Hindawi Publishing Corporation
Journal of Computer Networks and Communications
Volume 2012, Article ID 190639, 15 pages
doi:10.1155/2012/190639

Research Article

Experimental Evaluation of a SIP-Based Home Gateway with Multiple Wireless Interfaces for Domotics Systems

Rosario G. Garroppo, Loris Gazzarrini, Stefano Giordano, and Luca Tavanti

Dipartimento di Ingegneria dell'Informazione, Università di Pisa, 56126 Pisa, Italy

Correspondence should be addressed to Luca Tavanti, luca.tavanti@iet.unipi.it

Received 28 July 2012; Revised 9 October 2012; Accepted 9 October 2012

Academic Editor: Gildas Avoine

In modern houses, the presence of sensors and actuators is increasing, while *communication services* and *entertainment systems* had long since settled into everyday life. The utilization of wireless communication technologies, such as ZigBee, Wi-Fi, and Bluetooth, is attractive because of their short installation times and low costs. The research is moving towards the integration of the various home appliances and devices into a single domotics system, able to exploit the cooperation among the diverse subsystems and offer the end-user a single multiservice platform. In this scenario, the paper presents the experimental evaluation of a domotics framework centered on a SIP-based home gateway (SHG). While SIP is used to build a common control plane, the SHG is in charge of translating the user commands from and to the specific domotics languages. The analysis has been devoted to assess both the performance of the SHG software framework and the negative effects produced by the simultaneous interference among the three widespread wireless technologies.

1. Introduction

Domotics refers to a system that controls several (or all) home "services," such as lighting, HVAC (heating, ventilation, and air conditioning), communications, security, healthcare, and entertainment, in a integrated and automatic or semi-automatic way, allowing the user to manage them from a series of heterogeneous devices (e.g., touch panels, remotes, mobile handsets, and smartphones), either at home or from anywhere in the world. In the domotics archetype, all subsystems are able to talk to each other and interact in a seamless manner, realizing an intelligent structure that improves the quality of life, reduces the costs, and achieves energy savings. To put this paradigm into practice, the communication among the single devices and between the various subsystems is the fundamental operation. Hence, wired and wireless networks will be one of the building blocks of the present and future domotics solutions. On top of this somewhat "physical" element, a common control plane is also necessary, in order to unify the management operations into a single and portable user interface.

One of the major components of a domotics system is the set of sensors and actuators. These usually come in the form of one or more networks, backed either by a single technology or by different ones. Not always, however, do the specifications define a common control plane that is suitable to contemporarily manage devices belonging not only to different standards, but even to different application profiles. As a result, the burden of coordinating and making devices interoperate is often left entirely to the system implementer. Indeed, a scenario with mixed profiles and technologies is not so uncommon, especially in those environments where multiple services might be requested. One such example is exactly the "smart home" or domotics concept, in which several profiles and technologies (e.g., ZigBee's home automation, smart energy, and telecom services, or KNX's lighting, heating, and energy management—just to cite the most appealing ones) might all be present.

From the user perspective, the devices belonging to the diverse subsystems of the home services platform can be typically controlled through dedicated appliances located in the house (e.g., a touch panel, a smart telephone, a TV remote). However, this paradigm no longer holds for remote control operations that occur when the user is far from home. In this case the user would normally have a single device at hand, such as a notebook or a smartphone, by means of which he/she would like to control any device in the home, not just those belonging to a specific profile or technology, and possibly without complex configuration or selection procedures.

In addition to the need for a coordinating system for the DSANs, in today's houses we already find interpersonal communication and multimedia entertainment systems. Hence, the design of a domotics platform should also consider the integration of communication and multimedia applications with the DSAN-based services.

In this scenario, we describe an architecture designed to gain interoperability among devices belonging to different technologies and profiles. In our vision, the common control plane is realized through the Session Initiation Protocol (SIP) [1], while a *SIP-based home gateway* (SHG) translates the user commands from and to the specific DSAN language, thus allowing the user to control all domotics devices either at home or away from it, using his mobile terminal or his favorite SIP client, in a transparent, uniform, and simple way. The SHG, which is the major enabler of the envisioned system, is also devised to retain the compatibility with the existing SIP infrastructure and the deployed SIP clients, which can therefore be exploited in full.

Among the various domotics sensor and actuator networks (DSANs), wireless sensor networks (WSNs—note that the term "sensor" is often used for both sensors in the strict sense and for actuators too) are the version that is growing faster, due to shorter deploying times and simplified configuration. Several technologies and standards are nowadays available for the implementation of a WSN [2]. Especially the ones based on open or widely adopted standards, such as ZigBee, Bluetooth, *Z*-Wave, and KNX-RF, can undeniably be regarded as the most interesting ones. This is because they allow the deployment of large and almost self-configuring networks in relatively short times and at reduced costs. Two of these standards, ZigBee and Bluetooth, have been embedded into the SHG.

On the other hand, the current trend in multimedia and communication home systems is to move the physical transport services over the Wi-Fi technology. Thus, we have equipped our SHG also with a Wi-Fi interface, used for providing the above-mentioned "wideband" services.

The majority of these wireless standards operate into the unlicensed 2.4 GHz ISM band, which can be exploited by multiple users and networks at the same time. However, due to the mutual interference, the coexistence of different devices operating in proximity of each other can be troublesome. As proved by many authors [3, 4], this is especially true for ZigBee networks, whose performance is heavily influenced by the presence of Wi-Fi devices. While it is sometimes feasible to avoid the interference among devices sharing the same spectrum and implementing the same standard (e.g., collision avoidance schemes might work across separate networks), the use of incompatible modulations and channel access schemes makes it virtually impossible to ensure the coexistence among devices belonging to different technologies.

In summary, the design and implementation of a domotics gateway must face two key issues: integrating heterogeneous indoor devices and networks, allowing the composition of dynamic and pervasive services (including interpersonal communications and multimedia), and assuring the physical coexistence of the interfaces located on the gateway apparatus.

1.1. Contribution. We present a working prototype of the SHG, which is used to build a complete proof-of-concept of our SIP-based domotics architecture. A customized SIP event package and a notification server have also been developed to validate a possible extension to new services. The SHG was interfaced with an actual ZigBee network and a Bluetooth PAN, in addition to a generic Wi-Fi connection. We experimentally evaluated the performance of the SHG prototype, proving its ability to support large domotics systems.

In describing our SIP-based system, we also present the aspects that make it innovative. We arranged for the SHG to be the sole entity to have a SIP address, thus avoiding the overhead of having a SIP address for each home device. We designed and implemented a functional addressing and a control scheme to ease the user interaction with the system and an abstraction layer to decouple the implementations on the SIP and DSAN sides. We also show how we exploited some features of ZigBee to improve the integration with the SIP and the SHG.

Then, the paper reports an experimental study involving Wi-Fi, ZigBee, and Bluetooth networks. The goal of this study is to characterize the performance of the SHG in terms of the coexistence of the three systems, especially because they are all active in the same time and space, that is, in the prototype SHG board, and thus subject to strong mutual interference.

2. Related Work

In this section we just draw a sketch on the current state of the art in domotics systems and interference studies, with specific focus on the works whose topic is most similar to ours. The differences that make our contribution innovative are also pointed out.

Starting from the domotics area, some authors approached the integration between WSNs and control plane protocols by bringing customized or reduced versions of SIP or REST on the sensor nodes. For example, Luckenbach et al. [5] employed REST to provide clients connected to the Internet with the ability to directly interact with MICAz sensors. Similarly, Krishnamurthy and Lange [6, 7] proposed TinySIP, an architecture to offer to multiple clients the access to sensor-based information via SIP.

These kinds of approaches suffer from a series of drawbacks. Since the device is resource constrained, the protocols must be stripped of many functionalities. Due to the particular operative system running on the sensor nodes, the development times might be nonnegligible. Also, given the high heterogeneity of the devices, it might be necessary to repeat and modify the customization and development steps for every technology that is going to be integrated into the system. Finally, compatibility with deployed hardware and software is not retained. Conversely, our framework moves the development effort to a single high-end device (the SHG), allowing faster implementation times and full compatibility with both existing sensor and actuator devices and also with the SIP.

An approach that follows the philosophy of making an open and flexible service platform can be found in [8], whose authors started their work from an architecture similar to ours.

Acker et al. [9] presented a concept of ubiquitous home and facility control that exploits the IP Multimedia Subsystem (IMS), a SIP-based control architecture considered by mobile network operators.

The work closest to ours is perhaps the one by Bertran et al. [10, 11], who tested SIP as a universal communication bus for home automation environments. A SIP gateway and a series of SIP adapters and interpreters have been implemented and deployed to make all devices SIP compliant. However, there are some aspects that may put our framework one step ahead.

Bertran et al. did not consider the issues with addressing and reachability of the single DSAN devices. Conversely, we designed a functional addressing scheme that greatly simplifies the user interaction and does not require the DSAN nodes to register to any SIP server or other additional entities. Then, we devised a way of keeping the compatibility not only with the DSAN elements, but also with the user terminals. This allowed us to provide the user also with functionalities that are not natively supported by his/her device. This paradigm can even be extended to ensure forward compatibility with new domotics services. Conversely, Bertran et al. did not pay much attention to this aspect. A third distinguishing point is in the adaptation between the SIP and the DSAN worlds. While Bertran et al. design a single software module to be put in the gateway, we perform this operation in two steps, via the DFA layer. This allows to decouple the implementation of the two domains, making the system more flexible. Finally, we studied in much more detail the integration with two possible DSANs, namely, ZigBee and Bluetooth, and showed how it is possible to exploit their features to simplify the integration into the system. In [11], the main focus of the experimental platform was on the performance figures of the gateway (which, if we consider the current hardware technology, might not be the most relevant hurdle to the domotics development, as proved by our tests in Section 7.2).

A major disadvantage that is common to proposals like [8–10] is the need for every home device to register with its own URI. When the number of devices increases (heavily monitored and automated buildings may have hundreds of nodes), the user capability of handling them through their URIs is clearly hampered. The same shortcoming applies to the zone manager solution proposed by [6], in which the majority of the communications are possible only by knowing the address of each gateway to which the sensors of interest refer. On the other hand, in our system the sole SHG must register to an external SIP server (unless the SHG itself implements a registrar) and we can mask the multitude of DSAN nodes by means of the "functional addressing" method.

As for the coexistence of multiple wireless interfaces, we can find numerous analytical and simulation studies, especially about the performance of ZigBee under the interference of Wi-Fi and Bluetooth (such as [12], just to cite one). The major shortcoming of these approaches is that due to the very complex nature of the wireless channel and environment, there is no measure of their agreement with the reality, and thus their actual utility is somehow limited.

Sikora and Groza experimentally obtained the PER of a ZigBee system under the interference of Wi-Fi devices, Bluetooth devices, and also a microwave oven [4]. However, the study is limited to a single source of interference (e.g., either Wi-Fi or Bluetooth), and also the analysis of the coexistence of ZigBee and Bluetooth is not complete, since the (actually very few) results have been collected in one direction only (i.e., Bluetooth over ZigBee). Nevertheless, an interesting observation in Sikora and Groza's paper is about the presence of notable discrepancies between the collected experimental data and the simulation results provided by the IEEE 802.15.4 task group.

A similar experimental study was led by Musaloiu-Elefteri and Terzis, who evaluated the loss rate of a ZigBee system under Wi-Fi interference [3]. Starting from this result, they developed interference estimators and distributed algorithms to dynamically change the ZigBee operating channel. This approach was proved to drastically reduce the loss rate of ZigBee networks.

The authors of [13] present the results of an empirical study on the coexistence between IEEE 802.11b and Bluetooth devices. However, the primary objective was to develop an analytical model to estimate the mutual interference, rather than characterizing it in real world scenarios. Hence, to build such models, the experiments were controlled through the use of attenuators, signal generators, and coaxial cables, thus resulting in a rather idealistic environment.

From the analysis of the cited works, it emerges that in all cases, even in [4], the authors studied the interference of no more than two systems at a time. A two-way experimental analysis of the simultaneous interference among Wi-Fi, Bluetooth, and ZigBee can be found in [14], which confirms the weakness of ZigBee and also shows that some supposed interference-free ZigBee channels are in fact affected by the presence of Wi-Fi transmissions.

However, in all cited works, the interfering sources are always placed in physically disjoined devices. On the contrary, devices such as the domotics gateways are expected to embed several wireless interfaces onto the same board. In such cases, the interference effect might be even greater, due to the electrical couplings on the board. The experimental measurement we carried out over our prototype SHG was aimed at filling this gap.

3. Domotics Requirements and the SIP Control Plane

The complexity of the domotics system demands for a series of requirements that allows an easy integration among the subsystems and the development of a "friendly" and always available user interface. A set of the major requirements is represented by the following list (see also [8, 15] for similar surveys).

(i) The domotics system must implement and provide a *request/response* paradigm to allow the user to send commands to the DSAN devices and possibly have a feedback. Commands can also be exchanged among the various domotics entities.

(ii) Both the user and the system should be promptly notified when events of some importance occur in the environment. Thus, the network is expected to support asynchronous and/or periodic *event notification*.

(iii) Commands and events suit the need of exchanging small amounts of data in very short times. The use of *sessions* would instead allow the streaming of various types of data over a period of time (e.g., audio and video, but also fast varying sensor readings or large file transferrals).

(iv) The extensive adoption of mobile devices such as smartphones and tablets has made the connection to the global network available everywhere. As a consequence, the user should be regarded as a *mobile user*, who would want to control his/her home from different places and via diverse access technologies (e.g., wireless LAN, cellular, ADSL).

(v) Despite the heterogeneity of the various domotics subsystems, the user would hardly be keen on using several and different human interface devices (HIDs), remembering the network addresses of every DSAN device, or learning technology-specific aspects of its domotics system. Conversely, it would be beneficial if the user could interact with a unique interface layer and associate mnemonic names to the devices and their functions (i.e., what we later call "functional addresses", e.g., the room where they are placed and/or the action they perform). Therefore, the domotics system should *integrate the subsystems* at both the technical and the user interface level.

(vi) While the domotics idea is slowly gaining field, *communication services* and *entertainment systems* had long since settled into everyday life. Therefore, the design should seamlessly include these services into the domotics platform (see [10] for some interesting examples).

Among the many options for realizing the common control plane (see, e.g., [15–19]), we selected the Session Initiation Protocol for its numerous advantages. From the conceptual point of view, which relates to the operations that are to be carried out by the control plane, SIP provides a set of *methods* that fit well the necessities of DSAN control and management as follows.

(i) The low overhead of the `MESSAGE` method (no set-up phase is needed) perfectly matches the requirements of the *request/response* operations.

(ii) A publish/subscribe-notify semantic is available in SIP specifications and allows the user to be promptly notified of events that occur in the network. This allows an almost direct mapping of asynchronous and/or periodic *event notifications* to SIP methods.

(iii) SIP has been natively designed to offer *session management* capabilities (i.e., session creation, modification, and tear down).

(iv) The core SIP infrastructure exploits the `REGISTER` method to transparently manage the movement of the user between different points of attachment to the network.

From a more practical and implementation perspective, we can identify the following key points.

(i) SIP is a text-based protocol: message building and parsing is a relatively simple task. SIP parsers and interpreters are widely available. Hence, the development effort is greatly reduced.

(ii) The body of SIP messages is flexibly structured and can contain a wide variety of information. This allows an easy extension of the protocol to support customized DSAN-related data and commands.

(iii) A huge SIP infrastructure is already deployed and working; hence, there is no need to deploy new infrastructural elements (either servers or core-network software).

(iv) SIP works at the application layer, being transparent to the underlying physical and networking technologies. It can thus work as a gluing layer for heterogeneous systems.

A further valuable asset of SIP is the use of *mnemonic names*. Every SIP resource is associated to a URI (uniform resource identifier), a mnemonic text pattern based on the same syntax established for web services. This allows the user to remember names rather than complex numeric addresses. A way of exploiting this feature is presented later on in the paper.

Despite its numerous advantages, employing SIP for the control plane of our domotics system does not come for free. There are several issues that must be solved as follows.

(i) SIP is defined by a series of RFCs that provide only general indications on the use of the standardized procedures. The application to practical cases is left to the implementer, and it clearly depends on the specific scenario. Hence, the usage of SIP might require a preliminary phase to map the existing methods and design complementary procedures that fit the application requirements. One such example is the Event Notification Framework, a standardized but empty framework in which we have defined a new package to be used in our domotics architecture (see Section 6.3).

(ii) To be effective, a control plane must be pervasive and its procedures supported by all devices forming the system. However, porting SIP on devices with minimal processing and/or storage capabilities, such as the sensors and actuators, is a nontrivial task that is often reduced to porting just a subset of the original methods and features (see, e.g., [6]). Clearly, this approach is not optimal and should be avoided in favor of a complete transposition of the available paradigms and/or capabilities.

(iii) Though SIP is a mature and relatively widespread technology, the majority of end-user devices employ SIP to support very few services. Designing a system under the assumption that all user devices can support all SIP methods is undoubtedly appealing, but quite unrealistic as well. Conversely, defining the procedures to allow the users to take advantage of these paradigms by means of their current terminals is a harder but definitely more sensible task.

3.1. Selected SIP Methods. In this subsection we provide a brief description of the SIP methods we used and how we integrated them into the domotics system. Note that the integration mode is not univocal and other mappings can be implemented. Therefore, particular attention is paid to the reasons that drove our choice, how these methods have been exploited, and how they interact with the other elements of the system.

3.1.1. Instant Messaging. The SIP `MESSAGE` method [20] is used to supply the real-time dispatch of short text messages where each message is independent from the others. A `MESSAGE` transaction requires no session setup and does not establish a dialog. The UA receiving a `MESSAGE` must send an immediate reply to the sender to inform it about the successful or failed reception of the message—in case of success, the answer is `200 OK`.

We used this real-time and low-overhead method to implement the *request/response* paradigm (see Section 3). In detail, the *request* is mapped to a first `MESSAGE` transaction, and the *response* is mapped to a second `MESSAGE` transaction. Therefore, four SIP messages are necessary to realize the *request/response* paradigm. A typical usage case of this method is illustrated in Figure 1.

A very important aspect of the `MESSAGE` method is its compatibility with all existing SIP clients. Since every SIP client must support this method, this ensures that the basic managing functions of our system are also supported.

3.1.2. Publish/Subscribe-Notify Paradigm. The SIP Event Notification Framework (ENF), defined in [21], provides a way for SIP elements to learn when "something interesting" has happened somewhere in the network. The procedures to

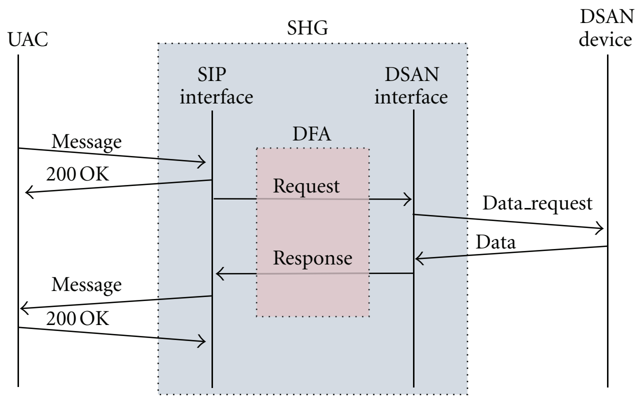

FIGURE 1: The *request/response* paradigm implemented via the `MESSAGE` method.

allow for the prompt distribution of such events are known as the Publish/Subscribe-Notify paradigm.

Briefly, an initial `SUBSCRIBE` message is sent by the subscriber (the user that is interested in the event) to the notifier (the node that is first aware of the event). If the subscription is accepted, a `200 OK` answer is sent to the subscriber. Then, the events are reported from the notifier to the subscriber by means of the `NOTIFY` method. Notifications can be sent either periodically or when the specific event occurs (or both).

SIP also provides a framework for the publication of event states on a notification server, called Event State Compositor (ESC). This task is accomplished using the `PUBLISH` method [22]. The ESC is then responsible for managing and distributing this information to the interested parties through the ENF.

The mapping of the complete Publish/Subscribe-Notify paradigm to the domotics architecture is shown in Figure 2. The figure shows both periodical and event-driven notifications.

Note that the ESC is a logical entity, which can physically reside in diverse parts of the system; in our prototype the ESC functions are provided by the SHG. In particular, the SHG is the only entity that publishes the events. DSAN devices are thus preserved from knowing anything about the SIP existence. In addition, the SHG can filter and compose events that are not available in the single DSAN domains.

3.1.3. Registration. In the proposed domotics architecture, just two elements must be registered: the user and the SHG. All sensors and actuators of the various subsystems are managed by the SHG via the specific DSAN interfaces. Thus they can be completely unaware of the SIP control plane. On the other side, the user can interact with the system by knowing just the SIP URI of the SHG and can refer to the DSAN devices through what we have called the "functional addressing" scheme (see Section 4.2), that is, a set of mnemonic names (such as the room names and the device functions). This makes the system extremely user-friendly and also highly scalable. No matter how many devices are in the house, the user can control them invariably

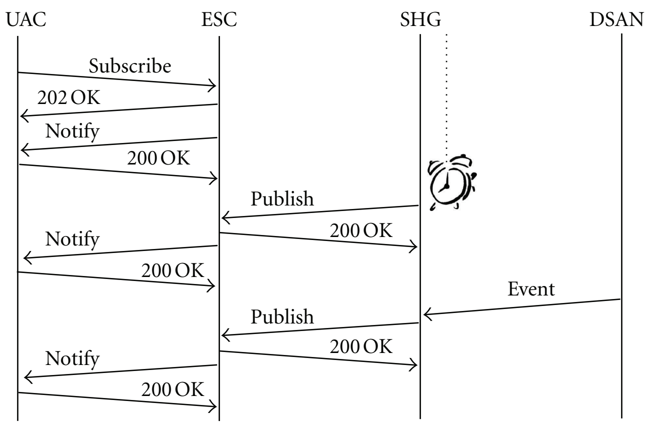

Figure 2: SIP message flow for publish, subscribe, and notify operations over the domotics system.

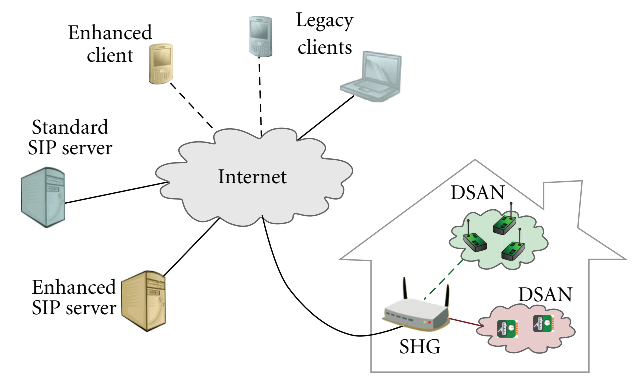

Figure 3: Reference architecture of the SIP-based domotics system.

through the same URI (the SHG one), from anywhere he/she is and from any SIP-enabled device he/she is using.

4. System Architecture

The general architecture of the conceived domotics system is illustrated in Figure 3. We can identify four major physical elements: the clients, the SIP servers, the SHG, and the DSANs.

The expert reader may have noticed that this kind of architecture is not completely novel: a similar picture has been presented, for example, by [8, 10]. This means that the scenario in which the domotics system is going to operate can be considered quite settled. Nevertheless, though addressing a similar architecture, the various works, and ours in particular, differ in several aspects, such as how the SIP is integrated into the framework and exploited by the designer, what semantics are taken into account, how they interact with the other components of the system, and how they can be beneficial to the user. Specifically, our approach targets a more scalable, transparent, and painless integration, both from the user perspective and from the DSANs' point of view.

The new and distinguishing elements of the architecture are described in the following.

4.1. SIP-Based Home Gateway. The SIP-based home gateway (SHG) is the key element of the system. It enables the remote control of the various DSANs by translating the messages and procedures from the SIP world to the specific DSAN technology and vice versa. Furthermore, it performs "intelligent" operations, such as piloting devices of a DSAN in response to events from another DSAN (e.g., turn on a KNX-enabled heater after a ZigBee sensor has reported a temperature/humidity change) and interpreting generic user commands and mapping them to device-specific actions.

The complete description of the SHG prototype is reported in Section 6.1.

4.2. Domotics Facility Abstraction. An intermediate entity, named domotics facility abstraction (DFA), has been introduced with a double goal: disjoin the implementation of the two domains of the system (i.e., the SIP and the DSANs) and create a single and user-friendly service abstraction. The former goal is meant to ease the development of the SIP and the DSAN interfaces, which may be carried on separately. The abstract definition of the domotics services enables the *functional addressing and control* paradigm employed for the user interface, leaving SIP on the pure transport layer, which is thus hidden to the user.

A pictorial description of the framework can be seen in Figure 4. The user is immersed into the functional service abstraction, which is implemented through the user interface on the client, and is understood and processed at the SHG. As it will be detailed in Section 6.1, the DFA is realized for the most part in the SHG, which stores the set of actions and performs the necessary tasks to accomplish the user's directives.

An example application of this framework can be swiftly provided. Imagine a user at a remote place (e.g., returning from a travel abroad) wishing to find the home at a comfortable temperature. He/she can then issue a simple command, such as "set home temperature 20." The client then wraps the command into the proper SIP procedure and conveys it to the SHG, where it is mapped to a DFA service. The SHG then cares for translating it into a proper set of DSAN operations, such as starting the HVAC system and setting an alarm threshold on the temperature sensors deployed in the house. When the desired temperature is reached, the SHG will automatically stop the HVAC system.

We can see from this example that the user demands a specific operation to be performed, but he/she is clearly ignoring all the technical processes of the domotics system, which are transparently handled by the SHG by means of the DFA layer.

4.2.1. System Configuration and Reconfiguration. Clearly, to unleash the capabilities of the DFA layer and the functional addressing and control scheme, a configuration phase must take place. In our vision, this phase can be split into two steps.

In the first step, occurring during the system development, the set of actions and user keywords must be defined and implemented. With reference to the previous example, the developer should make the SHG aware of the keywords "set," "home," and "temperature" and implement

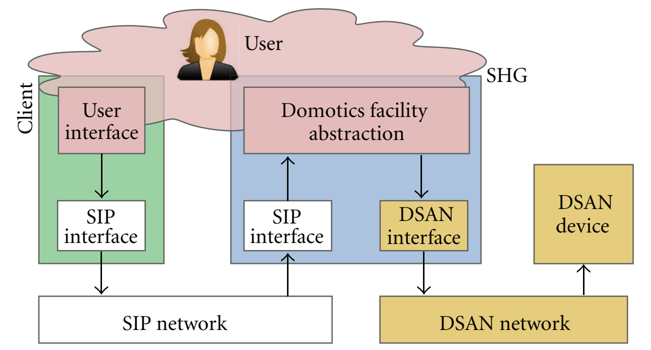

FIGURE 4: The functional paradigm implemented via the domotics facility abstraction.

the procedures that transform these keywords into real actions (such as sending a command to a HVAC actuator). Yet, these procedures cannot address a specific device, since the set of available devices will only be known at deployment time. Therefore the procedures can define just generic commands that become actual technology-specific actions once the DSAN devices are connected. For example, typical general HVAC actions could be "heat," "ventilate," and "cool." The completion of this phase defines the set of keywords and paradigms the user can take advantage of.

The second configuration step takes place at deployment time. Having a look at Figure 7, which shows the SHG internals, might help making the concept clearer. The installation-specific details, such as the plan of the house and the room names, are stored into the SHG database and/or file system. This task can be performed either by the installer or by the user. Offering a GUI to let the user install and/or configure the SHG might be a commercial choice. At the same time, the DSAN managers detect the connected devices and populate the SHG database, inserting information such as the device types, capabilities, and the actions they can perform. The physical position of the devices is inserted by the user/installer, after the devices have been registered with the SHG.

The system is now ready to work. When a user request is received, the SHG will map it to the appropriate action, search its database for the device(s) supporting that action, and issue the command(s) towards those DSAN devices. The full workflow of the SHG and its internals are described in Section 6.1.

Obviously, the information entered during the deployment phase can be modified later on, for example, as a consequence of device movement, replacement, or addition. Though being a more delicate operation, also the set of keywords and actions available to the user can be changed, for example, by upgrading the SHG firmware.

5. Wireless Technologies in the SHG

In this section we give a quick overview of the three wireless technologies, that is, Wi-Fi, ZigBee, and Bluetooth, that we have selected for the Home Network and hence, integrated onto the SHG prototype board. We also outline how these standards exploit the 2.4 GHz band and interact in this region of the spectrum.

5.1. Wi-Fi. The latest IEEE 802.11 standard [23] defines a CSMA/CA (carrier sense multiple access with collision avoidance) scheme as the mandatory medium access scheme. According to CSMA/CA, every Wi-Fi device shall listen to the medium before transmitting. The transmission is allowed only if the medium has been sensed idle for a predefined time period. In case the medium is sensed busy or after a collision, the device shall refrain from transmission for a period whose length is determined by a random variable (exponential backoff).

An IEEE 802.11 network can operate over one of the 11, 13, or 14 channels defined for the 2.4 GHz ISM band (the exact number depends on the local regulations). Each channel is 22 MHz wide, and the channels are partially overlapped (since the overall ISM bandwidth is just above 80 MHz). Therefore, no more than three networks can be contemporaneously operated in the same area in order to keep the transmissions of each free from interference from the others.

The operative channel and the transmission power are generally set statically (e.g., by the manufacturer or by the user at configuration time), even though dynamic channel selection (DCS) and transmit power control (TPC) routines have been defined for operations in the 5 GHz band. In the 2.4 GHz band, the maximum transmission power is 100 mW (20 dBm) in Europe and 1 W (30 dBm) in North America; in Japan, where power is measured in relation to bandwidth, the maximum allowed power is 10 mW/MHz.

Finally, the modulation scheme is either a DSSS (direct sequence spread spectrum) for the lower bit rates or an OFDM (orthogonal frequency division multiplexing) for the higher ones.

5.2. ZigBee. The IEEE 802.15.4 standard [24] specifies the physical and medium access control layers for low-rate wireless PANs, targeting a 10-meter communication range with a transfer rate of up to 250 kb/s.

Similar to Wi-Fi, 802.15.4 devices employ a CSMA/CA channel access algorithm and the DSSS modulation (actually, the latest release of the standard defines four modulation schemes, but in the 2.4 GHz band only the DSSS modulation is allowed).

Sixteen channels are defined for worldwide use in the 2.4 GHz band. However, differently from 802.11, they are much narrower (just 2 MHz) and do not overlap, so that up to sixteen 802.15.4 networks can easily coexist in the same area. When starting a new network, an energy detection (ED) functionality is used to determine the activity of other systems and thus decide the operating channel; yet there is no support for dynamic channel selection.

The latest ZigBee release has introduced the support for frequency hopping in the "ZigBee Pro" standard. In this way a PAN coordinator can move the whole PAN to another channel if the one in use is overloaded. However, this is not a fast, reliable, and energy saving way to solve the problem. In addition it is not mandatory to implement.

5.3. Bluetooth. Bluetooth is a standard communication protocol designed for connection-oriented services such as voice, with low power consumption and short-range operations. The output power depend on the device class, spanning from 1 to 100 mW. Accordingly, the expected range should go from 1 to 100 meters, even though the practical range is highly variable.

Bluetooth transmits on up to 79 channels in the 2402–2480 MHz range. Each channel is 1 MHz wide, and one guard channel is used at the lower and upper band edges. In order to reduce the interference from external sources, frequency hopping (FHSS) is used to spread the signal across all channels. Thus a single Bluetooth network uses the full available 2.4 GHz ISM band. Different networks can coexist in the same area by employing different hopping patterns or a time-shifted version of the same pattern. Since Specification v1.2, Bluetooth also includes an adaptive frequency hopping (AFH) scheme, which reduces the number of employed channels to improve its robustness against the interference.

The Bluetooth channel access procedure is based on a master-slave scheme, which is built on the top of a time division duplex (TDD) transmission scheme. The basic modulation is Gaussian frequency-shift keying (GFSK), which allows a transfer rate of up to 1 Mb/s. Since the introduction of the enhanced data rate (EDR) with specification v2.0, $\pi/4$-DQPSK (differential quadrature phase shift keying) and 8-DPSK modulations may also be used, bringing the data rate to 2 and 3 Mb/s, respectively.

5.4. Channels, Frequencies, and Modulations. Figure 5 shows the allocation of the ZigBee and Wi-Fi channels over the 2.4 GHz ISM band. Note that a single 802.11 channel completely overlaps with four ZigBee channels. Bluetooth channels are not reported, as the FHSS covers the whole available spectrum.

The three most used nonoverlapping Wi-Fi channels are 1, 6, and 11. In this case, two ZigBee channels should be free from interference from Wi-Fi transmissions, that is, channels 25 and 26 (the two rightmost ones). However, there is no assurance that using channels 25 and 26 solves the interference problem. For example, two channels might not be enough to allow the coexistence among several geographically overlapping PANs. In addition, though in North America ZigBee channels 25 and 26 can be really assumed free from Wi-Fi transmissions, in other regions such as Europe and Asia all Wi-Fi channels can be used, thus covering the complete set of ZigBee channels.

A further aspect making the coexistence of Wi-Fi and ZigBee difficult is the different allowed transmission power. In fact, the maximum Wi-Fi output power can be up to 100 times higher than the maximum allowed ZigBee transmission power (100 mW versus 1 mW). The same consideration holds for Wi-Fi and Bluetooth devices belonging to Classes 2 and 3.

6. Proof of Concept

To put the ideas expressed in the previous sections into practice, we have realized a small testbed involving all the elements of the architecture. The SHG, being the core and most innovative element, has been built from scratch. Two DSANs have been implemented using two sets of ZigBee and Bluetooth devices. Finally, to illustrate the potentials of expansion and customization of our architecture, we have defined and implemented the "home automation" package, a specific SIP event package for the domotics framework.

6.1. SIP-Based Home Gateway. The only requirements for building the SHG are the sufficient processing power and memory to run the software and the capability to interface with the technologies of the particular sensor networks to control.

With regard to the former aspect, we used a generic single-board computer (SBC) with a Texas Instrument AM 3730 processor (ARM Cortex-A8) running at 720 MHz with 256 MB of DRAM and 256 MB of NAND flash memory. As it will be shown in Section 7.2, this hardware is more than adequate. To give the SHG the physical interfaces towards the wireless networks, a ZigBee module has been embedded into the board and connected to the main processor via a serial interface; then a Hama Bluetooth adapter and a Wi-Fi card were inserted into the two USB ports. Figure 6 shows the prototype SHG.

The SHG software was built on top of Linux (with kernel 2.6.36), which provides the necessary support and development tools (e.g., a SIP library, the interface drivers). The software that implements the SHG functionalities has been written from scratch using the C++ language and then cross-compiled for the ARM platform. A multithreaded approach has been followed. Each user request is handled in parallel by a different thread. This helps improving the scalability performance of the SHG.

The internal software architecture of the SHG is reported in Figure 7. Starting from the top, the first object we meet is the SIP interface. This is nothing more than the SIP software (the GNU oSIP and eXosip libraries), which extracts the user's commands from the SIP messages and passes them to the next module in the form of plain text strings. These are then translated into the proper DFA actions by the translation module, which fetches the set of available DFA actions from the DFA Library. The output of this module is fed to the SHG engine, where we have placed the intelligence for executing the user's directives in the proper way. This is typically achieved via the creation of a series of elementary DSAN commands to be delivered to the various DSAN elements. The set of available elements and commands is retrieved from the device database. The SHG engine then passes the DSAN commands to the various DSAN managers, which are in charge of translating them into the technology-specific commands and performing all the operations to ensure that the specified actions are fulfilled. Finally, the ZigBee, Bluetooth, and Wi-Fi interfaces are the software modules (a custom software for ZigBee, the BlueZ stack for Bluetooth, and the Linux drivers and tools for Wi-Fi) that

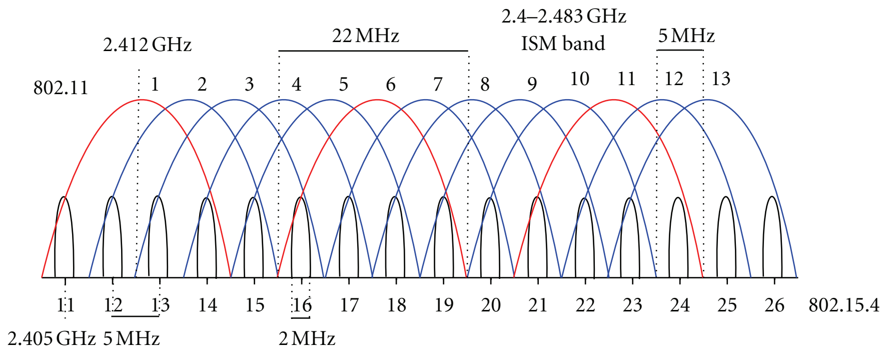

Figure 5: Channel occupancy of 802.11 and 802.15.4 systems.

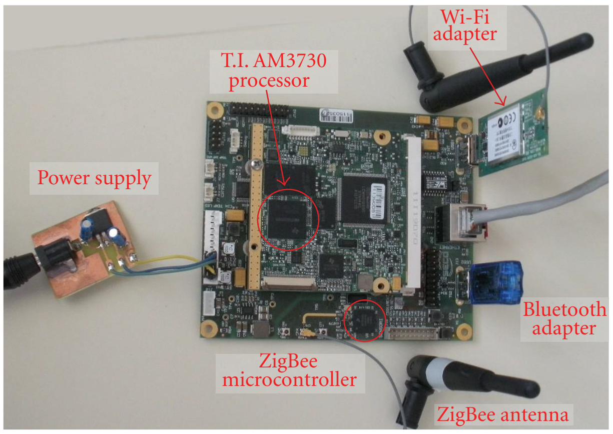

Figure 6: A photo of the SHG prototype, with the indication of the main components.

pilot the physical objects that are connected to the various sensors and actuators.

The device database (DdB) holds the set of available DSAN objects, with the related properties (e.g., commands, location, technology). The DdB is filled and kept up to date by the DSAN managers, which are aware of the number and types of devices connected through the various DSAN interfaces. Further information, such as the physical location of each device, can be inserted at configuration time either by the user or by the service provider.

The operations in the reverse direction, that is, from the DSAN networks to the SIP interface, are analogous to the ones mentioned above. The notifications from the sensors are passed, by means of the DSAN managers, to the SHG engine, which decides what actions are to be taken. For example, a new command might be issued towards the DSAN, or an information message can be sent to the user (or both). In the latter case, the message is passed to the translation module and finally to the SIP interface.

6.2. The Wireless Networks.

6.2. The Wireless Networks. This section briefly describes the setup of the two DSANs and of the Wi-Fi local area network. A few essential technical details are also given.

6.2.1. The ZigBee DSAN. The nodes of the ZigBee sensor network are based on the Freescale MC1322x board, which integrates a 32-bit ARM-7 MCU and a low-power 2.4 GHz transceiver. The fully compliant ZigBee stack provided by Freescale was installed on the nodes.

A custom application that supports environmental data collection (temperature and pressure), remote light control, and message routing has been developed on top of the ZigBee stack by means of the ZigBee Cluster Library (ZCL) functions. The APS ACK feature (an end-to-end acknowledgment mechanism) was enabled to make the ZigBee transmissions reliable.

Ambient data is retrieved both on regular time basis and on demand, and both approaches are available to the user, who can either subscribe to this event or ask the SHG to check a specific sensor value. As for remote light control, the MC1322x boards are equipped with an array of LEDs, which was used to mimic a multilevel light. For both ambient data collection and light control, we defined a set of textual commands. Combining them with the name of a room allows the user to set the desired light level or retrieve the sensor reading.

An important aspect of the ZigBee system is that it provides for a mechanism, known as *binding*, to connect endpoints (an "endpoint" is a logical wire connecting distributed applications residing on different nodes). Binding creates logical links between endpoints and maintains this information in a binding table. The binding table also has information about the services offered by the devices on the network. The ZigBee coordinator (ZC) typically holds the binding table for the whole network. A notable advantage of this structure is that it allows the implementation of the *service discovery* procedure via bindings. The services available inside the ZigBee network can thus be discovered directly within the ZigBee domain, without resorting to any additional software or external entities. With specific reference to the SIP control plane, this means that there is no need to port the SIP registration procedure to the ZigBee network, since this would be a duplication of the ZigBee service discovery.

6.2.2. The Bluetooth DSAN. The Hama Bluetooth adapter connected to the SHG board embeds a version 2.0 compliant

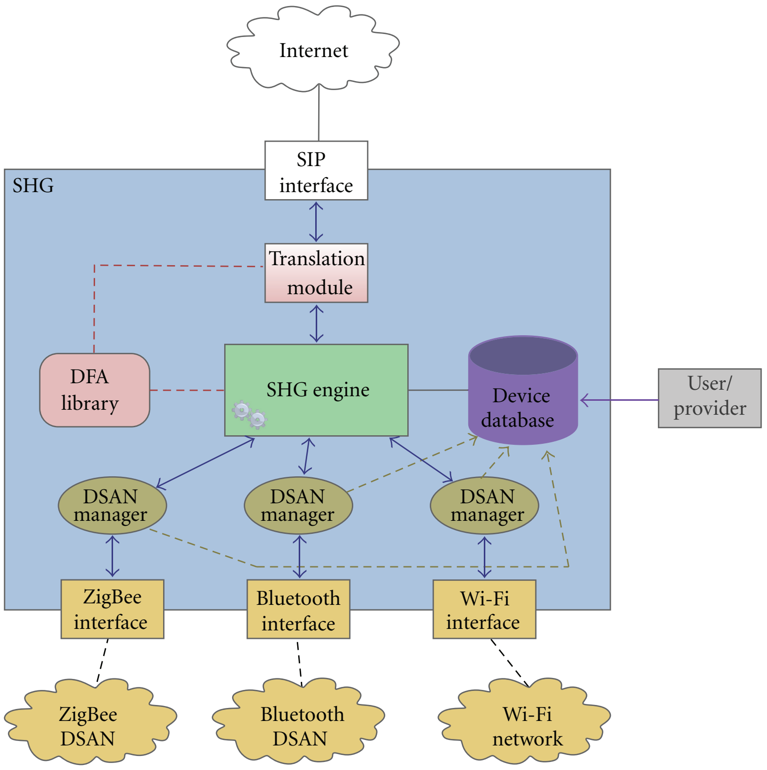

Figure 7: The software modules building the prototype SHG.

chipset supporting the EDR feature. It is a Class 2 device, with 2.5 mW (4 dBm) of output power allowing for an approximate range of 10 meters and a physical bit rate of 3 Mbps. To operate this device, we took advantage of the Bluetooth Linux stack (BlueZ), which gives support for basic operations such as scanning and pairing. On top of these basic functionalities, we built the Bluetooth interface, which is capable of listing and managing the connected devices.

For the purposes of validating the domotics system, we set up an audio streaming test. A Bluetooth-enabled headset (Sony DR-BT101) was used as the client device. The audio streaming was handled directly by the BlueZ (on the SHG side) and the headset, by means of the A2DP profile.

On the SIP-based control plane, we defined and implemented some simple commands, such as listing of the available content and playing an audio stream on the Bluetooth headset.

6.2.3. The Wi-Fi Local Area Network. To build the Wi-Fi LAN, we used two adapters based on the Ralink RT3572 chipset, a IEEE 802.11a/b/g/n compliant card. One of the adapters was installed on the SHG and the other on a common laptop PC. The drivers from the latest “compat-wireless” package have been used to pilot the card.

We set up a private IBSS network on one of the 2.4 GHz channels. The use of an IBSS topology rather than an “infrastructure” one is justified by the shorter set-up times (mostly in terms of driver and software configuration) but has no impact neither on the traffic at the transport and application layers nor on the physical layer mechanisms. The SHG and the PC are therefore two “peer” stations.

Traffic on the Wi-Fi connection has been generated by means of common test applications, such as FTP or iperf.

6.3. A Domotics Event Framework. The SIP Event Notification Framework (ENF) standardized in RFC 3265 [21], and later augmented by RFC 3903 [22], provides just the procedures that enable notification of events (as outlined in Section 3.1) but do not define any specific “event package.” Indeed, a few packages for the SIP ENF have currently been ratified. Among them, the Presence package [25] is probably the most popular and also the one that is implemented in some widely available SIP clients. However, this package does not suit well the needs of a domotics environment, as it provides just a single elementary functionality (the presence of a given user) and refers to the bare ENF, without taking advantage of the `PUBLISH` method.

To test our domotics system with a complete and flexible Publish/Subscribe-Notify paradigm, we built a new package, named “home automation.” The basic features of home automation are similar to the ones of Presence, but our package employs the PUBLISH method too. We designed it to embed a customized XML text, like the one illustrated in Figure 8, which contains domotics specific data (such as the values read by some ambient sensors). In this particular example, the XML snippet is sent from the SHG to an *enhanced client* to report about the readings of the sensors in the *Lab* room and also the current light level. Note that

```
<device room = "Lab"
      name = "dimmablelight">
    <attrib name = "Level"
          value = "33">
    </attrib>
</device>
<device room = "Lab"
      name = "ambientsensor">
    <attrib name = "Temperature"
          value = "20">
    </attrib>
    <attrib name = "Pressure"
          value = "308">
    </attrib>
</device>
```

Figure 8: Figure 8: Sample XML for the home automation event package.

the tags implement a possible functional naming abstraction of the DFA layer. Clearly, this XML scheme can be replaced with any other kind of text format, like REST or SOAP (the ones employed by the ZigBee Gateway [26]).

In order to correctly handle this package, we also built a customized ESC server. We employed Kamailio, an open-source SIP server released under GPL, to which we made some modifications. The changes mainly consisted in adding the specific home automation keywords to let it recognize the home automation package in a similar fashion to any other package.

Note that the XML text is not touched by the ESC (only a formal check is done) but is passed directly to the subscribers by means of the `NOTIFY` messages. Hence, SIP is immediately able to deliver this information using the existing infrastructure.

6.4. SIP Clients. We developed a test SIP client that supports the full Publish/Subscribe-Notify paradigm and the Home Automation package described in Section 6.3. One such client is in all aspects a SIP-compliant software, but with the extra feature that can control the DSAN with its native semantic.

6.5. SIP Servers. Servers build the necessary infrastructure for SIP to work properly. In our proof of concept, we employed two different servers. An external registrar and a proxy server provided by *iptel.org* were used as a sample of a preexisting SIP network element. This server is compliant to existing SIP standards and is completely unaware of the nature of our domotics testbed.

A customized SIP server was built in our lab by means of the Kamailio open-source software. As explained in Section 6.3, this was necessary to provide support for our home automation event package. Hence, this server is representative of a domotics-aware element in the SIP infrastructure. We called this server the enhanced notification server (ENS).

7. Performed Tests

The performed tests can be divided in two sets. The first series was aimed at assessing the performance of the prototype SHG in terms of capacity, scalability, and processing delay. The objective of the second set instead was to verify the amount of interference among the wireless interfaces on board of the SHG and the impact on the SHG performance.

Before discussing the tests, we outline the deployed networks and the environment where the tests have been carried out.

7.1. Network Topology. All tests have been carried out within the premises of the Dipartimento di Ingegneria dell'Informazione of the University of Pisa, Italy. This might indeed constitute a good environment for both kinds of tests: applications such as smart energy, building automation, and intrusion detection systems fit well this kind of structures, and we might indeed expect to find in the department several devices using different radio technologies working at the same time.

The realized testbed is made of five ZigBee sensor nodes, including the ZigBee coordinator (ZC), which is embedded in the SHG board as already shown in Figure 6. The physical location of the nodes is illustrated in Figure 9. All ZigBee nodes have a wireless path to the ZC. Due to the indoor environment, the nodes *Stairs*, *Office*, and *Corridor* use a multihop path. The Bluetooth headphones (*bths*) are placed in the same room of the SHG, approximately 8 meters apart; the PC acting as a Wi-Fi station (*sta1*) is placed in a room adjacent to the one with the SHG. We checked that the Bluetooth and Wi-Fi devices, as well as the *Lab* node, are within the operation range of the SHG. *sta0* represents the Wi-Fi adapter connected to the USB port of the SHG.

7.2. Performance of the SHG. We assessed the performance of the SHG in terms of two metrics: the number of served user requests per second (in short: SURPS) and the average response time.

To compute the first metric, we connected the SHG to a varying number of clients through our 100 Mbps local area network. Every client was programmed to send a continuous flow of 100 requests using the `MESSAGE` method. Each request is cast as soon as the previous response is received from the SHG (we recall that a response is implemented with a distinct `MESSAGE` transaction). In this way the SHG always has a pending request to process for each client. The auxiliary SIP procedures, such as registration, have been excluded in order to measure the raw SHG capacity. For the same reason, we did not connect the SHG to any real DSAN but implemented a fake interface that returns a response as soon as it receives a command. In practice, with reference to Figure 7, the processing path stops at the ZigBee interface. The Bluetooth and Wi-Fi networks were left inactive.

The collected numbers of total SURPS and mean SURPS per client, averaged over ten experiments, are reported in Figure 10 as a function of the number of connected clients. Focusing on the red lines (labeled "eXosip"), we can see that

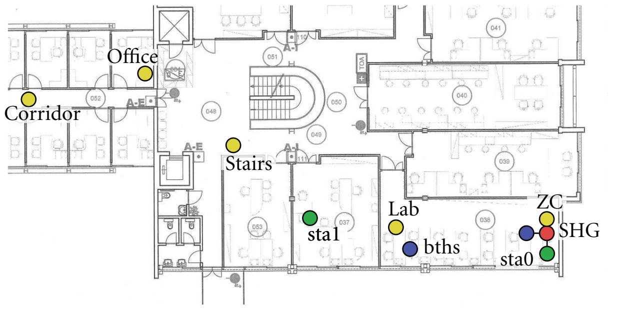

Figure 9: Map of the Dipartimento di Ingegneria dell'Informazione with the position of the ZigBee, Bluetooth, and Wi-Fi nodes (yellow, blue, and green discs, resp.); the SHG is also shown (red disc).

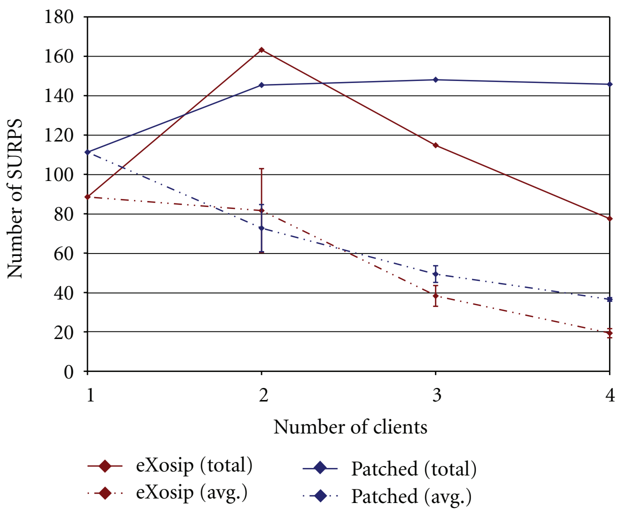

Figure 10: Total and average number of user requests per second served by the SHG; the standard deviation among the clients is also reported.

when a single client is connected, this can enjoy a service rate that is around 89 requests per second. This number can undoubtedly be deemed adequate not only for any human-based activity but also for any sensible automated application (see, e.g., [11]). In case of two clients, the number of total SURPS is almost doubled, but when further clients are added, there is a sudden performance drop. When four clients are connected, the total SURPS are even less than the single client case.

We have explored the reasons for such a tremendous degradation and found that it was due to the eXosip library integrated into the SHG. Without delving into the software details, this library presents some structures and timeouts that slow down the entire system when eXosip is called to serve many requests at the same time. We thus devised a simple patch that bypasses these shortcomings and repeated the SURPS test.

The results for the amended version are also shown in Figure 10 (the blue lines, labeled "patched"). The performance of the SHG has improved for almost any number of connected clients (it has slightly worsened for the two-client case only). More remarkable, however, is the fact that the trend in the number of SURPS is now much smoother and, above all, that the total number of SURPS reaches a stable level—it floors to about 146 SURPS. This means that the performance of the SHG is not appreciably influenced by the number of clients. Hence, we can reasonably affirm that the SHG can scale to serve many requests from different clients at the same time.

With the second performance test, we analyzed the behavior of the SHG from an internal point of view. We measured the time that elapses between the reception of a request `MESSAGE` and the issue of the response `MESSAGE`. The measurement was carried out with a single client, with the auxiliary SIP procedures at work, but still with the Bluetooth and Wi-Fi networks kept idle. We used two configurations. The first one is still based on the fake interface, whereas the second setting is an operative scenario with a real ZigBee DSAN attached. To keep the things simple, however, the ZigBee DSAN is composed of two nodes only: the coordinator (physically soldered to the SHG board) and a device node in direct communication range. The ZigBee network operated on channel 25, which is the most free from external interference sources.

The first row of Table 1 shows the processing times of the SHG in the two configurations. The difference is apparent, with them being apart by almost two orders of magnitude. This result does not leave much room for speculation and clearly identifies the bottleneck of the system with the domotics sensor network.

In the second row of Table 1, we have reported the performance of the SHG software when run on a generic PC based on an Intel Core i3 processor running at 2.66 GHz with 4 GB of RAM. The purpose of these figures is to provide a comparison with a "high-end" hardware. The PC is somewhat slower in running the software but is faster when the actual ZigBee network is attached. The "software" gap can be ascribed to the scarce optimization of both the code and the hardware, whereas the "DSAN" gap comes from the different connection with the ZigBee coordinator: serial (slower) on the SHG prototype and USB (faster) on the PC.

7.3. Effect of Interference. To check the effect of the interference on the domotics system, we set up a sort of "use-case" scenario. We assumed that the Wi-Fi and Bluetooth networks are used to deliver different but massive data to the user(s). Specifically, a FTP or HTTP transfer is conveyed over the Wi-Fi connection, whereas an audio streaming is performed by means of the Bluetooth devices. The SHG thus acts as the source of the data, and its Wi-Fi and Bluetooth interfaces work mainly as transmitters. Such a scenario can be mapped, for example, to a file download from the Internet (the FTP transfer) and a user listening to a song retrieved from a local repository (the audio streaming). As for the ZigBee DSAN, this is used to send commands to the sensors spread across the house. We exploit the *request/response* paradigm implemented via the double SIP `MESSAGE` transactions (as illustrated in Section 3.1.1). Hence, the SHG and the sensors alternate the roles of source and destination of the traffic.

Table 1: Processing delay of the SHG.

Hardware	Fake interface	Real ZigBee DSAN
SHG prototype	11.0 ms	177 ms
i3-based PC	34.8 ms	103 ms

Since the weak link in the chain is the ZigBee connection, our effort was mostly targeted at measuring the effect of the two "stronger" technologies, that is, Wi-Fi and Bluetooth, on the performance of the ZigBee subsystem and consequently on the capability of the user to control and have feedback from the ZigBee DSAN.

The test was organized in a similar manner as the performance experiment described in the previous section. A SIP client sends a continuous flow of 100 requests to the SHG via the local 100 Mbps Ethernet LAN. Each request is sent as soon as the previous response is received. No requests nor responses are lost in this segment of the system. The auxiliary SIP procedures (registration, publish, etc.) have been disabled, as they do not have any influence on the radio interference. The SHG then translates and casts the requests over the ZigBee network. Only one ZigBee sensor node is used for the test. This is sufficient for the purposes of the test, as the interference mostly occurs in the first wireless hop.

As for the wireless segment, we placed the Wi-Fi network on channels 1 or 11 and the ZigBee DSAN either on channel 25 or on channel 15. We did not test the system under overlapped ZigBee and Wi-Fi channels because we believe it is logical to assume that a deployed system will be smart enough to avoid such clearly troublesome kind of allocation. Also, we did not test the full range of possible combinations, which is not the purpose of the present work—the reader interested in this kind of analysis can refer to [14].

We collected four performance parameters: the average command service time registered on the client (in short: service delay), the average and the peak command execution time on the ZigBee interface (in short: execution delay and peak delay), and the number of lost commands (i.e., either lost requests or lost responses; we made ourselves certain that the losses can only occur on the ZigBee network).

To monitor the activity on the 2.4 GHz spectrum, including possible external interference sources (e.g., other Wi-Fi networks), we used the AirView2-EXT ISM-band spectrum analyzer (http://www.ubnt.com/airview). A screenshot of the power level in the test area has been taken before performing every experiment, to check whether strong external interferences are present and thus avoiding biased results.

Table 2 reports the outcome of the tests. The first test, labeled "0", is a preliminary test, used to benchmark the system when solely the ZigBee network is active (on channel 25). We can see that no commands are lost and that the peak execution delay is just a few milliseconds greater than the average. This indicates that the behavior of the ZigBee network is quite stable. Also, the average service and the execution delays differ only by 3 ms.

In the next test (1), we activated both the Bluetooth and the Wi-Fi networks, with Wi-Fi placed on channel 1, that

Table 2: Interference performance of the SHG.

Metric	Test 0	Test 1	Test 2	Test 3
Service delay	179.4 ms	202.0 ms	835.9 ms	937.5 ms
Execution delay	176.5 ms	197.2 ms	824.4 ms	913.6 ms
Peak delay	181.4 ms	1268 ms	5538 ms	5291 ms
Lost commands	0	0	0	0

is, the farthest possible from the ZigBee one. In this case, the interference is mostly due to Bluetooth, which covers the whole 2.4 GHz band. The performance drop is apparent, with an increase of 12% in the average time. The peak delay is the value that changed most, as it is now almost seven times the average execution delay. Thus, the ZigBee network can still bring all commands to completion, but its response time has become quite unpredictable. In absolute terms, however, even the highest values (1.268 s) can be deemed acceptable.

In test (2) we moved the Wi-Fi emissions closer to the ZigBee ones; that is, we put Wi-Fi on channel 11. In theory, there is still no overlapping between the ZigBee and the Wi-Fi channels. But in fact the ZigBee segment is heavily penalized, as proved by the values in Table 2. The average delays reach almost 1 second, with the peak execution delay going beyond 5 seconds. For some applications these values might be critical, for the user annoying. Note, however, that no commands are lost.

The reason for these figures lies in the long timeouts and the numerous retries that are allowed at the ZigBee application and MAC layers. For example, the default application retry timeout is 1.5 seconds, and the allowed number of retries is 3, both at the MAC and at the application layer. Thus, the ZigBee network, which is highly hampered by Wi-Fi, can take advantage of several attempts to deliver each packet, and consequently the overall transmission time grows very large.

To have a confirmation that Wi-Fi interferes with ZigBee even in nonoverlapping channels, we repeated the test by moving Wi-Fi to channel 1 and ZigBee to channel 15. The numbers of this test (3), which are very similar and even worse than the previous ones, indeed corroborate this fact.

8. Conclusions

The paper presented an architecture and a home gateway for realizing a domotics system with heterogeneous devices and user terminals. The architecture is based on the use of SIP as the common control plane and is centered on the SIP-based home gateway. A functional addressing scheme and an abstract translation layer (called DFA) are used to make the underlying technology transparent to the user. The DFA is the glue between the DSAN domain and the SIP world and simultaneously allows to separate the implementation of the SIP and DSAN interfaces. In addition, by choosing to expose a single SIP URI to the user (the SHG one), the system increases the user-friendliness and can be easily extended to large deployments. Note that this single-URI approach is neither an intrinsic feature of SIP nor of the domotics concept itself. Rather, it is a notable advantage of the way we built our architecture and the SHG. The positive impact of this approach is greater as the network grows larger.

We have built a proof of concept that includes the prototype SHG, three standard ZigBee, Bluetooth, and Wi-Fi networks, a newly defined SIP event package, and a customized event state compositor.

The performance of the SHG has been assessed in terms of served user requests per second, processing delay, and average and peak service delay. The effect of having the three wireless interfaces on the same board that operate on the same frequency band has also been evaluated.

The results proved the SHG ability to support a considerable number of requests per second, also from a different number of clients. Thus, the developed prototype can indeed be employed for large deployments, as it does have the ability to scale to any realistic requirement.

On the interference side, it emerged that ZigBee suffers the presence of both Bluetooth and Wi-Fi. Yet, while the former technology produces just a relatively small performance degradation, the presence of Wi-Fi is definitely more cumbersome, as the ability of the ZigBee network to accomplish its task in short times is heavily hampered. Though the weakness of ZigBee is well known, it is remarkable that this occurs even when Wi-Fi and ZigBee operate on channels that are nominally separated from each other. Our experiments showed a tremendous performance degradation when ZigBee and Wi-Fi are on adjacent channels. Nevertheless, by means of a proper configuration, we have also proved that it is possible to avoid command losses.

Acknowledgments

This work was supported by the Italian Ministry of Instruction, University and Research (MIUR) under the PRIN 2009 Research Project GATECOM. The authors would like to thank Luca Boggioni and Alessio Del Chiaro for their help in developing the prototypes and running the tests.

References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo et al., "SIP: session initiation protocol," RFC 3261, Internet Engineering Task Force, 2002.

[2] K. Sohraby, D. Minoli, and T. Znati, *Wireless Sensor Networks: Technology,Protocols, and Applications*, John Wiley and Sons, 2007.

[3] R. Musaloiu-Elefteri and A. Terzis, "Minimising the effect of WiFi interference in 802.15.4 wireless sensor networks," *International Journal of Sensor Networks*, vol. 3, no. 1, pp. 43–54, 2008.

[4] A. Sikora and V. F. Groza, "Coexistence of IEEE802.15.4 with other systems in the 2.4 GHz-ISM-band," in *Proceedings of the IEEE Instrumentation and Measurement Technology Conference*, vol. 3, pp. 1786–1791, May 2005.

[5] T. Luckenbach, P. Gober, S. Arbanowski, A. Kotsopoulos, and K. Kim, "TinyREST: a protocol for integrating sensor networks into the internet," in *Proceedings of the Workshop on Real-World Wireless Sensor Networks (REALWSN '05)*, June 2005.

[6] S. Krishnamurthy, "TinySIP: providing seamless access to sensor-based services," in *Proceedings of the 3rd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, MobiQuitous*, July 2006.

[7] S. Krishnamurthy and L. Lange, "Enabling distributed messaging with wireless sensor nodes using TinySIP," in *Ubiquitous Intelligence and Computing*, J. Indulska, J. Ma, L. Yang, T. Ungerer, and J. Cao, Eds., vol. 4611 of *Lecture Notes in Computer Science*, pp. 610–621, 2007.

[8] M. Alia, A. Bottaro, F. Camara, and B. Hardouin, "On the design of a SIP-based binding middleware for next generation home network services," in *Proceedings of the OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE*, pp. 497–514, 2008.

[9] R. Acker, S. Brandt, N. Buchmann, T. Fugmann, and M. Massoth, "Ubiquitous home control based on SIP and presence service," in *Proceedings of the 12th International Conference on Information Integration and Web-Based Applications and Services (iiWAS '10)*, pp. 759–762, November 2010.

[10] B. Bertran, C. Consel, P. Kadionik, and B. Lamer, "A SIP-based home automation platform: an experimental study," in *Proceedings of the 13th International Conference on Intelligence in Next Generation Networks (ICIN '09)*, Bordeaux, France, October 2009.

[11] B. Bertran, C. Consel, W. Jouve, H. Guan, and P. Kadionik, "SIP as a universal communication bus: a methodology and an experimental study," in *Proceedings of the IEEE International Conference on Communications (ICC '10)*, May 2010.

[12] S. Y. Shin, H. S. Park, S. Choi, and W. H. Kwon, "Packet error rate analysis of zigbee under WLAN and bluetooth interferences," *IEEE Transactions on Wireless Communications*, vol. 6, no. 8, pp. 2825–2830, 2007.

[13] I. Howitt, V. Mitter, and J. Gutierrez, "Empirical study for IEEE 802.11 and bluetooth interoperability," in *Proceedings of the IEEE Vehicular Technology Conference (VTS SPRING '01)*, pp. 1109–1113, May 2001.

[14] R. Garroppo, L. Gazzarrini, S. Giordano, and L. Tavanti, "Experimental assessment of the coexistence of wi-fi, zigbee, and bluetooth devices," in *Proceedings of the 12th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WOWMOM '11)*, pp. 1–9, Lucca, Italy, june 2011.

[15] H. Schulzrinne, X. Wu, S. Sidiroglou, and S. Berger, "Ubiquitous computing in home networks," *IEEE Communications Magazine*, vol. 41, no. 11, pp. 128–135, 2003.

[16] D. Bonino, E. Castellina, and F. Corno, "Automatic domotic device interoperation," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 2, pp. 499–506, 2009.

[17] F. Genova, M. Gaspardone, A. Cuda, M. Beoni, G. Fici, and M. Sorrentino, "Thermal and energy management system based on low cost wireless sensor network technology, to monitor, control and optimize energy consumption in telecom switch plants and data centres," in *Proceedings of the 4th International Conference on Telecommunication-Energy Special Conference (TELESCON '09)*, May 2009.

[18] A. Brown, M. Kolberg, D. Bushmitch, G. Lomako, and M. Tthew, "A SIP-based OSGi device communication service for mobile personal area networks," in *Proceedings of the 3rd IEEE Consumer Communications and Networking Conference, CCNC 2006*, pp. 502–508, January 2006.

[19] D. J. Cook, J. C. Augusto, and V. R. Jakkula, "Ambient intelligence: technologies, applications, and opportunities,"

Pervasive and Mobile Computing, vol. 5, no. 4, pp. 277–298, 2009.

[20] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle, "Session initiation protocol (SIP) extension for instant messaging," RFC 3428, Internet Engineering Task Force, 2002.

[21] A. B. Roach, "Session initiation protocol (SIP)-specific event notification," RFC 3265, Internet Engineering Task Force, 2002.

[22] A. Niemi, "Session initiation protocol (SIP) extension for event state publication," RFC 3903, Internet Engineering Task Force, 2004.

[23] "IEEE Standard 802.11-2007," December 2007.

[24] "IEEE Standard 802.15.4-2006," September 2006.

[25] J. Rosenberg, "A presence event package for the session initiation protocol (SIP)," RFC 3856, 2004.

[26] The Zigbee Alliance, "ZigBee Gateway Standard," 2010, http://zigbee.org/Standards/ZigBeeNetworkDevices/Overview.aspx.