PH.D. IN ELECTRONIC AND COMPUTER ENGINEERING
Dept. of Electrical and Electronic Engineering
University of Cagliari

Managing the Internet of Things based on its Social Structure

Michele Nitti

Advisor: Prof. Luigi Atzori

Curriculum: ING-INF/03 Telecomunicazioni

XXVI Cycle
2012/2013

To my family

Alla mia famiglia

Acknowledgments

My apologies to those who are not able to understand Italian, but the following acknowledgements are written in my mother tongue, in order for me to be able to express myself better.

Prima di tutto desidero ringraziare il mio tutor, Luigi, per aver fatto di me il ricercatore che sono, per avermi guidato in questi anni, per i continui nuovi stimoli e suggerimenti, per avermi dato la possibilità di confrontarmi con realtà diverse, e con sfide di cui ancora aspettiamo con ansia i risultati.

Un ringraziamento particolare va a tutto lo staff dell'MCLab, per le occasioni di confronto e di discussione (specialmente per quelle accese, in sala riunioni ;)), ma soprattutto per le pause caffè e i pranzi "domenicali".

Grazie a Nino, Lucia e Francesca, che mi hanno accolto a bracce aperte tra loro e mi hanno sempre voluto bene, regalandomi una seconda famiglia.

Non potrò mai ringraziare abbastanza i miei genitori, Nicola e Rita, non solo perché senza di loro non sarei qua, ma perché sono sempre stati i miei più grandi sostenitori. Grazie per avermi dato la possibilità di fare e diventare tutto ciò che desideravo, per non avermi mai fatto mancare il vostro appoggio e concesso sempre libera scelta; mi avete insegnato che la vita è fatta di sogni, e che è importante combattere per poterli raggiungere.

Grazie a mio fratello Marco, per le risate che riesci a farmi fare anche nei momenti più difficili. Sei l'amico su cui sono sempre sicuro di poter contare e che non mi delude mai.

Grazie a Valentina, la mia insostituibile compagna, per essere sempre presente, per condividere con me le gioie e le delusioni, per la tua "enorme" pazienza, per tutti i momenti in cui mi hai tirato su il morale. Se oggi sono come sono, una buona dose di colpe la hai pure tu.

Ultimi ma non ultimi, grazie a tutti gli amici: quelli che non vedo spesso quanto vorrei, quelli che sono partiti alla ricerca di una terra un po' più fortunata, e quelli delle pause pranzo. Grazie per ogni risata insieme.

Contents

Acronyms

API	Application Programming Interface .
C-LOR	Co-Location Object Relationship .
C-WOR	Co-Work Object Relationship .
CPS	Cyber Physical System .
CSV	Comma-Separated Values .
DARPA	Defense Advanced Research Projects Agency .
DHT	Distributed Hash Table .
EM	Embodied Microblogging .
EPC	Electronic Product Code .
GPS	Global Positioning System .
HTTP	HyperText Transfer Protocol .
INS	Inertial Navigation System .
IoT	Internet of Things .
IP	Internet Protocol .
IT	Information Technology .
JSON	JavaScript Object Notation .
NFC	Near Field Communications .
NIC	National Intelligence Council .
OC	Owner Control .
OOR	Ownership Object Relationship .
OWL-S	Ontology Web Language for Services .
P2P	Peer to Peer .
POR	Parental Object Relationship .
PTO	Pre-Trusted Object .
QoS	Quality of Service .
REST	REpresentational State Transfer .
RFID	Radio-Frequency IDentification .
RM	Relationship Management .
SA-REST	Semantic Annotations for Representational State Transfer .
SAWSDL	Semantic Annotations for WSDL .
SC	Service Composition .
SD	Service Discovery .

SIoT	Social Internet of Things .
SN	Social Network .
SOAP	Simple Object Access Protocol .
SOF	Social Organization Framework .
SOR	Social Object Relationship .
SWIM	Small World In Motion .
TM	Trustworthiness Management .
TVM/DTC	Dynamic Trust Computation of the Trust Value Measure .
Ucode	Ubiquitous code .
UMTS	Universal Mobile Telecommunications System .
UPC	Universal Product Code .
URI	Uniform Resource Identifier .
USDL	Unified Service Description Language .
UWB	Ultra-wide Band .
WDSL	Web Services Description Language .
WS	Web Service .
WSML	Web Service Modelling Language .
WSMO	Web Service Modelling Ontology .
WSN	Wireless Sensor Network .
XML	eXtensible Markup Language .

Introduction

"If small groups are included in the decision-making process, then they should be allowed to make decisions. If an organization sets up teams and then uses them for purely advisory purposes, it loses the true advantage that a team has: namely, collective wisdom." [1]

Society is moving towards an "always connected" paradigm, where the Internet user is shifting from persons to things, leading to the so called *Internet of Things* (IoT) scenario. The IoT vision integrates a large number of technologies and foresees to embody a variety of smart objects around us (such as sensors, actuators, smartphones, RFID, etc.) that, through unique addressing schemes and standard communication protocols, are able to interact with each others and cooperate with their neighbors to reach common goals [2, 3]. IoT is a hot research topic, as demonstrated by the increasing attention and the large worldwide investments devoted to it.

It is believed that the IoT will be composed of trillions of elements interacting in an extremely heterogeneous way in terms of requirements, behavior and capabilities; according to [4], by 2015 the RIFD devices alone will reach hundreds of billions. Unquestionably, the IoT will pervade every aspect of our world and will have a huge impact in our everyday life: indeed, as stated by the US *National Intelligence Council* (NIC) [5], "by 2025 Internet nodes may reside in everyday things – food packages, furniture, paper documents, and more".

Then, communications will not only involve persons but also things thus bringing about the IoT environment in which objects will have virtual counterparts on the Internet. Such virtual entities will produce and consume services, collaborate toward common goals and should be integrated with all the other services.

One of the biggest challenges that the research community is facing right now is to be able to organize such an ocean of devices so that the discovery of objects and services is performed efficiently and in a scalable way. Recently, several attempts have been made to apply concepts of social networking to the IoT.

There are scientific evidences that a large number of individuals tied in a social network can provide far more accurate answers to complex problems than a single individual (or a small group of – even knowledgeable – individuals) [1]. The exploitation of such a principle, applied to smart objects, has been widely investigated in Internet-related researches. Indeed, several schemes have been proposed that use social networks to search Internet resources, to route traffic, or to select effective policies for content distribution.

The idea that the convergence of the “Internet of Things” and the “Social Networks” worlds, which up to now were mostly kept separate by both scientific and industrial communities, is possible or even advisable is gaining momentum very quickly. This is due to the growing awareness that a *“Social Internet of Things”* (SIoT) paradigm carries with it many desirable implications in a future world populated by objects permeating the everyday life of human beings.

Therefore, the goal of this thesis is to define a possible architecture for the SIoT, which includes the functionalities required to integrate things into a social network, and the needed strategies to help things to create their relationships in such a way that the resulting social network is navigable. Moreover, it focuses on the trustworthiness management, so that interaction among objects that are friends can be done in a more reliable way and proposes a possible implementation of a SIoT network.

Since this thesis covers several aspects of the Social internet of Things, I will present the state of the art related to the specific research activities at the beginning of every Chapter. The rest of the thesis is structured as follows.

In Chapter 1, I identify appropriate policies for the establishment and the management of social relationships between objects, describe a possible architecture for the IoT that includes the functionalities required to integrate things into a social network and analyze the characteristics of the SIoT network structure by means of simulations.

Chapter 2 addresses the problem of the objects to manage a large number of friends, by analyzing possible strategies to drive the objects to select the appropriate links for the benefit of overall network navigability and to speed up the search of the services.

In Chapter 3, I focus on the problem of understanding how the information provided by members of the social IoT has to be processed so as to build a reliable system on the basis of the behavior of the objects and define two models for trustworthiness management starting from the solutions proposed for P2P and social networks.

Chapter 4 presents an implementation of a SIoT platform and its major functionalities: how to register a new social object to the platform, how the system manages the creation of new relationships, and how the devices create groups of members with similar characteristics.

Finally, in Chapter 5, conclusions will be drawn regarding the effectiveness of the proposed algorithms, and some possible future works will be sketched.

Related papers

- **Michele Nitti**, Luigi Atzori, Irena Pletikosa Cvijikj, “Network Navigability in the Social Internet of Things”, IEEE World Forum on Internet of Things (WF-IoT), 2014, to appear.

- Roberto Girau, **Michele Nitti**, Luigi Atzori, “Implementation of an Experimental Platform for the Social Internet of Things”, Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), pp.500-505, 3-5 July 2013.

- **Michele Nitti**, Roberto Girau, Luigi Atzori, “Trustworthiness Management in the Social Internet of Things”, IEEE Transactions on Knowledge and Data Engineering, vol. PP, no. 99, pp. 1.

- **Michele Nitti**, Roberto Girau, Luigi Atzori, Antonio Iera, Giacomo Morabito, “A subjective model for trustworthiness evaluation in the social Internet of Things”, IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), pp.18-23, 9-12 Sept. 2012.

- Luigi Atzori, Antonio Iera, Giacomo Morabito, **Michele Nitti**, “The Social Internet of Things (SIoT) When social networks meet the Internet of Things: Concept, architecture and network characterization”, Computer Networks, vol. 56, no. 16, pp 3594-3608, 14 November 2012.

Chapter 1

Concept, architecture and network characterization

Recently there has been quite a number of independent research activities that investigated the potentialities of integrating social networking concepts into Internet of Things (IoT) solutions. The resulting paradigm, named SIoT, has the potential to support novel applications and networking services for the IoT in more effective and efficient ways.

In fact, applying the social networking principles to the IoT can lead to several advantages:

- the SIoT structure can be shaped as required to guarantee the network navigability, so as that the discovery of objects and services is performed effectively and the scalability is guaranteed like in the human social networks;
- a level of trustworthiness can be established for leveraging the degree of interaction among things that are *friends*;
- models designed to study the social networks can be re-used to address IoT related issues (intrinsically related to extensive networks of interconnected objects).

Even if the idea of a Social Internet of Things (SIoT) has been already discussed, as I will explain in Section 1.1, in this paper I go beyond the state of the art in several ways:

- I identify appropriate policies for the establishment and management of social relationships between objects in such a way that the resulting social network is navigable;
- I describe a possible architecture for the IoT, which includes the functionality required to integrate *things* into a social network;
- I study the characteristics of the SIoT network structure. To this purpose I present some interesting results obtained through the SWIM mobility simulator [6].

A further major contribution of this paper is the provision of an overview (the first one, to the best of my knowledge) of the research activities aimed at the integration of social networks and the Internet of Things.

The Chapter is organized as follows. In Section 1.1 I provide a survey of the research activities that focus on the integration of social networking concepts into the Internet of Things. In Section 1.2 I present the basic idea of the SIoT and propose a classification of the types of social relationships that can be established between objects. In Section 1.3 I describe a proposal for a SIoT architecture and relevant functionalities, while in Section 1.4 I illustrate some examples of how the SIoT can be exploited. The structure of the resulting SIoT network is the subject of the analysis reported in Section 1.5.

1.1 State of the art

The collective intelligence emerging in social networks is an extremely interesting phenomenon that has been described in fascinating ways (e.g., [1]). This has been pointed out by many researchers – quoting the words by the *Defense Advanced Research Projects Agency* (DARPA) Director Regina Dugan during her keynote speech at IEEE Globecom 2010– as the key factor of a "*new era of wonder for science*". In fact, the incredible success of social networking websites, such as Twitter and Facebook, and the availability of data about the structure and dynamics of social networks collected through these websites, have attracted the attention of a large number of scientists from several areas [7]. In the context of communication and networking, for example, schemes have been proposed that exploit the similarity in the interests of friends – the so called *homophily* – to enhance the Internet search [8] or to optimize the peer-to-peer networks [9]. Also, schemes have been proposed that use social relationships to establish higher levels of trust and, thus, improve the efficiency and effectiveness of security solutions (e.g., [10] and [11]). In the mobile computing domain the intuition that individuals who are connected by social relationships are likely to meet more often than people who do not have any connection has been exploited. As a consequence, schemes have been proposed, which base the policy for data diffusion in opportunistic networks on the above property (see [12] or [13], for example).

A first idea of socialization between objects has been introduced by Holmquist et al. [14]. In that paper, the focus was on solutions that enable smart wireless devices, mostly wireless sensors, to establish temporary relationships. The authors also analyze how the owners of the sensor nodes should control such a process. However, that work is dated 2001 and both the concepts of the IoT and the online social networks were in their infancy.

More recent literature reports several researches and experimental applications based on a new generation of objects. These enter into humans' daily activities with a new attitude and a greater awareness of the fact that they are designed as "smart" objects with a potential for interaction with each other previously inconceivable.

In [15] the "things" connected to the Internet are clearly distinguished from the "things" participating within the Internet of social networks, which are named with the neologism Blogject, that is, "objects that blog". The theoretical concept of *Embodied Microblogging* (EM), introduced in [16], also challenges the current vision of IoTs. Rather than focusing on thing-to-thing or human-to-thing interactions, it proposes two novel roles that the augmented everyday objects will play: (i) mediate the human-to-human communication and (ii) support additional ways for making noticeable and noticing activities in everyday life. Also the authors of [17] show how to empower physical objects to share pictures, comments, and sensor data via social networks. They also discuss about the implications of the so called "socio-technical networks" in the context of the IoT.

Last but not least, the work in [18] introduces the idea of objects able to participate in conversations that were previously reserved to humans only. Those envisioned are objects aware of dynamic community structures; thus, they are able to develop a spontaneous networking infrastructure based on the information to be disseminated other than the information on the objects themselves.

Recently, the idea that the IoT and the social networks are two worlds not really that far apart from each other as one might think, has begun to appear in the literature. It is the case of the papers [19] and [20], for example. More specifically, in [19] the authors envision the future of the Internet as being characterized by what they name Ubiquitous IoT architecture, which resembles the *Social Organization Framework* (SOF) model. That work provides an insightful overview of the expected IoT network structure. However, it does not aim at exploiting the characteristics of the social networks into the IoT. Analogously, the research activities reported in [20] consider that, being things involved into the network together with people, the social networks can be built based on the Internet of Things and are meaningful to investigate the relations and evolution of the objects in the loT. Finally, the convergence of IoT and social networks has been considered in [21]. In that work, an individual can share the services offered by her/his smart objects with either her/his friends or their things. Accordingly, in [21] the reference social network is a social network of humans and it is utilized by things as an infrastructure for service advertisement, discovery, and access. That remarkable contribution somehow violates the IoT vision in which the objects should interact spontaneously to offer value-added services to humans.

An important step in the direction of the SIoT has been accomplished in [17]. There, the implications of the integration between the IoT and the social networks have been investigated and a few interesting exemplary applications are described. That paper, however, does not describe how the social relationships should be established by objects and does not propose any solution regarding the required architecture and protocols.

Finally, in a recent paper [22] social attributes, which reflect the social relations of nodes, have been analyzed. There a sort of quantification of the social relationships among mobile nodes is also performed by means of parameters such as an interaction factor and a distance factor. Besides, the authors study the behavior of mobile nodes by applying the typical theory of the social networks. In [22], however, it is assumed that there is a one–to–one correspondence between persons and objects. On the contrary, in the IoT, several objects can be carried by the same person while a large part of the objects will remain either static or embedded in the environment.

As a logic consequence of the studies described above, recently the name *Social Internet of Things* began to appear in official documents and published papers. This happens in form of either simple statements of objectives to be achieved within the activities of Strategic Research Agendas [23], or interesting attempts to explore the social potentialities of the Internet of Things building blocks [24].

1.2 A Social Internet of Things

The cited literature, however, still lacks in some basic aspects which should be addressed to fully achieve an actual “social networks of intelligent objects”. In fact, in analogy with the social networks of human beings, a SIoT network still needs: (i) the definition of a notion of social relationship among objects, (ii) the design of a reference architectural model implementing a social Internet of Things based on the codified inter-object relationships, (iii) the analysis of the social network structure, which derives from the objects interactions based on the defined social relationships. Only a thorough investigation of these three issues will allow for effectively extending the use of models designed to study social networks of humans [25] to social networks of things.

The definition of the novel paradigm of SIoT and the initial studies on the relevant social structures have been the focus of an initial investigations in [26]. In that paper, an embryonic idea of architecture has been suggested, by starting from an appropriate revision of those utilized by the major existing social networking websites [27].

Although in this Chapter I aim at addressing the items (ii) and (iii), a brief review of the introduced concepts relevant to the potential social links among objects is given in the rest of this section. This will ease the comprehension of the concepts introduced later and make the illustrated research self-consistent.

One can start from the idea that, in the future, things will be associated to the services they can deliver. Thus, within a given social network of objects, a key objective will be to publish information/services, find them, and discover novel resources to better implement the services also through an environmental awareness. This can be achieved by navigating a social network of "friend" objects instead of relying on typical Internet discovery tools that cannot scale to the trillions of future devices.

The choice of the best basic set of social relationships can be made by observing sample typologies of application and the inter-object interactions that these foresee. The next step is to bring social behaviors of objects back to the widely accepted four elementary relational models of the Friske's theory [28] [29] summarized in Table 1.1.

Table 1.1: Basic relational frames

Relational Model	**Brief description**
Communal sharing	equivalence and collectivity membership emerge against any form of individual distinctiveness
Equality matching	egalitarian relationships characterized by in-kind reciprocity and balanced exchange
Authority ranking	asymmetrical, based on precedence, hierarchy, status, command, and deference
Market pricing	based on proportionality, with interactions organized with reference to a common scale of ratio values

I claim that these patterns of interaction among human beings are directly applicable to possible social behaviors of typical objects that implement pervasive applications. There is no doubt that many applications and services should, in the future, be associated with groups of objects whose individuality will be "sacrificed" to the overall interest of providing services to users (as it is the case, for example, of applications involving the use of swarm intelligence and swarm robotics). It is equally true that many applications will involve an interaction among objects that will be performed "au pair", i.e., where each object will be the bearer of its specific service to the community. In addition, several services are already available, which involve the use of multiple objects that establish asymmetric relations (as, for example, in services based on Bluetooth, Zigbee, 6LoWPAN networks of sensors/actuators or Radio-Frequency IDentification (RFID) identification systems). In other services, the objects condition their relationship of "friendship" to the achievement of mutual benefits (this is the case, for example, of cooperative services designed to reduce the energy consumption of wireless devices). Those described above are merely examples of services that will surely find a placement in the future social network of smart objects and that rely on the same cited relational structures that Fiske has theorized for human beings.

From the analysis of possible service and application typologies, built upon the envisaged Social Internet of Things (more details will be given in a following section), one can also derive some basic relationships onto which relationship profiles will be defined within the reference system architecture. The kinds of relationships I define are those here summarized:

- *"Parental Object Relationship"* (POR): established among objects belonging to the same production batch, i.e., usually homogeneous objects originated in the same period by the same manufacturer.
- *"Co-Location Object Relationship"* (C-LOR): established among objects (either homogeneous or heterogeneous) used always in the same place (as in the case of sensors, actuators, and augmented objects used in the same environment such as a smart home or a smart city). Observe that, in certain cases, such C-LORs are established between objects that are unlikely to cooperate with each other to achieve a common goal. Nevertheless, they are still useful to fill the network with "short" links.
- *"Co-Work Object Relationship"* (C-WOR): established whenever objects collaborate to provide a common IoT application (as in case of objects that come in touch to be used together and cooperate for applications such as emergency response, telemedicine, etc.).
- *"Ownership Object Relationship"* (OOR): established among heterogeneous objects which belong to the same user (mobile phones, music players, game consoles, etc.) .
- *"Social Object Relationship"* (SOR): established when objects come into contact, sporadically or continuously, because their owners come in touch with each other during their lives (e.g., devices and sensors belonging to friends, classmates, travel companions, colleagues).

Please note that the establishment and management of such relationships should occur *without human intervention*. This is not in contrast with a future vision of a "fully networked human". This latter is responsible only to set the rules of the objects social interactions and then enjoys the services resulting from such interactions. This is a clear paradigm shift from other proposals, according to which the objects/devices just participate in the human social network built by their owners.

By following an approach inspired by human “social relationships” and “relational models” things mimic the human behavior just to effectively interact with each other. A clear advantage lies in the fact that, in so doing, models and principles, which already proved to be effective for the study of the human social networks, can be extended to the object communities.

In Section 1.5 of this paper I analyze the network structure arising from the above types of social relationships through numerical examples.

1.3 The SIoT system

In this Section I provide an overview of a possible implementation of the SIoT. More specifically, in Section 1.3.1 the envisioned reference architectural model is described, in Section 1.3.2 the major functions required to run the SIoT are illustrated, and in Section 1.3.3 the advantages and disadvantages of the proposed architecture are analyzed.

1.3.1 The architecture

To describe the proposed system I resort on the simple three-layer architectural model for IoT presented in [30]. It consists of: (i) the sensing layer, which is devoted to the data acquisition and node collaboration in short-range and local networks; (ii) the network layer, which is aimed at transferring data across different networks; and (iii) the application layer, where the IoT applications are deployed together with the middleware functionalities.

Figure 1.1 shows the resulting three-layer architecture. The three basic elements of the proposed system are: the SIoT Server, the Gateway, and the Object.

SIoT Server

The SIoT Server does not encompass the sensing layer but only the Network and the Application Layers. The Application Layer consists of three sublayers. The Base Sublayer includes the database for the storage and the management of the data and the relevant descriptors. These record the social member profiles and their relationships, as well as the activities carried out by the objects in the real and virtual worlds. Data about humans (object owners as well as visitors) are also managed.

The relevant ontologies are stored in a separate database and are used to represent a semantic view of the social activities. Such a view is extracted through appropriate semantic engines. Indeed, ontology and semantic services are necessary to provide a machine interpretable framework for representing functional and non-functional attributes and operations of the IoT devices.

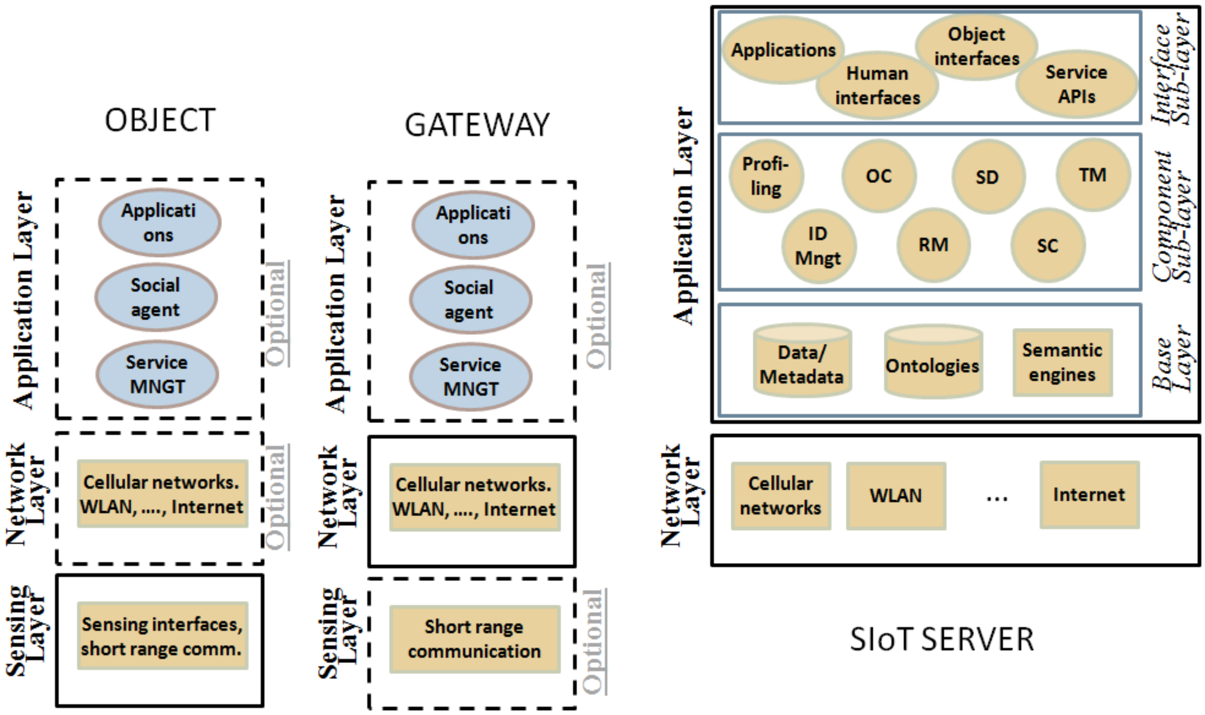


Figure 1.1: Proposed system architecture following the three-layer model made of the sensing, network, and application layer. The main SIoT components belongs to the application layer, wherein the Relationship Management (RM), Service Discovery (SD), Service Composition (SC), and Trustworthiness Management (TM) functionalities are located. The lines represent the optional layers in both the object and the gateway architecture.

In this context, several works have been already conducted, which could be a starting point for the definition of an ontology to be used in the SIoT system. One solution is to adopt the *Ontology Web Language for Services* (OWL-S) model that provides both rich expressive descriptions and well-defined semantics. This has already been used as the basis of a semantic service modeling framework for the IoT [31]. In this framework, services are used as an interface that represents the IoT resources (i.e. the physical world devices) and provide an access to the functions and capabilities of these resources. Also in [32], an ontology is considered as a fundamental attribute of the IoT with the role of supporting the agent (man or machine) who reads an electronic tag to understand the information in it. Ontologies to manage and control heterogeneous systems have been investigated in [33]. Here the authors foresee that without ontological classification and semantic annotation processes an automatic discovery will be impossible. In [34] the importance of the ontology has been analyzed from a social network perspective as a format to represent the object information which is relevant to end users.

There are several other approaches for creating semantic service descriptions [35], including: *Semantic Annotations for WSDL* (SAWSDL), *Unified Service Description Language* (USDL), *Web Service Modelling Language* (WSML), *Web Service Modelling Ontology* (WSMO), and *Semantic Annotations for Representational State Transfer* (SA-REST) [36]. All these models are good bases that can be exploited to describe the social objects in my model. In this context, it is also worth mentioning the Friend-of-a-Friend project (FOAF; www.foaf-project.org), which is aimed at creating a web of machine-readable pages describing people, the links between them, and the things they create. The results of this project are of particular interest for the description of the objects' social links.

With reference to Figure 1.1, the Component Sub-layer includes the tools that implement the core functionality of the SIoT system. The *ID management* is aimed at assigning an ID that universally identifies all the possible categories of objects. The *profiling* is aimed at configuring manually and automatically a (static or dynamic) information about the objects. The *Owner Control* (OC) is the module that enables the definition of the activities that can be performed by the object, the information that can be shared (and the set of objects which can access such information), as well as the type of relationships that can be set up. The *Relationship Management* (RM) is a key module in the network since the objects have not the intelligence of humans in selecting the friendships; thus, this intelligence needs to be incorporated into the SIoT. Main task of this component is to allow objects to start, update, and terminate their relationships with other objects (on the basis of the owner's control settings).

The *Service Discovery* (SD) is a fundamental component [37], which is aimed at finding which objects can provide the required service in the same way humans seek for friendships and for any information in the social networking services. The *Service Composition* (SC) component enables the interaction between objects. Most of the time, the interaction is related to an object that wishes either to retrieve an information about the real world or to find a specific service provided by another object. In fact, the main potential I see in deploying SIoT is its capability to foster such an information retrieval. Leveraging on the object relationships, the service discovery procedure finds the desired service, which is then activated by means of this component. Last but not least, the *Trustworthiness Management* (TM) component is aimed at understanding how the information provided by the other members shall be processed. Reliability is built on the basis of the behavior of the object and is strictly related to the relationship management module. Trustworthiness can be estimated by using notions well-known in the literature, such as centrality and prestige, which are crucial in the study of the social networks.

The third sub-layer, that is the Interface Sub-layer, is where the third-part interfaces to objects, humans, and services are located. This sub-layer may be mapped onto a single site, deployed in a federated way by different sites, or deployed in a cloud. Herein, I am not proposing any specific implementation solution.

Gateway and Objects

As to the Gateway and Objects systems, the combination of layers may vary mainly depending on the device characteristics. The following three scenarios can be foreseen. In a simple one, a dummy Object (e.g., either a RFID tag or a presence sensing device) that is equipped with a functionality of the lowest layer, is only enabled to send simple signals to another element (the Gateway). The Gateway is equipped with the whole set of functionalities of the three layers.

In another scenario, a device (e.g., a video camera) is able to sense the physical world information and to send the related data over an IP network. The object would then be set with the functionality of the Network Layer other than that of the Application one. Accordingly, there is no need for a Gateway with Application Layer functionality. An Application Layer in a server, somewhere in the Internet, with the gateway application layer functionality would be enough.

According to a third scenario, a smart object (e.g., a smartphone) may implement the functionality of the three layers so that the Gateway is not needed, but for some communication facilities targeted to maintain the Internet connectivity of the object. This is the case of a smartphone, which has enough computational power to perform all the three-layer operations and that may need a Gateway for ubiquitous network connectivity.

Whatever the scenario implemented, the Application Layer encompasses the SIoT applications, as well as the social agent and the service management agent, which are presented below. The social agent is devoted to the communication with the SIoT servers to update its profile, to update friendships, and to discover and request services from the social network. It also implements the methods to communicate directly with other objects when they are geographically close or when the service composition needs direct communications between objects. Finally, the service management agent is responsible for the interfaces with the humans that can control the behavior of the object when communicating within their social network.

1.3.2 Main SIoT processes

The main components of the proposed architecture are located in the Component Sub-layer. In fact, the SIoT is not intended as a solution for the sensing and networking in IoT, but to make the world of trillions of things manageable when facing the problem of service and information discovery. Additionally, it aims at laying the ground for autonomous interactions among objects (mainly through service discovery and composition) for the benefit of the human user.

In order to describe the interactions among the SIoT architectural elements, in Figure 1.2 I provide an overview of the processes related to four main SIoT activities, namely: entrance

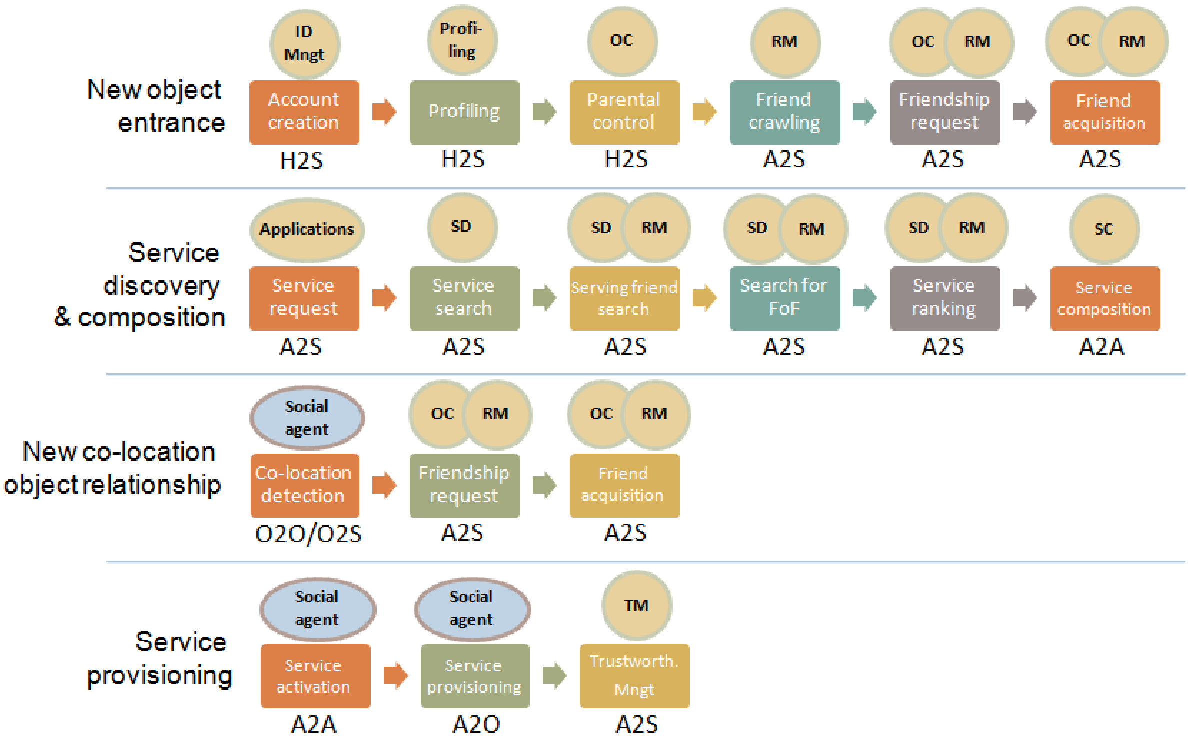


Figure 1.2: Processes related to four main SIoT activities: new object entrance; service discovery and composition; new object relationship; service provisioning

of a new object, service discovery and composition, new object relationship establishment, and service provisioning. In the figure, the square blocks represent the tasks involved in the analyzed activity (e.g., account creation, profiling, parental control). These have a label associated *i*2*j* that identifies the two elements that communicate to carry out the task (i, j = H, S, A, O, which stand for human, SIoT server, object agent, and object, respectively). Notice that herein the Gateway is not mentioned, even if it may take part in these processes when the agent is involved. This is because, in this context, the agent is defined as the software entity that implements the application functionalities of either the Object or the Gateway. On top of the task blocks I cite the main architectural components (see Figure 1.1) involved to carry out the relevant operations.

For what concerns the **entrance** of a new object into the system, the relevant activities are mostly carried out by the object owner, who communicates with the servers to create the account, insert the object profile data, and set the control parameters though the ID management and object profiling components. The ID scheme should be interoperable with the main identification schemes already in use in this area, such as: IPv6 addresses, *Universal Product Code* (UPC), *Electronic Product Code* (EPC), *Ubiquitous code* (Ucode), *OpenID*, *Uniform Resource Identifier* (URI). The profiling adds relevant information about the capabilities and history of the object to its relevant ID. Given the heterogeneity of the IoT nodes, SIoT members are organized in classes. Each class is defined on the basis of the main object features.

- *Class1* is assigned to mobile objects with large computational and communication capabilities. Examples of objects belonging to this category include smartphones, tablets, and vehicle control units.
- *Class2* is assigned to static objects with significant computational and communication capabilities; to this class belong object such as: displays, set top boxes, smart video cameras.
- *Class3* is assigned to objects with sensing capabilities only, that is objects capable of providing a measure of the environment status.
- *Class4* is assigned to the RFID- or Near Field Communications (NFC)-tagged objects.

Each class is then characterized by specific attributes, such as: *object category*, which further specifies the object typology within its class; *owner ID*; *object position*, which can be changing over the time depending on the object mobility features[1]; *power supply status*, that defines whether the object is either battery-powered (and the battery power level is provided), socket-connected (and whether is currently connected or not), or harvests power from the environment; *amount of traffic* generated in terms of number of connections and overall bit-rate.

Once the object profiling procedure is completed, then the agent (which may be running either on the object itself or in a separate system, depending on the object characteristics) completes the process by looking for friends in the SIoT servers. During this phase the object establishes the main relationships that are triggered by its profile as well. Such relationships include parental object and ownership object relationships. Other relationships are established afterwards, during the entire lifetime of the objects, and mainly depend on the objects' movements and service/information exchanges over the SIoT.

Service discovery and composition are triggered by the application running either on the SIoT servers or in close relationship with the agent (in the gateway or in the object). The application can be one of those mentioned in Section 1.4. The way in which this process is performed depends on the type of service that the application is looking for. Examples are: the provisioning of information about the surrounding environment, the status of an object and the activities carried out by the object owner, as well as the activation of a specific action from another object.

Once the service request has been triggered by the application, the process continues with one of the most innovative and crucial task of the designed system, that is, the **serving friend**

[1] Position of an object should be given in absolute terms and can be estimated directly (by the objects itself) or indirectly (it is provided to the object by some other elements in the system, which can estimate its own position). Obviously, in the latter case, the estimation of the position is not very accurate. However, note that accurate positioning is not required by the SIoT operations.

search. This is related to the procedure of looking at the "friend" object's profiles to see whether the required service is provided by one or more of them. In this process, the different types of relationships have not the same relevance to any application. For instance, if the requested service is of a "best practice sharing" type, then the parental relationship is the most important. In fact, the same problem has probably been addressed in the past by objects belonging to the same production batch. In the case of need of an information about the surrounding environment, co-location and co-work relationships are those that should be exploited. In fact, the corresponding friends are those that most probably had occasion to acquire an information on the surroundings. In case a friend able to provide the service has not been found, then, the graph of friendships is also crawled. Since more than a single service may be found, a ranking is required. The ranking can be executed according to different rules, among which: the serving object trustworthiness, the credit/debit relationship of the interacting objects, the object resources (in terms of residual battery power, bandwidth, communication and computing reliability, and so on). Several approaches can be adopted in this context to rank the potential service providers, as those described in [38].

When the described activity ends, the service composition is triggered, which consists of interfacing the requesting service with the required one. Note that the service composition involves A2A communications, while A2S communications are required during other tasks.

In the **new object relationship activity**, two objects become aware that they are neighbors for a period of time long enough to trigger friendship. In this scenario I am referring to the co-location, co-work, and social relationships, which are triggered in case of geographical proximity of the objects. The detection of this event is enabled by the use of short range communication facilities (e.g., NFC, Bluetooth, or ZigBee interfaces) that allow two objects to detect that they are within communication range of each other. There are other possibilities to detect this event. One is the use of the localization facilities (WiFi/Bluetooth triangulation, *Inertial Navigation Systems* (INSs), or even *Global Positioning System* (GPS)) already available within the objects to track their position over the time. During an upload of location information into its profile (in the SIoT server), an object can detect the co-location relationship with other objects. Whatever the way the object detects the co-location event, the object agent then requests the friendship, which may be accepted according to the owner control rules.

The **service provisioning process** consists in delivering the service previously discovered and composed with the requesting service. As an example, let us consider the scenario of a smartphone that is looking for information about radio signal coverage in the areas surrounding its current position (this is the fourth example discussed in the next section). To accomplish its target, the smartphone drives a service discovery and composition process to look for smartphones and personal computers that have already visited the areas of interest (and are then aware of the signal strength). Once the service has been composed, the requesting agent (installed in the smartphone itself) communicates with the agents providing the relevant information to activate the service. All the way through, the service requesting agent is able to extract important information about the trustworthiness of objects that provide the services. This information is uploaded to the SIoT server and thus is available to the whole community.

1.3.3 Analysis of the proposed SIoT system

The IoT domain is characterized by a significant fragmentation and by the presence of heterogeneous systems based on dissimilar architectures. This makes a synergistic integration process difficult to be carried out. The need for a clear reference architectural model that will allow the different systems to cooperate is, thus, strongly felt. As such a model is still missing, I tried to adhere to the following principles: define an architecture that could foster the interoperability with existing IoT components, protocols, interfaces, and functionalities; include mechanisms for the efficient integration of this architecture into the service layer of the Future Internet networking infrastructure.

The resulting architecture is expected to provide the following advantages:

- A separate layer devoted to the sensing of the physical world allows for an easy integration of the existing and widespread standards for short distance communication technologies, such as: RFID, *Ultra-wide Band* (UWB), NFC, and *Wireless Sensor Network* (WSN).
- A layer devoted to the data transport functionalities allows for interconnecting the separate networks involved in the IoT. In this context, I follow the successful structure of the current Internet architecture centered around the *Internet Protocol* (IP), which can be seen as the narrow waist between connected devices on the one side and applications and services on the other.
- The service discovery module is surely one of the key functionalities in the IoT arena and is present in most of the proposed IoT architectures [39]. It addresses the issue of handling queries that contain a semantic information through a kind of declarative language. In this way, pointers refer to objects that can provide the related service (i.e., provide the information that satisfies the initial query). In my architecture, I have defined a separate module devoted to this functionality to foster interoperability with external systems. Therefore, queries that require the access to entities that are not part of the SIoT, can be sent to external architectures where the equivalent module can

handle the process. Obviously, the discovery will rely on different principles with respect to the social-oriented approach I propose.

- The service composition module enables mashup interaction models (e.g., browsing, linking, bookmarking), which is a pillar of the Internet of Services, to be extended to the real-world. This fosters an open ecosystem of digitally augmented objects on top of which applications can be created (i.e., it promotes the integration of the Internet of Service with the Internet of Things).
- As proposed in most of the other relevant architectures (e.g., [21]), the gateway is a key component to allow objects with limited communication and/or computing capabilities to take part in the IoT. This is another strong point I wanted to keep in my architecture.

On the other hand in the proposed solution I identify the following two potential weaknesses:

- The relationship management functionality in my solution is implemented only into the Server, without a collaboration of the Gateway and the Objects. I decided for this solution to allow “non SIoT-enabled” devices and relevant gateways to take part in the SIoT without the need for updating their systems. This approach, however, has the disadvantage of requiring a continuous communication with the servers for the creation and the update of the relationships. In particular, it is necessary to send information about the activity of the owner monitored by the objects (e.g., position, use of the Internet connection, movements), so that the co-location, co-work, and social relationships can be detected.
- The discovery process in my approach is driven by the relationship links among objects, which are followed to find the target service providers. Once these are found, their trustworthiness levels are evaluated to select the most reliable ones. The amount of interaction between the service discovery and the relationship management modules is limited; for this reason, I have decided to keep these two modules separate. However, if the discovery is performed by following the links involving trusted nodes only (i.e., there is a need for navigation across “trusted” areas), then the number of interactions between the two above components may increase significantly. This may reduce the efficiency in resource discovery.

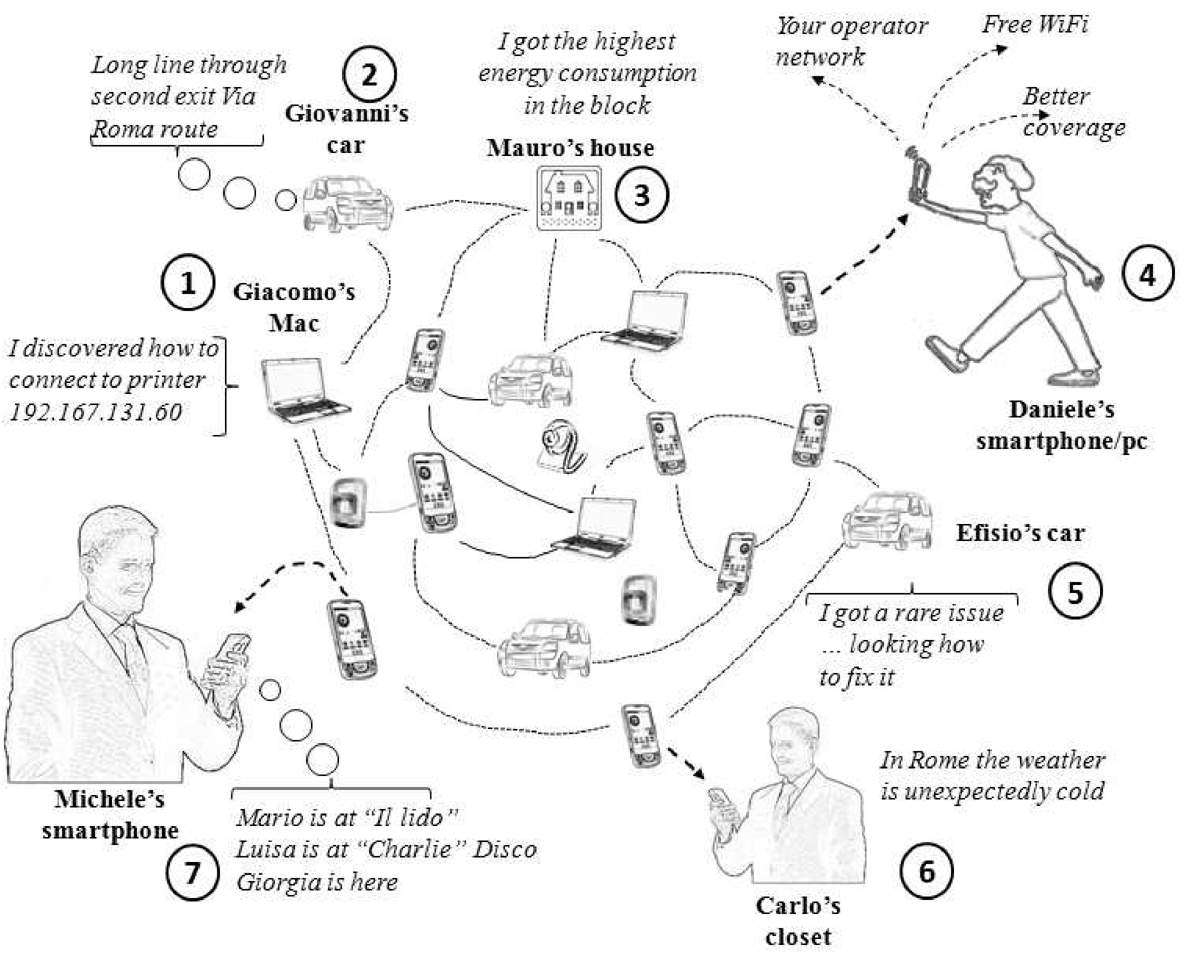

Long line through second exit Via Roma route
2
Giovanni's car
I got the highest energy consumption in the block
Mauro's house
3
Your operator network
Free WiFi
Better coverage
1
Giacomo's Mac
I discovered how to connect to printer 192.167.131.60
4
Daniele's smartphone/pc
Efisio's car
5
I got a rare issue ... looking how to fix it
Michele's smartphone
7
Mario is at "Il lido"
Luisa is at "Charlie" Disco
Giorgia is here
In Rome the weather is unexpectedly cold
6
Carlo's closet

Figure 1.3: Sketch of the sample applications in the Internet of social things.

1.4 Sample applications

Several applications can benefit from the availability of social relationships between things interconnected to a network composed of trillions of nodes. While a few interesting applications can be already defined, many others will show up in the years to come according to the increase in the number and categories of objects able to connect to the Internet. Figure 1.3 provides a sketch of some sample applications, which are described in the following:

1. Giacomo has just bought a new notebook. It is a Mac this time, because of the influence of the Mac closed-community of colleagues at his work premises. At the beginning, this new world is a jumble for him because of the difficulties in connecting to some network equipments (e.g., printers, faxes, and smartcard readers) that indeed appear to him not to be exactly Mac-friendly. By exploiting social relationships with other Mac computers in the same local area network, Giacomo's Mac can find a mate that has already addressed the same configuration issues and fix the problems. Looking for potential sources

of information through its social network (and exchanging best practices) is quite straightforward. In fact, features such as geographical location, class of object, brand, and typology, allow for identifying the right friends in the community. Note that Apple's Bonjour already provides the functions required for Apple devices to discover other Apple devices in the same network and share resources with them. However, Bonjour is a service/resource discovery solution which cannot satisfy the need described above. In other words, you can print using a printer shared by another Mac device, but you cannot use that printer if the latter Mac device is not active.

2. Luigi is a sales representative that frequently moves by car around the city to meet his customers. Unfortunately, the traffic has increased during the last year; this making his tour more and more problematic. However, by exploiting the social network, his car is able to gather information in advance about traffic congestion along possible routes and to choose the best path to get to the meeting in the scheduled time. Finding the right source of information in the IoT social network is easy for the car by contacting "friend" devices acquired by means of co-location relationships. By this I mean those cars with which Luigi's car shared some routes in the past but which belong to drivers that Luigi might not even know.
3. Antonio has bought a new house in a block recently built according to advanced eco-friendly principles. Each flat is equipped with controllers and sensors able to manage and measure energy consumption and production (photovoltaic and solar cells) during the whole day. By means of their IoT social network, the domestic controllers are able to exchange information on the energy usage with reference to: consumption and production of energy to perform local benchmarking, identification of the energy providers that best match the house needs (in terms of cost profiles), identification of the household appliances with the highest efficiency levels. A light in any house changes the color according to the energy saving level obtained by its owner, which differs from other houses in the block. Ownership and co-location relationships are exploited in this scenario.
4. Daniele is a frequent traveler for work and needs the network to connect to his colleagues, customers, and family. His smartphone is a member of the IoT social network and is able to get information about the places in his surroundings that are covered by a stronger signal, by less congested *Universal Mobile Telecommunications System* (UMTS) cells, by its operator base stations (for

free data service), and by free WiFi. Again, the right "friend" devices in this scenario are found by looking for specific member categories (smartphone and PCs) and geographical locations.

5. During the last year, Sergio's new car is having frequent problems, which seem to be difficult to fix. On the basis of the information collected by the junction box through the sensors located in the car, a profile of the problem and of the car is built. This profile is then exchanged within the IoT social network to look for similar problems that have already been addressed by other cars in the network (it is again a best practice exchange scenario). The search for the node that may help in fixing the issue is driven by the problem profile and mainly exploits parental relationships among cars.
6. There are several sensors that are going to be installed more and more numerous in any environment. These provide information about the status of environments in terms of temperature, crowdedness of the ambient (rooms, theaters, discos, and others), identity of the people, humidity level, and other parameters about the weather. All these objects may exchange friendship with the controller of Laura's closet. She is preparing for her next travel and automatically acquires the list of clothes to be used for a comfortable travel.

Some of the above applications are of interest for a large part of the social network members (I refer to these as *popular* applications), while others are of interest for only a restricted part of the objects community (I refer to these as *niche* applications). Additionally, the applications are classified into those that have a geographical relevance, that is, mainly involve the objects that are located in a specific area (*local*), and those that don't have a geographical relevance (*universal*).

In Table 1.2 I provide a classification of the mentioned applications. Example 1 is local and of the niche category since the interest is limited to the LAN area and to only a restricted subset of the computer categories. Example 2 mainly involves the vehicles in a geographical area, thus it is local but is of interest for the entire car category and can be considered universal. Example 3 is clearly local and belongs to the niche category. In Example 4, even if the exchanged information refers to a specific location, the scenario is that of a smartphone that travels all around the world so that it is not geographically restricted; and it is popular since it involves entirely the smartphone and the computer categories. Example 5 is surely the case of an application of interest for a restricted group of SIoT members (the cars of a specific category) and universal since there is no connection with the geographical location of the cars. For the same reasons, Example 6 is universal and it can be considered as belonging to the niche category.

Table 1.2: Characteristics of the sample applications of Figure 1.3 in terms of popularity and geographical extension.

	Local	**Universal**
Niche	1: Giacomo's Mac 3: Antonio's house	5: Sergio's car 6: Laura's closet
Popular	2: Luigi's car	4: Daniele's smartphone/pc

The popularity is for sure an important aspect to consider when developing an application, since the expected popularity can justify the high costs for designing and building it. However, in my vision, the social component sub-layer should be considered as a middleware that can be used by any application, both popular and of niche. The geographical connotation is, instead, important to understand whether the co-location and co-work relationships are important or not.

1.5 SIoT network characterization

In order to characterize the SIoT network, I will study the probability distributions of the geographical distance between nodes that are connected with each other as well as the probability distribution of the length of the shortest path between a pair of nodes randomly selected.

In order to estimate such a probability distribution, I would need the mobility data traces of a large number of objects. Unfortunately, such a data is not available to date. Therefore, I have exploited the *Small World In Motion* (SWIM) simulator ([40], [6]), which is able to capture the impact of social behavior in the mobility of humans. Logically, I have modified it to focus on the mobility of things rather than on the mobility of their owners.

Accordingly, this section is organized as follows. Firstly, I briefly describe the SWIM mobility model and the modifications introduced (Section 1.5.1); secondly, I analyze the numerical results obtained (Section 1.5.2).

1.5.1 Simulation environment

To produce mobility traces of objects I have used the mobility model called SWIM as a starting point [40], [6]. My choice is motivated by the ability of SWIM to take the impact of social behaviors on the movement of human beings into account. In fact, it has been proven that an appropriate tuning of the parameters of the SWIM mobility model allows to obtain accurate matching between the output of the model and the most popular mobility traces available in CRAWDAD [41].

Table 1.3: Configuration Parameters

Users	2000
User perception radius	0.0067
Simulation time	11 days
α	0.8

The basic intuition under the construction of the SWIM mobility model is that humans choose their destinations depending on the distance from their *home* and the popularity of such a destinations. In other words, if I assume that the area of my interest is divided into smaller areas called *cells* and that each user u is assigned a home $h(u)$, then in the selection of her next destination a given user u will assign a *weight* $w(C)$ to cell C equal to

$$w(C) = \alpha \cdot \text{distance}(h(u), C) + (1 - \alpha) \cdot \text{seen}(C) \tag{1.1}$$

where $distance(h(u), C)$ is a function of the distance between the home of user u and the cell C and decays as this distance increases, while $seen(C)$ keeps the popularity of the cell C into account. Indeed, this represents the number of users that have been observed by u the last time she visited cell C. In this context, I assume that at any time a user can *see* all the users within a certain distance, which I call *user perception radius*.

The parameter α is in the range [0;1] and is used to determine whether the users prefer to visit popular sites rather than nearby ones. Once the destination is selected, the user moves in a straight line towards it and with a constant speed proportional to the distance to travel.

I have chosen the parameter setting that matches the Cambridge scenario [41] . More specifically, I consider a scenario characterized by the parameters reported in Table 1.3, in which I have assumed that the area of interest is a unitary square.

However, the output of the original SWIM is a trace of the position of humans. In this paper, instead, I am interested in the mobility of things. Accordingly, I have extended the SWIM simulator as follows.

I assume that each user possesses a set of things connected to the SIoT. The number of owned things is selected randomly according to a normal distribution with average equal to 10 (such as: one or more smartphones, a tv, a personal computer, a car, a digital camera, a digital frame, one o more sensors at home, and a number of RFID objects). Furthermore, I assume that at any time the user carries a certain number of objects, that vary according to a normal distribution with average equal to the objects she possesses and leaves the others at home.

In this way, it is possible to simulate the movements of all the objects in the SIoT and post-process them.

1.5.2 Numerical results

In the following I show and analyze the numerical results obtained as explained in the previous section. More specifically, I study the characteristics of the random variable $X^{(A)}$. This is defined as the random variable representing the distance between two nodes that are tied by a social relationship of type A (in my case $A \in \{$POR, C-LOR, OOR, SOR, C-WOR$\}$). I am interested in the probability density function of $X^{(A)}$.

POR and C-LOR

Parental Object Relationships (POR) are independent (in the scales of interest) from the specific positions of nodes. In fact, in most cases objects tied by POR are distributed uniformly in the area of interest. Accordingly, I do not focus on the distribution of $X^{(POR)}$. Here, I only stress that POR can be utilized to build long links in the SIoT.

The distribution of $X^{(C-LOR)}$ is obvious as well. In fact, in this case, a link exists between two objects only if their distance is very small. Accordingly,

$$f_{X^{C-LOR}}(x) \approx \delta(x) \tag{1.2}$$

Observe that relationships of C-LOR type can be utilized by the applications to explore the environment surrounding a given object and, therefore, are extremely important in the context of smart environment applications.

OOR

In Figure 1.4 I represent the probability density function in case of Ownership Object Relationship, that is, I show $f_{X^{(OOR)}}(x)$ versus the value of the distance x. In the same figure, I also show the probability density function of the exponential and Gamma distributions that have average value and variance equal to those of $X^{(OOR)}$. In the figure it is evident that the exponential distribution does not provide an accurate approximation of $X^{(OOR)}$.

To better assess the accuracy of the approximation provided by the Gamma distribution, I need to clean the measured $f_{X^{(OOR)}}(x)$ and to this purpose I filter it. Specifically, I define the operator $\Phi(f)$ which can be applied to any sequence f of values and that returns another sequence $\{\Phi(f)\}$ such that its i-th value is the sum of the first i values in the sequence f, that is

$$\{\Phi(f)\}_i = \sum_{j=1}^{i} \{f\}_j \tag{1.3}$$

In Figure 1.5 I show $\{\Phi(f_{X^{(OOR)}}(x))\}$ and $\{\Phi(f_\Gamma(x)\}$ where $f_\Gamma(x)$ represents the Gamma

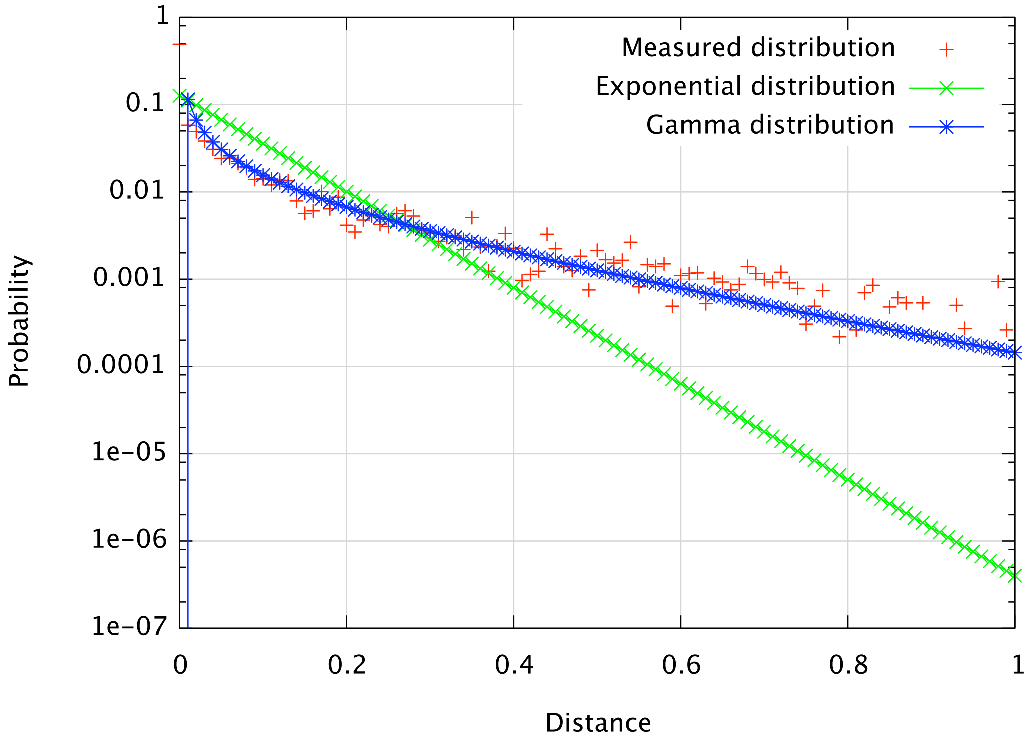


Figure 1.4: Probability density function of the variable $X^{(OOR)}$. In the figure I show the pdf of the exponential and Gamma distributions with the same average value and variance.

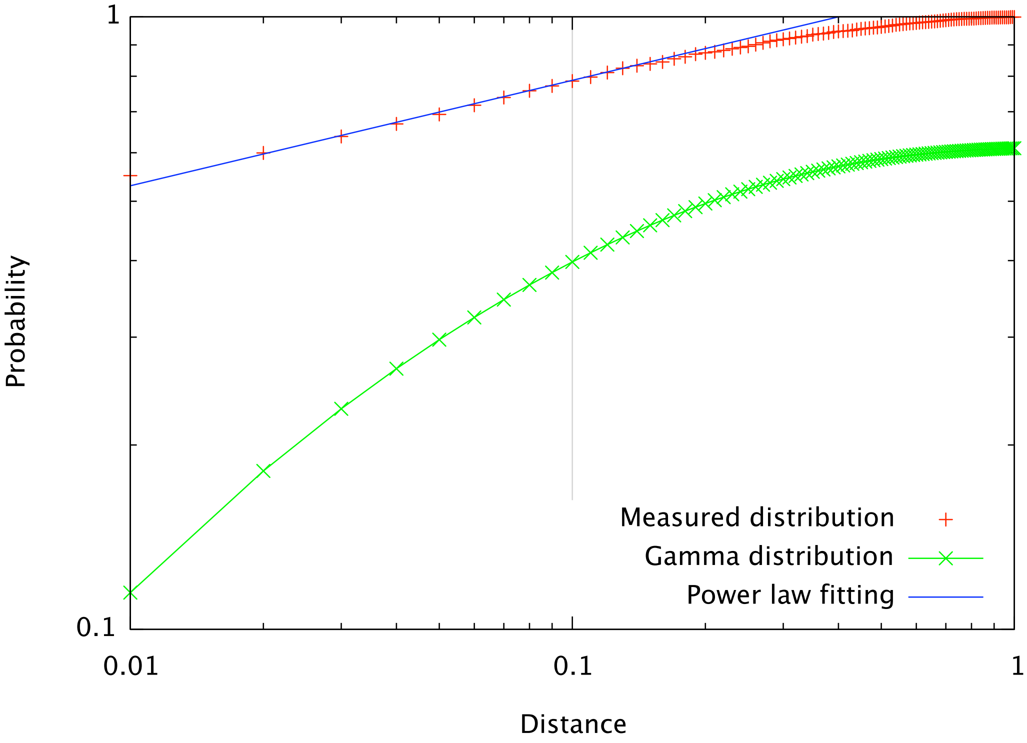


Figure 1.5: Values of $\{\Phi(f_{X^{(OOR)}}(x)\}$ and filtered pdf of the Gamma distributions with the same average value and variance.

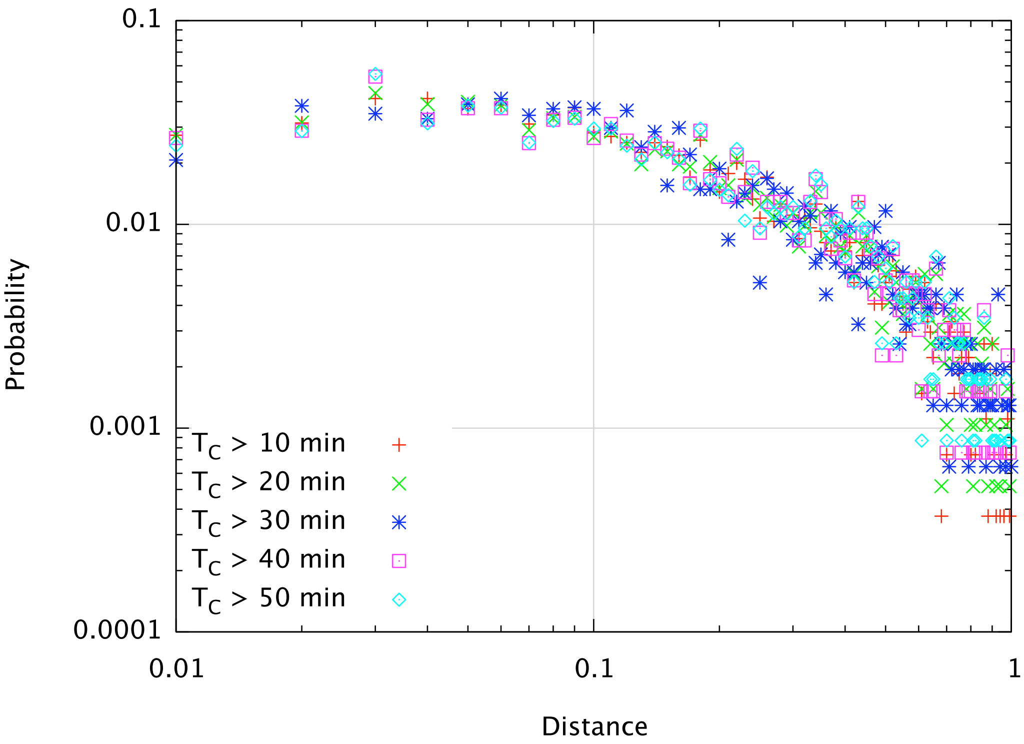


Figure 1.6: Probability density functions, $f_{X^{(SOR)}}(x)$, obtained for different values of T_C.

distribution that approximates $f_{X^{(OOR)}}(x)$. In Figure 1.5 it is evident that the Gamma distribution does not provide an accurate approximation of $f_{X^{(OOR)}}(x)$. In the same figure it is also evident that $\{\Phi(f_{X^{(OOR)}}(x))\}$ has a linear behavior when represented in log-log scale, which means that $f_{X^{(OOR)}}(x)$ is power-law. To demonstrate this, in Figure 1.5 I show the line which approximates $\{\Phi(f_X^{(OOR)}(x))\}$ in the log-log scale. In other words it is possible to approximate

$$f_{X^{(OOR)}}(x) \propto x^{\beta_{OOR}} \tag{1.4}$$

For example, in the case discussed above, I have that β_{OOR} is equal to -0.827.

SOR

In Figure 1.6 I represent the probability density functions, $f_{X^{(SOR)}}(x)$, of the distance between nodes connected by Social Object Relationships for different values of the parameter T_C. I have assumed that a relationship of the SOR type is established between objects if their owners meet at least N_C times, if successive meetings occur at intervals of duration longer than T_I, and if each of the meetings lasts longer than T_C. More specifically, in Figure 1.6 I assume that $N_C = 2$ and $T_I = 8$ hours. In order to "clean" the figure, I represent the values of $\{\Phi(f_{X^{(SOR)}}(x))\}$ in Figure 1.7. In the same figure I show the filtered pdf of the exponential distribution and the power law distribution which approximate $f_{X^{(SOR)}}(x)$. By observing the figure, one notices that the probability density function of $X^{(SOR)}$ is not significantly impacted by the specific value of

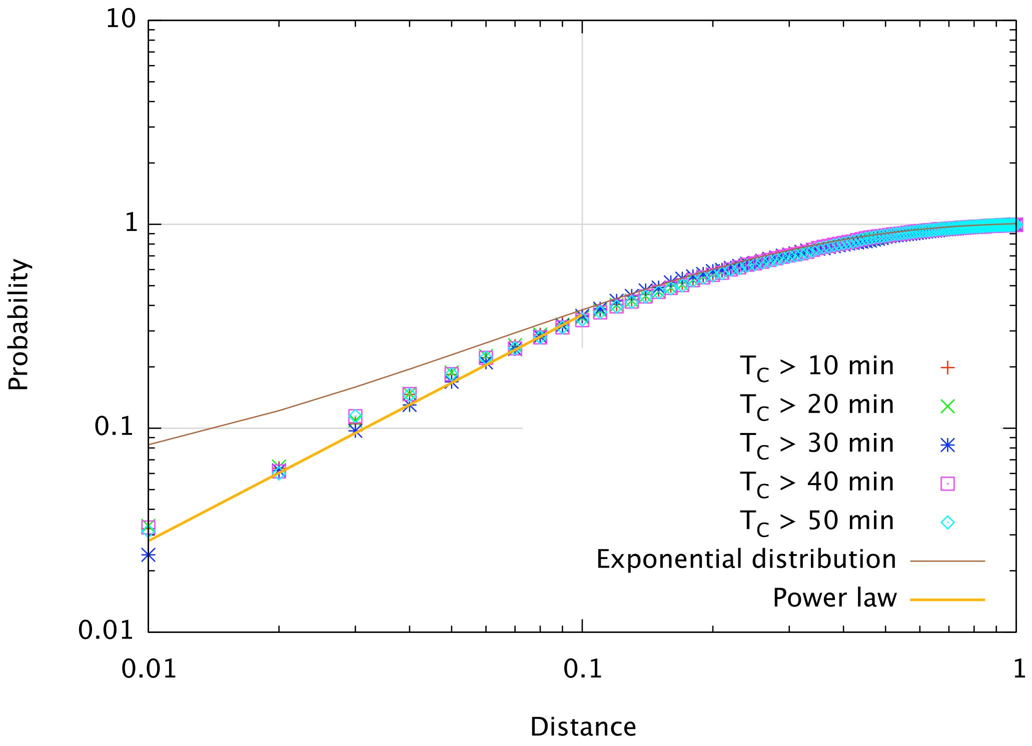


Figure 1.7: “Filtered” probability density functions, $f_{X^{(SOR)}}(x)$, obtained for different values of T_C, and filtered exponential distribution that approximates them.

T_C . Additionally, it arises that the exponential distribution provides an accurate approximation of $X^{(SOR)}$ for large values of x, while the power law distribution is more accurate for small values of x. Accordingly, $f_{X^{(SOR)}}(x)$ can be approximated as follows:

$$f_{X^{(SOR)}}(x) \propto \begin{cases} x^{\beta_{SOR}} & \text{if } x < x_{\text{thresh}}. \\ e^{-\gamma_{SOR} \cdot x} & \text{if } x > x_{\text{thresh}}. \end{cases} \tag{1.5}$$

In my case, for example, $\beta_{SOR} = 0.12$, $\gamma_{SOR} = 3.87$, and $x_{\text{Thresh}} = 1$.

This dichotomy in the behavior of $X^{(SOR)}$ – that is, it is power-law for low values of x and exponential for high values of x – is in line with what has been recently demonstrated in [42].

In Figure 1.8 I show the number of SOR relationships established versus the value of T_C. As expected, the number of relationships decreases as the value of T_C increases.

Same discussions can be done by observing Figure 1.9 where I show the probability density function $f_{X^{(SOR)}}(x)$ for different values of T_I. In this case, I have assumed that $N_C = 2$ and $T_C = 30$ min. Also in this case the number of relationships established decreases as the value of T_I increases.

Finally, similar observations can be done by considering Figures 1.10 and 1.11 which are analogous to 1.6 and 1.7, respectively, but have been obtained by using different values of N_C.

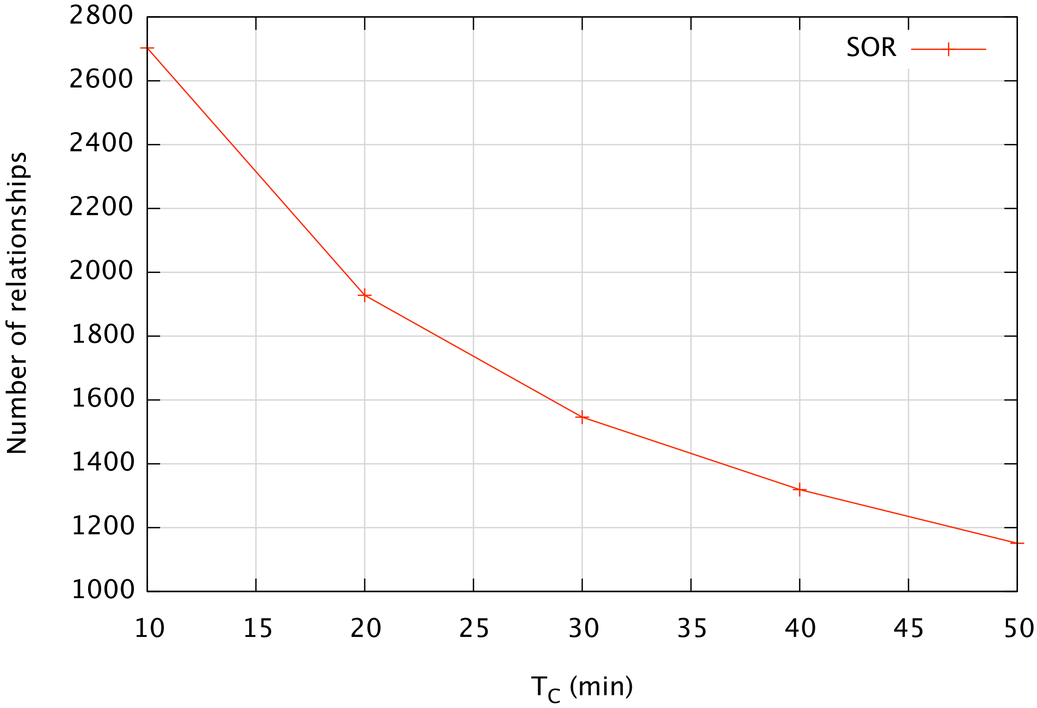


Figure 1.8: Number of SOR relationships established versus the value of T_C, when $N_C = 2$ and $T_I = 8$ hours.

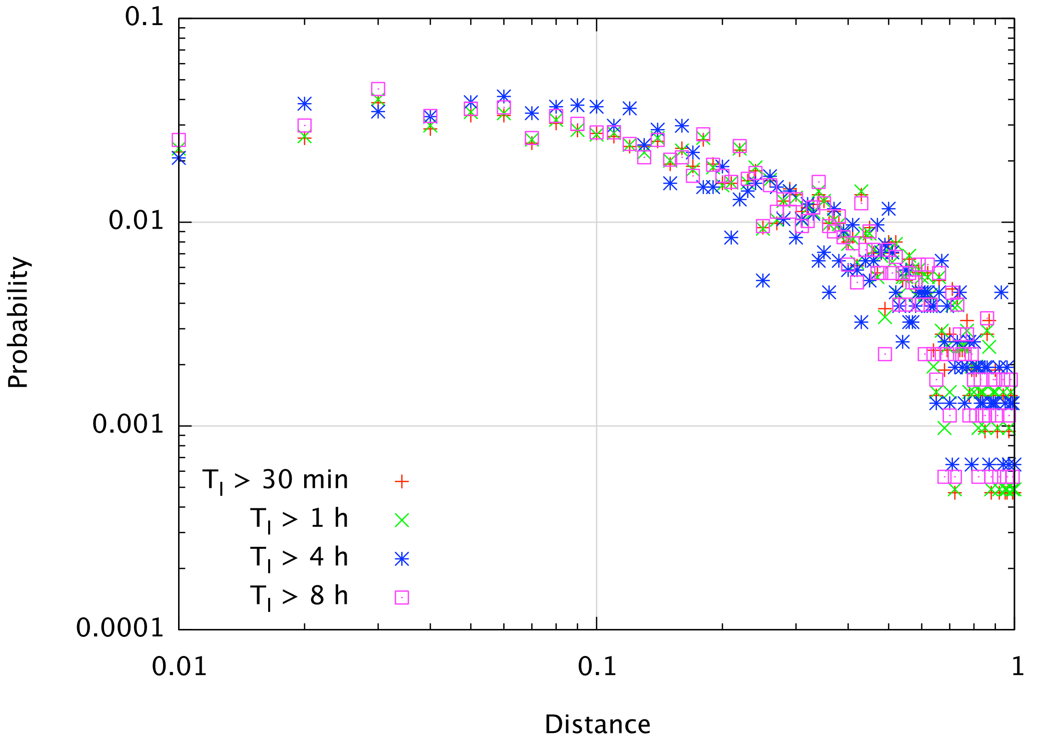


Figure 1.9: Probability density functions, $f_{X(SOR)}(x)$, obtained for different values of T_I.

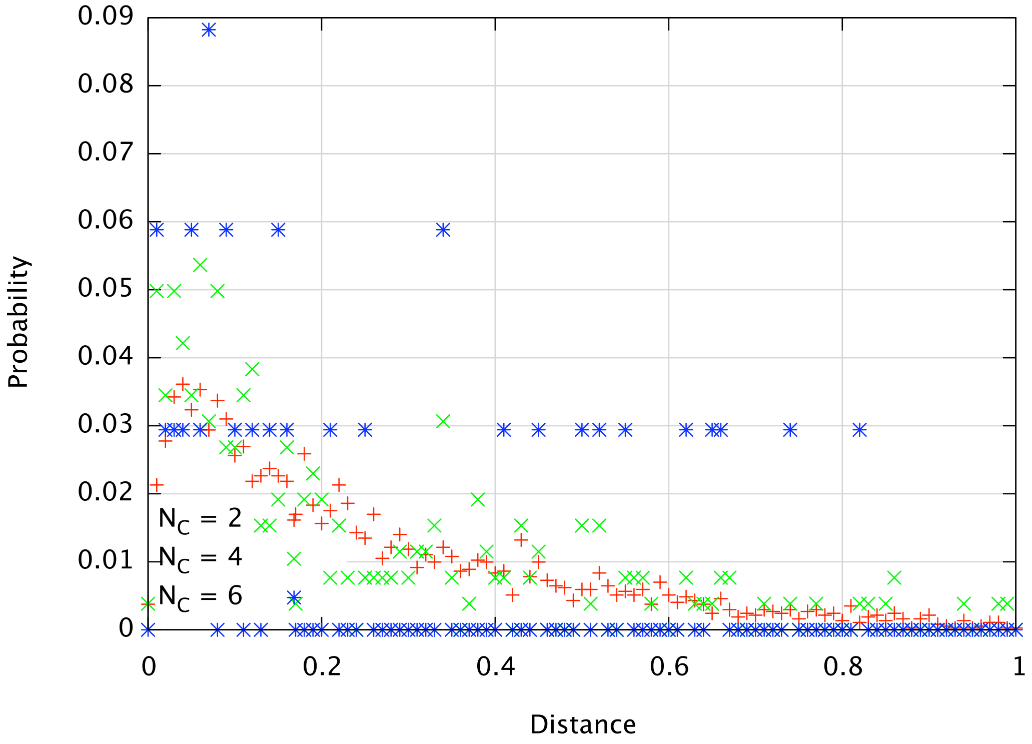


Figure 1.10: Probability density functions, $f_X^{(SOR)}(x)$, obtained for different values of N_C.

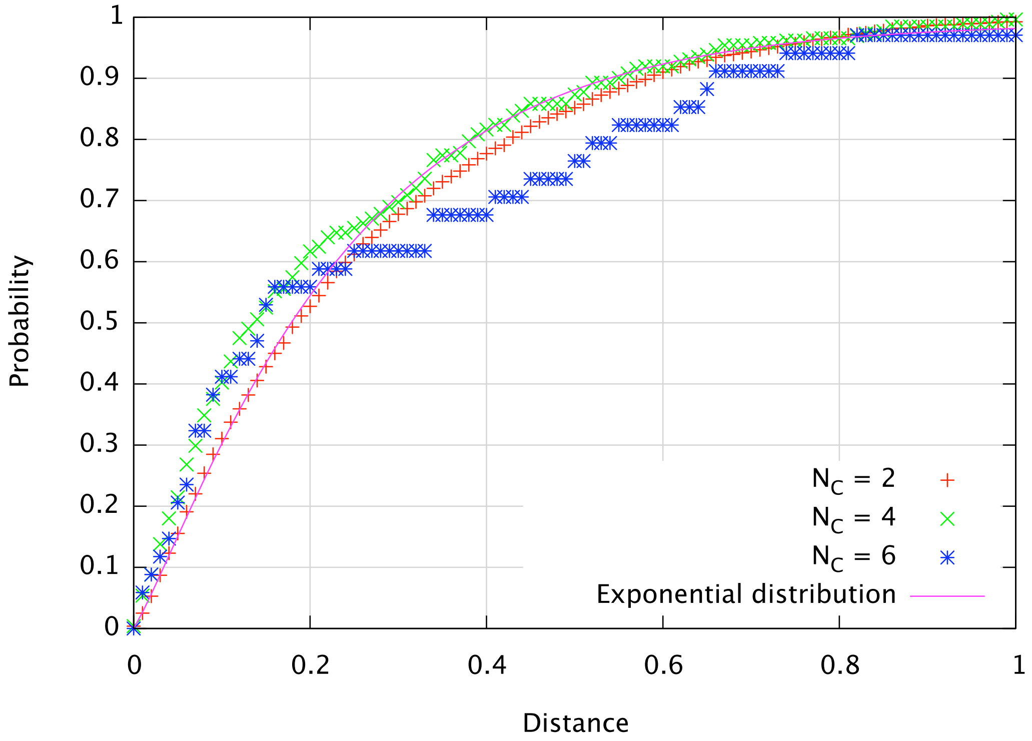


Figure 1.11: “Filtered” probability density function, $f_{X^{(SOR)}}(x)$, obtained for different values of N_C, and filtered exponential distribution which approximates them.

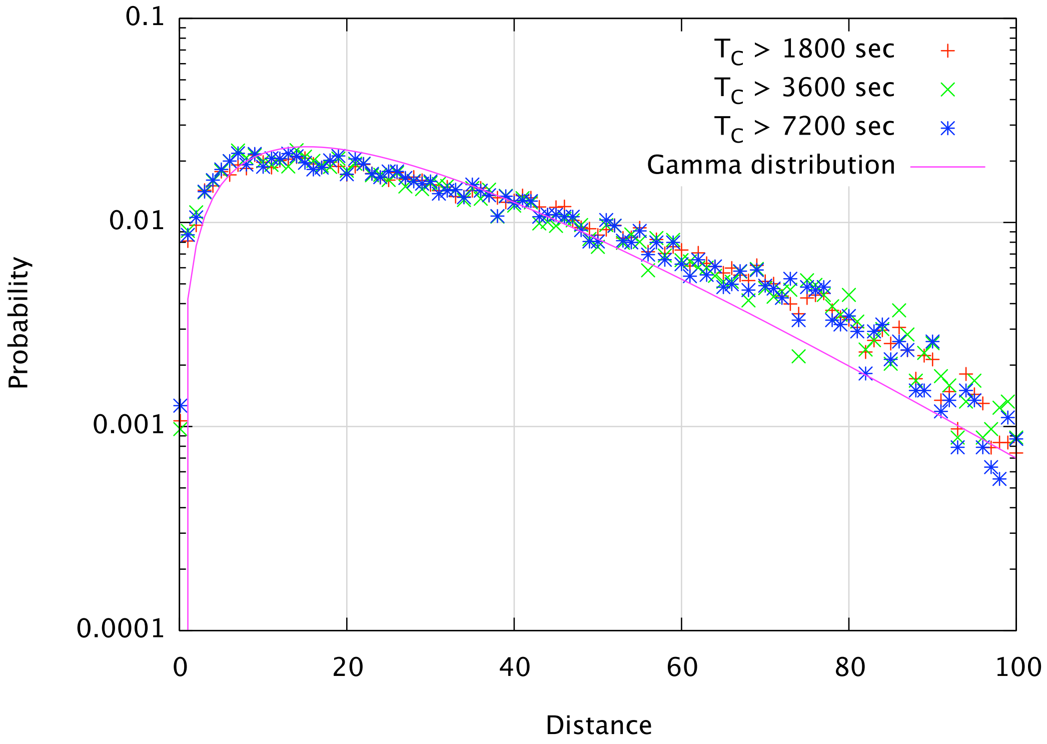


Figure 1.12: Probability density functions, $f_X^{(C-WOR)}(x)$, obtained for different values of T_C, and Gamma distribution which approximates them.

CWOR

In Figure 1.12 I show the pdf of $X^{(C-WOR)}$ obtained when I impose that a co-work social relationship is established only when the objects "meet" in a certain set of locations (offices, fabrics, laboratories, etc.) and that such meetings last for longer than T_C. More specifically, in the figure I represent the results obtained by considering different values of T_C; furthermore I show the Gamma distribution that approximates the above pdfs. By observing Figure 1.12, I notice that the value of T_C does not have a significant impact on the probability distributions $f_{X^{(C-WOR)}}(x)$ and that the Gamma distribution provides an accurate approximation of such pdfs. Indeed, I have

$$f_{X^{(C-WOR)}}(x) = x^{k-1}\frac{e^{-x/\theta}}{\Gamma(k)\theta^k} \tag{1.6}$$

where $\Gamma(k)$ is defined as follows:

$$\Gamma(k) = \int_0^\infty t^{k-1}e^{-t}dt \tag{1.7}$$

and $\theta = 15.93$ whereas $k = 2.11$.

In Figure 1.13 I show analogous curves when there are no predetermined locations in which co-work object relationships can be established.

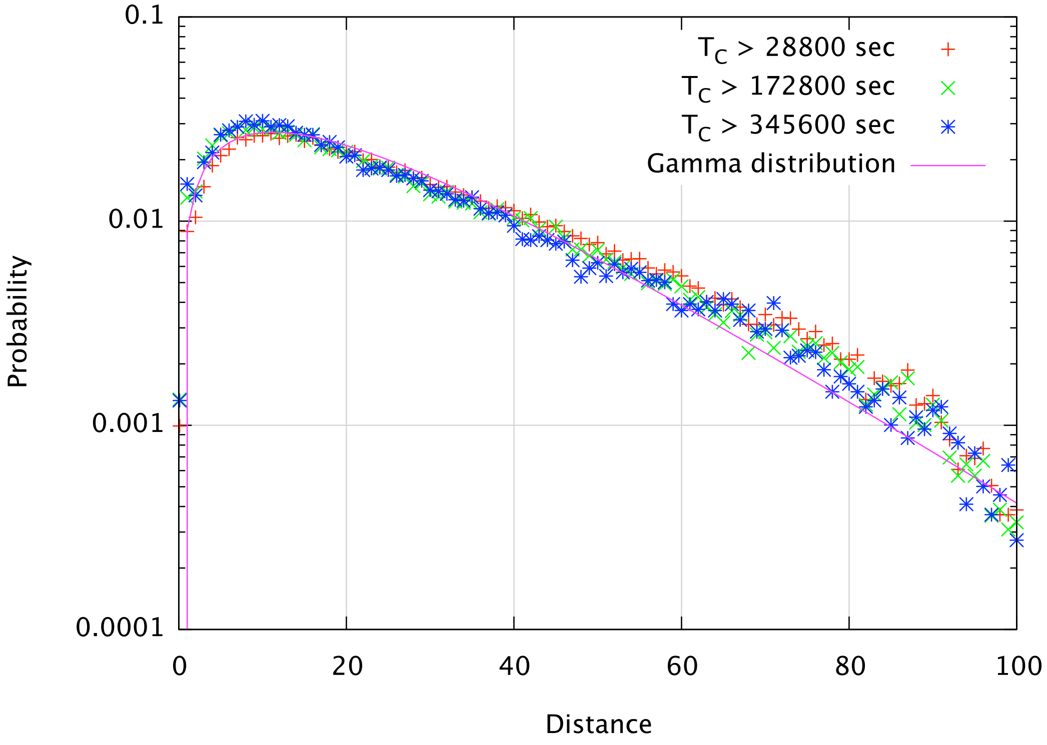


Figure 1.13: Probability density functions, $f_{X^{(C-WOR)}}(x)$, obtained for different values of T_C, and Gamma distribution which approximates them.

Chapter 2

Network Navigability

In the previous Chapter, I presented the new paradigm known as Social Internet of Things, which proposes the integration of social networking concepts into the Internet of Things. The underneath idea is that every object can look for the desired service using its friendships, in a distributed manner.

A SIoT network is based on the idea that every object can look for the desired service by using its relationships, querying its friends, the friends of its friends and so on in a distributed manner, in order to guarantee an efficient and scalable discovery of objects and services following the same principles that characterize the social networks between humans. The assumption that a SIoT network will be navigable is based on the principle of the sociologist Stanley Milgram about the small-world phenomenon. This paradigm refers to the existence of short chains of acquaintances among individual in societies [43].

According to this paradigm, each object has to store and manage the information related to the friendships, implement the search functions, and eventually employ additional tools such as the trustworthiness relationship module to evaluate the reliability of each friend. Clearly, the number of relationships affects the memory consumption, the use of computational power and battery, and the efficacy of the service search operations. It results that the selection of the friendships is key for a successful deployment of the SIoT. In this Chapter, I intend to address this issue by analyzing possible strategies for selection of appropriate links for the benefit of overall network navigability. I first propose five heuristics which are based on local network properties and that are expected to have an impact on the overall network structures. I then perform extensive experiments, which are intended to analyze the performance in terms of giant components, average degree of connections, local clustering and average path length.

The Chapter is organized as follows. In Section 2.1 I present the scenario of the social IoT and provide a quick survey of the solutions for the search of services in the IoT. In Section 2.2 I introduce the key aspects of network navigability, whereas Section 2.3 presents the strategies for link selection and the experimental evaluation.

2.1 Service search in IoT

In this section, I provide some examples of the existing solutions for service search in IoT context, in order to highlight existing problems. For example, in [44] every sensor carries a textual description in the form of keywords; data are organized through a two-tiered hierarchy of mediators where the ones in the lower level are responsible for groups of sensors in geographical areas, while the single mediator on the top level maintains an aggregated view of the entire network. However, this approach is not suitable for global search since it proposes a centralized approach that does not scale well in case of frequent data changes and only supports searches for pseudo static metadata. The approach in [45] follows a similar model: it uses three level of mediator instead of two, to deal with tag mobility, but still results inappropriate for global networks.

In [37], the authors propose a centralized system where objects are contacted based on a prediction model that calculates the probability of matching the query. In this way, the search engine does not need to contact all the sensors leading to good scalability with the number of objects; nevertheless, it is not scalable with the network traffic, since the number of possible results is significantly larger than the number of actual results, so a lot of sensors are contacted for no reason.

2.2 Reference scenario

2.2.1 Distributed search in the IoT

In the same way the search of documents of different kind, such as videos and web pages, is one of the most popular services on the Internet, the search of data from sensors and real-world entities is expected to be a major service in the IoT in the near future. However, the huge number of objects and the frequent changes in their data put a great stress on the service search.

In the SIoT, the objects inherit some capabilities of the humans and mimic their behavior when looking for new friends or services [26]. Indeed, the relationships devised for the SIoT follow the ones studied in sociological and anthropology fields, such as [28] and [14], since the owner sets the rules for their creation. The object then creates and manages several kinds of relationships and uses them to navigate the network, looking for services. The object asks its friends if they can provide a particular service or if they “know”, i.e. if they have any connections to, nodes that can provide it.

Figure 2.1 provides a simple example of a SIoT network, where links represent friendship ties while the bold line is the best route for node 1 to reach the requested service. In this network, when node 1 needs a particular service, it does not send a request to a centralized search engine, but it uses its own friendships to look for, in a decentralize manner, a node with the desired service, by contacting its friends and the friends of its friends. In this scenario, I aim to evaluate the impact of several strategies for link selection in order to select an optimal set of friendships to limit the use of computational resources needed for the search operations.

2.2.2 Key aspects of Network Navigability

In the past years, the problem of network navigability has been widely studied. As defined by Kleinberg [46], a network is navigable if it "contains short paths among all (or most) pairs of nodes". Several independent works, such as [47] and [48], formally describe the condition for navigability: all, or the most of, the nodes must be connected, i.e. a giant component must exist in the network, and the effective diameter must be low. In other words, the greatest distance between any pairs of nodes should not exceed $log_2(N)$, where N is the number of nodes in the network.

When each node has full knowledge of the global network connectivity, finding short communication paths is merely a matter of distributed computation. However, this solution is not practical since there should be a centralized entity, which would have to handle the requests from all the objects, or the nodes themselves have to communicate and exchange information among each other; either way a huge amount of traffic would be generated.

Nevertheless, starting from the Milgram experiment [43], Kleinberg concluded that there are structural clues that can help people to find a short path efficiently even without a global knowledge of a network [49] [25]. This means that there are properties in social networks that make decentralized search possible. Let us suppose to have a network as represented in Figure 2.1, where node 1 wants to get access to the information owned by node 10 (1 doesn't know where the information is located); obviously the optimum path leads through nodes 5 and 7. However, node 1 has three possible paths to choose from and only knows few information about its neighbors: the property that will guide node 1 to select node 5 as a next hop is that node 5 has a high degree of centrality, i.e. it has many connections. As such, node 5 represents then a network hub, i.e. a node that is connected to many other nodes. The ability for a node to quickly reach a network hub is assured by the existence of network clusters where nodes are highly interlinked: this characteristic is assured with high value of the local clustering coefficient, described by Watts and Strogatz [50], and is calculated for each node in a network. It measures how close the neighbors of a node are to being a clique, i.e. a complete graph, and it

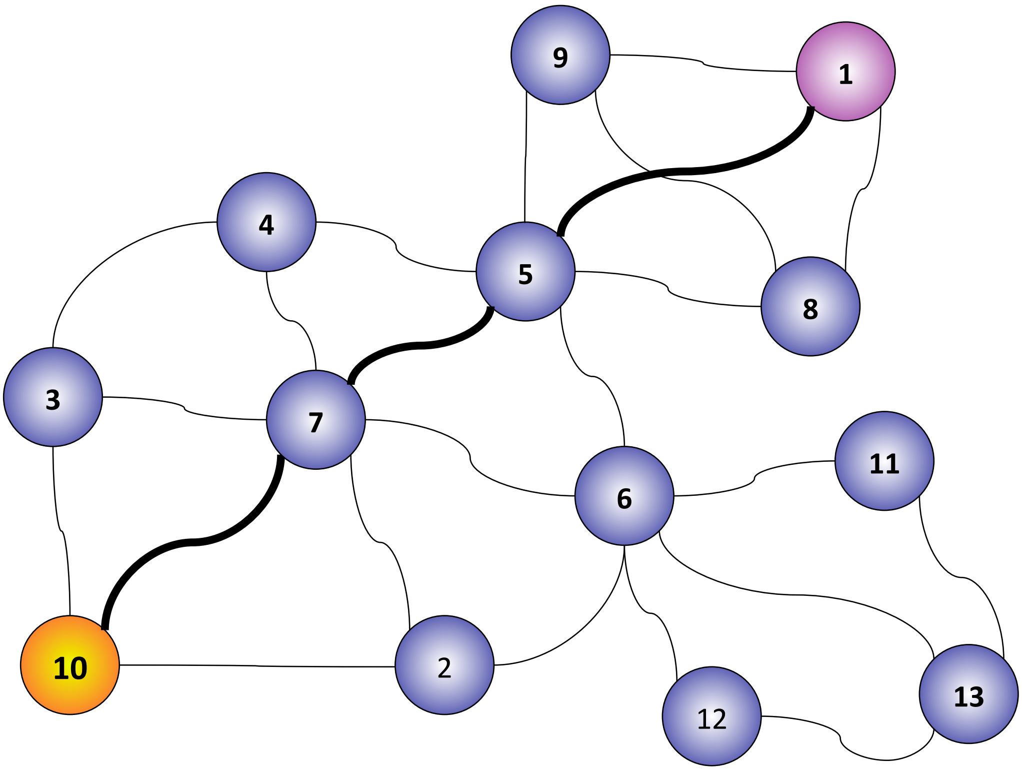


Figure 2.1: Decentralized search

is calculated as follows:

$$C_{local}(n) = \frac{2 * e_n}{k_n * (k_n - 1)} \tag{2.1}$$

where k_n represents the number of neighbors of the node n and e_n is the number of edges among the neighbors.

Still, node 5 needs some additional hints in order to choose node 7 over node 6, since both of them have the same degree. This characteristic is the node similarity, an external property to the network, derived from some additional information about the nodes. In the SIoT, node similarity will depend on the particular service requested and on the types of relationships involved.

The problem of global network navigability is then shifted to the problem of local network navigability, where neighboring nodes engage in negotiation to create, keep or discard their relations in order to create network hubs and clusters.

2.3 Experimental evaluation

2.3.1 Selection of network links

As described in Chapter 1, objects can create, through the mimic of their owner's behavior, several types of relationships. Other types of friendships could be added in the future, leaving to the node the hard work to cope with a huge number of connections. To make the service search process more efficient and scalable, I propose five heuristics to help the nodes in the process of selection of the best set of friends.

At first, a node accepts all the friendship requests until it reaches the maximum number of connections allowed - N_{max}. This parameter is intended to limit the computational capabilities a node needs to resolve a service search request. Then, a node applies one of the following strategies, to manage any further request:

1. A node refuses any new request of friendships so that the connections are static.
2. A node accepts new friendships and discards the old ones in order to maximize the number of nodes it can reach through its friends, i.e. to maximize the average degree of its friends; the node sorts its friends by their degree and the node with the lowest value is discarded.
3. A node accepts new friendships and discards the old ones in order to minimize the number of nodes it can reach through its friends, i.e. to minimize the average degree of its friends; the node sorts its friends by their degree and the node with the highest value is discarded.
4. A node accepts new friendships and discards the old ones in order to maximize its own local cluster coefficient; the node sorts its friends by the number of their common friends and the node with the lowest value is discarded.
5. A node accepts new friendships and discards the old ones in order to minimize its own local cluster coefficient; the node sorts its friends by the number of their common friends and the node with the highest value is discarded.

Let us consider a network example, as shown in Figure 2.2, where the maximum number of connections is set to $N_{max} = 5$ and let us suppose that node 2 sends a friendship request to node 1 (dashed line). Since node 1 has already reached N_{max} connections, the decision on this request will depend on the implemented strategy. If node 1 implements strategy 1, it will simply refuse the request; with strategy 2, node 1 checks the degree of all its friends and of node 2 and then it terminates the relationship with node 4, which has only one more friend, in

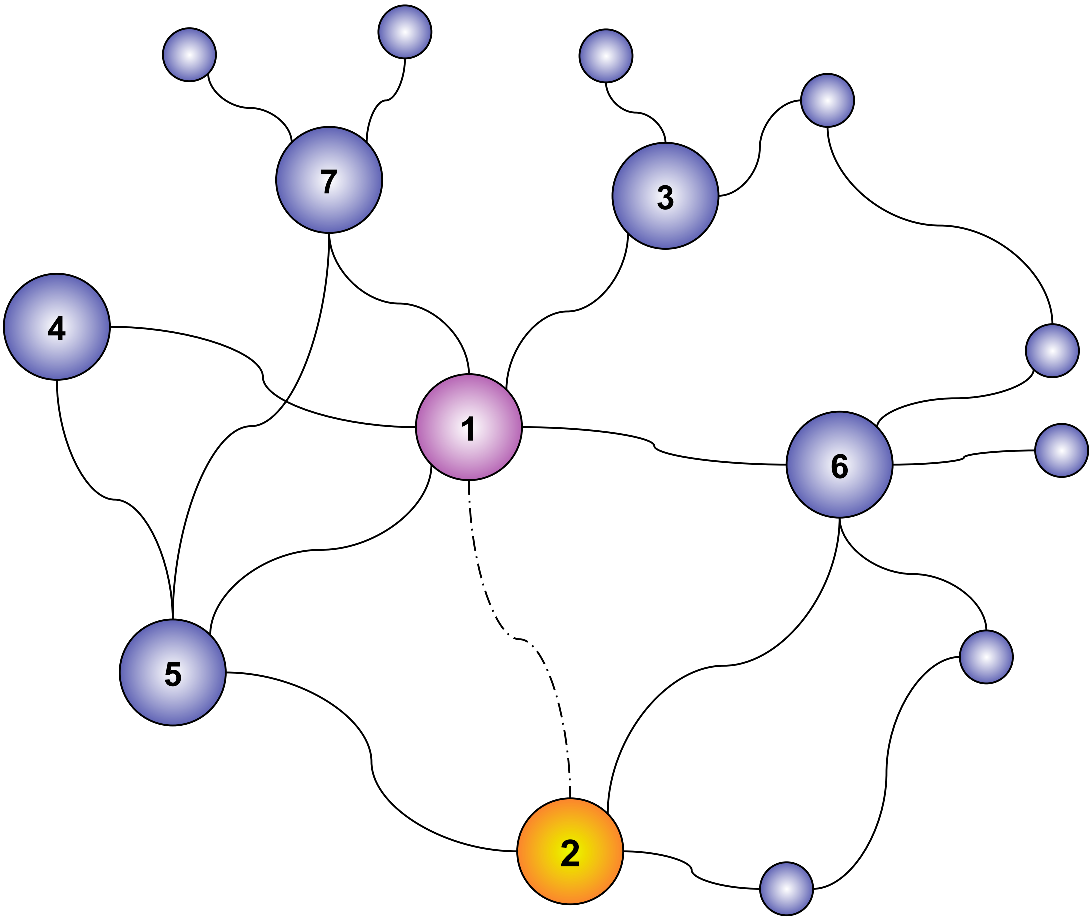


Figure 2.2: Selection of network links

order to accept the request from node 2 (3 friends). In the same way, using strategy 3, node 1 terminates the relation with node 6, which has N_{max} connections, and accepts the request. With strategy 4, node 1 compares the common friends among its friends and with the requester node and discards the node 3 with which it has no common friends. In a similar way, with strategy 5, node 1 discards the relation with node 5 to which it has the highest number of common friends.

2.3.2 Simulation setup

With this simulation analysis, I want to study the impact of each of the proposed strategies on the objects' network navigability.

To analyze the navigability of a SIoT network, I would need information about the requests of establishing new relationships the objects would receive on the basis of their profile, settings and movements. And I would need this information for huge numbers of real objects. Even if some platforms already exist that implement the SIoT paradigm, such as [51], this data is not available to date as real applications have not been deployed yet. For this reason I had to adopt an alternative solution to test my heuristics as follows:

Table 2.1: Parameters of Brightkite, SIoT network and Barabsi-Albert model

	Brightkite	SIoT network	BA model
Nodes	12275	14557	15000
Number of Edges	39515	67363	75000
Average Path Length	4.570	4.534	3.905
Average Clustering Coefficient	0.247	0.375	0.255
Diameter	14	14	6
Average Degree	6.631	9.808	10
Giant Component	84.32%	93.45%	100%

1. first I analyze a social network of humans;
2. from this, I extract the information needed to build the social network of objects;
3. in the next stage, I extract the characteristics of this network and use these to run a model that generates synthetic networks with similar properties;
4. and finally I apply the strategies described previously and analyze the results.

For the first step I relied on the real dataset of the location-based online social network Brightkite obtained from the Stanford Large Network Dataset Collection [52]. This dataset consists of more than 58k nodes and more than 200k edges, so in order to better analyze its properties and compare them to synthetic data, I consider only the nodes enclosed between Atlanta and Boston for a total of approximately 12k nodes and 40k edges. However, the output of the Brightkite dataset is a trace of the position of humans and of their relationships; since I am interested in the relationships of the objects I have extended it as follows (step 2): starting from the scaled network, I suppose that every person carries at least one smart object, for example a smartphone, so when they get in touch with their friends their objects also come into contact and have then the possibility to create a SOR. In a similar way, I also simulate the creation of C-WOR and C-LOR. The resulting SIoT network has around 14.5k nodes and 67k edges. The parameters of the two networks, obtained from Gephi [53], are showed in Table 2.1, while the node distribution is shown in Figures 2.3 and 2.4 for Brightkite and SIoT network respectively.

Both networks comply with the condition for network navigability: at global level, there is a giant component and the average path length is low; at local level, I can observe how the nodes are highly interlinked, thanks to the high values of the clustering coefficients, and the networks have a scale-free degree distribution thus indicating the existence of hubs.

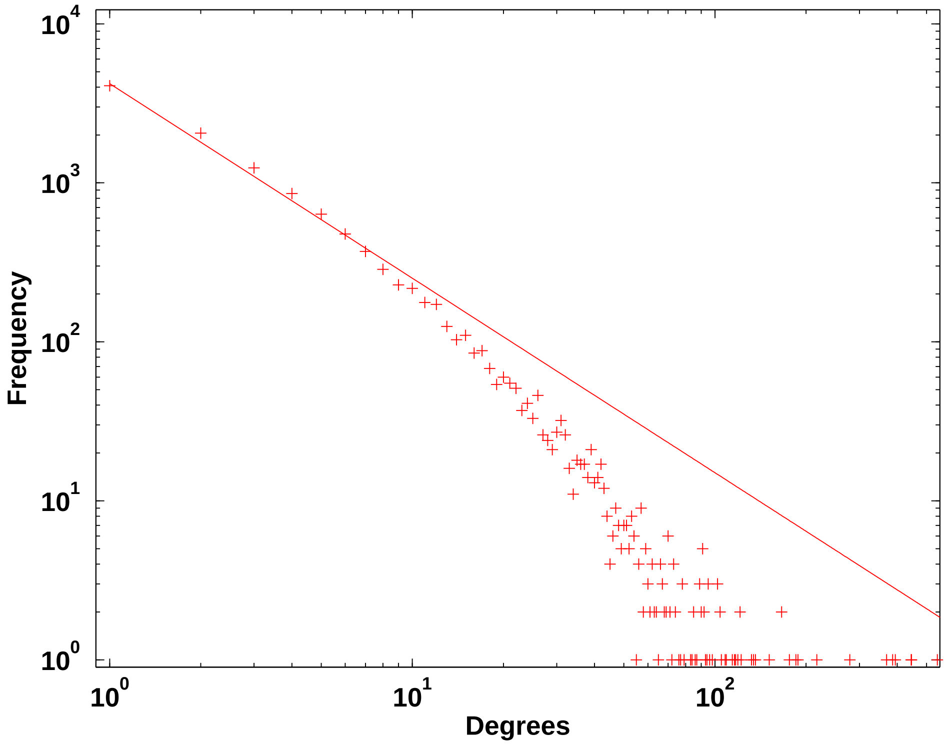


Figure 2.3: Degree distribution for Birghtkite

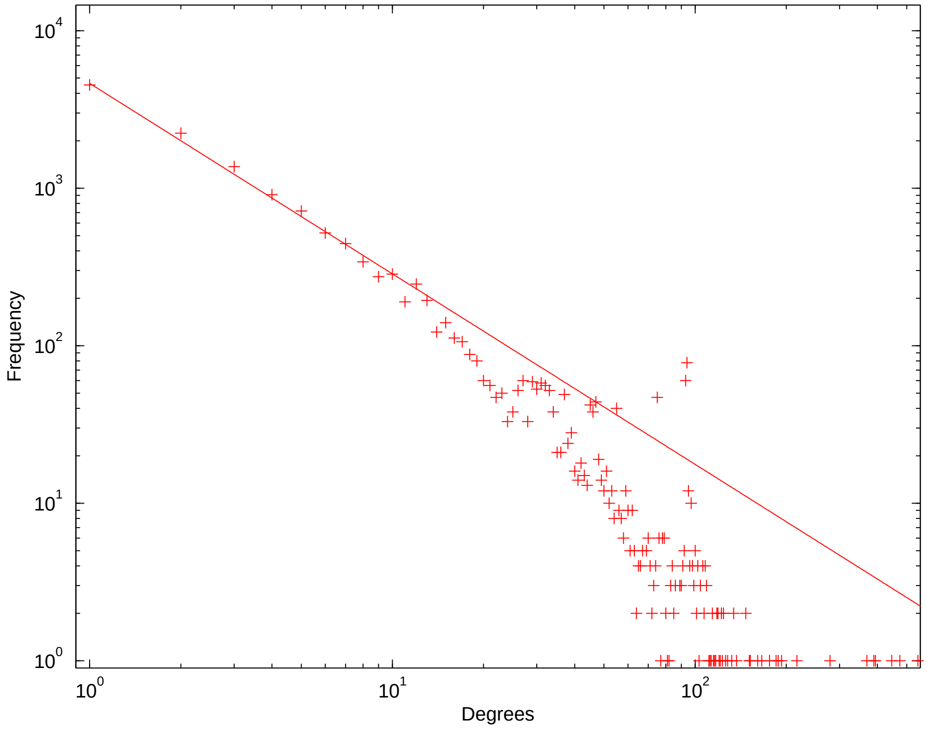


Figure 2.4: Degree distribution for SIoT

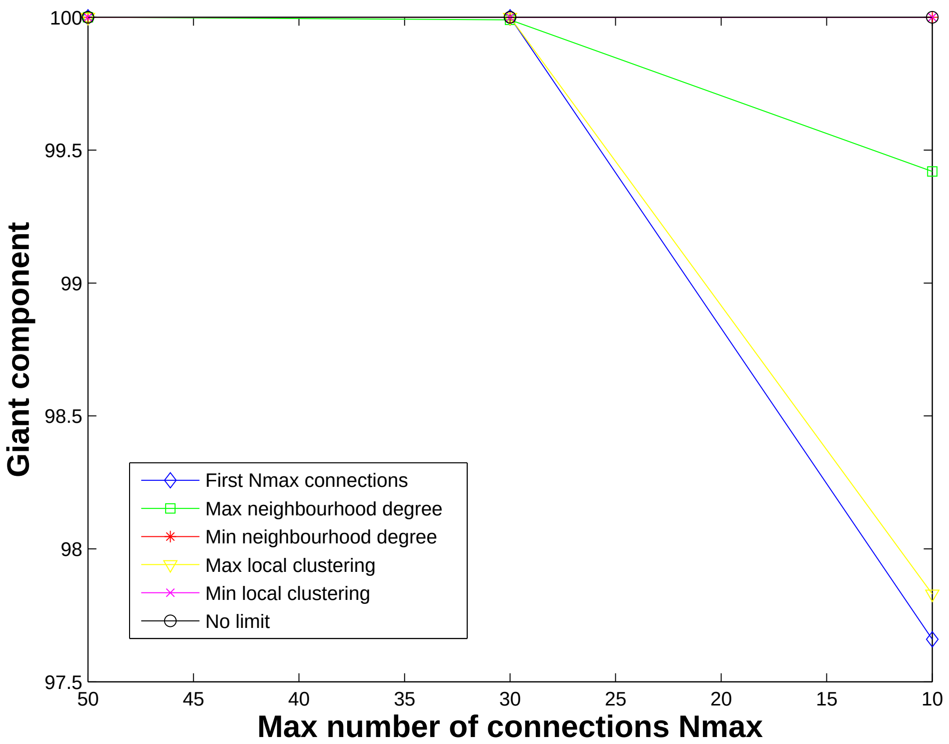


Figure 2.5: Giant component for all the strategies

Moreover, it is important to point out that the tail of the degree distribution deviates from the power law due to the scaling, where nodes near the borders do not have a complete set of friendships. Furthermore, even if the SIoT is expected to have a shorter average path length with respect to classical social networks, in this case this does not happen since the new relationships are due to C-LORs and C-WORs that are indeed short range; however, for the same reason, it is possible to observe a 50% increment of the average local clustering.

To generate and analyze similar networks, I rely on the Barabási-Albert model [3], which is able to generate scale-free networks based on preferential attachment. Starting with a small number of nodes, at each step, it adds a new node with m edges (m is a parameter for the model) linked to nodes which are already part of the system. The probability p_i that a new node will be connected to an existing node i depends on its degree k_i, so that $p_i = k_i/(\sum_j k_j)$ leading to the name preferential attachment. However, since this model generates networks with a low average cluster coefficient, I use the modified version from Holme and Kim [54], that adds a triad formation step to the model: after a node j connects to node i, it also connects to one of its neighbor, thus resulting in a triad formation. The results of this model, using 15k nodes, connecting each node to $m = 5$ other nodes and averaged over 5 runs, are shown in Table 2.1, and it can be observed that it represents a good approximation for the real scenario.

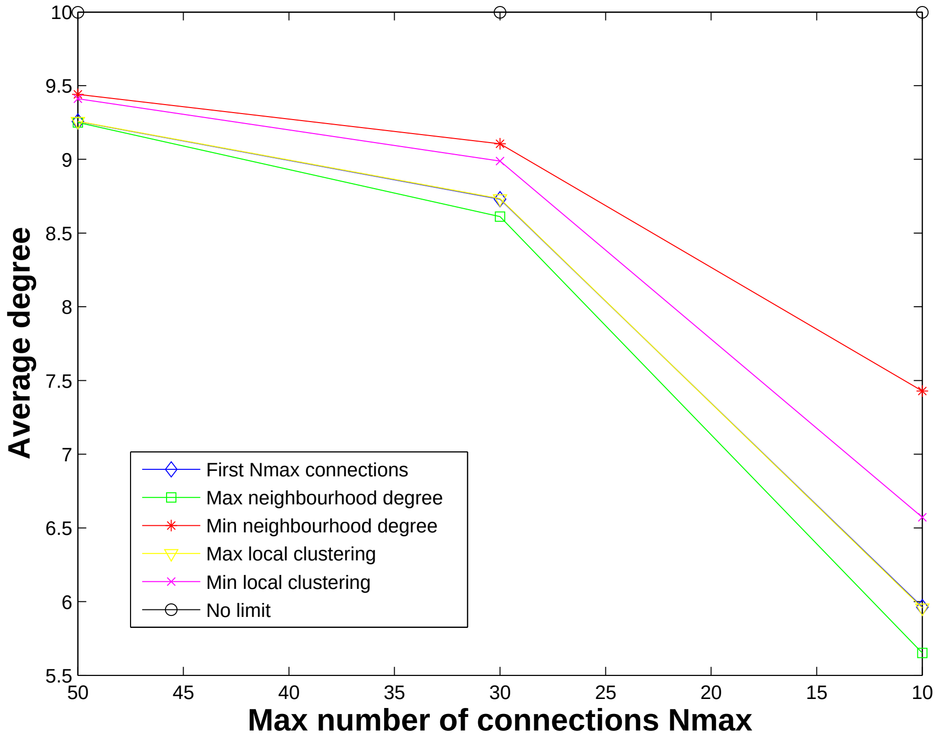


Figure 2.6: Average degree for all the strategies

2.3.3 Simulation results

Figure 2.5 shows the percentage of the giant component for all strategies. It is important to note that if I try to minimize the neighborhood degree or the local clustering, I can always achieve a giant component which includes all nodes. This happens due to the fact that, when a node with N_{max} connections receives a friendship request from a low connected node, it will always accept it to the detriment of a node with higher connectivity which has high probability to remain connected to the network. Moreover, I can observe that when using the strategy 1, 2 or 4, the dimension of the giant component naturally decreases with the reduction of the N_{max} value, thus making the network not fully navigable. In the case of using the strategy 2, a node connected to other nodes with N_{max} friends will not accept any other relation request, similarly to a node in a near-clique in strategy 4. Furthermore, I also want to point out, that with strategy 2 and 4 a node can not refuse or discard relationships if this action is going to isolate a node; in this way, I can achieve larger giant component and I do not have isolated nodes but at least isolated couples of nodes.

From Figure 2.6 I can observe how the average degree changes with different strategies. Strategy 3 tries to equalize the number of friendships between the nodes, resulting in a higher number of relationships in the network and consequently a higher average degree. Similarly,

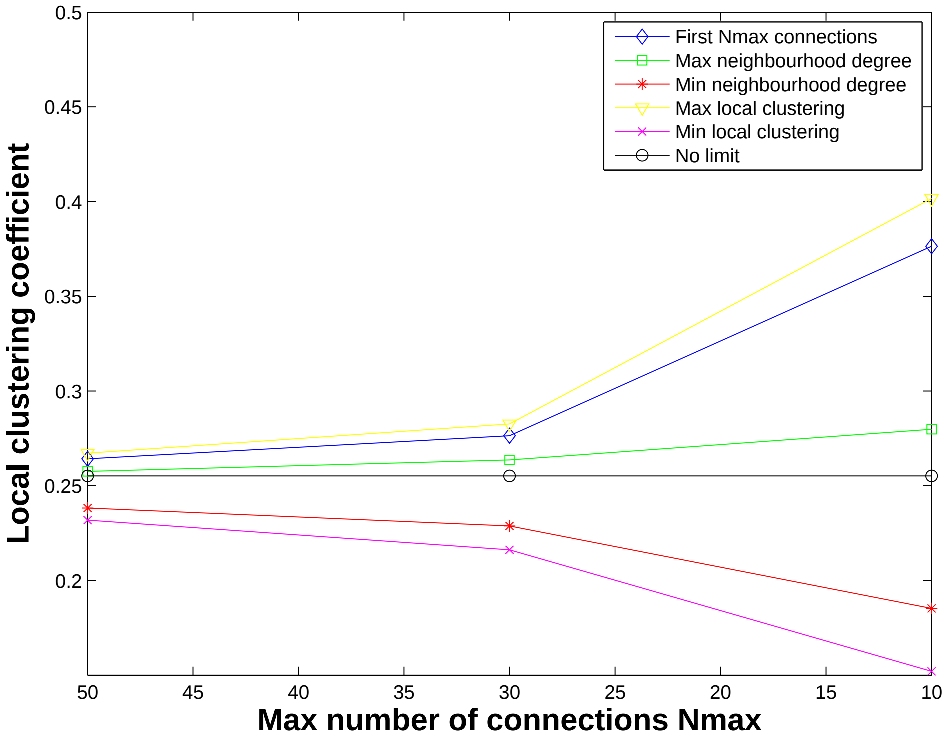


Figure 2.7: Local cluster coefficient for all the strategies

strategy 5 discards the nodes with higher local cluster coefficient, to connect with nodes with low values. Yet, since the local cluster coefficient is not directly connected to the number of friends, the average degree is lower than in strategy 3. Strategy 2 achieves the lowest average degree due to the fact that the resulting network has a core of high degree nodes, with N_{max} friendships, and highly interconnected between themselves. These nodes hardly accept any new friendship, leaving many nodes with a low degree.

Figure 2.7 shows the local cluster coefficient. Strategy 4 and 5 exhibit the highest and lowest value respectively, since they are designed to achieve these results. Strategy 1 has a high value due to the triad formation step in the model and to the fact that there is not further rearrangement of relationships after these has been created; this effect is even stronger when the number of maximum connections is decreasing. Strategy 2 achieves a higher value than the model since the core nodes in the network are highly interconnected. It is important to point out the behavior of the local clustering coefficient for Strategy 3: it has a lower value than the model and decreases with N_{max}. This is a result of the equalization of the number of friendships, leading to a high average degree and easily destroying the triad formation step in the model.

Figure 2.8 shows the average path length. It indicates that strategy 3 and 5 provide shorter paths than other strategies. This is due to the fact that these strategies manage to create many long distance relationships. On the other hand, strategy 4 has the worst performance for the

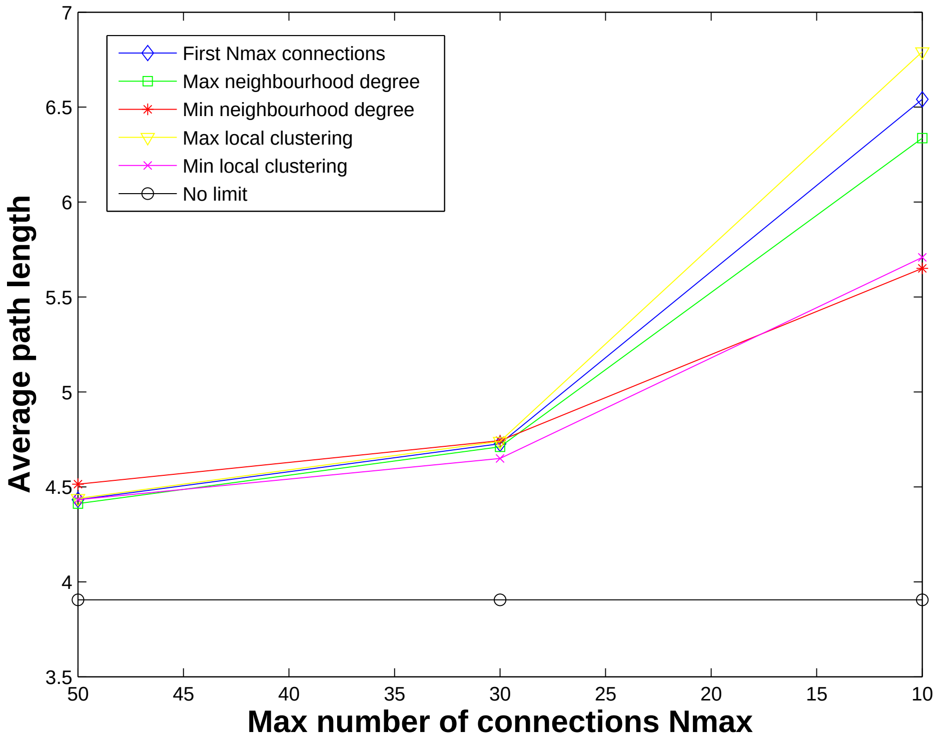


Figure 2.8: Average path length for all the strategies

exact opposite reason: nodes are too close to be a clique and have difficulties reaching other nodes; similar reasons hold also for strategy 1 and 2.

Chapter 3

Trustworthiness Management

The focus of this Chapter is on the problem of understanding how the information provided by members of the social IoT has to be processed so as to build a reliable system on the basis of the behavior of the objects. The major contributions presented in this Chapter are the followings:

- Definition of the problem of trustworthiness management in the social IoT, where the objects autonomously establish social relationships and use the resulting network to find the trusted peer(s) that can provide the desired service when needed.
- Definition of two models for trustworthiness management starting from the solutions proposed for *Peer to Peer* (P2P) and social networks. In the subjective model, more similar to the social scenario, each node computes the trustworthiness of its friends on the basis of its own experience and on the opinion of the friends in common with the potential service provider. In the objective model, obtained starting from the P2P scenario, the information about each node is distributed and stored making use of a *Distributed Hash Table* (DHT) structure so that any node can make use of the same information.
- Evaluation of the benefits of the trustworthiness management in the IoT, which shows how it can effectively isolate almost any malicious nodes in the network at the expenses of an increase in the network traffic caused by the exchange of feedback information.

This Chapter is organized as follows: in Section 3.1 I provide a survey of the research on trustworthiness management in P2P and social networks. In Section 3.2 I define the problem and introduce the used notations, whereas in Section 3.3 I illustrate the two models proposed. Section 3.4 presents the system performance.

3.1 Background

In the first subsection I review the techniques that have been proposed for trustworthiness management in P2P networks. This scenario is similar to mine, as in both cases there are services or objects that provide and request information from other peers and then in both cases the evaluation of the reliability of the members of the community is vital. However, it is not beneficial to apply directly, as they are, the solutions seen for P2P systems to the Social IoT, since all the information about the social aspects would be lost. Indeed, works dealing with trust evaluation in human social networks, which I review in the second subsection, provide me with important contributions on how to exploit the concepts of centrality, credibility and link characteristics in trust evaluation in the Social IoT.

3.1.1 State of the Art in P2P Networks Trust Management

There are only few works about the trust management in IoT. In [55], the authors propose a model based on fuzzy reputation for trust evaluation to enforce things cooperation in a WSN of IoT/*Cyber Physical System* (CPS) based on their behaviors. In [56], by the use of social trust and Quality of Service (QoS) trust, a hierarchical trust management protocol is proposed. In [57], the authors use a service classification estimation table to evaluate the user's trustworthiness. In [58] users' trustworthiness in social networks is used to assist the service composition between objects.

Instead, problem of interacting with unknown peers and isolating malicious peers has been deeply investigated in P2P networks. To calculate a peer trustworthiness, a system has to store the reputation information, encourage the sharing of this information among the peers, and define the rules that from the reputation bring to the peer trust level (see Table 3.1).

There are different approaches that can be used to store trustworthiness information. As described in [59], all information can be stored in a centralized storage to foster sharing and make easy the processing; however, it easily leads to a single point of failure. In [60] , the information is distributed in storage peers. Other approaches are the rater-based storage [61], where each peer stores trustworthiness information about the peers it has observed, and the ratee-based storage [62], where each peer stores its own reputation information recorded during the past transactions.

For a reputation system is important to incentive the peers to cooperate and solve some well-known problems, such as Free-riders [63] and Tragedy of Commons [64]. A solution is the one proposed in [65], where a peer can buy and sell reputation information from/to other peers and loses credit if it behaves maliciously. When a peer decides to share its information, the

Table 3.1: Approaches used for the storage, sharing, and processing of the reputation information

Storage	Sharing	Processing
Centralized	Local	Average
Distributed	Part	Weighted Average
Rater-Based	Global	Probabilistic Estimation
Ratee-Based		

system has to cope with how effectively share them. This problem can be handled in different ways: local share, part share, and global share. In local share, each peer manages only the information it is involved with [66]. In part share, each peer shares the information with a set of specific peers. In [61], the authors propose to share the data through a reputation chain of acquaintances and neighbors, since it is more reliable than using random peers [61] [62], and in [67] peers have the possibility to periodically exchange their information. In global share, a mechanism is adopted to collect the information of all peers. This can be done both with a centralized storage [59] and with a distributed storage [60] .

Once the information is collected, it is important to use a computation system that is able to extract a reliable value of the trustworthiness. A simple mechanism relies on the use of an arithmetic average [68] of all the reputation values a node has received. Other models apply a weight to the reputation values in different ways: in [69], the authors use different weights for acquaintance and stranger peers; in [70] the weights are chosen on the basis of the last reputation value a node has received; [71] considers the similarities between two peers in terms of released feedback to weight the reputation value. In [60] , the authors assume the existence of a digraph of social links between peers, where reputation values are assigned to the link based on the transactions between the peers connected through a link. Finally, some algorithms make use of probabilistic estimation techniques [72], [73], and the maximum likelihood estimation [73] to match the reputation value into the probability that a peer will cooperate.

3.1.2 State of the Art in Social Networks Trust Management

In the past few years, online social networks have become more and more popular and consequently several methods to calculate trust, and sometimes distrust between two person [74] have been proposed, together with key applications to allow users to secure their data [75]. In these scenarios, it is considered a person (say Alice) to trust another person (say Bob) if her actions are based on the belief that Bob's behavior will lead to a good outcome. However, some works (e.g., [76]), add another dimension to the traditional probability model of belief and disbelief, considering ignorance as an essential part of human behavior.

In [77], the authors classify online social networks in three generations based on the level of sociality they present and show trust relation mechanisms for each generation. The first generation is characterized by weak sociality where the relationship between partecipants is implicit and the partecipants can not make a new friend with a friend's friend; the second generation has medium sociality and relationship between partecipants is only binary (friend or not friend), but partecipants have the possibility to extend their relationship list by adding friends of friends even if only inside the same social network platform. In the third generation of social network, different types of relationship exist and partecipants can establish new relationships and conduct activities across different social networks. Furthermore, multiple types of relationship between users have lead to the development of relationship-based techniques for trust management in Social Networks [78][79]. According to this definition, it is possible to consider the SIoT belonging to the third generation with explicit non binary relationship between participants.

The main properties of trust are well defined and many works contribute to describe them ([80], [81], [82], [83] and [84]). One of the most important and controversial is the *transitivity*, based on the concept of recommendation of someone that is not directly known, i.e., if Alice trusts Bob and Bob trusts Eric then Alice trusts Eric. Indeed, it has been demonstrated in [84] that in real life trust is not always transitive but depends on the particular service requested. In [83], constraints are given so that trust can be considered transitive if the trust edges have the same purpose and only in this case the trust system can exploit this property. These constraints imply that different trust matrixes have to be stored for every service, since if Alice trusts Bob for fixing her car, she could not trust Bob for advising her a good restaurant.

Another important property is called *composability*. It is the ability to compose the recommendations from different friends into a unique value and then decide whether to trust or not someone. With different trust values from different friends, a composition function is needed in order to obtain accurate results.

Since trust is related to a person's past experience, another important property in social network is the *personalization*. Accordingly, it's not unusual that two people have different opinions about the same person. For the same reason, trust is also *asymmetric*, i.e., two people tied by a relationship may have different levels of trustworthiness each other.

3.2 Introduction to the Proposed Solution

The SIoT provides the objects with some capabilities of the humans when looking for and providing information in their social communities, i.e., the objects mimic the human social behavior [26]. The type of relationships that have been devised for the SIoT have been taken from some sociology and anthropology studies (e.g., [28] and [14]). [85] provides some experimental analyses when implementing this behavioral model on the IoT. As in most of the IoT architectures, in SIoT the owner has the control on the object functions and social interactions. Among the supervision functionalities, the system (the object) asks the owner to authorize the provisioning of a particular service/piece of information to other objects' requests. The owner then empowers the object to allows for providing the service or not depending on the specific request (requesting object owner identity and interaction context). This is done at the first occurrences whereas the system learn and behave accordingly for the next transactions. The owner behavior indeed depends on the (direct and indirect) relationships with the requester and on his personality (collaborative, selfish, greedy, malicious and other).

Within this scenario, I aim at designing and experimenting a dynamic trust model for assessing the trustworthiness level of nodes. The next subsection describes the used notation, the second subsection illustrates the trust models, and the third one described the main elements used in the adopted models.

3.2.1 Notation and Problem Definition

The main focus of this Chapter is the design of a dynamic trust model for assessing the trustworthiness level of nodes in a Social Internet of Things. In my modelling, the set of nodes in the SIoT is $\mathcal{P} = \{p_1, ..., p_i, ...p_M\}$ with cardinality M, where p_i represents a generic node. In my problem setting, let the network be described by an undirected graph $\mathcal{G} = \{\mathcal{P}, \mathcal{E}\}$, where $\mathcal{E} \subseteq \{\mathcal{P} \times \mathcal{P}\}$ is the set of edges, each representing a social relation between a couple of nodes. Let $\mathcal{N}_i = \{p_j \in \mathcal{P} : p_i, p_j \in \mathcal{E}\}$ be the neighbourhoods of node p_i, namely the nodes that share a relation with p_i, and $\mathcal{K}_{ij} = \{p_k \in \mathcal{P} : p_k \in \mathcal{N}_i \cap \mathcal{N}_j\}$ be the set of common friends between p_i and p_j.

Let $\mathcal{S}_j$ be the set of services that can be provided by p_j. The reference scenario is represented by p_i requesting a particular service S_h. I assume that the Service discovery component receives the request of this service from p_i and returns to it a set of nodes $\mathcal{Z}_h = \{p_j \in \mathcal{P} : S_h \in \mathcal{S}_j\}$ that are able to provide the service S_h. For each of this potential service providers $p_j \in \mathcal{Z}_h$, the Service discovery component returns a set of edges $\mathcal{R}_{ij} = \{p^a_{ij} p^b_{ij}\}$, which represents the sequence of social links that constitute the selected path from p_i to p_j in the SIoT. At this point, the Trustworthiness management component is expected to provide the important function of listing the trust level of any node in $\mathcal{Z}_h$. This is the objective of this Chapter.

Figure 3.1 provides a simple example of a generic graph $\mathcal{G}$ where: $\mathcal{P} = \{p_1, \cdots p_{10}\}$, with

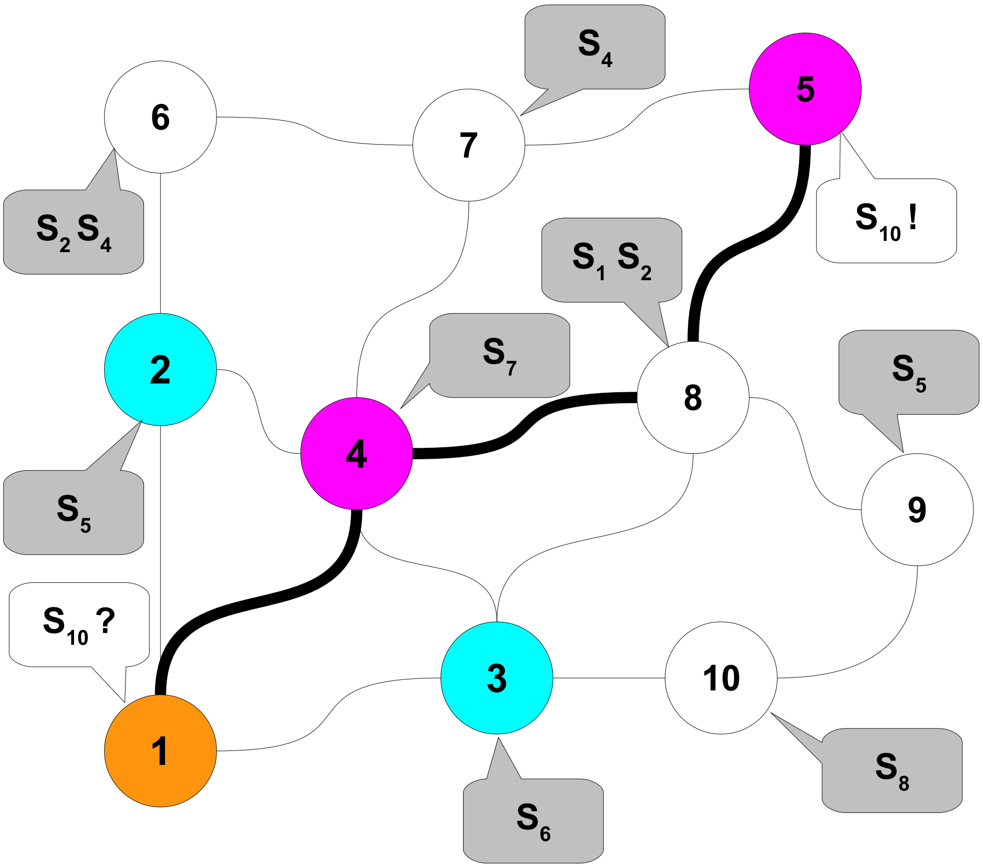


Figure 3.1: Representation of the network nodes

each node capable of providing one or two services, as highlighted in the grey cloud; p_1 is the node that is requesting the service S_{10}, as highlighted in the white cloud; $\mathcal{Z}_{10} = \{p_5\}$ is the set of nodes that can provide the requested service; $\mathcal{R}_{1,5} = \{p_1p_4, p_4p_8, p_8p_5\}$ is the set of edges that constitute the path returned by the Service discovery process for p_1 to reach p_5. In this figure, I also highlight the set $\mathcal{N}_1 = \{p_2, p_3, p_4\}$ of nodes that are friends of p_1 (in blue color). Within note that the set $\mathcal{K}_{1,4} = \{p_2, p_3\}$ of nodes represents the common friends between p_1 and p_4.

3.2.2 Trust Models

In such a scenario, I envision two possible models for the implementation of the Trustworthiness management component, based on the dimension of trust semantics [86]:

1. *Subjective trustworthiness*, derived from a social point of view, where each node p_i computes the trustworthiness of its $\mathcal{N}_i$ friends on the basis of its own experience and on the basis of that of its friends; I refer to this trustworthiness with T_{ij}, i.e., the trustworthiness of node p_j seen by node p_i. If p_i and p_j are

not friends, then the trustworthiness is calculated by word of mouth through a chain of friendships.

2. *Objective trustworthiness*, obtained from P2P scenarios, where the information about each node is distributed and stored making use of a DHT structure. This information is visible to every node but is only managed by special nodes that I call *Pre-Trusted Objects* (PTOs). I refer to this trustworthiness with T_j, i.e., the trustworthiness of p_j seen by the entire network.

Table 3.2 shows how the proposed models match the approaches described in Section 3.1.1 in terms of storage, sharing, and processing of the reputation information while Table 3.3 summarizes the properties taken from the social networks studies.

The proposed subjective approach shows all the properties typical of trust in online social networks, as described in Section 3.1.2. Indeed, the SIoT can be seen as an application where the objects establish relations and cooperate to provide new services to the users; according to this vision, trust is not related anymore to a particular service, since all the objects in the SIoT try to achieve the same goal and then it can be considered transitive in this scenario. Then, when p_i and p_j are not friends, the transitivity property is exploited. Still, a node uses a composability function to combine the recommendations from the $\mathcal{K}_{ij}$ friends. Moreover, trust is both personal and asymmetric since every object has its own opinion about the other nodes based on its personal experiences, which are different from node to node. These properties have been taken from the past works, whereas other new concepts have been introduced. When building the direct objects opinions, not only are the friendship links taken into account but also the type of relationship. When combining the indirect opinions about a node received from friends, I introduce weights that are built on the basis of the node credibility.

In the proposed objective approach, with the use of the Pre-Trusted Objects, the experiences of each node are shared with the entire network, so that there is not transitivity, personalization, and asymmetry. Nevertheless, a composability function is still exploited in order to build a unique trustworthiness value. Similarly to the subjective case, I take the mentioned property of composability from past works as well as the concepts of weighted feedback and credibility to estimate trust values. However, the relationship factor is introduced to estimate the credibility of a released feedback and the centrality is exploited to estimate the total trust value. According to this analysis, I can say that the proposed subjective model derives directly from the approaches adopted for trust management in social networks scenario, whereas the objective model takes the basis from the P2P-related approaches and exploits some properties of the social network area.

Table 3.2: Approaches taken from the P2P Studies for the Management of the Reputation Information according to Table 3.1

	Storage	**Sharing**	**Processing**
Subjective	Rater-Based	Part	Weighted Average
Objective	Distributed	Global	Weighted Average

Table 3.3: Properties taken from the Social Networks Studies

	Subjective	**Objective**
Transitivity	X	
Composability	X	X
Personalization	X	
Asymmetry	X	

3.2.3 Basic Trust Elements

Regardless of the particular model implemented, to estimate such reputation I identify seven major factors.

A **feedback system** allows a node p_i to provide an evaluation of the service it has received by the provider p_j. Feedback is represented by f_{ij}^l, which refers to each transaction l and can be expressed either in a binary way ($f_{ij}^l \in \{0,1\}$, i.e., p_i rates 1 if it is satisfied by the service and 0 otherwise), or using values in a continuous range ($f_{ij}^l \in [0,1]$) to evaluate different levels of satisfaction.

The **total number of transactions** between two nodes, indicated by N_{ij}, enables the model to detect if two nodes p_i and p_j have an abnormally high number of transactions.

The **credibility** of node p_i, referred to with C_{ji} (in a subjective way with respect to p_j) or C_i (objective) depending on the model used, represents a key factor in evaluating the information (feedback and trust level) provided by the nodes. This feature can assume values in the range $[0,1]$, with value 1 assigned to nodes with the highest credibility.

The **transaction factor** ω_{ij}^l indicates the relevance of transaction l between p_i and p_j. It is used to discriminate important transactions, $\omega_{ij}^l = 1$, from irrelevant ones, $\omega_{ij}^l = 0$, and can be used as a weight for the feedback. This parameter avoids nodes to build up their trustworthiness with small transactions and then become malicious for an important one. For example, a node builds up its reputation by being honest when providing information about temperature or humidity and then starts to act malicious when asked for a banking transaction. In addition, it can be used to discriminate the functionality of the transactions, so that a node can be trusted only for certain types of service.

To these, I add other two key factors that exploit the main features of the social network among the objects.

One is the **relationship factor** F_{ij} that is related to the type of relation that connects p_i to p_j and represents a unique characteristic of the SIoT. It is useful to either mitigate or enhance the information provide by a friend. Until now, a SIoT implementation does not exist yet, so there are not practical evidences about the weight to assign to each relationship to evaluate the trust. However, the forms of socialization among objects, fully presented in [85], have been devised to represent the human relationships and there are important studies about the connection between relationships and trust. It is a matter of fact that a close friend is more reliable than an acquaintance or a complete stranger [87]. Additionally, many works demonstrate how the relationship and the support from family members are stronger than those received from friends and acquaintances [88], [89]. Moreover, it has been proved from several independent activities that strong ties lead to stronger trust relationship; e.g., in [90] Krackhardt shows how the strong ties imply strong interaction ties for trust and trustworthiness, whereas in [91] Ruef suggests that trust and emotional support are the basic requirements for the creation of strong groups. Based on these considerations, I have assigned different values to F_{ij} on the basis of the relation that connects p_i to p_j (see Table 3.4). As it will be clear in the following higher values have higher impact on the computed trust. This is a possible setting that I use in this Chapter on the basis of the following reasoning (but other values can be used as well if justified by different principles). Between two objects that belong to the same owner and then are linked by an OOR, the relationship factor has been assigned with the highest value. According to the mentioned studies, C-LOR and the C-WOR have been set with only a slightly lower value since are established between domestic objects and objects of the same workplace, respectively. SORs are relationships established between objects that are encountered occasionally (then owned by acquaintances) and for this reason a smaller value is given. Finally, the PORs are the most risky, since they are created between objects of the same brand but that never met and depend only on the model object. If two nodes are tied by two or more relationships, the strongest relation with the highest factor is considered.

The other one is the **notion of centrality** of p_i that is refereed to with R_{ij} (with respect to p_j) in the subjective approach and with R_i in the objective approach. It provides a peculiar information of the social network since if a node has many relationships or is involved in many transactions, it is expected to assume a central role in the network. As described in [92], centrality is “related to group efficiency in problem-solving, perception of leadership and the personal satisfaction of participants”.

Table 3.4: Parameters for Relationship Factor and Computation Capabilities

Relationship Factor

Ownership Object Relationship	OOR	1
Co-Location Object Relationship	C-LOR	0.8
Co-Work Object Relationship	C-WOR	0.8
Social Object Relationship	SOR	0.6
Parental Object Relationship	POR	0.5

Computation Capabilities

Class 1	Smartphone, tablet, Set top box	0.8
Class 2	Sensor, RFID	0.2

Another important characteristics of the members of IoT is also considered. The **computation capability** of an object, namely its intelligence I_j. It is a static characteristic of the objects and does not vary over the time. The rational is that I expect a smart object to have more capabilities to cheat with respect to a "dummy" object, leading to riskier transactions. As a reference example, I can consider the case of an air conditioner that request information about the temperature value in a room. Then, the Service discovery process proposes two possible providers: a smartphone and a sensor. Obviously a smartphone is more powerful than a sensor, increasing the chances to act maliciously; accordingly, trusting the sensor instead of the smartphone leads to a safer choice. However the final decision also depends on the other factors used to compute the trustworthiness. To this, I divide the objects into two different classes, and assign to each class a different value, as shown in Table 3.4: Class1 is assigned to objects with great computational and communication capabilities; to this class belong objects such as smartphones, tablets, vehicle control units, displays, set top boxes, smart video cameras; Class2 is assigned to objects with only sensing capabilities, that is, any object just capable of providing a measure of the environment status and to the RFID-tagged objects.

3.3 Subjective and Objective Models

3.3.1 Subjective Trustworthiness

According to the subjective model, each node stores and manages the feedback needed to calculate the trustworthiness level locally. This is intended to avoid a single point of failure and infringement of the values of trustworthiness. I first describe the scenario where p_i and p_j are adjacent nodes, i.e., where they are linked by a social relationship. Then, I considered the other scenarios where they are farer each other in the social network. As already introduced, T_{ij} is the trustworthiness of p_j seen by p_i and is computed as follows

$$T_{ij} = (1 - \alpha - \beta) R_{ij} + \alpha O_{ij}^{dir} + \beta O_{ij}^{ind} \tag{3.1}$$

Accordingly, p_i computes the trustworthiness of its friends on the basis of their centrality R_{ij}, of its own direct experience O_{ij}^{dir}, and of the opinion O_{ij}^{ind} of the friends in common with node p_j ($\mathcal{K}_{ij}$). All these addends are in the range $[0, 1]$ and the weights are selected so that their sum is equal to 1 to have T_{ij} is in the range $[0, 1]$ as well.

The centrality of p_j with respect to p_i is defined as follows

$$R_{ij} = \frac{|\mathcal{K}_{ij}|}{\left(|\mathcal{N}_i| - 1\right)} \tag{3.2}$$

and represents how much p_j is central in the "life" of p_i and not how much it is considered central for the entire network. This aspect helps with preventing malicious nodes that build up many relationships to have high values of centrality for the entire network. Indeed, if two nodes have a lot of friends in common, this means they have similar evaluation parameters about building relationships. This is even more true if the SIoT considers the possibility to terminate a relationship when a very low value of trustworthiness is reached (which is not implemented now in the SIoT). In this way, only the trustworthy relationships are considered in the computation of the centrality and then it can better highlight nodes similarity.

When p_i needs the trustworthiness of p_j, it checks the last direct transactions and determines its own opinion as described in the following

$$\begin{aligned} O_{ij}^{dir} = & \left(\frac{\log(N_{ij}+1)}{1+\log(N_{ij}+1)}\right)(\gamma O_{ij}^{lon} + (1-\gamma) O_{ij}^{rec}) + \\ & + \left(\frac{1}{1+\log(N_{ij}+1)}\right)(\delta F_{ij} + (1-\delta)(1-I_j)) \end{aligned} \tag{3.3}$$

This equation tells us that even if no transactional history is available between the two nodes ($N_{ij} = 0$), p_i can judge p_j on the basis of the type of relation that links each other and on the computation capabilities. If some interactions already occurred between them, a long-term opinion O^{lon} and a short-term opinion O^{rec} are considered with different weights. Also when N_{ij} is not null the relationship factor and the computation capabilities are considered again, with a weight that decreases as N_{ij} increases. As shown in Figure 3.2, only for Class2 objects, when two nodes have not had any transactions yet, these are the only information available. But when other information becomes available from the transactions between p_i and p_j ($N_{ij} > 0$), the relationship factor and the computation capabilities start to lose their importance and eventually only the opinion built up with past transactions is considered.

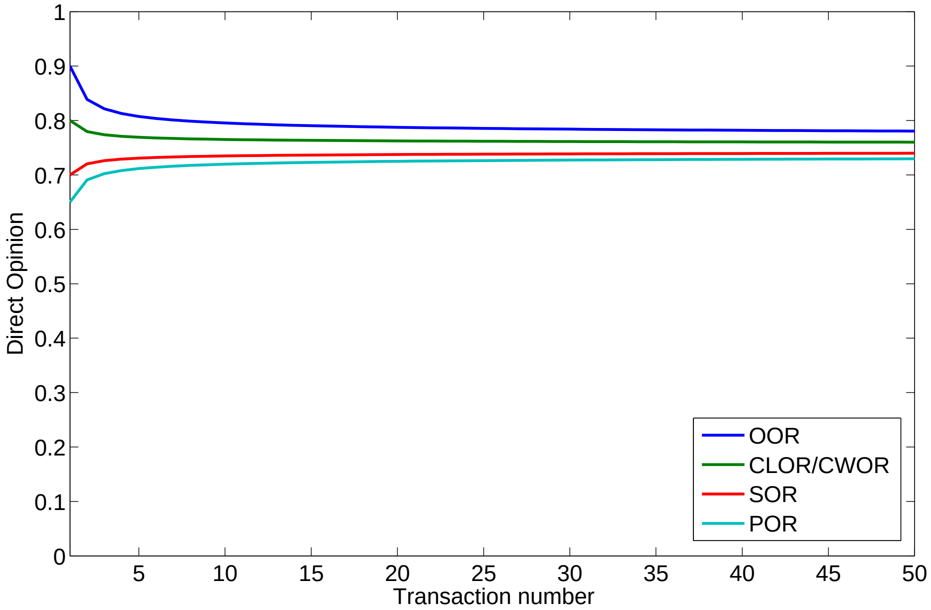


Figure 3.2: Direct opinion behavior according to the number of transaction and for different values of the relationship factor of Class2 objects, with $\gamma = \delta = 0.5$ and $O_{ij}^{lon} = O_{ij}^{rec} = 0.75$

The long and short-term opinions are computed as follows

$$O_{ij}^{lon} = \frac{\sum\limits_{l=1}^{L^{lon}} \omega_{ij}^{l} f_{ij}^{l}}{\sum\limits_{l=1}^{L^{lon}} \omega_{ij}^{l}} \tag{3.4}$$

$$O_{ij}^{rec} = \frac{\sum\limits_{l=1}^{L^{rec}} \omega_{ij}^{l} f_{ij}^{l}}{\sum\limits_{l=1}^{L^{rec}} \omega_{ij}^{l}} \tag{3.5}$$

L^{lon} and L^{rec} represent the lengths of the long-term and short-term opinion temporal windows, respectively ($L^{lon} > L^{rec}$), and l indexes from the latest transactions ($l = 1$) to the oldest one ($l = L^{lon}$). Moreover, the transaction factor ω_{ij} is used to weight the feedback messages. The short-term opinion is useful when evaluating the risk associated with a node, i.e., the possibility for a node to start acting in a malicious way or oscillating around a regime value after building up its reputation. In fact, the long-term opinion is not sensitive enough to suddenly detect this scenario, since it needs a long time to change the accumulated score.

The indirect opinion is expressed as

$$O_{ij}^{ind} = \frac{\sum\limits_{k=1}^{|\mathcal{K}_{ij}|} \left(C_{ik} O_{kj}^{dir}\right)}{\sum\limits_{k=1}^{|\mathcal{K}_{ij}|} C_{ik}} \tag{3.6}$$

where each of the common friends in $\mathcal{K}_{ij}$ gives its own opinion of p_j. In this expression, the credibility values are used to weight the different indirect opinions so that those provided by friends with low credibility impact less than those provided by "good" friends:

$$C_{ik} = \eta O_{ik}^{dir} + (1 - \eta) R_{ik} \tag{3.7}$$

From (3.7) it is possible to note that C_{ik} depends on the direct opinion and on the centrality. Note that the computation of the indirect opinion requires adjacent nodes to exchange information on their direct opinions and list of friends. To reduce the traffic load, it is possible for p_i to request the indirect opinion only to those nodes with a high credibility value.

(3.2) - (3.7) allow us to finally compute the subjective trustworthiness in (3.1). Indeed, for the idea itself of subjective trustworthiness, all the formulas I have shown in this section are not symmetric so that in general $T_{ij} \neq T_{ji}$.

If p_i, that requests the service, and p_j, that provides it, are not adjacent, i.e., are not linked by a direct social relationship, the computation of all the trustworthiness values is carried out by considering the sequence of friends that link indirectly p_i to p_j. The trustworthiness values between no-adjacent nodes T'_{ij} is computed as follows

$$T'_{ij} = \prod_{a,b:p_{ij}^a p_{ij}^b \in \mathcal{R}_{ij}} T_{ab} \tag{3.8}$$

The requester asks for the trust value of the provider through the route discovered by the Service discovery process (bold route in Figure 3.3(a)) and the values are obtained through word of mouth from requester to provider making use of the social relationship described above (green route in Figure 3.3(b)). Note that in (3.8) I am not considering the direct experiences of p_i with p_j. The reason is that in the subjective model, each node stores and manages the feedback and all the information needed to calculate the trustworthiness level of only adjacent nodes. If nodes used (3.1) to compute the trustworthiness of nodes that are not adjacent, they would need to store a huge amount of data, resulting in a burden on their memory, computation capabilities and battery.

At the end of each transaction, p_i assigns a feedback f_{ij}^l to the service received; in the case p_i and p_j are adjacent, p_i directly assigns this feedback to p_j. Moreover, p_i computes the

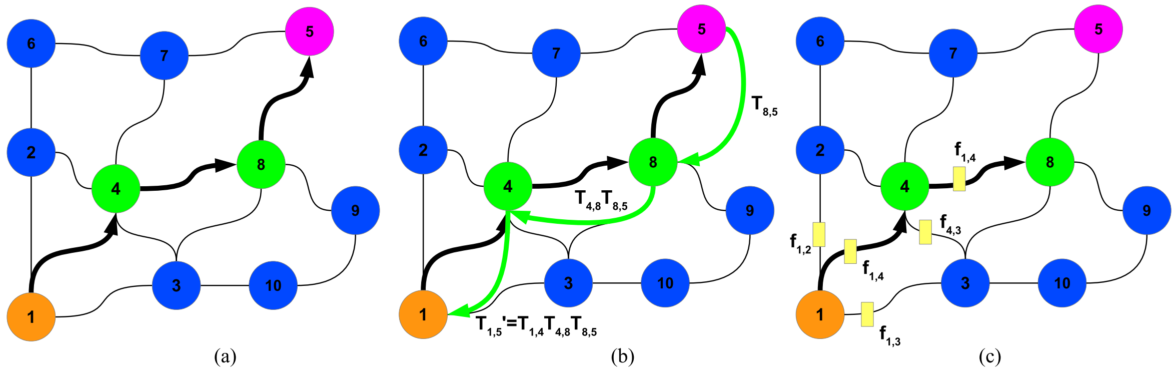


Figure 3.3: Trustworthiness evaluation process for the no-adjacent nodes p_1 and p_5: request of the trust value for a distant node (a), computation of the trust level by multiplying the trust level among adjacent nodes (b) and releasing feedback to the nodes involved in the transaction (c)

feedback to be assigned to the friends in $\mathcal{K}_{ij}$ that have contributed to the computation of the trustworthiness by providing O^{dir}_{ik}, so as to reward/penalize them for their advice. According to (3.9), if a node gave a positive opinion, it receives the same feedback as the provider, namely a positive feedback if the transaction was satisfactory, $f^l_{ij} \geq 0.5$, and a negative one otherwise, $f^l_{ij} < 0.5$; instead, if p_k gave a negative opinion, then it receives a negative feedback if the transaction was satisfactory and a positive one otherwise. Note that the feedback generated by p_i are stored locally and used for future trust evaluations.

$$f^l_{ik} = \begin{cases} f^l_{ij} & \text{if } O^{dir}_{kj} \geq 0.5 \\ 1 - f^l_{ij} & \text{if } O^{dir}_{kj} < 0.5 \end{cases} \tag{3.9}$$

In the case there is more than one degree of separation, the node p_i assigns a feedback to the adjacent node along the path to the provider. The same assignment is then performed by all the nodes along the path to the provider, unless a node with a low credibility is found (in this case the process is interrupted). With reference to Figure 3.3(c), p_1 stores the feedback about p_4 and nodes in $\mathcal{K}_{1,4}$ (i.e., p_2 and p_3) locally. Then it propagates the feedback to p_4, which accepts it only if the credibility of p_1 is high (greater than a predefined threshold). p_4 utilizes it to rate p_8, its last intermediate, and their common friends, in this case only p_3. Then p_4 propagates the feedback to p_8 and so on up to the provider of the service.

According to this approach, negative feedback is given not only to malicious nodes performing maliciously, but also to malicious nodes that give false references and even to nodes that do not act maliciously but are connected to portions of the network wich are not reliable.

3.3.2 Objective Trustworthiness

According to this approach, the values needed to compute the trustworthiness of a node are stored in a distributed system making use of a DHT structure on the network. Several DHT systems are available for this purpose, such as CAN [93], Chord [94], Pastry [95]. In the following, I refer to the Chord system since I have statistics to estimate the performance and open-source tools are commonly available for implementation and simulation.

A DHT system is based on an abstract keyspace, where each node is responsible for a set of keys. An overlay network then connects the nodes, allowing them to find the owner of any given key in the keyspace. To store a file, with a given filename and data, a key for the filename is generated through a hash function (SHA-1 with Chord) and the data and the key are sent to the node responsible for that key. If a node wants to retrieve the data, it first generates the key from the filename and then sends to the DHT a request for the node that holds the data with that key. Chord is a DHT structure that provides good scalability with respect to the network size, since the overhead for information retrieval scales as $O(logM)$ [94], where M are the nodes in the network. It is also very robust to the phenomenon of high churn-rate, i.e., to those nodes moving in and out of the network frequently. This feature is even more important in the IoT settings where the nodes are usually characterized by a more ephemeral connectivity with respect to the scenario of file-sharing.

In my scenario every node can query the DHT to retrieve the trustworthiness value of every other node in the network. In Figure 3.4, p_1 queries the DHT to retrieve information about the route discovered by the Service discovery process, namely p_4, p_8, and p_5. To avoid the problem of distributed storage approach where malicious nodes are selected as storage nodes, only special nodes, that I call *Pre-Trusted Objects* (PTOs), are able to store the data about feedback or trustworthiness values. PTOs do not provide any service and are integrated in the architecture; their number is decided based on the number of nodes in the SIoT, so that there is always a PTO available to manage the data. In Figure 3.4, p_1 sends the feedback about the transaction to the PTO, that has the role to calculate the new trustworthiness values of the nodes involved in the last transaction, taking into account the source of the feedback to avoid fake feedback. Then, through the DHT, it generates the key associated with the data and stores it in the node responsible for that key, that is p_7 in this case.

When p_i needs to know the latest trustworthiness value of p_j, it queries the DHT to retrieve it. In this case, there are no direct and indirect opinions since all the nodes can read the trustworthiness value of all other nodes in the DHT, and the trustworthiness is expressed as

$$T_j = (1 - \alpha - \beta)R_j + \alpha O_j^{lon} + \beta O_j^{rec} \tag{3.10}$$

Centrality is now based on the idea that a node is central in the network if it is involved in many transactions, as expressed in the following

$$R_j = \frac{(A_j + H_j)}{(Q_j + A_j + H_j)} \tag{3.11}$$

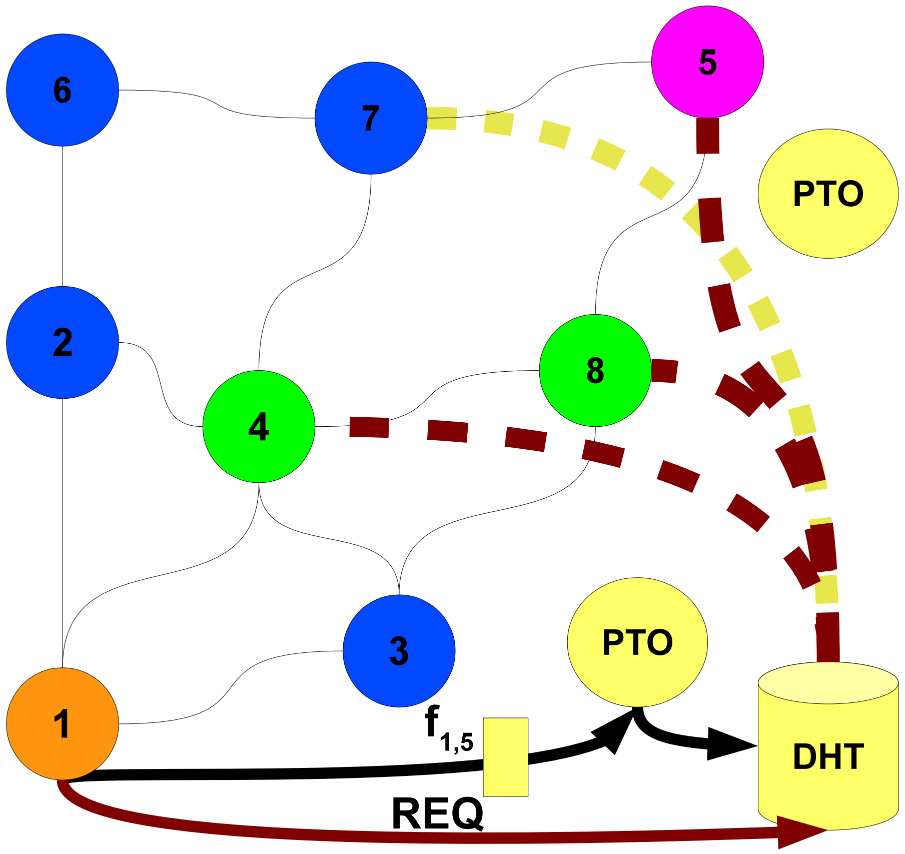


Figure 3.4: Objective case: request and store of the trust value

where Q_j is the number of times p_j requested a service, A_j is the number of times it acted as an intermediate node in a transaction, and H_j counts how many times it is the provider of a service. A node is considered central if it takes part actively to the SIoT, as either intermediate or provider of the service, in many transactions with respect to all its transactions.

Furthermore, in this approach, the short and long-term opinions are computed considering the feedback received from all the nodes that interacted with p_j

$$O_j^{lon} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L^{lon}} C_{ij}\omega_{ij}^{l} f_{ij}^{l}}{\sum_{i=1}^{M}\sum_{l=1}^{L^{lon}} C_{ij}\omega_{ij}^{l}} \qquad (3.12)$$

$$O_j^{rec} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L^{rec}} C_{ij}\omega_{ij}^{l} f_{ij}^{l}}{\sum_{i=1}^{M}\sum_{l=1}^{L^{rec}} C_{ij}\omega_{ij}^{l}} \qquad (3.13)$$

To limit the possibility of malicious nodes giving false feedback to subvert the reputation system, every feedback is weighted with the credibility of the node that provides it in addition to the transaction factor. The credibility is defined as follows

$$C_{ij} = \frac{(1-\gamma-\delta)T_i + \gamma(1-F_{ij}) + \delta(1-I_j)}{1+\log{(N_{ij}+1)}} \tag{3.14}$$

In this way, nodes with strong relations (i.e., with a small value of the relationship factor), with high computation capabilities or nodes that have a high number of transactions between them, receive a lower credibility. Indeed, this is motivated by the opinion that nodes that fall in this situation (strong relationship links, high intelligence and many interactions) are potential candidates to collusive malicious behavior.

3.4 Experimental Evaluation

This Section analyses the performance of the proposed models through simulations. Due to the lack of real data concerning some aspects of objects behavior, a complete theoretical analysis of the models performance cannot be achieved. For this reason in Appendix A, I provide a first theoretical analysis for the subjective model case.

3.4.1 Simulation Setup

To conduct my performance analysis, I needed mobility traces of a large number of objects. I resorted on the mobility model called *Small World In Motion* (SWIM) [40], [6] to generate the synthetic data and on the real dataset of the location-based online social network Brightkite obtained from the Stanford Large Network Dataset Collection [52].

The outputs of the SWIM model and the Brightkite dataset are traces of the position of humans. In this Chapter, instead, I am interested in the mobility of things. Accordingly, I have extended them as follows. I assume that each user owns a set of things that are connected to the SIoT and that during any movement the user carries half of these objects and leaves the others at home. I decided to run the experiments with about 800 nodes, considering that each person owns an average of 7 objects. Objects that stay at home create co-location relationships. Every node is produced by a specific company and is characterized by a model ID; this information is used to build the parental object relationships. The other relationships are created on the basis of the objects (and then owners) movements, mainly taking into account how often objects meet and for how long and where. All the details about the establishment of these social relationships are provided in [26] and [85]. Two different behaviors can be considered in a social network: one is always benevolent and cooperative so that I call the relevant node social nodes. The other one is a strategic behavior corresponding to an opportunistic participant who cheats whenever it is advantageous for it to do so. I call it malicious node and it gives bad services, false references, and false feedback. Its behavior is described by Algorithm 1. Accordingly, it only acts maliciously with objects that it meets occasionally or it has never met, in the same way a person behaves benevolent with close friends and family members and acts maliciously with everyone else (if she/he is malicious). Note that this object behavior is inherited from the owner that authorizes the object interactions according to her/his profile. However, this is true only if the object has enough computational compatibilities to distinguish one relationships from another; otherwise it acts maliciously with everyone. The percentage of malicious nodes is denoted by mp and it is set by default to 25%; I denote with mr the percentage of time in which these nodes behave maliciously (by default $mr = 100\%$).

Algorithm 1 Malicious node behavior

```
if malicious node belongs to Class 1 then
    switch (relationship factor)
    case OOR, C-LOR, C-WOR:
        act benevolent
    case SOR:
        act benevolent only with close friends
    case POR:
        act maliciously
    default:
        act maliciously
    end switch
end if
if malicious node belongs to Class 2 then
    act malicious with everyone
end if
```

At the start of each transaction, the simulator chooses randomly the node requesting the service and randomly select the nodes that can provide the service, corresponding to a percentage res of the total number of SIoT nodes (by default $res = 5\%$). The malicious node can then be the one requesting the service, the one providing the service or, only in the subjective approach, the one providing its opinion about another node. In the first case, it provides negative feedback to every node involved in the transaction; in the second case, it provides the wrong service and should then received a negative feedback; finally, in the third case, it provides a negative opinion about the other nodes.

Table 3.5 shows the simulation parameters of the system, and the different weights used with the two approaches. For simplicity, I decided to use a binary feedback system to rate the other nodes according to whether the transaction was satisfactory. For the same reason, I considered all the transactions equally important and I set the transaction factor to 1; finally, each

Table 3.5: Simulation parameters

General parameters

	Parameter	Description	Default
Community setting	M	# of nodes in the SIoT	800
	mp	% of malicious nodes	25 %
	mr	% of transactions a malicious nodes acts malicious	100 %
	res	% of nodes who respond to a transaction request	5 %
Trust computation	L_{lon}	# of transaction in the long-term opinion	50
	L_{rec}	# of transaction in the short-term opinion	5
	n	# of run for each experiment	4

Subjective model parameters

Parameter	Description	Value
α	weight of the direct opinion	0.4
β	weight of the indirect opinion	0.3
γ	weight of the long-term opinion	0.5
δ	weight of the relationship factor	0.5
η	weight of the direct opinion in the credibility	0.7

Objective model parameters

Parameter	Description	Value
α	weight of the long-term opinion	0.4
β	weight of the short-term opinion	0.4
γ	weight of the relationship factor in the credibility	0.3
δ	weight of the intelligence in the credibility	0.3

object randomly belongs to one of the computation capabilities classes. To find the optimal system setting I analyzed the models response at varying parameter values. The optimal configuration is provided in Table 3.5. To show the system response at different settings, Table 3.6 displays the transaction success rate when the system has reached the steady-state using the SWIM data. Each row refers to the change of only one parameter while the others keep the optimal setting. As expected, in the subjective approach, the direct opinion has a more impact

Table 3.6: Parameters setting

Subjective model values

$\alpha = 0.1$	0.88	$\alpha = 0.4$	0.93	$\alpha = 0.7$	0.91
$\beta = 0.1$	0.92	$\beta = 0.3$	0.94	$\beta = 0.6$	0.93
$\gamma = 0.2$	0.91	$\gamma = 0.5$	0.93	$\gamma = 0.8$	0.92
$\delta = 0.2$	0.9	$\delta = 0.5$	0.94	$\delta = 0.8$	0.92
$\eta = 0.1$	0.91	$\eta = 0.4$	0.93	$\eta = 0.7$	0.94

Objective model values

$\alpha = 0.2$	0.89	$\alpha = 0.4$	0.95	$\alpha = 0.6$	0.93
$\beta = 0.2$	0.91	$\beta = 0.4$	0.94	$\beta = 0.6$	0.93
$\gamma = 0.1$	0.92	$\gamma = 0.3$	0.95	$\gamma = 0.7$	0.93
$\delta = 0.1$	0.93	$\delta = 0.3$	0.94	$\delta = 0.7$	0.91

than the indirect opinion because it is affected by a node own experience, whereas in the objective approach, the most important parameter is the long-term opinion because it takes into account the story of the node. In both the approaches the centrality is the factor that less affects the performance, since it is a slow time variant factor.

After a node chooses the provider of the service on the basis of the highest computed trustworthiness level, it sends to it the service request. Depending on how the SIoT model is implemented, the service can be delivered either through the nodes that discovered the service, i.e., the social network is also used to transmit the service requests and the responses on top of the existing transport network (overlay structure) or directly relying on the beneath communication network (non-overlay structure). In the first case, a malicious node can interfere with the deliver of the service even if it is in the route from p_i to p_j since it is asked to forward the service request to p_j and the response back p_i. In the latter case, a malicious node can alter the service only if it is the provider.

3.4.2 Transaction Success Rate

In this section I present the results for the objective and subjective approaches in the case overlay network is used or not used. I compare the performance of the proposed models with those of the *Dynamic Trust Computation of the Trust Value Measure* (TVM/DTC) proposed for P2P networks, described in [71]. It relies on a reputation system, which defines a recursive function that uses the trust value of a peer as its feedback credibility measure. I also selected a trust management algorithm for social networks, named TidalTrust [96], which infers trust relationships between people that do not have direct connections through their indirect links. These comparisons are aimed at analyzing the improvements I obtain with respect to the state of the art in the

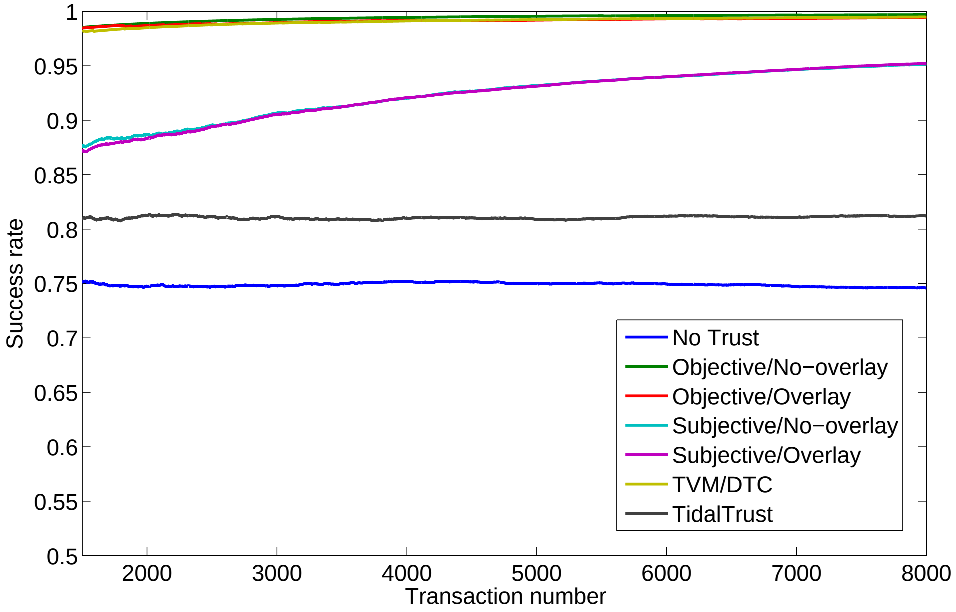


Figure 3.5: Transaction success rate in the SWIM scenario versus the total number of performed transactions with Class 2 objects

specific reference SIoT scenario. I also show the case in which a trust model is not used.

Figure 3.5 shows the success rate when the malicious nodes only belong to Class2 in the SWIM scenarios. It is possible to observe that the objective model has a faster convergence and presents an higher success rate. This happens since in the objective case, the feedback about a transaction is immediately available to the entire community bringing to a faster converge. Indeed, this model allows for isolating the malicious nodes as fast as 4000 transactions are reached (success rate equal to 99,9%). The subjective approach has indeed a slower transitory, since every node has to build up its own opinion. Still, it's important to point out that this scenario is a very basic one. Malicious Class2 objects are very easy to be identified since they don't behave differently according to the service client, so I can say that this scenario is typical of the P2P networks and then it is favorable for the objective model. Accordingly, the TVM/DTC algorithm presents performance comparable to my objective approach. Differently, Tidal Trust chooses the providers with a weak criteria since a Class2 object acts maliciously with everyone; nevertheless, with respect to the case where no trust algorithm is used, TidalTrust can still achieve significant success rates. Note that since the feedback system is not adopted, the performance don't improve as the number of transactions increases. Since this scenario is a very simplicistic one, I can not observe big differences between the overlay and non-overlay structure; they will be discussed in further simulations.

I now consider the same scenario but with Class1 malicious objects, which can modify their behavior based on the social relationships. Results are shown in Figure 3.6 for the SWIM

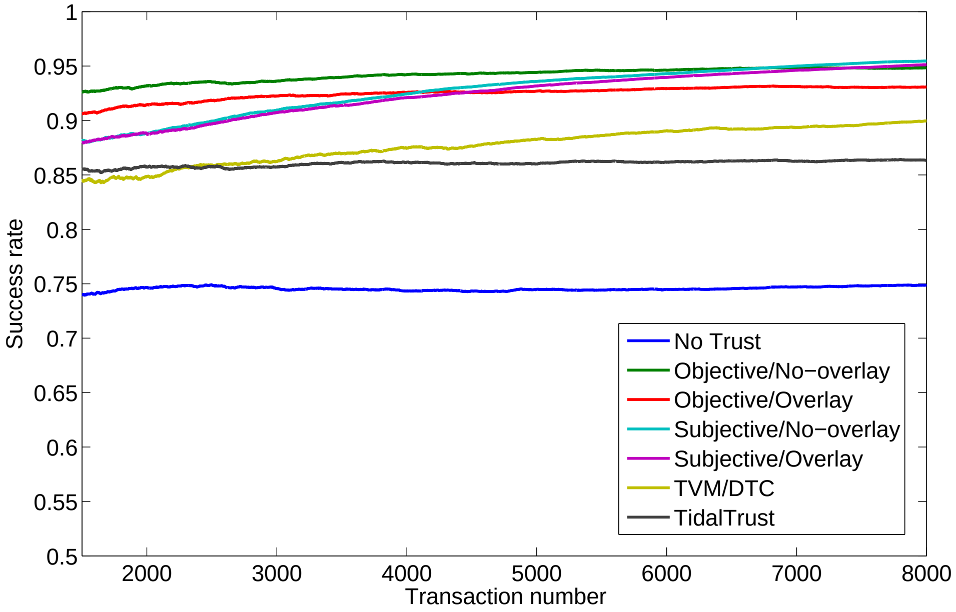


Figure 3.6: Transaction success rate in the SWIM scenario versus the total number of performed transactions with Class 1 objects

scenarios. Still it can be noted as the objective approach converges faster and reaches its steady-state after around 4000 transactions. However, in this case the node trustworthiness is global and mixes the opinions of both the nodes with which it behaved maliciously and the nodes with which it behaved benevolent. This is a drawback only partially addressed by using the relationship factor (see (3.14)), so that is more difficult to isolate the malicious nodes. With the subjective model each node stores its own trustworthiness data and has its own opinion about the network so that it is clearly more robust towards Class1 malicious objects behavior. As also discussed previously, this approach needs more time to converge but it manages to outperform the objective model after 7000 transactions. With respect to the scenario with Class2 objects, the steady-state performance is slightly worse; this is due to the indirect opinion (see (3.6)) a node receives from its neighbours, since all the rest of the key data is stored locally. This information depends on the relation between the reference nodes and the service provider, so that can be either positive or negative and can confuse the service requester; however this information is weighted with the credibility of the source node (see (3.7)), which depends only on the experience of the node that is performing the trustworthiness evaluation.

Another key observation related to the Class1 scenario is that the structure chosen to deliver the service influences the performance. In particular, the use of the overlay structure, where the social network is also used to transmit the service requests and responses, leads to lower performance; indeed, a malicious node can interfere with the delivery of the service because it is in the route from the requester to the provider and it is asked to forward the message.

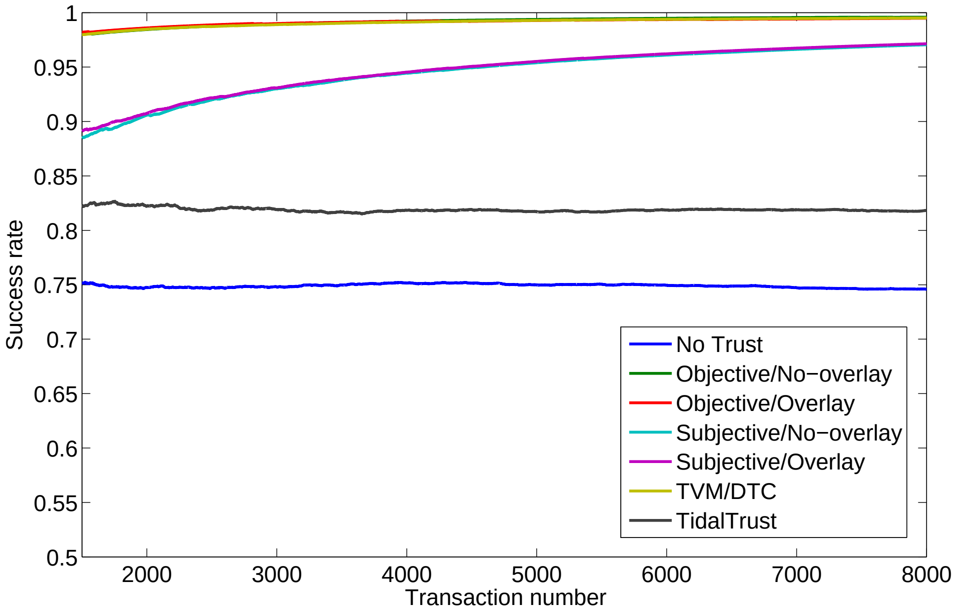


Figure 3.7: Transaction success rate in the Brightkite scenario versus the total number of performed transactions with Class 2 objects

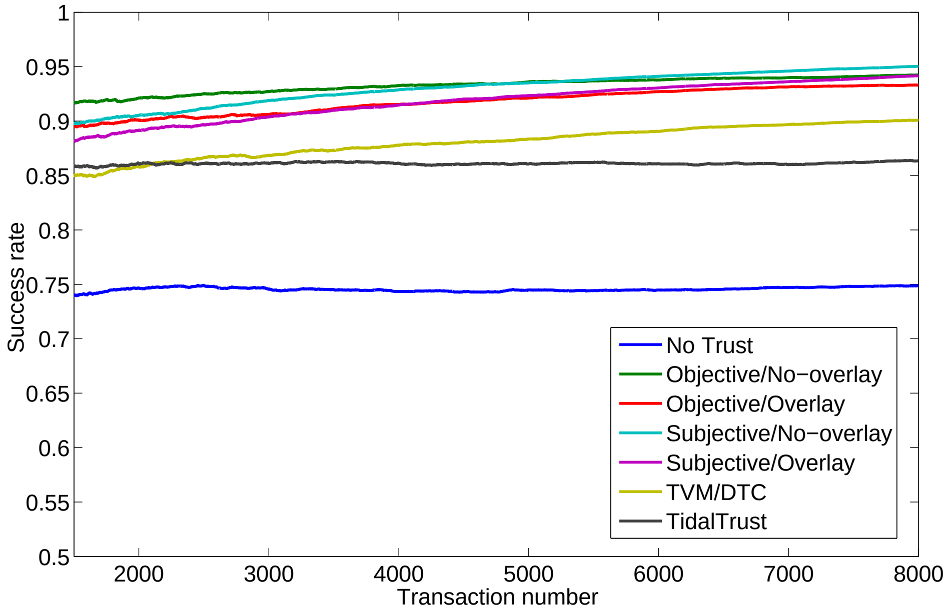


Figure 3.8: Transaction success rate in the Brightkite scenario versus the total number of performed transactions with Class 1 objects

This cannot happen in the non-overlay structure, where a malicious node can alter the service response only when acting as final provider.

Additionally, it is important to remark that adding the social behavior in the malicious nodes leads to an increase in the TidalTrust performance by almost 5% and a decrease in the TVM/DTC performance by almost 10%. However, it is clear that in the specific SIoT scenario, the well-known techniques for trustworthiness computation studied for either P2P or social networks are not enough to obtain a reliable system, and both my models, subjective and objective, using or not the overlay structure, can outperform these approaches.

Figs. 3.7 and 3.8 show the success rate in the Brightkite scenario when malicious nodes belong to Class2 and Class1, respectively. One of the main differences that can be noted by comparing the results obtained with the two dataset is that in the Brightkite scenario the subjective approach performs slight better than in the SWIM scenario. This is due to the fact that Brightkite is characterized by a shorter network diameter on average with respect to the social graphs generated with SWIM (3 hops instead of 4 hops). Accordingly, in Brightkite every node has more relationships with respect to the SWIM case, which are then exploited by the subjective model that strongly relies on objects direct experience.

I now want to analyze the results at varying percentage of the malicious nodes. Figs. 3.9 and 3.10 refer to the Brightkite scenario with the non-overlay structure and using the subjective and objective approaches, respectively. I note how the subjective approach always converges even with 70% of malicious nodes, since every node has its own vision of the network based on its own experiences. However, the accuracy of this approach decreases, since there is the need for more feedback messages to be collected to cope with the bad recommendations received. Instead, the objective approach is much more sensible to the malicious concentration since every node shares its opinion with the others: with 50% of malicious nodes in the network the performance reaches 0.7; if I further increase the number of malicious nodes, the performance dramatically drops since the opinion of a node is deeply influenced by malicious feedback with appropriate compensation from benevolent ones.

3.4.3 Dynamic Behavior

The focus of this set of experiments is to analyze how the proposed approaches work with three different dynamic behaviors of the nodes. In a first scenario, a node builds its reputation and then starts milking it; in a second scenario, a node tries to improve its reputation after having milked it; in a third scenario, the node oscillates between milking and building its reputation. Since I have already analyzed how my algorithms responds to false feedback, I now consider only the malicious behaviors without taking into account nodes providing dishonest feedback. The considered behaviors are independent from the particular networking structure adopted (whether it is overlay or not) or the scenario implemented (whether it is the SWIM or Brightkite scenario) so I only consider the differences between the two subjective and objective models. Figure 3.11 shows the computed trust value of a node that is milking its reputation; I can observe that, thanks to the short-term window, both algorithms are able to fast adapt to the change in

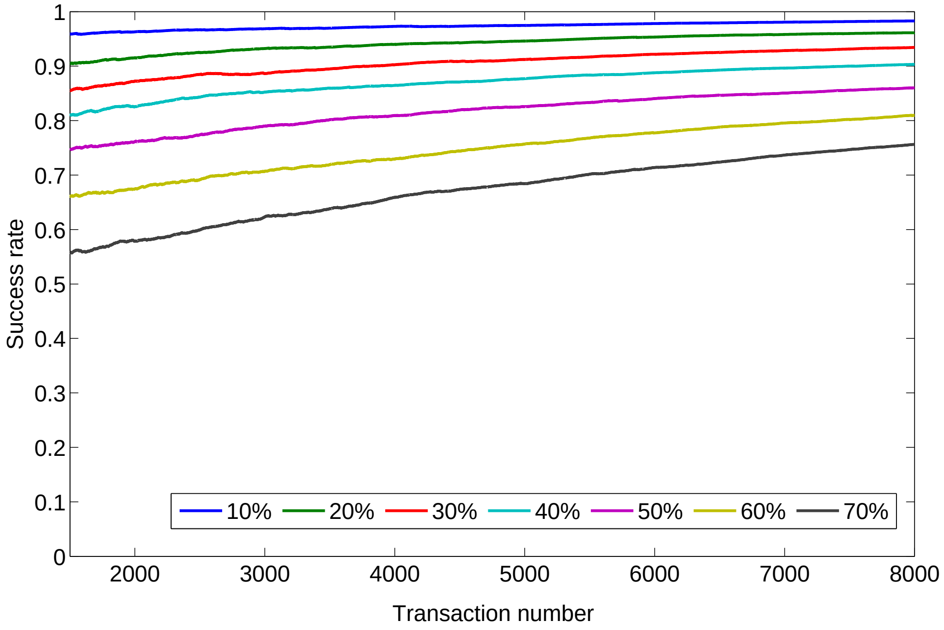


Figure 3.9: Transaction success rate of the subjective approach in the Brightkite scenario with a non-overlay structure at increasing values of mp

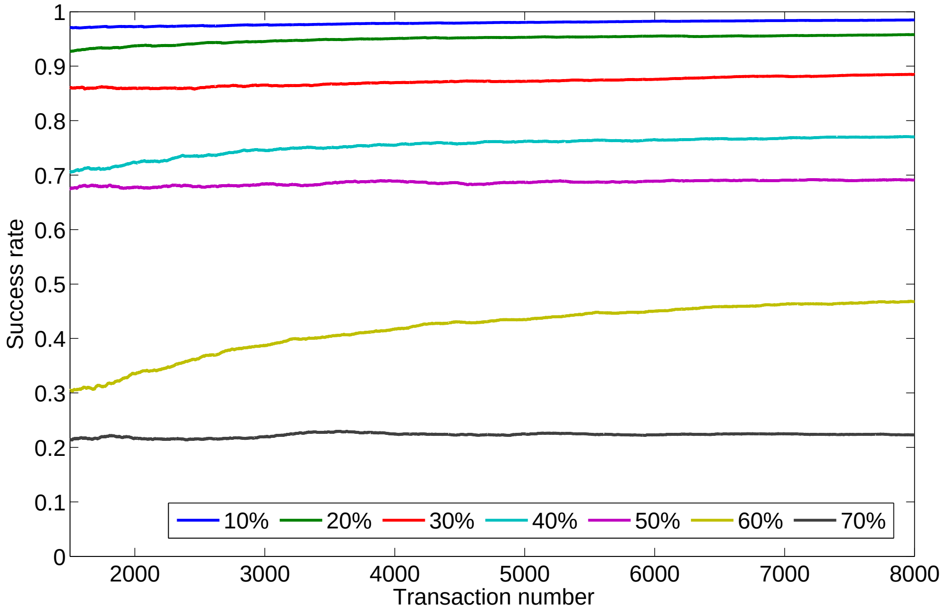


Figure 3.10: Transaction success rate of the objective approach in the Brightkite scenario with a non-overlay structure at increasing values of mp

the node behavior. The subjective approach is slightly slower since it has to mediate its opinion with that of its friends, that eventually still trust the malicious node. Similar considerations can be done for a node who is building its reputation as shown in Figure 3.12, and for a node with an oscillating behavior in Figure 3.13. These results clearly show how my approaches can cope with dynamic behaviors.

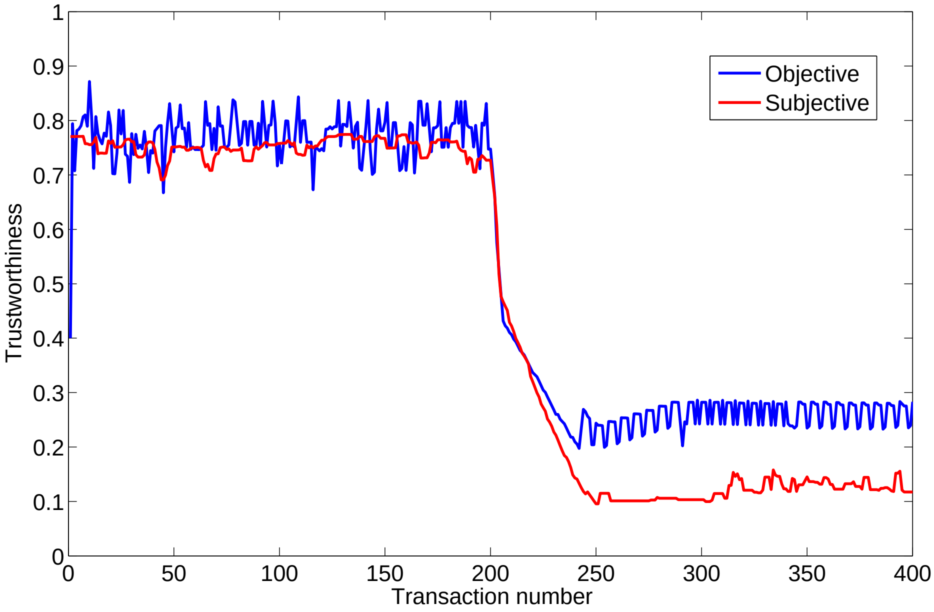


Figure 3.11: Dynamic behavior: milking reputation

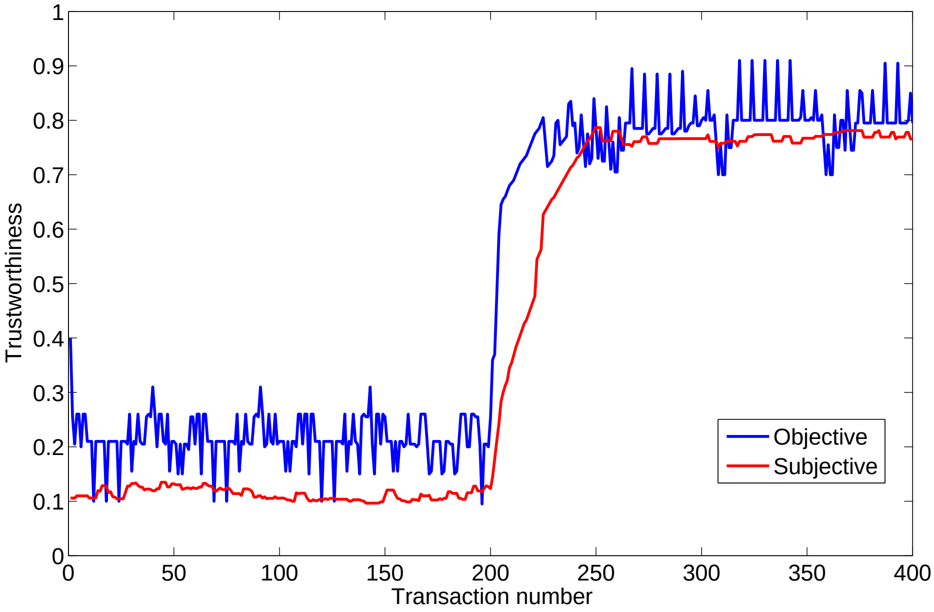


Figure 3.12: Dynamic behavior: building reputation

3.4.4 Runtime Overhead

I now analyze the runtime overhead for the SWIM scenario and how it scales with respect to the number of nodes. A comparison between the two approaches can be observed in Figure 3.14 for different number of nodes, 100 transactions and 10 different providers. In the objective approach, I consider the use of both the structure, overlay and non overlay; from [97], I extract the average number of queries to the lookup table in the DHT. Indeed in the overlay structure,

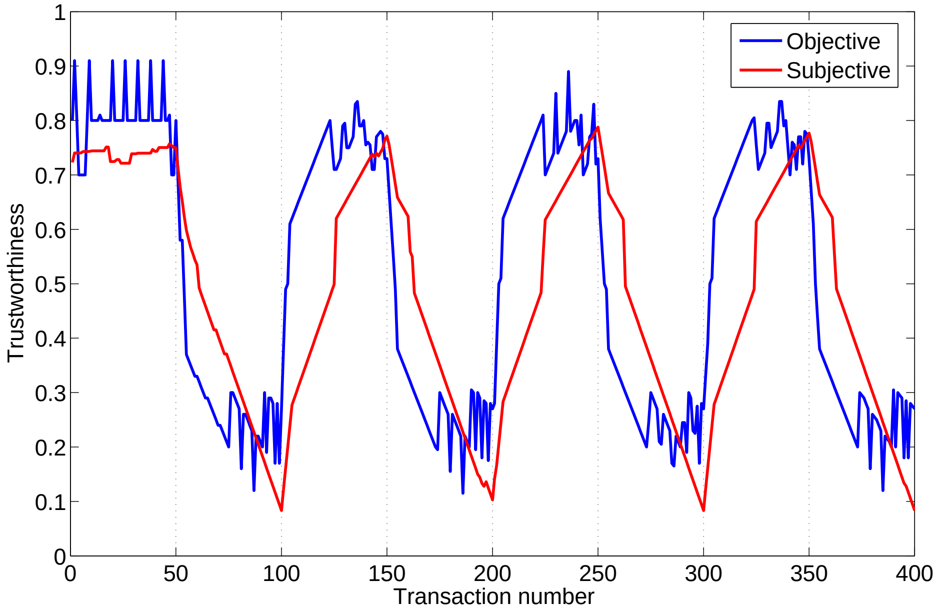


Figure 3.13: Dynamic behavior: oscillating reputation

a node needs to query the DHT for every node in the route while in the non-overlay structure only one query is needed so that the runtime overhead is clearly lower in the latter case. In the subjective approach, I analyze the differences between flooding the messages to all the nodes or using a threshold, where the benefits of using a threshold are clearly visible. The comparison shows how the objective algorithm outperforms the subjective one in terms of runtime overhead. However, it is important to deeply analyze the overhead in the subjective approach; in fact, we have to consider that service discovery and trustworthiness computation can be carried out at the same time. Moreover, in the simulations, I have considered the service providers to be uniformly distributed over the network, while it has been proved that friends share similar interests (bringing to the homophily phenomenon [98]), so that it is highly probable to find a service in the friends list. This would reduce the runtime overhead in the subjective approach.

If I consider the overhead due to the released feedback (Figure 3.15), we can observe how the subjective approach performs better than the objective one. This happens because the subjective approach stores locally the feedback of the nodes involved in the transaction and releases feedback to the intermediary only when there is more than one degree of separation between requester and provider. The objective approach, instead, releases the feedback to the PTO for every node involved in the transaction.

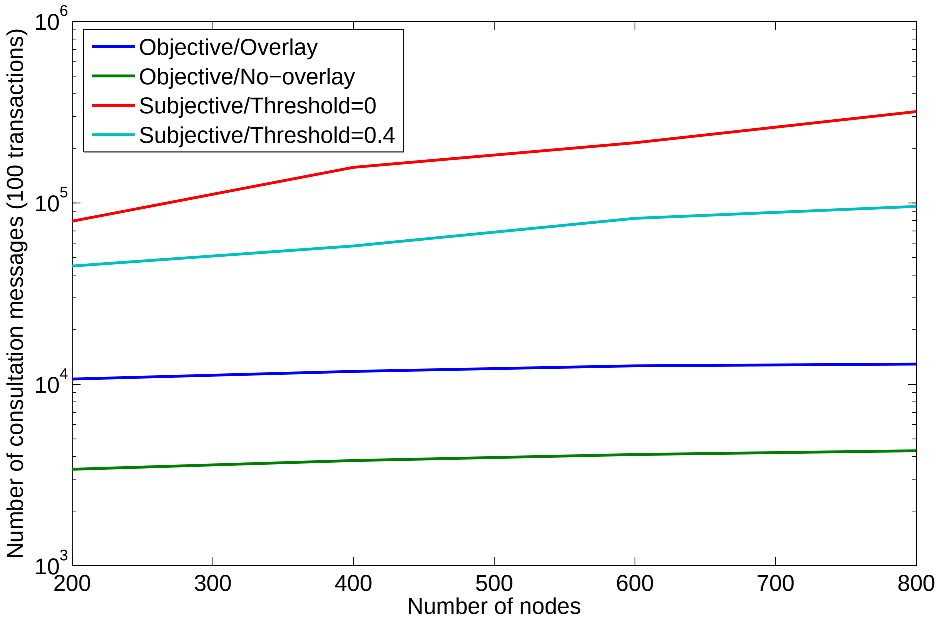


Figure 3.14: Comparison of the two models in terms of computation overhead

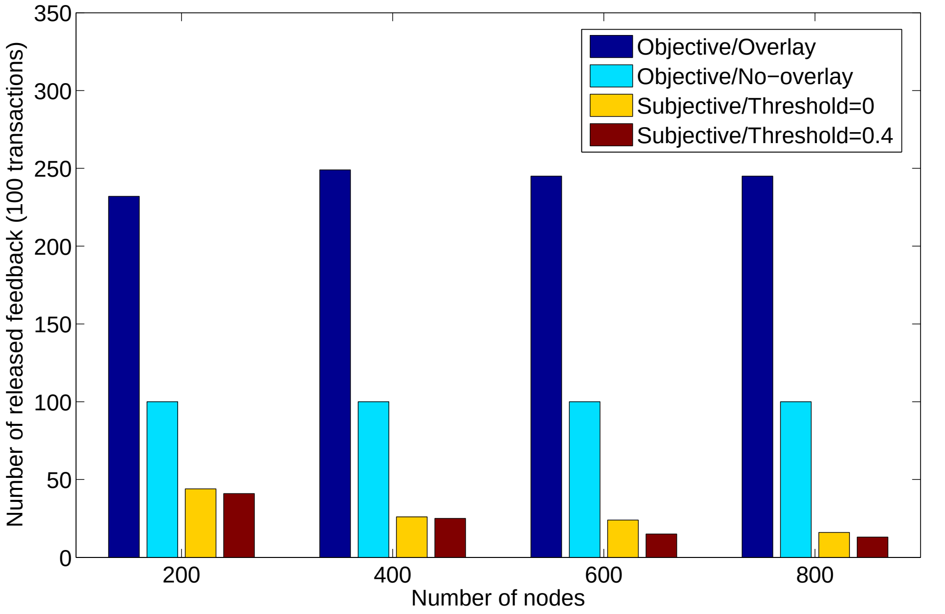


Figure 3.15: Released feedback messages

Chapter 4

Implementation of an experimental platform

The aim of this Chapter is to present my implementation of a SIoT platform. I begin by analyzing the major IoT implementations, pointing out their common characteristics that could be re-used for my goal.

In Section 4.1 I describes the guidelines that drove my implementation with respect to the already existing platforms, while in Section 4.2 I show the major functionalities of the proposed system: how to register a new social object to the platform, how the system manages the creation of new relationships, and how the devices create groups of members with similar characteristics. Section 4.3 illustrates some possible applications that can use my platform, where objects can create their own relationships and provide simple services to the users.

4.1 Background

In this section, I highlight some requirements which have been considered for the system specification, including a reasoned analysis of the state of the art of IoT implementations and the description of the solutions and choices which have driven the design and implementation of my system architecture.

4.1.1 IoT platforms

From the analysis of the IoT scenario [99], I identified around 10 different platforms. All these systems have common characteristics, especially the followings:

- the objects use a *HyperText Transfer Protocol* (HTTP) protocol to send and receive data. This choice allows a high interoperability among the different platforms;

- an intermediary server is used. The objects do not communicate directly with each other;
- every object has a “data point” associated with it on the server-side to keep track of the data sent;
- the methods POST and GET are used to send and request data;
- a tag is assigned to every data point;
- data point discovery is performed using tags through an internal search engine;
- the system identifies every object with its *Application Programming Interface* (API) key.

The need to store and process data from a multitude of sensors has led the implementation of the IoT to the creation of platforms with the main purpose of data logging. Among the European platforms, Cosm (former Pachube) [100] is one of the biggest ones: it has been presented as a platform to store and redistribute real-time data, freely usable, which manages millions of devices per day. One of the most dramatic demonstrations of Cosm’s potential was the visualizations of data that showed the radiation levels around the Japan and especially near the nuclear reactor in 2011.

Nimbits [101] is an open source java web application, built on Google App Engine, which can provide complex functionalities such as email alert, math calculations and complex queries on API Wolfram Alpha, in addition to simply storing and processing data. Every user can define data points and use them to share several kinds of data. The integration with Twitter, Facebook and Google+ allows to manage your data points, to share sensor diagrams, activate alarms, etc.

Paraimpu [102] is a Web of Things platform, which allows people to connect together sensors, actuators and other web applications, taking care to forward data among the objects [102]. Moreover, through the integration with Twitter, it is possible for a user to obtain and use data from a friend.

Finally, ThingSpeak [103], was founded as an open source branch of IoBridge [104] and shares with it the main features. ThingSpeak is an IoT application that allows users to store and retrieve data from objects through HTTP communications. Moreover, it enables the creation of several kinds of application, involving different pairs of API keys, such as GPS tracking and data logging. ThingSpeak API enables to perform averaging, summing, rounding and time-scaling. In addition, this platform allows for integrating several data representation, such as *JavaScript Object Notation* (JSON), *eXtensible Markup Language* (XML) and *Comma-Separated Values* (CSV).

However, none of these platforms foresees some kinds of social relationships between objects. Indeed, even if integration with main human social networks is allowed, the objects can not communicate independently.

4.1.2 RESTful vs WS-*

Web Services (WSs) are widely used in several *Information Technology* (IT) systems; they can be defined as development techniques for interoperable and distributed applications that make use of standard protocol such as HTTP [105]. WSs can be classified in two major categories: WS-* and *REpresentational State Transfer* (REST).

The former declares its functionalities and interfaces using a *Web Services Description Language* (WDSL) file. Communications are encapsulated using *Simple Object Access Protocol* (SOAP) usually using the HTTP protocol. WS-* is mainly used in enterprise applications where interoperability with other applications is not an important issue; moreover, for applications with advanced security requirements [106] or for WSNs solutions [107] it can provide good results.

The RESTful architecture is based on the concept of resources as representation of the objects, that are uniquely identified through URIs. Through the HTTP protocol is then possible to obtain, delete, post or update object information using a given method (GET, DELETE, POST, PUT). The payload of the message can be incapsulated in a negotiated format such as XML or JSON. A RESTful architecture is then lightweight and scalable and then fits perfectly with the principles and the current protocols of the Internet.

To maintain interoperability with existent IoT platforms and future implementations, I decide to adopt a RESTful approach, and then every entity in the SIoT is represented using JSON, XML or CSV format.

4.2 Platform Implementation

In this section I describe my implementation of the SIoT platform, pointing out its major functionalities required to run simple applications.

4.2.1 Server Architecture

As presented in [85] and in Chapter 1, Figure 4.1 shows the main components of the platform. The network layer is needed in order to transfer data across different networks, while the core of the proposed platform is represented by the application layer, where IoT applications and middleware functionalities are developed and which consists of three sublayers. Not all the functionalities have been implemented, since the platform is still in its infancy and the developed vertical applications are very specific.

The *Base Sublayer* includes the database for the storage and the management of different data types, such as temperature, latitude, longitude and humidity. Objects can memorize up to 16 different data fields; the first 12 fields have a fixed type, whereas the others 4 are reserved for future uses. For example, field 1 is always used to track temperature data, field 3 is used to track voltage data and so on. The last four fields are left for uses decided by client application developers.

The *Component Sublayer* implements the functionalities of: objects profiling, which is needed in order to configure information about the objects; ID management, which assigns a unique ID to every object in order to identify them; Owner Control (OC), which enables the users to specify the objects' behavior; and the Relationship Management (RM), which has to create and manage the relationships of every object.

The other functionalities, i.e. the Service Discovery and Composition (SD and SC) and the Trustworthiness Management (TM) ([108] and [109]), are not implemented and, when needed, are provided by the specific vertical application.

The *Interface Sublayer* is where the interfaces and the service APIs, such as read/write API keys, are located.

4.2.2 Server Functionalities

As a RESTful architecture, a URI is associated to every resource. These resources are modeled as follow:

- every object in the server is identified as a *channel*. A channel represents a real entity, such as a smartphone, a laptop or a sensor;
- every device can have one or more fields associated with it, based on the number of its sensors; each field is identified with a data point.

When a user wants to register a new channel, the profiling module is activated. As shown in Figure 4.2, the owner can then indicate the characteristics of the objects such as the name, a description and its mobility. For those devices with enough computation capabilities, such as a laptop or a smartphone, some information about the object itself are provided automatically by the application, e.g. the brand and the MAC address: in this way it is possible to greatly shorten the registration process for the benefit of the owner. Eventually, if the owner is registering a fixed device, such as a desktop or a printer, it is possible to insert the location of the object,

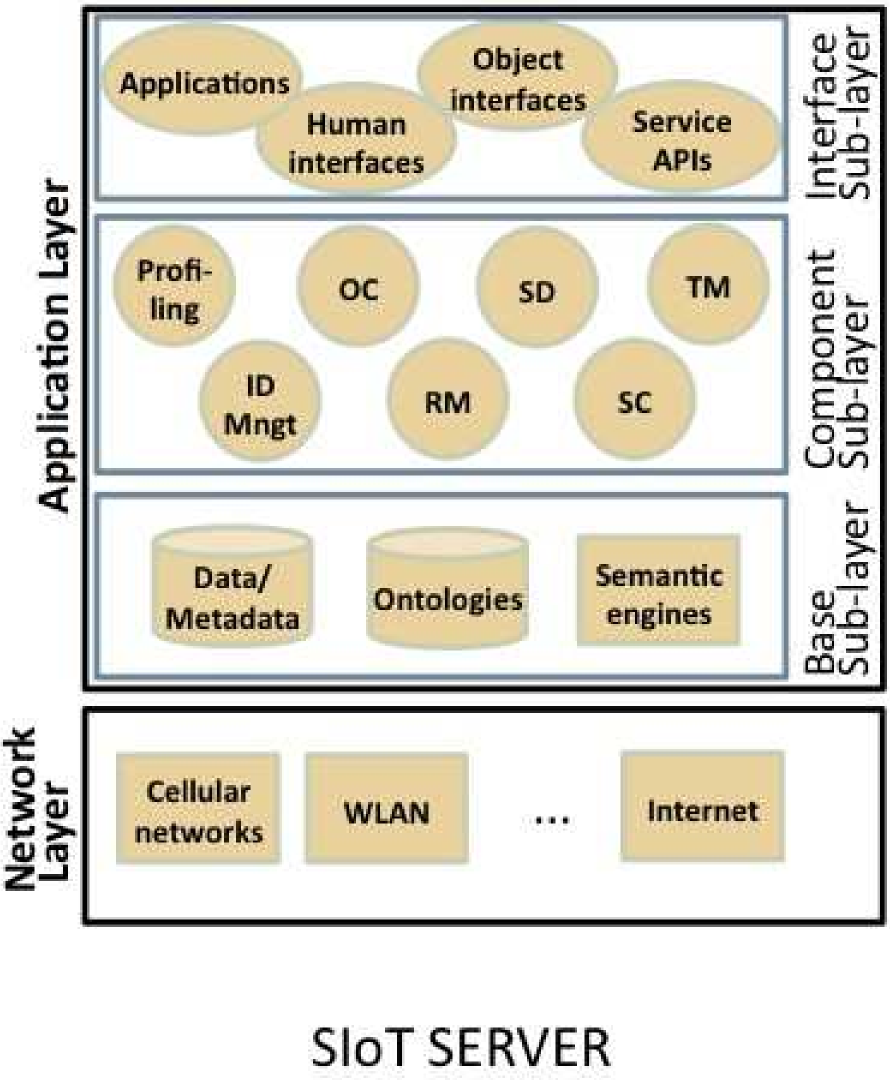
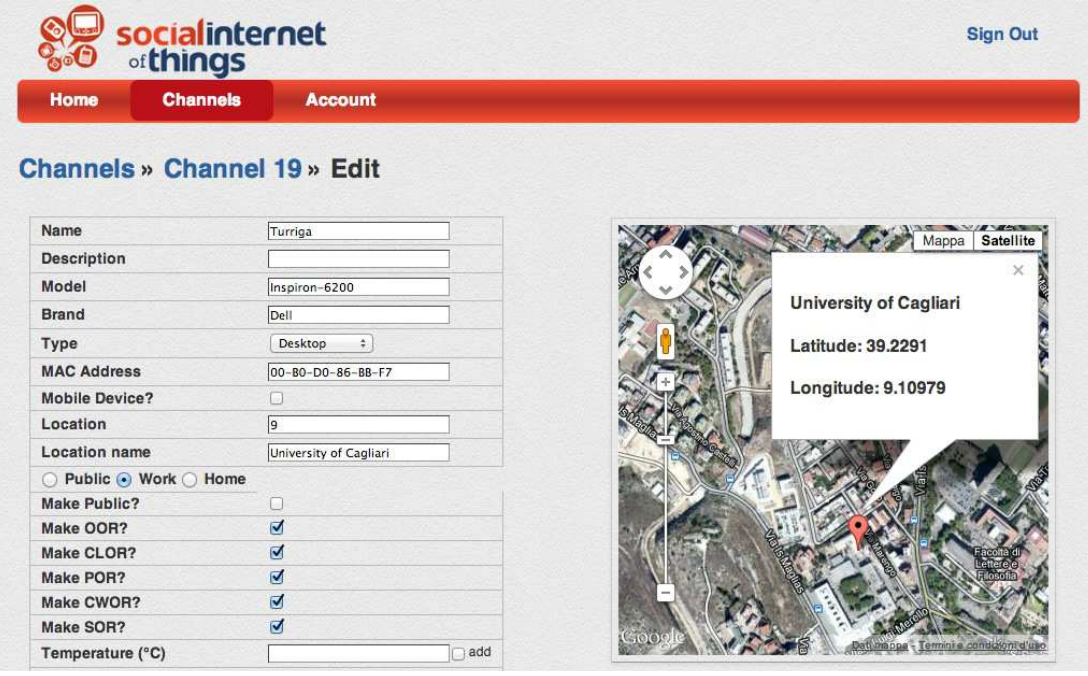


Figure 4.1: Proposed platform implementation

which enables the creation of location-based relationships, such as the co-location one. During the registration phase, the owner can choose which relationships the objects can create with other peers and which sensors, and consequently which fields, should be activated. When the registration is completed, the ID management module assigns a unique ID number to the object.

Objects can then start to create their own social relationships that are managed by the RM in two different ways:

- **Profiling relationship**. These relationships are generated based only on the profile information of the objects, and do not depend on the owner behavior. To this category belong *Ownership Object Relationship* (OOR), *Co-Location Object Relationship* (C-LOR) and *Parental Object Relaziontship* (POR). Indeed, OOR are created among objects registered in the SIoT by the same user.



Channels » Channel 19 » Edit

Name	Turriga
Description	
Model	Inspiron-6200
Brand	Dell
Type	Desktop
MAC Address	00-B0-D0-86-BB-F7
Mobile Device?	□
Location	9
Location name	University of Cagliari
○ Public ● Work ○ Home	
Make Public?	□
Make OOR?	☑
Make CLOR?	☑
Make POR?	☑
Make CWOR?	☑
Make SOR?	☑
Temperature (°C)	□ add

University of Cagliari

Latitude: 39.2291

Longitude: 9.10979

Figure 4.2: Registration of a new channel

When objects have the same value of the attribute *model*, a POR is created. To activate a C-LOR two objects need to be fixed in the same *location* (numeric ID).

- **Dynamic relationship**. These relationships are created when users, and consequently objects, interact with each other and satisfy the rules defined in [26]. To this category belong *Co-Work Object Relationship* (C-WOR) and *Social Object Relationship* (SOR). In particular, it is important that the server recognizes two objects in the same location, even if the objects are not in visibility.

The RM module is activated every time a new object is registered in the SIoT or every time an object sends information about its own location or about the IDs (i.e. mac address, RFID id) of the objects it has encountered. For Dynamic relationships, the RM module is activated by events about devices visibility when a device posts a sensed mac address or RFID. For C-WOR events, it is necessary either a work-type location post in the same time of the ID post or that at least one of the two devices is fixed in a work-type location.Two devices must be in visibility in two separate thirty-minute intervals spaced at least 8 hours for a friendship request storing. Every pass in the friendship request process is managed by the server and devices need only to send the sensor data. For example, as shown in Figure 4.3, device 1 senses the presence of another device, number 2, and then sends this information to the server, updating the relative data point, in particular the mac address data field. The server recognizes this field as a potential event and checks if this mac address belongs to a registered object. If this is the case, the RM module is activated to verify if, with the last data received, there are the conditions to create a new dynamic relationship. Eventually, the friendship request from device 1 is stored and if the device 2 performs a friendship request toward the device 1 as well, a new relationship is created.

When an object needs to send or retrieve its own data to the server, it uses its write API key, known only by the object itself. Instead, when an object needs to retrieve friends data from the server, it uses the read API key of the object to which it wants to retrieve data. The read API key is only known by the object itself and by its friends but it is allowed to share read API keys in those cases in which data from friend of a friend are required. Data from the server can be obtained using one of the two following methods:

- Pull. Every object requires data at regular intervals or when needed.
- Push. Data are sent from server to objects when available. Indeed, a HTTP daemon always listening is needed on every capable device. On smart devices, such as smartphone, tablet or laptop, it is also possible to use the push system of the operating system of the device itself.

If, during registration, the owner has set the object as public, any object that wants to retrieve information about the status of the public object, using the pull method, just needs to know its ID number; however, if an object wants to require data from a private object, it will also need the read API key of that object. Indeed, a device can retrieve the list of the IDs and the read API keys of its friends. This list can be obtained, in JSON, XML or CSV format as shown in the Listing 4.1, through a POST to the REST resource *friendships*, using the write API key.

Instead, with the push method, the server sends directly to all the objects in the friendship list the available data. In the same way, when a new friendship is created the server sends the updated list to the new object.

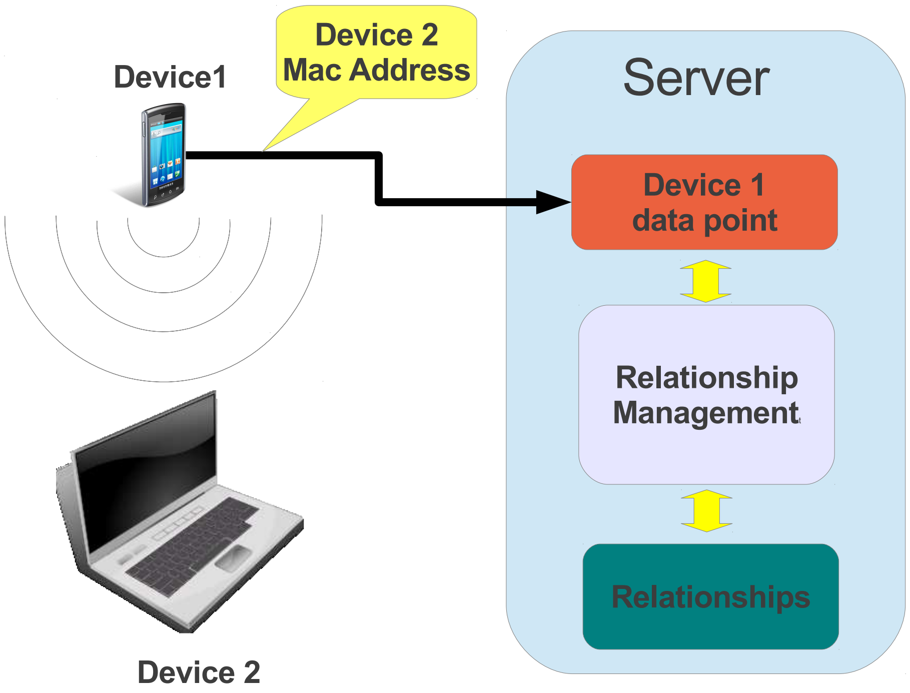


Figure 4.3: device identification during a meeting

```
<?xml version="1.0" encoding="UTF-8"?>
<relationships>
   <relation-type>OOR</relation-type>
   <channel-id type="integer">12</channel-id>
   <read-api-key>UXFN5F6SWXWIM3UM</read-api-key>
   <relation-type>OOR</relation-type>
   <channel-id type="integer">167</channel-id>
   <read-api-key>Q4CL7WQM5SUNSMS5</read-api-key>
   <relation-type>POR</relation-type>
   <channel-id type="integer">32</channel-id>
   <read-api-key>GIG8Z8WQBAPMO17G</read-api-key>
   <relation-type>POR</relation-type>
   <channel-id type="integer">126</channel-id>
   <read-api-key>WV3EOODHHISN8JZK</read-api-key>
   <relation-type>SOR</relation-type>
   <channel-id type="integer">4</channel-id>
   <read-api-key>C71DKJVOEDLU2K06</read-api-key>
</relationships>
```

Listing 4.1: Response to a *friendship* list request

4.2.3 Groups management

The relationships I identified so far, can include a large number of objects, and this leads to problems such as a long time for service discovery, since not all the objects can be helpful for a particular request. If I consider, for example, a large company, all the devices of every employee and the devices of the company itself, such as printers, scanners and desktops, would be tied by a co-work relationship, whereas it would be useful to have a group of objects belonging to different departments.

It is then important to have a tool to divide social relationships into groups. In the same way, humans create groups of particular interest (football teams, politic groups, online shopping) in *Social Networks* (SNs), the objects can create their own groups based on the applications they are using.

Three different solutions can be implemented, as described in the following:

Client-side groups management. The application must be able to create automatically the various groups with minimal user intervention and to do so, it has to verify some conditions that are specific to the use-case. These conditions are not limited by the fields of the device, i.e. its sensors, but may include other information. For example, an application can distinguish among the devices of employees that belong to different departments: it first verifies if they share a C-WOR and then it could check in a file for the employee ID to perfectly associate the devices of the other employees to the group. The benefit of this solution is that the groups created are exactly those required and specific to the application. The disadvantage is that, since the application manages the groups, different devices may create different groups that instead should be coincident, if, for example, a device has an out-of-date file, and moreover this solution increases the workload of the devices.

Server-side groups management. In this case, a user must create manually the group and set the rules, based on the fields associated to each device. Then the server takes care of binding to the group all the devices that comply with the above mentioned rules, in the same way the RM module creates the dynamic relationships. This solution has the advantage that the devices have a lighter workload but there may be excessive fragmentation of the groups, due to users that should belong to the same group creating new groups with different rules and the need to identify superusers for the creation of groups.

Hybrid solution for groups management. Groups are managed on the server-side, but the rules are set by the client using the fields provided by each device; information are tagged in order to help the server to identify the characteristic of each group. Only one group is created since the server associated all similar groups with the same tags, and then the workload on the devices remains light. The list of members of a group, as a REST resource, can be required in the same way of the friendship list.

4.2.4 SIoT Prototype Development

I realized my implementation based on ThingSpeak project [103], and I concentrated my efforts on social networking aspects. I implemented the relationship management module and the possibility to create and manage groups. Furthermore, I developed a location indexing system based on Google Maps to localize coherently fixed devices. I modified the channel structure to handle more than eight sensors and to manage text messages with tags among devices; I improved the owner control management of the objects and allowed devices to update remotely their own profile.

The service is available for tests here[1]. It is an alpha version still under development, but it is already capable to create relationships among registered devices. As ThingSpeak, SIoT is an open source project, and source code is released by the beta version.

4.3 Scenarios

In this section I describe some scenarios under development at the Faculty of Engineering of Cagliari that use the defined platform, where objects create their own relationships and groups, in order to provide several functionalities to the final users. Considering, smartphones, laptops and sensors in the area of the campus, I show some possible applications that exploit their social relationships. The focus of these applications is to provide usuful information to the object owners (the students, tipically) with minimal human intervention, except in the initial configuration phase. Every object can send messages, i.e. the data obtained by its sensors, to its own dashboard, and these messages can be seen by its friends as updates.

4.3.1 Lectures Information

The first application addresses the problem of communicating in an efficient way information such as the time of classes/tests and the availability of new teaching materials. The main problem to address concerns the identification of the actual recipients of these information. In the area of the campus, all the students' devices would be tied by a C-WOR; this relation could be useful to share generic information, such as holidays, employers strikes, student elections. However, it should be impossible to provide more granular services based on the field of study, the

[1] http://platform.social-iot.org/

academic year or the single classes. Here comes into play the groups of interests concept. With the use of the hybrid method, described in Section 4.2.3, the application decides the parameters to create the groups and the server manages and merges them, if necessary. This solution allows the same precision as the client-side method with a light workload on the devices.

The application gives information (alerts, web links, timetables and others) about the lectures to all the devices participating to the same group. As initial configuration of the application, it is possible to download the class timetable and the building map of the campus, for example using QR-codes. The application is able to assess when the professor and at least 50% of the students of the group are in the same location by sensing vicinity with the Bluetooth interface, and then it can notify to all the missing students of that group that a class is started and provide information to guide them to that class location.

4.3.2 Student car pooling

Rebecca needs to go to the university cafeteria after her class to have lunch but unfortunately she does not own a car. Her smartphone can create SOR with other students' smartphones at the cafeteria or C-WOR with her colleagues' smartphone, so when Rebecca needs a ride, she can simply use an application to discover if any other student is going to the cafeteria. The application automatically sends messages asking for a ride to all the devices that meet the parameters set by the user, for example:

- the number of common friends
- the membership to the same groups
- the frequency meeting

In the same way, the application can be set to receive requests only from devices with a certain relationship or that belong to certain groups and sort them accordingly. It is thus guaranteed a certain level of trustworthiness using only social parameters. As shown in Figure 4.4, consider for example the set of all the devices tied by a C-WOR in the campus in the grey area. Since Rebecca attended courses in Mathematics and Computer Science, she also belongs to groups A and B, respectively. Nicola's smartphone belongs to groups A and B as well, since Nicola is her study partner. Mario's laptop shares only a C-WOR with Rebecca's smartphone since he studies in another department, whereas Lisa's tablet is part of the group A. In this scenario, Nicola's device represents the most trustworthy device and then it represents the best choice to share a ride.

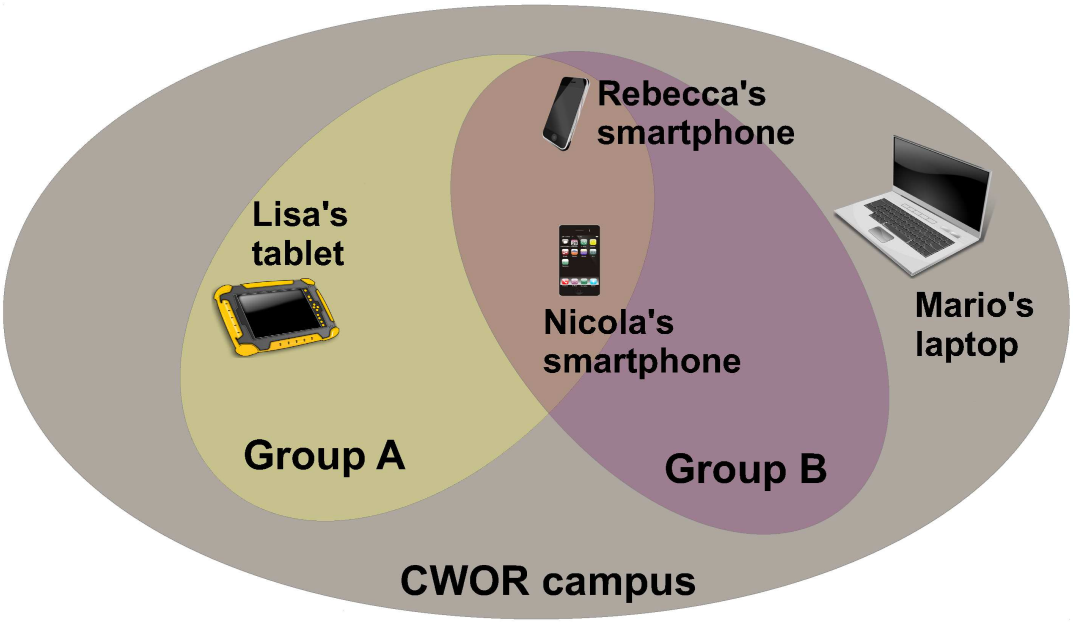


Figure 4.4: Trustworthiness by social parameters

4.3.3 Other university social life events

In the same way of the use-cases illustrated above, it can be possible to manage general information on the use of common areas such as university cafeteria, library and computer workstations. The library room is usually very crowded depending on the time of day, especially the PC workstations run out quickly. In this case, the application provides an assessment of the crowding of these environments by monitoring the number of devices and allows to give an approximate indication of the availability of seats. By the statistics of attendance, the application provides information on timing, categorized by crowding. Likewise, you can assess the crowding of canteen or cafeteria, in order to decide whether you prefer a less crowded place for time limits, or a popular spot for increasing socialization.

Chapter 5

Conclusions

In this thesis I have focused on the integration of social networking concepts into the Internet of Things, which leads to the so called “Social Internet of Things” (SIoT) paradigm. Recently, the SIoT has been the subject of several independent research activities as it promises to achieve scalable solutions in networks interconnecting trillions of nodes and to support new interesting applications.

More specifically, in Chapter 1 I have identified the types and the characteristics of the social relationships that can be established by objects in the SIoT. Furthermore, I have proposed a system architecture and the required basic functionality for an implementation of the SIoT. Finally, I have statistically analyzed the structure of the SIoT network.

My analysis has been based on the output of the SWIM mobility simulator. Results of such an analysis show that the probability distributions of the distance between nodes that are linked by a social relationship depend on the type of relationship. More specifically,

- for OOR relationships, such a distribution is characterized by a powerlaw behavior.
- for SOR relationships, such a distribution is characterized by a power law behavior for small distances and exponential behavior for large distances.
- for C-WOR relationships, such a distribution is characterized by a Gamma behavior

Furthermore, it is obvious that, for C-LOR relationships, the distances between connected nodes must be small. Whereas, for POR relationships, the existence of a relationship between two nodes is independent of their distance.

It follows that the above types of social relationships offer the possibilities to set long as well as short links and, therefore, their weight can be tuned in the SIoT in such a way that the resulting network structure offers the desired features in terms of navigability and scalability.

In Chapter 2 I have focused on the link selection in the social IoT. The driving idea is to select a narrow set of links in order for a node to manage more efficiently its friendships. I first demonstrate how a SIoT network has the characteristics of navigability and then I apply several heuristics for link selection and analyze the behavior of the network in terms of giant component, average degree, local cluster coefficient and average path length.

Chapter 3 have focused on the trustworthiness management in the social IoT by proposing subjective and objective approaches. In the subjective model each node computes the trustworthiness of its friends on the basis of its own experience and on the opinion of the friends in common with the potential service providers. In the objective model, the information about each node is distributed and stored making use of a Distributed Hash Table structure so that any node can make use of the same information. I have performed extensive simulations that show how both approaches allow for isolating the malicious node behavior. The major difference between the two methods is that the subjective approach has a slower transitory response, which is particularly evident when dealing with nodes with dynamic behaviors. However, it is practically immune to behaviors typical of social networks, where a malicious person modifies her actions based on the relationships. On the contrary, the objective approach suffers from this kind of behavior, since a node's trustworthiness is global for the entire network and this include both the opinion from the nodes with which it behaved maliciously and the opinion from the nodes with which it behaved benevolent. As to the overhead, the objective approach is significantly lighter since there is not the need to get the indirect opinions from friends as it is the case for the subjective approach. However, with the subjective model there is not the need to install and maintain pre-trusted objects nodes in the network to store the reputation values.

In Chapter 4, I briefly present some of the main platforms among the IoT implementations, and identify their common characteristics. Then, I propose the first, to the best of my knowledge, implementation of an experimental platform for the Social Internet of Things. The main innovations, with respect to the others IoT platforms, are the possibility for the objects to create their own relationships, based on the rules set by their owners, and to create groups of interest as it happens in human social networks. I also introduced some prototype applications, currently under development at the University of Cagliari.

Appendix A

Theoretical analysis of the subjective model performance

Herein we provide an analysis of the performance of the subjective model, whose objective is to discriminate benevolent nodes from malicious ones with the minim error. The resulting trustworthiness formula is made of three additive elements (3.1), namely the centrality, the direct opinion, and the indirect opinion, each one contributing to isolating malicious nodes.

The subjective centrality measures how much a node is central in another node "life". If we calculate the average centrality of node p_i over all its friendships $\mathcal{N}_i$, we obtain

$$R_i = \frac{\sum_{j=1}^{|\mathcal{N}_i|} \frac{|\mathcal{K}_{ij}|}{\left(|\mathcal{N}_i|-1\right)}}{|\mathcal{N}_i|} = \frac{\sum_{j=1}^{|\mathcal{N}_i|} |\mathcal{K}_{ij}|}{\left(|\mathcal{N}_i|-1\right)|\mathcal{N}_i|} \tag{A.1}$$

that corresponds to the local clustering coefficient for node p_i, and then gives an indication of how close node p_i's neighbors are to being a clique, i.e. a complete graph.

As there are no models to represent the behavior of the nodes in creating and updating the clusters of friends, we do not have the basis to compute the efficiency of this parameter, but evidences of its capacity to isolate malicious nodes can be found in literature. In [110] the authors state that "trust is to be built based not only on how well you know a person, but also on how well that person is known to the other people in your network" and then they show that, using local clustering for email filtering, it is possible to classify correctly up to 50% of the messages. Moreover, in [111], the authors show how trust networks are highly related to the creation of cluster.

When the nodes start to exchange services, they still do not have any information about how much they can trust each other. However, they can rely on the centrality and, for what concerns direct and indirect opinion, on the relationship factor and on the computation capabilities. When N_{ij} becomes high, the dependence of the direct opinion on the relationship factor and the computation capabilities decreases whereas that related to the past transactions increases. The feedback generated for each received service is provided by (3.9). To simplify the analysis, as done in the simulations, we assume a binary feedback system is used. When analyzing the received service, the client may introduce some errors due to several reasons and mostly because of the intrinsic difficulty in evaluating the quality of the received service. We then introduce probability e that a node gives the wrong feedback, so that the probability to give the correct feedback is $h = 1 - e$. The probability that p_i generates k correct feedback ($f_{ij} = 1$ when p_j is benevolent and $f_{ij} = 0$ when p_j is malicious) over n transactions with p_j, follows a binomial distribution

$$P(k) = \binom{n}{k} h^k (1-h)^{(n-k)} \tag{A.2}$$

Note that if we consider feedback having the same weights, the long term and short term opinions $O_{ij}^{lon/rec} = k$ if p_j is benevolent and $O_{ij}^{lon/rec} = 1 - k$ if p_j is malicious. Accordingly, these follow a binomial distribution as well, where the expected value is h if node p_j is benevolent, and $1 - h$ if it is malicious, and the variance is $h(1-h)$. This distribution can be approximated with a gaussian one (when $n > 30$) with the same variance and average values. When adding the two contributions from the short and long term opinions, considering $\gamma = 0.5$ as in the simulations, we obtain that the direct opinion is still a gaussian distribution with the same mean value ($\mu_b = p$ and $\mu_m = 1 - p$ based on the behavior of node p_j) and a variance equals to $h(1-h)/2$.

To calculate the distribution of the indirect opinion, we assume for simplicity that the credibility for all the nodes is the same; in this case, it is the sum of gaussian-distributed variables, so it follows a gaussian distribution as well. Considering that x$\%$ of the nodes are malicious, the average value for the indirect opinion is $(1 - 0.x)\mu_{b,m} + 0.x\mu_{m,b}$ while its variance is $\sigma^2 / |\mathcal{K}_{ij}|$.

Using the $erfc$ function to calculate the error when estimating the trustworthiness of a node, we obtain the results shown in Table A.1 for different values of the error probability and $x = 25\%$. Both the parameters can achieve low error probability. Indeed, the direct opinion is the parameter that most affects the trustworthiness calculation, and that leads to the smallest errors. However, when services start to circulate in the network, the first parameter that varies and gives actual information about the trustworthiness of a node is the indirect opinion. This happens because, if node p_i wants to evaluate the trustworthiness of node p_j, it is simply more probable that it can obtain information from one of the common friends $\mathcal{K}_{ij}$ than from a direct transition between p_i and p_j. Moreover, with the combination of these two parameters, it is possible to achieve more reliable results than using only one of them.

Table A.1: Probability to misjudge a node

Direct opinion

e	μ		σ^2	error
	benevolent	malicious		
0.1	0.9	0.1	0.045	$3.15 * 10^{-17}$
0.15	0.85	0.15	0.064	$9.63 * 10^{-7}$
0.2	0.8	0.2	0.08	$1.016 * 10^{-5}$

Indirect opinion

e	μ		σ^2	error
	benevolent	malicious		
0.1	0.7	0.3	0.024	$6.56 * 10^{-15}$
0.15	0.675	0.325	0.034	$1.084 * 10^{-5}$
0.2	0.65	0.35	0.043	$1.14 * 10^{-2}$

Direct + Indirect opinion ($\alpha = 0.6$ **and** $\beta = 0.4$)

e	μ		σ^2	error
	benevolent	malicious		
0.1	0.82	0.18	0.017	$8.72 * 10^{-77}$
0.15	0.78	0.22	0.024	$2.04 * 10^{-29}$
0.2	0.74	0.26	0.03	$8.31 * 10^{-14}$

List of Figures

List of Tables

Bibliography

[1] J. Surowiecki, *The wisdom of crowds*. Doubleday, 2004.

[2] S. Sarma, D. Brock, and K. Ashton, “The networked physical world: proposals for the next generation of computing commerce, and automatic identification,” *AutoID Center White Paper*, 1999.

[3] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” *Computer Networks*, vol. 54, no. 15, pp. 2787 – 2805, 2010.

[4] IDTechEx, R. Das, and P. Harrop, *RFID forecasts, players and opportunities 2011-2021*. IDTechEx, 2011.

[5] N. I. Council, *Disruptive Civil Technologies Six Technologies with Potential Impacts on US Interests Out to 2025*. IDTechEx, 2008. [Online]. Available: http://www.dni.gov/nic/NIC_home.html

[6] A. Mei and J. Stefa, “Swim: A simple model to generate small mobile worlds,” in *INFOCOM 2009, IEEE*, 2009, pp. 2106 –2113.

[7] J. Kleinberg, “The convergence of social and technological networks,” *Communications of the ACM*, vol. 51, no. 11, pp. 66 – 72, November 2008.

[8] A. Mislove, K. P. Gummadi, and P. Druschel, “Exploiting social networks for internet search,” in *Proc. of ACM HotNets 2006*, November 2006.

[9] A. Fast, D. Jensen, and B. N. Levine, “Creating social networks to improve peer-to-peer networking,” in *Proc. of ACM KDD’05*, August 2005.

[10] S. Marti, P. Ganesan, and H. Garcia-Molina, “Sprout: P2p routing with social networks,” in *Proc. of EDBT 2004*, March 2004.

[11] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, “Sybilguard: defending against sybil attacks via social networks,” in *Proc. of IEEE SigComm 2006*, September 2006.

[12] P. Costa, C. Mascolo, M. Musolesi, and G. P. Picco, “Socially-aware routing for publish/-subscribe in delay-tolerant mobile ad-hoc networks,” *IEEE Journal on Selected Areas of Communications*, vol. Vol. 26, no. 5, pp. 748–760, June 2008.

[13] A. Mei, G. Morabito, P. Santi, and J. Stefa, “Social-aware stateless forwarding in pocket switched networks,” in *Proc. of IEEE Infocom – Miniconference 2011*, April 2011.

[14] L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H.-W. Gellersen, “Smart-its friends: A technique for users to easily establish connections between smart artefacts,” in *Proceedings of the 3rd international conference on Ubiquitous Computing*, ser. UbiComp ’01. Springer-Verlag, 2001, pp. 116–122.

[15] J. Bleecker, “A manifesto for networked objects – cohabiting with pigeons, arphids and aibos in the internet of things – why things matter what’ s a blogject ? what about spimes?’,” in *Proc. of the 13th International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI*, 2006.

[16] E. Nazz and T. Sokoler, “Walky for embodied microblogging:sharing mundane activities through augmented everyday objects,” in *Proc. of the 13th International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI*, 2011.

[17] M. Kranz, L. Roalter, and F. Michahelles, “Things that twitter: social networks and the internet of things,” in *Proc. of What can the Internet of Things do for the Citizens (CIoT) Workshop at Pervasive’10*, May 2010.

[18] P. Mendes, “Social-driven internet of connected objects,” in *Proc. of the Interconnecting Smart Objects with the Internet Workshop*, 25th March 2011.

[19] H. Ning and Z. Wang, “Future internet of things architecture: Like mankind neural system or social organization framework?” *Communications Letters, IEEE*, vol. 15, no. 4, pp. 461 –463, 2011.

[20] L. Ding, P. Shi, and B. Liu, “The clustering of internet, internet of things and social network,” in *Proc. of the 3rd International Symposium on Knowledge Acquisition and Modeling*, 2010.

[21] D. Guinard, M. Fischer, and V. Trifa, “Sharing using social networks in a composable web of things,” in *PERCOM Workshops, 2010.*

[22] A. Jian, G. Xiaolin, Z. Wendong, and J. Jinhua, “Nodes social relations cognition for mobility-aware in the internet of things,” in *Proc. of the 4th International Conference on Cyber, Physical and Social Computing*, October 2011.

[23] “Finnish strategic centre for science and innovation: For information and communications (ICT) services, businesses, and technologies – internet of things strategic research agenda (IoT–SRA),” September 2011.

[24] E. A. K. amd N. D. Tselikas and A. C. Boucouvalas, “Integrating rfids and smart objects into a unified internet of things architecture,” *Advances in Internet of Things*, vol. 1, no. 1, pp. 5–12, 2011.

[25] J. Kleinberg, “The small-world phenomenon: an algorithmic perspective,” in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, ser. STOC ’00. New York, NY, USA: ACM, 2000, pp. 163–170. [Online]. Available: http://doi.acm.org/10.1145/335305.335325

[26] L. Atzori, A. Iera, and G. Morabito, “Siot: Giving a social structure to the internet of things,” *Communications Letters, IEEE*, vol. 15, 2011.

[27] D. M. Boyd and N. B. Ellison, “Social network sites: Definition, history, and scholarship,” *Journal of Computer-Mediated Communication*, vol. 1, no. 13, 2007.

[28] A. P. Fiske, “The four elementary forms of sociality: framework for a unified theory of social relations,” *Psychological review*, vol. 99, pp. 689–723, 1992.

[29] N. Haslam, “The four elementary forms of sociality: framework for a unified theory of social relations,” *Cognition*, vol. 53, pp. 59–90, 1994.

[30] L. Zheng *et al.*, “Technologies, applications, and governance in the internet of things,” *Internet of Things - Global Technological and Societal Trends*, River Publisher Ed. 2011.

[31] S. De, P. Barnaghi, M. Bauer, and S. Meissner, “Service modelling for the internet of things,” in *Proc. of the Federeted Conference on Computer Science and Information System*, 2011.

[32] Y. Huang and G. Li, “A semantic analysis for internet of things,” in *Proc. of the Intelligent Computation Technology and Automation Conference*, 2010.

[33] A. Katasonov, O. Kaykova, O. Khriyenko, S. Nikitin, and V. Terziyan, “Smart semantic middleware for the internet of things,” in *Proc. of the 5th International Conference on Informatics in Control Automation and Robotics*, 2008.

[34] J. Breslin and S. Decker, “The future of social networks on the internet,” *IEEE Internet Computing*, vol. 11, no. 6, pp. 86–90, 2007.

[35] D. Guinard, M. Mueller, and J. Pasquier, “Giving RFID a REST: Building a web-enabled EPCIS,” in *Proc. of Internet of Things 2010 Conference*, November 2010.

[36] R. Studer, S. Grimm, and A. A. (Eds.), *Semantic Web Services, Concepts, Technologies, and applications*. Springer-Verlag, 2007.

[37] B. Ostermaier, K. Romer, F. Mattern, M. Fahrmair, and W. Kellerer, “A real-time search engine for the web of things,” in *Internet of Things (IOT), 2010*, 2010, pp. 1–8.

[38] K. Romer, B. Ostermaier, F. Mattern, M. Fahrmair, and W. Kellerer, “Real-time search for real-world entities: A survey,” *Proceedings of the IEEE*, vol. 98, no. 11, pp. pp. 1887–1902, November 2010.

[39] “Internet of Things - Architecture, IoT–A – SOTA report on existing integration frameworks/architectures for WSN, RFID and other emerging IoT related technologies,” March 2011.

[40] S. Kosta, A. Mei, and J. Stefa, “Small world in motion (swim): Modeling communities in ad-hoc mobile networking,” in *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on*, 2010, pp. 1 –9.

[41] J. Leguay, A. Lindgren, J. Scott, T. Friedman, J. Crowcroft, and P. Hui, “CRAWDAD trace upmc/content/imote/cambridge (v. 2006-11-17),” Nov. 2006.

[42] H. Cai and D. Y. Eum, “Toward stochastic anatomy of inter-meeting time distribution under general mobility models,” in *Proc. ACM MobiHoc 2008*, Hong Kong SAR, China, May 2008.

[43] J. Travers, S. Milgram, J. Travers, and S. Milgram, “An experimental study of the small world problem,” *Sociometry*, vol. 32, pp. 425–443, 1969.

[44] H. Wang, C. C. Tan, and Q. Li, “Snoogle: A search engine for pervasive environments,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 21, no. 8, pp. 1188–1202, 2010.

[45] K.-K. Yap, V. Srinivasan, and M. Motani, “Max: human-centric search of the physical world.” in *SenSys*, J. Redi, H. Balakrishnan, and F. Zhao, Eds. ACM, 2005, pp. 166–179. [Online]. Available: http://dblp.uni-trier.de/db/conf/sensys/sensys2005.html#YapSM05

[46] J. Kleinberg, "Small-world phenomena and the dynamics of information," in *In Advances in Neural Information Processing Systems (NIPS) 14*. MIT Press, 2001, p. 2001.

[47] M. Bogu, D. Krioukov, and k. claffy, "Navigability of Complex Networks," *Nature Physics*, vol. 5, no. 1, pp. 74–80, Jan 2009.

[48] L. A. N. Amaral and J. M. Ottino, "Complex networks. augmenting the framework for the study of complex systems," *European Physical Journal B*, vol. 38, pp. 147–162, March 2004.

[49] J. Kleinberg, "Navigation in a small world," *Nature*, vol. 406, p. 845, 2000.

[50] D. J. Watts and S. H. Strogatz, "Collective dynamics of small-world networks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[51] R. Girau, M. Nitti, and L. Atzori, "Implementation of an experimental platform for the social internet of things," in *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2013 Seventh International Conference on*. IEEE, 2013, pp. 500–505.

[52] J. Leskovec, "Stanford large network dataset collection." [Online]. Available: http://snap.stanford.edu/data/

[53] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks," 2009. [Online]. Available: http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154

[54] P. Holme and B. J. Kim, "Growing scale-free networks with tunable clustering," *Physical Review E*, vol. 65, no. 2, p. 026107, 2002.

[55] D. Chen, G. Chang, D. Sun, J. Li, J. Jia, and X. Wang, "Trm-iot: A trust management model based on fuzzy reputation for internet of things." *Comput. Sci. Inf. Syst.*, vol. 8, no. 4, pp. 1207–1228, 2011.

[56] F. Bao, I.-R. Chen, M. Chang, and J.-H. Cho, "Hierarchical trust management for wireless sensor networks and its application to trust-based routing," in *ACM Symposium on Applied Computing*, 2011.

[57] Y. Liu, Z. Chen, F. Xia, X. Lv, and F. Bu, "A trust model based on service classification in mobile services," in *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, 2010, pp. 572–577.

[58] F. Bao and I.-R. Chen, “Trust management for the internet of things and its application to service composition,” in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*, 2012, pp. 1–6.

[59] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman, “Reputation systems,” *Commun. ACM*, vol. 43, pp. 45–48, 2000.

[60] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, “The eigentrust algorithm for reputation management in p2p networks,” in *Proceedings of the 12th international conference on World Wide Web*. ACM, 2003, pp. 640–651.

[61] A. A. Selcuk, E. Uzun, and M. R. Pariente, “A reputation-based trust management system for p2p networks,” in *Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid*. IEEE Computer Society, 2004, pp. 251–258.

[62] R. Sherwood, S. Lee, and B. Bhattacharjee, “Cooperative peer groups in nice,” *Comput. Netw.*, vol. 50, pp. 523–544, 2006.

[63] E. Adar and B. A. Huberman, “Free riding on gnutella,” 2000.

[64] M. Feldman, K. Lai, and J. Chuang, “Quantifying disincentives in peer-to-peer networks,” in *1st Workshop on Economics of Peer-to-Peer Systems*.

[65] R. Jurca and B. Faltings, “An incentive compatible reputation mechanism,” in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. ACM, pp. 1026–1027.

[66] S. Marti and H. Garcia-Molina, “Identity crisis: Anonymity vs. reputation in p2p systems,” in *Proceedings of the 3rd International Conference on Peer-to-Peer Computing*. IEEE Computer Society, 2003, p. 134.

[67] R. Zhou, K. Hwang, and M. Cai, “Gossiptrust for fast reputation aggregation in peer-to-peer networks,” *IEEE Trans. on Knowl. and Data Eng.*, vol. 20, pp. 1282–1295, 2008.

[68] Z. Liang and W. Shi, “Enforcing cooperative resource sharing in untrusted p2p computing environments,” *Mob. Netw. Appl.*, vol. 10, pp. 971–983.

[69] Y. Wang and J. Vassileva, “Bayesian network-based trust model,” in *Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence*, ser. WI ’03. IEEE Computer Society, 2003, pp. 372–.

[70] B. Yu, M. P. Singh, and K. Sycara, “Developing trust in large-scale peer-to-peer systems,” in *Proceedings of First IEEE Symposium on Multi-Agent Security and Survivability*, 2004, pp. 1–10.

[71] L. Xiong and L. Liu, “Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities,” *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, vol. 16, pp. 843–857, 2004.

[72] B. E. Commerce, A. Jsang, and R. Ismail, “The beta reputation system,” in *In Proceedings of the 15th Bled Electronic Commerce Conference*.

[73] Z. Despotovic and K. Aberer, “Maximum likelihood estimation of peers’ performance in p2p networks,” 2004.

[74] T. DuBois, J. Golbeck, and A. Srinivasan, “Predicting trust and distrust in social networks,” in *Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom)*. IEEE, 2011, pp. 418–424.

[75] B. Carminati, E. Ferrari, and J. Girardi, “Trust and share: Trusted information sharing in online social networks.” in *ICDE*. IEEE Computer Society, 2012, pp. 1281–1284.

[76] A. Jøsang, “Artificial reasoning with subjective logic,” in *Proceedings of the Second Australian Workshop on Commonsense Reasoning*, 1997.

[77] G. Liu, Y. Wang, and L. Li, “Trust management in three generations of web-based social networks,” in *Proceedings of the 2009 Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing*, ser. UIC-ATC ’09. IEEE Computer Society, 2009, pp. 446–451.

[78] P. W. Fong, “Relationship-based access control: protection model and policy language,” in *Proceedings of the first ACM conference on Data and application security and privacy*. ACM, 2011, pp. 191–202.

[79] B. Carminati, E. Ferrari, and M. Viviani, “A multi-dimensional and event-based model for trust computation in the social web,” in *Social Informatics*. Springer, 2012, pp. 323–336.

[80] J. A. Golbeck, “Computing and applying trust in web-based social networks,” Ph.D. dissertation, 2005.

[81] J. Golbeck and J. Hendler, "Inferring binary trust relationships in web-based social networks," *ACM Trans. Internet Technol.*, vol. 6, no. 4, pp. 497–529, Nov.

[82] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in *Proceedings of the 29th Australasian Computer Science Conference*. Australian Computer Society, 2006, pp. 85–94.

[83] A. Jøsang and S. Pope, "Semantic constraints for trust transitivity," in *Proceedings of the 2nd Asia-Pacific conference on Conceptual modelling*. Australian Computer Society, Inc., 2005, pp. 59–68.

[84] B. Christianson and W. S. Harbison, "Why isn't trust transitive?" in *Proceedings of the International Workshop on Security Protocols*. Springer-Verlag, 1997, pp. 171–176.

[85] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The social internet of things (siot)–when social networks meet the internet of things: Concept, architecture and network characterization," *Computer Networks*, 2012.

[86] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decis. Support Syst.*, vol. 43, no. 2, pp. 618–644, Mar.

[87] R. Ashri, S. Ramchurn, J. Sabater, M. Luck, and N. Jennings, "Trust evaluation through relationship analysis," in *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. ACM, 2005, pp. 1005–1011.

[88] M. Procidano and K. Heller, "Measures of perceived social support from friends and from family: Three validation studies," *American journal of community psychology*, vol. 11, no. 1, pp. 1–24, 1983.

[89] G. Zimet, N. Dahlem, S. Zimet, and G. Farley, "The multidimensional scale of perceived social support," *Journal of personality Assessment*, vol. 52, no. 1, pp. 30–41, 1988.

[90] D. Krackhardt, "The strength of strong ties: The importance of philos in organizations," *Networks and organizations: Structure, form, and action*, vol. 216, p. 239, 1992.

[91] M. Ruef, "Strong ties, weak ties and islands: structural and cultural predictors of organizational innovation," *Industrial and Corporate Change*, vol. 11, no. 3, pp. 427–449, 2002.

[92] L. Freeman, "Centrality in social networks conceptual clarification," *Social networks*, vol. 1, no. 3, pp. 215–239, 1979.

[93] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," *SIGCOMM Comput. Commun. Rev.*, vol. 31, pp. 161–172, 2001.

[94] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *SIGCOMM Comput. Commun. Rev.*, vol. 31, pp. 149–160, 2001.

[95] A. I. T. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*. Springer-Verlag, 2001, pp. 329–350.

[96] J. Golbeck, "Personalizing applications through integration of inferred trust values in semantic web-based social networks," *W8: Semantic Network Analysis*, p. 15, 2008.

[97] G. Chiola, G. Cordasco, L. Gargano, A. Negro, and V. Scarano, "Optimizing the finger table in chord-like dhts," in *Parallel and Distributed Processing Symposium, 2006. 20th International*, 2006, p. 8 pp.

[98] H. Bisgin, N. Agarwal, and X. Xu, "Investigating homophily in online social networks," in *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, vol. 1, 2010, pp. 533 –536.

[99] (2013) Postscapes. [Online]. Available: http://www.postscapes.com

[100] (2013) Cosm. [Online]. Available: http://cosm.com

[101] (2013) Nimbits. [Online]. Available: http://www.nimbits.com

[102] A. Piras, D. Carboni, and A. Pintus, "A platform to collect, manage and share heterogeneous sensor data," in *Networked Sensing Systems (INSS), 2012 Ninth International Conference on*, june 2012, pp. 1 –2.

[103] (2013) Thingspeak. [Online]. Available: http://www.thingspeak.com

[104] (2013) Iobridge. [Online]. Available: http://www.iobridge.com

[105] A. Castellani, N. Bui, P. Casari, M. Rossi, Z. Shelby, and M. Zorzi, "Architecture and protocols for the internet of things: A case study," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, 29 2010-april 2 2010, pp. 678 –683.

[106] D. Guinard, I. Ion, and S. Mayer, “In search of an internet of things service architecture: Rest or ws-*? a developers perspective,” *Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pp. 326–337, 2012.

[107] N. B. Priyantha, A. Kansal, M. Goraczko, and F. Zhao, “Tiny web services: design and implementation of interoperable and evolvable sensor networks,” in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, ser. SenSys ’08. New York, NY, USA: ACM, 2008, pp. 253–266. [Online]. Available: http://doi.acm.org/10.1145/1460412.1460438

[108] M. Nitti, R. Girau, and L. Atzori, “Trustworthiness management in the social internet of things,” 2013.

[109] M. Nitti, R. Girau, L. Atzori, A. Iera, and G. Morabito, “A subjective model for trustworthiness evaluation in the social internet of things,” in *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*. IEEE, 2012, pp. 18–23.

[110] O. P. Boykin and V. Roychowdhury, “Personal Email networks: an effective anti-spam tool,” *Condensed Matter cond-mat/0402143*, 2004.

[111] W. Yuan, D. Guan, Y.-K. Lee, S. Lee, and S. J. Hur, “Improved trust-aware recommender system using small-worldness of trust networks,” *Knowledge-Based Systems*, vol. 23, no. 3, pp. 232–238, 2010.