A Framework for Resource Selection in Internet of Things Testbeds

Michele Nati[1], Alexander Gluhak[1], Hamidreza Abangar[1], Stefan Meissner[1], and Rahim Tafazolli[1]

University of Surrey, Guildford, GU2 7XH, UK
{m.nati,a.gluhak,h.abangar,s.meissner,r.tafazolli}@surrey.ac.uk

Abstract. As the scale and heterogeneity of experimental environments increases, the selection of adequate testbed resources becomes a daunting task for the experimenter. Wrong choices or unexpected resource behavior can significantly decrease an experimenters productivity. These challenges are further amplified by the recent trend of moving testbeds from isolated labs to unpredictable real world environments to favor experimental evaluation under realistic conditions. This paper presents a framework for resource selection in large scale and heterogeneous Internet of Things testbeds, in order to support the experimenter with an increased understanding of available testbed resources, their expected behavior and topological relationships in the experimentation environment. Through an evaluation case study we demonstrate the effectiveness of our proposed framework.

Keywords: Testbeds, Resource selection, Internet of Things, Wireless Sensor Networks, Semantics.

1 Introduction

The promise of the Internet of Things (IoT) to bring new levels of efficiency and increased real world insights to a variety of business domains has fueled research in the recent years with the aim to make this vision become a reality. However, the lack of understanding how existing solutions can operate in various real world environments as well as the missing consensus on technologies and standards across different business sectors has hampered the wide scale deployment of IoT solutions. There is a growing need to evaluate, compare and benchmark emerging IoT solutions on larger scale and outside lab environments under realistic operational conditions and to mature these solutions further through experimentally driven research.

While suitable experimentation environments and testbeds are slowly emerging [1], there is still a lack of adequate tools in order to support user friendly and efficient experimentation in such environments [2] with reduced management complexity. In particular features such as increased heterogeneity of these testbeds and their scale represent severe challenges for the experimental users as well as the testbed management tools that are necessary to manage efficiently available testbed resources. For example the selection of an adequate set of experimentation resources for a planned experiment can be a daunting task in an environment of thousands of experimentation resources, with different capabilities and time-varying properties if the user is provided with non-intuitive user interfaces such as simple resource list which are commonly used in today's testbeds. Similarly testbed management tools must provide reduced complexity for testbed administrators to manage a potentially large number of experimentation resources. These tools must take into account the time-varying properties of testbed resources and their surrounding environment in order to provide users with reliable information for their selection and to optimize the re-use of the underlying testbed substrate for concurrent experiments.

In this paper, we present a framework for large scale and heterogeneous Internet of Things testbeds, which allows experimental users to efficiently select resources to fulfill specific experimentation requirements based on static and time varying properties of available testbed resources. More concretely we make the following specific technical contributions that are integrated into an holistic framework:

– We propose a mechanism that allows an experimental user to quickly verify whether there are suitable resources for an experiment based on static properties of the testbed. For this purpose an ontology for describing IoT testbed resources has been developed together with a semantic query mechanisms that identifies a suitable set of resources based on matching required properties to those of available IoT testbed resource description instances.
– We provide a tool that allows a user to visually explore the topology for the resource required for an experiment and further scope the exploration by specifying constraints such as time-varying properties of links between IoT resources.
– We provide mechanisms to efficiently update properties of resources in the system to closely match real world changes to the testbed infrastructure and surrounding environment.
– We describe an implementation of the above framework and evaluate its effectiveness on a case study in our testbed

The remaining paper is structured as follows. Section 2 surveys existing work, while section 3 provides an overview of our framework for resource selection, putting it into the context of IoT experimentation environments. Section 4 describes in more detail realization of our proposed framework components and underlying information models. In section 5 we evaluate the effectiveness of the proposed framework using a detailed case study and present conclusions in section 6.

2 Related Work

In the following section we briefly discuss related work in the field and how our work differs and improves on it. In particular we focus on how resource selection in existing Wireless Sensor Network and IoT testbeds is supported and how resources in these testbeds are described.

2.1 Resource Selection in Existing WSN and IoT Testbeds

The increasing trend in the field of Wireless Sensor Networks (WSNs) and IoT research to test the effectiveness of the proposed solutions on a real hardware deployments motivated the creation of a multitude of IoT testbeds and the development of numerous custom solutions for their management [2]. Most of the proposed solutions provide basic functions for executing experiments on top of a given set of resources, such as reprogramming resources with a given test image, resetting them in order to start and stop an experiment and to stream back debug messages generated at run-time by the selected resources to a central server, mainly through the use of a wired backbone infrastructure. Examples of such frameworks are MoteLab [3], TWIST [5] and their clones or evolutions, such as for instance the INDRIYA testbed [2]. These frameworks were mainly used to manage a set of homogeneous sensor nodes as experimentation resources and provided the user with little support in selecting suitable ones apart from a list-view providing not much more information than the unique identifier of the resource. Such approaches for resource selection put an increasing burden on the testbed user for larger testbeds with heterogeneous nodes.

In order to present topological relationships between nodes and environmental dependencies, some frameworks provide views to display a deployment map of the testbed resources, showing the hierarchy of the testbed infrastructure components [4] and/or physical links that may exist between different nodes [3]. While serving as decision aid for the experimental user, they are based on static information data bases that have been obtained through previous experiments or by manual user entry. Furthermore resource selection is still carried out manually through list-views, making it a tedious tasks for experimental users. In order to better capture the characteristics of heterogeneous resources, the WISEBED [6] framework defines an XML based language called WiseML able to describe testbed resources and topological links they may have. In order to select adequate experimentation resources, users must still either browse these WiseML descriptions or generated lists from these XML documents. Similarly, SWORD [17] represents an example of an XML based declarative resource discovery service for wide-area distributed systems, which successfully operates on top of overlay testbeds such as PlanetLab.

Our framework complements the above efforts by providing a holistic solution to simplify resource selection for large scale and heterogeneous IoT testbeds and corresponding framework management services.

2.2 Description of Testbed Resources

The increased heterogeneity in testbed hardware and the need to provide access to testbed resources in the context of federated testbeds has motivated more descriptive approaches of expressing resource capabilities inside of a testbed framework. Initial approaches made use of XML based description formats. Examples thereof are the aforementioned WiseML format or the RSpec [9] format of the ProtoGENI and PlanetLab control framework. The cOntrol, Management and Measurement Framework (OMF) developed within the ORBIT testbed, introduced a domain-specific language named OEDL [10], which allows to describe experiment specific resource requests through a Ruby based scripting language.

The potential of exploiting machine processing capabilities for the development of advanced testbed tools that increase the autonomy of testbed operation and productivity of the testbed users have seen semantic resource descriptions recently emerging. An early example is the Network Description Language (NDL) [11], its extension NDL-OWL and the Network Markup Language (NML).

There have been other attempts developing XML and RDF based resource descriptions for sensor networks, which are however not focused at modeling these as testbed resources. Examples are the Sensor Model Language (SensorML) [12], the Semantic Sensor Network specification of W3C [13] or OntoSensor [14]. These efforts mainly focus on the sensing capabilities and observations or how to expose sensors as service endpoints in web service architectures.

More closely to our work on semantic modeling of IoT testbed resources is the recent work of Ju et al. [7]. In their work the authors propose an ontology for describing heterogeneous resources of wireless sensor network called LENS (Language for Embedded Networked Sensing) which has been integrated within the testbed framework developed by KanseiGeni initiative [8]. In contrast, our approach does not propose an ontology model from scratch, but instead extends the recent W3C ontology on Semantic Sensor Network [13] by specifying further details pertinent to IoT testbed resources. While there are similarities in modeled concepts, there a differences in properties and how the relationships of concepts are expressed. Furthermore, the authors of [7] do not provide any details on the mechanisms to populate and maintain the static and dynamic properties of these models within the testbed framework during operation.

3 Overview of Framework Architecture

Experimentation on existing IoT testbeds typically follows an experimentation life-cycle that comprises different activity stages, including experiment specification, preparation of experimentation resources as well as the execution of experiments and the subsequent analysis of experimentation data [2].

One of the most critical task is the experimentation scenario design and the selection of appropriate testbed resources for the envisioned experiment. The latter is particularly challenging for large scale IoT testbeds such as the emerging SmartSantander facility [1] as the experimental user is confronted with thousands of possibly heterogeneous experimentation resources with different capabilities and specific connectivity characteristics, constraint by their deployment environment. This is further complicated by that fact that experimentation resources may fail or become temporarily unavailable for experimentation due to connectivity failure or other ongoing experimentation tasks. Furthermore time-varying interference levels at wireless experimentation resources due to ongoing experimentation at neighboring experimentation nodes or external sources may have an influence on the suitability of a particular experimentation resource.

Figure 1 shows a high level overview of the architecture of our proposed IoT experimentation framework. As can be seen from the figure, the framework functions and support tools are exposed through a graphical user interface called *TMON* towards the experimental users of the testbed. *TMON* allows simplified and user friendly access to a variety of different testbed services which are able to support the experimentation users during all stages of experimentation. This includes access to functions that assist the user during experiment specification and resource configuration phases but also during experiment execution and experimentation data analysis.

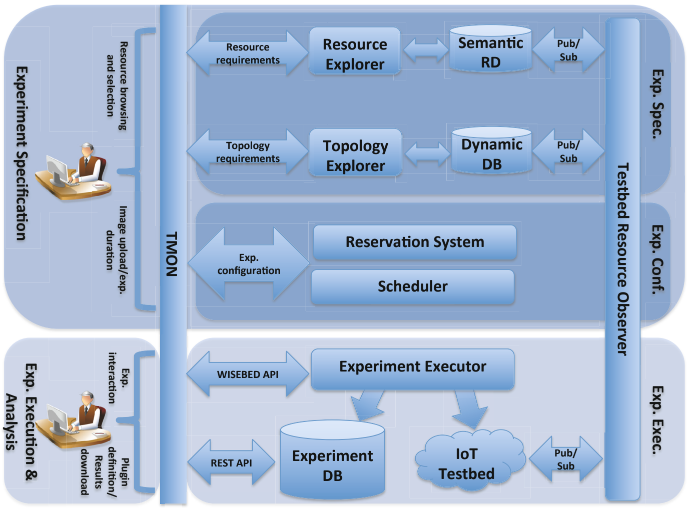


Fig. 1. Overview of the architecture of the proposed IoT framework for resource selection and management

For supporting the resource selection phase, the framework provides two dedicated functional components, the *resource explorer* and the *topology explorer* in order to assist the user with an exploration of available testbed resources and their static and dynamic properties and topological interdependencies. In our framework testbed resources and the preliminary static capabilities they provide are described by semantic resource descriptions, which are stored in a RDF data base. Through *TMON* a user can formulate visually queries for specific resource properties in order to satisfy the requirements for a particular experimentation scenario. The resource explorer evaluates these queries and performs a semantic matching against the semantic resource descriptions (RD), in order to provide the user with a selection of testbed resources fulfilling the desired properties. Once an initial subset of resources has been selected, the *topology explorer* allows a user to explore the topological relationships and characteristics of the links between nodes. As this information is quite dynamic and may change in particular for wireless links, it is updated regularly by the testbed management framework and kept in a separate database. This data base also includes other dynamic properties such as the interference levels experienced in different wireless channels. *Testbed resource observers* that are attached to each testbed resource are able to detect the availability of new testbed resources, as well as corresponding status and topology information and ensure that the information available to the proposed testbed framework functions are always up-to-date. The preliminary event based communication is realized through a publish subscribe messaging bus inside of the framework.

Once a user has selected a suitable set of experimentation resources, the experimentation specification is completed through *TMON* by provisioning of images for experimentation and the specification experimentation timing requirements. The experimentation configurations are passed to the reservation system and scheduler for execution of the experiment. An *experiment executor* controls the execution of experiments and allows testbed users to interact with the experiments. During the experimentation phase, experimentation results and traces are collected to an *experimentation database.* *TMON* provides the user with different views to the experimentation data, allowing quick visual inspection of the behavior of an experiment during execution or a detailed analysis after experimentation.

The discussions in the remaining paper will focus on the components and tools that allow efficient specification of testbed resources.

3.1 Resource Exploration

The purpose of the resource exploration tool is to simplify the discovery of available experimentation resources and the identification of a suitable sub-set for an envisioned experiment. The design of an adequate experimentation scenario is not an easy task and requires the user to have a thorough understanding of the underlying testbed resources, their capabilities and inter-relationships. For larger testbeds such information may be difficult to know or obtain upfront. Starting with a rough idea for an experimentation scenario an experimentation user will explore the availability of suitable resources and may iteratively adapt and refine the experimentation scenario to match the characteristics offered by a testbed environment. The availability of critical information for the resource selection process is crucial, so is the efficiency with which such information can be accessed and searched directly by the human experimenter or by tools supporting him.

In order to leverage the increased machine processing capabilities of the growing eco-system of the semantic web, one of our design decision has been to semantically describe our testbed resources. Instead of reinventing an ontology from scratch for our resource model, we have carefully evaluated existing state of the art ontologies in the sensor network and IoT domain and selected the Semantic Sensor Network (SSN) ontology [13] as a starting point. We have then extended

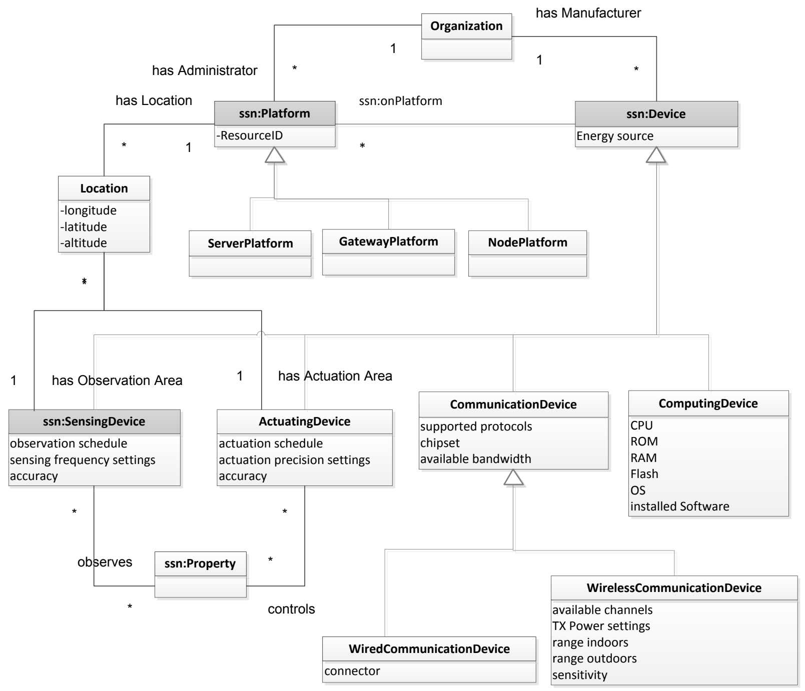


Fig. 2. UML diagram of the resource description model

the SSN ontology with concepts that are pertinent to IoT and sensor network testbed resources, including the most critical information that an experimenter may require for an adequate selection of testbed resources. Figure 2 provides an overview of the key concepts of SSN highlighted in grey and our proposed extensions.

Our proposed resource model takes into consideration the three possible device tiers that state of the art IoT facilities [2] are comprised of, by subclassing platforms for IoT node tier, gateway tier and server tier from the original SSN platform concept. A platform is deployed at a particular location, which has implication for most of the different devices that it hosts. Devices that are attached to a platform can be computing and communication devices and for the IoT node platforms often sensing and actuation devices. Communication devices can be wired or wireless. Sensing devices observe properties of their surrounding environment, which are constraint to a particular observation area. Likewise actuator may influence properties within their respective actuation area. The SSN ontology so far only specifies sensing devices.

For each of the concepts a variety of attributes have been defined that may be of relevance to the experimenter for resource selection. Experimentation code, e.g. a protocol implementation, is often developed based on a particular operating system, e.g. TinyOS and may have certain requirements on processing capabilities or available memory of the underlying node platform. It is therefore important to know whether there are nodes supporting the execution of the experimentation code. Some experiments may require specific sensing capabilities and/or cover specific geographic locations with these. In other cases experimenters may have a specific interest in communication devices available at IoT and GW nodes and what settings are possible for their configuration during experimentation. It is important to note that our proposed resource model is a starting point and will evolve as new hardware or software features become available or new experimentation use cases emerge.

The resource descriptions are created an maintained in a triple store, which can be queried through a SPARQL query engine. In order to hide the user from the details of constructing SPARQL queries, our resource explorer provides an front-end for visual query specification. These query specification contain the above described resource types available, constrained by desired properties and required numbers. The visual query specifications are then translated by the resource explorer into a set of SPARQL queries which are submitted to the query endpoint. Only those resources will be returned that satisfy the filters applied in the query. While gaining quickly effective feedback about availability of suitable resources, the user can use the identified subset as a starting point for further topology exploration.

3.2 Topology Exploration

Based on the nature of an experiment, an experimenter may be interested also to investigate the relationship among the resources and between the resources and the experimentation environment. The *topology explorer* provides the experimenter with an interface to explore these relationships based on an underlying physical network topology model as depicted in Figure 3 for IoT node platforms. A physical topology consists of node platforms and links. A link is defined between two communication devices that are hosted by a node platform. Each link is described by a source device and by a sink device, thus two links between to devices may exist with their specific properties. This is in particular the case for wireless interfaces where link behavior is inherently asymmetric. Furthermore in order to account for interference in an experimentation environment a corresponding interference module describes the perceived interference at the sink side of a link.

Examples of a topology visualization are provided in provided in figures 4 and 5. The topology explorer takes as input a node set and a particular setting configuration of the communication devices. For example, for an 802.15.4 based radio, the interface takes as input one of the 16 available channels and one of the transmission power level available for the radio chip featured by the respective resource. Based on this information, all the links connecting the selected resources are visualized. The topology explorer also offers the possibility to filter only the links with specific properties in terms of Packet Error Rate (PER), Link Quality Indicator (LQI), Receive Signal Strength Indicator (RSSI). A given

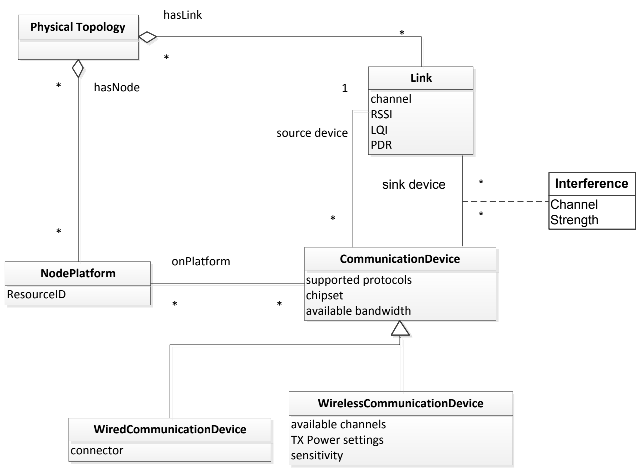


Fig. 3. UML diagram showing topological relationships between resources

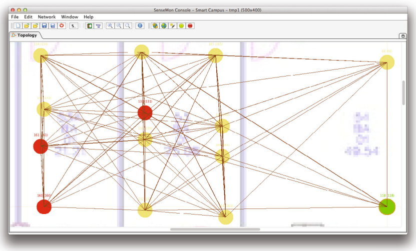


Fig. 4. TMON Topology Explorer, interference free topology

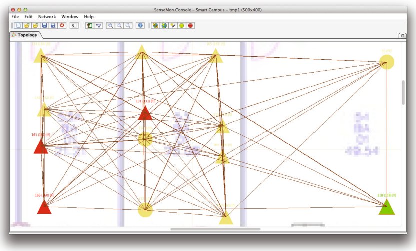


Fig. 5. TMON Topology Explorer, interference affected topology

pre-defined sets of topology exploration rules are also defined, implemented and accessible to the user in order to highlight nodes forming a connected component, nodes forming a clique, or all the nodes geographically connected to a selected destination sink. Further user-defined rules can be also easily defined based on a plug-in mechanism provided by the *TMON* interface, which offers APIs for accessing topology information and their visualization.

Figure 4 shows the nodes that are geographically connected (yellow nodes) to a selected sink (green node) with a path made by only nodes closer to the sink than the considered node and for a given selected configuration of channel (20), power level setting (7) and PER (0.9). Red nodes represent dead-end nodes in a geographic sense, while blue nodes (not present in the considered topology) are the disconnected ones, because no links between them and the rest of the topology exist for the selected parameter. All these provided information allow the user to gain a better understanding of the relations between resources and check if they match what it will expect to see during its experimentation phase in terms of topology connectivity, multi-hop path between a given source and destination nodes or other features.

Apart from displaying available links between nodes, a user can also explore existing interference affecting the links in an experimentation environment. For a scenario similar to that depicted above (channel 15, power 7, PER 0.9), Figure 5 shows the interference that was present during the characterization of the respective links by highlighting the occurrence of interference above a certain threshold level with exclamation mark symbols next to the affected links and by changing the node shape to a triangle. By adjusting channels and interference levels an experimenter can thus quickly determine where his experiment may be affected by an overlapping interferer in the environment. In our initial implementation, we consider as interference sources the presence of WiFi Access Point operating close to a resource in a range of frequencies overlapping with those selected by the user for its experiment. A range of configurable parameters for selecting the interfering sources based on the quality and power of the IEEE 802.11 signal affecting a given resource is also provided to the user. Our model can be extended to other interferer based on available detection mechanisms in the testbed.

The outcomes of both resource and topology exploration phase is a set of suitable experimentation resources that can be further reserved and configured for subsequent experimentation.

3.3 Resource Observation Plane

The resource and topology explorer are only effective, if the information kept in the underlying information models about the resources is up-to-date and reflects the current conditions in the testbed. This is the responsibility of the *testbed resource observer* in our framework. *Testbed resource observers* have two primary tasks. The first one is to keep track of the availability status of testbed resources in the testbed and detectable changes to resource properties and reflect those in the semantic resource descriptions. The second task is to keep continuously track of topology and environmental related information of the testbed, such as link characteristics or experienced interference levels in the surroundings and reflect those in the topology data base. In the following we describe the realization of each of the two functionalities in more detail.

The first time a new resource is added to the testbed, an administrator provisions a semantic resource description (possibly by customizing a pre-existing template for a resource type), which is added to semantic resource database. This ensures that only authorized resources are able to attach to the testbed framework. These descriptions are only removed, if an administrator decides to discontinue the use of a particular testbed resource.

Our assumption for our framework is that IoT testbeds follow a three tier architecture, which is the case for more advanced IoT testbeds of larger scale [2]. In such architectures, IoT nodes attach to GW tier devices which in turn attach to server tier devices. The *testbed resource observer* function is realized by a distributed process framework that is mainly deployed across nodes of the server and gateway tier. *Testbed resource observer* instances of the GW tier not only observe information about their own node, but also take care of observing directly attached IoT nodes. In the case of a two tier architecture with no GW tier, the testbed resource observer instances on the server tier take care of directly attached IoT nodes. Our design tries to minimize the reliance on IoT node tier as much as possible, due to the resource constraint nature of these devices to support a dedicated observation plane stack in parallel to experimentation code.

Availability information for testbed resources is kept as soft-state and requires periodic update from *Testbed resource observer* instances. Respective instances report the resource identifier of the corresponding testbed resources they are responsible for using registration messages, which are matched against the semantic resource descriptions in the data base. Only previously configured resources are considered. Resource identifiers are in the form of URNs composed of a testbed identifier prefix (preconfigured by an administrator at the testbed resource observer instance) and a unique resource identifier that can be discovered local at a resource. The latter can be the MAC address of the interface through which a node attaches to the testbed or a serial id for the case of some IoT devices, e.g TelosB.

While the update process for server tier and GW tier devices is straight forward, some more details for the IoT tier need to be explained. The *testbed resource observer* instances of the GW device discover directly attached IoT nodes either through implicit detection of their attachments or through an explicit registration of the IoT node instances. The first case works well for IoT nodes that are connected through a wired infrastructure. For example in our testbed all sensor nodes are attached via USB connections to the GWs, which can discover attachment and detachment of USB devices by observing the USB bus events or device maps of the underlying operating system. In the case only wireless links exist to IoT resources, each IoT resource must be configured with a bootstrap image that is able to communicate availability information and minimum device properties (such as the resource id) to the *testbed resource observer* instance of a gateway device. Apart from updating availability information or changes to properties that can be locally detected, e.g. energy levels, *testbed resource observer* instances also update discovered topology related information between IoT nodes and GW nodes.

A more complex process is the maintenance of IoT node related topology information and environmental characteristics such as interference for which the *testbed resource observer* relies on further support functionality. The update of the IoT node related topology information is coordinated across different testbed resource observer instances and requires the explicit installation of a profiling image on the IoT nodes. The outcomes of such characterization is then reflected by the resource observers in the topology database. Update of IoT node topology related information is carried out automatically at periodic intervals during idle time of the testbed resources or based when significant changes in interference are detected in the surrounding environment. In order to avoid interference with ongoing or scheduled experiments, the profiling activities are carefully coordinated using knowledge on utilized channels from the experiment specifications. The interference characterization of the environment requires interference detectors to be present at the gateway nodes. As our IoT nodes operate in the 2.4 GHz ISM band, WiFi interferer represent the major problem. Our current implementation characterizes interference caused by WiFi access points in the surroundings by frequently scanning available sources and recording the corresponding channels together with signal strength and quality indicators through a WiFi card attached to the gateway node. As a GW device and directly attached IoT nodes are physically close located in our testbed architecture, we make the simplifying assumption that the measured interference at the GW nodes caused by the presence of WiFi signals also affects the IoT nodes in the same manner. The *testbed resource observer* updates the corresponding information in the topology database. Interference characterization takes also place at the same time link characterization experiments are performed. As interference characterization does not rely on code executing on IoT nodes, it can take place more frequently, even during user generated experiments, and annotated to the corresponding experimentation data.

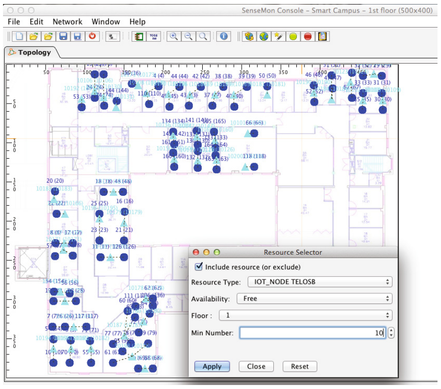
Finally the testbed resource observer instances communicate with framework services and between each other through a publish subscribe messaging bus. Our current implementation is based on an MQTT broker and client implementations. However details of the underlying communication framework is out of scope of this paper.

4 Evaluation Use Case

In the following we present a WSN protocol evaluation as an initial case study to demonstrate the usefulness of the resource selection framework in our IoT testbed. The evaluation is carried out in the SmartCCSR IoT testbed deployment, which is part of the SmartSantander experimental facility. The testbed consists of 250 freely programmable sensor nodes deployed in a real world office environment across the two floors of the CCSR building, at the University of Surrey, covering all the desks and communal areas of the research centre. The deployed IoT nodes consist of 200 TelosB based platforms and about 50 SunSpots, which are heterogeneous in their sensing capabilities. While the SunSpot platforms provide only on-board sensing capabilities (accelerometer, temperature, light), the TelosB platforms provide various sensing modalities varying from on-board sensors such as temperature, light and humidity to external sensors mounted on a custom board providing energy metering, noise and light levels, temperature and motion.

The WSN protocol utilized in this case study is a TinyOS implementation of a geographic routing protocol for wireless sensor networks [15]. The goal is to carry out an initial small scale evaluation of protocol behavior on a dense multi-hop connected network with approximately 15-20 nodes. Figure 6) shows an overview of all experimental resources on the first floor of the building.

A first challenge the experimenter is faced with is to find suitable testbed resources out of 250 available experimentation nodes inside of the building that match the expected node platform and topology requirements. While sensing modalities are not important for the experiment, the experimenter knows that all nodes with energy meters attached to it are currently utilized for a longer term study [16]. Furthermore the experiments must be run on TelosB motes as Sunspots do not support execution of his TinyOS based experimentation code. Through *TMON*, the user specifies desired resource properties in a visual query, indicating TelosB as node platform type and to use nodes with no energy meters as sensing devices. Figure 7 shows the result of the query, which identifies a subset of these nodes matching the requirements. Only 14 nodes are available, however a visual inspection with the topology explorer allows the user to verify that a suitable connected multi-hop topology exists, which will be otherwise a tedious process.



Include resource (or exclude): ✓

Resource Type: IOT_NODE TELOSB

Availability: Free

Floor : 1

Min Number: 10

Fig. 6. TMON Topology Explorer, 1st floor testbed deployment

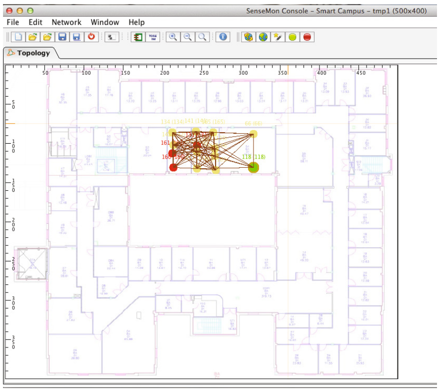


Fig. 7. TMON Topology Explorer, resource query result

Having identified a potentially suitable resource set, the experimenter now has to chose a suitable channel and power level for its planned experimentation. In order to have a fully connected multi-hop topology with the existing geographic distribution, the user requires to fine tune of the transmission power the selected set of nodes carefully. Using topology explorer, he can observer a fully connected topology at a transmission power level of 7. He further discovers potential dead-end nodes that are good experimentation cases for the behavior evaluation of the protocol.

However the protocol performance and behavior greatly depends on the behavior of the links between the experimentation nodes. Lack of detailed knowledge about the link behavior and environmental interference can lead to misinterpretation of the protocol behavior, which we will try to demonstrate in the following experimentation scenarios.

Table 1 summarizes 5 scenarios that show different channel that are affected by different levels of interference coming from WiFi access points in the surroundings. While the Scenarios 1 to 4 are derived using the knowledge gained by the system during the link characterization phase, the Scenario 5 is artificially created by modifying the testbed environment at experimentation time.

Table 1. Experiment scenarios

Scenario	802.15.4 Ch.	Interfering 802.11 Ch.	Affected nodes	Degree of Interference
1	11	1	14/14	High
2	15	2	11/14	Medium
3	16	6	5/14	Low
4	20	None	0/14	None
5	20	7	2/14	Very Low

We ran our geographic routing protocol 5 consecutive times, each over a period of 15 mins with an average generation packet rate of 1 packet every 2 seconds. In each of the considered scenarios, we compute the fraction of packet successfully delivered to the sink (PDR) and their end-to-end latency. During each run the *Testbed observer* characterizes the interference environment and we use this data for annotating each set of results. No deviations were observed to the interference previously captured in our resource models, apart for the Scenario 5 for with we intentionally turned on an new Access Point in the nodes proximity that operates on a new interfering channel (7). The results of the evaluation are reported in Table 2.

Table 2. Experiment results

Scenario	PDR	End-to-end Latency (s)
1	0	0
2	0.75	1.42
3	0.98	2.42
4	1.0	2.96
5	1.0	2.84

As expected, in an high interference scenario, such as 1, no packet are received. This is due to a characteristics of our protocol, that performs a Clear Channel Assessment of the channel based on energy level before each transmission. Due to the high interference strength affecting all the nodes in this scenario, all of them are prevent to transmit any packets. In order to exclude possible bug in the protocol, it has been tested with the same channel on a different set of nodes, not affected by this interference. The behavior of the other scenarios is quite predictable based on the collected interference information. The medium interference scenario performs better then the high interference one, with 75% of the packet delivered, but performs worse to the scenario 3 and 4 were less interference is experienced. This is due to the fact that interference prevents some node to deliver their packets, as they appear to be disconnected from the topology. As expected an higher PDR translates in an higher traffic for the network and in an higher latency experienced in delivering packets. Finally, as expected, Scenario 5 behaves very similar to Scenario 4, due to the very low interference the nodes are exposed.

Even for the small scenario that we presented for reason of simplicity, the experimenter would have had difficulties to explain the protocol behavior for randomly selected channels without such explicit knowledge of interference provided by our framework. He could have confused protocol behavior for an implementation bug or spent significant amount of time playing with channel setting in order to find a suitable topology characteristics. As the scale of experiments and testbeds grows, the benefits of our framework become even more evident.

5 Conclusions

Our initial evaluation shows that the proposed resource selection framework provides a useful tool when experimenting in unpredictable real world environments of larger scale with heterogeneous testbed resources. It allows experimenters to become more productive by providing an increased understanding of available testbed resources, their expected behavior and topological relationships in the experimentation environment. In the coming months, the proposed framework for resource selection will be integrated into the Santander testbed site, which represents a urban scale outdoor experimentation environment. We believe that such tools are essential for more productive experimentation in such challenging IoT testbed environments. While the initial focus of our work has been on catering towards the challenges of IoT environments, our framework could be applicable to testbeds of other Future Internet technologies. We have already started work on integrating mobile phone based experimentation devices into our framework and hope to investigate soon the required adaptations and the effectiveness of our framework in federated Future Internet testbeds.

Acknowledgment. This work has been partially supported by the European Union under contract numbers ICT- 2009-257992 (SmartSantander) and ICT-2009-257521 (IOT-A).

References

1. Smartsatander Project, `http://www.smartsantander.eu/`
2. Gluhak, A., Krco, S., Nati, M., Pfisterer, D., Mitton, N., Razafindralambo, T.: A survey on facilities for experimental internet of things research. IEEE Communications Magazine 49(11), 58–67 (2011)

3. Werner-Allen, G., Swieskowski, P., Welsh, M.: Motelab: a wireless sensor network testbed. In: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, IPSN 2005. IEEE Press, Piscataway (2005)
4. Sakamuri, D., Zhang, H.: Elements of sensornet testbed design. In: Yang Xiao, H.C., Li, F.H. (eds.) Handbook of Sensor Networks, ch. 35, pp. 1–36. World Scientific Publishing Co. (2009)
5. Handziski, V., Köpke, A., Willig, A., Wolisz, A.: Twist: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In: Proceedings of the 2nd International Workshop on Multi-hop Ad Hoc Networks: From Theory to Reality, REALMAN 2006, pp. 63–70. ACM, New York (2006)
6. Coulson, G., Porter, B., Chatzigiannakis, I., Koninis, C., Fischer, S., Pfisterer, D., Bimschas, D., Braun, T., Hurni, P., Anwander, M., Wagenknecht, G., Fekete, S., Kroeller, A., Baumgartner, T.: Flexible Experimentation in Wireless Sensor Networks. Communications of the ACM 55(1), 82–90 (2012)
7. Ju, X., Zhang, H., Zeng, W., Sridharan, M., Li, J., Arora, A., Ramnath, R., Xin, Y.: LENS: Resource Specification for Wireless Sensor Network Experimentation Infrastructures. In: Proceedings of the 6th International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WinTECH), Las Vegas, Nevada (September 2011)
8. Sridharan, M., Zeng, W., Leal, W., Ju, X., Ramnath, R., Zhang, H., Arora, A.: Kanseigenie: Software infrastructure for resource management and programmability of wireless sensor network fabrics. In: K. M. S., et al. (eds.) Next Generation Internet Architectures and Protocols. Springer, New York (2010)
9. ProtoGENI RSpec, `http://www.protogeni.net/trac/protogeni/wiki/RSpec`
10. OEDL - The OMF Experiment Description Language, `http://mytestbed.net/projects/omf/wiki/The_Experiment_Controller_API`
11. Network Description Language, `http://www.science.uva.nl/research/sne/ndl`
12. Botts, M., Robin, A.: OpenGIS Sensor Model Language (SensorML) Implementation Specification
13. Semantic Sensor Network, `http://www.w3.org/2005/Incubator/ssn/charter`
14. Russomanno, D.J., Kothari, C.R., Thomas, O.A.: Building a Sensor Ontology: A Practical Approach Leveraging ISO and OGC. In: The 2005 International Conference on Artificial Intelligence, Las Vegas, NV (2005)
15. Casari, P., Nati, M., Petrioli, C., Zorzi, M.: Efficient Non Planar Routing around Dead Ends in Sparse Topologies using Random Forwarding. In: Proceedings of IEEE International Conference on Communications, ICC 2007, Scotland, UK (June 2007)
16. REDUCE project, `http://info.ee.surrey.ac.uk/CCSR/REDUCE/`
17. Oppenheimer, D., Albrecht, J., Patterson, D., Vahdat, A.: Design and implementation tradeoffs for wide-area resource discovery. ACM Transactions on Internet Technology 8(2) (May 2008)