the network. Besides, in order to take into account the scarcity of resources in the future IoT, the cloud approach [3] proposes offloading resource intensive tasks to more capable nodes. In fact, the cloud offers both flexibility and a high scalability level. However, the cloud architecture does not deal with the connectivity challenges at lower levels of the network.

In a nutshell, we do believe that a well-defined architecture is required instead of letting the current Internet raise to the IoT in an uncontrolled way. Issues like security need to be addressed during design time. In addition, we consider that the different proposed architectures are not contradictory; an hybrid architecture including several approaches might be an efficient way to address the IoT's specificities. Based on the commonly accepted three-layer architecture, each approach might be implemented in the appropriate layer. For instance, the cloud approach affects the application layer whereby the future applications will need to be ubiquitously accessible, while the IDRA approach could be implemented in the network layer to secure a dynamic adaptation of the network.

5 Conclusions

Internet Of Things brings the possibility to connect billions of every-day's objects to the Internet, allowing them to interact and to share data. This prospect open new doors toward a future where the real and virtual world merge seamlessly through the massive deployment of embedded devices. The IoT has the potential to add a new dimension by enabling communications with and among smart objects, leading to the vision of anytime, anywhere, any media, and anything communication paradigm. Though, a lot

still to be done in order to fulfill the IoT vision. A scalable, backward compatible, and secure architecture is required to bring the IoT concept closer to reality. In this chapter, we have provided an overview on the main proposed architectures in the literature, along with the building blocks technologies that are considered well-adapted to suit IoTs requirements. We have also introduced a classification highlighting the suitability of the proposed architectures to IoT characteristics. In addition, we have underlined the main shortcomings of the current approaches and proposed our vision regarding the IoT's future architecture based on the current state of the art. As a future research direction, we plan to design a suitable approach to deal with the different challenges of the IoT at each layer of the network.

References

1. Atzori, L., Iera, A., Morabito, G.: The internet of things: a survey. Comput. Netw. **54**, 2787–2805 (2010)
2. Miorandi, D., Sicari, S., Pellegrini, F.D., Chlamtac, I.: Internet of things: vision, applications and research challenges. Ad Hoc Netw. **10**, 1497–1516 (2012)
3. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (IoT): a vision, architectural elements, and future directions. Future Gener. Comput. Syst. **29**(7), 24 (2007)
4. Yun, M., Yuxin, B.: Research on the architecture and key technology of internet of things (IoT) applied on smart grid. In: Advances in Energy Engineering (ICAEE), pp. 69–72 (2010)
5. CASAGRAS: Casagras project. http://www.rfidglobal.eu (2009)
6. BRIDGE: Bridge: building radio frequency identification solutions for the global environment. http://www.bridge-project.eu (2009)
7. Dunkels, A., Vasseur, J.: Internet protocol for smart objects. IPSO Alliance, White paper 1. http://www.ipso-alliance.org, September 2008
8. Bergmann, N.W., Robinson, P.: Server-based internet of things architecture. In: The 9th Annual IEEE Consumer Communications and Networking Conference (2012)
9. Atzori, L., Iera, A., Morabito, G., Nitti, M.: The social internet of things (SIoT) when social networks meet the internet of things: concept, architecture and network characterization. Comput. Netw. **56**(16), 3594–3608 (2012)
10. Hassan, M.M., Song, B., Huh, E.N.: A framework of sensor-cloud integration opportunities and challenges. In: ICUIMC 09, January 2009. ACM (2009)
11. Hu, G., Tay, W.P., Wen, Y.: Cloud robotics: architecture, challenges and applications. IEEE Netw. **26**(3), 21–28 (2012)
12. Ye, W., Heidemann, J., Estrin, D.: An energy-efficient MAC protocol for wireless sensor networks. In: INFOCOM. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 1567–1576 (2002)
13. Curt, S., Srivastava, M.: Energy efficient routing in wireless sensor networks. IEEE Mil. Commun. Conf. MILCOM Commun. Netw.-Centric Oper.: Creat. Inf. Force **1**, 357–361 (2001)
14. Abdmeziem, M.R., Tandjaoui, D.: Tailoring Mikey-ticket to e-health applications in the context of internet of things. In: International Conference on Advanced Networking, Distributed Systems and Applications (Short Papers), pp. 72–77, June 2014
15. Want, R.: An introduction to RFID technology. IEEE Pervasive Comput. **5**(1), 25–33 (2006)
16. Nath, B., Reynolds, F., Want, R.: RFID technology and applications. IEEE Pervasive Comput. **5**(1), 22–24 (2006)
17. Akyildiz, I., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: a survey. Comput. Netw. **38**(4), 393–422 (2002)

18. Zhang, L., Wang, Z.: Integration of RFID into wireless sensor networks: architectures, opportunities and challenging problems. In: Grid and Cooperative Computing Workshops. GCCW'06, pp. 463–469. IEEE (2006)
19. Papazoglou, P., Georgakopoulos, D.: Service oriented computing. Commun. ACM **46**(10), 25–28 (2003)
20. Wei, Y., Blake, B.: Service-oriented computing and cloud computing. IEEE Internet Comput. **14**(6), 72–75 (2010)
21. Zhou, J., Leppanen, T., Harjula, E., Ylianttila, M., Ojala, T., Yu, C., Jin, H., Yang, L.: Cloudthings: a common architecture for integrating the internet of things with cloud computing. In: 17th International Conference on Computer Supported Cooperative Work in Design (CSCWD), pp. 651–657. IEEE (2013)
22. Kovatsch, M., Mayer, S., Ostermaier, B.: Moving application logic from the firmware to the cloud: towards the thin server architecture for the internet of things. In: Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), pp. 751–756. IEEE (2012)
23. Won, S.S., Hamby, A.M., Swanson, R.A.: Hypoglycemia, brain energetics, and hypoglycemic neuronal death. Glia **55**(12), 1280–1286 (2007)
24. Kamei, K., Nishio, S., Hagita, N.: Cloud networked robotics. IEEE Netw. **26**(3), 28–34 (2012)
25. Wu, M., Lu, T., Ling, F., Sun, J., Du, H.: Research on the architecture of internet of things. In: 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE). IEEE (2010)
26. Vasseur, J., Dunkels, A.: Interconnecting Smart Objects with IP: The Next Internet. Morgan Kaufmann, San Francisco (2010)
27. IEEE Std 802.15.4-2003, pp.0–1670, 2003 doi:10.1109/IEEESTD.2003.94389
28. Shelby, Z., Bormann, C.: 6LoWPAN: The Wireless Embedded Internet, vol. 43. Wiley, Chichester (2011)
29. Winter, T.: RPL: IPv6 routing protocol for low-power and lossy networks (2012)
30. Shelby, Z., Hartke, K., Bormann, C.: The constrained application protocol (CoAP). RFC 7252, June 2014
31. Song, J., Han, S., Mok, K., Chen, D., Lucas, M., Nixon, M.: WirelessHART: applying wireless technology in real-time industrial process control. In: Real-Time and Embedded Technology and Applications Symposium, RTAS'08, pp. 377–386. IEEE (2008)
32. Sensei project. http://www.ict-sensei.org/ (2010)
33. Bui, N.: Internet of things architecture. Technical report, Project co-funded by the European Commission within the Seventh Framework Program (2011)
34. Schroth, C.: The internet of services: global industrialization of information intensive services. In: Proceedings of 2nd IEEE International Conference on Digital Information Management, ICDIM'07 (2007)
35. Ishaq, I., Hoebeke, J., Moerman, I., Demeester, P.: Internet of things virtual networks. In: IEEE International Conference on Green Computing and Communications, Conference on Internet of Things, and Conference on Cyber, Physical and Social Computing, pp. 293–300 (2012)
36. Poorter, E., Moerman, I., Demeester, P.: Enabling direct connectivity between heterogeneous objects in the internet of things through a network-service-oriented architecture. J. Wirel. Commun. Netw. **2011**(1), 1–14 (2011)
37. Overview of the currently available network services in the IDRA architecture. http://idraproject.net/protocols-and-applications
38. Hada, H., Mitsugi, J.: EPC based internet of things architecture. In: IEEE International Conference on RFID-Technologies and Applications, pp. 527–532 (2011)
39. Sheng, Z., Yang, S., Yu, Y., Vasilakos, A., Mccann, J., Leung, K.: A survey on the IETF protocol suite for the internet of things: standards, challenges, and opportunities. IEEE Wirel. Commun. **20**(6), 91–98 (2013)
40. Medjek, F., Tandjaoui, D., Abdmeziem, M.R., Djedjig, N.: Analytical evaluation of the impacts of Sybil attacks against RPL under mobility. In: International Symposium on Programming and Systems, April 2015. IEEE

Cloud of Things: Integration of IoT with Cloud Computing

Mohammad Aazam, Eui-Nam Huh, Marc St-Hilaire, Chung-Horng Lung and Ioannis Lambadaris

Abstract With rapidly increasing Wireless Sensor Networks (WSNs) and Internet of Things (IoTs) based services; a lot of data is being generated. It is becoming very difficult to manage power constrained small sensors and other data generating devices. With IoTs, anything can become part of the Internet and generate data. Moreover, data generated needs to be managed according to its requirements, in order to create more valuable services. For this purpose, integration of IoTs with cloud computing is becoming very important. This new paradigm is termed as Cloud of Things (CoTs). CoTs provide means to handle increasing data and other resources of underlying IoTs and WSNs. It also helps in creating an extended portfolio of services that can be provided with this amalgamation. In future, CoTs are going to play a very vital role. In this chapter, the importance of CoT, its architecture, working, and the issues involved are discussed.

Keywords Internet of Things (IoT) · Cloud computing · Cloud of Things (CoT)

M. Aazam · E.-N. Huh (✉)
Kyung Hee University, Suwon, Korea
e-mail: johnhuh@khu.ac.kr

M. Aazam
e-mail: aazam@ieee.org

M. Aazam · M. St-Hilaire · C.-H. Lung · I. Lambadaris
Department of Systems and Computer Engineering,
Carleton University, Ottawa, Canada
e-mail: marc_st_hilaire@carleton.ca

C.-H. Lung
e-mail: chlung@sce.carleton.ca

I. Lambadaris
e-mail: ioannis@sce.carleton.ca

A. Koubaa and E. Shakshuki (eds.), *Robots and Sensor Clouds*,
Studies in Systems, Decision and Control 36,
DOI 10.1007/978-3-319-22168-7_4

1 Introduction

Internet of Things (IoT) is a technological revolution that represents the future of connectivity and reachability. In IoT, 'things' refer to any object on the face of the Earth, whether it is a communicating device or a non-communicating dumb object. From a smart device to a leaf of a tree or a bottle of beverage, anything can be part of Internet.

IoT works on the basis of Machine-to-Machine (M2M) communications, but not limited to it. M2M refers to communication between two machines, without human intervention. In IoT, even non-connected entities can become part of IoT, with a data communicating device, like a bar-code or an RFID tag, sensed through a device (may even be a smartphone sensing it), which eventually is connected to the Internet. Non-intelligent objects, known as 'things', become the communicating nodes in IoT. IoT based services are gaining importance rapidly. Since 2011, the number of connected devices has already exceeded the number of people on Earth. Already, connected devices have reached 9 billion and are expected to grow more rapidly and reach 24 billion by 2020 [21]. With an increasing number of heterogeneous devices connected to IoT and generating data, it is going to be a great challenge for a standalone IoT to perform power and bandwidth constrained tasks efficiently. In this regard, IoT and cloud computing amalgamation has been envisioned [7, 9]. There comes a situation when a cloud is connected with an IoT that generates multimedia data. Visual Sensor Network or CCTV connected to the cloud can be examples of such scenario. Since multimedia content consumes more processing power, storage space, and scheduling resources, it will be very important to manage them effectively and perform efficient resource management in the cloud. Other than that, mission critical and latency sensitive IoT services require a very quick response and processing. In that case, it is not feasible to communicate through distant cloud, over the Internet.

This chapter focuses on energy efficiency, resource management, and creating new and extended portfolio of services through Cloud of Things. Latency sensitive and emergency related services can take benefit from this new paradigm to a great extent. Integration of Internet of Things with cloud computing is gaining importance, with the way the trend is going on in ubiquitous computing world. Literally, everything is going to be connected to the Internet and its data would be used for various progressive purposes, creating not only information from it, but also, knowledge and even wisdom. IoT becoming so pervasive that it is becoming important to integrate it with cloud computing because of the amount of data IoTs could generate and their requirement to have the privilege of virtual resource utilization and storage capacity, but also, to make it possible to create more usefulness from the data generated by IoTs and develop smart applications for the users. The integration of IoT with cloud computing, referred to as Cloud of Things [7, 9], requires a smart gateway to perform the rich tasks and preprocessing, which sensors and light IoTs are not capable of doing. We have also discussed some of the key challenges involved in CoT and the proposal of smart gateway based communication.

It is not going to be that simple to allow everything to become a part of IoT and then having all the resources available through cloud computing. There lies some issues that have to be taken care of to allow CoT to prevail. Other than data and resources, the cloud has to deal with the business point of view as well. CoT will create more business opportunities, making it a bigger target for the attackers. Security, privacy, and specially, identity protection becomes very important in hybrid clouds, where there is an essence of private and public clouds, used by businesses. In CoT, heterogeneous networks will be involved, which support different types of data and services. The network must have the flexibility to support all types of data, according to their requirements, with QoS support. Some of the key issues in this regard will be discussed later in the chapter.

2 Contributions of the Book Chapter

In this chapter, we have discussed the importance of the integration of IoT with cloud computing, called CoT, for the purpose of better resource management and energy efficiency. We have presented the working scenario of CoT. Moreover, the challenges CoT may face are discussed along with their appropriate envisioned solutions. To have standard architecture of CoT, the challenges discussed in this chapter will help a great deal to focus the future research and development. This chapter will help in making directions towards not only standardizing CoT, but also, gives directions towards more practical implications of CoT and addressing the noteworthy issues in this regard.

3 Related Work

CoT is still in its start; therefore, there is no standard architecture available for data communication, media storage, compression, and media delivery. Already done studies mainly provide very abstract and simplistic architectural blueprints. Pritee Parwekar et al. present the concept of IoT and cloud computing integration in [35]. However, the authors only provide a very abstract overview of the concept. On the other hand, in this chapter, we discuss IoT-cloud integration with details on the working architecture, working scenarios, and potential issues along with the directions towards their solutions. Salvator Distefano et al. also present in [16] the concept of CoT. The authors discuss the architecture of CoT, however noteworthy issues are not included in their work. Madoka Yuriyama et al. [46] discuss sensor-cloud infrastructure, but the authors only focus on how sensor-cloud infrastructure virtualizes a physical sensor as a virtual sensor in the cloud. Mohammad Mehedi Hassan et al. [22] also presented sensor-cloud integration. Even though the authors claim to present issues involved in this regard, the study does not present any of the key problems faced by the sensor-cloud integration. In our work, challenges like:

resource management, energy efficiency, heterogeneous protocols support, etc., are discussed.

Many underlying IoT devices, like: smartphones, tablet computers, and media related sensor networks such as Visual Sensor Networks (VSN), entertainment systems in vehicular ad hoc networks (VANETS), require efficient media processing. Intel-HP Viewpoint paper [2] presents an industrial overview of the media cloud in this regard, but IoT and the above mentioned scenarios are not part of that study. It is stated that media cloud is the solution to suffice the dramatically increasing trends of media content and media consumption. For media content delivery, QoS is going to be the main concern. Regarding customized QoS provisioning, we presented an end to end QoS provisioning mechanism using the Flow Label of IPv6 and Multi-Protocol Label Switching (MPLS) [11]. To reduce delay and jitter of media streaming, better QoS is required, for which W. Zhu et al. [48] propose the media-edge cloud (MEC) architecture. The authors state that the MEC is a cloudlet which locates at the edge of the cloud. MEC is composed of storage space, Central Processing Unit (CPU), and graphics processing unit (GPU) clusters. The MEC stores, processes, and transmits media content at the edge, thus incurring a shorter delay. In turn, the media cloud is composed of MECs, which can be managed in a centralized or peer-to-peer (P2P) way. The authors do not present the cost-effect of this proposal. Moreover, the MEC only acts as a proxy. Transcoding and resource management tasks are still not done at this point.

Liam McNamara et al. present a demo application for low powered devices and sensors, for the purpose of storage of data in the cloud [32]. Geoffrey C. Fox et al. also present characteristics of cloud based Internet of Things [19]. In their work, the authors present an open-source cloud-IoT framework. Rogers Owen et al. present [36] a resource allocation mechanism in cloud arena, but their study lacks the scenario where IoT is involved. Their study is also only limited to standard cloud resource management. Ki-Woong Park et al. [34] present a billing system with some security features. To resolve different types of disputes in future, a mutually verifiable billing system is presented. Their work only focuses on the reliability of transactions made in purchasing and consuming resources. They do not focus on the overall resource management specially with CoT. Only transactions security in cloud is discussed. Wei Wang et al. [41] propose a brokerage service for reservation of instances. The authors propose a brokerage service for on-demand reservation of resources, for IaaS clouds. Their work is limited to cloud only services. This brokerage model can be extended to Fogs for IoTs, as discussed in our prior works [5, 7]. Same is the case with Foued Jrad et al., who present in [24, 25] a generic architecture of the broker. They present how the broker handles SLA management and interoperability of resources. Yichao Yang et al. present resource allocation algorithm in a simplistic way [45]. Ewa Deelman et al. present performance tradeoffs of different resource provisioning plans. They also present tradeoffs in terms of storage fee of Amazon S3 [15]. The scope of these studies can be extended by incorporating IoT based resource management. Shadi Ibrahim et al. present the concept of fairness in pricing with respect to micro-economics [23], but do not discuss how pricing should be done for different types of services, specially in the case of IoT. Nikolay Grozev et al. present basic taxonomies

for inter-cloud architecture [20], which lacks relationship of it with IoT. Buyya et al. presents architectural fundamental of inter-cloud computing [13] which also does not include IoT. David Villegas et al. present in [40] how multiple clouds are influenced by creating a cloud federation environment. Kan Yang et al. present in [44] a dynamic auditing protocol for ensuring the integrity of stored data in the cloud. They present an auditing framework for cloud storage. Zhen Xiao et al. present in [43] a resource allocation system that uses virtualization technology to dynamically allocate resources, according to the demands of the service. In their study, they present measuring the unevenness in resource utilization. IoT based environment is not considered in this study. D. Cenk Erdil, in [17], presents an approach for resource information sharing through proxies. In situations where clouds are distant and there is no direct control, proxies can be used to make resource information available to them. This study only focuses on the importance of resource information sharing. Research is now required to be more towards IoT-cloud integration. Rakpong et al. consider resource allocation in mobile cloud computing environment in their work [26]. They discuss about communication/radio resources and computing resources, but their work only focuses on decision making for coalition of resources, to increase service providers revenue. Kenji Tei and Levent Gurgen discuss in [38] about CoT. They term this paradigm as ClouT. The authors endorse that for efficiently managing energy and economic growth, CoT is an inevitable requirement. According to the authors, more than half of the world's population lives in cities. With the advent of smart cities, it is going to be literally impossible to handle the data generated and manage the services. Anuj Sehgal et al. [37] discuss about resource management of devices in IoT. Devices and sensors in IoT are resource constrained. Other than power, memory and processing capabilities, interoperability is going to be a big concern. In this case, the authors advise to use the IPv6 protocol, due to its large address space and the number of already existing protocols, capable of functioning over IP.

4 Internet of Things

IoT, the term first introduced by Kevin Ashton in 1998, is the future of Internet and ubiquitous computing [42]. This technological revolution represents the future of connectivity and reachability. Unlike the traditional networks of embedded systems, IoT is capable of interconnecting heterogeneous devices, having diverse functionalities, produced by different manufacturers [29]. The objects become communicating nodes over the Internet, through data communication means, primarily through Radio Frequency Identification (RFID) tags. IoT includes smart objects as well. Smart objects are those objects that are not only physical entities, but also digital ones and perform some tasks for humans and the environment. This is why, IoT is not only a hardware and software paradigm, but also includes interaction and social aspects as well [28].

In its simplest terms, IoT refers to a network of inter-connected things, objects, or devices on a massive scale. These objects are able to connect to the Internet. These objects, in huge numbers, are made smart; they sense their surroundings, they gather and exchange data with other similar devices. Based on the gathered data, the devices make intelligent decision to trigger an action or send the data to a server over the Internet and wait for its decision. Most common nodes in IoT are sensors [30] used in many areas from industrial process control to inside ovens and refrigerators and RFID chips [33] used as tags in many products of everyday use. Almost all of these smart devices have a short communication range and require very little power to operate. Bluetooth [31] and IEEE ZigBee [4] are the most common communication technologies used in this regard.

Another aspect of these devices is their network topology. A single smart device (e.g. in our refrigerator) will communicate to a router installed in the house or with a cellular tower and the same thing will happen for similar devices installed in other equipment and places. But in places where a large number of these devices are used, an aggregation point might be required to collect the data and then send it to a remote server. Examples of such deployment can be industrial process control, monitoring of utilities supply lines, such as oil pipelines or water sewage lines, product supply chain in a warehouse or some secured area.

IoT also presents many possible scenarios where heterogeneous devices interact with each other and then pass on the information to a central authority. One such scenario is the amalgamation of Intelligent Transportation System (ITS) with IoT. Nowadays, ITS is a very active research area which envisions to provide commuters safer and time efficient travel to their destinations. Several sensors on-board a vehicle and also on traffic signal poles monitor and sense the traffic situation on the roads. On the one hand, this real time information is presented to the drivers to see the traffic situation towards their destination and plan the journey accordingly. Moreover, this information is also sent to a traffic control authority to monitor the traffic congestion around the city and then direct traffic to alternate routes or change the time of traffic signals duration on demand.

Another point is that by looking at mainstream applications, it is evident that they do not require too much bandwidth. At least at this moment, the IoT applications probably have less bandwidth requirements than HD video streaming and Video on Demand (VoD) applications we use today. The data is transmitted in short bursts and at regular intervals. What these applications demand, however, is short latency in network access, transmission and guaranteed delivery of the data. Security is another aspect which is important. We are migrating towards the era of M2M communications but to enable automation in our daily life we must take absolute security and privacy concerns very seriously. Many of the envisioned applications require response or command to perform some action in minimum time possible. Hence a prioritized and quick access to the network will be the basis of IoT. In short, there is a plethora of possible applications, services, devices, communication technologies, network topologies etc. all contributing to the complex architecture of IoT.

Some envisioned IoT application areas include:

- Smart Cities
- Product Manufacturing
- Agriculture Automation
- Logistic Services
- Security, Monitoring, and Surveillance
- Smart Vehicles
- Green and Energy Efficient Homes
- Tele Medicine and Healthcare
- Product Monitoring
- Environment Monitoring
- Emergency Management

Realizing the potential of IoT, Intel has coined its own term of 'Embedded Internet' [1]. The concept is not largely different from the traditional IoT but Intel realizes that smart devices embedded into many devices will be the norm in the future. They will communicate with other larger systems and among each other. This brings new opportunities for product and service developers to generate revenue sources. The architecture of IoT is usually considered to be 3-layer, with perception layer, network layer, and application layer, but some [27, 42] add two more layers: middleware layer and business layer. This five layer architecture is described in Fig. 1. This layered architecture provides an overview of how IoT service provisioning is divided and what types of stages are involved for the data to be produced and ultimately, create services.

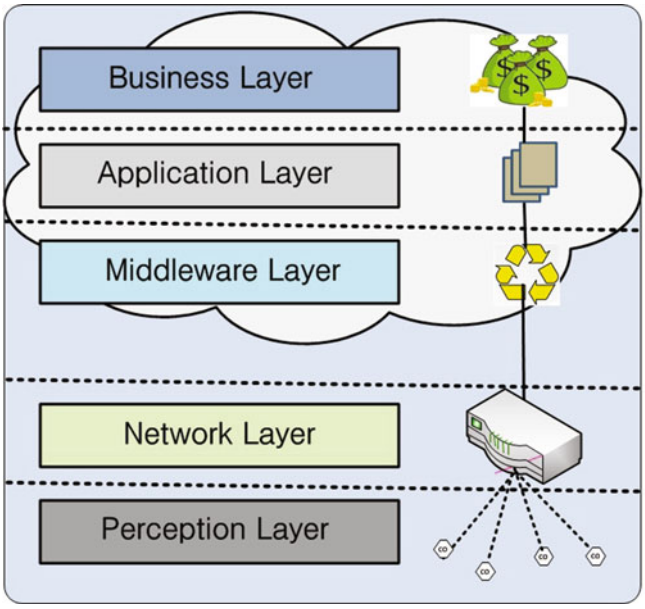


Fig. 1 Internet of things layers

The perception layer is the lowest layer in the IoT architecture. As the name suggests, its purpose is to perceive the data from the environment. All the data collection and data sensing part is done at this layer [39]. Sensors, bar code labels, RFID tags, GPS, and camera, lie in this layer. Identifying objects/things and gathering data are the main purpose of this layer. The network layer is like the Network and Transport layers of OSI model. It collects the data from the perception layer and sends it to the Internet. The network layer may only include a gateway, having one interface connected to the sensor network and another to the Internet. In some scenarios, it may include a network management center or information processing center. The middleware layer receives data from the network layer. Its purpose is service management and storage of data. It also performs information processing and takes decisions automatically based on the results. It then passes the output to the next layer, the application layer [27]. The application layer performs the final presentation of data. The application layer receives information from the middleware layer and provides global management of the application presenting that information, based on the information processed by the middleware layer. Depending upon the type of devices and their purpose in perception layer and then on the way they have been processed by the middleware layer, according to the needs of the user, the application layer presents the data in the form of: smart city, smart home, smart transportation, vehicle tracking, smart farming, smart health and many other kinds of applications [27]. The business layer is all about making money from the service being provided. Data received at the application layer is molded into a meaningful service and then further services are created from those existing services. Furthermore, information is processed to make it knowledge and further efficient means of usage makes it wisdom, which can earn a good amount of money to the service provider.

5 Cloud Computing

Cloud computing newly arose and advanced swiftly as a capable as well as preordained technology. Cloud computing platform brings with it highly scalable, manageable, and schedulable virtual servers, storage, computing power, and virtual networking, according to user's requirements. Therefore, it can provide solution package for the digital data revolution, if accordingly designed for IoTs and integrated with the advanced technologies on data processing, transmission, and storage. On average, a user generates content very quickly as long as its storage space permits [21]. Most of the content may be used frequently by the user, which requires to be accessed easily. Media management is among the key aspects of cloud computing, since cloud makes it possible to store, manage, and share large amount of digital media. For media content related IoTs, this feature is going to play a very important role. In future, several multimedia services for the users who are on the go, such as smartphone, tablet, and laptop users, vehicular ad hoc networks, various emergency and rescue related services will be available. For such services, cloud computing is going to play a very important role in service and resource management. Specially

with the extended cloud, Fog Computing [5], also known as Edge Computing, or Micro-Datacenter (MDC), the cloud will be more diversely used. Cloud computing is a handy solution for processing content in distributed environments. It provides ubiquitous access to the content, without the hassle of keeping large storage and computing devices. Sharing large amount of digital content is another feature that cloud computing provides. Other than social media, traditional cloud computing provides additional features of collaboration and editing of content. Likewise, if content is to be shared, downloading individual files one by one is not easy. Cloud computing caters this issue, since all the content can be accessed at once by other parties, with whom the content is being shared. Furthermore, more context-aware services can be provided through cloud computing, since IoT and sensor nodes are not rich enough in resources to accomplish such tasks. Data stored in the cloud can also be further analyzed, in order to create more customized and useful services.

6 Cloud of Things

We are moving towards web 3.0, the ubiquitous computing web. Since the number of connected devices is rapidly increasing, hence, the amount of data will also be increasing. Storing that data locally and temporarily will not be possible anymore. There is going to be a need of rental storage space. Moreover, this huge amount of data must also be utilized in the way it deserves. Data must not only be processed to form information and further, to form knowledge, but it should be made a mean of wisdom for the user. This asks for more processing, which is not possible at the IoT end, where devices are low cost and light-weight. Again, processing and computation must also be available there on rental basis. All this is possible with cloud computing. IoT and cloud computing working in integration makes a new paradigm, which is called Cloud of Things [5, 7, 9]. IoT provides sophisticated means of communication with the broader world, the web, through ubiquitous networks and devices. On the other hand, cloud computing provides scalable network access, according to the demands [47]. Figure 2 presents an overall communication pattern of CoT. This figure provides an overall picture of how IoT-cloud communication will take place. Various IoTs generate data, which passes through each of the layer presented in Fig. 1. The data is communicated through a communication channel. Different examples are illustrated in the Fig. 2. The data ultimately reaches the cloud, which stores, processes, and secures the data, according to the requirements of the service. Once the service is created, it is made available to the end user, which is residing on the other side of the cloud, at the access layer.

Other than sensors and IoT nodes, smartphones are also going to be part of IoT. Thanks to the advanced and capable access networks, like 3G, 4G, LTE, LTE-Advanced, WiBro, etc., a lot of multimedia communication is going to take place. CoT will play an important role in this regard, not only in delivering the service, but also, managing it.

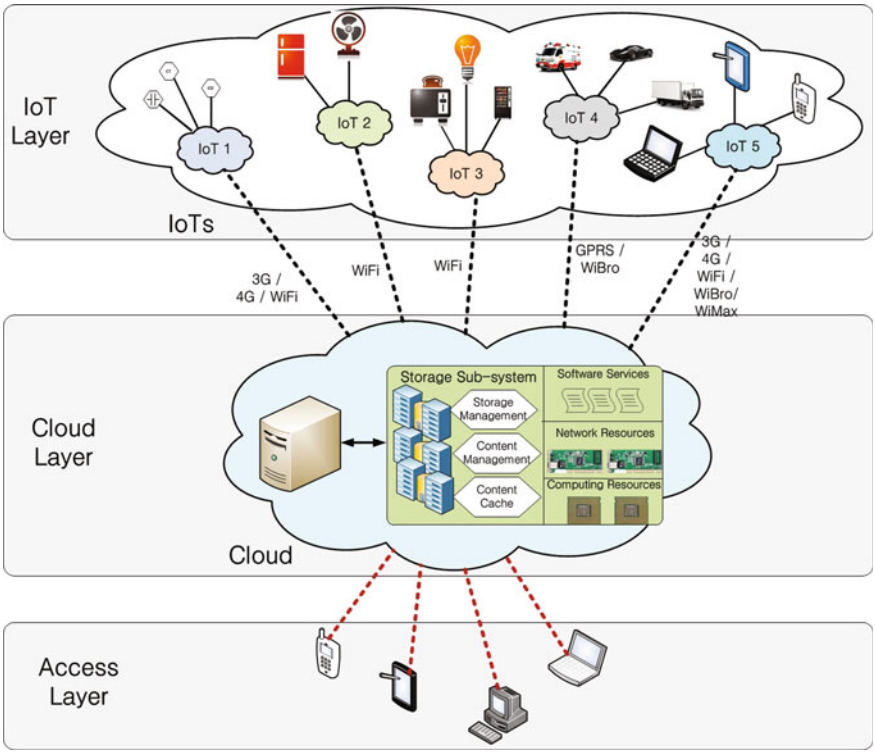


Fig. 2 IoTs and cloud—data communication

7 Challanges Associated with Cloud of Things

It is not going to be that simple to allow everything to become part of IoT and then having all the resources available through cloud computing. There lies some issues that have to be taken care of to allow CoT to prevail. Other than data and resources, the cloud has to deal with the business point of view as well. CoT will create more business opportunities, making it bigger target for the attackers. Security, privacy, and specially, identity protection becomes very important in hybrid clouds, where there is an essence of private and public clouds, used by businesses [21]. In CoT, heterogeneous networks will be involved, which support different types of data and services. The network must have the flexibility to support all types of data, according to their requirements, with QoS support [21].

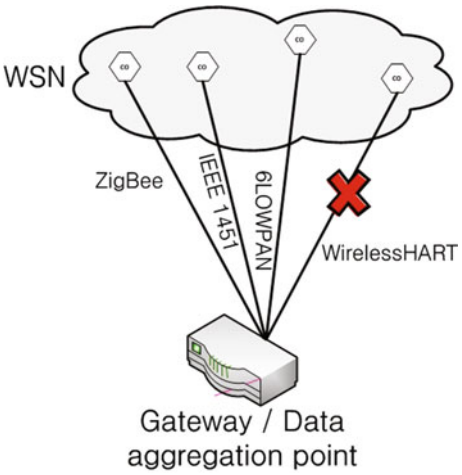


Fig. 3 Protocol support, illustrative scenario

7.1 Protocol Support

For different things to be connected to the Internet, different protocols will have to coexist. Even if there are homogenous entities, for example a sensor IoT or Wireless Sensors Network, then there is still a possibility that sensors use different protocols, such as WirelessHART, ZigBee, IEEE 1451, Constrained Application Protocol (CoAP), and 6LOWPAN. As shown as an illustrative scenario in Fig. 3, some of the protocols will be supported by the gateway device, while some other protocols might not be. With CoT, this problems is going to increase, specially because of mobile cloud computing accessibility. With smartphones and tablet computers, when various healthcare service and other sensors based applications are accessed, protocol support is going to play an important role.

It all depends upon the gateway as well as the sensor being used. From the user's perspective, cheaper or easily available sensor would be a preference. Consequently, it cannot be guaranteed whether a newly added sensor will be successfully configured or not. One of the solutions to this kind of problem is mapping of standardized protocols in the gateway.

7.2 Energy Efficiency

With the omnipresence of sensor networks and their connectivity with the cloud, this will inevitable lead to a lot of data communications, which consumes a lot of power. A typical wireless sensor node is composed of four components: sensing unit, processing unit, transceiver, and power unit. In case of video sensing, video encoding and decoding, power plays a vital role. Normally, video encoding is more complex,

as compared to decoding. The reason behind this is that for efficient compression, the encoder has to analyze the redundancy in the video [14]. It is not going to be suitable to have a temporary power supply, like batteries and have to replace them every now and then. With billions of sensors and low power devices, it is beyond possibility. Having efficient usage of energy and rather permanent power supply would be required. There should be means for sensors to generate power from the environment, like solar energy, vibration, and air [18]. Likewise, effective sleep mode can be very handy in this regard as well. Another solution presented in [5] is bringing cloud resources locally, known as Fog Computing. Fog refers to a localized cloud, which can be used for process offloading purpose for the underlying IoT devices.

7.3 Resource Allocation

When IoTs of entirely different and unexpected things would be asking for resources in a cloud, resource allocation would be a challenge. In fact, it would be very difficult to decide how much a particular resource may be required by an entity or a particular IoT. Depending upon the sensor and the purpose for which sensor is being used, the type, amount, and frequency of data generation, resource allocation has to be mapped. Sending a sample packet from the newly added node can also be useful. One of the solutions is to bring a middleware, like Broker or Fog [5], which can perform all the resource management. Resource management algorithms can be implemented on the middleware and all the underlying devices are handled accordingly. With CoT, devices are going to communicate with the cloud. Therefore, cloud resources can also be managed at middleware layer.

7.4 Identity Management

Communicating nodes over the Internet are identified uniquely. When objects are becoming part of Internet (IoT), they also need a unique identification. Similarly, in case of mobile devices, like mobile sensor nodes on vehicles, tablet computers, smartphones, and other objects, need to have identity mapping in the new network they have just entered. With CoT, the sensors become ubiquitously available, making identity more of a concern. Since IPv6 address space is believed to be enough to support even this kind of ubiquitous networking, assigning IPv6 addresses can be more than a reasonable way in this regard.

7.5 IPv6 Deployment

If IPv6 is to be used for the identification of communicating objects, then formal deployment of IPv6 would also be an issue. Unless a proper, standardized, and efficient mechanism of IPv4–IPv6 coexistence is adopted, objects being assigned IPv6 would be of no great benefit. Since IPv4 and IPv6 are not directly interoperable, they have to be made to coexist. Most common mechanism is this regard is tunneling, but it incurs loss of data, because of heterogeneous fields in the headers of both of these IP versions [6, 8, 10, 12]. Tunneling also bears additional overhead of encapsulation and decapsulation, which may affect delay sensitive applications.

7.6 Service Discovery

With Cloud of Things, the cloud manager or broker has the responsibility to discover new services for the users. In IoT, any object can become part of it at any moment and can leave the IoT at any moment. As mentioned earlier, IoT will also be consisting of mobile nodes. It would be an issue to discover new services and their status and update the service advertisement accordingly. For complex and bigger IoTs, there may be a need of IoT manager as well, which can handle the responsibility of managing the status of IoT nodes, track mobile nodes and keep the updated status of existing nodes and newly added nodes of its IoT. A uniform way of service discovery would be required for this purpose.

7.7 Quality of Service Provisioning

As the amount of data increases and the type and unpredictability also comes into play, QoS becomes an issue. At any moment, any type and amount of data can be triggered. It may also be an emergency data as well. Dynamic prioritization of the requests would be required on the cloud side [21]. QoS would mostly be measured in terms of bandwidth, delay, jitter, and packet loss ratio [11]. Depending upon the type of data and its urgency to be sent to the sync node, QoS must be supported. A dynamic end-to-end QoS provisioning mechanism, using the Flow Label of IPv6 and Multi-Protocol Label Switching (MPLS) is discussed in [11].

7.8 Location of Data Storage

Location also matters for critical and latency or jitter sensitive services. Time sensitive data, like video, should be stored at the closest possible physical location to the user, so that delay is minimized. For multimedia data, nearest possible virtual storage server must be allocated. Figure 4 depicts an illustrative scenario.

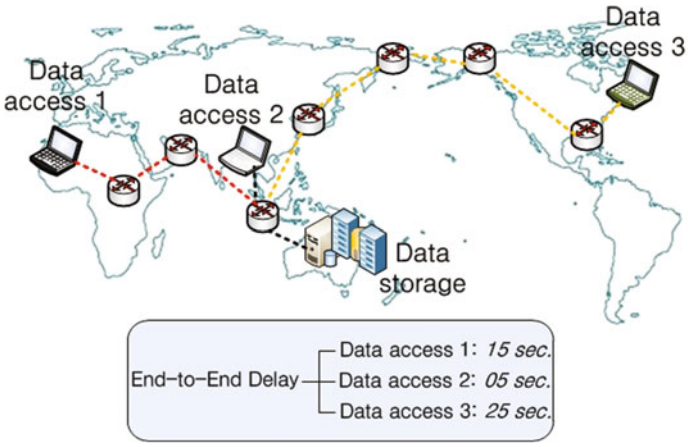


Fig. 4 Location of data storage and its effect

7.9 Security and Privacy

Security and privacy will become more of an issue with the kind of ubiquitous computing we are going to have in future. Data security would be an issue on the IoT side as well as on the cloud side. Similarly, in terms of privacy, more concern would be there. On Feb 01, 2013, it was read on The Independent [3], stating, "British internet users' personal information on major 'cloud' storage services can be spied upon routinely by US authorities". Thus, sensitive or private data must also be stored in a virtual storage server located inside the users country or trusted geographical domain, which can be a friendly country as well.

7.10 Communication of Unnecessary Data

When anything would be able to connect to the Internet and generate data, there is a possibility that at some stage it is no longer necessary to upload the data to the cloud or sync devices. Momentarily, the data may not be required. In that scenario, either the device must be stopped from generating data or the gateway device must decide when it is required to stop uploading the data for preserving resources of the network and cloud, for that while. It will also help in efficient utilization of power. One example could be a Visual Sensors Network. Since VSNs data is video based, therefore, it is large sized. At times there is no need to upload the captured data in the cloud. The data can be uploaded on a particular trigger. Before that, the data could be stored in a local storage (e.g., Fog), attached with the gateway device. Based on the feedback of the application, the gateway device decides when to upload the data and when not. For this purpose, the gateway device, connecting IoT to the cloud,

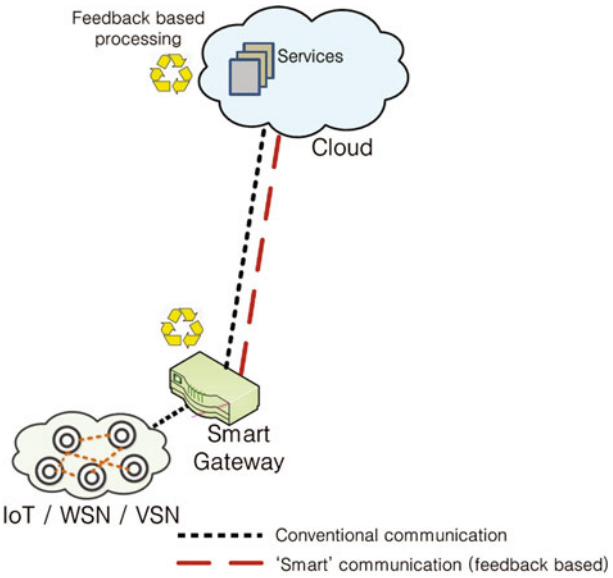


Fig. 5 Smart gateway, communicating data only when it is needed

should be having extra functionality to do a little processing before sending it to the Internet and eventually to the cloud. Based on the feedback from the application, the gateway must decide the timings and type of data to be sent. This kind of gateway, referred to as 'smart gateway', would help in better utilization of the network and cloud resources. The data collected from WSNs and IoTs will be transmitted through gateways to cloud. The received data is then stored in the cloud and from there it is provided as a service to the users. The generic communication of Smart Gateway with cloud and IoTs is presented in Fig. 5.

8 Conclusion

With rapidly increasing IoT services, service management, quality of service, efficiency, and user's satisfaction are becoming crucial tasks. The future lies in the concept of CoT, in which IoTs are amalgamated with cloud computing for better resource managements and service provisioning. In case of multimedia content, a lot of resources are required. With CoT, sensors and other resource constrained devices will be managed through cloud computing. Moreover, for data storage and other tasks, cloud resources would be utilized in this regard. CoT is still in its infancy, therefore, there is no standard architecture available. We have presented some of the key challenges CoT has to deal with, along with their potential solutions. Working

further on those directions would contribute in standardizing CoT. The challenges discussed are therefore future directions in CoT related research and development.

Acknowledgments This work was supported by the IT R&D program of MSIP/IITP [2014044078-003, Development of Modularized In-Memory Virtual Desktop System Technology for High Speed Cloud Service]. The corresponding author is Prof. Eui-Nam Huh.

This research was also supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2010-0020725). The corresponding author is Prof. Eui-Nam Huh.

References

1. Rise of the embedded internet. In: White Paper
2. Moving to the media cloud. In: Viewpoint Paper, November 2010
3. British internet users' personal information on major 'cloud' storage services can be spied upon routinely by us authorities. http://www.independent.co.uk/life-style/gadgets-and-tech/news/british-internet-users-personal-information-on-major-cloud-storage-services-can-be-spied-upon-routinely-by-us-authorities-8471819.html (2015)
4. IEEE 802.14 WPAN TASK GROUP 4 (tg4). http://www.ieee802.org/15/pub/TG4.html (2015)
5. Aazam, M., Huh, E.-N.: Fog computing and smart gateway based communication for cloud of things. In: The Proceedings of IEEE Future Internet of Things and Cloud (FiCloud), Barcelona, Spain, pp. 27–29 (2014)
6. Aazam, M., Huh, E.-N.: Impact of IPv4-IPv6 coexistence in cloud virtualization environment. Annals of Telecommunications-annales des Télécommunications, **69**(9–10), 485–496 (2014)
7. Aazam, M., Phuoc Hung, P., Huh, E.-N.: Smart gateway based communication for cloud of things. In: 2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), pp. 1–6. IEEE (2014)
8. Aazam, M., Khan, I., Alam, M., Qayyum, A.: Comparison of IPv6 tunneled traffic of teredo and ISATAP over test-bed setup. In: 2010 International Conference on Information and Emerging Technologies (ICIET). IEEE (2010)
9. Aazam, M., Khan, I., Abdullah Alsaffar, A., Huh, E.-N.: Cloud of things: integrating internet of things and cloud computing and the issues involved. In: 2014 11th International Bhurban Conference on Applied Sciences and Technology (IBCAST). IEEE (2014)
10. Aazam, M., Atif Hussain Shah, S., Khan, I., Qayyum, A.: Deployment and performance evaluation of Teredo and ISATAP over real test-bed setup. In: Proceedings of the International Conference on Management of Emergent Digital EcoSystems, pp. 229–233. ACM (2010)
11. Aazam, M., Syed, A.M., Huh, E.-N.: Redefining flow label in IPv6 and MPLS headers for end to end QOS in virtual networking for thin client. In: 2013 19th Asia-Pacific Conference on Communications (APCC), pp. 585–590. IEEE (2013)
12. Aazam, M., Syed, A.M., Shah, S.A.H., Khan, I., Alam, M.: Evaluation of 6to4 and ISATAP on a test LAN. In: 2011 IEEE Symposium on Computers & Informatics (ISCI), pp. 46–50. IEEE (2011)
13. Buyya, R., Ranjan, R., Calheiros, R.N.: InterCloud: utility-oriented federation of cloud computing environments for scaling of application services. In: Algorithms and Architectures for Parallel Processing, pp. 13–31. Springer, Berlin (2010)
14. Chen, Y.-K.: Challenges and opportunities of internet of things. In: 2012 17th Asia and South Pacific, Design Automation Conference (ASP-DAC), pp. 383–388. IEEE (2012)
15. Deelman, E., Singh, G., Livny, M., Berriman, B., Good, J.: The cost of doing science on the cloud: the montage example. In: Proceedings of the 2008 ACM/IEEE Conference on Supercomputing, p. 50. IEEE Press (2008)

16. Distefano, S., Merlino, G., Puliafito, A.: Enabling the cloud of things. In: 2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), pp. 858–863. IEEE (2012)
17. Cenk Erdil, D.: Autonomic cloud resource sharing for intercloud federations. Future Gener. Comput. Syst. **29**(7), 1700–1708 (2013)
18. Evans, D.: The internet of things: how the next evolution of the internet is changing everything. In: CISCO White Paper 1 (2011)
19. Fox, G.C., Kamburugamuve, S., Hartman, R.D.: Architecture and measured characteristics of a cloud based internet of things. In: 2012 International Conference on Collaboration Technologies and Systems (CTS), pp. 6–12. IEEE (2012)
20. Grozev, N., Buyya, R.: Inter-cloud architectures and application brokering: taxonomy and survey. Softw.: Pract. Exp. **44**(3), 369–390 (2014)
21. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (IoT): a vision, architectural elements, and future directions. Future Gener. Comput. Syst. **29**(7), 1645–1660 (2013)
22. Mehedi Hassan, M., Song, B., Huh, E.-N.: A framework of sensor-cloud integration opportunities and challenges. In: Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication, pp. 618–626. ACM (2009)
23. Ibrahim, S., He, B., Jin, H.: Towards pay-as-you-consume cloud computing. In: 2011 IEEE International Conference on Services Computing (SCC), pp. 370–377. IEEE (2011)
24. Jrad, F., Tao, J., Streit, A.: SLA based service brokering in intercloud environments. CLOSER **2012**, 76–81 (2012)
25. Jrad, F., Tao, J., Streit, A.: A broker-based framework for multi-cloud workflows. In: Proceedings of the 2013 International Workshop on Multi-cloud Applications and Federated Clouds, pp. 61–68. ACM (2013)
26. Kaewpuang, R., Niyato, D., Wang, P., Hossain, E.: A framework for cooperative resource management in mobile cloud computing. IEEE J. Sel. Areas Commun. **31**(12), 2685–2700 (2013)
27. Khan, R., Ullah Khan, S., Zaheer, R., Khan, S.: Future internet: the internet of things architecture, possible applications and key challenges. In: FIT, pp. 257–260 (2012)
28. Kortuem, G., Kawsar, F., Fitton, D., Sundramoorthy, V.: Smart objects as building blocks for the internet of things. IEEE Internet Comput. **14**(1), 44–51 (2010)
29. Kovatsch, M., Mayer, S., Ostermaier, B.: Moving application logic from the firmware to the cloud: towards the thin server architecture for the internet of things. In: 2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), pp. 751–756. IEEE (2012)
30. Liu, W., Zhao, X., Xiao, J., Wu, Y.: Automatic vehicle classification instrument based on multiple sensor information fusion. In: Third International Conference on Information Technology and Applications. ICITA 2005. vol. 1, pp. 379–382. IEEE (2005)
31. McDermott-Wells, P.: What is bluetooth? IEEE Potentials **23**(5), 33–35 (2004)
32. McNamara, L., Nahas, B.A., Duquennoy, S., Eriksson, J., Voigt, T.: Demo abstract: Sicsthsense-dispersing the cloud (2014)
33. Meingast, M., King, J., Mulligan, D.K.: Embedded RFID and everyday things: a case study of the security and privacy risks of the us e-passport. In: IEEE International Conference on RFID, 2007, pp. 7–14. IEEE (2007)
34. Park, K.-W., Han, J., Chung, J., Park, K.H.: Themis: a mutually verifiable billing system for the cloud computing environment. IEEE Trans. Serv. Comput. **6**(3), 300–313 (2013)
35. Parwekar, P.: From internet of things towards cloud of things. In: 2011 2nd International Conference on Computer and Communication Technology (ICCCT), pp. 329–333. IEEE (2011)
36. Rogers, O., Cliff, D.: A financial brokerage model for cloud computing. J. Cloud Comput. **1**(1), 1–12 (2012)
37. Sehgal, A., Perelman, V., Kuryla, S., Schonwalder, J.: Management of resource constrained devices in the internet of things. IEEE Commun. Mag. **50**(12), 144–149 (2012)
38. Tei, K., Gurgen, L.: Clout: cloud of things for empowering the citizen clout in smart cities. In: 2014 IEEE World Forum on Internet of Things (WF-IoT), pp. 369–370. IEEE (2014)

39. Uckelmann, D., Harrison, M., Michahelles, F.: Architecting the Internet of Things. Springer Science & Business Media, Berlin (2011)
40. Villegas, D., Bobroff, N., Rodero, I., Delgado, J., Liu, Y., Devarakonda, A., Fong, L., Masoud Sadjadi, S., Parashar, M.: Cloud federation in a layered service model. J. Comput. Syst. Sci. **78**(5), 1330–1344 (2012)
41. Wang, W., Niu, D., Li, B., Liang, B.: Dynamic cloud resource reservation via cloud brokerage. In: 2013 IEEE 33rd International Conference on Distributed Computing Systems (ICDCS), pp. 400–409. IEEE (2013)
42. Wu, M., Lu, T.-J., Ling, F.-Y., Sun, J., Du, H.-Y.: Research on the architecture of internet of things. In: 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), vol. 5, pp. V5–484. IEEE (2010)
43. Xiao, Z., Song, W., Chen, Q.: Dynamic resource allocation using virtual machines for cloud computing environment. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1107–1117 (2013)
44. Yang, K., Jia, X.: An efficient and secure dynamic auditing protocol for data storage in cloud computing. IEEE Trans. Parallel Distrib. Syst. **24**(9), 1717–1726 (2013)
45. Yang, Y., Zhou, Y., Liang, L., He, D., Sun, Z.: A service-oriented broker for bulk data transfer in cloud computing. In: 2010 9th International Conference on Grid and Cooperative Computing (GCC), pp. 264–269. IEEE (2010)
46. Yuriyama, M., Kushida, T.: Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing. In: 2010 13th International Conference on Network-Based Information Systems (NBiS), pp. 1–8. IEEE (2010)
47. Zhou, J., Leppanen, T., Harjula, E., Ylianttila, M., Ojala, T., Yu, C., Jin, H., Tianruo Yang, L.: Cloudthings: a common architecture for integrating the internet of things with cloud computing. In: 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design (CSCWD), pp. 651–657. IEEE (2013)
48. Zhu, W., Luo, C., Wang, J., Li, S.: Multimedia cloud computing. IEEE Signal Process. Mag. **28**(3), 59–69 (2011)