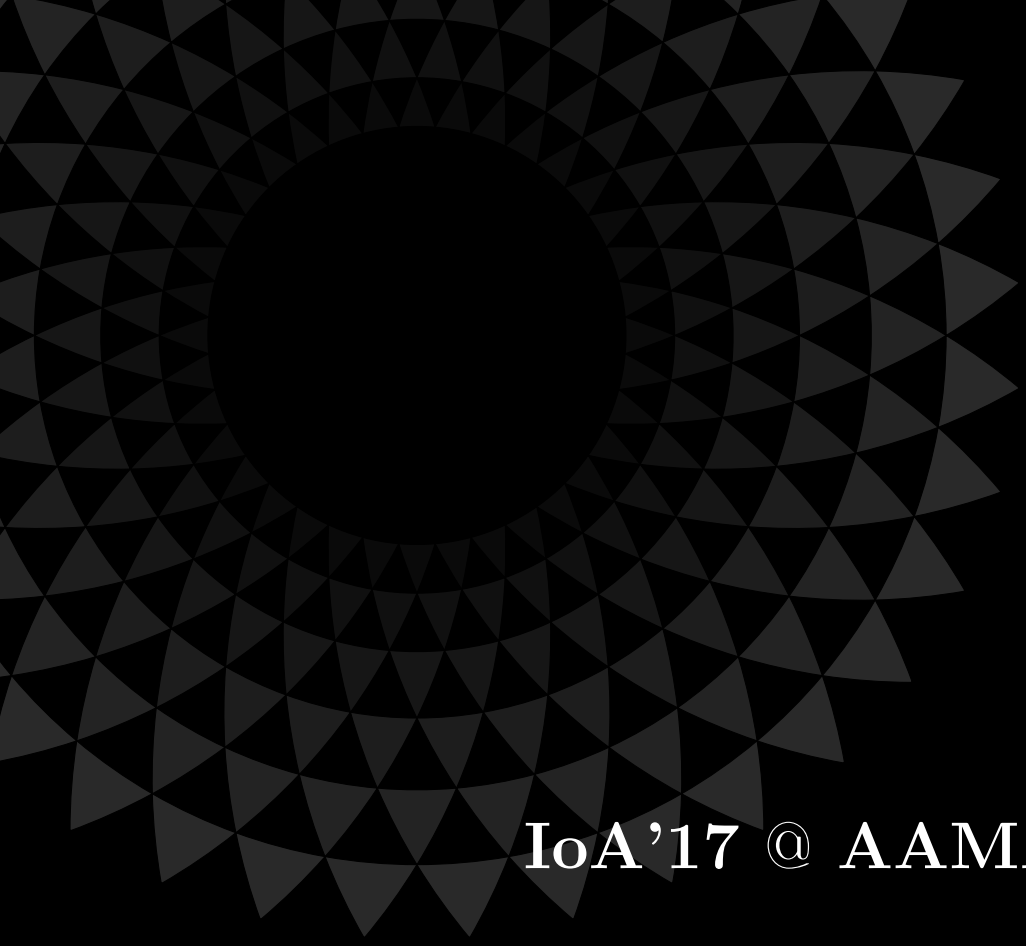


IoA'17 @ AAMAS

São Paulo, Brazil

May 8, 2017

Saad Alqithami
Marco Lützenberger (Eds.)

IoA 2017
Internet of Agents

2^{nd} International Workshop,
São Paolu, Brazil
May 8, 2017

Pre-Proceedings (Informal)

Preface

Internet of Things (IoT) has recently gained much attention due to the overwhelming advantages it brings to our daily lives. The metaphor Things in an IoT does not only represent a variety of hardware, such as simple sensors, actuators or complex devices (e.g., vehicles, charging stations, heating control) it can also comprise pure software instances that might be service-oriented or data-driven. Thus, huge interoperable networks evolve producing large amounts of data and providing a tremendous quantity of services that are supposed to be handled in a distributed manner. An IoT offers the possibility to develop completely new cross-domain solutions but also comes with a lot of challenges for areas such as data aggregation, automated service selection or distributed planning.

On the confluence between IoT and Multi-Agent Systems (MAS), there is a strong ongoing trend for the IoT to adapt to the intelligent nature/architecture of the MAS that we termed as "Internet of Agents" (IoA). The IoA workshop emphasizes on the current range of nascent network centric agents that operate on IoT infrastructures. From a broader point of view, this includes agents' efforts to form collaborative units as well as the techniques and methodologies for constructing online agents that represent interests of their actual counterparts (e.g., hardware devices or sensors) and take actions autonomously. This workshop will reflect the impacts of the dynamic network proliferation and start of smart agent systems that exploit and explore opportunities heralded by the fast social-pace of interconnectivity. From a narrow point of view, we aim to collect, to present, and to discuss (practical) applications and cognitive foundations surrounding agent-based approaches that successfully overcome well-known challenges of software that operates on IoT environments in order to identify promising scenarios that motivate a broader use of agent-technology for the development of IoT-software.

This pre-proceedings volume contains papers to be presented at the IoA workshop 2017: the Second International Workshop on Internet of Agent held at AAMAS 2017, the International Conference on Autonomous Agents and Multi-Agent Systems - May 8, 2017, São Paulo, Brazil. The workshop attracted nine submissions. Each submission was reviewed by at least three members of the Program Committee and finally lead to the acceptance of six papers (four are regular-papers and the other two are position-papers) to be presented and discussed during the workshop time. The workshop will also feature a tutorial of an AR-Application (JIAC-V) and a live demo for the participants.

May 2017

Saad Alqithami and Marco Lützenberger
Program Co-Chairs
IoA'17

Organization

The second international workshop on the Internet of Agents (IoA) is organized in conjunction with the sixteenth international conference on Autonomous Agents and Multi-Agent Systems (AAMAS) in São Paulo, Brazil - May 8, 2017.

Workshop Co-Chairs

Saad Alqithami	(Albaha University, Saudi Arabia)
Marco Lützenberger	(Technische Universität Berlin, Germany)

Program Committee

Jim Chen	(DoD National Defense University, United States)
Saurabh Mittal	(Technische Universität Berlin, Germany)
Henry Hexmoor	(Southern Illinois University, United States)
Nils Masuch	(Technische Universität Berlin, Germany)
Pablo Hernandez-Leal	(Centrum Wiskunde & Informatica, Netherland)
Predrag Tošić	(Washington State University, United States)
Johannes Fhndrich	(Technische Universität Berlin, Germany)
Haifeng Zhang	(Carnegie Mellon University, United States)

Table of Contents

Internet of Agents: From Set of Autonomous Agents to Network Object

Vladimir Gorodetsky*

St. Petersburg Institute for Informatics and Automation, Russian Academy of Sciences 39, 14-th Liniya, 199178 St. Petersburg, Russia
gor@iias.spb.su

Abstract. Internet of things (IoT) is a concept emphasizing networked viewpoint to the set of objects of physical, virtual and social worlds possessing embedded computational capabilities and interacting with each other and with external environment. The value of this concept is that it considers many diverse entities within common software environment and shared information space enriched with access to cloud resources and Web 2.0 services. To make this concept feasible, the IT professionals proposed to model and implement IoT applications in terms of autonomous agents and multi-agent systems. This framework called recently *Internet of Agents* (IoA) is evaluated now as a long term and productive trend. The paper justifies advantages of IoA framework for IoT applications, presents, for the latter, IoA-based formal model illustrated with a number of modern practically important applications. The paper proposes an architecture for a software-communication infrastructure constituting a common agent environment that supports agent interactions within shared information space thus transforming the agent set into a single whole represented as a networked object. The paper also outlines the set of basic services the infrastructure has to provide for the agents in various use cases and justifies feasibility of the proposed infrastructure for IoA framework as applied to IoT applications.

Keywords: Internet of Things, Autonomous agent, Network centric agents, Self-organized network, Emergent behavior, Coordinated agent behavior, Agent communication and interaction infrastructure.

1 Introduction

Internet of Things (IoT) unites physical, virtual and social objects possessing embedded computational and interactive capabilities enriched with full access to Web 2.0 cloud-based semantic information space set up over communication infrastructure of Internet. For example, IoT can unite, in a single system, cloud-based resources, data and services, software and information of social networks together with human societies existing in them, and sensor networks constituting

* ORCHID 000-003-4481-5052

distributed information sources about states of some real world objects. In fact, IoT provides, for human social environment and industry, with novel intelligent Information Technology (IT) concept of high potential. The novel property of IoT concept is that it considers the set of heterogeneous objects as interacting autonomous entities "living" within shared distributed software and communication environment and shared semantic information space.

Let us emphasize that exactly intensive *interaction capabilities* of the distributed IoT autonomous entities constitute their basic computational and control mechanisms leading to emergent behavior of an IoT application as a single whole. In industrial IoT applications, the total number of such autonomous objects can be numerous, e.g. it can be tens of thousands and even many more. These and other characteristics of IoT applications imply a number of specific requirements to methodologies and tools exploited for their design and implementation.

Intuitively, it is about evidently that the technology relying on distributed autonomous software agents situated in shared software, information and communication environments best fits the requirements to the design and implementation of the IoT applications. This statement is especially correct in cases if the total number of autonomous agents is astronomical, whereas the agents are simple enough. Indeed, the Road Map [1] defines agent-based computing as interaction-based paradigm. Accordingly, agent-based technology is oriented to the class of applications that can be split into relatively independent modules, which internal computations are autonomous and not visible for other modules, whereas module dependencies are implementable through agents message exchange. These features of agent-based computing, actually, best fit the peculiarities of IoT applications and motivate IT professionals to propose and investigate multi-agent system (MAS) framework as potential approach to the design methodology and implementation technology for IoT applications. This specialization of MAS framework and technology is called now as Internet of Agents (IoA) [2].

However, to make the IoA framework potential reality, with regard to the design and the implementation of IoT applications, it is necessary to make, at least, two things. The first is to determine the class of IoT applications for which IoA framework is really convenient and efficient, and the second is to adapt or to update the exiting agent design and implementation technologies to make them well applicable to the IoT applications. Both these subjects are discussed of the paper.

Below, section 2 outlines a wide class of currently hot IoT applications selected as the research subject of the paper, and presents, for the applications of this class, a high-level task statement that is illustrated with several particular examples of such applications. Section 3 discusses the important specific issues associated with IoA framework technology as applied to IoT applications. Section 4 describes the proposed architecture of software and communication infrastructure constituting a shared agent environment that is capable to support for agent interactions within the common software and information spaces.

In fact, such an infrastructure transforms the IoA agent set into a single whole that operates as an emerged network meta-object. Section 4 also outlines the set of basic services that the infrastructure has to provide, for the IoA framework agents, in various use cases. Conclusion summarizes the paper results and justifies feasibility and practical usability of the proposed infrastructure, for IoA framework, as applied to IoT applications.

2 Formal Problem Statement and Examples of IoA-based Models of IoT Applications

2.1 Problem Description and Formal statement

It would be incorrect to state that IoA framework is the best one for the design and implementation of any IoT applications. Nevertheless, one can indicate several specific characteristics of such applications for which the above statement is correct. Preliminary analysis and some small experience makes it possible to determine these characteristics as follows:

1. IoT application objective is to produce services on query basis, whereas web services either may be the target of a query or may be required for informational support for the query;
2. IoT application uses information obtained from many autonomous distributed sources, e.g., from intelligent sensors united in a network and/or from virtual entities like devices monitoring computer network Internet traffic for security purposes, etc., or generated by IoT application software components;
3. IoT application is interaction intensive, i.e., in it, many objects have to interact, e.g., to exchange information, to match some information, to synchronize some collective behavior events and so on. Typical case of intensive interaction is given by a system responsible for coordination and synchronization of collective behavior of multiple distributed autonomous objects managed by a protocol.

The most popular example satisfying the aforementioned features is intelligent large-scale sensor network, in which every sensor is capable to interact, perform local computations and supply for the perceived and preliminary processed information to many consumers. It can be multi-user intelligent sensor network perceiving geophysical data, weather-related and/or ecology-related information. Another example is large-scale multi-purpose distributed surveillance system performing, along with other activities, perimeter security tasks, and so on.

Many logistics applications can also be built as IoT applications, although, at first glance, they are completely different. Below in this section several examples of logistics services are described and, among them, city express delivery service, city post delivery service, Internet shopping service, novel business models of taxi services like ride-sharing, etc. It can be shown that commercial distributed orbital surveillance systems composed of many small satellites collecting, on

request basis, visual information about ground objects can also be built as an example of IoT applications [3]. Modern paradigm of self-organized management of B2B production as well as transportation logistics networks actively exploiting various web services provides with many examples of IoT applications too (see, for example, [4, 5]). The list of examples of IoT applications matching the properties indicated at the beginning of this section can be continued. All these applications can formally be specified in common terms as described below.

Let us assume that $S_u \in \boldsymbol{S}$, where S_u is an order of a random order flow $\boldsymbol{S}$ arriving in a system destined to fulfill these orders on request. For example, S_u can be an order to produce some complex product (e.g., machinery, electronic devices, customized car, etc.), or to provide for a service (to deliver a parcel from a post office to a customer using cross-docking, e.g.). The objects generating orders (customers, clients, software objects, etc.) are denoted below as $O_i \in \boldsymbol{O}$. In general case, the latter can also be the objects either of physical, or of virtual, or of social world. Let us assume that object $O_i \in \boldsymbol{O}$ can be assigned the coordinates $G_i \in \boldsymbol{G}$ and time instant $T_u^{(i)} \in \boldsymbol{T}$ when order S_u is generated by this object. The object coordinates can be given in a coordinate system, e.g., in geographical coordinates $< \mathit{latitude}, \mathit{longitude} >$, or in some local coordinate system like $<$ *street name*, *street address* $>$, and these coordinate systems are uniquely mutually transformable. The orders are generated in real time mode and their lifetime lasts from $T_u^{(i)} \in \boldsymbol{T}$ and till the time instant of its completion or breaking of its fulfillment due to a reason.

On the other hand, it is assumed that many potential order executors/service providers $P_j \in \boldsymbol{P}$ exist, each of them is assigned coordinates $G_j \in \boldsymbol{G}$, and the coordinates can depend on time if the executor or service provider is mobile. Each order can arrive to any executor/service provider, according to a policy constituted by IoT execution/service provision system $\mathcal{R}$.

One more set $\boldsymbol{E}$ of objects E_k, $E_k \in \boldsymbol{E}$, involved in the order fulfillment can exist in the system $\mathcal{R}$. Their role is explained below by several examples.

An important component of the problem statement under description is a software and communication infrastructure $\mathcal{X}$ supporting bidirectional communications and message exchange between all the pairs of entities introduced, i.e. between the pairs $< O_i \in \boldsymbol{O},\ P_j \in \boldsymbol{P} >$, $< P_j \in \boldsymbol{P},\ E_k \in \boldsymbol{E} >$, $< O_i \in \boldsymbol{O},\ E_k \in \boldsymbol{E} >$, $< P_j \in \boldsymbol{P},\ P_s \in \boldsymbol{P} >$, $< E_k \in \boldsymbol{E},\ E_k \in \boldsymbol{E} >$, etc. At that, message exchange can be a multi-step procedure fulfilled according to a protocol.

The objective of the service provision system $\mathcal{R}$ is planning and scheduling of the fulfillment of the orders arriving in real time with required Quality-of-Service (QoS) value. QoS value is evaluated according to a set of indices $\boldsymbol{I} = \{I_1, ..., I_m\}$ and some policy that is capable to constitute order relation "$\geq$" for any pair of $\boldsymbol{I}_r$, $\boldsymbol{I}_s$ to determine the "better" relationship for any two variants of order execution/servicing.

In IoA framework, each entity of the sets $\boldsymbol{S}$, $\boldsymbol{O}$, $\boldsymbol{P}$, and $\boldsymbol{E}$ in the software environment is represented with a software agent of the definite role. Because of such conceptualization, the target execution/service provision task is presented in terms of a set of agents solving the problem through intensive interactions

according to a various protocols implementing different use cases of the whole order execution system operation.

Below a number of relatively simple applications is outlined in order to justify the fact that all of them are of class IoT applications and that they can be specified in terms of common problem statement introduced above.

2.2 Unmanned Car Taxi Service

It is expected that such taxi service can become reality about in ten years. The absence of human driver on car board will force remarkable change of the taxi service business model, including order execution planning and taxi scheduling technologies.

Even now, some advanced taxi service business models try to minimize the role of human factor, while trusting some responsible operations to software agents. One of such models practically exploited in London is described in [6]. In it, every car considered as a *resource* is assigned software agent of the role *Resource agent*, and every customers order interpreted as a *need* is assigned software agent of the role *Need agent*. The planning and scheduling procedures run without human intervention, while using rich negotiations among agents to find the best matching of the resources against the needs, for every order planning and scheduling. Wireless communication environment and access to web services supports for implementation of this business model. In fact, this case is one of the earliest implementation of IoA framework for real-life application. Its main distinction from the future IoA implementation of unmanned car taxi service is that the model presented in [6] operates in centralized mode, whereas future ones are expected to be purely distributed.

It is expected also that, for unmanned car taxi service, the role of interacting software agents will become much richer and more important due to several issues. One of them is that dispatcher has to collaborate with the car on-board agents, but not with a human driver. In addition, it is expected that future client-focused taxi service will operate as B2B network of particular taxi companies, and, thus, client will be capable to online select service without indicating the servicing company. Therefore, the agents will operate and interact in purely distributed environment. This IoA environment will contain many more interacting software agents and it will be a complex task to manage them in the common environment.

Let us show that unmanned car taxi service is a particular case of IoT applications which formal model was introduced above in this section. The following denotations are used below, for unmanned car taxi service case: $S_u \in \boldsymbol{S}$ - a particular order of the taxi service order flow $\boldsymbol{S}$; $G_i \in \boldsymbol{G}$ and $T_0^{(u)} \in \boldsymbol{T}$ are coordinates of client and time instant of the order S_u arrival, respectively; $P_j \in \boldsymbol{P}$ is a call center receiving the order and $E_k \in \boldsymbol{E}$ is an arbitrary unmanned taxi car operating on-line. Thus, one can see that the task in question can be viewed as a particular case of the general IoT servicing problem statement introduced earlier in this section.

Further, let us map, to each object introduced in the above problem statement, a software agent containing object-related information and assigned corresponding role. Let us place the total set of agents into a shared software and communication infrastructure $\mathcal{X}$ supporting agent message exchange and the agent negotiation protocols implementing distributed planning and scheduling procedures. Let us also assume that the aforementioned agents are provided with access to Web 2.0 services (to navigate, to get road traffic-related information, etc.). Then the resulting model of unmanned taxi service system will look as an example of what is called IoA conceptual model of IoT application. Let us remind that, since a lot of clients' orders should be served simultaneously and a lot of unmanned cars can operate at that time in the city, the total number of interacting agents, in this model, can be formidable, e.g. hundreds of thousands and even more. The main objective of the agent interaction solving planning and scheduling task is matching the attributes of the orders against capabilities of taxi cars that will require the very intensive interactions.

2.3 Ride-sharing Taxi Service with Unmanned Cars

This business model of taxi service is currently actively discussed (see, for example, [7]). Its expected advantage is about three-time decrease of needed car number that, in turn, should significantly lower the street traffic intensity. In general, this task is close to the previous one but it needs more intensive interactions due to cross-docking task coordinating in time and space attributes of routs of several taxi cars serving simultaneously several orders (to match the time dependent coordinates of taxi cars including hired ones against current position of a novel client or even several ones).

2.4 Courier Service

This kind of service is currently very popular in big cities and not only here. It is used for direct document delivery, between business offices, post mailing and parcel delivery to addressee, Internet shopping delivery, etc.

Different business models of courier service exist. They differ in using or not of intermediate item storages, in kind of delivery transport, in delivery strategy exploited, and so on. Like taxi service, it can use or not use of cross docking, when several couriers participate in delivery procedure of an item. In such case, couriers have to coordinate their delivery paths in space and time to optimize some service KPIs.

It is simple to see that formal models of courier services practically similar to those ones for taxi services and therefore their formal problem statements are similar.

2.5 B2B-networks for Production Logistics

Networked business is now considered as a perspective business model, at that B2B network is a central model of such business. The European Commission

programs like FP6, FP7, Horizon 2020 actively promote this model especially in what concerns with production and transportation logistics [8]. Indeed, the aforementioned programs support several decades of projects dealing with B2B production networks investigating and developing various aspects of B2B network design and implementation. The same model is the subject of intensive research in the USA and in Pacific regions too.

It is important to note that distributed business model is a long-term and sustainable trend caused by globalization of labor market, specialization of production plants and division of labor. This business model is motivated by the resulting decrease of product cost and improvement of product quality. The novel opportunities for doing of distributed business emerge due to semantic Web 2.0 too. Active development of ontology-based semantic data models well contributing to provision for informational compatibility of different participants of B2B network supports for and strengthens of this trend.

Let us remind that B2B network is understood as a geographically distributed network of companies capable either to produce some kind of products, or possessing some special complimentary technologies, or capable to provide for some classes of services, etc. The companies constituting B2B production or/and transportation network are interested in cooperation since the companies united in the B2B network, as a rule, complement each other in their capabilities and therefore can produce wider range of products for less cost.

From formal problem statement viewpoint, it is assumed that orders for products and/or services can arrive from external environment. The network nodes can bear the orders too. The order source can also be a consumer of the resulting order product/service. The order fulfilment technology is usually constituted of partially ordered set of plain activities each requiring the definite specific technology as well as adequate technical and human resources and capabilities. The B2B network nodes cooperate to perform these activities.

B2B production network is a typical example of IoT application involving physical objects (plants, shops, machinery, sensors, etc.), social objects (workers, managers, engineers, etc.) and virtual objects (software components, communication channels, security software, databases, etc.) operating in shared communication, information and semantic spaces with access to web services.

Abstract problem statement, for B2B network management (order execution allocation, among the network nodes, coordinated scheduling of order fulfilments, supervising of real-time network node performance, etc.) is similar to the other IoT applications described above. However, it is more diverse, in some aspects, it is more complex compared to the aforementioned applications, and that is why it can play the role of a case study/benchmark to discover various peculiarities of usage of IoA framework for IoT applications. For example, it can be used to discover and analyze the basic use cases and typical protocols to be supported by software and communication infrastructure shared by all agents of IoT applications if it is implemented using IoA framework.

3 Internet of Agents Technology: What is Novel?

One can think that multi-agent system (MAS) technology that has thirty-year history of research and developments and, thus, possessing a number of methodologies and software tools can be simply adapted to the requirements of IoA. However, this is not the case and there are many reasons to fault.

The first is that MAS-technology, up to now, unfortunately, has not proposed well-developed and methodologically mature means capable to cope with design and implementation of applications of industrial level. In the recent time, this problem is discussed in a number of papers devoted to the current state-of-the-art with MAS technology for industry [2, 9-12]. These papers indicate this fact as a serious drawback of MAS theory and practice. Indirectly this statement is confidently confirmed by the following well-known fact: industrial IT-community has not accepted MAS-technology so far and exploits it rather exceptionally than "as a rule".

The second reason is that the existing MAS methodologies and supporting software tools are developed for other than IoT classes of applications. Indeed, for many IoT applications, simple but intensively interacting agent community best fits peculiarities of the system to model. Instead of this, the existing MAS technologies based on BDI-model of agents focus their attention on design and implementation of rather complex agents employing sophisticated knowledge structures and reasoning mechanisms with relatively poor interactions. For example, the JADE agent platform [13] that is the most popular one, in the agent community, counts mostly on pairwise agent interactions. Accordingly, to implement interaction protocols specifying multi-step distributed behavior of many agents, special software should be developed.

The third reason is that, in fact, IoT applications deal not only with a set of objects but also with a single whole that is a network meta-object comprising a great number of networked simple autonomous agents. Accordingly, the IoA framework as applied to the conceptual modeling and software implementation of IoT applications should take into consideration two-level structure of the object it deals with. Indeed, on the one side, it deals with the set of spatially distributed heterogeneous objects of physical, social, and virtual natures specified in terms of a large set of massively interacting agents. On the other side, it is a single whole, an emergent networked meta-object, exhibiting emergent behavior in result of numerous *local interactions of agents*. Let us note that such class of objects is studied in the multi-agent self-organization field.

This is very important fact because the design and implementation technology for self-organized MAS completely differs from such technology for other classes of MAS applications. It is well known that standard MAS technology exploits top-down design of applications, whereas, for self-organized MAS applications, the bottom-up design technology is used. The reason of the bottom up design technology is the fact that the behavior of self-organizing applications is determined by local agent interaction and it is not feasible to try to predict the emergent behavior of the emerged meta-object.

It is worth noting that, in the real-time taxi management system mentioned

in previous section and referred to [6], the *local interactions* of pairs of agents are the core of the service planning and scheduling algorithms. In such interactions, the so-called *Resource agent* representing a taxi car capabilities and objectives is matched against *Need agent* representing the requirements of order to taxi service. The matching target is to check compatibility particular resource presented by *Resource agents* and requirements of the order under planning and scheduling represented by *Need agent*. The total number of the agent local interactions realizing such pairwise matching is very big and the authors reported that the system operates in self-organizing mode, in result of such numerous local interactions. It is no doubt that IoA implementation of typical IoT applications of the class under consideration will exhibit self-organizing behavior too.

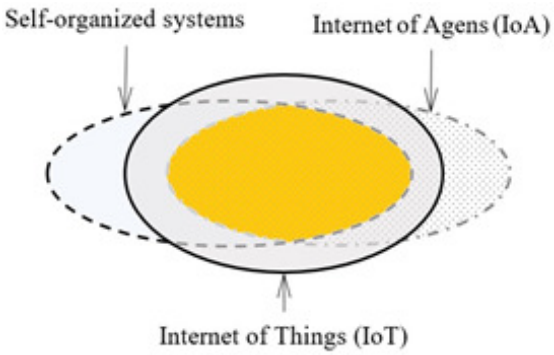


Fig. 1. Internet of Things, Internet of Agents and Self-organized systems. Yellow area corresponds to the IoT applications discussed in the paper

It is not the fact that the described IoA framework will be the best model and provide convenient technology for all classes of IoT applications. More probably, not all IoT applications will exhibit self-organizing behavior if they designed and implemented in IoA framework. Fig. 1 illustrates graphically the author as imagination of set-theoretical relationships among three statements: "Application belong to IoT class", "IoA best fits an application as conceptual and technological framework", and "Implemented application exhibit self-organizing behavior". Intersection of these sets determines the set of IoT applications considered in this paper.

The motivation of this choice is as follows. It is well known that agent-based self-organized systems are capable to successfully perform large-scale applications, whereas other frameworks completely fail with such applications. Many examples justifying advantages of practical use of multi-agent self-organization approach are presented in [14-16].

However, a number of challenging problems arises and will arise in practical use and implementation of IoA framework for IoT applications under discussion. Perhaps, the most important of them is to build a software and communication

infrastructure that turns the set of heterogeneous distributed agents into a single whole, i.e. into a system. In fact, design and implementation of such infrastructure is the key issue of IoA framework-based technology for IoT applications.

In the multi-agent technology world, agent platform can be viewed as a prototype of such infrastructure. However, the existing platforms possess very limited capabilities. For example, the most popular agent platform, JADE [13], can support only centralized *Yellow pages* service, whereas infrastructure supporting IoA framework has to be distributed. Further, a typical feature of IoA model of IoT application is intensive interactions of many agents, according to a multitude of protocols implementing various use cases. However, JADE mostly supports pairwise agent interactions in dialog style and, therefore, special software to implement agent interaction protocols has to be developed in case of using JADE platform.

Software and communication infrastructure of IoA framework has to supply, for interacting agents, with many additional services whose the existing agent platforms are not capable to provide. Next section analyzes this issue in more details.

4 Distributed Software and Communication Infrastructure Supporting IoA Framework Technology

It was emphasized above that the infrastructure discussed in the paper is intended to transform the set of autonomous agents into the network meta-object operating as a single whole, which behavior emerges as the result of the local interactions of autonomous agents. Let us consider the particular IoA functions the infrastructure has to support according to the requirements put by the agents existing inside it. Below, these functions are also called as infrastructure services, or capabilities. It is assumed that the infrastructure services are supplied either for particular agents or for the emerged network meta-object as a whole.

The "on default" assumption is that the infrastructure in question is distributed and consists of a number of local components, the infrastructure *instances*, or *local* infrastructure *instances*. As a rule but not mandatory, each such instance is set up on a separate computer, or operates on embedded computing device of a sensor, or on some appliance, or on mobile device, etc. Therefore, the infrastructure instances constitute together the network-based distributed software and communication infrastructure in which the infrastructure instances are set up on particular nodes.

Since some nodes of the infrastructure network can be mobile (they can move in space and time) and the network as a whole should be open system, new nodes can enter the network along the time scale and, vice versa, some nodes can leave the network from time to time. An assumption is also that network topology can be dynamic not only due to change of the network node population, but also, sometimes, due to mobility of the network nodes and limited communication ranges of wireless communication channels that are typically used by IoT applications.

Let us consider the basic services the infrastructure has to provide for implementation of IoA framework in IoT environment. Let us note that the security service that is mandatory for any IoA infrastructure is not considered below.

4.1 Communication service

Internet constitutes the physical basis for communication service, but due to dynamic topology of the IoT network nodes it cannot use only TCP/IP-based routing. The network has ad-hoc structure, in which every node can have different neighbors at different time instants. The infrastructure communication service has to provide, for agents set up over the IoT nodes, with communication channels to deliver agent messages to this or that addressees, i.e. to other agents located on the same node or on other IoT network nodes. In open ad-hoc network, "the term *peer* represents a computational entity capable to share resources like information, processing, presence, etc. with other peers" [17]. Peer is responsible for mapping of a particular communication channel to particular agent message. Its function is also to support the actual list of peer contacts comprising the pairs <available communication channel, the name of destination peer>, i.e. the list immediate neighbors of the peer, in the other words.

What concerns the p2p routing issue, this task for ad-hoc IoT network is solved by special service, *Yellow page* service providing, for every output message of the node, with a mechanism (protocol) to search for addressee (addressees). Of course, this protocol is not mandatory the TCP/IP protocol, in general case.

4.2 Message addressing service

This service provides for a node output message with an addressee search mechanism designed specifically for open network environment. In general case, this mechanism implements some p2p protocol intended to search for an agent either on addressee name basis, or on the basis of service requested in the message. In both cases, the most frequently exploited mechanisms are based on the use of *Yellow page*- and *White page* services with subsequent use of a p2p routine protocol for addressee search.

For the networks with dynamic topologies, in 2006, FIPA proposed a *Functional Architecture Specification Draft* specifying, along with other issues, reference model for distributed *Yellow page*- and *White page* services [17]. It proposed distributed p2p agent platform implementing the standard FIPA agent services supported by centralized FIPA-compliant platform. This architecture implements complete detachment of agent connection with peer of application layer using concept of overlay network. This detachment admits independent modification of application agents and peer that is set up over the same infrastructure node. Let us note that if communication is organized in p2p style then this layer is also organized as overlay network set up over the TCP/IP-layer.

Recently, several other possibilities designed to support for the message addressing service were proposed. Among them, systems using the concept of actor, various implementations of *Java Messaging System* standard (*JMS* standard) are

proposed.

Akka-library specifically developed for actor-based concept of agents is becoming popular due to the transparent technology designed for distributed agent-based system development. In particular, it is popular due to *Akka*-library module called *Akka Cluster*. The latter provides for a number of mechanisms supporting for open network operation.

Several successful implementations of *JMS* mechanism for open and p2p network operation support are descried in [18]. For example, *Apache ZooKeeper* technology is used for implementation of *Yellow page* services. Although this technology is centralized, the central network node (leader) is selected from the set of other nodes of the network. If *ZooKeeper* leader fails, a new leader is selected via use of a standard consensus protocol. This technology is used by Twitter Company in the *Storm* solution implementing cluster technology for processing of real-time data flows.

A limited own experience has proved the advantages of using *Akka*-library for implementation of distributed message addressing service based on the architecture close to one proposed by FIPA in [17].

4.3 Network openness support

Agent network openness has to be supported by a set of standard protocols according to which any new node can join the network, declare its services and get/establish information about its immediate neighbors and services they are capable to provide. To leave the network, the node has also to run some protocol. In both cases, the infrastructure has to reflect novel information about its updated topology.

These policies can be implemented by various methods. One of them is *JXTA* technology [19] that is *Java*-implementation of the standard developed by *UPnP*-forum [20]. However, its holder, Oracle, does not currently support this technology and that is why the perspectives of the *JXTA* technology are now unclear. Nevertheless, the technology exists and it can be implemented in a renovated manner.

4.4 Service supporting for node informational compatibility

This service plays the important role in providing, for the network nodes, with cooperation and coordination capabilities. Let us note, that, in *European B2B production network Road map* [21], this task is emphasized as top most one, among other objectives of EC programs to 2025. Several examples of applications intensively exploiting cooperation were given in section 2, e.g. cross-docking in taxi service and in courier service.

According to the modern opinion, semantic information models using shared ontology constitute the basis for distributed system information compatibility. Horizon 2020 Program section dealing with the distributed business support, robot collective behavior, etc. supports decade of the projects specifically devoted to the research and development in this area (see, for example, [22]).

Additional means destined for achievement of informational compatibility of nodes of an IoT application are based of use of standard interfaces, and network protocols supporting distributed agent message exchange and use of standard languages for specification of message syntax and its content. The *ACL*-language that is the de-facto standard in the agent community is an example [23].

4.5 Support for interaction protocols in various application use cases (scenarios)

The task objective is to support distributed operation of a large number of agents in cooperative and/or coordinated manner. Particular example is cross-docking case of delivery, in distributed courier service. More general example is planning of distributed fulfilment of a complex business process represented by partially ordered set of operations if a single company cannot execute this business process in full. The essence of this planning procedure is assignment of particular operations of a complex business process to concrete executors (network nodes) that possess the required technological capabilities and resources. An example of such distributed planning algorithms (protocols) is given in [24, 25].

In various IoT applications, different application-specific protocols can be required. Self-organizing IoT applications formalized within IoA framework also require using local protocols determined by the kind of self-organization mechanisms selected. Contract Net protocol that is frequently exploited as market-based self-organization protocol is a showcase of such protocols [26].

General architecture of the software and communication infrastructure is depicted in Fig. 2 [25]. It was developed specifically for B2B production self-organizing network. It comprises the following components:

- *Akka* - library of actors; the infrastructure exploits it for implementation of basic functions supporting for message transport (*White page* and *Yellow page* services, in particular);
- *IoA_Platform_Manager* - it is the main actor supervising with operations of all services available at the node-based instance of the distributed infrastructure;
- *NomadicLinksProtocol* - it is a layer of message transport mechanism supporting for referential integrity of distributed data stricture storing *Yellow pages* table of message addressing service;
- *Partial_Order_Library* - it is a software library set up over *NomadicLinkProtocol* that implements distributed algorithm manipulating with complex business processes of partial order structures to be planned, scheduled and fulfilled in distributed mode. Example of functions of this library can be found in [24];
- *Scheduling Protocol* - it implements the distributed scheduling algorithm while using the particular algorithm of the library Partial_Order_Library;
- *Yellow_Pages* - distributed Yellow pages is exploited for search for agents (by name or by services they are capable to provide). The mechanisms underlying *Yellow page* service provision are well developed and well known, from multi-agent literature;

– *Planning_Protocol* - protocol implementing the allocation of parts of the decomposed order to the network nodes that should fulfill these parts of the order. An example of such protocol exploiting Contract Net protocol and its implementation is given in [27];
– *Control Protocol* - it is the protocol performing monitoring and supervising of real-time operation of the order fulfilment system including management of exceptional situations;
– *Execution System* - it is a meta-protocol coordinating the operations of all the protocols, in the local node;
– *Node API* - it provides, for application agents of the node, with API of other node-located infrastructure component.

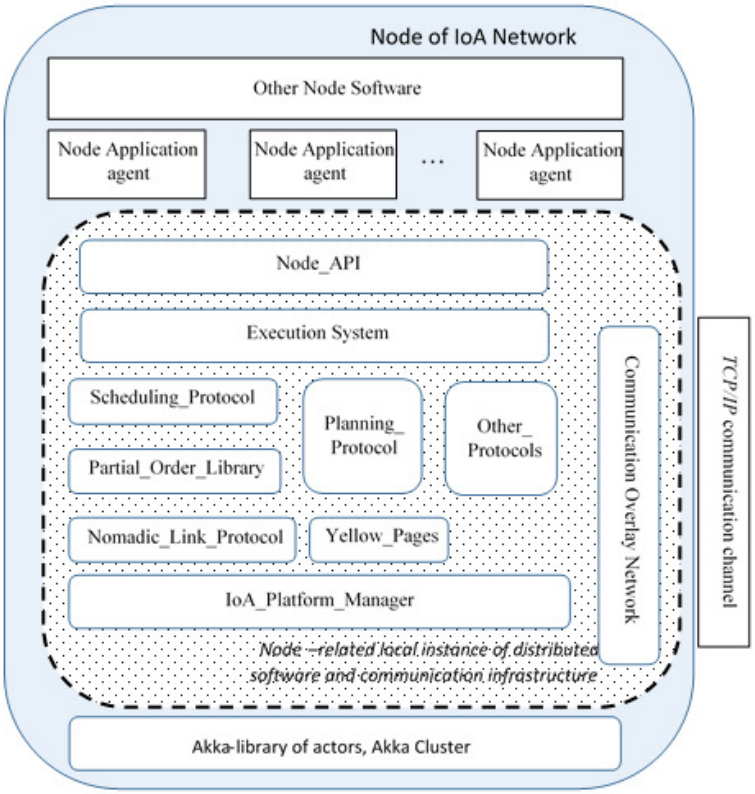


Fig. 2. Typical architecture of a local component of the distributed software and communication infrastructure supporting for distributed operation of IoA system and its interactions with node-located application agents and other node-located software components

The architecture of the local instance of the distributed software and communication component depicted in Fig. 2 presents only its basic functionalities and, in real-life case, it should be more complex and more sophisticated. It also does not cover expected application-dependent peculiarities, but it provides for some understanding what this infrastructure is about, in its essence, and what component it can consist of.

5 Conclusion

IoT concept of application is a whiz-bang framework being intensively developed and received general recognition during the recent times. It is designed to model large-scale applications involving, in operation, a great number of diverse objects of physical, virtual and social worlds. Usually, these objects are either the sources or consumers of highly diverse information. The objects and semantic services of Web 2.0 set up over communication infrastructure of Internet can also be the components of IoT applications. These applications are of great social and commercial value.

Currently, active research and development are directed to search for formal framework for IoT application conceptual modelling, design and implementation. The recently introduced concept of IoA is one of such potentially efficient and effective frameworks.

The paper outlines the essence, specific characteristics, properties and potential capabilities of the IoA framework as applied to IoT applications, presents an abstract problem statement of an IoT application in terms of IoA conceptualization and demonstrates the latter by several particular IoT applications having the similar problem statements, within IoA framework.

It is shown in the paper that IoA framework reduces IoT application conceptual model to a large-scale meta-object of network structure that operates as a single whole, and which behavior emerges from multiple local interactions of numerous number of simple agents constituting typical self-organizing system.

The paper proposes architecture of the main component of IoT applications modelled and implemented based on IoA framework that is software and communication infrastructure forming an environment within which the agents are born, live, and die. Exactly this infrastructure transforms the set of agents into a single whole that is a meta-object of the network structure. Analysis of basic services that the infrastructure has to provide in typical use cases of IoT application operations is also outlined.

The general conclusion of the paper is that, even now, many particular solutions that can support IoA modelling and implementation issues of IoT applications exist, thus advocating for the IoA framework feasibility.

Acknowledgment. The presented research is supported by the Russian foundation for Basic Research (Project # 14-07-00493). Some results of this research where received in the framework of two other projects: SPIIRAS Project # 0073-2015-0003 and Project # 214 of the Program 1.5 of the Russian Academy

of Sciences Presidium. I also express my appreciation to my former Ph.D. student O. Bukhvalov participated in development of architecture described in section 4 and its experimental software implementation as applied to the B2B production network.

References

1. Luck M., et al.: Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing). AgentLink: http://www.agentlink.org/roadmap (2005)
2. Yu H., Shen Z., Leung C.: From Internet of Things to Internet of Agents. In Proceedings of 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, IEEE Computer Society, 2013, pp. 1054-1057 (2013). doi 10.1109/GreenCom-iThings-CPSCom.2013.179.
3. Gorodetsky V., Karsaev O.: Distributed Surveillance System Based on Self-organized Collective Behavior of Small Satellite Cluster. Transactions of South Federal University, Technical Sciences, No. 3 (2017). P. 141-155 (in Russian).
4. JCtransNet. http://www.jctrans.net.
5. RosettaNet. www.rosettanet.org.
6. Rzevsky G., Skobelev,P. : Managing complexity. WIT Press, London-Boston. (2014). 156 p.
7. Imagine New York City With 3,000 Taxis Instead of 13,000. http://spectrum.ieee.org/cars-that-think/transportation/efficiency/imagine-new-york-city-with-only-3000-minivans-instead-of-taxis-it-could-work.
8. Data base of EC Programs FP-5, FP-6, FP-7 and Horizon-2020. http://cordis.europa.eu/search/advanced_en?projects.
9. Gorodetsky V., Bukhvalov O., Sobelev P., Mayorov I.: Survey on Modern State-of-the-Art and Perspectives of Industrial Applications of Multi-Agent Systems. Large-scale System Control (2017). (in Russian).
10. Mueller J., Fisher K. Application Impact of Multiagent Systems and Technologies: A Survey In Agent-Oriented Software Engineering book series. Springer, pp. 1-26 (2013) DOI 10.1007/978-3-642-54432-3_3
11. DeLoach, S.A.: Moving multi-agent systems from research to practice, International Journal Agent-Oriented Software Engineering. Vol. 3. No. 4, pp. 378-382 (2009) DOI 10.1504/IJAOSE.2009.025315
12. Letaio, P., Vrba, P.: Recent Developments and Future Trends of Industrial Agents // Proceedings of HoloMAS-2011, Holonic and Multi-Agent Systems for Manufacturing, Volume 6867 of the series Lecture Notes in Computer Science. Springer Verlag, pp. 15-28. (2011-2012) DOI 10.1007/978-3-642-23181-0_2
13. JADE. http://sharon.cselt.it/projects/jade.
14. Gorodetsky V.: Multi-agent self-organization in B2B Networks. In Proceedings of XII Russian symposium on Control Systems. Moscow, 14-16 June 2014. Trapesnikovs Institute of Control Problems, http://vspu2014.ipu.ru/proceedings/vspu2014.zip. pp. 8954 - 8965 (2014).
15. Gorodetskii, V.I.: Self-Organization and Multiagent Systems: I. Models of Multiagent Self-Organization. Journal of Computer and Systems Sciences International, Vol. 51, No. 2, Pleiades Publishing, Ltd., 2012, pp. 256-281 (2012) DOI 10.1134/S106423071201008X

16. Gorodetskii, V.I.: Self-Organization and Multiagent Systems: II. Applications and the Development Technology. Journal of Computer and Systems Sciences International, Vol. 51, No. 3. Pleiades Publishing, Ltd., 2012, pp. 391-409 (2012) DOI 10.1134/S1064230712020062
17. Nomadic Agent Working Group, http://www.fipa.org/subgroups/P2PNA-WG-docs/P2PNA-Spec-Draft0.12.doc.
18. Vollset E., Ingham D., Ezhilchelvan P. JMS on Mobile Ad-hoc Networks. http://www.cs.cornell.edu/ einar/pubs/vollset03jmsmanet.pdf.
19. JXTA. https://en.wikipedia.org/wiki/JXTA.
20. UPnP Forum http://upnp.org/specs.
21. Future Internet Enterprise Systems (FInES): Research Roadmap 2025, http://cordis.europa.eu/fp7/ict/enet/documents/fines-research-roadmap-v30_en.pdf.
22. http://swarms.e.
23. ACL - Agent Communication Language. http://fipa.org/specs/fipa00061/SC00061G.pdf.
24. Bukhvalov O., Gorodetsky V., Karsaev O, Koudryavtsev G., Samoylov V. Privacy-Preserved Distributed Coordination of Production Scheduling in B2B Networks: A Multi-agent Approach. In Proceedings of 7-th IFAC Conference on Manufacturing Modeling, Management, and Control, 2013, Volume 7, Part 1, pp. 2122-2127 (2013).
25. Gorodetsky, V., Bukhvalov, O.: Self-organized B2B Networks: Concept, Architecture and Algorithmic Support. In: Mechatronics, Automation and Control, No 5 (2017).
26. Smith, G.: Contract Net Protocol: High-level Communication and Control in a Distributed Problem Solver // IEEE Transactions on Computers, C-29 (12), pp. 1104-1113. (1980)
27. Contract Net Protocol. http://www.fipa.org/specs/fipa00030/SC00030H.html.

A Role-Based Approach for Orchestrating Emergent Configurations in the Internet of Things

Radu-Casian Mihailescu, Romina Spalazzese, Paul Davidsson, Clint Heyer

Department of Computer Science, Malmö University
Internet of Things and People Research Center,
Malmö, Sweden
{radu.c.mihailescu, romina.spalazzese, paul.davidsson, clint.heyer}@mah.se

Abstract. The Internet of Things (IoT) is envisioned as a global network of connected things enabling ubiquitous machine-to-machine (M2M) communication. With estimations of billions of sensors and devices to be connected in the coming years, the IoT has been advocated as having a great potential to impact the way we live, but also how we work. However, the connectivity aspect in itself only accounts for the underlying M2M infrastructure. In order to properly support engineering IoT systems and applications, it is key to orchestrate heterogeneous 'things' in a seamless, adaptive and dynamic manner, such that the system can exhibit a goal-directed behaviour and take appropriate actions. Yet, this form of interaction between things needs to take a user-centric approach and by no means elude the users' requirements. To this end, contextualisation is an important feature of the system, allowing it to infer user activities and prompt the user with relevant information and interactions even in the absence of intentional commands. In this work we propose a role-based model for *emergent configurations of connected systems* as a means to model, manage, and reason about IoT systems including the user's interaction with them. We put a special focus on integrating the user perspective in order to guide the emergent configurations such that systems goals are aligned with the users' intentions. We discuss related scientific and technical challenges and provide several uses cases outlining the concept of emergent configurations.

1 Introduction

Connecting everything that can benefit from being connected, from both digital and physical worlds, is becoming reality nowadays. IoT is already successfully adopted in several application domains such as environmental monitoring, healthcare service, transportation, inventory and production management among others [2, 64]. Recent studies show that IoT changed the way people use the Internet, mobile devices and sensors [23].

However, in order to reach its full potential, the IoT domain has to overcome a number of shortcomings. On the one hand, in the enterprise space, the majority of IoT solutions suffer from the so-called 'cloud silo' problem. A common

practice in today's IoT solutions is to have processes and data kept in separate data centers, making it difficult to access it or to interact with other systems. This design is often dictated by the companies' business models or due to certain security concerns. Nonetheless, these silos can cause real issues by keeping data and services locked away and unavailable for an efficient use by external third parties. On the other hand, in the retail space of devising smart things as the building blocks of IoT, noticeably, *things* are typically designed to follow a simple request-response pattern enacted by the user. This clearly imposes many restrictions in terms of the user experience and makes it impractical to specify requests and goals that concern more than one device at a time.

We aim to address these challenges encountered across the entire IoT spectrum in a unified manner. The focus of our current work is to provide means to orchestrate heterogeneous *things* in a seamless, adaptive and dynamic manner, such that the system can exhibit a goal-directed behaviour and take appropriate actions, according to the user's desires and intentions. As we shall see in the following sections, this raises many interesting research issues open for cross-pollination with the multiagent domain. Informally, we term an *Emergent Configuration (EC)* as a set of things with their functionalities and services that connect and cooperate temporarily to achieve a goal [11]. Given the continuously changing character of many IoT systems, ECs can change unpredictably. This tendency is even more pronounced in domains that present pervasive characteristics such as mobile users, devices and sensors. Thus, it is important to provide the user with a coherent system at any point in time, capable to align to the user's context and goals.

To this end, we put forward in this paper a general theoretical model for designing *Emergent Configurations*. Our goal is to provide an encompassing framework where new mechanisms addressing the different aspects outlined by our model can be developed and deployed. Specifically, ECs set the stage for addressing and reasoning about three key categories of challenges that represent a barrier to effectively enable today's IoT systems:

(i) challenges in *interoperability* - include aspects of how can devices operate across different types of networks, how to connect and deal with mobile devices, how to integrate devices of different hardware and software specifications, how to ensure ease of scalability?

(ii) challenges in *coordination*: are concerned with how can devices collectively operate based on the user delegated goals to the system, how can devices separately adapt to runtime conditions while a consistent view of the system is maintained by the user, how to resolve data inconsistencies when retrieving the data from various sources?

(iii) challenges in *user interaction* - how to improve usability and provide novel ways of interaction with *sets* of devices, how to provide a user-friendly interface for a variety of devices, how to avoid overwhelming the user with data, how to improve usefulness and automatically identify relevant information, how to make the system proactive in supporting the user's tasks?

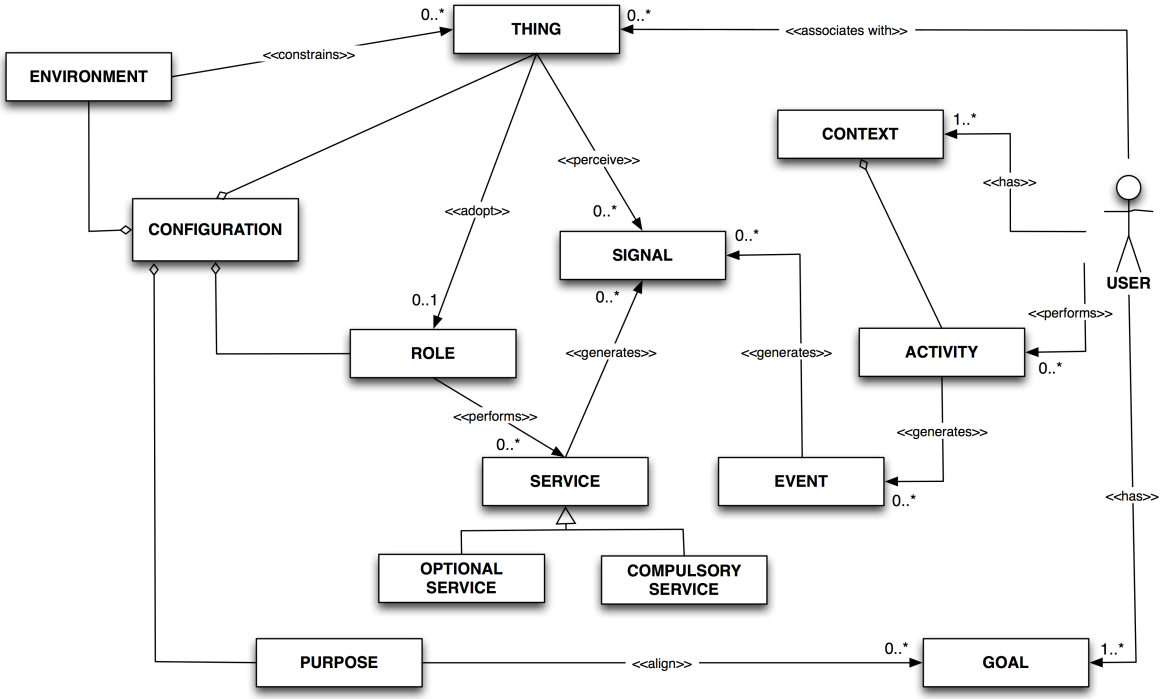


Fig. 1. Conceptual model of the role-based approach

The reminder of the paper is organized as follows. In Section 2 we present the basics of our proposed model characterizing *Emergent Configurations*. In Section 3 we outline a representative EC use-case developed in co-production with industry partners. In Section 2 we discuss and point towards several key enabling technologies. Section 5 concludes and draws future research directions.

2 Emergent Configurations for the Internet of Things

2.1 Basics of the proposed model

In this section we lay the groundwork for providing a model of emergent configurations in an IoT ecosystem. The proposed framework focuses in particular on the importance of *roles* adopted by devices in various configuration settings. The aim is to provide a framework that is general enough to model different IoT scenarios and can capture the key aspects of adaptability to dynamic environments, while maintaining a coherent user perspective of the system according to its delegated objectives. This approach makes it easier to reason about integrating algorithms, which deal with different subproblems of the domain, as well as to reveal gaps in existing research. In Figure 1 we give an overview of the different elements considered in our model, which we describe hereafter.

Definition 1 *A connected thing is an uniquely identifiable (embedded) computing system that has at least the capability to exchange data.*

Basically, a *connected thing* can represent anything, spanning from the simplest RFID tag or sensor, to actuators, devices, computing-enabled physical world objects, smartphones, computers, or even items of an increased complexity such as modern fully equipped self-driving cars. It is further assumed that *things* which are equipped with sensing capabilities can perceive, collect and index timestamped *signals* received from the environment or communication messages from other *things* in the environment. As a general approach, signal sequences registered over a certain time-frame are aggregated and classified to a higher-level representation, generating a new sequence of *events*. Similarly, in a hierarchical manner, the system can construct higher-level abstractions from the underlying sensing data.

Definition 2 *A linear sequence of signals or events $\epsilon_1, \epsilon_2, \cdots \epsilon_t$ obtained over the time period t is perceived as (part of) the activity performed by the user.*

Suppose as an example a scenario where a smartphone is used to track signals from the accelerometer, gyro and GPS, inferring events such as accelerating speed, entering a park and continue at increased speed, in order to characterize a 'jogging in the park' activity. The activity carried out by the user can be further enriched with additional information (e.g. time of day, weather, etc.), which captures the overall *context* of the user [13, 17].

Definition 3 *Context is denoted here as the set of current statements about the user, which are either directly sensed from the environment or inferred.*

In fact, there is an extensive body of work focusing on the area of activity recognition and context-awareness, as it pertains to the various sensing modalities (e.g. smartphones [35], vision [49], ambient sensors [40,42]) and the different application domains (e.g. health [10], smart living spaces [39], sports [8]). For a comprehensive survey on context-aware computing we further refer the reader to [5,46]. The underlying goal is to devise techniques that seamlessly acquire context information without the explicit intervention or feedback of the user. The use of semantics is one potential avenue for introducing meta-mechanisms that capture the relation between contexts and trigger the appropriate techniques for context inference, in a more general setting that can cover a wider spectrum of senors and domains. In addition, an important part of the context is the *user profiling*, which retains the user's preferences.

Definition 4 *The environment $\mathcal{E} = \langle \Lambda, \mathcal{T} \rangle$ specifies a set of constraints $\Lambda = \{c_1, \cdots, c_n\}$ which induce the set of things $\mathcal{T}$, providing a clear delimitation between the things in $\mathcal{T}$ and those outside this set. Constraints are further differentiated into physical constraints (e.g. devices at a certain location, communication ports, resource configuration) and virtual constraints (e.g. communication protocols, restrictions over virtual platforms, latency requirements).*

Definition 5 *A role is a tuple* $\Pi = \langle \varphi, \chi, \mathcal{S} \rangle$*, which specifies an invocation function* $\varphi : \chi \rightarrow \mathcal{S}$*, which activates, for a given condition in the set* χ*, an item* $s \in \mathcal{S}$ *from the restricted set of possible actions or service type interactions* $\mathcal{S}$ *available for this role.*

Items in the condition set χ can denote anything from incoming messages to perceived events or activities. Note that, a role can also be interpreted as a certain skill set $\mathcal{S}$, based on the actions and services that the role can perform.

Definition 6 *A configuration is a tuple* $\mathcal{C} = \langle \mathcal{R}, \mathcal{T}, \Delta, \beta, \mathcal{E} \rangle$*, where* $\mathcal{R}$ *is a collection of roles,* $\mathcal{T}$ *is a collection of things in* $\mathcal{E}$ *and* $\Delta : \mathcal{R} \rightarrow \mathcal{T}$ *represents a mapping of roles to things. With* β *we denote the purpose of the configuration and* $\mathcal{E}$ *represents the environment.*

It is important to highlight that function φ also specifies the *type* of action or service invoked by a certain role r. On the one hand, we distinguishing between services $E(r)$ that are available to that respective role in the configuration, and services provided by this role, $P(r)$, to be accessed by other roles in the configuration. On the other hand, we differentiate between services $M(r)$ that are compulsory in the configuration and those that are optional $O(r)$. The set of compulsory services in a configuration $\mathcal{C}$ induces the set of compulsory roles $\Gamma_{\mathcal{C}}(\mathcal{R})$, required in order to run the configuration. Optional roles $\gamma_{\mathcal{C}}(\mathcal{R})$ can, for instance, be used to map to *things* in the environment that are only temporary part of the configuration.

In other words, *roles* provide the interaction constraints between the different entities in the configuration and can be assimilated in a broader acceptation to a workflow or (negotiation) protocol, that *connected things* adopting different roles need to abide to. A simple example would be to determine the device with the highest QoS for a certain service among a list of possible candidates. This would entail prescribing the roles of service provider and service consumer, whereas the interaction could specify a simple Contract Net protocol [57], which follows, on the provider side, a predefined sequence of broadcasting a call for bids, followed by receiving and evaluating bids and finally, determining the winner.

Going back to the overview in Fig. 1, the inferred activity of the user captures her underlying goals in a given context, which in turn dictates what roles her *associated connected things* can adopt in a configuration. We use the term associated connected things in relation to the user to denote any means which interfaces the user with the configuration. They can range from something as trivial as a PIR sensor detecting presence of a person, to more advanced means of interaction, such as a smartphone, which can require some form of user feedback. In the situation when different parts of the configuration retain different information about the user, it is important to maintain a coherent view and to resolve potential data inconsistencies, when retrieving the data from various sources. Trust-based data fusion techniques [65] can represent a viable manner for prioritizing, processing and harmonizing data from heterogeneous sources.

Note, that the purpose of the configuration needs to be aligned with the user's goals, otherwise its significance to the user would be meaningless. This brings about a number of interesting research questions, that situates the user as the focal point of decision making in the system.

For instance, it is important to identify what is an appropriate way to specify and represent user goals. The use of ontologies could provide a way for representing, on the one hand, the user goals and on the other hand, the purpose of the configuration. If the user and the configuration's ontologies align, then we could reason that the configuration can provide a relevant function for the user. A dialog-based user-interaction can allow the user to guide the actual execution run of the configuration. Also, to what extent should the system use unobtrusive techniques in order to elicit the user's goals and obtain contextual information, and when should this information be explicitly specified based on the user's interaction with the system? There is an underlying trade-off here between the proactiveness of the system and the transparency of the process. If the system displays an increased level of autonomy, which is not clearly understood by the user, this can in fact hinder its usability and frustrate the user. In this regard, the recently introduced research avenue of UbiCARS [38], which aims to characterize ubiquitous recommender systems, can bring interesting contributions to the concepts emphasized here.

2.2 Emergence

Given that the interaction between *connected things* in a configuration is mediated through *roles*, another important aspect concerns the way according to which roles are established. It is important to allow the users or the system itself to modify the system behavior, in terms of prescribing new roles in a configuration or evolving more specialized roles to carry out certain tasks.

Definition 7 *Let $\mathcal{U}$ denote a user with goal $\mathcal{G}$. We term an Emergent Configuration, the outcome of generating a configuration $\mathcal{C} = \langle \mathcal{R}, \mathcal{T}, \Delta, \beta, \mathcal{E} \rangle$, where the purpose β of the configuration accommodates the goal $\mathcal{G}$ of user $\mathcal{U}$, given the environment $\mathcal{E}$.*

In this sense, we make the distinction between *weak* and *strong* emergence. In the former case, we have *static* roles and we are only concerned with identifying a suitable mapping Δ from *roles* to *things*. In effect, this represents an open-ended process where things can join the configuration on-the-fly, adopting certain roles, or may choose to leave the configuration at a certain point in time. This may also include situations where mobile software agents, which are performing certain roles, migrate from device to device in order to achieve the purpose of the configuration. For the later case, we assume that *roles* are *dynamic* and can change over time in order to better accommodate the goals of the user and the purpose of the configuration. A key feature of the system is its capability to adapt roles in order to correspond with the functionalities of the connected things adopting those roles.

On the one hand, role adaptation can be triggered by explicitly receiving instructions from the user regarding the behavior of the configuration. Interaction design can play an essential part in this regard. On the other hand, the *things* associated to the user, through continuous monitoring, can determine changes in the context or activities perform by the user, and therefore infer a new user goal that sets in motion at run-time a role adaptation procedure. In the more extreme case, we can envision a setting where either the configuration is built in a top-down manner, solely from specifying a certain *user goal* (which also represents *purpose* of the configuration) or alternatively, a collection of roles are evolved to serve a newly acquired purpose in a self-organizing fashion. This can be translated in a simple example by considering a smart home environment where the user is presented with a new type of functionality, such as begin able to use the smartphone in order to control and operate a smart TV. In this instance, the smartphone and the TV adopt the roles of master and slave respectively.

2.3 Managing emergent configurations

Another aspect that has to be considered is linked to the way according to which a configuration is managed. A configuration $\mathcal{C}$, where $\mathcal{R} = \Gamma_{\mathcal{C}}(\mathcal{R})$ and $\gamma_{\mathcal{C}}(\mathcal{R}) = \emptyset$, meaning that configuration $\mathcal{C}$ includes only compulsory roles, is termed a *centralized* configuration. For example, the *leader* role can be compulsory for initiating a configuration and regulating the interaction between the other members of the configuration. Then, if all the roles specified by the configuration are compulsory, the configuration is a centralized one. At the opposite end of the spectrum, in a *decentralized* configuration, we have $\mathcal{R} = \gamma_{\mathcal{C}}(\mathcal{R})$ and $\Gamma_{\mathcal{C}}(\mathcal{R}) = \emptyset$. Here, there is no device that assumes a compulsory role and the set of services provided by the configuration depends solely on the optional roles adopted by devices in the configuration at that time and the services they can receive and be expected to deliver. We term a *hybrid* configuration one that involves both compulsory and optional roles ($\mathcal{R} = \Gamma_{\mathcal{C}}(\mathcal{R}) \cup \gamma_{\mathcal{C}}(\mathcal{R}), \Gamma_{\mathcal{C}}(\mathcal{R}) \neq \emptyset, \gamma_{\mathcal{C}}(\mathcal{R}) \neq \emptyset$), in the sense that compulsory roles are instrumental and all the other roles depend on the services provided by them.

The request for adopting a certain role can either be solicited by a member of the configuration (e.g. the leader advertises roles), or proposed by a device in the environment. This can take the form of a simple 'handshaking' procedure or, in more complex scenarios, it can imply that a negotiation occurs which can result in a service level agreement (SLA) [1], that regulates future interactions. A normative extension of roles using representations such as obligations and prohibitions could further be used to enforce certain device behaviors [12]. What is then an appropriate way to handle communication? An option inspired from MAS is to communicate using ACL (Agent Communication Language), a standard language for agent communication defined by FIPA (The Foundation of Intelligent Physical Agents) [60]. Also relating to implementation issues, we point towards model-driven engineering as a proposed support technology for ECs and the IoT, and regard [11] as a possible architecture provided for self-adaptive systems.

2.4 Discussion

Finally, perhaps nearest to our work, we can relate the model proposed here to several efforts addressing the design of ontologies for ubiquitous and pervasive systems (e.g. CoDAMoS ontology [50]). However, the research in this area is focused on developing specialized ontologies dedicated to model specific domains, such as BOnSAI [58], which addresses energy savings in smart building environments and has a particular emphasis on modeling devices, appliances and actuators. Another example is the OntoAMI model [54], which presents a simplistic semantic model for universal use across ambient intelligence applications, but fails to incorporate the *user* into their approach. Alternatively, in this work, we put forward a novel concept of *Emergent Configurations* to denote a set of things with their functionalities and services that connect and cooperate temporarily to achieve a goal and provide a framework based on the notion of *roles* to support reasoning and developing such systems. Moreover, we put special emphasis on the *user* and explicitly account for the user's goals based on its current context and activity, in order to guide the emergence of configurations, such that the system's goals are aligned with the users intentions.

It is important to point out a number of key benefits attained via our role-based approach. From a development standpoint, roles enable a decoupling between the interaction aspects, which prescribe how collaboration between devices occurs, and the internal computational processes of each device, which need not be exposed to other third parties. From the conceptual point of view, roles can be referred to as design patterns, in the sense that a set of roles that are used to fulfill the purpose of a configuration could further be (adapted and) reused for similar purposes. That is, roles can promote quite general, high-level solutions, which can then be parametrized and instantiated in various specific applications. Roles encapsulate certain functionalities, which equip devices with the necessary building blocks for establishing dynamic interactions in continuously changing environments and which can be directly related to the user's goal. In this manner, devices can dynamically adapt in a versatile manner to different coordination protocols and different functionalities by assuming various roles in different configurations.

3 Use case of emergent configurations

In this section we introduce the "Mesh Presenter" prototype[1] which has been developed as part of the ECOS project[2], where we investigate how to model, interact with, and reason about ECs. We use this straightforward case study (see Fig. 2) to emphasize the expressiveness of our formal model, which clarifies based on precise definitions how to dynamically integrate different entities and stakeholders to a configuration and how to regulate their behavior.

[1] https://vimeo.com/130228635
[2] http://iotap.mah.se/ecos/

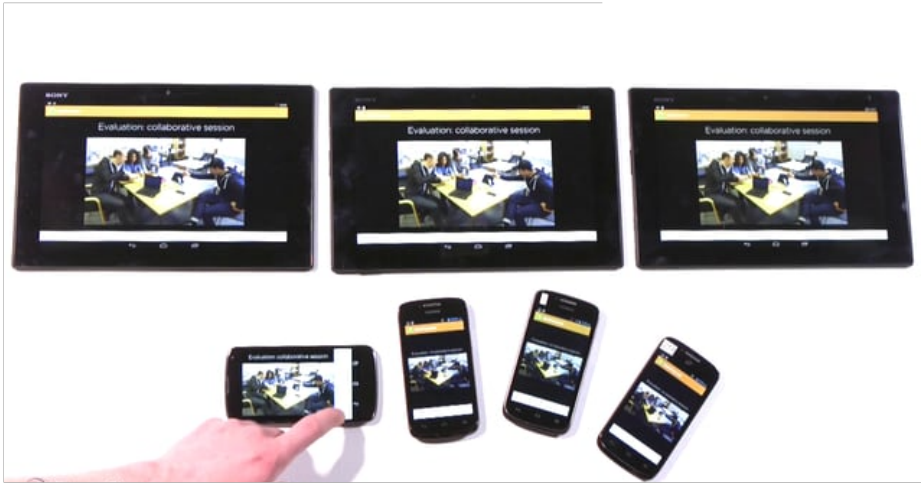


Fig. 2. Emergent Configuration use-case: Mesh Presenter

The *purpose* of the "Mesh Presenter" (MP) configuration is that of providing an *on-the-fly* virtual collaborative workspace for sharing and generating content on smart devices (e.g. smartphone, tablets, projectors) co-located in a meeting room. In more detail, we describe the MP configuration according to the definitions introduced in Section 2. Recall that roles represent one of the key underpinnings of our approach, providing a way to mediate interaction between different entities in the configuration in a dynamic manner.

Specifically, the MP configuration pertains that the set of roles $\mathcal{R}$ includes two types of roles. On the one hand, there is a compulsory role referred as $\Gamma(\mathcal{R}) = \{presenter\}$ and an optional role termed $\gamma(\mathcal{R}) = \{reviewer\}$. Each of these roles carries out a certain functionality, which can be described in terms of services. According to our model, we differentiate between two categories of services associated to each role. The set of services provided by the presenter include *P(presenter)* = {*share presentation; add/remove reviewer; enable presenter control*}. This means that the reviewer role can request the presenter to execute any of these services. The presenter can grant or revoke access to any of the above-mentioned services to the reviewer. Given the collaborative context of this configuration, the reviewer can also actively participate in the content generation process by exposing the following service *P(reviewer)* = {*share content*}. However, the presenter is the one who decides if the new data (e.g. image, slides) should be broadcasted on all devices within the configuration. For the sake of simplicity, in this usecase, having only two roles, the following sets coincide *P(presenter)* = *E(reviewer)* and *E(presenter)* = *P(reviewer)*, meaning that all services provided by one of the roles are accessible by the other. In addition, it is required that all services in *P(presenter)* are implemented for this particular

role, thus *M(presenter)=P(presenter)* and $O(presenter) = \emptyset$. The same applies for the reviewer role.

In order to have a well-defined configuration several additional aspects need to be specified. We assume that users interact with the MP configuration by means of their associated smart devices. Importantly, the configuration environment $\mathcal{E}$ determines the set of devices that the presenter can allow to be part of the configuration. In this case, the inclusion constraint is based on a proximity rationale (i.e. presence in the meeting room). Also, the environment restricts the manner in which communication can take place, in this case being mesh connectivity. Contextualization also plays a significant part here. The real-time incoming data from the sensors of the device, such as a smartphone, are used to infer the user's activity. Once the *context* of 'being present in a meeting' is recognized, this acts as a precondition for the reviewer role to invoke the service of joining the configuration. Similarly, when the presenter determines that another device has entered its environment $\mathcal{E}$ it will forward an inquiry to that device to join the configuration. User feedback is required in both of these instances in order to finalize the procedure. The preconditions for invoking all of the remaining services, such as sharing a presentation or granting presenter control to another device are all triggered based on a simple user interface, which basically summaries function φ, that is responsible for service invocation.

Note that, in the case of the MP configuration, we are dealing with a *weak* type of emergence, as we are essentially assigning fixed roles to devices that are joining the configuration. Alternatively, a scenario that would exhibit a *strong* emergence would entail that roles evolve and specialize over time, such that they can provide, new functionalities for the configuration, new services or new types of interaction (e.g. a new reviewer role that can connect remotely to the configuration). Finally, with respect to managing the configuration, we point out that MP is a *hybri*d configuration as it consists of both compulsory and optional roles. A configuration can exist even in the absence of a reviewer, which can assume a temporary role, but not in the absence of the presenter which takes here a coordinator type of role.

4 Enabling technologies

In our work, we are taking a *user-centric* approach, by placing the users at the very center of the IoT ecosystem, which they have to manage in an efficient and effective way such that it can service their needs. We identify in the following, several key enabling technologies for IoT and point towards the contribution they can bring in the context of our model. Also, we relate them back to the three classes of problems outlined in Section 1. Notably, we identify multiagent systems as arguably the main area of interest for advancing the EC concept presented in this work. Likewise, the user-centric approach advocated here, deploying a role-based solution for orchestrating IoT environments, provides an interesting perspective for new developments in MAS.

4.1 Middleware

The large majority of the research in IoT literature has focused thus far on dealing with the first class of challenges identified in the introduction, namely issues in M2M interoperability. In this sense, significant progress has already been made in terms of middleware platforms (see [15, 52] for a comprehensive survey) designed to abstract hardware-specific issues and facilitate interoperability of various components, such as ad-hoc communication, discovery of new devices or establishing new communication links. These solutions also simplify the development process at the application level and exempts developers from handling implementation at the lower layers. In particular, service-oriented architectures (SOA) [21, 45] have emerged as a popular way for providing a middleware with standard protocols, common interfaces and well-defined components. This can be regarded as an approach by means of which connected things can easily interact and collaborate such that certain ECs are formed, according to the virtual constraints of the environment. Fog computing [7] can also be regarded as an architectural blueprint for ECs, given its distributed infrastructure comprising of heterogeneous resources, which needs to be managed in a distributed fashion.

Another important aspect in our model that defines an *emergent configuration* is the mapping function Δ, which relates roles to things. A good candidate for implementing this function is given in [48], where the authors propose a method for automatically deploying and configuring applications onto a heterogeneous hardware infrastructure, by taking into account the hardware properties and characteristics. Alternatively, a multiagent system can act as an intermediary layer, providing the necessary level of autonomy for reasoning about adopting different roles in the configuration. At the same time, agent-based platforms, such as Calvin [47], provide a lightweight distributed runtime, allowing the system to dynamically deploy agents according to requirements and hardware constraints. Along the same lines, self-adaptive systems (SaS) [51] can play an important role in enabling run-time reallocation, reconfiguration and increasing reuse (e.g. thanks to model-driven technologies [11]). SaS are concerned with modifying the system's behavior at runtime in response to their perception of the environment or the system itself. In the last decade, the research in SaS has mainly focused on technical solutions, while less effort has been devoted to the human involvement that is the special perspective we advocate here.

4.2 Multiagent-based coordination

Regarding the second class of challenges outlined in the introduction, in order to address the problem of device coordination, multi-agent systems (MAS) [61] is yet another important area we draw inspiration from in the realization of the concept of emergent configurations in IoT. In particular, we draw the parallel between ECs and an *open* MAS, which assumes a high level of interaction between a large number of highly stochastic, heterogeneous and possibly self-interested agents operating within a dynamic environment. In such a system agents may form groups to solve more complex problems via cooperation, given that many

tasks cannot be completed by a single agent because of limited resources or capabilities. A relevant example in this sense is given in [41], where the authors propose an online coalition formation scheme, having agents negotiate and formulate speculative coordination solutions, with respect to the estimated behavior of the system. This type of learning-based coordination can be instrumental for ECs in the way that roles evolve over time (representing in our scenario a case of strong emergence).

Importantly, agents running on devices can also act as a middleware in a configuration, enabling the interaction between devices, while at the same time adopting different roles in a versatile manner according to the user's goals. To tackle the problem of today's IoT platforms designed as vertical silos, we can envision a scenario, where at least one agent assumes a special *broker* role, responsible for interfacing with each IoT platform, while the configuration emerges at a cross-platform level. This means that we could have for instance a situation where the configuration consumes data from a number of sensors registered to one IoT platform, then applies a data analysis component from another big data platform and visualizes the result on the projector that is managed by yet another local cloud platform. Notably, since a role-based approach is particularly aimed at providing a high-level description focusing on the interaction between *things*, this has the advantage of simplifying and supporting the development of large scale applications.

In [44] the authors introduce the notion of emergent coordination and discuss its potential for efficiently handling coordination in open environments. The authors advocate that a careful intentional design of a limited set of interaction rules and a common communication protocol could produce the desired emergent property, which is a higher level property caused by the interaction of its lower level components. This brings about the question as to what are the key ingredients required in order to engineer a system such that it displays certain emergent properties, how does this relate to the types of emergence identified in ECs and what is the role of the user in this context?

Moreover, MAS allows to address challenges of autonomous and decentralized decision-making in a flexible manner, by decomposing complex tasks and assigning subproblems to loosely coupled components that interact and coordinate autonomously to solve system-level design goals. In a similar manner, ECs enhance open MAS by factoring in the user perspective. That is, ECs explore the extent of users involvement in the system not only as final user, but also as an active support for the system to guide or enable the execution of specific goals. An interesting preliminary work is [9], in which human participants are explicitly modeled and the authors reason about human involvement in self-adaptation, focusing on the role of human participants as effectors during the adaptation execution phase.

4.3 Interaction Design

Finally, from a user interaction perspective, again, people need to be involved not just as end-users, but as co-constructors of the system itself. A common

strategy for co-construction has been designing for 'end-user programmability', such as rule based systems [14] and 'programming-by-example' [18]. Here, devices are often used as the 'control panel' to represent, configure and control a complex system and its constituent components ([19], [29]). The disadvantage of this paradigm is that it draws attention away from the task-at-hand, making it difficult to continue the flow of activity [53] and can suffer from device-centric rather than activity-centric design [20]. Thus, the challenge of designing IoT systems and applications that take a user-centric approach in order to guide the emergent configurations, such that the system's goals are aligned with the users' intentions, is still an open research question. Introducing roles takes one step further in this direction by means of binding the users' goals to certain roles assumed by their devices. Moreover, roles can be evolved in order to match changes in the user's goals. The framework described in this paper represents a vision that has to be tackled gradually, encompassing all of the identified challenges.

5 Conclusions and future work

In this paper we defined and characterized emergent configurations of connected systems, as an approach to model, manage, and reason about collaborative IoT systems consisting of a number of autonomous yet coordinated devices and the user's interaction with them. We propose herein a role-based approach to ECs that captures a key missing aspect in today's IoT systems, namely, adaptability to dynamic environments, along with maintaining a coherent user perspective of the systems aligned with the user's objectives. The exploitation of roles at run-time enables *things* to dynamically adopt specific roles in order to satisfy the user's goal or cope with dynamic environments. Our aim is to utilize a common framework, which can support a wide range of IoT applications, instead of targeting specific scenarios, each of which can exercise particular aspects of the framework. We discussed scientific and technical challenges involving several research disciplines at the same time and we provided concrete cases showing how ECs can be used and help the IoT in practice.

As future work we plan to continue the development of the proposed framework, with a special emphasis on interaction design and the ways in which we can deliver an enhanced user experience providing functionality even in the absence of intentional commands, as well as on designing techniques to ease the development and deployment of ECs in real case scenarios. Moreover, in this work we are strictly concerned with generating a configuration with respect to the goal of a single user. In order to accommodate multiple user, conflict resolution mechanism (such as negotiation procedures) may need to be included, in the event that the goals of different users cannot co-exist in a given configuration. Regarding the evaluation, we would also like to identify and develop more full-fledged usecases that fully exploit the expressiveness of our model (e.g. configurations that display *strong* emergence). It would be interesting to see the level of reusability we can attain by casting roles to new usecases and the impact of using the concept of roles throughout the analysis, design and implementa-

tion phases. Besides the development of additional cases and validation against domain-specific settings, the framework introduced here is also well suited for the integration, validation and comparison of already existing algorithm and mechanism or future proposals. Among them, we mention dynamic discovery of *things* and services, goal adaptation mechanisms, approaches for configuration monitoring, context acquisition and inference, or novel ways for user interaction.

Acknowledgments. This work is partially financed by the Knowledge Foundation (KKS) through the Internet of Things and People research profile (Malmö University, Sweden).

References

1. A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwiga, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. Web services agreement specification (WS-agreement). Technical report, 2007. `https://www.ogf.org/documents/GFD.107.pdf`.
2. Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787 – 2805, 2010.
3. Jon A Benediktsson and P H Swain. Consensus theoretic classification methods. *Systems, Man and Cybernetics, IEEE Transactions on*, 22(4):688–704, 1992.
4. Ayse B Bener, Volkan Ozadali, and Erdem Savas Ilhan. Semantic matchmaker with precondition and effect matching using SWRL. 36(5):9371–9377, 2009.
5. Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, and Daniele Riboni. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2):161 – 180, 2010.
6. Jürgen Beyerer. *Verfahren zur quantitativen statistischen Bewertung von Zusatzwissen in der Messtechnik*, volume 8 of *VDI Fortschritt-Bericht*. VDI/Verl., 1999.
7. Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, pages 13–16, New York, NY, USA, 2012. ACM.
8. Andreas Bulling, Ulf Blanke, and Bernt Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Computing Surveys*, 46(3):33:1–33:33, 2014.
9. Javier Cámara, Gabriel A. Moreno, and David Garlan. Reasoning about human participation in self-adaptive systems. In *10th IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2015, Florence, Italy, May 18-19, 2015*, pages 146–156, 2015.
10. Saisakul Chernbumroong, Shuang Cang, Anthony Atkins, and Hongnian Yu. Elderly activities recognition and classification for applications in assisted living. *Expert Systems with Applications*, 40(5):1662 – 1674, 2013.
11. Federico Ciccozzi and Romina Spalazzese. MDE4IoT: Supporting the Internet of Things with Model-Driven Engineering. In *Proc. of the 10th International Symposium on Intelligent Distributed Computing (IDC'16)*. Springer - (To appear), 2016.

12. N. Criado, E. Argente, and V. Botti. Open issues for normative multi-agent systems. *AI Commun.*, 24(3):233–264, 2011.
13. Valentin Cristea, Ciprian Dobre, and Florin Pop. *Internet of Things and Inter-cooperative Computational Technologies for Collective Intelligence*, chapter Context-Aware Environments for the Internet of Things, pages 25–49. Springer Berlin Heidelberg, 2013.
14. Luigi De Russis and Fulvio Corno. Homerules: A tangible end-user programming interface for smart homes. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pages 2109–2114. ACM, 2015.
15. Jose Delgado. *Internet of Things and Inter-cooperative Computational Technologies for Collective Intelligence*, chapter Service Interoperability in the Internet of Things, pages 51–87. Springer Berlin Heidelberg, 2013.
16. Emanuele Della Valle, Dario Cerizza, and Irene Celino. The mediators centric approach to automatic web service discovery of glue. *MEDIATE2005*, 168:35–50, 2005.
17. Anind K. Dey. Understanding and using context. *Personal Ubiquitous Computing*, 5(1):4–7, 2001.
18. Anind K Dey, Raffay Hamid, Chris Beckmann, Ian Li, and Daniel Hsu. a cappella: programming by demonstration of context-aware applications. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 33–40. ACM, 2004.
19. Nicolas Ducheneaut, Trevor F Smith, James Bo Begole, Mark W Newman, and Chris Beckmann. The orbital browser: composing ubicomp services using only rotation and selection. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems*, pages 321–326. ACM, 2006.
20. W Keith Edwards, Mark W Newman, Jana Z Sedivy, and Trevor F Smith. Experiences with recombinant computing: Exploring ad hoc interoperability in evolving digital networks. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 16(1):3, 2009.
21. Thomas Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.
22. Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer-Verlag Berlin Heidelberg, 2007.
23. Dave Evans. How the next evolution of the internet is changing everything. Technical report, Cisco Internet Business Solutions Group, 2011. `http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf`.
24. Johannes Fähndrich. *Analyse von Verfahren zur Kombination von Expertenwissen in Form von Wahrscheinlichkeitsverteilungen im Hinblick auf die verteilte lokale Bayes'sche Fusion*. PhD thesis, Karlsruhe Institut of Technology, May 2010.
25. Johannes Fähndrich, Tobias Küster, and Nils Masuch. Semantic Service Management and Orchestration for Adaptive and Evolving Processes. *International Journal on Advances in Internet Technology*, 9(4):75–88, 2016.
26. Johannes Fähndrich, Nils Masuch, Hilmi Yildirim, and Sahin Albayrak. Towards Automated Service Matchmaking and Planning for Multi-Agent Systems with OWL-S – Approach and Challenges. In *Service-Oriented Computing – ICSOC 2013 Workshops*, pages 240–247. Springer International Publishing, Cham, January 2014.
27. Christian Genest. Pooling operators with the marginalization property. *The Canadian Journal of Statistics/La Revue Canadienne de Statistique*, 12(2):153–163, 1984.

28. Christian Genest, S Weerahandi, and J V Zidek. Aggregating opinions through logarithmic pooling. 17(1):61–70, June 1984.
29. Yucheng Jin, Chi Tai Dang, Christian Prehofer, and Elisabeth André. A multi-display system for deploying and controlling home automation. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, pages 399–402. ACM, 2014.
30. Matthias Klusch, Benedikt Fries, and Katia Sycara. OWLS-MX: A hybrid Semantic Web service matchmaker for OWL-S services. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(2):121–133, April 2009.
31. Matthias Klusch, Andreas Gerber, and Marcus Schmidt. Semantic web service composition planning with owls-xplan. pages 55–62, 2005.
32. Matthias Klusch and Patrick Kapahnke. The iSeM matchmaker: A flexible approach for adaptive hybrid semantic service selection. 15:1–14, September 2012.
33. Matthias Klusch, Patrick Kapahnke, and Ingo Zinnikus. SAWSDL-MX2: A Machine-Learning Approach for Integrating Semantic Web Service Matchmaking Variants. In *2009 IEEE International Conference on Web Services (ICWS)*, pages 335–342. IEEE Computer Society, IEEE, 2009.
34. Matthias Klusch, Ulrich K uster, Alain Leger, David Martin, and Massimo Paolucci. 5^{th} International Semantic Service Selection Contest - Performance Evaluation of Semantic Service Matchmakers. November 2012.
35. Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explorations Newsletter*, 12(2):74–82, 2011.
36. Steffen Lamparter and Anupriya Ankolekar. Automated selection of configurable web services. *8. Int. Tagung Wirtschaftsinformatik*, 2007.
37. Nils Masuch, B Hirsch, M Burkhardt, A Heßler, and S Albayrak. SeMa^2: A Hybrid Semantic Service Matching Approach. In *Semantic Web Services*, pages 35–47. Springer Berlin Heidelberg, 2012.
38. Christos Mettouris and George A. Papadopoulos. Ubiquitous recommender systems. *Computing*, 96(3):223–257, 2014.
39. Radu-Casian Mihailescu and Paul Davidsson. Integration of smart home technologies for district heating control in pervasive smart grids. In *Proceedings of the First International Workshop on Pervasive Smart Living Spaces*, 2017.
40. Radu-Casian Mihailescu, Paul Davidsson, and Jan A. Persson. Multiagent model for agile context inference based on articial immune systems and sparse distributed representations. In *Multi-Agent Systems and Agreement Technologies - 13th European Conference, EUMAS 2015, and Third International Conference, AT 2015, Athens, Greece, December 17-18, 2015, Revised Selected Papers*, pages 82–87, 2015.
41. Radu-Casian Mihailescu, Matthias Klusch, and Sascha Ossowski. e COOP: privacy-preserving dynamic coalition formation for power regulation in the smart grid. In *Proceedings of the Second International Conference on Agreement Technologies, AT 2013, Beijing, China, August 1-2*, pages 19–31, 2013.
42. Radu-Casian Mihailescu, Jan A. Persson, Paul Davidsson, and Ulrik Eklund. Towards collaborative sensing using dynamic intelligent virtual sensors. In *Intelligent Distributed Computing X - Proceedings of the 10th International Symposium on Intelligent Distributed Computing - IDC 2016, Paris, France, October 10-12 2016*, pages 217–226, 2016.
43. P A Morris. Combining expert judgments: A Bayesian approach. *Management Science*, 23(7):679–693, 1977.

44. Sascha Ossowski and Ronaldo Menezes. On coordination and its significance to distributed and multi-agent systems. *Concurrency and Computation: Practice and Experience*, 18(4):359–370, 2006.
45. Mike P. Papazoglou and Willem-Jan Heuvel. Service oriented architectures: Approaches, technologies and research issues. *The VLDB Journal*, 16(3):389–415, July 2007.
46. C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE Communications Surveys Tutorials*, 16(1):414–454, 2014.
47. Per Persson and Ola Angelsmark. Calvin – merging cloud and IoT. *Procedia Computer Science*, 52:210 – 217, 2015.
48. Ferdinand Piette, Cédric Dinont, Amal El Fallah Seghrouchni, and Patrick Taillibert. Deployment and configuration of applications for ambient systems. In *Proceedings of the 6th International Conference on Ambient Systems, Networks and Technologies*, ANT-2015, pages 373–380. Procedia Computer Science, 2015.
49. Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976 – 990, 2010.
50. Davy Preuveneers and E Berbers. Automated context-driven composition of pervasive services to alleviate non-functional concerns. *International Journal of Computing and Information Sciences*, 3:28–38, 2005.
51. R. de Lemos et al. Software engineering for self-adaptive systems: A second research roadmap. In Rogério de Lemos, Holger Giese, Hausi A. Müller, and Mary Shaw, editors, *Software Engineering for Self-Adaptive Systems II*, volume 7475, pages 1–32. Springer-Verlag, 2013.
52. Mohammad Abdur Razzaque, Marija Milojevic-Jevric, Andrei Palade, and Siobhan Clarke. Middleware for internet of things: A survey. *IEEE Internet of Things Journal*, 3(1):70–95, 2016.
53. Christof Roduner, Marc Langheinrich, Christian Floerkemeier, and Beat Schwarzentrub. Operating appliances with mobile phones–strengths and limits of a universal interaction device. In *Pervasive Computing*, pages 198–215. Springer, 2007.
54. Maria J. Santofimia, Francisco Moya, Felix J. Villanueva, David Villa, and Juan C. Lopez. An agent-based approach towards automatic service composition in ambient intelligence. *Artificial Intelligence Review*, 29(3):265–276, 2009.
55. Marco Luca Sbodio. SPARQLent: A SPARQL Based Intelligent Agent Performing Service Matchmaking. pages 83–105. Springer Berlin Heidelberg, Berlin, Heidelberg, April 2012.
56. G Shafer. *A mathematical theory of evidence*, volume 1. Princeton university press, 1976.
57. R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113, 1980.
58. Thanos G. Stavropoulos, Dimitris Vrakas, Danai Vlachava, and Nick Bassiliades. BOnSAI: A smart building ontology for ambient intelligence. In *Proceedings of the Second International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, pages 30:1–30:12, New York, NY, USA, 2012. ACM.
59. M Stone. The opinion pool. *The Annals of Mathematical Statistics*, 32(4):1339–1342, 1961.
60. The Fundation of Intelligent Physical Agents. FIPA specifications. Technical report, 2002. `http://www.fipa.org/repository/standardspecs.html`.

61. Michael Wooldridge. *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition, 2009.
62. Jian Wu and Zhaohui Wu. Similarity-based web service matchmaking. *Services Computing*, 2005.
63. Zhibiao Wu and Martha Palmer. Verbs Semantics and Lexical Selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.
64. L. D. Xu, W. He, and S. Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, 2014.
65. Zheng Yan, Peng Zhang, and Athanasios V. Vasilakos. A survey on trust management for internet of things. *Journal of Network and Computer Applications*, 42:120 – 134, 2014.

Implementation and Evaluation of Negotiation Mechanism on Server-less IoT Application Platform

Takuma Oide[1⋆], Toru Abe[2], and Takuo Suganuma[2]

[1] Graduate School of Information Sciences, Tohoku University, Japan, oide@ci.cc.tohoku.ac.jp
[2] Cyberscience Center, Tohoku University, Japan, {beto, suganuma}@tohoku.ac.jp

Abstract. The rapid progress of mobile sensor devices such as smartphones has brought us into the Internet of Things (IoT) era. Simultaneously, the concept of edge computing, alongside the current centralized cloud paradigm, has attracted significant attention as a new model that enables user-aware privacy protection and saves network resources. With these points in mind, we have studied a server-less contract-based IoT application platform in which participants can share their data in a completely distributed manner. In this paper, we introduce the concept of part-contract negotiation and report on the design of a mechanism that takes into account dynamic user context in order to increase sensing data sharing among all participating users. Finally, we also clarify the conditions to ensure that the requirements are fulfilled and, through simulation results, we show that approximately 100% of the tasks can be performed by considering the user context.

Keywords: Crowdsourcing · Sensor Platform · Negotiation · Participatory Sensing

1 Introduction

The rapid growth on performance of mobile sensors embedded in devices such as smartphones has brought us into the Internet of Things (IoT) era, and we can now inexpensively utilize vast amounts of sensor data generated by these functional devices, consequently, this information can be used for various purposes such as activity recognition and environmental condition monitoring.

Due to the special requirements of server-side processing, sensor devices and network infrastructure, conventional sensor-based applications present a high initial / running costs which, in consequence, impose strong limitations on the use of collected sensing data. In contrast, the IoT environment allows us the application of sensor cloud [10] and participatory sensing [6] schemes, in which general sensors, such as the ones commonly found in daily use devices, are more likely to be used rather than specific purpose built-in sensors. Applications based

⋆ Research Fellow of Japan Society for the Promotion of Science

on these paradigms enable enhanced data fusion and collaboration, which means we are now able to share data on demand. However, as more IoT devices become ubiquitous in our lives, sensor-based applications need to solve other technical challenges; in particular, providing privacy-sensitive data to *clouds* requires critical attention. One direct way to tackle this issue is by applying encryption and anonymization mechanisms to protect the data stored in clouds whenever those systems need to provide data to other users. However, since anonymization is a complex theory, and thus difficult for general users to understand, they may find it difficult to feel safe and may refuse to send their personal data regardless of the applied mechanism.

Fog computing [8] and edge computing [23] are state-of-the-art paradigms in which mobile sensor devices play both computing and data generation roles. Moreover, these paradigms are expected to address user privacy issues, as well as cloud centralization problems, by processing parts of various procedures on the user-side infrastructure before sending data to the Internet.

In previous publications [13,14], we proposed a server-less distributed, contract-oriented, sensor-based, application platform that utilizes the edge computing paradigm to share data among participants. This platform had mainly the following features:

- Minimal required interaction with participants whenever they provide and collect data. Each participant has individualized agent components that discover data-sharing candidates and negotiate conditions directly with them.
- Flexible policy application for data usage with other participants. Direct communication among agents enables flexible negotiation depending on the context (e.g., number of participants, their relationship or purpose of their communication).
- Server-less architecture, which means that there is no need for a central server or cloud. The only requirement is the installation of the platform on each participant edge device.

However, since sudden context changes were not considered when creating the initial contract, frequent mismatches occurred between senders and receivers, generating a large number of maintenance messages to rectify the changed conditions. In response to those points, this paper introduces the *part-contract* concept and proposes a novel context-based negotiation algorithm.

In this paper, we will introduce our part-contract concept and propose a negotiation protocol that considers changes to the participants' context based on our previous mechanism. The protocol can be adapted to a wide variety of sensor-based application platforms implemented on the edge side. We will also show the performance of the protocol via simulation results and discuss its use in actual environment settings.

Contributions of this paper are as follows:

- We report on the design and implementation of a negotiation protocol created by adopting our part-contract strategy, and demonstrate its effectiveness through simulation results. The protocol can improve the percentage

of the achieved segment as part of the whole task segment, which means achieving aggressive sensor data sharing among participants without requiring active participation from user senders.
– We clarify the necessary conditions when participants launch their on-demand application in order to ensure active data sharing amongst them. Required functions and performance are also discussed.

The remainder of this paper is organized as follows: Section 2 presents descriptions and assessments of related works. Section 3 describes a platform use case scenario and outlines the design of our proposed negotiation flow. Section 4 shows the effectiveness of the proposal and discusses how to develop platform-based IoT applications through simulation results. Conclusions and future work are presented in Sect. 3.

2 Related Work

Conventional sensor-based applications are based on a participatory sensing paradigm [6] that gathers specific types of data generated from participant smartphones and sends it to centralized clouds [1, 2, 7, 16, 17]. In most cases, the processed results are then sent back to the participants. For example, a life log system [16] that links up with an existing participatory sensing network [2] acquires positional information and acceleration information from a participant's smartphone, performs behavior estimation, and shares atmospheric information such as CO_2 concentration. In addition to this, air pollution maps [7] and noise maps [1, 17], created by collecting environmental / noise information obtained from small-sized highly functionalized sensor devices such as smartphones, are proposed. This architecture has succeeded in integrating highly functional mobile-edge devices into the system components. However, there are still some issues concerning sensor usage versatility. For instance, participants must provide their data to each particular system, and their data is not shared with other participants.

Nowadays, a wide variety of IoT sensor-equipped devices have been developed, and the trend of connecting them to the Internet is rapidly growing. However, even if the collected sensing-data could be useful to more users, current applications are designed in such a way that the data is to be used by a single user.

Some studies [4, 18, 20, 24] have proposed sensor data sharing platforms that allow participants to provide their data for general use to anyone – providing the participants grant permission. In these platforms, participants first register the required sensor-information and their intended usage to a *task*, after which other participants decide which task they are willing to reward by providing their own data in return. To that end, it is important to design a motivation mechanism that encourages participants to provide their data to others by convincing them that such participation could also help them achieve their goals, thereby preventing interaction imbalances. However, with these platforms, it is necessary to view the task information of others when data is provided, and the time and

effort has increased compared with the use of a conventional sensor-based application that provides fixed data. It is conceivable that the fact that discovery of the tasks that it can provide itself is manual, and the fact that changes in the data distribution setting due to the change in the environment or its own provision policy being manual is a great obstacle to providing motivation.

To tackle this issue, the primary approach is applying incentive mechanisms such as [3, 11, 12, 15, 21, 22, 25], to motivate participants to help others complete their tasks. By following these theories, both providers and consumers do not suffer any disadvantage and the system can maximize the social surplus. However, these approaches still require participants to select tasks that are suitable to their policy from a pool of tasks that other participants have previously registered. This complicated interaction decreases both a sender's motivation to participate as well as a receiver's chance to complete their tasks. In addition, the difficulty to adopt these approaches to applications due to complexity of the theories is pointed out in a survey article [9].

In contrast, our proposed platform [13] reduces the complex interactions between users and the system, thereby encouraging senders to provide their data, by using agent components to perform most of the processes instead of requiring the participants to handle the details. Initially, they also register their policy for each sensor-equipped device as well as the tasks, and afterwards the agent components automatically communicate with each other to create contracts and achieve tasks based on their owner's preset policies and approved task list. However, since sensing data are transmitted without the owner's explicit awareness, unexpected context changes (e.g., movement from a particular area, policy changes) can occur while sending data according to the contract.

3 Design

3.1 Use Case Scenario

To illustrate the proposed platform, we adopt the following scenario as shown in Fig. 1.

A user named Alice installs the platform application into her mobile phone because she wants to share sensing data with others. Initially, she has to register a disclosure policy for each sensor device installed on her phone. On Sunday morning, she decides to go to either a zoo or an aquarium, but she wants to know how crowded and the weather conditions in both locations beforehand. She then inputs her requirements concerning the areas and kinds of data she wants to obtain, after which she immediately obtains location data and temperature data generated from smartphones of participating users who are currently at those locations. Based on the received information, she decides to go to the zoo instead of the aquarium, since it is sunny in the zoo area and the zoo itself is currently not too crowded. After spending the day at the zoo, once she returns at night the recorded data on the platform shows the amount and type of data that were utilized by other people, e.g., her coarse location data ("one user around the zoo") is referred once when she was walking in the zoo.

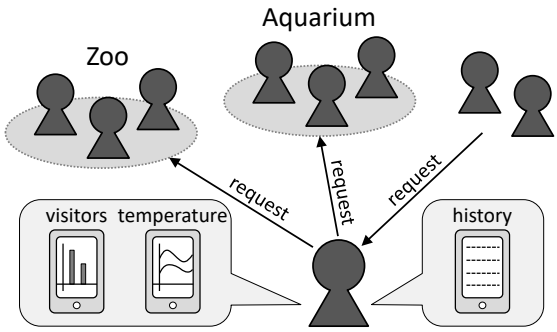


Fig. 1. Use case scenario of proposed platform

It should be noted that in this scenario, users already visiting the zoo or the aquarium are not informed about the details of the real-time sensor data transmission from their devices. They simply receive a notification (such as a vibration tag) indicating that an interaction has taken place. As a result, actions that are inconvenient in terms of the contracts, such as users leaving the zoo during their data sensing-provision agreement with Alice, or users restricting their disclosure levels in order to conserve battery power, may occur.

This is one of the critical concerns of our platform because, in conventional task-based approaches, senders must explicitly decide which tasks they will provide their data to, and thus decide whether they are willing to participate in each task based on the rewards offered. Contrastingly, in our platform agent components actively perform these processes in order to remove such time-consuming work from the users.

In addition to the example use case above, we also assume the following applications that is suitable for our proposed platform.

Health management system: Managers observe health condition of participants in real time at for example marathon race event and sports meeting by using this system. The system changes frequency and parameters of sensor data collected from biometric sensors attached to the body of participants such as heartbeat / blood pressure sensors and others according to the health condition of the observed person and displays them in real time. Compared to the conventional health management system, since it is possible to control the distribution of unnecessary sensor data, it is expected that the power consumption of the biosensor can be suppressed.

Multimedia streaming system: Users freely share multimedia data among them during watching sports games and music live events, and display in real time by using this system. The system can limit the distribution area of multimedia streaming to specific areas such as sports venues and live venues. Compared to conventional multimedia streaming systems, it is not necessary to install infrastructure, and transmission by device-to-device communication makes it possible to expect improvement of quality of experience (QoE).

Tourist information sharing system: Tourists shares information about the places with on-site users and users who send request to the area. The system can select just related information to specific areas such as sightseeing spots where the user is and the user plans to go. Compared to information sharing systems that use conventional social networking services, etc., improvement in convenience can be expected because selection of related information is triggered by the user being in the position itself.

3.2 Sequence Flow

Figure 2 shows the sequence flow chart of the scenario described in the previous subsection.

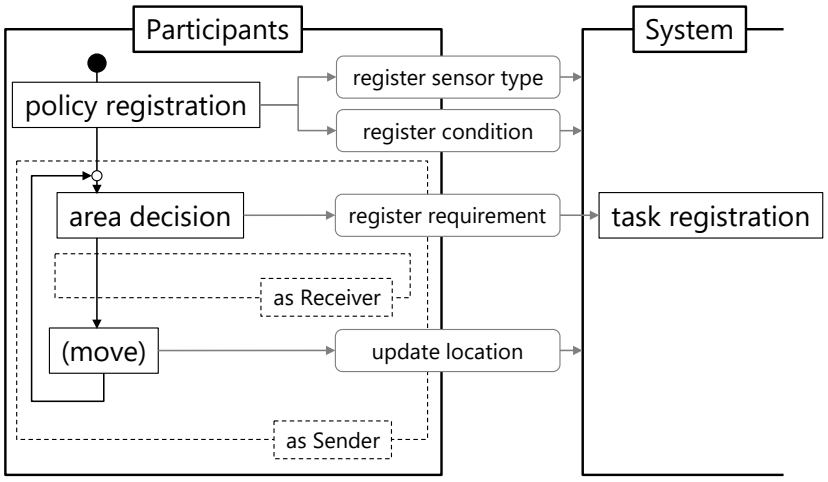


Fig. 2. Model chart of participant behavior. Participants act as both receivers and senders.

The system launches following participant policy registration, after which sender agents are generated for each sensor device that may have knowledge about relevant conditions, such as relationships among receivers (e.g., only friends, or people listed in contract history), and the range of relevant area (e.g., one kilometer around the location or one hop forwarding on a Mobile Ad-hoc Network (MANET)). After registration, the agents behave as senders instead of participants and begin to receive task announcement requests from other agents.

After participants have registered their areas of interest and the type of data they hope to acquire, the system registers new receiver agents containing task-related information. The registration process includes a task period, a point of interest, required receiving frequency, and so on. During the period registered tasks are active, the agents behave as receivers instead of participants as well

as senders for tasks listed by other participants. Receiver agents immediately start to collect required data based on contracts made with the sensor agents of participating users who are currently at specific locations. In the case of the use case scenario in the previous subsection, they collect location data to produce a crowd map, and then perform weather monitoring by acquiring temperature and rain-fall data. The detail of negotiation and contract making process among agents are described later.

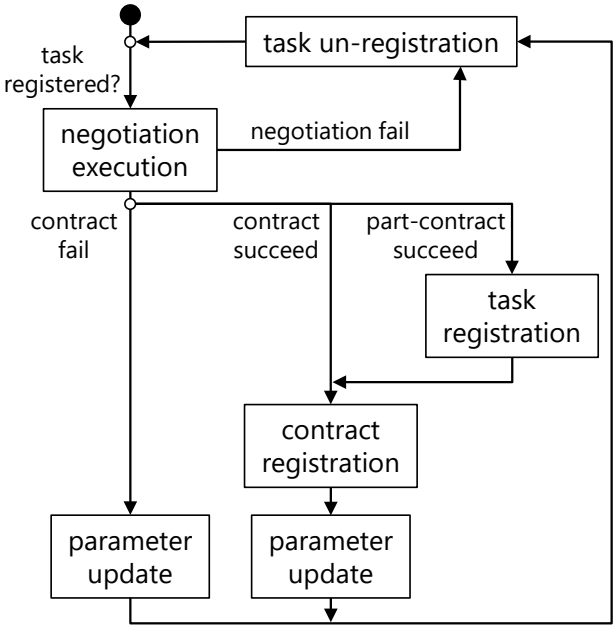
Whenever participants move, the system updates their location on the logical overlay network, which allows them to be discovered when location-based search queries are launched. Agents can discover how many other agents are at those locations without linking to any central control components (such as servers) by submitting location-based search queries to the automatically constructed structured peer-to-peer overlay network among participants' devices [5, 19].

This figure clearly shows that participants can easily use the platform and share their data without possessing a large amount of technical knowledge, and without requiring tedious interactions with the system.

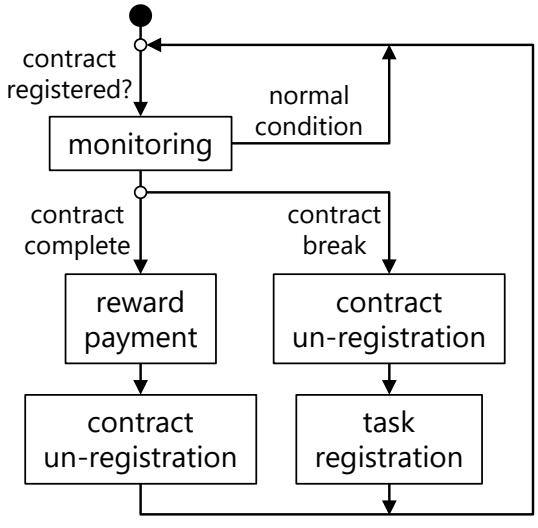
While tasks are registered, the task-centric loop is activated, as shown in Fig. 3(a). Receiver agents start the negotiation flow with the sender agents previously registered by other participants. The system generates receiver agents for each task in order to start the negotiation flow with the sender agents previously registered by other participants. If a sufficient number of sender agents cannot be found in a specific area, the agent removes the task and ends its life cycle. In other conditions, the agent continues the flow and compares task conditions and proposals in the replies received. When any proposal or set of proposals does not fulfill the set conditions, no contract is made. At that time, the agent updates the parameters used in the negotiation phase.

If some sender agents reply with favorable bids, the receiver agent negotiates contracts with those agents and registers the resulting contracts in the system. From the moment on, they begin to share sensor data based on those contracts but if some requirement attributes (e.g., received data amounts) are deficient in some parts of a task period, the receiver agent registers additional subtasks that includes the most recently updated attributes and processed in the next loop interaction.

After registering contracts on the task-centric loop, the contract-centric loop is launched as described in Fig. 3(b) in order to monitor how sensor data is shared among agents. When contracts have expired correctly, the sender agents receive the rewards set forth in the contract conditions in exchange for providing their data, and delete the contract afterwards. If participants move out from a specific area or change their policy while their agents are still transferring sensor data, some contracts are canceled because the property of shared data no longer meet the contract conditions. When that happens, the agents remove the contract and the receiver agent registers the task again in order to retry negotiations with other agents in the next task-centric loop.



(a) task-centric chart



(b) contract-centric chart

Fig. 3. Model chart of system behavior. There are two main loops: a task-centric loop and a contract-centric loop.

3.3 Basic Negotiation Flow

The basic ideas behind the negotiation flow and communication protocols among agents were described in detail in our previous publications [13, 14]. In this subsection we briefly describe the entire negotiation flow procedure, and then we focus on the part-contract strategy designed for reducing contract cancellations for the next subsection.

Figure 4 shows the basic negotiation flow corresponding to the *negotiation execution* part previously shown in Fig. 3(a). First, a receiver agent opens negotiations and sends `TASK_ANNOUNCE` messages to sender agents discovered. The message includes a task period $T := [t_s, t_e)$, a point of interest (POI), required receiving frequency f_{min}, and so on. Sender agents receiving the message evaluate if they should submit bids to the receiver agent by comparing the message with the context and their owners' policies. After a while, the receiver agent aggregates `BID` reply messages in order to select a subset of candidates that best fulfill the requirements, at the lowest cost, and then generates contracts and sets adjusted conditions for each successful bid. Finally, after receiving the `AWARD` message from the receiver, successful agents begin providing their data according to the conditions in the contract.

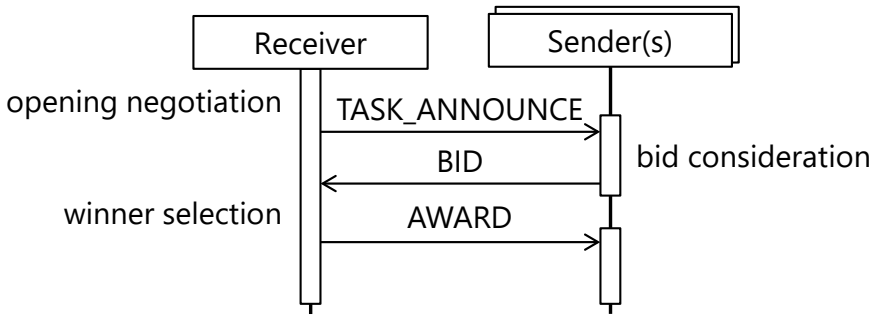


Fig. 4. Negotiation flow between receiver and sender agents. This part corresponds to the *negotiation execution* block in the task-centric chart.

In this platform, the application consists of mainly two components: Sender agent, which is the provider of the data; and Receiver agent, which is the consumer of the data. There is only one receiver agents per application, but there can be multiple sender agents. Distribution of data is performed from the sender(s) to the receiver, and the properties such as frequency and usage are determined based on the agreement concluded beforehand among the constituent agents. There is no central servers and the data directly flow to the target receiver.

3.4 Part-Contract Strategy

Since participants move around freely and change their policies irrespective of their contracts, there exists a potential risk of contracts being canceled without any warning, and as a consequence, creating dissatisfaction on both receiver and sender. Receivers are unable to obtain sufficient amounts of required data and they need to renegotiate with other participants, while senders are unable to obtain rewards in return for providing data. In this paper, we introduce the concept of part-contract strategy (PCS) that reduces cancellation probabilities in two way: (i) by shortening bid periods based on the relationship between POI and its location, and (ii) by adopting a part-contract when the receiver agent selects successful bids among the available candidates.

Figure 5 shows a conceptual diagram depicting how to decide the condition of bids during a *bid consideration* phase in Fig. 4, where $b_1^i \cdots b_6^i$ are new bids of each sub-period divided according to other contracts previously made with other participants, and f_{max}^i is the maximum total frequency that indicates the limit load for providing data. Originally, sender agents are willing to bid using all residual resources in a full task period $[t_s, t_e)$ in order to obtain maximum reward amounts from receivers. However, the probability of unexpected owner behavior increases when longer task periods are promised. Therefore they reduce their bid periods to $[t_s, t'_e)$, which is shorter than the original task period, based on the owners' locations relative to the center of the POI. This does not reduce chances for senders to provide data in future periods because the receiver will re-advertise its adjusted task after making part-contracts and registering sub-tasks for the remaining periods.

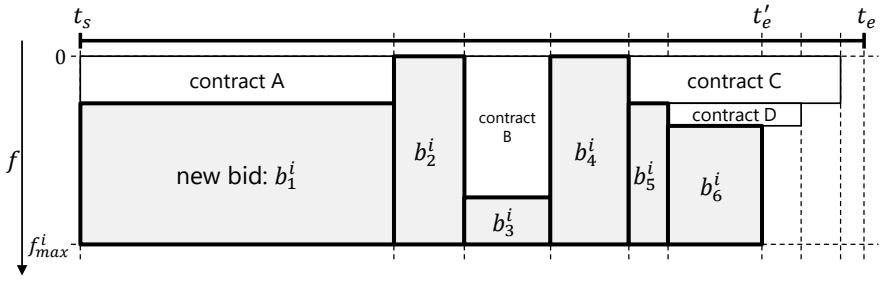


Fig. 5. Conceptual diagram of sender agent i bid decision.

Figure 6 shows a conceptual diagram of a winner bid selection during the *winner selection* phase in Fig. 4, where three senders i, j, and k have submitted bids of b_n^i, b_n^j, and b_n^k, respectively. In this situation, the receiver agent creates contracts for the recently completed segments in which the total amount of data provided exceeds the required value f_{min}. At that time, it also registers sub-tasks related to the unfilled segments prior to retrying negotiations with other newly arriving sender agents in the POI.

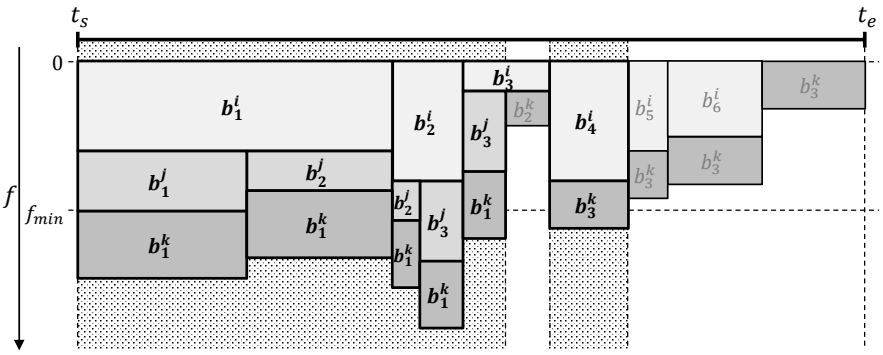


Fig. 6. Conceptual diagram of receiver agent award decision. Three sender agents i, j, and k replied with their bids.

4 Simulation Evaluation

4.1 Simulation Setup

In this simulation, participants move randomly over a 1,000 × 1,000 m simulated field on which they have been evenly distributed as shown in Fig. 7. They independently choose their POIs at intervals in order to obtain local environmental data while Alice collects zoo and aquarium data as part of the use case scenario. More specifically, each participant repeatedly performs the following procedure until the simulation ends:

1. Determines a destination area as POI, the range of which is represented as p_r.
2. Collects environmental data in the POI using the platform.
3. Moves into the area.
4. Determines a new destination area after arriving to the previously set destination.

Since participants act without paying attention to the contracts their agents are executing, it should be noted that some participants will miss the chance to achieve their requirements because of contract cancellations.

Table 1 outlines parameters used in our simulation. We changed the number of participants to confirm how the results are influenced when the balance between senders and receivers is changed. f_{max} indicates that the senders' capacity of providing their data; for example, when the value of f_{max} is set to 1, the senders do not provide their data simultaneously more that one receiver. Fixing f_{min} with 5 means that a receiver agent collects data from five sender agents when their maximum data provision frequency (f_{max}) is set to 1. Underlined values ($n = 200, f_{max} = 1$) are used as default values when the value of n and f_{max} are not mentioned.

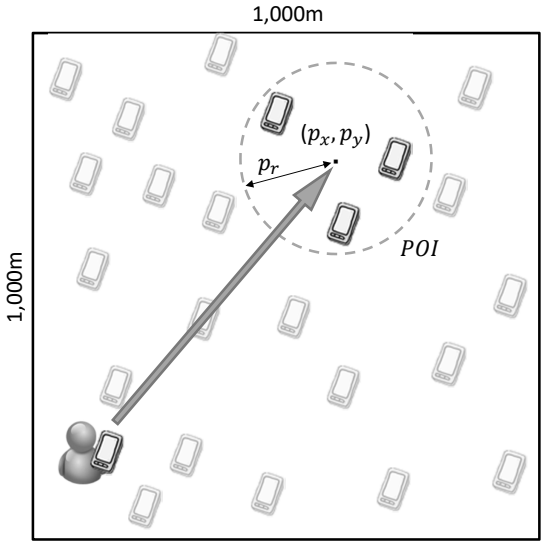


Fig. 7. Simulation field on which nodes move randomly and share data by using the negotiation protocol.

Table 1. Simulation Parameters

Field size	$1,000(m) \times 1,000(m)$
Number of participants	$50, 100, \underline{200}, 300, 400, 500$
Walking speed of users	4 (km/h)
Collection period	60 (s)
Range of collection area p_r	200 (m)
f_{min}	5 (Hz)
f_{max}	$\underline{1}, 2, 3, 4, 5, 10, 100$ (Hz)

As the metrics of this evaluation, three values are evaluated to define the effectiveness of our protocol; *Success rate*, *Completion rate*, and *Cover rate*. These metrics are calculated as shown below:

$$\text{Success rate} := \frac{\text{\# of published contracts}}{\text{\# of task announcements}}$$

$$\text{Completion rate} := \frac{\text{\# of expired contracts}}{\text{\# of registered contracts}}$$

$$\text{Cover rate} := \frac{\text{Sum of expired time segment}}{\text{Sum of required time segment}}$$

Success rate is the ratio of the number of contracts made by agents to the total number of tasks announced. However, it is noteworthy to mention that in the case where the initial negotiation fails are ignored, since a failure assessment is decided based on participant shortages, not on the protocol itself.

Completion rate indicates the amount of contracts expired without any obstacles such as unexpected moves and/or policy changes. Agents prefer choosing more stable bids to be expired if the value becomes higher.

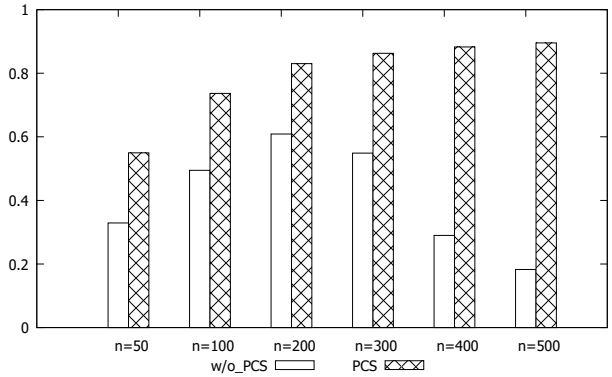
Cover rate, which is the most important value, indicates how long sensor data are shared among platform participants. For example, if Alice requests a minute of data for each of the two sites and actually gets 30 seconds of data from the zoo and 50 seconds of data from the aquarium, her *cover rate* would be 0.67.

4.2 Simulation Results

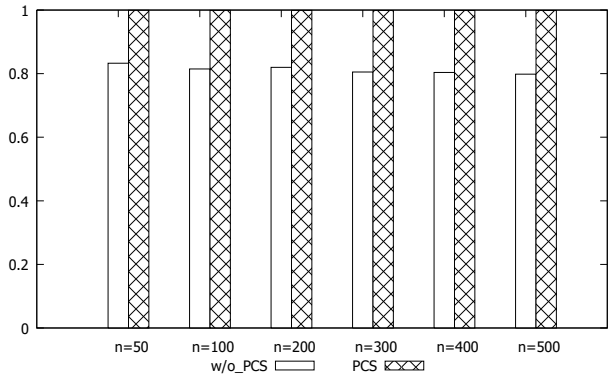
Versus Number of Participants. Figure 8 shows the performance values when the number of participants were changed, where `w/o PCS` permits only whole-contracts, which is the approach used in previous publications. In contrast, `PCS` permits the creation of part-contracts, as described in Sect. 3.4.

As shown in Fig. 8(a), the *success rate* on `w/o PCS` is lower than `PCS` at all cases and reach a peak for $n = 200$, since the value increases as the number of senders increase when $n = 50, 100, 200$, whereas the value decreases as the number of receivers increase for $n = 200, 300, 400, 500$. In this system model, participants act as both senders and receivers, so linking large numbers of people together does not always produce positive results. In contrast, the protocol showed better performance when the `PCS` was applied, as senders cooperate by providing their resources incrementally in order to fulfill a certain task. The *cover rate* shown in Fig. 8(c) changed similarly, but was lower in all cases, which indicates that increasing the number of sender candidates by stimulating users to participate (or expanding $POIs$) is not an effective approach to share large data amounts among participants. Finally, the *completion rate* shown in Fig. 8(b) reached roughly 100% on `PCS`, which is important to reduce the number of renegotiation processes and to offer rewards for provided data.

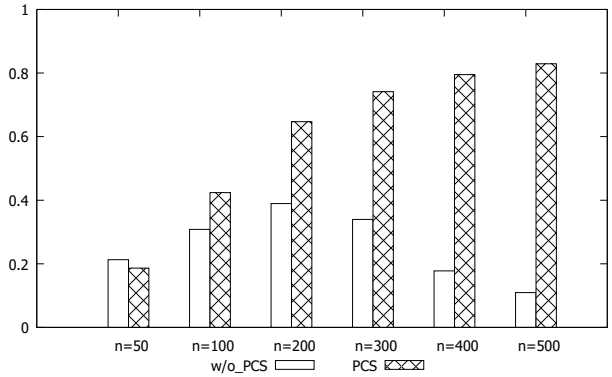
Versus Maximum Frequency of Senders. Another approach to encourage active data sharing is to increase the maximum frequency of senders f_{max}, which represents their positivity for providing data. Figure 9 indicates the performance values when f_{max} is changed between 1 and 100 while the required receiver frequency f_{min} is fixed at 5. The *success rate* and the *cover rate* increased consistently along with f_{max}, as shown in Fig. 9(a) and Fig. 9(c), because a rise of f_{max} indicates increase of the number of just senders, unlike the case of increase of n and p_r. Note that both values on `PCS` reached 100% when f_{max} is equal to or more than 5; which is the value of the required receiver frequency f_{min}. Therefore, increasing sender positivity for providing data is an effective approach to enhance receiver satisfaction and to encourage active data sharing because it is easier to apply into the system than the previously mentioned approaches. Increasing the number of participants is out of system function and expanding p_r will change the original receiver's requirements. The *completion rate* in Fig. 9(b) is basically the same as in the case of Fig. 8(b), since the value changes depends



(a) Success rate



(b) Completion rate



(c) Cover rate

Fig. 8. Performance values versus number of participants.

on movement parameters (e.g., walking speed, collection period), and not on n and f_{max}.

Measuring Communication Cost. Applying the PCS results in numerous message exchanges between agents after partitioning a task into various subtasks is one of critical issues for services on MANET. Figure 10 describe the average amount of received `TASK_ANNOUNCE` messages per participant during the simulation. In the case of varying n (Fig. 10(a)), the values of both `w/o PCS` and `PCS` increased linearly, accordingly with the linear increase in the number of candidates for each POI and also because the value for `PCS` is about three to four times bigger than it is on `w/o PCS`. On the other hand, when varying f_{max} in Fig. 10(b), the value for `w/o PCS` remains constant because both the number of tasks and candidates remain the same, while the value for `PCS` decreases as f_{max} increases. In particular, the value for `PCS` stays around 400 when $f_{max} = 5, 10, 100$, which is about 2.5 times higher than that for `w/o PCS`. In both cases, the increasing rate is a significant issue, especially for mobile devices and MANET users.

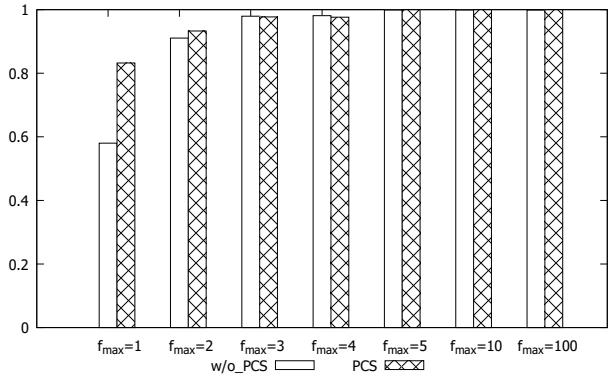
One way to reduce partitioning is to ensure that agents accurately estimate owner movements, which enables senders to bid with their maximum time segment and avoids ineffectual task divisions. Table 2 summarizes the results of `w/o PCS`, `PCS` and `PCS+`, where `PCS+` used simulation parameters related to user movements such as destination area, loitering time at its current location (described as the collection period in Table 1), and walking speed in order to estimate an accurate value of t'_e. In the case of `PCS`, as described in Sect. 3.4, it uses only the current user's position and POI in a `TASK_ANNOUNCE` message. As shown in Table 2, `PCS+` reduces communication cost among agents while maintaining high performance values.

Table 2. Comparison between `PCS` and `PCS+`

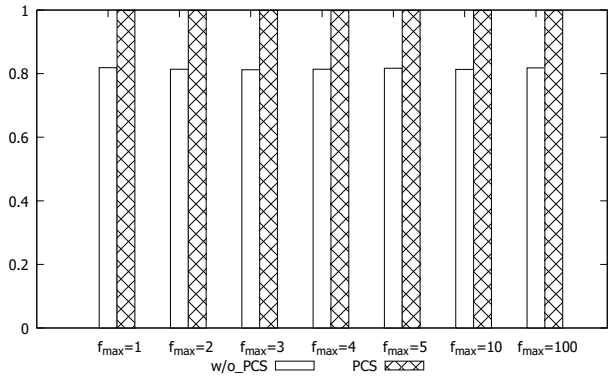
metrics	`w/o PCS`	`PCS`	`PCS+`
Success rate	0.584	0.830	0.920
Completion rate	0.813	0.999	1.000
Cover rate	0.364	0.648	0.830
Average number of `TASK_ANNOUNCE`	146	626	152

4.3 Discussion

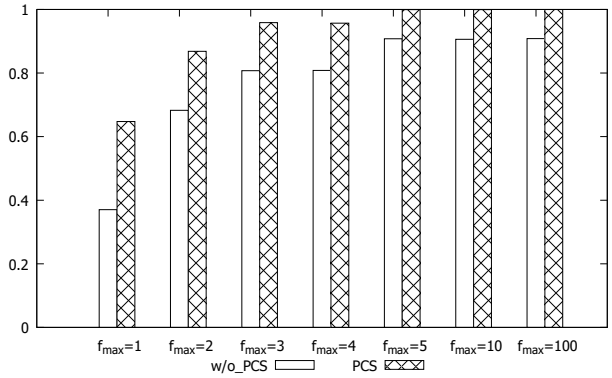
In our platform, agents generate tasks, discover candidates in $POIs$, negotiate and make contracts with them, and share sensor data – all without any assistance from centralized servers and without direct user involvement. Therefore, it was necessary to carefully design a sequence flow and a negotiation protocol to



(a) Success rate

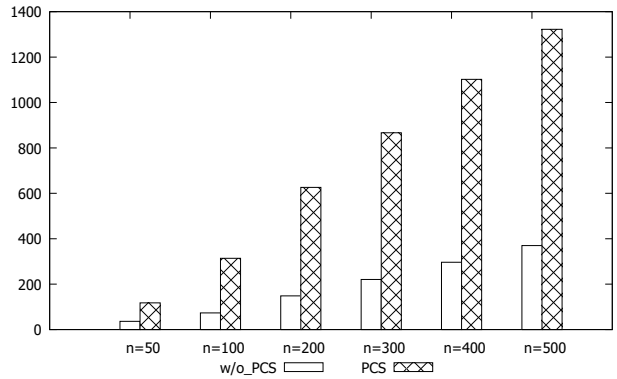


(b) Completion rate

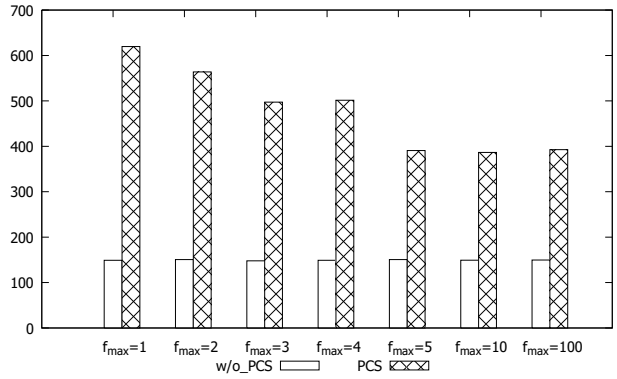


(c) Cover rate

Fig. 9. Performance values versus maximum frequency of senders.



(a) Varying number of participants



(b) Varying maximum frequency of senders

Fig. 10. Average amount of received `TASK_ANNOUNCE` message per participant.

restrain environmental change influences, including unexpected user actions, and as previously described, `PCS` has the effect of enhancing protocol performance. In this subsection, based on the simulation results, in the following paragraphs we will briefly discuss on how IoT applications should be implemented on the platform.

Improvements to the *cover rate* indicate effects on user experience as the value denotes the rate at which the system fulfills user requirements. Changing n and POI is certain to increase the value because sender candidates will also increase, but the change also generates gaps between requirements and generated tasks. On the other hand, changing the value of f_{max} was found to be more effective, especially when f_{max} is set equal to or more than f_{min}. One simple way to fulfill such condition is to ensure that the applications are restricted to a selectable range of f_{min} up to the user's f_{max}, this means that if a user wants to obtain a certain amount of sensor data, he or she must provide the same

amount of data to other participating users, which will lead to massive amount of sensor data sharing among participants. Rewarding participants for providing their data is another approach to increase the motivation of participants to act as senders. The approach is more effective than the restriction method because it also indirectly affects n growth, which means that determining how to best apply incentive mechanisms to applications will be a challenging study, as discussed in Sect. 2.

To decrease communication costs, it is also important to provide stable services that use mobile phones and mobile networks. Estimating user's movement by using entered parameters has a significant potential to enhance protocol performance with just a few message exchanges, but it is not a practical approach because participants have insufficient motivation for inputting their destinations and walking speeds. Accordingly, other incentive mechanisms that enhance their motivation to input such data, as well as mechanisms for providing data, will need to be applied. One potentially effective solution would be to apply trajectory-tracking algorithms that are primarily used in the mobile sensor network field. However, the trade-off between reducing communication costs and the increasing battery consumption must also be considered.

5 Conclusion

In this paper, we reported on the design of a part-contract negotiation algorithm that considers dynamic changes to user context in order to increase sensing data sharing among all users. The simulation results showed that this part-contract strategy performed better when participants moved around freely without considering the contracts their agents made. We also made clear the relationship between active sender data provision and receiver satisfaction, confirming that enhancing motivation to provide data via incentive mechanisms is more effective approach compared to other methods.

In future works, we intend to implement various applications based on our platform and evaluate their usability, as well as the effectiveness of the platform itself. Especially, we will develop a tourist information sharing system described in Sect. 3.1. A function of the system that entering a particular area becomes a trigger to get information about the place will indicate the feature of our platform well. In addition, we also plan to design and implement an ad-hoc device-to-device network protocol for commercial smartphones as a network layer structure in order to achieve flexible connectivity among participants by ensuring seamless switching between long- and short-distance communication protocols.

Acknowledgments. This work was supported by JSPS KAKENHI Grant Number 15J09912.

References

1. Noisetube, `http://noisetube.net/`

2. Opensense, http://opensense.epfl.ch/
3. Al-Fagih, A.E., Al-Turjman, F.M., Alsalih, W.M., Hassanein, H.S.: A priced public sensing framework for heterogeneous iot architectures. IEEE Transactions on Emerging Topics in Computing 1(1), 133–147 (jun 2013)
4. Arnaboldi, V., Conti, M., Delmastro, F.: Cameo: A novel context-aware middleware for opportunistic mobile social networks. Pervasive and Mobile Computing 11, 148–167 (2014)
5. Bharambe, A.R., Agrawal, M., Seshan, S.: Mercury: Supporting scalable multi-attribute range queries. SIGCOMM Comput. Commun. Rev. 34(4), 353–366 (Aug 2004)
6. Burke, J.A., Estrin, D., Hansen, M., Parker, A., Ramanathan, N., Reddy, S., Srivastava, M.B.: Participatory sensing. In: Proc. World Sensor Web Workshop (SenSys '06) (Oct 2006)
7. Dutta, P., Aoki, P.M., Kumar, N., Mainwaring, A., Myers, C., Willett, W., Woodruff, A.: Common sense: Participatory urban sensing using a network of handheld air quality monitors. In: Proc. the 7th ACM Conference on Embedded Networked Sensor Systems. pp. 349–350. SenSys '09, ACM (2009)
8. Flavio, B., Rodolfo, M., Jiang, Z., Sateesh, A.: Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing. pp. 13–15 (2012)
9. Guo, B., Wang, Z., Yu, Z., Wang, Y., Yen, N.Y., Huang, R., Zhou, X.: Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm. ACM Computing Surveys (CSUR) 48(1), 7:1–7:31 (Aug 2015)
10. Hassan, M.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. ICUIMC '09 (2009)
11. Jin, H., Su, L., Chen, D., Nahrstedt, K., Xu, J.: Quality of information aware incentive mechanisms for mobile crowd sensing systems. In: Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '15). pp. 167–176 (2015)
12. Lee, J.S., Hoh, B.: Sell your experiences: A market mechanism based incentive for participatory sensing. In: Proceedings of the 2010 IEEE International Conference on Pervasive Computing and Communications (PerCom '10). pp. 60–68 (March 2010)
13. Oide, T., Abe, T., Suganuma, T.: A design of contract-oriented sensor application platform. In: Proceedings of the the Sixth IEEE Workshop on Pervasive Collaboration and Social Networking. pp. 172–177 (Mar 2015)
14. Oide, T., Abe, T., Suganuma, T.: A broker-less participatory sensing scheme by user matching mechanism based on market price approach. In: Proceedings of 2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops). pp. 1–6 (mar 2016)
15. Peng, D., Wu, F., Chen, G.: Pay as how well you do: A quality based incentive mechanism for crowdsensing. In: Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '15). pp. 177–186 (2015)
16. Predić, B., Yan, Z., Eberle, J., Stojanovic, D., Aberer, K.: Exposuresense: Integrating daily activities with air quality using mobile participatory sensing. In: 2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops). pp. 303–305 (mar 2013)

17. Rana, R.K., Chou, C.T., Kanhere, S.S., Bulusu, N., Hu, W.: Ear-phone : An end-to-end participatory urban noise mapping system. In: Proceedings of the International Conference on Information Processing in Sensor Networks. pp. 105–116 (Apr 2010)
18. Sunyoung, K., Jennifer, M., Eric, P.: Sensr: Evaluating a flexible framework for authoring mobile data-collection tools for citizen science. In: Proceedings of the 2013 Conference on Computer Supported Cooperative Work. pp. 1453–1462 (Feb 2013)
19. Takeda, A., Oide, T., Takahashi, A.: Simple dynamic load balancing mechanism for structured p2p network and its evaluation. International Journal of Grid and Utility Computing 3(2), 126–135 (2012)
20. Tangmunarunkit, H., Hsieh, C.K., Longstaff, B., Nolen, S., Jenkins, J., Ketcham, C., Selsky, J., Alquaddoomi, F., George, D., Kang, J., Khalapyan, Z., Ooms, J., Ramanathan, N., Estrin, D.: Ohmage: A general and extensible end-to-end participatory sensing platform. ACM Transactions on Intelligent Systems and Technology (TIST) 6(3), 38:1–38:21 (Apr 2015)
21. Tsujimori, T., Thepvilojanapong, N., Ohta, Y., Zhao, Y., Tobe, Y.: History-based incentive for crowd sensing. In: Proceedings of the 2014 International Workshop on Web Intelligence and Smart Sensing (IWWISS '14). pp. 2:1–2:6 (2014)
22. Yang, D., Xue, G., Fang, X., Tang, J.: Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing. In: Proceedings of the 18th Annual International Conference on Mobile Computing and Networking (MobiCom '12). pp. 173–184 (2012)
23. Yi, S., Li, C., Li, Q.: A survey of fog computing: Concepts, applications and issues. In: Proceedings of the 2015 Workshop on Mobile Big Data. pp. 37–42. Mobidata '15 (2015)
24. Zaman, J., Meuter, W.D.: Discopar: Distributed components for participatory campaigning. In: Proceedings of the Sixth IEEE Workshop on Pervasive Collaboration and Social Networking. pp. 160–165 (May 2015)
25. Zhang, X., Yang, Z., Zhou, Z., Cai, H., Chen, L., Li, X.: Free market of crowdsourcing: Incentive mechanism design for mobile sensing. IEEE Transactions on Parallel and Distributed Systems 25(12), 3190–3200 (Dec 2014)

Learning Mechanisms on OWL-S Service Descriptions for Automated Action Selection

Johannes Fähndrich, Nils Masuch, Lars Borchert, and Sahin Albayrak

DAI-Laboratory of the Technische Universität Berlin
Department of Electrical Engineering and Computer Science
Ernst-Reuter-Platz 7, 10587 Berlin, Germany
johannes.faehndrich@dai-labor.de (Corresponding author)

Abstract. With the increase complexity of IoT systems, the well known development paradigm for software development like Service Oriented Architectures and Multi Agent Systems have to be adapted to fit the now challenges of IoT environments. Here the vast amount of available components and the specific implementation for special purpose hardware calls for an abstraction of functionality as well as to new development tools which allow to handle the dynamism of IoT applications. Here a multitude of special purpose sensors with view computational resources have to be combined to create emerging software. Most of the hardware reaches an return of investment only offering its service to third party developers. For a developer to be able to integrate those services into an application, a adaptive search mechanism need to be developed, which is able to specialize in the different domains of e.g. IoT applications.

1 Introduction

Distributed systems based on the Service Oriented Architecture (SOA) paradigm have become more and more popular in recent years due to different trends like the Internet-of-Things or Big Data. This approach is accompanied by the planning community that chose planning based upon services as one subject to research. However, there are some challenges that come up with this vision. Usually services are not described by one central entity but by different humans, which—in their domain—have a context dependent view on the described objects of the world. A first step to cope with this challenge is the usage of one domain representation that all service developers use for their respective service description. Nevertheless, the way how each developer defines relevant attributes like input, output, preconditions or effects can be extremely different, which makes the general interpretation of a service's functionality difficult. State-of-the-art approaches therefore integrate service matchmaking components based upon semantic service descriptions, which can be used to find fitting services in each state of an agent planning algorithm. Therefore service matchmaking can be interpreted as one step planning and can seamlessly be integrated into the action selection mechanism of an AI planning algorithm in order to find a connection

between the initial state and the goal state within the search space. This approach has been subject to research and will be surveyed in the next section (see Section 2).

Usually service matchmaking components do not rely on one single matching strategy but use different mechanisms to compare a service request with an advertisement. One question that arises out of that is how to weight each strategy in order to aggregate them to a single result. The optimal weighting can depend on several attributes like the application domain, the quality of the ontology, the completeness of information of the service descriptions and the language being used for the description. Due to this volatile optimal weighting we propose to integrate learning mechanisms into the service matchmaking process, which adapts the weightings to the problem dynamically.

In a prior work [26], we outlined an iterative process that uses a service matchmaker to find matching services in the action selection process of a planning process. In this paper we describe the implementation and evaluation of the new matching aggregation process and different learning approaches for defining an optimal weighting for the semantic service matchmaker SeMa^2.

The remainder of the paper is structured as follows. In Section 2 we will present and discuss the related work. This includes an overview of service description languages, service matchmakers and service planners. In Section 3 we introduce the challenges associated with automated service planning and describe our concept of extending $SeMa^2$ by a comprehensive aggregation process. Subsequently, in Section 4 our approach of learning the weights for different matching techniques is presented in detail and evaluated. Eventually, we conclude the work and give an outlook on future work in Section 3.

2 Related Work

In this section we will at first give an overview about the related work of semantic service matchmakers followed by learning approaches that make service matchmaking adaptable.

2.1 Service Matchmaker

M. Klusch and *P. Kapahnke* [34] present a hybrid service matchmaking component called *iSeM*, which performs logical as well as syntactical and structural matching. iSem participated in the 2012 S3 Contest and was the only matchmaker able to process the provided PE service specifications in SWRL. Besides logical matching filters for input and output parameters the solution applied a strict logical specification plug-in matching. This means, that the component checks whether there exists a transformation of the requested/provided preconditions/effects, in order to infer from a requested precondition to a provided one and from a provided effect to a requested one. This process is called θ-subsumption and is in the case of iSem provided without any consideration of

instances. The inferencing is being done after SWRL rules are converted into PROLOG.

Another work performing PE matching is SPARQLent [55], which not only performs matchmaking but also planning. It participated in the 2012 S3 OWL-S Contest and assumes that PE are described in the query language SPARQL. Hence, the selection process is based on query containment relations not only considering PE, but also input and output concepts.

In contrast to the other approaches the work of *Valle et al.* [16] named *GLUE* is based on WSML. Since WSML already comes with a conceptual model of service discovery, this work proposes a refinement that has a specific focus on mediating goals of different ontologies. The reasoning process itself is performed with F-Logic.

In the work of *Lamparter et.al.* [36] the authors present a proprietary service definition based on OWL-DL. Beside input and output parameters, the algorithm also considers pricing functions described in SWRL and configurations under which a service is executable. The latter refers to some form of condition checking, since the requester can search for services that provide a specific, desired configuration. The reasoning on services is being done with SPARQL queries.

Bener et al. [4] extend SAM (Semantic Advanced Matchmaker) by PE matching strategies based on OWL-S and SWRL. The matching procedure for conditions is separated into four matching modules, namely subsumption based scoring, semantic distance scoring, WordNet based scoring, finalized and aggregated via a bipartite matching approach. The work introduces weights for the aggregation of different matching results, which are shown in Table 1 taken from [4].

Table 1. Scoring assessments

Relationship	**Score**
Exact	1.0
Plug-in for Preconditions	0.6
Plug-in for Effects	0.4
Subsume for Preconditions	0.4
Subsume for Effects	0.6
Fail	0.0

The weights in this case are fixed and thus not learned. Furthermore, those weights only concern discrete level of matches.

In conclusion the described approaches rely on different languages for the description of conditions ranging from decidable ones, such as OWL-DL and WSML-DL to undecidable ones, such as SWRL, F-Logic and PROLOG. However, in most of the related work on service matchmakers regarding PE matching the undecidable rule languages have been limited in its expressiveness in order to guarantee termination. So far, the service matchmakers have included different similarity measures and have introduced some weights to model their importance against each other. The weights started out to classify two types

of similarity measures and got more detailed to the point where different parts of the service description like preconditions and effect are weighted differently. These approaches have one fact in common: the choice of the weights are fixed and do not adapt to the context of use.

2.2 Learning Service Matchmaker

In order to optimize the result of the service matching a learning phase can be introduced to adjust the parameters of a service matchmaker to the properties of the domain. The parameters to learn depend on the service matchmaker and thus its flexibility depends on the parameters that can be observed.

In *Klusch et al.* [33] the authors introduced a formal model of defining weights for the aggregation of different similarity measures with the names w_w-similarity and w_s-structural similarity measure. The aggregation method has been learned using a Support Vector Machine (SVM) approach based on training data. The matchmaker component that invokes this approach is designed to match SA-WSDL services.

M. Klusch and *P. Kapahnke* [32] introduce another learning service matchmaker by extending the approach of a prior work [30] for OWL-S service descriptions. Here matching results of different matching types are aggregated using a weighted mean. The authors introduce different types of matching results that are weighted. Firstly, approximated logical matching, which is divided into approximated logical plug-in and subsumed-by matching. Secondly, non-logic-based approximated matching, which are text and structural semantic similarity-based signature matching. The weights of this aggregation are also learned using a SVM. This supervised learning approach is replicated in our work, but with a different learning algorithm. The relevance set that is used to rank the matching results are reused with a genetic algorithm and a hill-climbing search.

To the best of our knowledge there exist only these both approaches that utilize machine-learning techniques in order to cope with the challenge of aggregating service matchmaking techniques.

3 SeMa^2 and its Expert System

The service matchmaker SeMa^2 [37] follows a hybrid approach combining logic-based and non-logic-based matching techniques using OWL-S and SWRL as part of a single-agent system. In a subsequent work [26], an extension of SeMa^2 was proposed, where each of the matching techniques (in the remainder named similarity measures) are encapsulated in one agent. The resulting multi-agent system presents a group of experts, each able to express its opinion about a match by means of a probability. In the following, we will describe the implementation of this extension with a selected set of similarity measures. Afterwards we will present the probabilistic model used to aggregate the different expert opinions.

3.1 The Expert System

In this section we present the extended architecture of SeMa^2 and discuss the relation between different matching experts. An overview of the expert system is illustrated in Fig. 1.

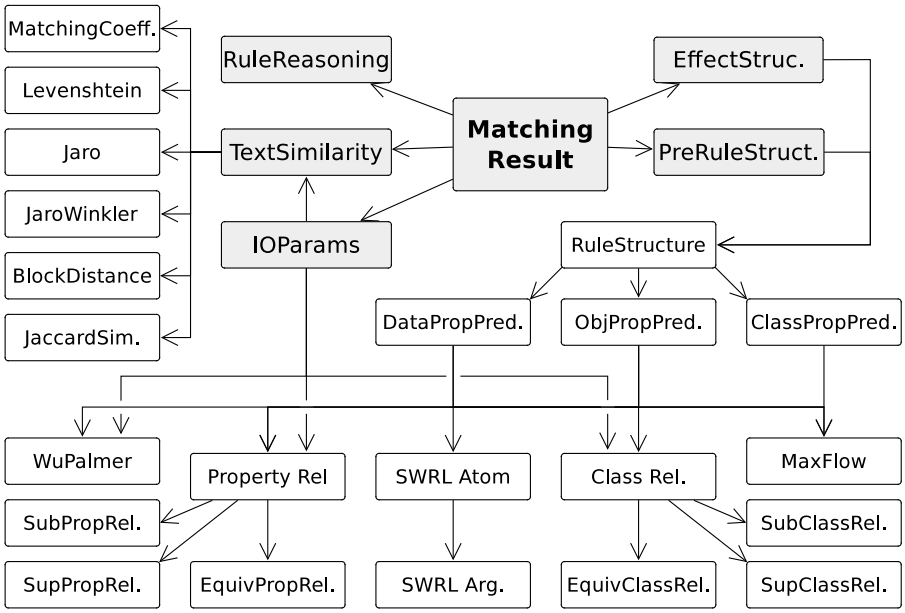


Fig. 1. Expert structure of the SeMa^2 architecture.

The schematic depiction shown in Fig. 1 highlights that lower experts are used by the upper ones to build opinions. Each node in the figure is an expert type. Each expert can have multiple instances in the implementation depending on its use in the hierarchy. The edges of the tree represent weights, which are used to aggregate the opinion of the lower expert into the upper experts opinion. For example, the *TextSimilarity Expert* uses six other experts to establish its opinion. In total, the SeMa^2 consists of 81 experts. We will now have a look at different excerpts of the experts.

We start with the overall result of the matching process, which is the opinion of the *MatchingResult Expert* which aggregates the matching results of the *TextSimilarityExpert*, the *PreRuleStructureMatcherExpert* (structural precondition matching), the *EffectStructureMatcherExpert* (structural effect matching), the *IOParamMatcherExpert* (structural input/output matching) and the *RuleReasoningMatcherExpert*, responsible for validity checks of instances on the service's rules. Fig. 1 shows those experts in gray.

The comparison of the arguments of a rule's predicate presents the next improvement opportunity. Here the used algorithm abstracts from the order the parameters are used but uses a 'is equal' as comparison. This has the advantage to produce a binary acceptance criteria but lacks the ability to recognize hypernym or hyponym concepts. The proposed extension here is the use of a probabilistic measure which is able to distinguish a semantic distance between the concepts of interest. Such measures are subject of research, e.g. *Benner* [4] uses such measures for service matchmaking. Again, we define such measures as expert opinion and use an information fusion method to aggregate such opinions.

3.2 Scoring and aggregation of matching techniques

There are many different similarity measures to extend the PE matching in a probabilistic framework similar to the related work (cf. [4, 34]). Some of them are: *Semantic distance based scoring* [62] analyzing the embedded ontology of two concepts to find the shortest path from one concept to another; *WordNet based scoring* [4] that can be used to find lexical similarity in used words. If two concepts out of different ontologies need to be matched a *bipartite matching score* is able to rate the similarity by i.e. the maximum cardinality match counting the edges between the different concepts. More sophisticated methods use ontology matching to find a semantic relation between concepts. Logic based scoring like proposed in *Approximated Logical Matching* [34] can be interpreted as similarity measures using reasoning on formal features of the rules describing the preconditions and effects.

Those similarity measure each compute a similarity score interpreted as an probability where 1.0 is a perfect match and 0.0 is no match at all. We now have a look on how a probabilistic model can formalize such an expert opinion.

Probabilistic model of opinion Like in our previous work [26] we apply the results of *Morris* [43] to model the expert opinions as probabilities $p_i(R, S)$. We will have a closer look into this formalization in the following section. As an expert observes two concepts and elaborates their semantic distance we can abstract its opinion as $p_i(\Theta|d)$ where Θ is the subject of interest and d are the observations. An expert can then collect evidence for its opinion by conducting multiple observations d_i. Each observation might then be interpreted as evidence to strengthen the experts' opinion. Following *Beyerer* [6] a Bayesian interpretation of the conditional probability $p_i(\Theta|d)$ could be interpreted as a degree of confidence or even better as a degree of belief. With such an interpretation we can use this formalism to model the expert opinions as described in the following equation:

$$\underbrace{p(\Theta|d)}_{A-Posteriori} = \frac{p(d|\Theta)p(\Theta)}{p(d)} \propto \overbrace{p(d|\Theta)}^{Likelihood-Function} \underbrace{p(\Theta)}_{A-Priori} . \tag{1}$$

Here the subject of interest is Θ, *i.e.* the equivalence of a Horn-clause. The observations or information used by the expert to assess its opinion is formalized in d. An example of this d could be the attached ontologies to the concept in order to calculate the semantic distance. The expert can update its opinion after observing another d using Bayesian fusion by calculating the product described in Eq. 1. If, for example, one concept is a hypernym of the other, $p(d|\Theta)$ could be proportional to the minimal distance between those two concepts [62]. Furthermore, $p(\Theta)$ allows the expert to formalize a-priory knowledge about the probability of Θ.

For the same request and advertisement pair the different measures might have different similarities. And with that, the experts have different opinions. If this happens, an aggregation of the different opinions needs to be found.

Opinion aggregation As we have shown elsewhere [26] different aggregation functions -so called pooling methods- can be used to aggregate the opinions of experts. The opinions $p_i(\Theta|d)$ are collected and need to be fused to one score. Since the experts are not always equally important the possibility to prioritize by weightings of the different expert opinions is a requirement for the fusion method. With the probabilistic formalization of the expert opinions method like the *Dempster-Schafer theory of evidence* [56], fuzzy logic or artificial neuronal networks can be used for information fusion [24]. Subsequently to prior work [26] this work uses a method of opinion aggregation called pooling method formalized in a function $K(p_1, \ldots, p_n)(\Theta)$. It is acquired by adapting a weighted mean to the aggregation of opinions. We choose a weighted arithmetic mean called linear opinion pool [59] as our first pooling method. This arithmetic mean has been generalized by *Genest* [27] to be able to use weights in the interval $[-1, 1]$ in a more general class of linear opinion pools (GenLinOP). This opinion pool has the form of Eq. 2.

$$K(p_1, \ldots, p_n)(\Theta) = \sum_{i=1}^{n} w_i p_i(\Theta) + \left[1 - \sum_{i=1}^{n} w_i\right] R(\Theta) \tag{2}$$

Here, $w_1, \ldots, w_n \in [-1, 1]$ are weights representing the confidence in an expert and R is an arbitrary probability function, with the restriction:

$$\forall J \subseteq \{1, \ldots, n\} : \left|\sum_{j \in J} w_j\right| \leq 1.$$

The method shown in Eq. 2 has been chosen because of its theoretical sound standing. Other pooling methods have been and are continued to be evaluated which is subject to research. The GenLinOP has the possibility to include—besides the opinion of the group—an a-priori established probability which can be modeled as $R(\Theta)$. We normalize the weights to $\left|\sum_{j \in J} w_j\right| = 1$ which lets us neglect the a-posteriori opinion $R(\Theta)$.

The GenLinOP is compared against the Logarithmic opinion pool [28] (LogOP) and the weighted harmonic mean (HARM) since the GenLinOP has several

theoretic weaknesses [3] like allowing dictatorships. There is a infinite number of pooling methods where each of which has its own properties which are of concern in different applications. The LogOP for example has the curios behavior of a 'veto'. Meaning that if a single expert has the opinion of $p_i(\theta) = 0$ then the aggregated opinion is $K(p_1(\theta), ..., p_n(\theta)) = 0$ independent of the given weights. Eq. 3 describes the LogOP:

$$K(p_1(\theta), ..., p_n(\theta)) = \frac{\prod_{i=1}^{n} [p_i(\theta)]^{w_i}}{\sum_{\theta \in \Theta} \prod_{i=1}^{n} [p_i(\theta)]^{w_i}} \tag{3}$$

with weights $w_i \in [0, 1]$ and $\sum_{i=1}^{n} w_i = 1$.

One of the properties which make the LogOP interesting for the opinion aggregation is that it as what is called the unanimity property which states that if all experts have the same opinion, the aggregated opinion is equal to this opinion.

The third aggregation method we will analyze is the weighted harmonic mean.

$$K(p_1(\theta), ..., p_n(\theta)) = \frac{\sum_{i=1}^{n} w_i}{\sum_{j=1}^{n} \frac{w_j}{p_j(\theta)}} \;\middle|\; w_i \in [0, 1]; \sum_{i=1}^{n} w_i = 1; p_j(\theta) \neq 0. \tag{4}$$

The harmonic mean has the down side of not allowing opinions of 0. Which leads to the interpretation of the expert opinion that there is no total miss match at all. Such an postulate is subject to discussion and depends on the domain, which is model by the experts.

Taking this theoretical framework as a basis, we implement the different measures used in the service matching as experts returning a probability $p_i(\Theta)$ and aggregate them with a pooling method $K(p_1, \ldots, p_n)(\Theta)$. For an example we have adapted the comparison of the arguments of a predicate. The expert opinion modeled as probability is as follows:

$$p(\Theta) = \begin{cases} dist(a_r, a_s)^{-1} & \text{, if } 1 \geq dist(a_r, a_s) > 0 \\ 1.0 & \text{, if } a_r.getIRI() \equiv a_s.getIRI() \\ 0 & \text{, else} \end{cases}. \tag{5}$$

This leads us to the question on how to measure the distance between concepts. This question has been subject to research, e.g. by *Euzenat* [22]. The next section gives a short overview of one conceptual metric implemented in our approach as proof of concept.

In Eq. 5 the first case ($dist(a_r, a_s)^{-1}$) defines the distance between the two concepts in different ways regarding the distance appropriate to the properties of the concept. Each distance measure is implemented in one expert and we have e.g. identified Class-Property, Data-Property, Object-Property and Class-Relationship experts in SeMa^2. Class properties are well know and often analyzed

in the related work. The Pellet reasoner supports reasoning on class properties and as example distance between concepts the Wu-Palmer measure [63] shown in Eq. 6 has been implemented in one expert.

$$dist_{WP}(a_r, a_s) = \frac{2 \times \eth(\triangle(a_r, a_s))}{\eth(a_r) + \eth(a_s)}. \tag{6}$$

With $\eth$ being the distance to the root element of the tree and $\triangle$ is the lowers common subsumer of the concepts a_r and a_s. Since the for Class-Properties $dist_{WP}$ can be calculated using Pellet but for Object- and Data-Properties Pellet denies reasoning support so an own implementation has been created.

In a similar manner all other similarity measures are turned into probabilistic expert opinions. With this change e.g. we are able to distinguish partial argument matches. We want to emphasize the importance of such a partial match for planning tasks. Here multiple services can be used to fulfill the arguments of a predicate in a precondition. Thus on a higher level, we are able to use multiple services to fulfill the preconditions of a successor task or service.

The matching on arguments is done by a bipartite graph matching solved with the max flow algorithm, where the edges between request and advertisement nodes are the opinion of the expert. Fig. 2 shows an example of such an bipartite graph matching problem.

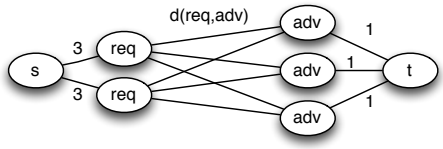


Fig. 2. Example of the max flow problem to solve for a bipartite graph matching.

Here the edges exiting the source s have the weight of the sum of the edges entering the sink t because in this way if one request parameter matchmaker all advertisement parameters the maximal flow can be seen as perfect match. The result of the maximal flow though this graph is divided by the sum of the weights of edges which enter the sink and forms the opinion of the 'MaxFlowExpert'.

One particular challenge is the measure for a class relationship in regards to a matching tasks. In the related work this is sometimes called subsumes and subsumed-by relation (cf. [34]). The question at hand here is: If an exact match of concepts in a ontology is detonated with 1.0 and no match is 0.0 how much is a sub- or super match? Since all choices done here would be arbitrary, we decided to let the service matchmaker learn those parameters in an offline stage prior the actual matching. The grey marked sets (relationship experts) in Fig. 1 are such experts, where only one expert might return 1.0 and the other converge to 0.0. In this case the weights for these experts denote the influence of an exact-, super- or sub-match.

The learning is done via a genetic algorithm using the Watchmaker Framework[1]. Where the weights for the 81 experts is seen as a DNA pattern and are mutated at each generation. The Mutation and the details of the assessment of each generation is explained in the next section.

The result of the learning is a weight vector describing the confidence in an expert regarding its opinion, which are used in the Pooling Methods. Since the expert structure is not flat some of the experts depend on opinions of other experts and with that the weights are dependent as well. Furthermore, the learning takes place on the expert instances and not on a general scope since the same expert, e.g. the TextSimilarityExpert, can be used to evaluate different properties of a service description. For example, the name of the service might have more affect on the service matching result then a text similarity of some parameter. Thus the weights for those each instances has to be learned separately.

Further it is questionable if the learning of the weights of the pooling method is significant.

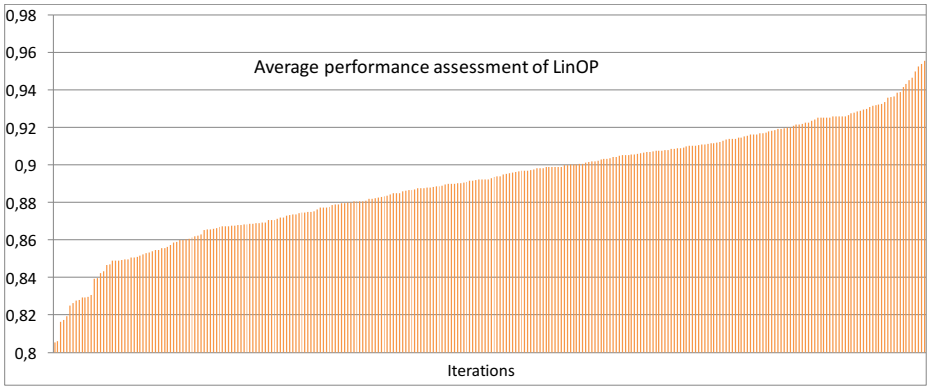


Fig. 3. Example weight generations during the learning (X-axis) and their assessment (Y-axis) of the LinOP.

As shown in figure 3 the different weight vectors yield different results of the LinOP reaching form 0.8 to 0.95 thus the selection of weights is significant.

4 Learning the Weights

This section will introduce the learning approach for the weights of the aggregation methods and presents the first experimental results. As mentioned above we used genetic algorithms to optimize the weights of the aggregation methods.

Similar to *Klusch* [31] the learning is done in an offline phase before the matching task. Giving the service matchmaker an opportunity to adapt to the domain it is used in.

[1] http://watchmaker.uncommons.org/

Learning algorithm We used the Semantic Web Service Matchmaker Evaluation Environment (SME) [2] for the evaluation. The test collection of services and request from the S3 Contest[3] with 1080 services and 42 requests has been used. This test collection provides a relevance rating consisting of sets of relevant and irrelevant services for the 42 requests. This is used to evaluate the matching performance of the service matchmaker with varying weights. As quality measure the normalized discounted cumulative gain (NDCG) is used. This quality measure optimizes the amount of relevant service with high matching values as shown in Eq 7.

$$DCG = \sum_{i=1}^{n} \frac{2^{rel_i} - 1}{log_1(i+1)} \;\Big|\; NDCG = \frac{DCG}{Best\ DCG} \tag{7}$$

With these preconditions a supervised learning method has been chosen: Genetic Algorithms. We used the Watchmaker Framework for the support of the generic algorithm life cycle. The genome is modeled as a list of all 81 weights each reaching from 0 to 1 with the restrictions among siblings having their weights normalized to sum up to 1. The first generation is initialized with random weights. With each generation the experts with better NDCG values have thus higher weights, until one round of an N-Fold cross validation is completed. We have implemented an mutation function, which changes the weight of an expert depending on the NDCG value he reached. Following this approach the experts with higher NDCG value have bigger weights in the next generation of the weight vector. The evaluation has been done with an N-fold cross-validation with different N reaching from 1 (single fold) over 7 to 42 (leave one out). The results which are shown in the next section have been taken from the 7-fold-cross-validation.

Evaluation In this section we unveil the results of the learning phase of the $SeMa^2$ regarding the different pooling methods. The results have been produced in three experiments where each experiment has run for approximately one day. We have to notice that not the absolute height of the weights are subject to research, since those weights might change with the domain they are learned in. We rather emphasize that the learning yield different results depending on the pooling method and further that the quality of the matching result highly depends on the weights chosen.

The result of the learning process using the HARM as aggregation method are depicted in Fig. 4: One is the average weights (blue) and the other one is the weight configuration yielding the best matching results (orange) of an NDCG 0.9195.

To compare our results to the ones of *Benner* [4] and the ones published by *Klusch* and *Kapahnke* [32] which do weight the first level of similarity measures, we have highlighted the first level of experts in the weight distributions in Fig. 4,

[2] http://projects.semwebcentral.org/projects/sme2/
[3] http://www-ags.dfki.uni-sb.de/ klusch/s3/

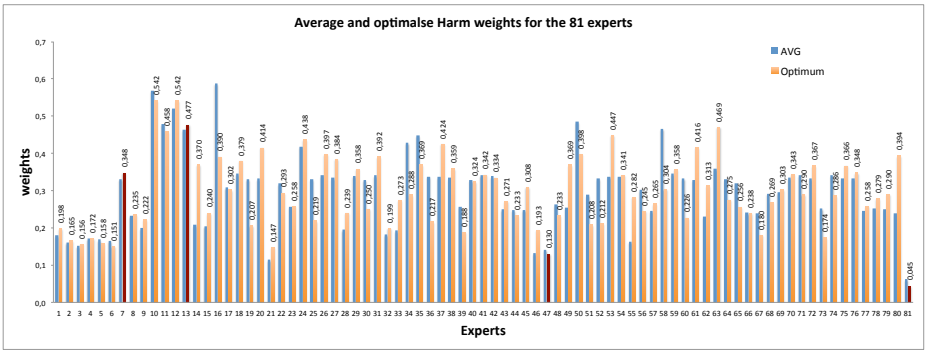


Fig. 4. Learned weights for the 81 experts using HARM.

5 and 6 in the order from left to right: Expert 7 being the *TextSimilarity Expert*, 13 the *IOParamMatcher Expert*, 47 the *PreRuleStructureMatcher Expert* and 81 the *EffectStructureMatcher Expert*. It has to be noted that the hierarchy of experts has multiple levels and only the first one is emphasized here. Additionally the weights are not shown as final weights but rather as the assigned weights. To get the actual influence of the expert to the overall result the weights in the hierarchy need to be multiplied so that the relevance of the children is weighted with the relevance of the parent expert. This is not done here since the overall weights are not of interest since they are domain dependent.

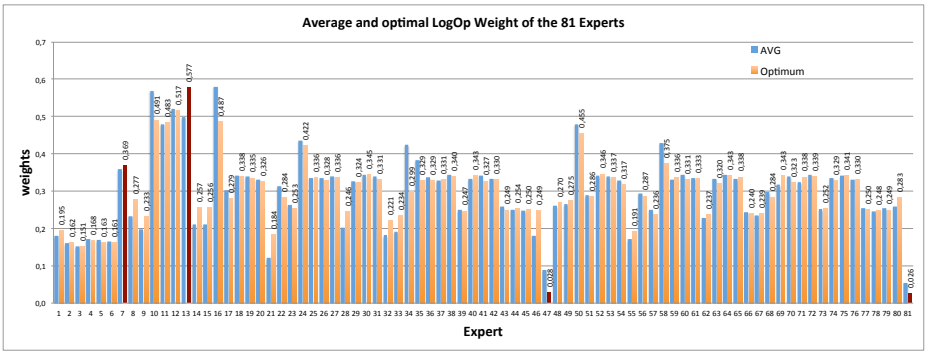


Fig. 5. Weighted logarithmic Mean

Secondly the resulting weights of the learning using the LogOP is shown in Fig. 5. The LogOP has reached a optimum of 0.8932, which is the lowest of the three aggregation methods analyzed. For the LogOP aggregation the optimal weights are mostly lower than the average. Since the weights are in the interval $[0, 1]$ and they are used in an exponent, opinions with small weight convert faster against 1.0, thus in the product of the weighted opinions the opinion is ignored.

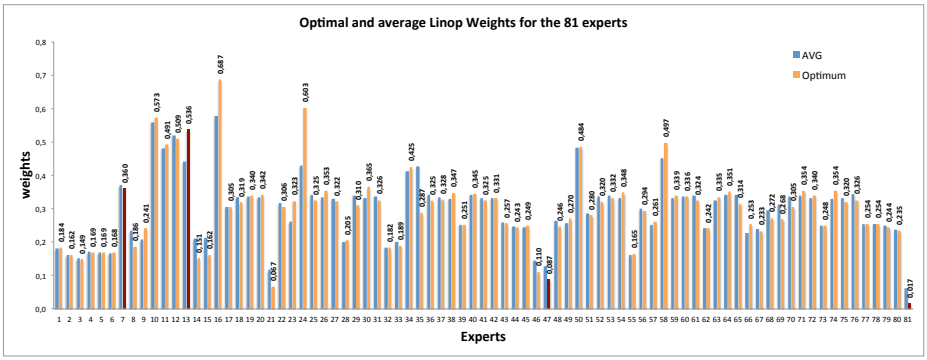


Fig. 6. Weighted arithmetic Mean

The LinOP, shown in Fig. 6, has reached the highest matching assessment of 0.9553 of the three aggregation methods. It can be noticed that the LinOP does tend to the extremes of the weights. This can be seen for instance on agent number 16 and 21, where in both cases the weights are the smallest and the biggest one of the three weight sets. Furthermore the normalization of the weight seams to overcast the result of the learning. Since the sum of the weights are always normalized to 1.0, experts with many siblings do have smaller weights than experts with less siblings. The weights for expert number 1 to 6 are a good example for that, since those are the text similarity experts. It can be noticed that the weight for the different pooling methods are similar. This is in first place because of the bias of the normalization and secondly it is because the experts are weighted due to their performance.

Furthermore all three results of learning the weights seam open to scrutiny since the weights in average underestimate the optimal solution found for big weights and overestimate them when looking at small weights. That behavior is introduced by the the random element in the mutation step of the genetic learning algorithm. Here in small weights, the probability to draw a bigger weight during the leaning is high, thus in average the weight should be higher than the optimal and vice versa with big weights. This can be seen in Fig. 6 with experts 81 and 24.

Overall the weights for the first level of experts can be compared to the result of *Benner* [4] or *Klusch* and *Kapahnke* [32] but it is not a general proposition of weights for semantic service matching since for instance only a few of the services of our test collection have a description of their effect. Thus the weights for the effect matching are low. In other domains the textual description of the service for instance might be neglected, which results in an reduction of the text similarity experts weights.

The logical conclusion is to use an LinOP as an pooling method since it yielded the best results. But we are further intrigued by the different properties of the pooling methods and will analyze further aggregation methods in different domains.

5 Conclusion

We have created an multi-agent expert system realizing a service matcher, where every expert estimates an similarity measure of a service request to a service advertisement. This matching process is seen as an action selection step in an agent planning algorithm. Each expert postulates an opinion concerning the match regarding a part of the service description and advertisement. The different expert opinions are aggregated using pooling methods. The main contribution of this article is the experimental evaluation of different pooling methods namely the Linear and Logarithmic Opinion Pool and the Weighted Harmonic Mean. For this evaluation the weights of the pooling methods are learned using an genetic algorithm and the result are evaluated using a standard service collection. Regarding the overall performance the weighted arithmetic mean seams favorable for this service collection since it produced the best overall matching result of an NDCG value of 0.9553.

With this approach an agent might adapt its action selection process to the domain of use resulting in a performance increase during the planning. The SeMa^2 has been used and extended in an software development methodology in [25] by using service planning to create service compositions. Selecting one service to satisfy a query assumes that this given query has been foreseen and a corresponding service has been implemented. Without loss of generality we assume that this is not always the case, making it necessary to compose multiple services to fulfill a query. Thus for future work the challenge of utilizing such an service matcher in an planning algorithm remains. In future work we will integrate this selection process into the JIAC V agent framework (see: http://www.jiac.de).

References

1. A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwiga, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. Web services agreement specification (WS-agreement). Technical report, 2007. `https://www.ogf.org/documents/GFD.107.pdf`.
2. Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787 – 2805, 2010.
3. Jon A Benediktsson and P H Swain. Consensus theoretic classification methods. *Systems, Man and Cybernetics, IEEE Transactions on*, 22(4):688–704, 1992.
4. Ayse B Bener, Volkan Ozadali, and Erdem Savas Ilhan. Semantic matchmaker with precondition and effect matching using SWRL. 36(5):9371–9377, 2009.
5. Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, and Daniele Riboni. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2):161 – 180, 2010.
6. Jürgen Beyerer. *Verfahren zur quantitativen statistischen Bewertung von Zusatzwissen in der Messtechnik*, volume 8 of *VDI Fortschritt-Bericht*. VDI/Verl., 1999.
7. Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the*

MCC Workshop on Mobile Cloud Computing, pages 13–16, New York, NY, USA, 2012. ACM.

8. Andreas Bulling, Ulf Blanke, and Bernt Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Computing Surveys*, 46(3):33:1–33:33, 2014.
9. Javier Cámara, Gabriel A. Moreno, and David Garlan. Reasoning about human participation in self-adaptive systems. In *10th IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2015, Florence, Italy, May 18-19, 2015*, pages 146–156, 2015.
10. Saisakul Chernbumroong, Shuang Cang, Anthony Atkins, and Hongnian Yu. Elderly activities recognition and classification for applications in assisted living. *Expert Systems with Applications*, 40(5):1662 – 1674, 2013.
11. Federico Ciccozzi and Romina Spalazzese. MDE4IoT: Supporting the Internet of Things with Model-Driven Engineering. In *Proc. of the 10th International Symposium on Intelligent Distributed Computing (IDC'16)*. Springer - (To appear), 2016.
12. N. Criado, E. Argente, and V. Botti. Open issues for normative multi-agent systems. *AI Commun.*, 24(3):233–264, 2011.
13. Valentin Cristea, Ciprian Dobre, and Florin Pop. *Internet of Things and Inter-cooperative Computational Technologies for Collective Intelligence*, chapter Context-Aware Environments for the Internet of Things, pages 25–49. Springer Berlin Heidelberg, 2013.
14. Luigi De Russis and Fulvio Corno. Homerules: A tangible end-user programming interface for smart homes. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pages 2109–2114. ACM, 2015.
15. Jose Delgado. *Internet of Things and Inter-cooperative Computational Technologies for Collective Intelligence*, chapter Service Interoperability in the Internet of Things, pages 51–87. Springer Berlin Heidelberg, 2013.
16. Emanuele Della Valle, Dario Cerizza, and Irene Celino. The mediators centric approach to automatic web service discovery of glue. *MEDIATE2005*, 168:35–50, 2005.
17. Anind K. Dey. Understanding and using context. *Personal Ubiquitous Computing*, 5(1):4–7, 2001.
18. Anind K Dey, Raffay Hamid, Chris Beckmann, Ian Li, and Daniel Hsu. a cappella: programming by demonstration of context-aware applications. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 33–40. ACM, 2004.
19. Nicolas Ducheneaut, Trevor F Smith, James Bo Begole, Mark W Newman, and Chris Beckmann. The orbital browser: composing ubicomp services using only rotation and selection. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems*, pages 321–326. ACM, 2006.
20. W Keith Edwards, Mark W Newman, Jana Z Sedivy, and Trevor F Smith. Experiences with recombinant computing: Exploring ad hoc interoperability in evolving digital networks. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 16(1):3, 2009.
21. Thomas Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.
22. Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer-Verlag Berlin Heidelberg, 2007.

23. Dave Evans. How the next evolution of the internet is changing everything. Technical report, Cisco Internet Business Solutions Group, 2011. `http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf`.
24. Johannes Fähndrich. *Analyse von Verfahren zur Kombination von Expertenwissen in Form von Wahrscheinlichkeitsverteilungen im Hinblick auf die verteilte lokale Bayes'sche Fusion*. PhD thesis, Karlsruhe Institut of Technology, May 2010.
25. Johannes Fähndrich, Tobias Küster, and Nils Masuch. Semantic Service Management and Orchestration for Adaptive and Evolving Processes. *International Journal on Advances in Internet Technology*, 9(4):75–88, 2016.
26. Johannes Fähndrich, Nils Masuch, Hilmi Yildirim, and Sahin Albayrak. Towards Automated Service Matchmaking and Planning for Multi-Agent Systems with OWL-S – Approach and Challenges. In *Service-Oriented Computing – ICSOC 2013 Workshops*, pages 240–247. Springer International Publishing, Cham, January 2014.
27. Christian Genest. Pooling operators with the marginalization property. *The Canadian Journal of Statistics/La Revue Canadienne de Statistique*, 12(2):153–163, 1984.
28. Christian Genest, S Weerahandi, and J V Zidek. Aggregating opinions through logarithmic pooling. 17(1):61–70, June 1984.
29. Yucheng Jin, Chi Tai Dang, Christian Prehofer, and Elisabeth André. A multi-display system for deploying and controlling home automation. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, pages 399–402. ACM, 2014.
30. Matthias Klusch, Benedikt Fries, and Katia Sycara. OWLS-MX: A hybrid Semantic Web service matchmaker for OWL-S services. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(2):121–133, April 2009.
31. Matthias Klusch, Andreas Gerber, and Marcus Schmidt. Semantic web service composition planning with owls-xplan. pages 55–62, 2005.
32. Matthias Klusch and Patrick Kapahnke. The iSeM matchmaker: A flexible approach for adaptive hybrid semantic service selection. 15:1–14, September 2012.
33. Matthias Klusch, Patrick Kapahnke, and Ingo Zinnikus. SAWSDL-MX2: A Machine-Learning Approach for Integrating Semantic Web Service Matchmaking Variants. In *2009 IEEE International Conference on Web Services (ICWS)*, pages 335–342. IEEE Computer Society, IEEE, 2009.
34. Matthias Klusch, Ulrich K uster, Alain Leger, David Martin, and Massimo Paolucci. 5^{th} International Semantic Service Selection Contest - Performance Evaluation of Semantic Service Matchmakers. November 2012.
35. Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explorations Newsletter*, 12(2):74–82, 2011.
36. Steffen Lamparter and Anupriya Ankolekar. Automated selection of configurable web services. *8. Int. Tagung Wirtschaftsinformatik*, 2007.
37. Nils Masuch, B Hirsch, M Burkhardt, A Heßler, and S Albayrak. SeMa^2: A Hybrid Semantic Service Matching Approach. In *Semantic Web Services*, pages 35–47. Springer Berlin Heidelberg, 2012.
38. Christos Mettouris and George A. Papadopoulos. Ubiquitous recommender systems. *Computing*, 96(3):223–257, 2014.
39. Radu-Casian Mihailescu and Paul Davidsson. Integration of smart home technologies for district heating control in pervasive smart grids. In *Proceedings of the First International Workshop on Pervasive Smart Living Spaces*, 2017.

40. Radu-Casian Mihailescu, Paul Davidsson, and Jan A. Persson. Multiagent model for agile context inference based on articial immune systems and sparse distributed representations. In *Multi-Agent Systems and Agreement Technologies - 13th European Conference, EUMAS 2015, and Third International Conference, AT 2015, Athens, Greece, December 17-18, 2015, Revised Selected Papers*, pages 82–87, 2015.
41. Radu-Casian Mihailescu, Matthias Klusch, and Sascha Ossowski. e COOP: privacy-preserving dynamic coalition formation for power regulation in the smart grid. In *Proceedings of the Second International Conference on Agreement Technologies, AT 2013, Beijing, China, August 1-2*, pages 19–31, 2013.
42. Radu-Casian Mihailescu, Jan A. Persson, Paul Davidsson, and Ulrik Eklund. Towards collaborative sensing using dynamic intelligent virtual sensors. In *Intelligent Distributed Computing X - Proceedings of the 10th International Symposium on Intelligent Distributed Computing - IDC 2016, Paris, France, October 10-12 2016*, pages 217–226, 2016.
43. P A Morris. Combining expert judgments: A Bayesian approach. *Management Science*, 23(7):679–693, 1977.
44. Sascha Ossowski and Ronaldo Menezes. On coordination and its significance to distributed and multi-agent systems. *Concurrency and Computation: Practice and Experience*, 18(4):359–370, 2006.
45. Mike P. Papazoglou and Willem-Jan Heuvel. Service oriented architectures: Approaches, technologies and research issues. *The VLDB Journal*, 16(3):389–415, July 2007.
46. C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE Communications Surveys Tutorials*, 16(1):414–454, 2014.
47. Per Persson and Ola Angelsmark. Calvin – merging cloud and IoT. *Procedia Computer Science*, 52:210 – 217, 2015.
48. Ferdinand Piette, Cédric Dinont, Amal El Fallah Seghrouchni, and Patrick Taillibert. Deployment and configuration of applications for ambient systems. In *Proceedings of the 6th International Conference on Ambient Systems, Networks and Technologies*, ANT-2015, pages 373–380. Procedia Computer Science, 2015.
49. Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976 – 990, 2010.
50. Davy Preuveneers and E Berbers. Automated context-driven composition of pervasive services to alleviate non-functional concerns. *International Journal of Computing and Information Sciences*, 3:28–38, 2005.
51. R. de Lemos et al. Software engineering for self-adaptive systems: A second research roadmap. In Rogério de Lemos, Holger Giese, Hausi A. Müller, and Mary Shaw, editors, *Software Engineering for Self-Adaptive Systems II*, volume 7475, pages 1–32. Springer-Verlag, 2013.
52. Mohammad Abdur Razzaque, Marija Milojevic-Jevric, Andrei Palade, and Siobhan Clarke. Middleware for internet of things: A survey. *IEEE Internet of Things Journal*, 3(1):70–95, 2016.
53. Christof Roduner, Marc Langheinrich, Christian Floerkemeier, and Beat Schwarzentrub. Operating appliances with mobile phones–strengths and limits of a universal interaction device. In *Pervasive Computing*, pages 198–215. Springer, 2007.
54. Maria J. Santofimia, Francisco Moya, Felix J. Villanueva, David Villa, and Juan C. Lopez. An agent-based approach towards automatic service composition in ambient intelligence. *Artificial Intelligence Review*, 29(3):265–276, 2009.

55. Marco Luca Sbodio. SPARQLent: A SPARQL Based Intelligent Agent Performing Service Matchmaking. pages 83–105. Springer Berlin Heidelberg, Berlin, Heidelberg, April 2012.
56. G Shafer. *A mathematical theory of evidence*, volume 1. Princeton university press, 1976.
57. R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113, 1980.
58. Thanos G. Stavropoulos, Dimitris Vrakas, Danai Vlachava, and Nick Bassiliades. BOnSAI: A smart building ontology for ambient intelligence. In *Proceedings of the Second International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, pages 30:1–30:12, New York, NY, USA, 2012. ACM.
59. M Stone. The opinion pool. *The Annals of Mathematical Statistics*, 32(4):1339–1342, 1961.
60. The Fundation of Intelligent Physical Agents. FIPA specifications. Technical report, 2002. `http://www.fipa.org/repository/standardspecs.html`.
61. Michael Wooldridge. *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition, 2009.
62. Jian Wu and Zhaohui Wu. Similarity-based web service matchmaking. *Services Computing*, 2005.
63. Zhibiao Wu and Martha Palmer. Verbs Semantics and Lexical Selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.
64. L. D. Xu, W. He, and S. Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, 2014.
65. Zheng Yan, Peng Zhang, and Athanasios V. Vasilakos. A survey on trust management for internet of things. *Journal of Network and Computer Applications*, 42:120 – 134, 2014.

Some Thoughts on Programming Models, Middleware and Self-Healing Capabilities for the Next-Generation Internet-of-Agents

Predrag T. Tošić

School of EECS, Washington State University
Pullman, Washington, USA
Pedja.tosic@gmail.com

Abstract. *Internet-of-Things*(IoT) is one of the most important new paradigms and technological advances in the realm of the "consumer", soon-to-be-present-in-every-household Internet-powered cyber-physical systems of this decade and likely of many years to come. We are interested in the distributed intelligence, agent-based programming and multi-agent systems aspects of what it will take to enable the reliable, secure, interoperable and human-friendly next-generation IoT, and *Internet-of-Agents* (IoA) as IoT's software and computational intelligence foundation. In this position paper, three important aspects of distributed intelligence for IoT/IoA are briefly discussed. One, we address suitable programming abstractions for the software agents that would provide much of IoT's inter-operability, enabling different users, devices and platforms to communicate and coordinate with each other. We revisit the key ideas behind agent-based programming, and focus on the Actor model as a potentially very suitable programming paradigm for an inherently (in both physical and logical senses) open, distributed and heterogeneous infrastructure such as the IoA. Second, we briefly discuss some critical desiderata for the next-generation IoT middleware design. Last but not least, to address cyber-security aspects of IoT holistically, we outline some of the elements of computational intelligence that could enable the self-healing and self-recovery capabilities of the next-generation IoT. With such self-healing capabilities, future cyber-attacks would cause much less disruption, and have more "sand-boxed" and shorter-lasting adverse impact on the IoT users.

Keywords: Internet-of-Things/Internet-of-Agents · Distributed Intelligence · Agent-Oriented Programming · Multi-Agent Systems · Self-Healing Systems

1 Introduction and Motivation

Internet-of-Things (IoT) is arguably one of the most important new paradigms and technological advances in the realm of the "consumer", present-in-every-household Internet-powered cyber-physical systems of this decade. Moreover,

while it will likely undergo many architectural and other changes, IoT is here to stay, which is why it has been a focus of both industry leaders in Internet technologies, smart devices, platforms, and applications, and increasingly also among the academic researchers.

Due to inherently open and heterogeneous nature of IoT, as well as the needs for preserving user's data privacy, confidentiality and integrity, IoT architectures and deployments need to address security and privacy from the get-go as the first-order design and implementation concerns [1]. On the other hand, a large (and ever growing) number of heterogeneous devices renders necessary the design that will ensure interoperability, integration and scalability across a variety of users, devices and compute platforms [2]. From a system architecture standpoint, these challenges of IoT design and deployment imply that the IoT middleware needs to be more flexible, layered/modular and complex than the familiar middleware platforms deployed with "traditional" distributed and cyber-physical applications.

The list of technological and research challenges behind enabling the next-generation IoT is quite long and versatile. We are particularly interested in the middleware architecture aspects, as well as the applied AI, and especially distributed intelligence and multi-agent systems aspects, of what it will take to enable the reliable, secure, inter-operable and human-friendly next-generation Internet-of-Things.

The promise of *Internet of Things* (IoT) and, in particular, *Internet of Agents* (IoA) as the IoT's distributed software and computational intelligence foundation, lie in enabling different devices, platforms and technologies to effectively interact with each other, including communication, coordination, and various forms of cooperation. Since different agents in this "soup" of IoT/IoA in general represent different human users or organizations, these agents will have different goals with respect to each other [3]. Sometimes, these goals may even be mutually exclusive or conflicting. Just like in human societies, certain groups of agents will have better aligned interests and goals with each other than with agents outside of their "circle". Similarly, again just like with human societies, certain agents will be more reputable or trustworthy than others when it comes to delivering on promises or commitments made as a part of a multi-agent cooperation, or with respect to the quality of expertise they provide about a particular domain [4].

Some research prototypes of IoT architectures have attempted to incorporate cognitive-like capabilities into the middleware design from the get-go; one such prominent example is the iCORE project, based on the Virtual Objects (VO) paradigm [5]. These VOs are semantically rich virtual representation of the capabilities and resources provided by real-world objects and devices. As such, VO paradigm and iCORE design build on the rich research literature on distributed resource and task allocation in heterogeneous multi-agent systems – another example of how computational intelligence and Distributed AI contribute to advancing IoT state-of-the-art.

These observations suggest some important elements, indeed a roadmap, for the next-generation AI/MAS research and design of effective autonomous soft-

ware agents for this up-and-coming Internet of Agents (IoA). The focus of [3] is on the *desiderata* pertaining to intelligent software agents' abilities to coordinate and cooperate with other agents and create coalitions, taking into account other agents' capabilities, "expertise" and reputations.

The rest of the paper organizes our main ideas into three parts. First, we discuss what would be the most appropriate programming abstractions for the NG IoT/IoA; in that context, we revisit the Agent-based Programming paradigm, and then argue that an extended Actor model is a suitable programming paradigm for the next-generation IoT. We then discuss the challenges and some desiderata for the NG IoT middleware design. Last but not least, we discuss the self-monitoring/healing/recovery aspects of IoT, and argue that the applied AI techniques combined with appropriate computational "capture" of the self-adapting and self-recovery capabilities of biological organisms, may provide the right paradigm and design principles for making an intrinsically open, heterogeneous and very unpredictable cyber-physical system such as IoT capable of adapting to and surviving various types of failures and cyber-attacks.

2 Agent-based Programming for IoT and IoA

Agent-oriented programming (AOP) [6] emerged back in the 1990s as a novel programming paradigm spurred by the growth in intelligent agent software technologies and applications at that time. Tracing history of programming languages and how the nature of dominant applications has been driving the most popular programming paradigms during different eras of computing, AOP however can be contextualized as (certainly, in hind-sight) a very natural successor to the object-oriented programming (OOP) paradigm (e.g., [7]). In particular, the transition from object-oriented towards agent-oriented programming was motivated by the emergence of open distributed computing platforms in the late 1980s and throughout 1990s, which naturally required that concurrency and resource sharing be explicitly controllable and therefore easy to capture and exploit in the underlying programming model.

As computing started becoming increasingly distributed both physically and logically, and these distributed systems getting increasingly heterogeneous, complex and open, the individual components of complex software systems started being transitioning from non-autonomous components of typically either very monolithic or else hierarchical architectures towards each such component becoming increasingly sophisticated, autonomous and complex (sub)system in its own right, with the overarching architecture entailing only a loose coupling of many such components into an overarching larger system. Further, such designed systems started becoming of larger scales, highly decentralized, and usually based on a peer-to-peer (P2P) paradigm, as opposed to being either centralized or hierarchical. This, in a nutshell, is our view of how the AOP paradigm, attempting to meet the increasing demands in terms of autonomy, flexibility and complexity of the individual components in such distributed systems, was born [6].

We emphasize key characteristics of (what were back then, emerging) agent-based applications of the 1990s that required this programming paradigm and methodology shift with respect to the "old school" imperative programming (and, to a lesser extent, also the "traditional" object-oriented programming):

- larger scales and higher degree of system complexity;
- physical and logical decentralization of compute tasks, resources and data;
- increasing heterogeneity of computing resources, platforms, tasks and users;
- inherent openness, where new "actors" come and go, and it cannot be anticipated at the design time, how many or how diverse tasks, resources and/or users may "join" the infrastructure at various points of time in the future;
- great need for fault-tolerance, reliability, and availability in the likely presence of individual "compute node" and/or communication link failures;
- concerns about data privacy, confidentiality and integrity, as well as access control to resources;
- preparedness for and resilience to cyber-attacks of known and unknown varieties, from familiar and unfamiliar "bad guys" trying to disrupt the infrastructure, compromise or steal data of individual users, deny legitimate users' access to data or compute resources, etc.

Reflecting upon the above list, it is immediate that today's and tomorrow's IoT has virtually all of the above characteristics – if anything, the "features" listed above are even more pronounced in the IoT context than what was the case with agent-oriented and service-oriented software architectures of the 1990s and early 2000s. This is why, in our opinion, revisiting and especially identifying how to adapt or extend those methodologies and programming abstractions based on *Service-Oriented Architecture* (SOA) and *Agent-Oriented Programming* (AOP) paradigms will be of utmost importance for IoT, specifically for the IoT software architecture and middleware design, in the years to come. Due to space constraints, we discuss here only one such programming abstraction – the *Actor model* for open distributed computing.

The actor model was introduced with a particular goal of specifying distributed computation in large-scale, open distributed systems [8, 9]. An actor encapsulates its internal state, as well as its behavior, which includes both data on which to compute, and procedures or methods to be applied to that data. Each actor has a unique name and a "mail box" to which other actors can send messages; actors communicate with each other via asynchronous message passing. In OOP, an object has a state and a set of procedures or methods that can change that state. An actor, additionally, encapsulates a thread of control thereby enabling *concurrency* (based on asynchronous exchange of messages among actors). A new actor can be created as desired, and various actors might be entering or leaving a given actor system, that is, this "soup" of actors, at various times.

All these characteristics of the actor model make it very versatile and suitable for open distributed systems. We note the actor model was originally introduced in the 1980s, with the main goal of extending the OOP to capture multi-threading and concurrency as the first-order entities. Actor model was

later expanded to provide a simple yet versatile communication and concurrency programming model for Distributed AI [9]. Many years later, we find the general actor-based computing paradigm quite appealing for the middleware as well as "application software" (that would run on top of that middleware) for the next-generation IoT, precisely because the model was designed with large-scale, highly heterogeneous and open (in which new users and devices come and go as they please) distributed systems in mind – that is, systems like the IoT.

In summary, the actor model has many wonderful properties from the standpoint of middleware and software design for the Internet-of-Things. What actors do not have, however, are mechanisms explicitly addressing the (actor data and/or state) integrity, privacy, confidentiality and related aspects; likewise, the actor model does not provide for self-healing defensive mechanisms in the presence of cyber-attacks. An open distributed software system based on actors will be fault tolerant with respect to individual actors joining or leaving the system, becoming too slow or unresponsive, or dying. However, the original actor model can do little about more Byzantine types of "bad behavior"; for example, dealing with malicious actors in the system. Therefore, should the future software infrastructures for IoT be based on some variation of the actor model, the cyber-security, self-monitoring and self-healing capabilities for such software infrastructure would need to be provided as "extras"? An interesting potential direction for future research, therefore, is how to extend the actor model so that certain "core" cyber-security aspects too are captures in the extended model as "first order entities". An alternative approach is, to design an actor-based architecture which would include "non-user", special-purpose actors ensuring data and system integrity, user privacy and confidentiality, providing monitoring and defensive mechanisms in case of cyber-attacks, and so on.

3 Some Desiderata for the IoA Middleware

The rise of middleware technologies and underpinning R&D was prompted by the evolution in computing and a paradigm shift from tightly coupled multiprocessor systems towards distributed, loosely coupled systems in which each node is usually a standalone computer, and all these "nodes" are interconnected in a local or wide-area or other network, communicating and cooperating with each other on various data and/or compute/CPU intensive tasks [10]. Once deployments of "clustered" systems involving "nodes" that run different software including different underlying Operating Systems (OS) became commonplace (in the 1990s), an additional layer of abstraction between the applications and the underlying hardware became necessary – a layer enabling different compute nodes in the system, especially when running on different operating systems, being able to communicate, coordinate, share tasks and resources with each other, etc.

However, the traditional clustered distributed systems were not characterized by the levels of openness and unpredictability characterizing IoT: addition of new nodes (computers or other interconnected devices) to the system, for example,

was in general infrequent and predictable. In stark contrast, in IoT context, new devices or even platforms (with potentially many "attached" devices) can join or leave the existing "soup" of devices, users and software agents at any time, with arbitrary frequency, and in unpredictable time patterns. Hence, designing a highly flexible, portable, adaptable and capable of self-reconfiguration middleware layer for the IoT kind of distributed systems is much more challenging, than for the more predictable and structured (and less open) traditional distributed clustered systems.

Recent years have seen an upsurge in research and prototype design on middleware for IoT. Some prominent examples of R&D efforts include [5,11–14]. One frequently encountered theme in this pursuit of the "right" kind of middleware for IoT, is the SOA design as the guiding principle: abstracting users and devices into "smart objects" and web services, and then providing a semantics-rich layer on top of that abstraction, that in turn enables those smart objects and services to exchange data, knowledge, tasks and resources with each other. Given the heterogeneity of devices found in IoT, the need for this kind of abstraction providing a uniform interfacing framework of interaction appears quite natural. What is tricky here, of course, is finding the right kind(s) of abstraction and interfacing, meeting the needs not only of the existing devices and applications but also predicting new types of devices and especially applications that will likely "want to join" IoT in the upcoming years. Here's a not-too-far-fetched example: consider the agent-based software system controlling a self-driven car being able to interact with devices/agents controlling various ambient and other aspects pertaining to the house, garage, driveway to the garage, are there any (humans) at home at the time of car's arrival, and so on. What mere 10-15 years back would likely sound like science-fiction in this realm, will actually become commodity technology over the next 10-15 years – in some cases, likely sooner than that.

Some researchers have realized the utmost importance of dealing with security, privacy, confidentiality and trust issues in the IoT context. In [14], the key idea is to integrate the (human) user directly into the "trust chain", ensuring visibility and transparency in the IoT security and reliability properties. [14] also wants to provide manufacturers of devices and platforms with a simple, unified interface of conveying the security, reliability and trust considerations to the users in simple, non-technical language – so that those users can reason about the tradeoffs or potential issues, and make decisions accordingly, without being cyber-security experts. The downside of this approach is fairly obvious: while human users are indeed given more direct, explicit control, the cognitive burden on the human is increased. In reality, most people want to go about their daily lives and have their various devices and technology doing the work for them, without having to worry much about how or why something may malfunction, and how to fix it (or even how to go about preventing bad things from happening via direct human intervention).

The Butler prototype [15] wants to allow users to manage their individual profiles by allowing data replication and identity control (incl. access control)

across different applications "belonging" to the same user. The objectives behind this are two-fold: to give the users sense of control, and to provide a middleware framework capable of integrating various user data (e.g., location, interests, current activity) in the overarching privacy and security protocols. From the standpoint of those designing such privacy and security protocols, this is potentially a good news: highly individualized, contextual information about users and their behaviors may provide information to those protocols helping them detect anomalous or suspicious behavior, trigger cyber-protection mechanisms as appropriate, and similar. However, again, the approach proposed in [15] places the additional burden on the users, majority of whom can be safely assumed not to be particularly "cyber-savvy". Finding the right tradeoff between full or near-full automation and computational intelligence that humans can trust on one hand, and still leaving human in control but without cognitive or physical "overload", will likely remain a major research and technology development challenge in this area for many years to come.

4 Designing Self-* IoT

A very open, very heterogeneous and very complex decentralized, many-user system such as IoT can be expected to be inherently difficult to control, formally reason about, or be able to predict the behavior of. Additionally, such complex open distributed cyber-physical systems are hard to protect from various types of attacks from the outside, and especially from malicious behaviors by "intruders" or "bad apples" from the inside. Recent Distributed-Denial-of-Service (DDOS) attacks on the IoT infrastructure (in the fall of 2016) exposed some of the vulnerabilities of the present-day IoT, as well as demonstrated just how disruptive and harmful a cyber-attack of even a relatively modest sophistication can be [16, 17].

While educating the users is very important, relying IoT's security on human users of smart phones, smart ambient sensors and other devices to always do the "right thing" won't suffice: IoT technologies are meant to enter every household, and their user base will keep growing, including children, the elderly and adults with little or no IT or "cyber-anything" training or understanding. Further, technologies supposed to be invisible "helping hand" with daily life yet that put too big a cognitive or other burden on users, are likely to either not be used correctly or even jettisoned altogether. Hence, IoT needs to be made cyber-secure without excessive reliance on human users of devices, applications and services going an extra mile in changing password on a myriad of devices, monitoring and validating appropriate access or other settings, and so on. Rather, cyber-security of IoT devices, services and platforms should be built-in and made user-friendly, low-maintenance and as little cumbersome to end-users as possible.

Further, while the traditional cyber-security measures applied to the wireless communication technologies as well as software infrastructure and middleware will continue being necessary, crucial components of overall IoT's cyber-security, these measures and methodologies won't suffice, either. Given the intrinsically

open and unpredictable nature of IoT, it will be practically impossible to predict all types of future users, devices and applications – and consequently, all possible kinds of malicious or other undesirable behaviors – at the design time (or even, for that matter, at the time of initial deployment of e.g. an infrastructure in a 'smart' household or office environment). Hence, the IoT systems need to be enabled with adaptation, self-monitoring, self-recovery and self-healing. For those desiderata to be achieved, AI, machine learning and computational intelligence will have to have key roles.

Some particular aspects in which machine learning, data/pattern mining and computational intelligence techniques are needed to enable the self-* capabilities of IoT, include (but are not limited to) the following:

- Collecting, storing and analyzing behavioral patterns from a variety of devices, services, applications and users (including which devices and applications "belong" to each user);
- Identifying anomalous behavior patterns, and either recommending to the human user(s), or in some case automatically/autonomously performing corrective actions;
- Pattern mining incl. inferring and predicting new types of malicious or abnormal behavior, incl. patters not seen before;
- Being able (re)-act similarly to the immune system of a biological organism with respect to identified intruders or other causes of trouble, without necessarily having an a priori knowledge in what shapes or forms intruders may show;
- Ability to reason and meta-reason about devices, users, and behaviors;
- Reinforcement Learning and Meta-Learning over the good policies, behaviors, corrective actions to "fix problems" in the past, etc. – in order to improve (over time) at how are various cyber-threats and cyber-attacks dealt with; and more.

Securing various types of wireless communications involved in IoT, protecting the integrity, privacy and confidentiality of users' data, applying a variety of existing digital forensics, information assurance and related techniques all already have important roles in cyber-security of IoT. Moreover, the quality and "intelligence" of solutions in those areas will only grow in importance in the years to come, as IoT technologies and solutions mature and become more pervasive. Our main take-away, however, is that the "traditional" cyber-security methodologies and solutions won't suffice for an intrinsically open and highly complex cyber-physical and cyber-secure system such as IoT. Further, we argue that the role of AI, computational intelligence and data analytics (broadly defined) will be absolutely crucial to make IoT more secure, reliable and trustworthy than it currently is, and to keep it as secure and reliable as possible in the years and decades to come.

5 Summary and Concluding Thoughts

We review some of the key challenges in a prominent next-generation cyber-physical/cyber-secure system, the (NG) Internet-of-Things. IoT offers a promise of making many aspects of our daily lives easier, more comfortable and enjoyable, and less encumbered with menial tasks and worries – but it also poses a number of technology and R&D challenges. Among those challenges, we have focused on three: what are the suitable programming abstractions and models for the software infrastructure for IoT/IoA, what are the fundamental challenges and design "wish list" when it comes to the design of the IoT middleware, and lastly, how can AI, machine learning and computational intelligence enable IoT to become a self-healing, self-correcting autonomous system capable of protecting itself from cyber-threats of both known and unknown varieties.

We do not claim to have the solutions for the issues we have identified in this paper; rather, our purpose is to highlight some of the key challenges, and offer a partial roadmap of what is needed to address those challenges in order to fulfill the promise of IoT, so that indeed it becomes a scalable, inter-operable, flexible, reliable and trustworthy cyber-secure ambient intelligence system that helps us improve how we work, live, play and interact with each other. The pervasiveness of IoT technologies and integrated solutions is bound to steadily keep growing in the years to come. By delivering on these promises, the next-generation IoT would improve quality of a variety of aspects of people's daily lives arguably at a scale not seen since the world-wide adoption of the Internet itself.

References

1. S. Sicari, et al. "Security, privacy and trust in Internet of Things: the road ahead", *Computer Networks*, vol. 76, pp. 146-164, 2015
2. S. Sicari, et al. "A secure and quality-aware prototypical architecture for the IoT", *Information Systems*, vol. 58, pp. 43-55, Elsevier, 2016
3. 1. P. Tosic. "Reputation-based Distributed Coordination for Heterogeneous Autonomous Agents", Proc. Internet of Agents (IoA-2016), Web Intelligence Workshops (WIW-2016), Omaha, Nebraska, 2016
4. P. Tosic, Y. Wu. "Towards Networks of Search Engines and Other Digital Experts: A Distributed Intelligence Approach", Proc. 8th Int'l Conf. u- & e-Service, Science & Technology (UNESST-15), pp. 35-38, 2015
5. iCORE Project, http://www.iot-icore.edu
6. Y. Shoham. "Agent-oriented programming", *Artificial Intelligence*, vol. 60, 1993
7. N. R. Jennings. "An agent-based approach for building complex software systems", *Communications of the ACM*, vol. 44 (4), 2001
8. G. Agha. "Concurrent Object-Oriented Programming", *Communications of the ACM*, vol. 33 (9), 1990
9. G. Agha, N. Jamali. "Concurrent Programming for DAI", in G. Weiss (ed.), "Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence", The MIT Press, Cambridge, Massachusetts, 1999
10. P. Tosic. "Understanding Autonomous Agents: A Cybernetics and Systems Science Perspective", Proc. IEEE/SAI Future Technologies Conference (FTC-2016), San Francisco, California, 2016

11. A. Rizzardi, et al. “Networked smart objects: moving data processing closer to the source”, Proc. 2nd EAI Int’l Conference on IoT as a Service, 2015
12. IoT-EST project, http://ict-iotest.eu/iotest
13. Ebbits project, http://www.ebbits-project.eu/
14. uTRUST: Usable Trust in the Internet of Things project, www.utrustit.eu/
15. The BUTLER Project, www.iot-butler.eu
16. S. Cobb, “10 things to know about the Oct. 21 [2016] IoT DDoS attacks”, October 2016 https://www.welivesecurity.com/2016/10/24/10-things-know-october-21-iot-ddos-attacks/
17. L. H. Newman. “What We Know About Friday’s Massive East Coast Internet Outage”, *Wired* magazine, Security section, October 2016 https://www.wired.com/2016/10/internet-outage-ddos-dns-dyn/

Multiobjective Automated Argumentation Among Internet of Things

Henry Hexmoor and Kane Rodriguez

Computer Science Department
Southern Illinois University
Carbondale, Il 62901, USA
{Hexmoor;KRodriguez}@cs.siu.edu

Abstract. Internet of Things must rely on their agent counterparts, termed recently as *Internet of Agents* (IoA), in order to achieve competing objectives. We have considered production of sets of arguments that support different objectives at the same time. This will lead to myriad forms of contradictions. We present an account of automated argumentation system that facilitate contradictions.

Keywords: Automated Argumentation · Online Organizations · Multiagent Systems

1 Introduction

Cyber-physical systems (CPS) are algorithmically controlled mechanisms involving networked devices [1]. Whereas physical components of CPS (e.g., robots sand devices) are tangible, embodied, and occupy physical space, cyber components are largely disembodied, intangible, and location-independent. As such, Internet of Things (IoT) are a subset of CPS. In sharp contrast to the passive view for entities of things as objects, agent inhabitants of IoT are active and may take action proactively. In this perspective, things are enlivened with agent overlays that take advantage of smart sensors and provide intended decision-making capacities for things.

Numerous suggestions posit that things in physical proximity form social ties creating collaboration networks. Minimally, things provide profiles that include goods and services relevant to other things. We are focusing on interactions among people and things that lead to creation of trust, delegation, role arbitration, and thus collaboration. Nodes may perceive a level of social capital as experience prior and expected future beneficial interactions. Much needs to be developed to exploit the spectrum of sociality. We have used crowd evacuation as an illustrating case study and an exemplar for other scenarios. In the burgeoning era of cyber physical systems, it is essential that embedded IoT nodes work together on matters of common interest and using automated argumentation reach agreements or at least commonly identify the strongest position on consequential topics. For instance, a driverless car must determine the best course of action

when confronted with unavoidable collision [2]. Legal considerations as well as ethical resolutions will remain outside current proposed work. However, future explorations may bring them into our focus. We define machine to machine social argumentation as negotiation that include but goes beyond argumentation when individuals are in a socially connected network as in [3]. Settings where humans and things form collaborative teams are fascinating but remain outside our current scope. Instead, we target machine to machine argumentation as in the case of vehicle to vehicle networks. There have been attempts to form autonomous robotic ad hoc coalitions; e.g., [4]. Similarly, IoT nodes that monitor health status of occupants in a building must agree on the safest building location for people to congregate. This is crucial for all types of disaster from weather concerns to the active shooter incidents.

Argumentation is the process in which agents exchange and evaluate interacting and inevitably conflicting arguments. It is a form of dialog during which beliefs, understanding and opinions are presented, explained, compared, and defended. The arguments are the basis for inferences, negotiations, conflict resolution, and conclusions drawn by logical reasoning. Argumentation is one of the oldest research foci and one of the most enduring ones in Artificial Intelligence [5, 6] and in parallel in Philosophy, first reported in [7] and most recently in [8]. Automated Argumentation has been adapted to many domains including computational law and multi-agent negotiations [9]. The most vigorous and prolific argumentation research has been conducted with Argugrid[1], which is a grid based research consortium funded by the European Union and directed by Dr. Francesca Toni of Imperial College in London, United Kingdom. Whereas social abstract argumentation [3] facilitate online argumentation among human social media participants as can be found on Facebook, we aim to facilitate social argumentation chiefly among machines. Numerous suggestions posit that things in physical proximity form social links creating social networks. Minimally, things provide profiles that include goods and services relevant to other things [10]. For effective interaction with human peers and other animals, things need to be equipped with biological sensors (i.e., biosensors) so that their corresponding agents would ascertain conditions of their bio-organism cohabitants. For example, graphene nano-sensors are available for passive sensing of bacteria. Other typical passive biosensor exemplars are motion and vibration sensors, thermometers, audio and visual sensors, touch and tensile sensors, barometric pressure sensors, and a variety of chemical sensors. By fusing sensory information, a thing may determine bio-organism presence including humans at given radii from it. Agents controlling things can use biosensors as proximity sensors and behave in socially meaningful ways. Once agents inhabiting things perceive bio-presence, they may perceive and initiate as well as expect reciprocal sociality. Reciprocally, humans may perceive electro-mechanical things by sensing energy and wireless networking measures.

In the context of smart IoT devices, the first task is identification of arguments generated by their corresponding agent. Each agent is designed to receive

[1] www.argugrid.eu

sensory data and perform problem solving that produces an output, which might be a mere perception or an action to perform.

The problem solving module shown in Figure 1 is an expert system that encapsulates agent problem solving. Agents will fuse one or more sensory data for determining an input for reasoning. The expert system will include design and a model current applicable conditions. A periodically generated argument is a pair of sensed data and an output encapsulated as an atomic abstract argument that will be cast to compete with other arguments in the system argument pool.

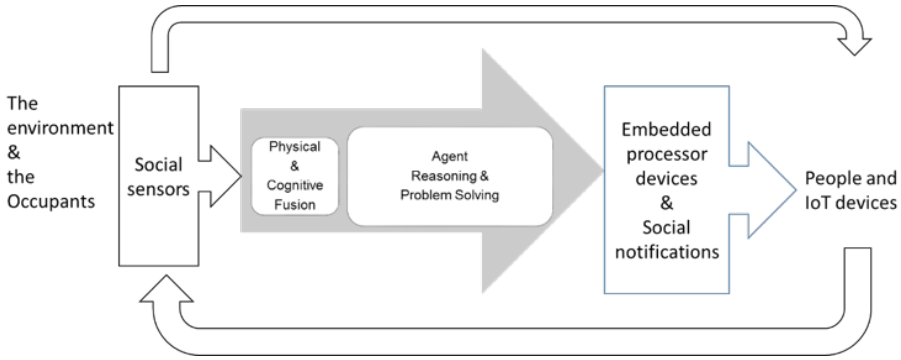


Fig. 1. An agent corresponding to a smart IoT device

Consider the following three arguments with the structure of <sensed condition, then warrant; therefore, recommended action>. This argument further illustrated in Figure 2 and the components of Toulmin model are shown in Figure 2. The warrant portion is simplified here for illustration purposes only. In the proposed work, warrants will be produced as a result of using a model that formulates and propagates danger/safety levels for each room based on modeled dynamics of a moving assailant posing danger to occupants [11].

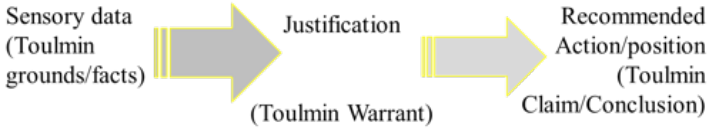


Fig. 2. Illustration of an argument components

2 An Expert System for Vehicular Lane Selection

Using a Toulmin style form of argument formation, we have developed an expert system (ES) for lane selection among smart vehicles that embody two main components: (a) an inference engine, and (b) an inter-agent argumentation resolution component.

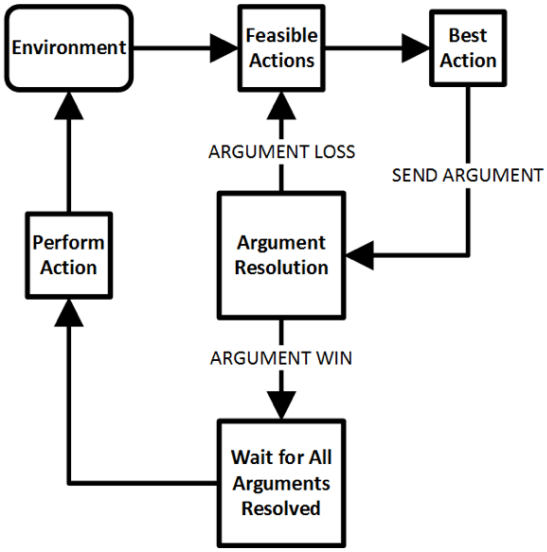


Fig. 3. The Life Cycle of Argument Formation, Conflict Resolution, and Action Enactment

As shown in Figure 3, the inference engines use context taken from the environment, as well as from argument losses, and formulates all possible feasible actions. Upon feasible action derivation, the best action in accordance with the ES objective is determined. The argument for this respective action encapsulates the projected position that the vehicle expects to occupy as a result of their chosen action and also encapsulates the vehicle itself. The argument by each agent is consolidated into an argument pool where the overseeing system will identify conflicting arguments that are projecting two vehicles to enter the same global expressway position. The overseeing system will gather votes from each vehicle corresponding to their acceptance or rejection of each argument within the pool. After the social support for each argument is known, conflicts between arguments are resolved, with each argument with the most support within a conflicted arguments subset of the entire argument pool being given permission to perform their next action. Upon approval and modification of all argument actions such that there are no longer any conflicts between arguments, each agent will then perform their chosen action.

Once the inference engine has determined all feasible actions, rule sets pertaining to each of the objectives then determine the best action for the said objective. In our current testbed, these ES objectives may be in one of three priority modes: (1) global emission, (2) local lane congestion, or (3) personal travel time. The congestion-based objective is concerned with reducing the congestion of the vehicles current lane, the travel time objective is concerned with attaining and exceeding the vehicles preferred speed, and the emission objective is inclined to reduce the global emission levels caused by all vehicles on the highway. The objective that a system is in directly affects the formulated argument. The following three arguments exhibit possible conclusions and actions pertaining to each objective.

- *Travel Time Objective*:
  a_1 = Since my lane has a max speed limit below my adjusted preferred speed and the lane above is feasible; then I want a faster lane and can move up; therefore, I should move up one lane.
- *Emission Objective*:
  a_2 = Since my lane has a minimum speed limit above my emission adjusted preferred speed and the lane below is feasible; then I am comfortable with moving to a slower lane and I can move down; therefore, I should move down one lane.
- *Congestion Objective*:
  a_3 = Since my lane has a high relative congestion and the lane above me is feasible, within my adjusted preferred speed range, and has a low relative congestion; then moving up a lane will benefit my local lane congestion and satisfy my adjusted preferred speed requirement; therefore, I should move up one lane.

Vehicles who determine a need for lane change generate corresponding arguments. Naturally, multiple vehicles attempting to change lane change into the same lane at the same time will be physically conflicting corresponding to attacks between respective arguments. We are working on several conflict resolution strategies including a social voting mechanism.

3 Conclusion

We have been addressing the need for codifying interaction among agents that represent nodes of a group of cyber-enabled systems with automated argumentation that extends abstract, Dung style argumentation. We illustrated the model of abstract arguments to components of a Toulmin style argument and explained how expert systems can be used to produce arguments. Arguments are grouped for competition by the overarching objectives used by peer agents that generate them. We have considered smart vehicles who may consider specific objectives that lead them to prefer specific lanes. This process creates conflicts between vehicles attempting lane change. Arguments that represent intentions for lane change are pitted against one another in argument conflicts.

For simplicity, we considered conflicting vehicles to share their underlying objective that govern their pattern of driving. A natural extension for further research is to consider argument conflicts that arise from agents possessing heterogeneous objectives. Our methodology heralds a step toward automated negotiation beyond automated argumentation that has been the focus of present work. Many other cyber physical environments embody stages for opposing positions that may benefit from automated argumentation as a tool for collaboration.

References

1. G. Fortino, A. Guerrieri, W., Russo, C., Savaglio., 2013. ?Integration of Agent-based and Cloud Computing for the Smart Objects-oriented IoT?,, In Proceedings of the 2014 IEEE 18th International Conference on Computer Supported Cooperative Work in Design, IEEE.
2. N. Goodall, 2016. Can You Program Ethics Into a Self-Driving Car? IEEE Spectrum.
3. J. Leite and J. Martins, 2011. Social Abstract Argumentation, In Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, 2011.
4. H. Yu, Z. Shen, C. Leung, 2013. From Internet of Things to Internet of Agents, In IEEE International Conference on and IEEE Cyber, Physical, and Social Computing, PP., 1054 - 1057 , IEEE PRESS.
5. T.J.M. Bench-Capon and P.E. Dunne, Argumentation in artificial intelligence, Artificial Intelligence, vol. 171(10-15), 2007, pp. 619-641.
6. I. Rahwan and G.R. Simari, Argumentation in Artificial Intelligence, Springer, eds. 2009.
7. S. Toulman, Uses of Arguments, Cambridge University Press, 1958.
8. J. Pollock, Defeasible reasoning and degrees of justification, in Argument & Computation, Taylor and Francis, 2010.
9. S. Fatima, M. J. Wooldridge and N. R. Jennings, An agenda based framework for multi-issues negotiation, Artificial Intelligence Journal, 152(1), 2004, pp. 1-45.
10. P. Hinds, T. Roberts, H. Jones, 2004. Whose Job Is It Anyway? A Study of Human/Robot Interaction in a Collaborative Task, Journal of Human-Computer Interaction, Volume 19, pp. 151-181, Lawrence Erlbaum Associates pub.
11. E. Lovellette and H. Hexmoor, Abstract Argumentation Using Voronoi Diagrams, IEEE Collaborative Systems and Technologies.

Author Index