LENS: Resource Specification for Wireless Sensor Network Experimentation Infrastructures

Xi Ju*, Hongwei Zhang*, Wenjie Zeng†,
Mukundan Sridharan†, Jing Li†, Anish Arora†, Rajiv Ramnath†, Yufeng Xin‡
*Wayne State University, {xiju,hongwei}@wayne.edu
†The Ohio State University, {zengw,sridhara, ljing, anish, ramnath}@cse.ohio-state.edu
‡RENCI, yxin@renci.org

ABSTRACT

As a first step towards predictable, repeatable WSN experimentation, we propose the resource specification language LENS (a.k.a. Language for Embedded Networked Sensing) for WSN experimentation infrastructures. Using the Resource Description Framework (RDF) and the Web Ontology Language (OWL), LENS defines a semantic ontology for WSN resources; LENS enables explicit control and measurement of uncertainty factors, and it enables reasoning about the relationships between WSN resources. Focusing on basic concepts of WSNs, LENS supports resource specification in a wide range of WSN experimentation infrastructures, and it is extensible to support potentially unforeseen technologies. LENS is also compatible with specification languages for other network resources such as optical networks. As a part of the NSF GENI initiative, we have implemented LENS in the KanseiGenie control framework, and LENS has been actively used to support experimentation in the federated WSN infrastructure involving Kansei and NetEye. Enabling reasoning about uncertainty factors in experimentation, LENS is expected to serve as a basis for developing methodologies and tools for predictable, repeatable WSN experimentation.

Categories and Subject Descriptors

C.0 [**Computer Systems Organization**]: Systems specification methodology

General Terms

Experimentation, Measurement, Languages, Reliability

Keywords

WSN experimentation infrastructure; resource specification; testbed federation; GENI

1. INTRODUCTION

Society has witnessed a dramatic growth in its use of sensing technologies over the last decade or two. The emergent ubiquity of sensing has been fueled by the ability to network "edge" sensors, often in a wireless manner. In particular, a number of cyber-physical monitoring and control applications have been based on Wireless Sensor Networks (WSNs) at the edge that interoperate with enterprise networks and other sensor networks. WSNs are providing not only persistent, fine-grain information, they are essentially becoming programmable "fabrics" — virtualized, service-oriented, commodity platforms that support application evolution and co-existence with other networks that span multiple administrative domains and even national boundaries. Next generation networks embodying such interoperation and programmability include OGC Sensor Web [16], MSN/Virtual Earth extensions for sensor data [2], NEON ecological observatory [14], and Next Generation 9-1-1 [7].

Experimentation infrastructure for WSNs has progressed hand-in-hand with their growth. Early infrastructure consisted of testbeds that enabled fidelity in emulating field deployments, which was lacking in analysis and simulations. Testbeds have been great time savers: they inverted the equation between deployment time and data collection time: today, the former takes minutes while the latter can last from days to years. Testbeds have facilitated testing over multiple sensor data sets, environments, signal processing algorithms, and networking protocols. With the emergence of people-centric participatory sensing, they have begun to support long-lived concurrent applications. Importantly, from a science perspective, they have led to discovery of phenomena in low-power radios and validation of models, thereby improving the effectiveness of analysis and simulation tools [32, 39, 27].

It is perhaps surprising then that the gap between theory and practice is still large. Even though models for simulation and analysis have been refined, substantial divergences remain between experimental results and analysis- or simulation-based predictions. Also, the literature is replete with experimental results in different settings of interest which draw opposing claims about the effectiveness of primitives, metrics, or protocols. As quick examples, the data collection protocol MintRoute has been shown to perform well in [36] but poorly in [24], and scheduling based on the protocol interference model has been shown to perform worse than on the physical interference model in [28] but the former performs better than the latter in [22]. We attribute these prob-

lems of unpredictability and unrepeatability of experiment behavior to inadequacies in the experimentation infrastructure: *Uncertainty factors (such as environmental fading, interference, component variability, antenna orientation, localization error, etc.) are mostly left unspecified, unmeasured, and implicit today, even though they have significant impact on correctness and performance*. Indeed, in the cited examples, it was later found that these seemingly conflicting results were due to experimental factors (e.g., network traffic pattern and link reliability) which were neither controlled in experimentation nor considered in data analysis [23, 37].

Our hypothesis, therefore, is that the apparent disagreements between extant experimental results and the results of other experiments, analyses or simulations could be resolved if the relevant uncertainty factors and errors were known (or estimated) and explicitly considered in comparing the results. For instance, the mean or variance of a performance metric of, say, the links chosen by a routing protocol depends upon the path loss coefficient, the level of interference, the level of correlation between links, etc. If any of these factors are significantly different in two experimental settings, then comparing the results while ignoring the factors would be flawed. Conversely, to obtain experimental results that conform to analytical predictions, we will need to augment the state-of-the-art in robustly estimating these factors as well as in analysis techniques for calculating performance in terms of these factors.

As a first step towards addressing uncertainty factors for predictable, repeatable WSN experimentation, we focus on WSN resource specification in this paper, and we make the following contributions:

- We identify the design principles for WSN resource specification languages, and, accordingly, we propose LENS (a.k.a. Language for Embedded Networked Sensing) as the language for specifying resources in WSN experimentation. Using the Resource Description Framework (RDF) [11] and the Web Ontology Language (OWL) [18], LENS defines a semantic ontology for WSN resources; LENS enables explicit control and measurement of uncertainty factors, and it enables reasoning about the relationships between WSN resources (e.g., node distribution and link correlation). Focusing on basic concepts of WSNs, LENS supports resource specification in a wide range of WSN experimentation infrastructures, and it is extensible to support potentially unforeseen technologies. LENS is also compatible with specification languages for other network resources such as optical networks, so that LENS readily supports experimentation that involves heterogeneous network resources. To the best of our knowledge, LENS is the first ontology language for resource specification in WSN experimentation and wireless experimentation in general.

- We implement LENS in the KanseiGenie control framework [33]. Serving as a basis for whole-life-cycle resource management in federated WSN experimentation infrastructures, LENS enables experimentation with multiple, heterogeneous WSNs and thus facilitates sensitivity and regression analyses of protocols.

- We evaluate LENS by examining its expressiveness in describing WSN resource requests and experimentation infrastructures, and we examine LENS' capability in supporting experimentation (e.g., quantifying network diameter) with controllable and observable factors. LENS has also been actively used for experimentation in the federated WSN infrastructure involving Kansei [25] and NetEye [3].

The rest of the paper is organized as follows. We discuss related work in Section 2. Then we present the design of LENS in Section 3, and we discuss the integration of LENS with KanseiGenie control framework in Section 4. To demonstrate the expressiveness and utility of LENS in experimentation, we present example use cases of LENS in Section 5. We make concluding remarks in Section 6.

2. RELATED WORK

Most closely related to LENS are the Network Description Language (NDL) [4], the extension of NDL using Web-Ontology-Language (NDL-OWL) [5], and the Network Markup Language (NML) [6]. NDL is likely the first language that leverages ontology to describe network resources. NDL-OWL is an extension of NDL; it reuses the key concepts of NDL, but it also introduces new concepts such as those for domain-specific services, experimentation requests, and compute resources. NML is an ongoing initiative that aims to design an extensible schema for describing hybrid networks, where inter-domain network graphs are created and further abstracted at various layers so as to facilitate end-to-end path provisioning. While pioneering the application of ontology to network resource specification, NDL, NDL-OWL, and NML have all focused on wired networks (and optical networks in particular); none of them have examined the challenges (e.g., impact of uncertainty factors) of specifying WSN resources. Our work on LENS fills the gap in ontology-based resource specification in WSN experimentation infrastructures; in fact, we are currently working with the NML/NDL team to explore the potential of employing LENS as the WSN resource specification language in the broad initiative of NML.

As a part of the NSF GENI initiative [15], the ProtoGENI [10] and PlanetLab [9] control frameworks propose the declarative, XML-based resource specification language RSpec. The cOntrol, Management and Measurement Framework (OMF) [30], originally developed for the ORBIT testbed [17], introduces a domain-specific language named OEDL; OEDL is based on the Ruby scripting language [12] and utilizes Ruby's meta-programming capabilities to describe experiment-specific resource requests. Neither RSpec nor OEDL is based

on RDF/OWL, thus they do not readily allow for reasoning about the relationships and constraints between resources. Unlike LENS, RSpec and OEDL do not address the challenges that uncertainty factors pose to resource specification and experiment analysis.

In WSNs, Sensor Model Language (SensorML) [19] defines models and XML schemas for describing general sensor data processing. Using web ontologies, Sensor Web Enablement (SWE) [20] and Semantic Sensor Network [13] extend web services to enable the creation of web-accessible sensor observations and service-based sensor network. OntoSensor [31] defines sensor ontologies for data/knowledge fusion in semantic web. None of these efforts aim to define ontology-based resource specification languages for WSN experimentation.

3. THE DESIGN OF LENS

3.1 Design principles

Towards a basis for resource management and experiment reasoning in federated, heterogeneous WSN experimentation infrastructures, we identify the following principles for defining languages of WSN resource specification:

Whole-life-cycle resource management. Resource specification is needed for the whole life-cycle of resource request, resource delegation/allocation (from owners to brokers/consumers), resource monitoring, and resource release. The language should enable specification of resources during the whole life-cycle of resource management which may well involve multiple actors/principals, multiple experimentation infrastructures, and multiple administrative domains.

Controllability vs. observability. Complex spatio-temporal dynamics are characteristic of WSN platforms. Depending upon platform capabilities, some of their uncertainty factors are both observable and controllable (e.g., co-channel interference may be controlled by supporting channel selection mechanisms in the experimentation infrastructure), others are observable but not controllable (e.g., slow changing large-scale path loss in static networks), and still others are neither (e.g., fast changing small-scale fast fading). We should therefore distinguish factors that are controllable from those that are not. Based on this distinction, we propose that experimenters be allowed to specify desired values of controllable factors. They may also specify desired values for factors (e.g., path loss exponent) which are only controllable provided that the federated infrastructure has a choice of values for those factors from across different infrastructures in the federation.[1] For factors that are observable only (i.e., not controllable), experimenters may request estimates for those factors from the experimentation infrastructure, so that their impact on protocol behavior can be analyzed (e.g., for the purpose of sensitivity and regression analysis).

Relationship modeling between resources. In WSNs, there are inherent relationship/dependencies between resources. Examples include the geometric relation between nodes, the channel relation (e.g., path loss) between nodes, the property correlation between links, and the dependencies between nodes, transceivers, and wireless channels/spectrum. For network-centric resource specification, the language should enable reasoning about the relationship between resources.

Use-case neutrality. The specific resources and resource requirements may vary across experiment stages, across experiments, and across experimentation infrastructures. The language should enable a unified approach to representing resources in the presence of heterogeneous use cases and heterogeneous resources.

Extensibility. WSN technologies, platforms, and applications evolve over time, and the language should be readily extensible to support emerging and even unforeseen resources and use cases.

Interoperability. WSNs are increasingly integrated with enterprise networks and have become an integral part of the network ecosystem. For experimentation with multiple network types, a WSN specification language should facilitate the integration of WSNs with other networks such as optical networks.

Based on these principles, we define LENS as an ontology language for specifying WSN resources, and we discuss LENS in more detail in the next subsection.

3.2 Design overview

We define LENS as an ontology language for specifying WSN resources in experimentation. Figure 1 shows, via

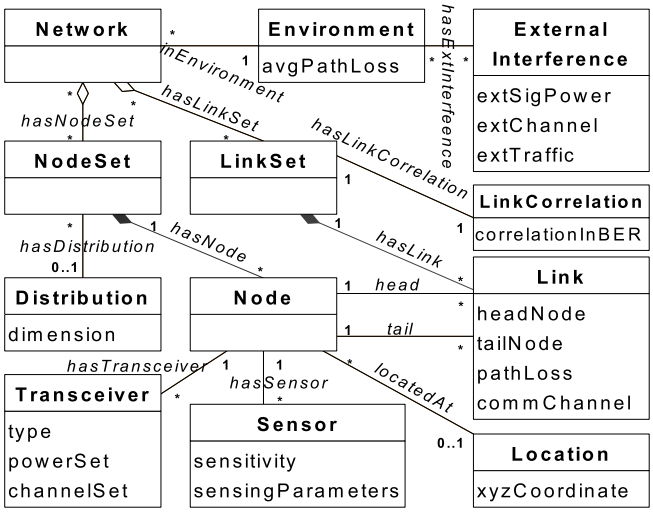

Figure 1: LENS ontology (with key concepts presented)

UML, the key concepts of LENS. (Interested readers can find the complete ontology definition of LENS in [26].) In the figure, the boxes represent the key concepts (e.g., Network, NodeSet, and LinkSet) of LENS which are defined

[1] A finer characterization would exclude fast-varying factors (e.g., small-scale fast fading, or cross-link correlation which is subject to unpredictable co-channel interference [38]) whose conformance to a specified value cannot be practically maintained by choice alone.

as Web Ontology Language (OWL) or Resource Description Framework (RDF) classes; within some boxes, we also present a few key properties of the corresponding concepts (e.g., average path loss in an environment). The line between two boxes defines the relationship between the corresponding concepts; the text on a line is a predicate; the annotation at the line-ends follows the following convention: 1) the "hollow-diamond" shape represents the "aggregation" relationship, with the hollow-diamond on the containing class of the line that connects contained classes to the containing class; 2) the "filled-diamond" shape represents the "composition" relationship, with the filled-diamond on the containing class of the line that connects contained classes to the containing class; 3) the "star" shape denotes zero, one, or more than one instances of the corresponding concept; 4) a "1" denotes a unique instance of the corresponding concept; and 5) the "0..1" denotes zero or one instance of the corresponding concept. We describe LENS using the OWL/RDF ontology language so that LENS can be readily integrated with other commonly-used ontologies such as the NDL [4] and NDL-OWL [5] for optical networks. Based on OWL/RDF instead of basic XML, LENS enables a flexible, semantic query-based approach to processing resource requests and the related resource management, and LENS enables reasoning about the relationship between resources.

The top-level concept in the LENS ontology is *Network*, which describes a physical infrastructure for embedded networked sensing (e.g., a wireless or wired sensor network). Network serves as an abstract, container-type concept that can be instantiated to denote a specific WSN experimental infrastructure or a part of an infrastructure that can be allocated as an integrated unit.

At the second level, LENS defines the concepts *Environment*, *NodeSet*, and *LinkSet* which, together, completely characterize a network. *Environment* allows for characterizing network-wide environmental conditions such as average path loss, temperature, humidity, external interference, and testbed settings (e.g., indoor vs. outdoor). The predicate "hasExtInterference" is encoded as an object property, separating the definition of *ExternalInterference* from that of Environment so that interference properties (e.g., wireless channel, signal power, and traffic load) can be treated as an independent concept. Environment and ExternalInterference allow experimenters to explicitly specify uncertainty factors in addition to specifying WSN resources, thus enabling predictable, repeatable experimentation. For uncertainty factors (e.g., testbed setting) that are readily controllable, experimentation infrastructure can control those factors when satisfying experimenters' resource requests. For uncertainty factors (e.g., average path loss and external interference) that are observable but may not be controllable, they can be measured and presented to experimenters for experiment analysis.

NodeSet and LinkSet are both collection concepts that are similar to the concept of Bag in RDF. A NodeSet consists of zero or multiple *Node*s, and a LinkSet consists of zero or multiple *Link*s. In WSN experimentation infrastructures, devices such as motes, stargates, smart phones are instances of Node, and each Node has its Transceiver, Sensor, Location, MobilityPattern, etc. In WSNs, it is important to characterize the channel properties (e.g., signal power loss) between nodes, and we define the concept of Link to discuss these properties. These link properties are useful in protocol design (e.g., for interference-oriented scheduling), and they are also useful for experimentation. An experimenter may not choose to explicitly specify Links in his/her resource request, but an experimenter may well specify resources in a way that can be facilitated by the concept of Link. For instance, an experimenter may say "Give me a Poisson random network with 100 TelosB nodes such that the average number of nodes in a one-hop neighborhood can be controlled to be between 5 and 20." For the resource broker (to be discussed in more detail in Section 4) that allocates resources, the knowledge of inter-node path loss (as facilitated by the concept of Link) can be instrumental in deciding how to satisfy this type of resource requests. In the mean time, while an experimenter may not explicitly specify links in resource request, he/she may specify measurement service as a part of his experiment request. For instance, an experimenter may say "Give me data showing the correlation between links that share the common transmitters." In this case, the concept of Link is also useful and needed in discussing link correlations. Note that link correlations can be used for protocol analysis and design [34].

As discussed in Section 3.1, there are inherent relationship/dependencies between WSN resources. In particular, spatial node distribution is a basic property that defines spatial constraints on node distribution and that has non-negligible impact on network performance. Due to the broadcast nature of most WSN communication technologies, the properties of geographically co-located links (e.g., those sharing a common head-node) tend to be correlated, and link correlation has significant impact on protocols such as opportunistic routing and broadcast. To characterize these relationship between nodes and links, LENS defines the concepts of *Distribution* and *LinkCorrelation* to describe spatial node distribution and link correlation respectively.

Besides the aforementioned concepts, LENS also defines concepts such as Transceiver, Sensor, and Location to describe properties of Node. Due to the limitation of space, we delegate detailed discussion of these concepts and the detailed properties of LENS concepts to [26]. Note that even though LENS has been defined mostly with WSNs in mind, the concepts in LENS is sufficient to specify resources in wired embedded sensing networks too. Focusing on fundamental concepts in embedded networked sensing networks and defined via OWL/RDF, LENS is also readily extensible to support new networking technologies and new use-cases, for instance, by defining new OWL/RDF classes and/or properties.

With the ontology language LENS, WSN resources can be

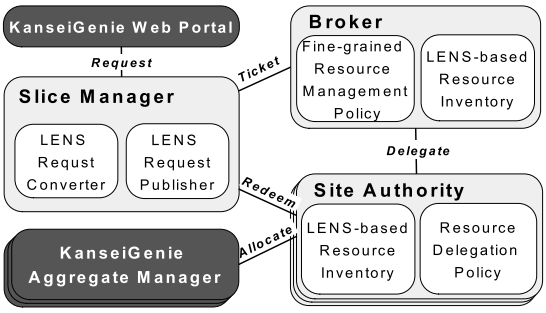

Figure 2: LENS integrated with KanseiGenie/ORCA

described in a structured, concise manner, and the WSN resource specifications enable experiment control frameworks to manage network resources in heterogeneous, federated WSN experimentation infrastructures. We discuss the application of LENS to resource management in the KanseiGenie control framework in the next section.

4. INTEGRATION OF LENS WITH KANSEIGENIE CONTROL FRAMEWORK

As a part of the NSF GENI [15] software system, KanseiGenie [33] is a software suite for resource management and experiment control in federated WSN infrastructures. As shown in Figure 2, three key components of KanseiGenie are its web portal, aggregate manager, and federated resource management core, where the federated resource management core consists of three sub-components of site authority, broker, and slice manager. The web portal is the web-based front-end via which experimenters interact with the federated WSN infrastructures, for purposes such as requesting resources, scheduling experiments, and retrieving experiment data. For each WSN infrastructure of the KanseiGenie WSN federation, the corresponding aggregate manager takes charge of experiment control (e.g., start/end, instrumentation) and resource management for the infrastructure.

The federated resource management core is based on the Open Resource Control Architecture (ORCA) [8], and it coordinates with the web portal and the aggregate managers to manage resource delegation, allocation, request, and redemption across the federated WSN infrastructures. In particular, each WSN infrastructure has a site authority that serves as the interface between the infrastructure's aggregate manager and the resource management core; the broker is the central part of the resource management core, and it manages federation-wide resource allocation; the slice manager is the interface between the web portal and the resource management core, and it creates, configures, and manages one or more slices of resources. In what follows, we discuss how LENS is integrated with the KanseiGenie resource management core for the whole-life-cycle, federated resource management. (Note that the broker can be distributed as the federation scales up, but, for ease of presentation, we focus on a centralized broker in the discussion below).

```
<lens:Node rdf:ID="KanseiMote118">
   <lens:hasTransceiver
        rdf:resource="#RF_CC2420"/>
   <lens:hasLocation
        rdf:resource="#Coordinates118"/>
   <lens:hasNodeId
        rdf:datatype="&xsd;string">
        118</lens:hasNodeId>
   <lens:hasResourcePoolType
        rdf:datatype="&xsd;int">
        3</lens:hasResourcePoolType>
   <lens:hasProductType
        rdf:datatype="&xsd;string">
        TelosB</lens:hasProductType>
   <lens:hasSensor
        rdf:resource="#LightSensorT1"/>
   <lens:hasSoftware xml:lang="en">
        TinyOS</lens:hasSoftware>
   <lens:hasCPU xml:lang="en"
        MSP430</lens:hasCpu>
    ... ...
</lens:Node>
```

Figure 3: Example of LENS Node Instance

Site authority: resource delegation. One major role of a site authority is to delegate resources of the corresponding WSN infrastructure to the broker, so that the broker can perform federation-wide resource management (e.g., optimal resource allocation). Depending on the resource sharing policy of the infrastructure's owner, the resource-delegation-policy plug-in of the site authority determines the part of the infrastructure resources to be delegated to the broker for federation-wide management. As a part of the delegation process, the delegated resources are described in LENS and communicated to the broker via LENS files. As an example, Figure 3 shows the LENS specification of a TelosB node, which includes its transceiver, location, ID, sensor, etc. Before delegating its resources to the broker, the site authority makes sure that the LENS files describing the resources conform with the LENS ontology rules to ensure consistency of the resource description.

Broker: resource allocation. The broker receives resource delegations from the site authorities of the federation; from the LENS files describing the delegated resources, the broker builds up the resource inventory which serves as a basis for the federation-wide resource management. In particular, the resource inventory keeps an up-to-date record of federation-wide resources that are sharable subject to certain policy constraints. After receiving a resource request from an experimenter via the slice manager (to be discussed shortly), the broker queries its resource inventory and tries to satisfy the request by allocating resources to the experimenter according to certain resource allocation algorithms and policies (e.g., access control policy). Querying the resource inventory can be assisted by the SPARQL (SPARQL Protocol and RDF Query Language) [29] package of the Jena semantic web toolkit [21]. For instance, Figure 4 shows

```
SELECT ?network ?envpathloss     ?extchannel
       ?setting ?node ?nodetype ?dist
FROM   <KanseiLensSub.owl>
WHERE  {
   ?network lens:inEnvironment  ?env
   ?env     factors:avgPathLoss ?envpathloss
   ?env     factors:hasExtInterference ?extint
   ?extint  factors:hasChannel  ?extchannel
   ?network lens:hasNodeSet     ?nodeset
   ?nodeset lens:hasNode        ?node
   ?node    topology:hasProductType  ?nodetype
   FILTER   (?envpathloss > 0)
   OPTIONAL {
       ?env     factors:envSettings  ?setting
       ?nodeset lens:hasDistribution ?dist
   }
}
Results = [
   'network'     = 'KanseiGenieSubNet',
   'envpathloss'= '4.7',
   'extchannel' = '6',
   'setting'     = 'indoor',
   'node'        = 'kanseimote118',
   'node'        = 'kanseimote119', ...,
   'nodetype'    = 'TelosB'
   'dist'        = 'griddistributionT'
]
```

Figure 4: Example SPARQL Query and Results

an example SPARQL query and the results for finding a network satisfying certain requirements on wireless path loss, node distribution, external interference, etc.

Slice manager: resource request. The slice manager is the interface between the web portal and the resource management core, and an experimenter requests resources through the slice manager. In particular, an experimenter may submit the resource request by filling out the experiment specification (which includes a primitive resource specification) on the KanseiGenie web portal, or by issuing a XML-RPC call to the slice manger with a LENS-described resource specification; for the primitive resource specification via the web portal, the slide manager uses a LENS request converter to translate the primitive request into a LENS-based request. Before submitting the resource request, the LENS-request-publisher in the slice manager validates the LENS-described resource specification to make sure that it is correctly formulated according to the LENS ontology rules; the validation is done using related Jena APIs [21]. After submitting the resource request to the broker, the slice manager may receive a ticket from the broker which specifies, via LENS, the resources allocated by the broker. Using the ticket, the slice manager can redeem resources from one or more WSN infrastructure through their site authorities.

5. EXAMPLE USE CASES OF LENS

Through its integration with the KanseiGenie control framework, LENS has been actively used for experiments in the federated Kansei [25] and NetEye [3] WSN testbeds. Our experience has shown that LENS is expressive enough to support a wide variety of resources and resource management mechanisms, we have also observed that the explicit modeling of uncertainty factors in LENS helps experiment design and analysis. In what follows, we demonstrate by examples how LENS supports resource specification in both KanseiGenie and MoteLab, which represents different WSN testbed implementations; we also demonstrate how LENS supports better experiment design and analysis by enabling explicit control and measurement of uncertainty factors.

```
request='sliceID=34,' +
        'description=demo,'     +
        'startTime=\now,'       +
        'endTime=\end,'         +
        'arrayType=TELOSB,'     +
        'exeFile=147,'          +
        'addressOffset=0'
```

Figure 5: String-based experiment specification in KanseiGenie

5.1 LENS-based resource specification in KanseiGenie and MoteLab

Focusing on fundamental concepts in WSNs, LENS enables resource specification in heterogeneous WSNs in a way well beyond its current deployment in the KanseiGenie testbeds. For demonstration, we discuss how LENS allows for specifying resource requests that are possible in the original KanseiGenie implementation and in MoteLab [35].

LENS in KanseiGenie. To access resources of the KanseiGenie testbeds, an experimenter can either go to the KanseiGenie web portal to fill out the resource-request-form manually, or the experimenter can remotely invoke the web service and include LENS-specified resource request in the web service call. The latter approach is readily supported by KanseiGenie since the KanseiGenie slice manager understands/-supports LENS-based resource requests. In the former approach (which is also the original approach used by KanseiGenie before the introduction of LENS), a resource request is usually expressed as a raw string or as XML data. To support this type of resource requests, we have found that a simple converter is enough to parse the request and convert it to LENS-based specification. For instance, Figure 5 shows a simple string-based experiment specification issued by the KanseiGenie web portal. The experiment specification includes specification of the resources requested. In particular, the sliceID points to an experiment slice of testbed devices; to convert the specification to be LENS-based, the converter only needs to create instances of the Node class defined by LENS accordingly. Moreover, Node properties (e.g., type and logical ID) can be obtained by checking the assigned values to arrayType and addressOffest in the string-based specification. We have implemented the converter as a part of the KanseiGenie slice manager.

LENS in MoteLab. MoteLab [35] is a WSN testbed deployed at three floors of the EECS building at Harvard Uni-

```
<lens:Network rdf:ID="MoteLabNetwork">
  <lens:hasNodeSet rdf:resource="#NS_m"/>
  <lens:hasLinkSet rdf:resource="#LS_m"/>
  <lens:inEnvironment rdf:resource="#Env"/>
</lens:Network>
<owl:Class rdf:ID="MoteLabReservation"/>
<MoteLabReservation rdf:ID="Reserv_m">
  <rsvDuration rdf:datatype="&xsd;int">
      30</rsvDuration>
  <rsvMigClassFile xml:lang="en">
      SerialMsg.class</rsvMigClassFile>
  <rsvTargetNetwork
      rdf:resource="#MoteLabNetwork"/>
  <rsvSite rdf:datatype="&xsd;string">
      MoteLab</rsvSite>
  <rsvStartingTime
      rdf:datatype="&xsd;dateTime">
          2011-05-22T17:08:00
  </rsvStartingTime>
  <rsvTosExecutable xml:lang="en">
      CTPtrial.exe</rsvTosExecutable>
  <rsvUnits rdf:datatype="&xsd;int">
      150</rsvUnits>
</MoteLabReservation>
```

Figure 6: LENS-based resource specification for MoteLab

versity. In MoteLab, an experimenter specifies his experiment request by filling in several PHP forms at the MoteLab web portal, and a part of the experiment specification identifies the resources (e.g., the set of nodes) to be used for the experiment. Figure 6 shows an example experiment specification which includes a LENS-based specification for MoteLab resources needed for the experiment. In the specification, a MoteLab reservation is created where a NodeSet and a LinkSet in an Environment are generated for the requested MoteLabNetwork. The set of Nodes in the NodeSet and the set of Links in the LinkSet can be further specified via LENS, but we skip the details here due to the limitation of space. This example shows that LENS is expressive enough to support resource requests that can be defined in MoteLab. MoteLab is similar to several other WSN testbeds such as Indriya [1] in terms of software systems, deployment, etc., thus we expect LENS to be applicable to other testbeds besides MoteLab and KanseiGenie.

5.2 Network diameter measurement across Kansei and NetEye

The performance of many network protocols (e.g., the convergence time of consensus and routing protocols) depends on network diameter, which is the largest hop-count between any pair of nodes. For this evaluation, we regard two nodes as connected if and only if the bidirectional packet reception ratio between them is no less than a threshold of 95%, and, for ease of demonstration, we consider networks consisting of straight lines of nodes in the NetEye and Kansei WSN experimentation infrastructures. In what follows, we illustrate how using LENS to explicitly specify the measurement of observable factors and the control of controllable factors helps experiment analysis and design.

Suppose an experimenter wants to understand the network diameter of a straight line of nine TelosB nodes where the distance between every two consecutive nodes is two feet. So he runs the measurement in NetEye and Kansei and finds the results to be as shown in Figure 7. He observes that, even

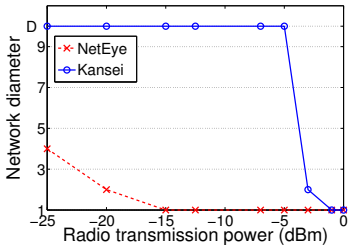

Figure 7: Network diameter in NetEye and Kansei

though the two line networks in NetEye and Kansei share the same node platform (i.e., TelosB), transmission power, and node layout, the diameters are quite different, with the diameter being much smaller in NetEye. It turns out the major cause for the difference is the fact that the wireless path loss differs in NetEye and Kansei. For instance, we compute the average path-loss-exponent (PLE) based on the following large-scale path-loss model for path loss from a node i to another node j:

$$PL_{ij} = PL_0 + 10 \cdot \alpha_{ij} \cdot \log_{10} \frac{d_{ij}}{d_0} \ ,$$

where PL_{ij} is the signal power loss from i to j, α_{ij} is the PLE from i to j, PL_0 is the signal power loss at the reference distance d_0, and d_{ij} is the distance from i to j. We find that the average PLE in NetEye is 2.61, but the average PLE in Kansei is 3.69. Suppose the experimenter has used LENS to specify that the average PLE be measured during experiments, then the experimenter can get feedback from NetEye and Kansei on their respective PLEs, and he/she can use this information to analyze experiment results.

Now suppose the experimenter wants to control network diameter by controlling the distance between every two consecutive nodes in the line network, and he/she is willing to consider the case of a 2 feet inter-node separation and the case of a 4 feet inter-node separation. It is conceivable that the experimentation infrastructures already have instrumentation data which show the relationship between inter-node distance and network diameter. For NetEye, for instance, Figure 8 shows the network diameter for 2-feet and 4-feet inter-node separation. As expected, the network diameter increases with the inter-node separation. Based on this instrumentation data and the desired network diameter, the experimenter can use LENS to specify controlling the inter-node separation to be the desired distance, and the experimentation infrastructure (more specifically, the broker in Figure 2) can allocate nodes accordingly. This way, the experimenter can control the controllable factors to satisfy his resource requirement in experiment design.

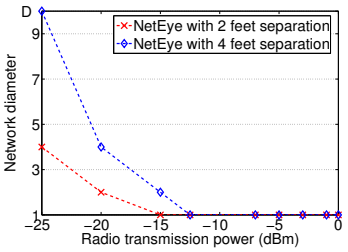

Figure 8: Network diameter in NetEye with different inter-node distance

6. CONCLUDING REMARKS

As a first step towards predictable, repeatable WSN experimentation, we have defined the ontology-based resource specification language LENS. LENS enables explicit control and measurement of uncertainty factors during experimentation, and it enables reasoning about the relationships between WSN resources. We have implemented LENS in the KanseiGenie control framework, and LENS has been successfully supporting experimentation in the federated WSN infrastructure involving Kansei and NetEye. Enabled by LENS, uncertainty factors in experimentation can be controlled and/or measured, and this capability is expected to serve as a basis for developing methodologies and tools for predictable, repeatable WSN experimentation. LENS has been defined mainly for WSN experimentation infrastructures, but LENS also supports resource specification in wired sensor networks; we also expect LENS to be able to serve as a basis for resource specification in production WSN infrastructures. We will explore these opportunities and directions in our future work.

Acknowledgment

This work is supported in part by NSF awards CNS-1054634 and GENI-1633 as well as a grant from Ford Research. We are also grateful for the constructive discussions with Ilia Baldine, Jeroen Van Der Ham, Aaron Helsinger, and Max Ott on network resource specification.

7. REFERENCES

[1] Indriya testbed. http://indriya.comp.nus.edu.sg/.
[2] Microsoft Research SensorMap project. http://atom.research.microsoft.com/sensormap/.
[3] NetEye testbed. http://neteye.cs.wayne.edu/neteye/home.php.
[4] Network description language. http://www.science.uva.nl/research/sne/ndl.
[5] Network description language plus OWL. https://geni-orca.renci.org/trac/wiki/NDL-OWL.
[6] Network markup language. http://forge.ogf.org/sf/projects/nml-wg.
[7] Next generation 9-1-1. http://www.its.dot.gov/ng911/pdf/USDOT_NG911_FINAL_System_Design.pdf.
[8] Open Resource Control Architecture (ORCA). http://nicl.cod.cs.duke.edu/orca/.
[9] PlanetLab. http://groups.geni.net/geni/wiki/PlanetLab.
[10] ProtoGENI. http://www.protogeni.net/trac/protogeni/wiki/RSpec.
[11] Resource Description Framework (RDF). http://www.w3.org/RDF/.
[12] Ruby scripting language. http://www.ruby-lang.org.
[13] Semantic senor netwok. http://www.w3.org/2005/Incubator/ssn/.
[14] NEON ecological observatory. http://www.neoninc.org/.
[15] NSF GENI program. http://www.geni.net/.
[16] OGC sensor web. http://www.opengeospatial.org/pub/www/ows6/index.html.
[17] ORBIT wireless network testbed. http://www.orbit-lab.org/.
[18] Web ontology language (owl). http://www.w3.org/2004/OWL/.
[19] M. BOTTS. Sensorml website: http://vast. uah. edu.
[20] M. Botts, G. Percivall, C. Reed, and J. Davidson. Ogc® sensor web enablement: Overview and high level architecture. *GeoSensor networks*, pages 175–190, 2008.
[21] J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson. Jena: implementing the semantic web recommendations. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 74–83. ACM, 2004.
[22] D. Chafekar, V. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. Approximation algorithms for computing capacity of wireless networks with sinr constraints. In *IEEE INFOCOM*, 2008.
[23] X. Che, X. Liu, X. Ju, and H. Zhang. Adaptive instantiation of the protocol interference model in mission-critical wireless networks. In *IEEE SECON*, 2010.
[24] Y.-R. Choi, M. Gouda, H. Zhang, and A. Arora. Stabilization of grid routing in sensor networks. *AIAA Journal of Aerospace Computing, Information, and Communication*, 3:214–233, 2006.
[25] E. Ertin, A. Arora, R. Ramnath, M. Nesterenko, V. Naik, S. Bapat, V. Kulathumani, M. Sridharan, H. Zhang, and H. Cao. Kansei: A testbed for sensing at scale. In *IEEE/ACM IPSN/SPOTS*, 2006.
[26] X. Ju, H. Zhang, W. Zeng, A. Arora, M. Sridharan, J. Li, R. Ramnath, and Y. Xin. LENS: Language for embedded networked sensing. Technical Note DNC-TR-11-03 (http://www.cs.wayne.edu/~hzhang/group/TR/DNC-TR-11-03.pdf), Wayne State University, 2011.
[27] H. Lee, A. Cerpa, and P. Levis. Improving wireless simulation through noise modeling. In *ACM/IEEE IPSN*, 2007.
[28] R. Maheshwari, S. Jain, and S. Das. A measurement study of interference modeling and scheduling in low-power wireless networks. In *ACM SenSys*, 2008.
[29] E. PrudHommeaux, A. Seaborne, et al. Sparql query language for rdf. *W3C working draft*, 4, 2006.
[30] T. Rakotoarivelo, M. Ott, G. Jourjon, and I. Seskar. Omf: a control and management framework for networking testbeds. *ACM SIGOPS Operating Systems Review*, 43(4):54–59, 2010.
[31] D. Russomanno, C. Kothari, and O. Thomas. Building a sensor ontology: A practical approach leveraging iso and ogc models. In *The 2005 International Conference on Artificial Intelligence*, pages 17–18. Citeseer, 2005.
[32] D. Son, B. Krishnamachari, and J. Heidemann. Experimental analysis of concurrent packet transmissions in low-power wireless networks. In *ACM SenSys*, 2006.
[33] M. Sridharan, W. Zeng, W. Leal, X. Ju, R. Ramnath, H. Zhang, and A. Arora. From Kansei to KanseiGenie: Architecture of federated, programmable wireless sensor fabrics. In *ICST TridentCom*, 2010.
[34] K. Srinivasan, M. Jain, J. I. Choi, T. Azim, E. S. Kim, P. Levis, and B. Krishnamachari. The κ-factor: Inferring protocol performance using inter-link reception correlation. In *ACM MobiCom*, 2010.
[35] G. Werner-Allen, P. Swieskowski, and M. Welsh. Motelab: A wireless sensor network testbed. In *IEEE/ACM IPSN/SPOTS*, 2005.
[36] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *ACM SenSys*, 2003.
[37] H. Zhang, A. Arora, and P. Sinha. Link estimation and routing in sensor network backbones: Beacon-based or data-driven? *IEEE Transactions on Mobile Computing*, May 2009.
[38] H. Zhang, L. Sang, and A. Arora. Experimental analysis of link estimation methods in low power wireless networks. Technical report, Wayne State University (http://www.cs.wayne.edu/~hzhang/group/TR/DNC-TR-08-06.pdf), 2008.
[39] M. Zuniga and B. Krishnamachari. An analysis of unreliability and asymmetry in low-power wireless links. *ACM Transactions on Sensor Networks*, 3(2), 2007.