A Semantic Infrastructure for Personalizable Context-Aware Environments

Simon Scerri, Jeremy Debattista, Judie Attard, Ismael Rivera

■ *Although a number of initiatives provide personalized context-aware guidance for niche use cases, a standard framework for context awareness remains lacking. This article explains how semantic technology has been exploited to generate a centralized repository of personal activity context. This data drives advanced features such as personal situation recognition and customizable rules for the context-sensitive management of personal devices and data sharing. As a proof of concept, we demonstrate how an innovative context-aware system has successfully adopted such an infrastructure.*

Today's wide range of smart personal devices and online services generate a constant stream of information that can indicate a person's plans, activities, and situations. By treating these devices as part of a personal sensor network and analyzing the generated information collectively, valuable context information can be gathered and interpreted in an endless number of scenarios. Efforts by both private industry and research communities have produced a wide variety of similar context-aware techniques and services. However, this technology remains strictly tied to a native application, system, or device, thus limiting its reuse and integration to address new scenarios.

To address the above problem, we propose a semantic infrastructure for context-aware systems, based on machine-processable ontologies that cater for personal information scattered across multiple personal sources. Moreover, the semantic infrastructure also covers context-related information obtained from the personal sensor network. Thus, context information created by numerous disconnected platforms, including social networking services (for example, check-ins, tags), physical data sensors (for example, GPS), and legacy tools and applications (for example, calendar), can all be integrated and processed centrally. Domain models covering some of these personal information domains

 ISSN 0738-4602

already exist, and have been adopted as a lingua franca by a number of pervasive system architectures. Therefore, we propose a new semantic layer to interpret this information within the personal context dimension. The resulting context representation can be utilized by any service that is able to process data expressed in the resource description framework (RDF)[1] format.

In this article we demonstrate how the above strategy enables us to provide various levels of intelligent support. In particular, we explain how it has been applied to develop a context-aware system that learns how to recognize the various situations between which people alternate in their life. For the more tech-savvy users, we developed avant-garde user interfaces that extend the simplistic concept of email filters to a person's entire digital sphere. This enables the creation of context-driven rules to specify which action should take place given one or more perceived conditions, for example, change device to silent mode when entering the office or cinema, change your online presence to available when leaving, or provide notifications when someone posts something about you on social networks.

A Semantic Infrastructure

Ontologies have been proposed as common domain models for exchange within a collaborative infrastructure (Fuchs et al. 2008). In line with this approach, we make us of a comprehensive ontology framework as the foundations for a semantic collaborative infrastructure. The framework is described in the Modeling the Personal Information Sphere subsection that follows.

To enable multiple devices and accounts as nodes in a personal sensor network, the representation of information items is extended within the context dimension. When using the term *context*, we adhere to a definition by Dey (2001), and refer to any kind of information that can be used to characterize the situation of an entity. The Di.me context ontology (DCON [2] extends information items with temporal data to indicate which of them characterize a person's current activity, task, or situation. An overview of DCON's knowledge-modeling capabilities is provided in the second subsection.

Once the distributed personal information items and the various context changes affecting them are represented using the same format, we ventured to take the email filters paradigm to a new level. Using these information items as blocks, we modeled an ontology that is capable of wrapping them as a series of conditions that trigger one or more desired actions when fulfilled. This has the effect of enabling the regulation and management of all known personal items. The di.me rule management ontology (DRMO),[3] described later, is responsible for this knowledge transition.

Modeling the Personal Information Sphere

The ontology framework consists of a number of vocabularies, a majority of which were employed by the social semantic desktop[4] and the digital.me[5] projects. Personal information gathered from the distributed personal sources is semantically lifted onto a unified and dynamically updated representation of a user's personal information model (PIM), modeled as an instance of the PIM ontology (PIMO) (Sauermann, van Elst, and Dengel 2007). The PIM maintains an integrated personal knowledge base (KB) containing all of your personal information, including data stored in your devices and online accounts. The Di.me device ontology (DDO) and Di.me account ontology (DAO) are used to represent each personal information source that is registered. Information items extracted from them are represented by the Nepomuk information element ontology (NIE) domain ontologies,[6] which comprehensively model files (Nepomuk file ontology, — NFO), events (Nepomuk calendar ontology — NCAL), tasks (Nepomuk task model ontology — TMO), address books and profiles (Nepomuk contact ontology — NCO), multimedia (Nepomuk multimedia ontology — NMM), and messages (Nepomuk message ontology — NMO) on personal devices (Sintek et al. 2009); and social network activities extracted from personal online accounts (Di.me livePost ontology — DLPO) (Scerri et al. 2012).

The aforementioned PIM items are described with context-independent metadata, for example, file size and name, person name and address. DCON is used to enhance them with context-dependent information, for example, the file is being modified, the person is nearby. A single instance of DCON, the live context, is used to maintain a centralized representation of a person's activity context. Thus, for example, current or upcoming events extracted from a calendar service (as NCAL instances) are attached to the live context. Files currently being edited and running applications (both handled as NFO instances), as well as locations (check in's) and people (tagged) from the latest social network microposts (as DLPO instances), are also streamed to the same representation.

Apart from wrapping existing PIM items, DCON also provides vocabulary for information that is more transient in nature, for example, current temperature, geographical coordinates, and others.). Although this kind of information does not merit long-term representation in the PIM, it is nevertheless valuable for machines to understand a person's situation. In order to facilitate its interpretation, discrete context values retrieved from sensors (for example, temperature, time of day/year) are mapped into predefined categorizations (Hot, Late Evening, Weekend, and so on) provided by the Di.me Presence Ontology (DPO). This ontology models high-level context-related information whose level of abstraction makes it independent of time. This makes it eas-

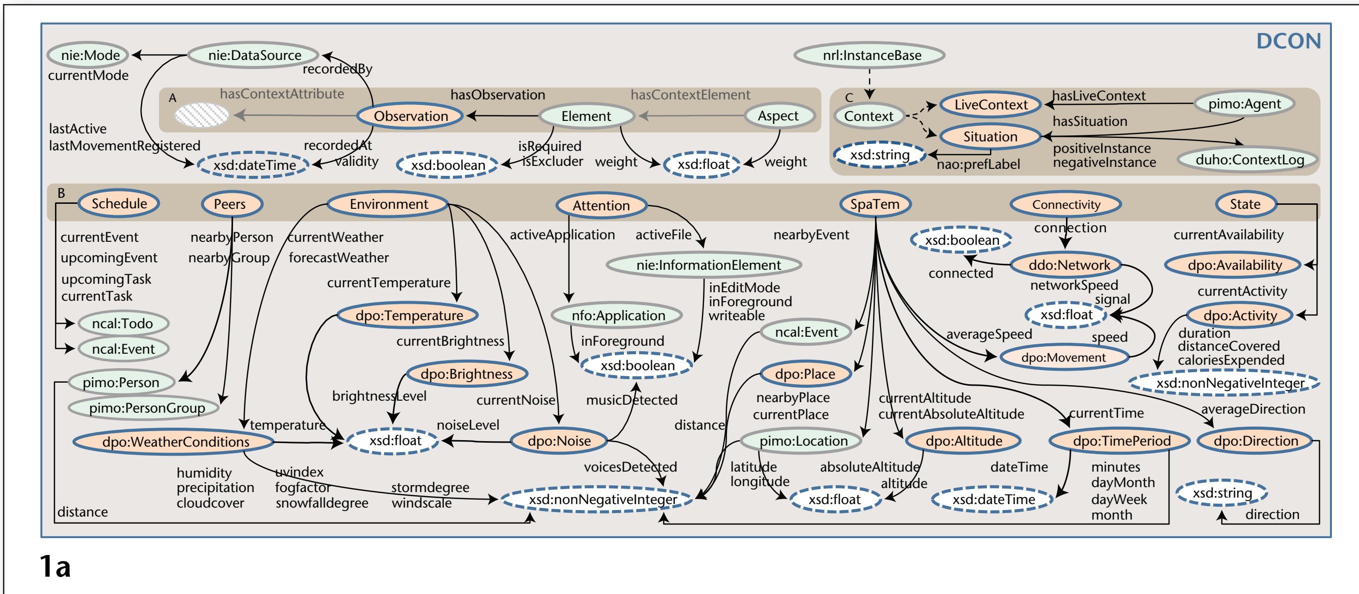
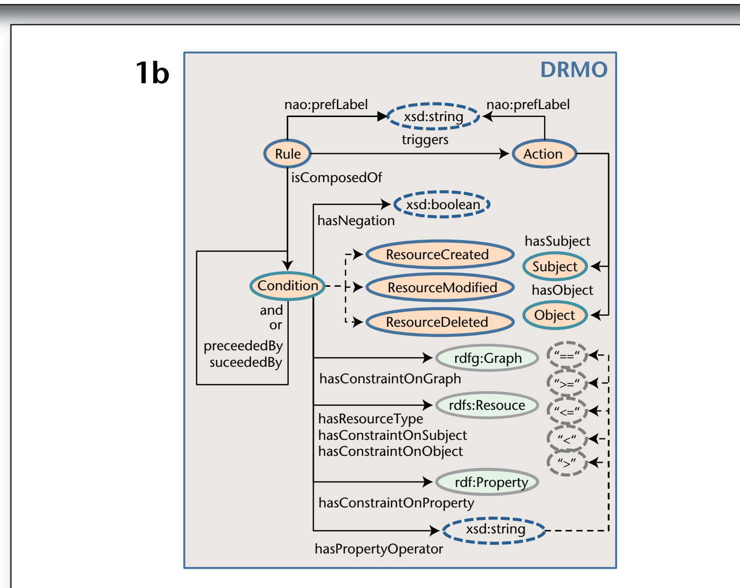


Figure 1. DCON (Top) and DRMO (Right)



ier to interpret than a specific DCON value. A common property of all DPO representations, which range from weather categories to generic user activities (for example, Working, Sports), is that unlike the raw context values, they can recur.

The distinction between inactive PIM items introduces the need to persist multiple time-dependent representations. This enables machines to compare current circumstances with past situations. The Di.me User History Ontology (DUHO) enables past live context snapshots to be logged. This ontology is crucial for the situation-recognition task, which requires past situation examples.

Modeling User Activity Context

DCON, visualized in figure 1, aggregates PIM items that are part of the current context, such as a connected WiFi network (ddo:Network), the detected location (pimo:Location), a DPO temperature category (dpo:Temperature). These items are also enriched with context-dependent properties, such as current signal strength, the exact temperature. As shown in the gray box, marked A, an entity's context is made up of a number of aspects, each of which refers to a particular set of context elements. This categorization of context information is inspired by a simpler context model designed by Schwarz (2006).

DCON defines seven types of aspects (gray box B): Schedule (recent/current tasks and events), Peers (nearby individuals/groups of people), Environment (ambient temperature, light and sound, and current/forecast weather), Attention (files/applications in use), SpaTem (live spatial-temporal information), Connectivity (connected networks), and State (online availability and detected activities). These aspects are abstract concepts whose purpose is to classify and better structure the perceived context elements, for example, temperature readings, known calendar events, detected networks. As shown in figure 1, context elements are linked to concepts provided by the earlier-described ontologies (for example, ncal:Event, dpo:Temperature).

Elements possess a number of context attributes, which define them (for example, exact temperature reading, network signal strength). Whereas time-independent attributes (such as network name, event date) are provided by the domain ontologies, each of their observations is extended with context attributes (shown as a blank object in gray box A). Furthermore, since elements can be detected simultaneously by one or more context sensors, we also introduce the intermediary concept of observations. As a result, as shown in figure 1, context attributes are not directly attached to the element representations, but to their observation(s). Thus, if a person is carrying two devices, each of which is able to detect the environment's temperature (context element), DCON is able to handle both temperature readings (attributes) simultaneously.

Each observation is linked to a data source, timestamped, and also carries a validity period, which is predefined depending on the element type. Based on these periods, a garbage collector can remove outdated information. For example, a Wi-Fi in range (connectivity aspect) is connected to one device (observation 1) but not another (observation 2). For each data source, DCON also registers the current mode (such as silent mode), the time of last activity and last registered movement.

Aspects and elements can be assigned a specific weight. Weights are not relevant for the live context. However, they are crucial in the characterization process of stored situations, as will be described later. Situations are approximated based on one or more of their instances (gray box C). Instances can be positive (that is, situation examples) or negative (counterexamples), and consist of past live context snapshots. Thus, at the schema level, live contexts and situations have few differences. However, the former is meant to be unique and up to date, whereas as many distinct situations as required can be stored. Moreover, situations are independent of time since they can recur. To improve the characterization process, some elements can be marked as excluders or required — meaning that their observation, or lack of, excludes a situation from the candidates.

Building Blocks for Context-Driven Rules

The Di.me rule management ontology (DRMO) is able to string PIM concepts and items as conditions for personalisable context-driven rules. Using both time-independent and time-dependent attributes provided by the ontology framework, it allows for each condition to be customized by selecting those circumstances under which the condition holds true. For instance, when adopting a specific person as a condition, one can specify that the person must be nearby (a time-dependent attribute provided by DCON). However, if adopting the person concept as a condition, one can filter this to state also that the person must be a member of a specific group (a time-independent attribute specified by PIMO). Adopting concepts rather than specific items is akin to enabling wildcards — in the above example, signifying that any nearby person satisfying the membership attribute will match the condition. We refer to these filters as constraints. Multiple conditions can be strung together, with the ultimate objective of firing predefined actions when all the conditions are met.

DRMO, also shown in figure 1, supports all the above concepts and relationships. Conditions can be joined together using four logical operators: *and* (both joined conditions have to occur), *or* (occurrence of any joined condition), *succession* and *precession* (a condition must occur before, or after, another). DRMO conditions can also be negated. The ontology supports three general types of conditions, corresponding to PIM items (resources) being created, modified, or deleted. DCON changes are always classified as modified items, for example, a live context aspect registers a new person detected nearby (attached to the Peers aspect). Item creation and deletion are strictly reserved for PIM items (for example, new email reaches the inbox, a new file is created or deleted).

A condition can be flexibly constrained in two different ways. A constraint can be placed on the condition as a subject or the object of an RDF triple, together with the specified property. Thus, the constraints on object or subject attributes shown in figure 1 are always used in conjunction with the constraint on property attribute. We explain the purpose of these constraints through a simple example. If a person (PIMO) condition needs to be filtered to say that the person must belong to a particular group, a constraint on object is created to look for a triple stating that the person (subject) is a member of (property) a group (the constrained object). Inversely, the same filter can be achieved by constraining the person condition such that a group (constrained subject) contains (property) the person (object). Constraints may also have relational operators to filter out datatype values rather then relationships, for example, to denote a person (subject) having a trust value (property) that is greater than (the property operator) a fixed value, or a file-type condition to match files containing (property operator) a specific keyword.

The semantics of DRMO actions are meant to be understood by a context-aware system, for example, an email action should send the respective email. Since in several cases items in a rule's condition are linked to the action, DRMO defines an action subject and object. In the example, the email message is the subject, and the recipient is the object.

Situation Recognition and Adaptation

To enable situation recognition, the live context is compared to stored situations at a configurable regular interval to return a similarity score. Both representations consist of DCON instances and are stored as RDF graphs. As we explain below, the comparison takes into account all DCON schema levels: aspects, observations, elements, and attributes. In addition, we also describe a technique for the semiautomatic characterization of situations, given that it is input for a number of positive and negative training examples.

A Context Similarity Score Function

A context-matching function determines the goodness of fit of the live context *l* against each candidate situation $s \in S$. The matching is centered on the information characterizing *l*, comparing each bit of information to the corresponding contents of each *s*. The latter are prefiltered by considering the two special properties described at the end of the DCON description section. If a context element present in *l* is marked as an excluder in *s*, the latter is immediately removed from the matching queue. Inversely, if a context element is marked as a requirement but is not present in *l*, this candidate situation is also discarded.

The matching algorithm, which is fully described in Attard, Scerri, Rivera, and Handschuh (2013a), can be summarized as follows. Each DCON aspect (for example, Connectivity) in *l* is compared against the same aspect in each of the remaining *s* graphs. This comparison depends on the elements nested within the aspect, organized by their role. For example, the DCON (nearby) Peers aspect can refer to two lists of elements: detected nearby people (role 1), and detected nearby groups (role 2). In this case, two confusion matrices are generated to compare the elements for each role. The average of the values within each matrix is then returned as the role similarity score. In turn, the average of these scores yields the aspect similarity score. Finally, the aspect score average returns the overall situation similarity.

As explained earlier, DCON supports the detection of elements from multiple devices. For example, a nearby person can be detected through network-based proximity on a device, but also through social tagging on a social network. Thus, rather than comparing elements in *l* directly with the corresponding elements in *s,* the algorithm also takes into account the provenance of each observation. This is based on the assumption that people tend to carry around the same devices in repeat situations. For example, if a person habitually leaves his or her laptop in the office during lunch break, but carries a mobile device along, a repeat situation will have a higher chance of being recognized if the observations deriving from the same device are given precedence. Essentially, this means that the algorithm also factors in the source of the sensed data, and gives matching devices a higher influence in computing context similarity. Specifically, scores for observations deriving from the same or different device are squeezed into a range of between 50 to 100 percent and 0 to 50 percent similarity, respectively.

At the deepest level, the similarity of each observation is calculated by considering the overlap between their attributes. Attribute comparison is not always trivial, since their data types vary (for example, network signal, person distance). For this reason, different constraint-based matchers are employed depending on the type and specified range (for example, network signal standard range, temperature ranges defined by DPO). The range is sometimes only partially specified (for example, no upper limit for the physical distance attribute). Other comparisons employ algorithms, for example, to determine the offset of two locations based on longitude and latitude.

The matching algorithm also employs a system of weights, as supported by DCON, which give a positive or negative bias to elements and attributes that are more representative of a situation. Element weights are initially neutral but can be adjusted semiautomatically after observing that, for example, for the situation Working@Office the office location is highly important, whereas the number of people nearby is not so important. The range of element weights varies between –1 and 1. Attribute weights are predefined by the DCON schema and give more prominence to those attributes that are indicative of an element's contribution to context. For example, a nearby Wi-Fi signal (connectivity aspect) is more indicative of a user's situation than its connection speed.

As reported by Attard, Scerri, Rivera, and Handschuh (2013a), the automatic matching function was evaluated against the manual (de)activation of personal situations by people participating in a trial, which lasted two weeks. Results indicate that the technique is not suitable to recognize deterministically a recurrent situation, since only 54 percent of automatically matched situations corresponded to the manual gold standard. However, the matching technique identifies the highest-matching situations in a majority of cases (70.2 percent), making it suitable for context-aware systems to suggest possibly recurring situations.

Semiautomatic Situation Characterization

Given that a means is provided for the training of situations, we define a semiautomatic situation adjustment function. Situations can be trained by manual (de)activation through a user interface (UI) to generate negative and positive examples. The adjustment is effected by adding previously unobserved elements and increasing or decreasing their weights as appropriate. The technique, presented by Attard, Scerri, Rivera, and Handschuh (2013b), gradually adapts the element weights using an inverse trigonometric function, which generates steeper increases for weights in the vicinity of 0, slowing down when nearing 1 or –1. If the situation described in the earlier example is activated, it is subject to the following changes. First, person A, who was observed in *l* but is not yet in *s,* is added to the latter with a neutral weight (0). Second, the above weight function is applied to adjust the weight for person B.

The situation characterization technique was evaluated, and the full results reported by Attard, Scerri, Rivera, and Handschuh (2013b). Through the same

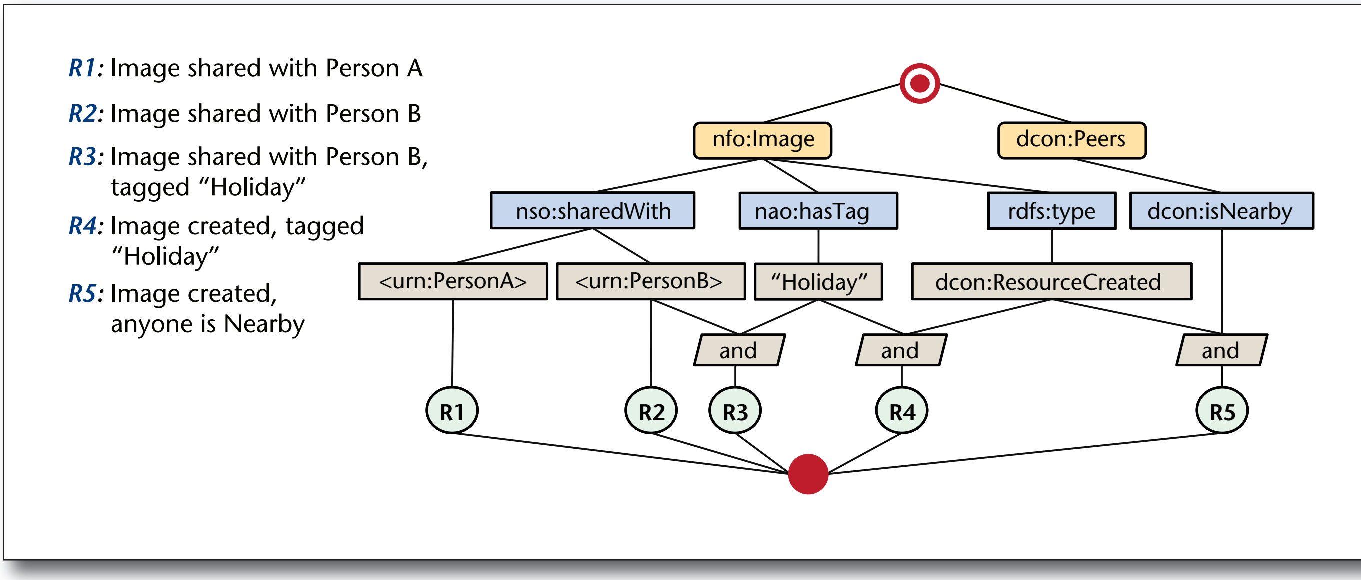


Figure 2. An Example of a Rule Network Embodying Five Rules.

exercise described in the previous section, live context representations were persisted each time an evaluator (de)activated a situation. These examples were all used to simulate the characterization of the situation. The context matching technique was then reapplied to match each of the live context representations against the modified situation. The results indicate that there is further room for improvement, since although the similarity of 78.6 percent of the positive situation examples increased, so did 50 percent of the negative ones.

Ontology-Based Event Processing

We now describe a technique that compares perceived PIM events to the conditions of DRMO rules. Events include PIM item changes (for example, file changed, modified or deleted) and all context changes supported by DCON (for example, file opened, nearby person registered, and so forth). For the comparison to take place, the rules are loaded at runtime and kept in memory. To optimize this process, a rule network (RN) is constructed to minimize search time. New events are broadcast directly to the RN, which is processed with each change to determine whether any of the rules should fire.

A Rule Network Generator

To generate the RN, each rule is decomposed into conditions and their constraints (refer to the DRMO description). Each distinct condition type is attached to the network's root as a topmost node. In the example shown in figure 2, the condition types are Nearby Person (represented by the DCON Peers aspect), and Image. To each of these condition nodes the constrained properties are attached, followed by the constrained object. This trio (condition, property, object) forms the basis for querying RDF triples in the PIM graph. However, the DRMO value constraint is optional, and its absence denotes a "wildcard." For example, as shown in figure 2, Rule 5 (R5) entirely omits constraints, effectively expressing that a personal device needs to register the presence of any nearby person for the rule to fire.

The join nodes join conditions and constraints that need to be activated in parallel. For example, the *and* join node leading to R5 joins this rule's two constrained values, to say that an image needs to be created while a person is nearby. Join nodes do not correspond directly to the earlier introduced DRMO condition operators. In particular, DRMO conditions joined by the *or* operator will result in two distinct rule nodes; for example, R1 and R2 in figure 2 could derive from one DRMO instance saying that an image is shared with person A or person B. This decreases response time due to fewer triple patterns being added to the queries.

A join node refers to two ordered inputs. Conditions joined by the *succession*- and *precession*-type operators generate two specialized join nodes, which indicate which condition should occur first: the one on the left in the former case; the one on the right in the latter. Other join nodes have no restraint on their order. For each input, the intermediate query and intermediate result is stored. These are used as a shortcut to deactivate entire network paths or subpaths once a partial condition is no longer satisfied. The intermediate query returns a result for each of the inputs. Unless both queries return a valid result (that is, they match perceived events) the (sub)paths leading to the join node are deactivated.

At the rule-node level, the resultant rules are transformed to a SPARQL query.[7] SPARQL enables the fast querying of PIM items corresponding to the registered events. More details on the RN generator and the SPARQL transformation have been published in the paper by Debattista, Scerri, Rivera, and Handschuh (2013).

Checking Events Against the Rule Network

With each event perceived by the personal sensor network, a new entry is broadcast to the RN. This includes the event's resource type (for example, Person, Document), the event operation (for example, person is nearby, document modified), a time stamp, and a pointer to the PIM location where the resource is stored (with the exception of items that have just been deleted). The latter is crucial since it enables a context-aware system to locate the item in the case of actions that require its manipulation, for example, changing the person's trust level, and others.

The lifetime of an event is regulated so that the RN is filtered only based on the latest events. We adopt the consumption mode, maximum event lifetime, and time-based windows techniques from the paper by Walzer, Breddin, and Groch (2008). The first retains the most recent events in a join node; the second expires them after a predefined maximum amount of time; the third expires events after a predefined time window based on their type (refer to the DCON validity property described earlier). In addition, a garbage collection service discards expired partial results stored in the join nodes, for example, context-dependent events registered in the live context following a further change (for example, a registered nearby person is no longer detected).

The RN processing algorithm, fully described by Debattista, Scerri, Rivera, and Handschuh (2013), checks for rule activation with each perceived event. Candidate rules are filtered (PL) using the K-shortest path algorithm (Yen 1971). The algorithm discovers common subpaths (C_p), thus reducing the number of PIM queries required. It iterates through the set of ordered paths, executing queries stored in the rule nodes. If a path has a join node, the list of shortest paths is checked for the first ordered input. If a query returns a result, a rule has been triggered. Since rules sharing the same condition are represented by the same branch in the network, when a condition is matched, all other paths are excluded from further iteration.

In the paper by Debattista, Scerri, Rivera, and Handschuh (2013), a network-based event processor was compared to a sequential approach. Results show that the RN is still initialized within an acceptable time-frame (~ 760 milliseconds for 50 rules). It also performs much better than the alternative, taking the same amount of time (<1.5 milliseconds) to process rules having between 1–16 distinct event types. A load test indicates that this event processor can process 100 parallel events (which is within the bounds of the envisaged use) in less than 0.1 seconds, with up to 10,000 events being consumed and processed in around 2 seconds.

An Ontology-Based Context-Aware System

A contribution of the Digital.Me project, the di.me userware[8] is a context-aware personal information management system that targets scattered data from multiple personal devices and online accounts. The userware is a proof of concept for the context-aware models and technology presented in this article since it employs the described ontologies, context-matching, and event processing techniques.

The decentralized di.me architecture (Thiel et al. 2012) enables peer-to-peer sharing of distributed personal information. Each person in the di.me network owns a personal server containing his or her PIM representation. Figure 3 shows a simplified version of the architecture, focusing on components that enable its ontology-driven context-aware features. An information extractor extracts information from the registered personal sources. This information is semantically lifted onto the ontologies, and the PIMO service stores the result in a PIM graph containing time-independent descriptions of various PIM items. A Context Listener is responsible for registering all context changes from the personal sensor network, streaming them onto the PIMO service to update the centralized live context graph. The context listener applies the described technique for situation matching and adaptation. On system initialization, the event processor (EP) generates a rule network based on the stored DRMO rules. With each event registered by the PIM and live context graphs, the EP filters the rules to determine if any one of them has been satisfied, upon which the corresponding action(s) is retrieved and executed.

The di.me userware employs a user interface (Scerri et al. 2013) that enables people with basic computer proficiency to make full use of the powerful features described in this article. Complex ontological knowledge is abstracted between various UI items corresponding to different PIM items, DCON elements, and their various attributes. As described in the following paragraphs, these enable the customization and visualization of personalized situations and context-driven rules. Videos demonstrating the described functionality (among others) are also available online.[9]

Situation Suggestion and Visualization

Personal situations can be saved through a button in the main UI dashboard. Saved situations are identified by personal labels (for example, Working@Office). On saving, context elements registered by the live context are automatically added to the situ-

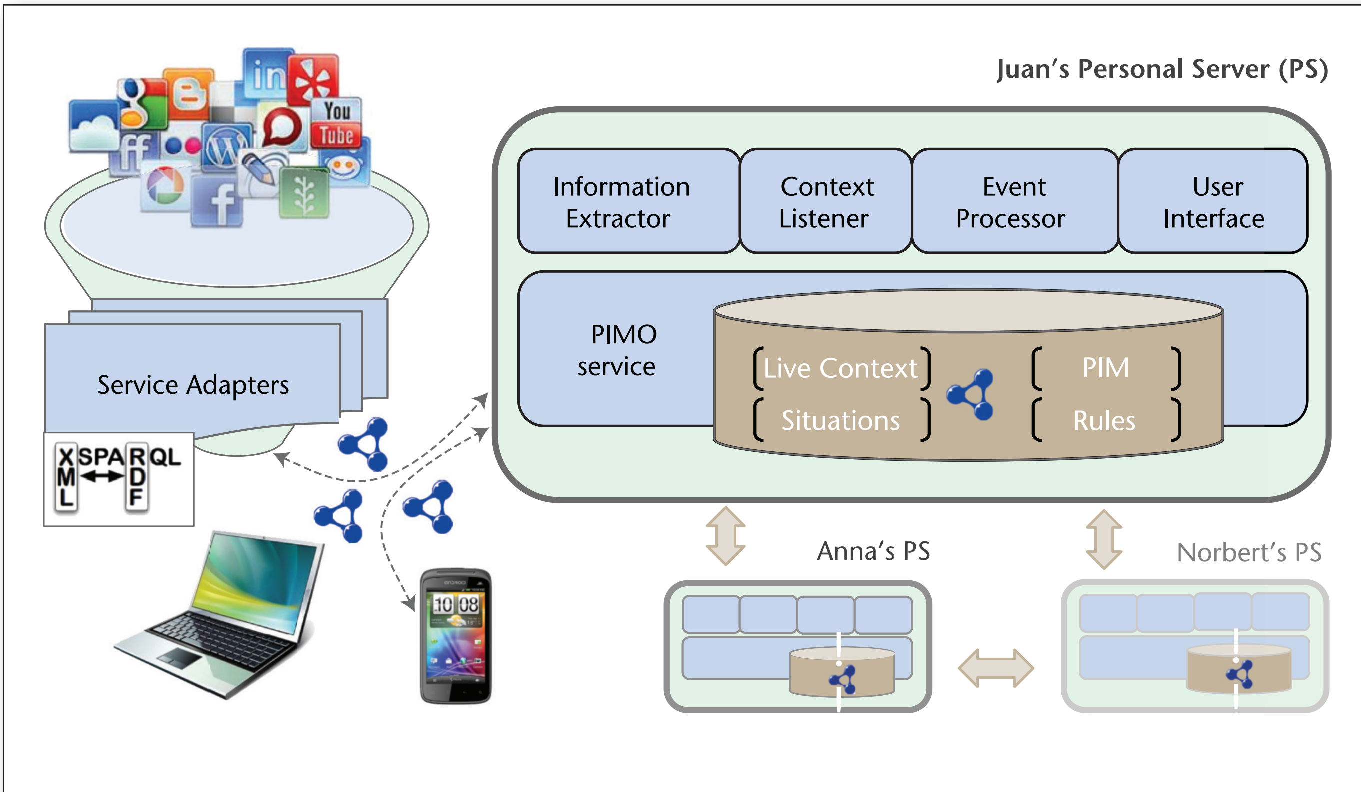


Figure 3. di.me Userware Architecture.

ation's representation, and their weights initialized to zero. Future versions of the prototype will allow owners to remove irrelevant context elements (for example, Sunny weather), and manually modify their weights (for example, Anna is more linked to this situation than Juan). Figure 4, left, shows an example of situation visualization in di.me. As indicated by the weights, the situation has already been trained.

The Context Listener compares situations with each context update. A list of ranked situations is returned and is easily accessible through the main dashboard. The score (0–100 percent) indicates their similarity to the current live context, as shown in the right side of figure 4. Through the shown bar, users can (de)activate situations. This action results in the storing of further positive/negative examples, which are then used to semiautomatically improve the situation representations.

Personalizable Context-Driven Rules

The Rule Manager, accessible through the settings tab, enables nontechnical people to create customized context-driven rules. It is designed as a Lego-like UI that enables owners to drag and drop objects and apply filters. The left side of figure 5 shows the following completed rule: "IF (friends are nearby) and (I'm at a social event) and (I take a photo) THEN [Ask if I want to share it with my friends]." The UI guides the owner to define each condition, starting by selecting an object (the condition type). In the example, the first condition relates to a nearby person, the second to a situation, the third to an image. Depending on the dragged object, users are then shown available filters (the property to be constrained). Depending on which filter is selected, the UI then shows the available items. For the Nearby Group filter, all saved groups are returned, including the one selected: Friends.

The aforementioned behavior is subject to some exceptions. For example, when a situation is selected as an object (for example, Social Event in figure 5) the filter selection (Situation Activated) is skipped since it is implied. Similarly, when the New Item filter for PIM items (for example, documents and images) is selected, the items step is skipped since it is implied that the item to be matched does not yet exist. The shown example includes three conditions, which are by default joined by the only supported DRMO logical operator: *and*.

The selection of an action finalizes the rule. A message is requested so that when the rule triggers it, it is sent to the di.me notification stream. The notification for the shown rule is shown in figure 5, bottom. When a rule is saved, the di.me rule manager wraps it as a JSON[10] representation and sends it to the EP.

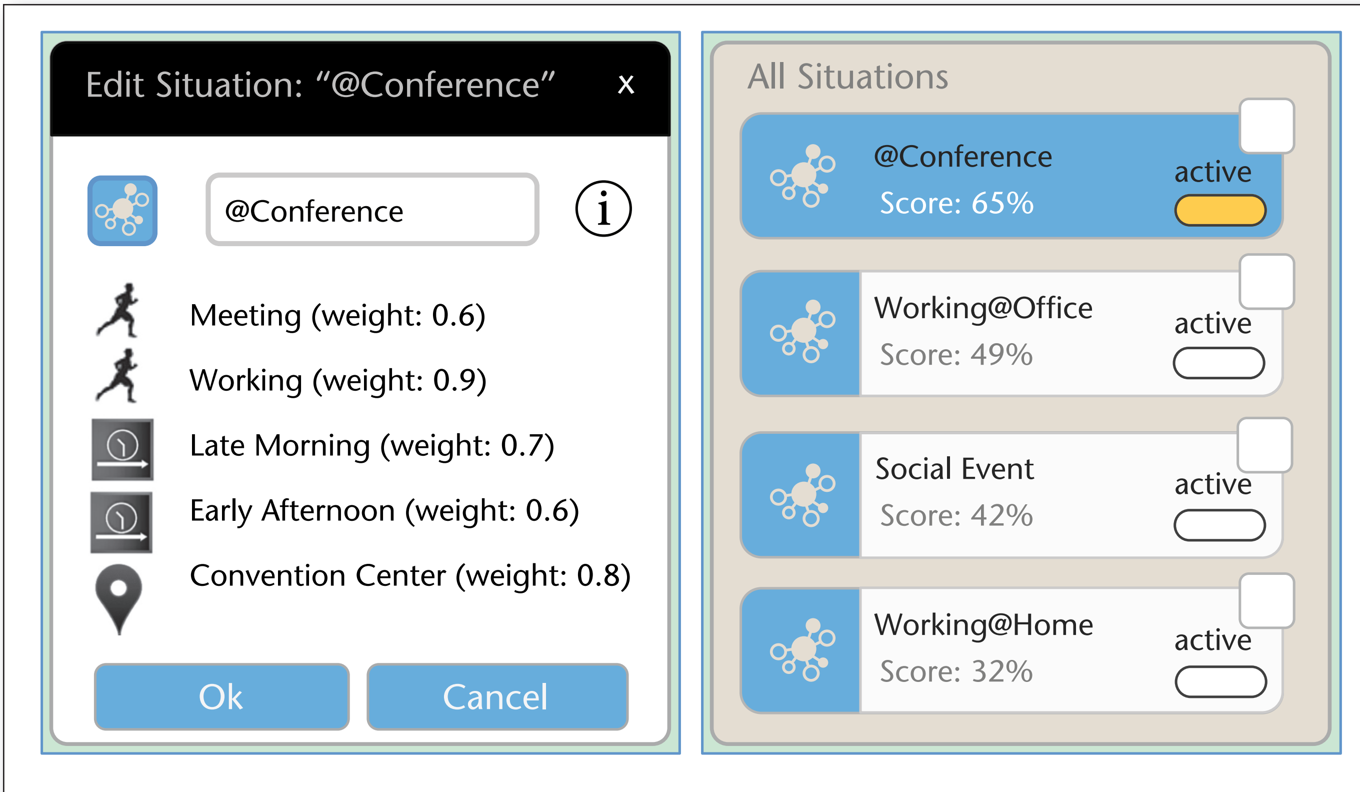


Figure 4. Situation Management and Suggestions.

The latter then transforms into a DRMO representation and sends it to the PIMO service for storage. The rule network is reinitialized to incorporate the new rule.

Related Work

Efforts that are directly comparable to the envisaged personalizable context-aware system include Nokia Situations[11] and the on{x}[12] and IFTTT[13] services. However, the comprehensiveness of the ontology framework presented in this article remains unrivaled. The adoption of a machine-processable data modeling standard also extends the applicability toward multiple personal information sources. Thus, the resulting context-aware features are not restricted to one device, but are applicable to all known nodes in the personal sensor network. As a result, personal situations are characterized, and personalized rules can be triggered, based on context events being streamed by multiple sources.

Few of the comparable efforts utilize ontologies as their data representation standard. Moon, Park, and Kim (2009) propose a context-aware system offering personalized services based on context information acquired from ambient sensors. Although this system utilizes an ontology for context-representation purposes, their model is severely limited in comparison to the DCON model. SemMF (Oldakowski and Bizer 2005) is a semantic-matching framework, which enables the calculation of semantic similarity between RDF graph representations. Given we provide models for the representation of context as graphs, SemMF was considered for the context-matching task, but was dismissed since, for example, it does not differ between node types. The graph matcher described in this article is a more sophisticated extension of SemMF, specialized for matching DCON context graphs.

Solutions implementing event processing require an underlying rule language. Systems like SECE (Beltran, Arabshian, and Schulzrinne 2012) propose languages that cannot be easily reused outside of their native framework. SECE enables user-defined rules for recommendations, based on open linked-data services. Since our user-defined rules are dependent on PIM data, the use of ontologies and RDF was a more natural choice. In comparison to XML-based languages like RuleML,[14] RDF makes it easier to achieve semantic interoperability (Decker et al. 2000). Nevertheless, the structure of DRMO rules is inspired by the event-condition-action (ECA) pattern described for SECE.

Complex event processing (CEP) and rule-based systems are commonly used to allow rules to define how perceived events are processed. The presented EP

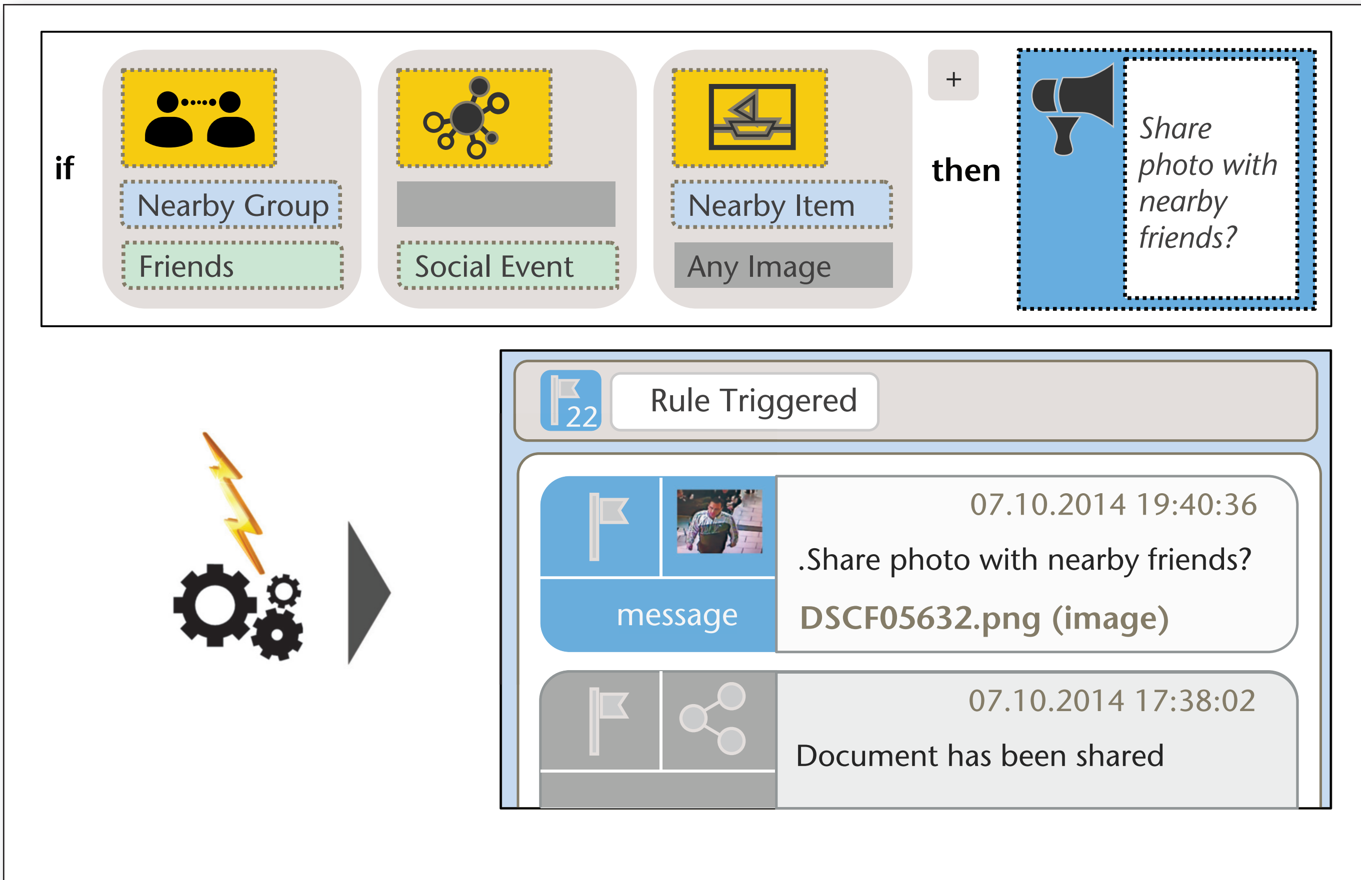


Figure 5. A User-Defined Rule (Top) and the Notification upon Triggering (Bottom).

exploits CEP properties to enable the use of temporal constraints in rules and to intercept data and events from multiple sources, while having a rule processing algorithm to filter and trigger relevant rules. In many rule-based systems, the Rete algorithm (Forgy 1982) is used to match facts with rule patterns. The Rete algorithm was not suitable for our objective, since it requires all network values to be explicit, that is, it does not support rules that include *any* values (for example, any person nearby). Besides allowing for implicit values in the tree, unlike RETE, the RN described in this paper also supports the temporal aspect. In contrast to Rete, the purpose of the RN is not directly to match rule conditions with event patterns. Rather, it filters candidate rules by forming subgraphs having perceived PIM items as their root node.

Conclusions and Future Work

As evidenced by mainstream shifts toward cloud-based services and cross-device/service identity management, there is an increasing need for centralized management of distributed personal sources and information contained within. In this article we explain how a semantic infrastructure for the elicitation, interpretation, and integration of personal activity context from a personal sensor network has been provided. An existing number of ontologies have been extended to cover context-related PIM information that is *about me* in addition to the more conventional information that is *for me*.[15] Based on the availability of these two kinds of PIM information, we describe a technique for the semiautomatic, adaptive recognition of personal situations and a technique for personalizable context-driven rules that extend the paradigm of email filters to the entire personal information sphere. The value of our contributions lies in their universal applicability — any context-aware (distributed) system can utilize the ontologies to integrate the information required to drive these features. As we demonstrate through the di.me userware, user interfaces can also be designed to enable people with various levels of technical proficiency to make the most of the ontology-based knowledge and configure their own situations and rules.

The techniques presented are subject to the following future improvements. An optimized technique needs to be devised to determine when the context matching should be performed, instead of executing the algorithm at a predefined frequency so as to ensure scalability. The matching accuracy can also be improved with the aim of reaching a precision that is suitable for deterministic situation recognition. To address the current method's cold-start problem, and the lack of initial training examples, data-mining techniques for the automatic discovery of repeated patterns in the live context will be investigated. Similarly, we will investigate the possibility to enable the identification of new rules, based on the availability of a users context history. The UI designed for the di.me userware can continue to serve as a test bed for this novel technology, although we are very interested in the prospect of its takeup by other RDF-enabled context-aware systems, or online services.

Acknowledgments

This work is supported by the European Commission under the 7th Framework Program FP7/2007-2013 (digital.me — ICT-257787).

Notes

1. w3.org/TR/rdf-primer.
2. semanticdesktop.org/ontologies/2011/10/05/dcon.
3. semanticdesktop.org/ontologies/2012/03/06/drmo.
4. nepomuk.semanticdesktop.org.
5. dime-project.eu.
6. semanticdesktop.org/ontologies/2007/01/19/nie.
7. w3.org/TR/rdf-sparql-query.
8. Original source code for both the di.me client and server is provided as open source at github.com/dime-project.
9. See vimeo.com/dimeproject/videos.
10. json.org.
11. pastillilabs.com/situations.
12. onx.ms.
13. ifttt.com.
14. ruleml.org.
15. See D. Karger, 2012, Personal (Information Management) Is Not (Personal Information) Management (groups.csail.mit.edu/haystack/blog/2012/02/17/personal-information-management-is-not-personal-inf)

References

Attard, J.; Scerri, S.; Rivera, I.; and Handschuh, S. 2013a. Ontology-Based Situation Recognition for Context-Aware Systems. Paper presented at the 9th International I-Semantics Conference, Graz Austria, 4–6 September..

Attard, J.; Scerri, S.; Rivera, I.; and Handschuh, S. 2013b. User Preference Learning for Context-Aware Situation Identification. Paper presented at the 5th International Workshop on Acquisition, Representation, and Reasoning with Contextualized Knowledge (ARCOE-LogIC), Corunna, Spain, 15 September.

Beltran, V.; Arabshian, K.; and Schulzrinne, H. 2012. Ontology-Based User-Defined Rules and Context-Aware Service Composition System. In *The Semantic Web: Proceedings of the 10th International Conference*. Lecture Notes in Computer Science volume 7031. Berlin: Springer-Verlag.

Debattista, J.; Scerri, S.; Rivera, I.; and Handschuh, S. 2013. Processing Ubiquitous Personal Event Streams to Provide User-Controlled Support. In *Web Information Systems Engineering: Proceedings of the 14th International Conference,* Part II. Lecture Notes in Computer Science 8181, 375–360. Berlin: Springer.

Decker, S.; Melnik, S.; van Harmelen, F.; Fensel, D.; Klein, M.; Broekstra, J.; Erdmann, M.; and Horrocks, I. 2000. The Semantic Web: The Roles of XML and RDF. *IEEE Internet Computing* 4(5): 63–73.

Dey, A. K. 2001. Understanding and Using Context. Technical Report, Future Computing Environments Group, Georgia Institute of Technology, Atlanta, GA.

Forgy, C. L. 1982. Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. *Artificial Intelligence* 19(1): 17–37.

Fuchs, S.; Rass, S.; Lamprecht, B.; and Kyamakya, K. 2008. A Model for Ontology-Based Scene Description for Context-Aware Driver Assistance Systems. Paper presented at the 1st International Conference on Ambient Media and Systems, Sarasota, FL, 11–14 November.

Moon, A.; Park, Y.; and Kim, S. 2009. Short paper: Situation-Awareness Model for Higher Order Network Knowledge Management Platform. In *Proceedings of the 2009 International Workshop on Semantic Sensor Networks*. Ceur Workshop Proceedings, volume 522. Aachen, Germany: RWTH Aachen University.

Oldakowski, R., and Bizer, C. 2005. SemMF: A Framework for Calculating Semantic Similarity of Objects Represented as RDF Graphs. Poster presented at the 4th International Semantic Web Conference (ISWC 2005), Galway, Ireland, 6–10 November.

Sauermann, L.; van Elst, L.; and Dengel, A. 2007. PIMO — A Framework for Representing Personal Information Models. Paper presented at the 3rd International Conference on Semantic Technologies, Graz, Austria, 5–7 September.

Scerri, S.; Cortis, K.; Rivera, I.; and Handschuh, S. 2012. Knowledge Discovery in Distributed Social Web Sharing Activities. In *Making Sense of Microposts (#MSM2012): Proceedings of the World Wide Web 2012 Workshop,* 26–33. Ceur Workshop Proceedings, volume 838. Aachen, Germany: RWTH Aachen University.

Scerri, S.; Schuller, A.; Rivera, I.; Attard, J.; Debattista, J.; Valla, M.; Hermann, F.; and Handschuh, S. 2013. Interacting with a Context-Aware Personal Information Sharing System. In *Human-Computer Interaction: Users and Contexts of Use: Proceedings of the 15th International Conference*. Lecture Notes in Computer Science, volume 8006. Berlin: Springer.

Schwarz, S. 2006. A Context Model for Personal Knowledge Management Applications. In *Modeling and Retrieval of Context, Second International Workshop, Revised Selected Papers,* Lecture Notes in Computer Science volume 3946, ed. T. R. Berghofer, S. Schulz, and D. B. Leake, 18–33. Berlin: Springer.

Sintek, M.; Handschuh, S.; Scerri, S.; and van Elst, L. 2009. Technologies for the Social Semantic Desktop. In *Reasoning Web: Semantic Technologies for Information Systems,* Lecture Notes in Computer Science, volume 5689, 222–254. Berlin: Springer.

Thiel, S.; Bourimi, M.; Giménez, R.; Scerri, S.; Schuller, A.; Valla, M.; Wrobel, S.; Frà C.; and Hermann, F. 2012. A Requirements-Driven Approach Towards Decentralized Social Networks. In *Future Information Technology, Applica-*

tion, and Service. FutureTech 2012, volume 1. Lecture Notes in Electrical Engineering, 164. Berlin: Springer.

Walzer, K.; Breddin, T.; and Groch, M. 2008. Relative Temporal Constraints in the RETE Algorithm for Complex Event Detection. In *Proceedings of the 2nd International Conference on Distributed Event-Based Systems* (DEBS), 147–155. New York: Association for Computing Machinery.

Yen, J. Y. 1971. Finding the K-Shortest Loopless Paths in a Network. *Management Science* 17(11): 712–716

Simon Scerri is a postdoctoral research fellow at Fraunhofer IAIS and an assistant lecturer at Universität Bonn, Germany. He received his Ph.D. from the National University of Ireland, Galway (2011), and has published in the areas of web 2.0, semantic web applications, ontologies, knowledge engineering, information extraction, text analytics, collaborative systems, human-computer interaction, social web, and context-aware systems. Other research interests include linked data, data visualisation, computational linguistics, pervasive systems, and recommender systems. Scerri led the research efforts reported in this article as a postdoc at the Digital Enterprise Research Institute (DERI), the world's largest semantic web research institute.

Jeremy Debattista is a Ph.D. student at the University of Bonn. Debattista has a keen interest in the semantic web, particularly linked open data, and ontology engineering. He holds a M.App.Sc. in computer engineering from the National University of Ireland, Galway. In the DERI team working on the contributions presented in this article, Debattista contributed a rule-management vocabulary and a rule processor to listen to events perceived by personal devices, in order to triggers customized rules in a scalable fashion. He is currently working on efforts to improve data quality in linked open data.

Judie Attard is a Ph.D. student at the University of Bonn. Attard's interests include context-aware systems, context recognition, knowledge representation, and linked open data. She holds a M.App.Sc. in computer engineering from the National University of Ireland, Galway. As part of the DERI team working in this area, Attard developed an algorithm for context matching that employs a function for semiautomatic situation recognition, and a method for the semiautomatic adaptation of the matching function. She is currently working in the area of linked open data.

Ismael Rivera is the lead engineer at Altocloud, a cloud-based analytics and communications platform. Rivera has more than 10 years experience in software research and development and has become a knowledgeable software engineer and web advocate with great experience in NoSQL, graph algorithms, interoperability, APIs, and scalable back-end solutions. Rivera holds a M.Sc. in computer engineering from Universidad Carlos III de Madrid and has previously worked as a research associate at the National University of Ireland, Galway (DERI), and Telefonica R&D (Madrid). In DERI, Rivera led the technical developments required to enable the contributions presented in this article.