Thingsonomy: Tackling Variety in Internet of Things Events

To assure rapid adoption of Internet of Things (IoT) applications, it's important to have scalable middleware that can abstract developers and users from the IoT infrastructure. Event processing systems could fill this gap between IoT infrastructure and application layers. However, semantic coupling poses a challenge to scalability in IoT's highly semantically heterogeneous and dynamic environment. Here, the authors propose a solution based on loosely coupled producers and consumers enabled with approximate semantic matching of events. This approach emphasizes a practitioner's perspective for building software that can tackle the heterogeneity of IoT events.

Souleiman Hasan and Edward Curry
National University of Ireland, Galway

The Internet of Things (IoT) builds on the success of Internet technologies to connect physical objects, or things, to the Internet and enable a plethora of applications such as assisted driving, augmented reality, or smart and comfortable homes.[1] A basic requirement to realize the IoT is an infrastructure of communication solutions and interoperability standards, such as the IETF's Constrained Application Protocol (CoAP).[1] The IoT also needs middleware that can abstract the application developers from the underlying technologies; this is crucial to IoT applications' adoption and evolution.[1]

Event-based technology has played an important role in the middleware space to enable scalable software architecture based on its loosely coupled interaction model. Nevertheless, event-based systems assume a high level of semantic agreement between event producers and consumers, which is challenging for largely heterogeneous environments such as smart cities, because of the difficulties establishing common semantic agreements. Current approaches use granular semantic models such as ontologies, but such models are time consuming to build and agree upon, and thus limit scalability.

To address these concerns, here we extend an event-based architecture to encompass the *semantic normalization* functionality needed in the IoT (crossing the semantic boundaries between systems). It guides practitioners to build IoT applications where exchanged events convey semantics, and at the same time frees parties from rigid agreements. The architecture includes a semantic model based on the statistical co-occurrence of terms in large textual corpora such as Wikipedia; thematic tagging of events and subscriptions; and an approximate probabilistic event matcher.

 Published by the IEEE Computer Society 1089-7801/15/$31.00

The IoT and Event-Based Systems

From a high-level architectural perspective, we can divide the IoT into three tiers[1]:

- *Sensing and communication.* These technologies form the basic infrastructure for the IoT to map the world of things into the world of computationally processable information. RFID plays a key role within this tier, where RFID tags are attached to real-world things and RFID readers are responsible for instrumenting their information into the Internet. Communication and networking standards such as the IPv6 over low-power wireless personal area networks (6LoWPAN) and the CoAP protocols[1] serve this IoT layer.
- *Middleware layer.* This tier encompasses common functionalities and abstracts application developers and users from IoT infrastructure details. Among the technologies to contribute to this layer are service-oriented architectures (SOA)[1] and message-oriented middleware (MOM). Event-processing systems are a more generic version of MOM, which supports functionalities such as early event filtering, spatiotemporal correlation, sequencing, event enrichment, event aggregation, and complex event processing.
- *Application layer.* This layer builds on the middleware to provide users with direct and potentially domain-specific benefits. The IoT promises new domains of applications in transportation and logistics, healthcare, smart environments, analytics, and personal and social media.

In the following we present a concrete scenario, to better explain the challenges of the IoT, as well as the advantages of our work.

Motivation

Bob works in the town hall planning department of a smart city. He's interested in finding the energy usage of street lights during peak electricity usage in different areas. He can find such information using an expression of an Event Processing Language (EPL), such as Esper's language (see http://esper.codehaus.org), as follows:

```
every a=StreetLightsEvents(
    a.type= 'energy consumption event'
    and a.area.consumptionPeak='true')
```

Although the required information sources are available from the street lights, the event's semantics differ from one area to another due to different sensor manufacturers. For instance, events contain terms such as *energy consumption* and *electricity usage* to refer to the same thing. The scenario requires a large set of rules with high definition and maintenance costs to cover events' semantic heterogeneity.

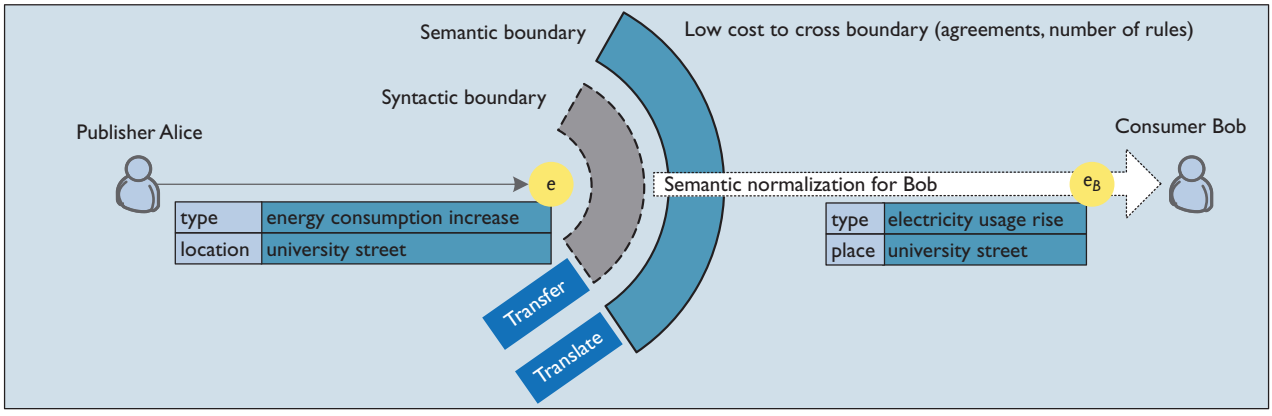
Cross-Boundary Information Exchange

In a system of systems such as the IoT, information items such as events need to cross system boundaries to enable cooperation. Paul Carlile[2] recognizes two boundary levels that might exist in a given knowledge-exchange scenario:

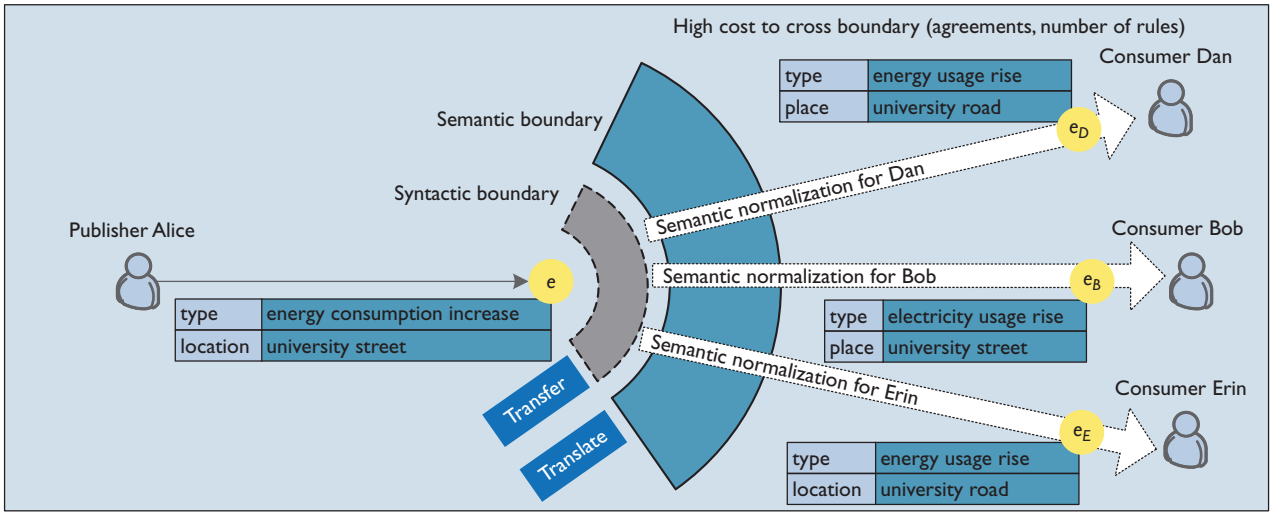
- A *syntactic boundary* affects the basic knowledge transfer mechanism between participants. In a broad sense, it's concerned with data formats, participants' interaction time, and addressing which of these exist in most event-based environments (see Figure 1).
- A *semantic boundary* starts to appear when new event sources or consumers make some meanings unclear or ambiguous. Semantic boundaries are inherent in large-scale, open, and heterogeneous environments, such as the IoT shown in Figure 1. Thus, it's crucial to have semantic normalization, translating heterogeneous information items into a common meaning model for developers.

The IoT requires an open mode of information exchange, in which system boundaries are crossed frequently. This puts openness as an inevitable requirement for IoT technologies. Event-based systems have great potential to contribute to realizing the IoT, due to their decoupled nature. Nonetheless, they don't easily cross semantic boundaries due to assumptions of semantic agreements on terms within events and subscriptions. (See the related sidebar for others' semantic normalization approaches.)

In the event-based paradigm, event sources fire instantaneous and atomic information items called events. Event consumers use rules or subscriptions to detect events and react to them. Events are the only means of interaction between sources and consumers. This results in decoupling event production and consumption, and thus increasing scalability by "removing



(a)



(b)

Figure 1. Boundaries to event exchange in (a) a small-scale known environment, and (b) a large-scale open environment such as the Internet of Things (IoT). Because systems' boundaries are crossed frequently, openness is an inevitable requirement for IoT technologies.

explicit dependencies between the interacting participants."[3]

Event-based systems decouple participants on three dimensions[3]:

- *space decoupling* suggests that participants don't need to know each other,
- *time decoupling* means that participants don't need to be active at the same time, and
- *synchronization decoupling* suggests that event producers and consumers aren't blocked while producing or consuming events.

The space, time, and synchronization decoupling dimensions that Patrick Eugster and his colleagues defined[3] contribute to event transfer across Carlile's syntactic boundaries.

However, event-based systems can simultaneously be tightly coupled by the semantics of exchanged events. Traditional deployments of event systems assume a mutual agreement on event types, attributes, and values to achieve semantic normalization, and this forms an explicit dependency between participants. For example, if a smart city event source marks an event with the type *parking space occupied*, all event consumers would have to use this exact event type in their rules. A new event consumer to the system can't use a rule with the term *garage spot occupied* to handle such events.

Current Approaches for Semantic Normalization

We compiled a list of current semantic normalization approaches in Table 1. In the *content-based approach*, event sources and consumers use the same event types, attributes, and values as assumed in traditional content-based publish/subscribe systems such as *Scalable Internet Event Notification Architectures* (SIENA).[1] The approach has high semantic coupling between parties and works well in environments with a low level of data heterogeneity.

In the *concept-based approach*, participants can use different terms and still expect event matchers to pair them properly, thanks to an explicit knowledge representation that encodes semantic relationships between terms. Examples of knowledge representations are thesauri and ontologies, as in the Semantic Toronto Publish/Subscribe System (S-TOPSS)[3] and semantic pub/sub.[4] Building such knowledge representations is a time-consuming process.

André Freitas and his colleagues proposed an approximate query-processing approach for databases based on distributional semantics.[7] In our previous work,[5,6] we proposed an approximate semantic event-processing approach and showed that the model is suitable when participants agree on some event types, attributes, or values, while performance decreases significantly with an absolute 100 percent degree of required approximation.

References

1. A. Carzaniga, D.S. Rosenblum, and A.L. Wolf, "Achieving Scalability and Expressiveness in an Internet-Scale Event Notification Service," *Proc. 19th Ann. ACM Symp. Principles of Distributed Computing*, 2000, pp. 219–227.
2. S. Hasan and E. Curry, "Thematic Event Processing," *Proc. 15th Int'l Middleware Conf.*, 2014, pp. 109–120.
3. M. Petrovic, I. Burcea, and H.-A. Jacobsen, "S-ToPSS: Semantic Toronto Publish/Subscribe System," *Proc. 29th Int'l Conf. Very Large Databases*, vol. 29, 2003, pp. 1101–1104.
4. L. Zeng and H. Lei, "A Semantic Publish/Subscribe System," *Proc. IEEE Int'l Conf. E-Commerce Technology for Dynamic E-Business*, 2004, pp. 32–39.
5. S. Hasan, S. O'Riain, and E. Curry, "Approximate Semantic Matching of Heterogeneous Events," *Proc. 6th ACM Int'l Conf. Distributed Event-Based Systems*, 2012, pp. 252–263.
6. S. Hasan and E. Curry, "Approximate Semantic Matching of Events for The Internet of Things," *ACM Trans. Internet Technology*, vol. 14, no. 1, 2014, pp. 2:1–2:23.
7. A. Freitas et al., "Querying Linked Data Using Semantic Relatedness: A Vocabulary Independent Approach," *Proc. 16th Int'l Conf. Applications of Natural Language to Information Systems*, 2011, pp. 40–51.

Table 1. Approaches to semantic normalization.[2]

Criteria	Content-based[1]	Concept-based[3,4]	Approximate semantic event processing[5,6]	Thematic event processing[2]
Matching	Exact string matching	Boolean semantic matching	Approximate semantic matching	Approximate semantic matching
Semantic coupling	Term-level full agreement	Concept-level shared agreement	Loose agreement	Loose agreement
Semantics	Not explicit	Top-down ontology-based	Statistical distributional semantics	Statistical distributional semantics
Domain specificity cost	Defining a large number of domain rules	Defining a domain-specific ontology	Indexing a domain-specific corpus	Parametrizing the vector space of an open domain corpus
Effectiveness (F_1 score)	100%	Depends on the domains and number of concept models	Depends on the corpus	Depends on the corpus and theme tags. Outperforms nonthematic approximate approach.
Cost	Defining a large number of rules and establishing shared agreement on terms	Establishing shared agreement on ontologies	Minimal agreement on a large textual corpus	Minimal agreement on a large textual corpus and associating good thematic tags
Efficiency (throughput)	High	Medium to high	Medium to high	Medium to high

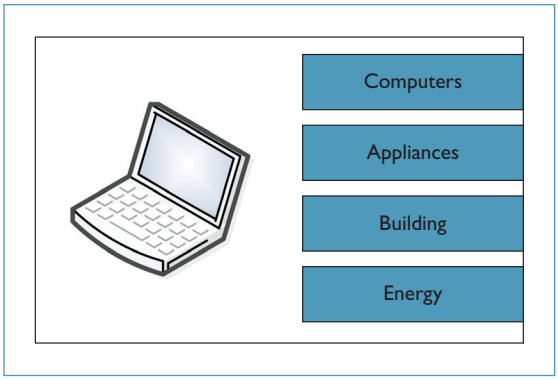


Figure 2. An example thingsonomy for tagging a device's energy-consumption events. Associating events and subscriptions with extra tags improves effectiveness and time efficiency in heterogeneous environments and domain-specific knowledge exchange.

Requirements, Thematic Event Processing, and Thingsonomies

We tackle the following requirements to address event variety in IoT middleware and application layers:

- *Low cost for integrating and accessing heterogeneous IoT devices.* A main task for integration is normalizing several heterogeneous data items into common models.
- *Effective and near real-time processing of IoT events.* Processing middleware should be able to match items of interest with a high detection rate of true positives and negatives and with low latency.

Our thematic event-processing approach builds on the analogy of the widespread use of social tagging, or folksonomies.[4] It has been observed that imposing fixed or agreed-upon top-down taxonomies on users to describe Web content (such as images) is unfeasible.[4] Instead, users tag and discover content via bottom-up and user-generated tags called *folksonomies*. Consequently, many social-tagging platforms have flourished, such as Flicker, Twitter, and Delicious.

We suggest associating thematic tags that describe the themes of types, attributes, and values and clarify their meanings. We call these tags *thingsonomies*, for things and taxonomies. The hypothesis is that associating events and subscriptions with extra tags improves effectiveness and time efficiency in heterogeneous environments and domain-specific knowledge exchange. Figure 2 shows an example thingsonomy for tagging energy-consumption events coming from a laptop.

Figure 3 illustrates the thematic event-processing approach's main components. Thematic events can cross semantic boundaries because first, they free users from a priori semantic top-down agreements and thus enable event exchange across such boundaries, and second, they carry approximations of event meanings composed of payloads and thematic tags, which when combined, carry less semantic ambiguities. An approximate matcher exploits the associated thematic tags to improve the quality of its uncertain matching. In the following, we detail more about the components and their functions in steps 1 through 6.

Step 1

The first step in designing an IoT architecture enabled with semantic normalization is to build a semantic model that enables the system to automatically establish relationships between various terms such as "computer" versus "laptop." Our approach adopts a distributional model of semantics based on statistical indexing of a large corpus of textual documents (see the "Distributional Semantics" sidebar). Such a model is easy to build automatically,[5] and the practitioner's main task is selecting the corpus. We can start with an initial documents corpus, such as Wikipedia, and incrementally revise it to suit the use cases.

Step 2

The next step is to expose a semantic-relatedness measure based on the built semantic model through a conventional interface such as REST and JavaScript Object Notation (JSON).[5] For example, a request for relatedness between "electricity" and "energy" is invoked through the API

```
http://example.com/esa?term1=energy&
term2=electricity
```

with the result being returned as a JSON object as follows:

```
{"relatedness" : 0.154}
```

Such a result makes sense only in comparison with the relatedness of other terms, such that "electricity" is closer to "energy" than to "office," for instance.

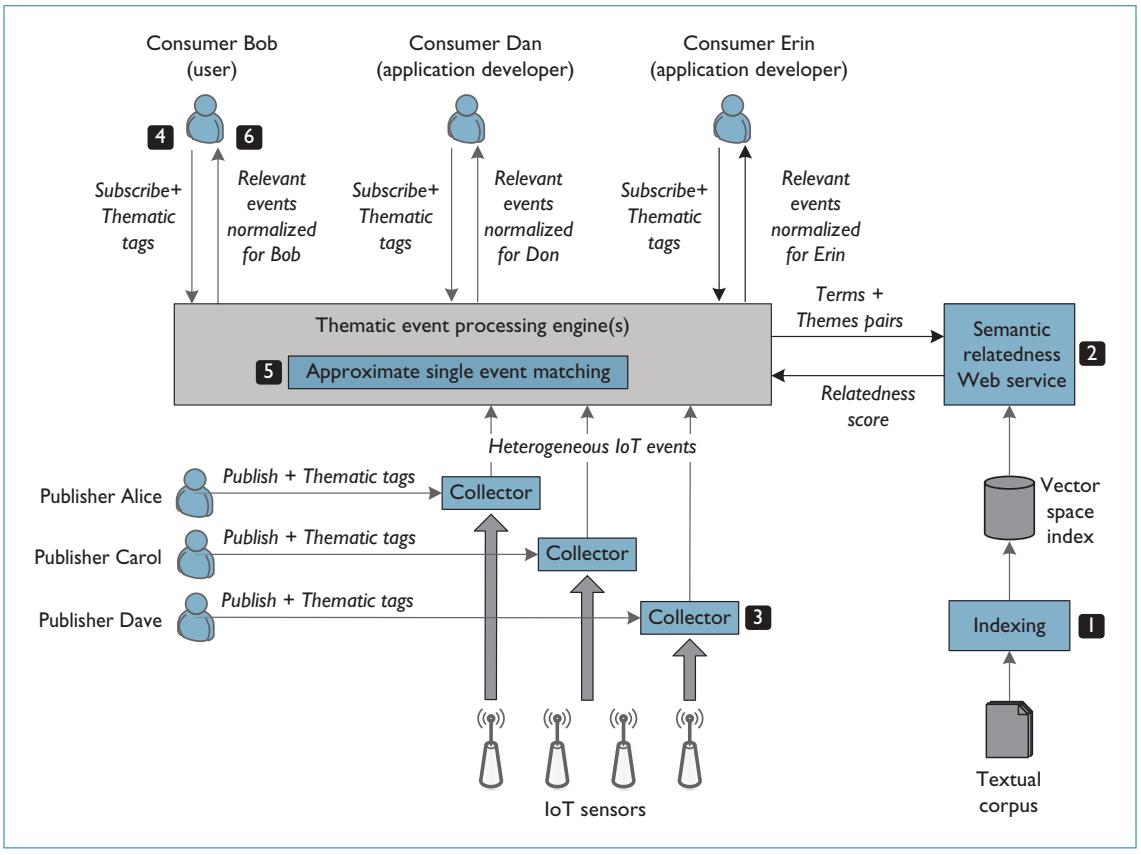


Figure 3. Architecture for loosely coupled semantic normalization for Internet of Things software. We enumerate more about the components in steps 1 through 6 in the main text.

Distributional Semantics

Distributional semantics is based on the hypothesis that similar and related words appear in similar contexts. Distributional models are quite useful for assessing semantic similarity and relatedness between terms. A *semantic measure Web service* (see Figure 3 in the main text) is a function that quantifies the similarity/relatedness between two terms and typically has its values in [0, 1]. We can construct distributional models automatically from the statistical co-occurrence of words in a corpus of documents. This model is formalized as a *vector space*, which provides a computationally efficient framework for calculating similarity scores and represents a good fit for the requirements of loose semantic coupling and real-time performance for an event-based Internet of Things.

A widely used example is the distributional Explicit Semantic Analysis semantic measure *esa*, constructed from a Wikipedia corpus.[1] In a nutshell, the Wikipedia-based *esa* builds an index of words based on the Wikipedia articles they appear in, hence the *indexing* in Figure 3.[2] A word becomes a vector of articles, and the more that articles are shared between two words, the more related the words are. For example, *esa*(“parking”, “garage”) > *esa*(“parking”, “energy”), because the former appears frequently in common articles. Typically, semantic relatedness between a pair of terms is measured using cosine distance between the two vectors representing the two terms. In our thematic model, the *esa* measure is also parameterized with the theme tags. We use those tags to project the terms’ vectors (to get more domain-specific meaning vectors), which we then pass to the distance function.

References

1. E. Gabrilovich and S. Markovitch, “Computing Semantic Relatedness Using Wikipedia-Based Explicit Semantic Analysis,” *Proc. 20th Int’l Joint Conf. Artificial Intelligence*, 2007, pp. 1606–1611.
2. D. Carvalho et al., “EasyESA: A Low-Effort Infrastructure for Explicit Semantic Analysis (Demonstration Paper in Proceedings),” *Proc. 13th Int’l Semantic Web Conf.*, 2014, pp. 177–180.

Evaluating IoT Semantic Normalization

We evaluate the normalization quality achieved by establishing a gold standard set of subscriptions and events of known ground truth for true matchings. For each subscription, we identify the set of relevant events. *Precision* represents the ratio of correctly matched events versus all matched events. *Recall* represents the ratio of correctly matched events versus all relevant ones. We measure the built software's effectiveness by precision, recall, and a derivative measure that combines both in one number, such as the F_1 score. We measure efficiency using *event throughput*, which represents the amount of processed events per time unit in the Internet of Things (IoT) middleware layer from the sensors to the applications.

We chose test events and subscription sets based on the use cases. For example, in previous work,[1] we synthesized a set of around 15,000 events of up to 10 attribute-value pairs per event, and around 100 approximate subscriptions from real-world smart city deployments in Europe (such as the SmartSantander project,[2] which employs a set of sensors to monitor temperature, noise, traffic, parking, and so on).

We expanded seed events into the final set, and generated ground truth matching and thematic tags. Figure A illustrates the resulting effectiveness and efficiency of the approximate matcher working with Wikipedia-based *esa*. Each cell in the figure shows the result that corresponds to a combination of numbers of thematic tags associated with events (the *x*-axis), and subscriptions (the *y*-axis).

Results show that the thematic approach is limited when users provide only a small number of tags for subscriptions, and when the approach requires hard real-time deadlines. Otherwise, results suggest that the use of less terms to describe events (around 2–7), and more to describe subscriptions (around 2–15), achieves a good matching quality (up to 85 percent) and throughput (up to 800 events/second), together with fewer error rates. Good performance results are concentrated in the middle left part of the squares in Figure A (more red cells).

Results also show that the approach is scalable to highly semantically heterogeneous environments, because of the lightweight amount of tagging required and the low number of approximate subscriptions, which is about 100 subscriptions. This would cost users an equivalent of around 48,000 exact subscription rules.

References

1. S. Hasan and E. Curry, "Thematic Event Processing," *Proc. 15th Int'l Middleware Conf.*, 2014, pp. 109–120.
2. L. Sanchez et al., "SmartSantander: The Meeting Point between Future Internet Research and Experimentation and the Smart Cities," *Future Network & Mobile Summit*, 2011, pp. 1–8.

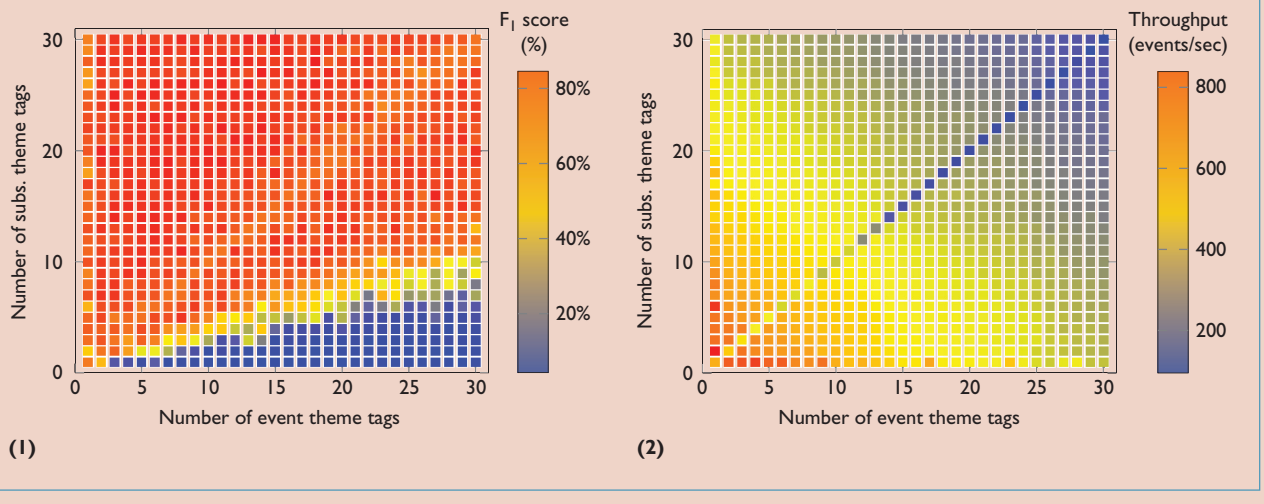


Figure A. Evaluating Internet of Things semantic normalization: (1) effectiveness and (2) time efficiency. Performance varies with tags with effectiveness as high as 85 percent and efficiency of 800 events/second.

Step 3

In the third step, publishers accompany their events with a set of thematic tags at the data collectors. Such tags represent approximately the domain and meaning of the terms used to describe the event attributes and values. Let an event of an *increased energy consumption* be represented as follows:

```
{type: increased energy consumption
       event,
measurement unit: kilowatt hour,
device: computer, office: room 112}
```

An example of thematic tags for this event is

```
{computer, appliances, building, energy}
```

Step 4

Next, subscribers associate their subscriptions with thematic tags. We use a language that introduces the tilde ~ operator, which signifies that the user wants the matcher to match the term used or any term semantically similar to it. We represent a subscription for *increased energy consumption* as follows:

```
{type= increased energy usage event~,
device~= laptop~, office= room 112}
```

Example thematic tags are

```
{power, computers}
```

Step 5

In the fifth step, the system normalizes events and matches them to suitable subscriptions. The example event and subscription don't use exactly the same terms to describe the type or device, hence "energy consumption" versus "energy usage," and "computer" versus "laptop." Nevertheless, the matcher shouldn't consider the event as a negative match to the subscription. For this reason, our model employs a probabilistic matcher that uses a measure to estimate semantic similarity and relatedness between various terms. Functionally, it tries to establish possible mappings between subscription predicates and event tuples. For example, the matcher establishes the most probable mapping of previous examples as follows:

```
σ* = {(type = increased energy
      consumption event ↔ type:
      increased energy usage event),
      (device~ = laptop~ ↔ device:
      computer),(office = room 112 ↔
      office: room 112)}
```

Step 6

This final step represents the return of events matching a subscription to its initiator. The matcher establishes probabilistic matching, and as a result, forwards the normalized event along with an uncertainty value that reflects the amount of semantic normalization conducted all the way from publishers to subscribers.

To evaluate the proposed architecture, we used a framework conceived from evaluation methodologies of information retrieval search engines. The framework is built on the concepts of matching precision, recall, and F_1 score along with throughput, as we discuss in the sidebar, "Evaluating IoT Semantic Normalization."

Design Considerations

The degree of approximation is the number of tilde ~ operators used in subscriptions. We use it to quantify the engine's approximation during semantic normalization. The proposed approach works better and needs fewer tags with lower degrees of approximations, given that exact string matching can help filter many events. For example, in some applications, we assume several agreements, such as the units of measurements in smart grids.

Besides, using semantic-relatedness services instead of exact string comparison is costly from a time-performance viewpoint. Thus, applications with hard real-time deadlines (for example, some security systems) might not be ideal applications. It might be better to pay the cost of establishing semantic agreements and use a traditional publish/subscribe system rather than leaving semantic approximation to the matcher.

Despite the challenges of building IoT software that overcomes event semantic variety in a loosely coupled manner, there are many practical reasons to build such software via thingsonomies for semantic normalization in an event-based middleware. Future work is to test the suitability of various corpora with respect to each domain, such as energy and traffic. This also includes the use of cloud computing and parallel processing to improve efficiency within applications that have real-time constraints. ◻

Acknowledgments

This work has been supported in part by Science Foundation Ireland under grant SFI/12/RC/2289.

References

1. L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A Survey," *Computer Networks*, vol. 54, no. 15, 2010, pp. 2787–2805.
2. P.R. Carlile, "Transferring, Translating, and Transforming: An Integrative Framework for Managing Knowledge across Boundaries," *Organization Science*, vol. 15, no. 5, 2004, pp. 555–568.
3. P.T. Eugster et al., "The Many Faces of Publish/Subscribe," *ACM Computing Surveys*, vol. 35, no. 2, 2003, pp. 114–131.

4. M. Gupta et al., "Survey on Social Tagging Techniques," *Proc. ACM SIGKDD Explorations Newsletter*, vol. 12, no. 1, 2010, pp. 58–72.
5. D. Carvalho et al., "EasyESA: A Low-Effort Infrastructure for Explicit Semantic Analysis," *Proc. 13th Int'l Semantic Web Conf.*, demonstration paper, 2014, pp. 177–180.

Souleiman Hasan is a PhD student at the Insight Centre for Data Analytics at the National University of Ireland, Galway. His main research interests include the Internet of Things, Semantic Web, event processing, and semantic matching. Hasan has a BSc in information technology engineering from Damascus University. Contact him at souleiman.hasan@insight-centre.org.

Edward Curry is a research leader and lecturer at the Insight Centre for Data Analytics at the National University of Ireland, Galway. His interests include the Internet of Things, enterprise-linked data, energy informatics, semantic information management, and community-based data curation. Curry has a PhD in computer science from the National University of Ireland, Galway. Contact him at edward.curry@insight-centre.org.