# Linked Data and Complex Event Processing for the Smart Energy Grid*

Andreas Wagner[1], Darko Anicic[2], Roland Stühmer[2], Nenad Stojanovic[2], Andreas Harth[1], and Rudi Studer[1,2]

[1] Institute AIFB, Karlsruhe Institute of Technology, Germany
{a.wagner,harth,studer}@kit.edu
[2] FZI Forschungszentrum Informatik, Germany
{danicic,stuehmer,nstojano}@fzi.de

**Abstract.** The Smart Grid aims at making the current energy grid more efficient and eco-friendly. The Smart Grid features an IT-layer, which allows communication between a multitude of stakeholders and will have to be integrated with other "smart" systems (e.g., smart factories or smart cities) to operate effectively. Thus, many participants will be involved and will exchange large volumes of data, leading to a heterogeneous system with ad-hoc data exchange in which centralised coordination and control will be very difficult to achieve. In this paper, we outline our envisioned Smart Grid architecture based on maturing Semantic Web technologies. Further, we show how Linked Data principles may be used for enabling decentralised publishing and resource discovery, ultimately fostering data integration. Current Linked Data principles apply to static sources but could be extended to include streaming sources as well. Dealing with streaming data, complex event processing enables timely reaction to new data, in order for the Smart Grid to quickly and efficiently react to complex and distributed data.

## 1 Introduction

The Smart Grid – a radical redesign of the traditional energy grid – aims at profoundly changing the way how energy is created, distributed and consumed, thereby saving a considerable amount of energy [3, 4].

An architecture for the Smart Grid has to be (1) flexible, i.e., fulfil customer requirements, but also allow for future extensions, (2) accessible, i.e., allow access to/from all participants, (3) reliable, i.e., assure quality of supply, and (4) economic, i.e., provide best value and allow for innovation and competition [3]. In other words, the energy grid has to become more flexible "like the Internet"[3] and allow for open access to data in the network to a large number of participants.

One of the results of the new Smart Grid is that more data (e.g., about consumption) is available to energy producers (e.g., for a more detailed billing), but also to other grid participants for improving the grid's quality and efficiency. Thus,

* This work was in part supported by the German Federal Ministry of Economics and Technology (MeRegioMobil, Grant 01ME09005) and the European Community's Seventh Framework Programme FP7/2007-2013 (PlanetData, Grant 257641 and PLAY, Grant 258659).

[3] *Building the energy Internet*, The Economist, May 11, 2004.

not one single participant, but many participants (e.g., power utilities, technology vendors, service companies, customers) will have control over the grid and the communication flows within. The Smart Grid participants need to organise data exchange (e.g., power consumption data for billing and planning). In addition, car and appliance manufacturers have the opportunity to collect detailed data about the day-to-day usage of their products to improve their design. Further, as factories and urban infrastructure are being upgraded with information technology, all these systems have to be connected and data from the other grids has to be integrated with data originating from the energy grid.

**Requirements.** Given the vision of a Smart Grid, we can derive several high-level requirements for a communication infrastructure:

- The Smart Grid needs flexible, open and light-weight data access procedures to enable seamless communication between Smart Grid participants, leading to a more efficient and innovative energy grid. Standards only available under restrictive licenses to a selected number of participants or over-specified regulations will stifle innovation and hinder the achievement of the overall Smart Grid goals. Thus, the communication standards should be open to facilitate the introduction of new products and energy production methods, and to lower the barrier for new participants entering the energy market.
- The new roles and communications processes within the Smart Grid require flexible data models, which enable a distinction between syntactic and semantic content. Further, the data access procedures and data representation should support large-scale data integration.
- Leveraging the rich data sources present in the Smart Grid, methods are needed to quickly and efficiently react to emitted events and to complex situations, which are only detectable by combining several events. Thus, awareness is created for situations, which would otherwise been unnoticed.

In light of the Smart Grid communication architecture requirements, we advocate an architecture based on Semantic Web technologies. The Semantic Web standards are widely used, well-known, accepted, available under royalty-free licensing, and therefore can provide a solid basis for applications in the Smart Grid. The schema-less and self-describing nature of semantic technologies facilitates flexible integration of new data schema and data integration and exchange. In addition, Linked Data technologies allow data publishing and integration in large, distributed environments.[4] Lastly, Semantic Web data is self-describing, which allows the description and modelling of events from a large number of sources (which can be unknown at design time). Events should be well structured and must be processed in a distributed network of event sources, processing agents and sinks, which all need an understanding of what an event signifies. Overall, Semantic Web technologies are suitable (or can be extended) to process streams of such events.

**Outline.** The rest of the paper is structured as follows: We introduce our communication architecture for the Smart Grid in Section 2, followed by an example scenario illustrating Linked Data in the Smart Grid in Section 3. In Section 4, we outline why complex event processing is needed in the Smart Grid and how it may be integrated in our architecture. We conclude with Section 5.

[4] `http://fi-ghent.fi-week.eu/fia-session-i-linked-open-data/`

## 2 A Semantic Web Communication Architecture for the Smart Grid

Employing a layered architecture leads to a more flexible and versatile Smart Grid communication, as varying technologies may be integrated and functionalities can be modified or replaced. Further, we wish an architecture to be decentralised and thus omit a single point of failure, in order to provide the desired reliability. In the following, we will argue that there are strong parallels between the Smart Grid communication architecture and the (Semantic) Web Stack – an adaptation of which would result in a layered and decentralised architecture.

### 2.1 Data Access Layers

In order to allow full access to/from all participants, we need a naming mechanism to uniquely identify each participant. Also, the Smart Grid needs flexible, open and scalable data access procedures. Flexibility means that a communication architecture should be able to facilitate heterogeneous participants employing hardware of lower or higher specification. Further, procedures only available under restrictive licenses to a selected number of participants might hinder innovation. Thus, standards should be open and royalty-free. As huge amounts of data are handled within the Smart Grid, data access procedures should be light-weight, i.e., scale well w.r.t. the data volume.

We advocate URIs for identification of participants. We employ a TCP/IP stack with HTTP as transfer protocol for establishing a connection and accessing data. However, standard Internet protocols are usually not adequate for low-power devices, due to their overhead from the various protocol headers. Thus, special protocols developed for low-power devices (e.g., sensors) may be adapted: e.g., a light-weight layered architecture such as IEEE 802.15.4 (physical and MAC layer), 6LoWPAN (internet layer, IPv6 version for IEEE 802.15.4 networks) or a single layer coupled with a middle-ware layer (for communication with TCP/IP networks), e.g., [5].

### 2.2 Data Representation Layers

Now, let us describe our data representation. We need structured and machine interpretable data models for representation of data semantics and context, in order to allow flexible data integration, thereby fostering the access of heterogeneous participants (e.g., employing different data schema). Furthermore, using formal data semantics, we allow advanced complex processing mechanism to be applied.

Therefore, we advocate the Resource Description Framework (RDF) [2] for expressing graph-structured data. RDF Schema (RDFS) adds additional expressivity in order to support the design of simple vocabularies encoded in RDF [1]. Finally, using Linked Data principles in compliance with [8], data described by means of RDF(S) can be related and integrated between different sources. The four Linked Data principles are: (1) Use URIs as names for things. (2) Use HTTP URIs so that people can look up those names. (3) When someone looks up a URI, provide useful information, using the standards. (4) Include links to other URIs. so that they can discover more things.

## 3 A Linked Data Scenario

In the following we outline a Smart Grid scenario illustrating how Linked Data enables innovative use of accumulated data. Further, hte scenario describes how data sharing in a decoupled energy market is supported.

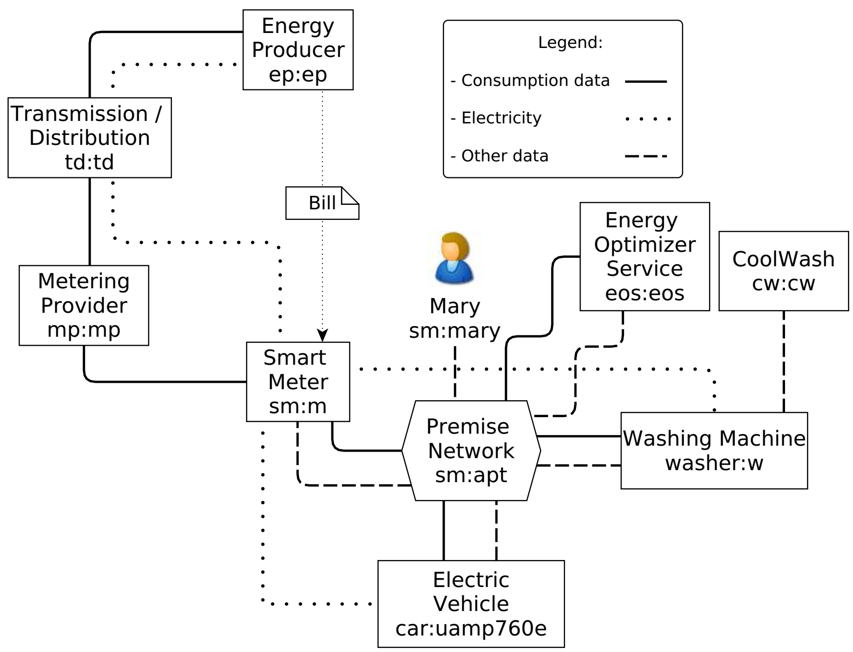


**Fig. 1.** Mary's Smart Home Example

Consider a consumer Mary, who lives at an apartment fitted with a smart meter. Mary owns a CoolWash washer and an UltraAmp 760e electric car (amongst other devices). The scenario is depicted in Figure 1. For our scenario we require that these devices are accessible via TCP/IP and have the following hostnames: `smartmeter.example.org`, `washer.example.org`, and `car.example.org`. We assume that each device is accessible via HTTP, and URIs identify each resource (e.g., a person, a vehicle, or an appliance). For instance, the URI denoting Mary is `http://smartmeter.example.org/data#mary`. Please note that we use namespace definitions as listed in Table 1 for brevity.

If we perform a lookup on Mary's URI, the server (in our case the smart meter) returns a RDF file describing the resource (i.e., Mary):

```
sm:mary rdf:type foaf:Person ;
  foaf:name "Mary Doe" ;
  foaf:based_near sm:apt .
```

| *Prefix* | *URI* | *Description* |
|---|---|---|
| sm | `https://smartmeter.example.org/data#` | Smart Meter |
| washer | `http://washer.example.org/data#` | Washer |
| car | `http://car.example.org/data#` | Car |
| cw | `http://coolwash.example.org/data#` | CoolWash Inc Data |
| eos | `http://optimiser.example.org/data#` | Energy Optimizer Service Inc Data |
| ep | `http://energy.example.org/data#` | Energy Producer Inc Data |
| td | `http://transmission.example.org/data#` | Transmission and Distribution Inc Data |
| mp | `http://metering.example.org/data#` | Metering Provider Inc Data |
| sg | `http://smartgrid.example.org/vocab#` | Smart Grid Vocabulary |
| xsd | `http://www.w3.org/2001/XMLSchema#` | XML Schema Vocabulary |
| rdf | `http://www.w3.org/1999/02/22-rdf-syntax-ns#` | RDF Vocabulary |
| rdfs | `http://www.w3.org/2000/01/rdf-schema#` | RDF Schema Vocabulary |
| foaf | `http://xmlns.com/foaf/0.1/` | Person Vocabulary |
| geo | `http://www.w3.org/2003/01/geo/wgs84_pos#` | Geo-location Vocabulary |
| ical | `http://www.w3.org/2002/12/cal/ical#` | Temporal Vocabulary |

**Table 1.** Namespace Abbreviations

A request on `sm:apt` returns more data pertaining to the premise (such as latitude and longitude or address). Requests on the washer URI (`washer:w`) provide data describing the appliance, including links to energy consumption data and data about the previously selected washing programs:

```
washer:w
  rdf:type sg:Appliance ;
  sg:manufacturer cw:cw ;
  sg:owner sm:mary ;
  cw:washingData washer:program40 ;
  sg:consumption sm:data20100310 .
```

A lookup on `washer:program40` returns:

```
washer:program40
  rdf:type cw:WashingData ;
  foaf:name "Program 40 C" ;
  cw:totalCount "23"^^xsd:int .
```

Energy usage data resides at the smart meter, so performing a lookup on `sm:usage2010031100` results in the following data snippet, indicating a consumption of 1.04 kWh during a late-night wash:

```
sm:data20100310
  rdf:type sg:Consumption ;
  rdf:value "1.04"^^sg:kWh ;
  ical:dtstart "2010-03-10T00:00:00" ;
  ical:dtend   "2010-03-10T01:00:00" .
```

In contrast to on-premise appliances, the UltraAmp 760e is mobile. We assume a TCP/IP connection to the car (e.g., via 3G), so requests may be performed as well.

A HTTP lookup on the car (`car:uamp760e`) may provide the model description and its current location:

```
car:uamp760e rdf:type sg:Vehicle ;
  foaf:name "UltraAmp 760e" .
  geo:location _:loc20100331 .
_:loc20100331 dc:date "2010-03-31T12:23:45";
  geo:lat "49.0047222" ;
  geo:lon "8.3858333" .
```

Note, a lookup targets a device directly, rather than requiring a centralised location, which warehouses all data. We assume that access to the smart meter is done via an encrypted channel (e.g., `https`), and recording of consumption data adheres to legal requirements.

Now, assume that the manufacturer of the CoolWash machine wants to request data about the washing machine (e.g., to optimise future versions of the appliance based on real-world usage), a metering system provider wants to request power consumption data (e.g., for billing), and an energy optimisation consultancy wants to request all energy-related data (to help Mary optimise her energy consumption). Thus, data sharing is essential in a decoupled market scenario, where not only a single participant has access to a customer's usage data, but many Smart Grid participants. Further, data sharing and integration is also needed for roaming, as Mary's electric vehicle may be charged at an off-premise charging station. The electric vehicle (`car:uamp760e`) identifies itself at the charging station, and the power consumption data is sent to the customer's utility company for billing purposes.

## 4 Complex Event Processing over Linked Smart Grid Data

In this section, we will briefly introduce complex event processing (CEP) and outline one of its use-cases in the Smart Grid (i.e., a home energy optimiser service). Further, we will show how existing complex event processing may be applied together with Linked Data, allowing real-time data integration on a large scale, thereby enabling the Smart Grid to (quickly and efficiently) react to complex, distributed data.

### 4.1 Complex Event Processing in the Smart Grid

**Complex Event Processing.** Recently, there has been a significant paradigm shift towards real-time computing. Previously, queries against databases and data warehouses were concerned with looking at what happened in the past. On the other hand, complex event processing (CEP) is concerned with processing real-time events, i.e., CEP is concerned with what has just happened or what is about to happen in the future.

An *event* represents something that occurs, happens or changes the current state of affairs. For example, an event may signify a sensor reading, the current energy consumption, a price-change signal, some piece of information becoming available, a deviation and so forth. An event can also represent something that did not happen (i.e., an absence of an event within a certain time frame).

The overall complex event processing goal can be described as follows: Within some dynamic setting, events take place. Those *atomic events* are instantaneous (i.e., they happen at one specific point in time and have a duration of zero). Notifications about these event occurrence, together with their timestamps and possibly further associated data (e.g., involved entities, numerical parameters of the event, or provenance data), enter the CEP system in the order of their occurrence. The CEP system further features a set of *complex event descriptions*, by means of which *complex events* can be specified as temporal constellations of atomic events. The complex events, thus can be used again to compose even more complex events and so forth. As opposed to atomic events, complex events are not considered instantaneous, but are endowed with a time interval denoting when the event started and when it ended. Now, the purpose of a CEP system is to detect complex events within an input stream of atomic events. That is, the system needs to notify that the occurrence of a certain complex event has been detected, as soon as the system is notified of an atomic event that completes a sequence, matching the complex event (according to the complex event description). This notification may be accompanied by additional information composed from the atomic events' data. As a consequence of a detection (and depending on associated data), corresponding actions may be taken.

We assume that in a Smart Grid scenario (especially related to a household), many unexpected events can happen, so that the whole "smart system" must be able to react properly on those events/situations. Let us illustrate the need for CEP on Mary's exemplary household from the previous section:

**A Home Energy Optimiser Scenario.** To plan an optimised use of energy, there are various data sources relevant for a home energy optimiser service (`eos:eos`): (1) the household appliances (e.g., `washer:w` or `car:uamp760e`), (2) the energy provider (`ep:ep`), (3) external information providers and (4) the customer (and her preferences) (e.g., Mary `sm:mary`).

For the service to operate, the household appliances need to communicate their energy demand for the next period. The demand includes the energy required for Mary's actions (e.g., a selected washing program) or automated actions (e.g., cooling her fridge). The energy provider sends a price signal (e.g., valid for the next 24 hours), energy consumption recommendations (e.g., recommendations to postpone energy consumption) or other power system related information to Mary. External information sources may provide data on typical consumption patterns (e.g., consumption in Mary's neighborhood or consumption of similar costumers), and information related to household appliances (e.g., data on the appliance programs, or optimal consumptions for a common set of the programs). Lastly, Mary may provide data on her consumption patterns (e.g., time-frame to run the washing machine), her schedule (e.g., vacations plans) or her individual wishes (e.g., activate a certain household device).

Using the above data (i.e., events) as input, the energy optimising service (`eos:eos`) should discover patterns of immanent high consumption from the log data and react before those patterns appear in reality. For example, the service could learn that Mary has bad habits of switching on appliances everywhere and thus could remind her of her behaviour (thereby creating awareness) or take other actions (e.g., change her appliance settings).

The energy optimising service may also change the consumption schedule in order to maximize its efficiency. To do so, the optimiser processes real-time data (e.g., the current energy price, actual consumption of Mary's appliances etc.) and compares them to optimal consumptions from external information sources. The service could recommend changing certain parameters in Mary's household (e.g., her consumption patterns and appliance settings), or indicate that some appliances and programs are suboptimal with respect to the current energy saving standards.

Finally, the optimiser service could mine Mary's data for interesting pattern (so-called *energy data mining*, which supports the process of discovering new, interesting consumption patterns). The most interesting patterns are outliers that indicate a "strange" behaviour in Mary's energy consumption. The outlier patterns should be recognized in real-time, in order to enable a timely reaction (e.g., Mary left her TV on and went to bed).

### 4.2 Complex Event Processing with Linked Data

While existing Semantic Web technologies and reasoning engines are constantly being improved in dealing with *time invariant* domain knowledge, they lack in support for processing *real-time* streaming data (events). Real-time data is valuable only if it is captured, processed, and delivered instantly.

The World Wide Web and Semantic Web are still inherently based on the *request-response* paradigm. That is, a user or a software agent poses a request and receive an answer accordingly. For instance, consider the data access and the data sharing in Mary's household (see Section 3).

Instead, in complex event processing, information is computed in real-time, and *pushed* to a user (or an agent) rather than being *pulled*. CEP is a set of technologies and practices, which enable users to receive information as soon as it is published (rather than requiring periodic updates). Hence, there is no need to *pull* information, it will be delivered to users nearly at the moment it is published. For instance, there will be no more waiting for web services to communicate from one polling instance to another.

We noticed a paradigm shift from information *pull* to information *push*; or from *request-response*-based to *event-driven* web services. However, before realising the paradigm shift, HTTP as a transfer protocol needs to be extended. Instead of an one-time request-response connection, a client needs to open a persistent connection to server, which can then send data as the data arrives. Currently, work in this area includes protocols such as Google's Pubsubhubbub[5], XMPP[6], HTTP Streaming[7], Comet[8], HTML 5 WebSockets[9] etc. We advocate such or other extensions of the HTTP protocol to enable streaming data in an RDF format. The

[5] Pubsubhubbub: `http://code.google.com/p/pubsubhubbub/`
[6] XMPP: `http://xmpp.org/xmpp-protocols/`
[7] HTTP Streaming: `http://ajaxpatterns.org/HTTP_Streaming`
[8] Comet: `http://en.wikipedia.org/wiki/Comet_(programming)`
[9] HTML 5 WebSockets: `http://dev.w3.org/html5/websockets/`

extension will enable real-time (push) processing. Also, it could be the foundation for a *reasoning* service over both, RDF(S) streaming and static data.

Complex event processing, extended with reasoning capabilities, has potential to enable real-time intelligence in the Smart Grid. Current CEP systems cannot reason, and hence cannot utilise the various datasets and ontologies available today (e.g., via Linked (Open) Data). To address this issue and provide a CEP framework with inference capabilities, we have proposed ETALIS Language for Events [6] and provided an open source reference implementation[10]. ETALIS can process sensor streaming data in order to detect complex events (e.g., Mary left her stove on over night). Further, it can process static RDF data such as descriptions about a user, household, appliances, and other information sources (see Section 3), in order to detect the context in which sensor data is processed, and reason accordingly.

Enabling flexible CEP on the Web requires at least the following two building blocks:

- *Description of event sources*: a CEP engine requires data about event sources (e.g., network address of the sensor, update frequency or physical location). Additional information (such as the responsible person or organisation or licensing information) can be used to enable elaborate queries. When publishing these source descriptions as Linked Data (and hence interlinking source descriptions), decentralised discovery of new sources becomes possible. The W3C Semantic Sensor Network Incubator Group[11] has a proposal for an RDF representation of such source descriptions.
- *Common access mechanism*: a common access protocol facilitates integration of streams from multiple sources. Linked Data mandates the use of HTTP as transport protocol. HTTP 1.1 supports persistent connections [12], which allow client and server to keep a connection open for an extended period of time. Such persistent connections could be used to stream events from a server to a client. Ideally, such a scenario would use RDF in a serialisation amenable to line-by-line processing (such as Turtle [7]).

### 4.3 Distributed Complex Event Processing

In the Smart Grid events occur at many distributed sources and reactions are needed at many levels of distribution. Distributed processing in CEP helps scale CEP in order to fulfil requirements posed by the deployment in the Smart Grid. Distributed CEP could help to minimise network traffic by performing its CEP operations as close to the data sources as possible [11].

Our approach at building a distributed CEP system relies on the concept of the event processing network (EPN) (named by Luckham in [10]) and further described in [9]. On the one hand, the EPN is a concept of how to structure and describe any event processing system. On the other hand, the EPN is a natural fit for distribution. An EPN consists of event processing *agents* and *channels*, i.e., nodes and edges. Event processing agents (EPAs) perform operations on

[10] ETALIS: `http://code.google.com/p/etalis/`
[11] `http://www.w3.org/2005/Incubator/ssn/`
[12] `http://www.w3.org/Protocols/rfc2616/rfc2616-sec8.html`

events (e.g., filtering, enriching, detecting patterns, etc.). Channels connect EPAs. Finding strata of independent EPAs in the network, a horizontal partitioning can be derived. Events are then routed to subsequent strata [9].

Overall, we believe that CEP in a distributed fashion is a well-suited technology regarding several key requirements of the Smart Grid.

## 5 Conclusion

In this paper, we have argued that royalty-free (Semantic) Web standards can provide the foundation for the Smart Grid communication architecture. We have presented a scenario applying Linked Data principles for Smart Grid data publishing and access, resulting in a distributed system, where any participant may gain access to data in a flexible manner. Complex event processing fosters flexibility, robustness and efficiency in the Smart Grid via constant monitoring over (dynamic) patterns (e.g., outliers in energy consumption), in order to react and adjust to new events in the grid as soon as possible (e.g., prevent an energy outage via early discovery and countermeasures).

## References

1. *RDF Vocabulary Description Language 1.0: RDF Schema*. W3C Recommendation, 2004. `http://www.w3.org/TR/rdf-schema/`.
2. *Resource Description Framework (RDF): Concepts and Abstract Syntax*. W3C Recommendation, 2004. `http://www.w3.org/TR/rdf-concepts/`.
3. European Technology Platform - SmartGrids Vision and Strategy for Europes Electricity Networks of the Future. European Comission, 2006.
4. NIST Framework and Roadmap for Smart Grid Interoperability Standards. National Institute of Standards and Technology, 2010.
5. K. Aberer, M. Hauswirth, and A. Salehi. A middleware for fast and flexible sensor network deployment. In *VLDB*, pages 1199–1202, 2006.
6. D. Anicic, P. Fodor, S. Rudolph, and N. Stojanovic. A rule-based language for complex event processing and reasoning. In *Conference on Web Reasoning and Rule Systems (RR 2010)*, 2010.
7. D. Beckett and T. Berners-Lee. *Turtle - Terse RDF Triple Language*. W3C Team Submission, 2008. `http://www.w3.org/TeamSubmission/turtle/`.
8. T. Berners-Lee. Linked data design issues. Technical report, W3C, 2006. `http://www.w3.org/DesignIssues/LinkedData.html`.
9. G. T. Lakshmanan, Y. G. Rabinovich, and O. Etzion. A stratified approach for supporting high throughput event processing applications. In *Conference on Distributed Event-Based Systems*, 2009.
10. D. C. Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.
11. P. R. Pietzuch, B. Shand, and J. Bacon. Composite event detection as a generic middleware extension. *IEEE Network*, 18(1):44–55, 2004.