Towards Unified and Native Enrichment in Event Processing Systems

Souleiman Hasan
Digital Enterprise Research Institute, National University of Ireland, Galway
souleiman.hasan@deri.org

Sean O'Riain
Digital Enterprise Research Institute, National University of Ireland, Galway
sean.oriain@deri.org

Edward Curry
Digital Enterprise Research Institute, National University of Ireland, Galway
ed.curry@deri.org

ABSTRACT
Events are encapsulated pieces of information that flow from one event agent to another. In order to process an event, additional information that is external to the event is often needed. This is achieved using a process called event enrichment. Current approaches to event enrichment are external to event processing engines and are handled by specialized agents. Within large-scale environments with high heterogeneity among events, the enrichment process may become difficult to maintain. This paper examines event enrichment in terms of information completeness and presents a unified model for event enrichment that takes place natively within the event processing engine. The paper describes the requirements of event enrichment and highlights its challenges such as finding enrichment sources, retrieval of information items, finding complementary information and its fusion with events. It then details an instantiation of the model using Semantic Web and Linked Data technologies. Enrichment is realised by dynamically guiding a spreading activation algorithm in a Linked Data graph. Multiple spreading activation strategies have been evaluated on a set of Wikipedia events and experimentation shows the viability of the approach.

Categories and Subject Descriptors
H.3.3. [**Information Storage and Retrieval**]: Information Search and Retrieval---*information filtering*.

General Terms
Algorithms, Experimentation, Languages, Theory.

Keywords
Information Completeness; Event Enrichment; Event Processing; Linked Data; Spreading Activation.

1. INTRODUCTION
Event-based technology is becoming more widely needed with the rise of new applications ranging from Smart Homes to Smart Cities and the Internet-of-Things [1]. Event-based systems enable a decoupled mode of interaction between participants which makes it suitable for large-scale and distributed environments such as sensor networks and mobile environment [11]. There are estimates that by the end of 2020 fifty billion devices will be connected to mobile networks [20] which would push event-based technology to its limits.

DEBS'13, June 29-July 3, 2013, Arlington, Texas, USA.

While the basic information item in an event-based system is an event, it is not uncommon that normal users require the system to handle information that is not encoded in the event. Such information typically comes from legacy databases or web data sources. This causes an information completeness problem for events to be sufficient for tasks such as subscription matching. One current solution to the information completeness issue is to develop external, static and dedicated event processing agents that retrieve information from legacy data sources and enrich the event before it is propagated for further processing. For example, an energy consumption event is generated by a smart electric heater containing the heater's serial number. An enricher retrieves information about the room and floor of the heater from a building management system database and adds it to the event which can then be considered when matching users' interests in high energy consumption events from that specific room or floor.

Future applications of event-based systems are large-scale applications such as the Internet-of-Things where the number of tasks that require information not included in events increases. In these environments the enrichment agents can quickly become difficult to develop and maintain. We argue that the problem lies in the approach taken in current event-based middleware where an event is considered as a closed world. For example, if a subscription tests a specific property that is not included in the event, then that is considered a negative match by default. No attempt is made to try and complement information in the event before judging on positive or negative matching.

The need to complement incomplete events has been recognized by the event processing community. Hinze et al. [15] states that "event enrichment calls for an understanding not only of the events but also for the external sources of information". Hohpe and Woolf [16] dedicates a set of patterns such as message translator, content enricher, and aggregator to address several problems that can be classified under event incompleteness. Teymourian et al. [25] investigates the improvement of expressiveness and flexibility of complex event processing systems via the usage of background knowledge about events and their relations to other concepts in the application domain.

Patterns by Hohpe and Woolf [16] reflect the current state-of-the-art and practice in the design of event processing networks where dedicated agents are assigned with well-defined tasks to overcome some incompleteness issues. For example, they propose the use of dedicated event enrichment agents to access a database and retrieve necessary information that is added to events before they

propagate to consumers. However, such agents are ad-hoc and tailored to the particular situations they are designed for. That contradicts with the event processing vision detailed by Etzion and Niblett [10] which calls for a unified and declarative way to process events. Enrichment agents are non-native to the paradigm, and as event processing systems scale out to large and highly heterogeneous environments, the maintenance of such enrichment agents becomes difficult.

Other related work focuses on the fusion of background knowledge with events using a query answering paradigm that spans events and background knowledge. However, such approaches make some assumptions that may not hold in many situations. For example, the work of Teymourian et al. [25] assumes that the background knowledge and events have the same data format and semantics, and that the knowledge base is accessible via a query service making the federation of the query feasible.

We think that in order to make advancement with respect to the event incompleteness problem, it is crucial to deal with the abstract characteristics of the problem and to integrate it into the event processing paradigm so it becomes a native component of event processing engines. Event enrichment can be done closer to the producer's side or closer to the consumer's side. In this paper we explore enrichment which is unified with consumption logic (matching) as consumers can better judge the content completeness of events with respect to their information needs. The contribution of this paper is threefold:

- A unified and native model of event enrichment is proposed along with its formalism.
- An instantiation of the model based on dereferenceable Linked Data and spreading activation is presented.
- An evaluation framework for event enrichment based on assessment of event completeness and enrichment precision is discussed.

The rest of this paper is organized as follows: Section 2 motivates the problem of information incompleteness in event processing systems while Section 3 outlines the dimensions of information incompleteness and the challenges for event enrichment. Section 4 explains the main concepts of the proposed model, its formalism and some potential implications. Section 5 details an instantiation of the proposed model based on Linked Data and spreading activation. Experiment and evaluation are explained in Section 6 and Section 7 summarizes related work. The paper concludes and discusses some potential future directions in Section 8.

2. MOTIVATIONAL SCENARIO

A sustainability officer is an employee who is responsible for assuring the company commitment to its social responsibility programs. For example, the sustainability officer would be interested in situations where energy consumption, and hence CO_2 emissions, of a particular department or building is excessive with regard to company or international standards [7,13].

In order for the sustainability officer to do the job, an event-based middleware is set up. Various energy-related sensors and real-time sources are instrumented so events flow into the middleware. Events in such a scenario are encapsulated with minimal information recording for instance a device name and the amount of energy used. An example attribute-value event describing the energy consumption of a heater is shown in Example 1.

Example 1: An Energy Consumption Event

{(type, "energy consumption"),
(device, "heater1"),
(consumption, "high")}

Non-technical users such as the sustainability officer tend to include higher level and business concepts and checks in their subscriptions to events. Examples of these are the *"room"* or the *"floor"* where the event was originated, or the *"business unit"* or *"project"* with which the device is associated. One example subscription is shown in Example 2.

Example 2: A Subscription for High Energy Consumption

{(type= "energy consumption")
and (floor= "second floor")
and (consumption="high")}

The events do not have information about the *"floor"* to answer the subscription in Example 2. Thus, in order to meet information requirement for this subscription, additional information sources in the enterprise such as data about the building would need to be exploited. Dedicated software agents need to be developed to enrich events with sufficient information. A large number of subscriptions may require dedicated enrichment agents. As a result, enrichment routines can become a burden to develop and maintain.

3. EVENT ENRICHMENT

The event-based interaction paradigm is based on the principle of decoupling the various parties which are involved in the interaction, namely event producers and consumers. The main advantage of decoupling the production and consumption of events is an increased scalability by "removing explicit dependencies between the interacting participants" [11]. The three common dimensions of decoupling between event producers and consumers are space, time and synchronization [11]. Thus, the only feasible way of interaction between participants becomes confined to exchanging events which carry payloads of information, making such a system event centric.

While the inherent feature of decoupling has its own virtues, it introduces other challenges in the event-based paradigm. An important one is the fact that event producers should have minimal assumptions on the information needs of event consumers. As a result, the content of an event payload becomes independent of consumers' needs. This can lead to information incompleteness on the consumers' side because there is not enough information in the event to process it.

If an event consumer ignores the concerns of information incompleteness and tries to conduct matching between its subscription and events, this may result in a high false positives or false negatives rate due to lack of relevant information in the events needed for the correct matching result. Thus, the consideration of the various dimensions of incompleteness becomes crucial to decrease the number of false positives/negatives in the matching process.

3.1 Dimensions of Incompleteness

Event incompleteness is a relative concept; it does not only depend on the event but also on the event consumption logic that is implemented by an event consumer. Event consumers may vary from simple User Interface agents to complex event processing

engines. In order to simplify the discussion on event consumers, we limit discussion to content-based matchers of single events using a subscription language to match events. These are common in the publish/subscribe paradigm and are usually implemented using a message-oriented middleware [9]. However, generalization to other types of event consumers is possible in light of the formalism we present in Section 4.1.

Given a particular event consumption logic, event incompleteness has a broad set of orthogonal dimensions. We have defined the dimensions based on an analysis of the enterprise integration patterns of Hohpe and Woolf [16]. This analysis produces general dimensions of incompleteness as follows:

1. **Event Format:** The event lacks the syntactical structure that can be processed by an event consumer. For example, let an event be as follows:

   "energy consumption of the heater in the second floor is high"

   This event is in plain text language syntax and thus cannot be processed by an event consumer which uses the subscription from Example 2. This is because the subscription expects attribute-value syntax not available in the event.

2. **Event Semantics:** The event lacks references to an interpretation scheme that can be used by an event consumer to understand what the event payload really means. For example, let an event be as follows:

   {"energy consumption", "second floor", "high"}

   This event is in tuple structure. It lacks the reference scheme according to which an event consumer which uses the subscription from Example 2 can interpret the actual indication of the term *"high"*.

3. **Complementary Background Knowledge:** The event lacks the amount of information required by an event consumer, and the complementary information resides in an enrichment source. For example, let an event be as follows:

   {(type, "energy consumption"), (device, "heater1"), (consumption, "high")}

   This event cannot be processed by an event consumer which uses the subscription from Example 2 because the event lacks any information about the "*floor"* in which the event occurred. This complementary information is likely to exist in a building management system database which has a fact such as *{("heater1", exists_in, "second floor")}*.

4. **Complementary Transformation:** The event lacks the amount of information required by the event consumer, and the complementary information can be obtained via a reasoning process over the event. For example, let an event be as follows:

   {(type, "energy consumption"), (device, "heater1"), (watt_hour, "1500")}

   Let the event consumer use the following subscription:

   {(type="energy consumption") and (device="heater1") and (kilowatt_hour= "1.5")}

   The event lacks the property "*kilowatt_hour"* and thus is incomplete with respect to the consumer. However, this information can be obtained by a calculation on the actual event itself using a reasoning rule such as: *kilowatt_hour= watt_hour/1000*.

5. **Temporal Segmentation:** A single event does not have the amount of information required by an event consumer, and the complementary information resides in other events which occurred previously or are going to occur in the future. For example, it is common to have three-phase electricity power feeds to buildings. Clamp-on power monitoring sensors are usually installed on every 1-phase cable entering the building. This results in three events arriving at a specified rate one after the other:

   {(type, "power consumption"), (consumer, "building"), (watt_phase1, "3000")}

   {(type, "power consumption"), (consumer, "building"), (watt_phase2, "2800")}

   {(type, "power consumption"), (consumer, "building"), (watt_phase3, "3200")}

   Let an event consumer use a subscription such as the following:

   {(type= "power consumption") and (consumer ="building") and (watt_all_phases="9000")}

   The consumer finds that all the events lack the knowledge about the three-phases power consumption. However, such information can be obtained by temporally aggregating three events from all the phases in order to get the overall power consumption that can be processed by the consumer.

3.2 Challenges for Event Enrichment

The term *event enrichment* is used in this paper to refer to any process that is done on events in order to overcome fully or partially an event incompleteness problem that spans one or more of the event incompleteness dimensions explained in Section 3.1.

For the sake of simplicity throughout the rest of this paper, we leave the temporal segmentation dimension to future work. Reasoning for complementary transformation over events is assumed to be done beforehand with the result stored in a knowledge base. That turns the complementary transformation dimension into the complementary background knowledge dimension. Given the final set of incompleteness dimensions, four fundamental challenges are recognized:

1. Determination of the Enrichment Source (ES)

The first challenge to face event enrichment is the decision on which enrichment source(s) to use. The challenge comes from the fact that event producers and consumers are decoupled and potentially have various perspectives of where complementary information for an event may exist. Determining the enrichment source may be statically stated by the event producer or consumer making this challenge easy to overcome. However, if sources are not known beforehand then a source discovery process is needed. Some possible enrichment sources include:

- **Wikis**: The Wikipedia online corpus for instance *"http://en.wikipedia.org/wiki/"* can be considered as a textual domain-agnostic enrichment source.
- **Relational Databases**: An example is a Building Management System (BMS) relational database described by the connection string *"Server=www.example.com\rdbms; Database=BMS-DB;"*
- **Linked Data** [2]: The DBpedia corpus for example can be addressed by its domain *"http://dbpedia.org/resource/"*.

2. Retrieval of Information Items from the enrichment Source

The access and retrieval mechanism poses a challenge to the enrichment process as it affects its ability to retrieve atomic information items from the enrichment source. Retrieval of information items can be quite challenging if network transfer has reliability issues or if the retrieval speed forms a bottleneck in the system. The exact retrieval mechanism will depend on the selected enrichment source. Some example retrieval mechanisms include:

- **Wikis**: A retrieval mechanism for Wikipedia is a search operation against its search API followed by an HTTP GET request to get a Wikipedia article as the information item.
- **Relational Databases**: A retrieval mechanism for a relational database is a SQL query against a query interface, with the retrieved rows as the information items.
- **Linked Data**: A retrieval mechanism for the DBpedia corpus for instance is looking up (dereferencing) URIs [3] of the resources, with the RDF [17] graphs of these URIs being the information items retrieved.

3. Finding Complementary Information for an Event in the Enrichment Source

The ability of the enrichment process to retrieve atomic information items from an enrichment source is faced with the challenge to determine which of the information items can complement an event and should be retrieved. Several ways to find complementary information are:

- **Wikis**: To find complementary information in the Wikipedia corpus, articles related to a term in the event can be searched and then links from these articles are followed one step deep and ultimately all the resulting articles are retrieved.
- **Relational Databases**: To find complementary information in a relational database, one can formulate a SQL query with some specific primary keys coming from the event.
- **Linked Data**: To find complementary information in the DBpedia corpus, a spreading activation [5] of URIs can be conducted starting from seed URIs and following the links in the data cloud with some termination conditions.

4. Fusion of Complementary Information with the Event

The final challenge is integrating and fusing the complementary information items with the event. This challenge stems from the several formats and semantics of data models that are used by the enrichment source and by the event. Multiple instances of fusion are presented in Example 3.

Example 3: Fusion Methods

Let an event be the attribute-value map

{(type, "energy consumption"),
(device, "heater1"),
(consumption, "high")}.

Let the enrichment source be a relational database with the relations *<heater, room>* and *<room, floor>* containing respectively the rows:

<heater1, room123>
<room123, second floor>

One possible fusion method is to add two attribute-value pairs to the event so it becomes:

{(type, "energy consumption"),
(device, "heater1"),
(consumption, "high"),
(room, "room123"),
(floor, "second floor")}

Another fusion method is to add one attribute value pair which contains the location to the event so it becomes:

{(type, "energy consumption"),
(device, "heater1"),
(consumption, "high"),
(location, "room123, second floor")}.

4. UNIFIED AND NATIVE ENRICHMENT MODEL

The key pillar of the proposed model is the recognition of enrichment as a core task of event processing engines. In addition, the enrichment behaviour of an event processing engine can be dictated to the engine using a uniform and declarative mechanism. The cornerstone of the model is the concept of an *enrichment element* that is a declarative specification for the engine to enrich events with complementary information items.

The model proposes that the enrichment element is described using a set of declarative language constructs similar to the ones used currently for matching purposes. In order to systematically characterize the language constructs needed for the enrichment element, we propose four language clauses that are mapped to the four enrichment challenges as follows:

1. **ENRICH FROM** clause which allows the engine to determine the enrichment source(s) explicitly.
2. **RETRIEVE BY** clause which allows the engine to determine the retrieval mechanism for atomic information items.
3. **FIND BY** clause which specifies the approach which would dictate the retrieval of a subset of information items from the enrichment source(s) that can complement the event.
4. **FUSE BY** clause which defines the fusion approach to integrate retrieved complementary information with the incomplete event.

The next issue is to determine who is responsible for defining the enrichment elements. Reviewing the clauses of an enrichment element shows that some of these can be specified by the event producer and/or the event consumer. Specifically, the enrichment source and retrieval mechanism can be defined by the producer who may know them at the time of producing the event.

The model proposes that all the enrichment clauses are described by the event consumer. That is because the consumer has a better understanding of the information need at the consumption side. This is also aligned with scenarios where the event producer has little assumptions on information needs of the consumers and where decoupling is the norm. This adds to our previous work on loose semantic coupling and approximate matching in event processing systems [14].

Consequently, the model suggests that the enrichment element co-exists with the matching element which forms subscriptions in current systems. The resulting subscription which contains enrichment and matching elements is called a *unified subscription*.

By having unified subscriptions, enrichment can be brought to the core of the event processing engine. It operates based on the enrichment element and uses the matching element to conduct an enrichment process over the incoming incomplete events and enrichment source(s) to produce enriched events that can then be matched against the matching element. It is called a *native enricher* in this model. While implementation details of the enricher are left to particular instantiations, the proposed model suggests that the enricher not only uses the enrichment clauses to operate, but also the matching element to guide the enrichment process. Figure 1 depicts the proposed enrichment model.

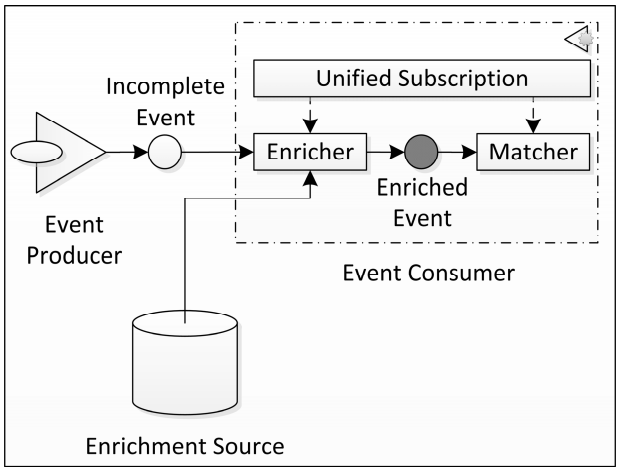

Figure 1. Unified and native enrichment model

Example 4 presents a simple instantiation of the enrichment clauses and the native enricher.

Example 4: Instantiation for Plain Text Events

Events are represented as bags of words. Let an event be as follows:

{"energy", "consumption", "heater1", "high"}

Let the matching element of subscriptions be represented as a bag of words as follows:

{"energy", "second", "floor", "high"}

The semantics of the event matching is that all words in the matching element need to be found in the event for a positive match, otherwise it is a negative match.

We assume that the enrichment source for the system is an enterprise wiki of text articles called *enterprise-wiki*. The wiki contains an article titled *"second floor"* which contains the term *"heater1"*. The wiki can be searched via a term search API which returns a list of articles containing the term. The API is accessible via a RESTful web service. When the API is searched with the term *"heater1"* the article titled *"second floor"* is returned.

The enrichment clauses are defined as the following:

- ENRICH FROM specifies the name of the wiki.
- RETRIEVE BY specifies the access protocol.
- FIND BY specifies the search mechanism.
- FUSE BY defines the fusion method to extract words from the retrieved article's title or from the article content, and if to add the new words to the event or to replace its words by the new found words.

A full example unified subscription becomes:

```
ENRICH FROM 'enterprise-wiki'.
RETRIEVE BY 'HTTP GET'.
FIND BY 'term search'.
FUSE BY 'title terms' 'add'.
{"energy", "second", "floor",
"high"}.
```

When the event *{"energy", "consumption", "heater1", "high"}* arrives to the system, the native enricher uses the words in the event to search the *enterprise-wiki* using each word at a time. Assuming that the enricher firstly retrieves the article titled *"second floor"*, it extracts the single words from the article's title and adds them to the event. The enriched event becomes as follows:

{"energy", "consumption", "heater1", "high", "second", "floor"}

Other articles are retrieved and fused similarly. The matching element is then evaluated against the enriched event. As a result, the matcher finds a positive match.

4.1 Formalism

The model is represented using the quadruple $(\mathcal{L}, E, ES, U)$, where:

- $\mathcal{L}$ is the unified subscription language.
- E is the set of events.
- ES is a set of information items that form the source of enrichment.
- U is the universe which contains all the possible information items.

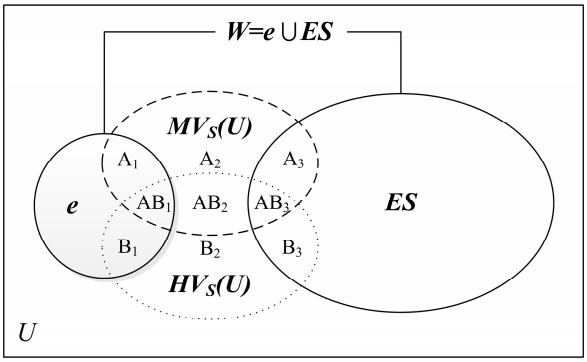

Figure 2. The universe *U*, the event *e*, the enrichment source *ES*, the world *W*, the enrichment view HV_S, and a matching view MV_S

The model has two underlying assumptions concerning valid information items and common information items. Valid information items are those which are considered to be true facts. Given an event $e \in E$, we assume that the only valid information items are those which exist in the event e or in the enrichment source ES. In other words, this assumption is equivalent to a Closed World Assumption (CWA) where the world $W = e \cup ES$. In

fact, it is worth mentioning that traditional event processing systems usually make a closed world assumption at the matching stage, where the world $W=e$. The principal assumption that the world is limited to the event causes the incorrect decisions of the matcher in judging many positive and negative matches.

The other assumption concerns common information items between events and the enrichment source. We assume that there is no intersection between the content of e and ES, i.e. $e \cap ES = \Phi$. The purpose of this assumption is to simplify the description of the model. However, in reality the event may have been published with some information items that also exist in the enrichment source. Nevertheless, the model is easily extended to the case where $e \cap ES \neq \Phi$. When conducting enrichment in practice, the information items in ES which are already in e can simply be discarded to turn the assumption into a valid assumption. Figure 2 illustrates the various concepts of the model.

Let S be a subscription in $\mathcal{L}$, S is a pair (H_S, M_S), where:

- H_S is the enrichment clauses element of S.
- M_S is the matching predicates element of S.

The model is described through the following definitions.

Definition 1: Boolean Matching Element

Let S be a unified subscription and I a set of information items:

$$M_S \text{ is a Boolean matching element} \quad (1)$$
$$\Leftrightarrow M_S(I) \in \{True, False\}$$

Definition 2: Approximate Matching Element

Let S be a unified subscription and I a set of information items:

$$M_S \text{ is an approximate matching element} \quad (2)$$
$$\Leftrightarrow M_S(I) \in \Re$$

Definition 3: Unknown Matching Result

Let S be a unified subscription and I a set of information items:

$$M_S(I) = Unknown \quad (3)$$
$$\Leftrightarrow (M_S \text{ is a Boolean matching element}$$
$$\wedge M_S(I) \notin \{True, False\})$$
$$\vee (M_S \text{ is an approximate matching element}$$
$$\wedge M_S(I) \notin \Re)$$

Definition 4: Matching View

Let S be a unified subscription and I a set of information items:

$$MV_S \text{ is a matching view of } S \text{ on } I \quad (4)$$
$$\Leftrightarrow M_S(MV_S(I)) \neq Unknown$$

Definition 5: Enrichment View

Let S be a unified subscription and I a set of information items:

$$HV_S \text{ is an enrichment view of } S \text{ on } I \quad (5)$$
$$\Leftrightarrow HV_S(I) = \{ii : ii \in I, ii \text{ is retrieved during enrichment}\}$$

Definition 6: Complete Event

Let S be a unified subscription and e an event from E:

$$e \text{ is complete with respect to } M_S \Leftrightarrow \quad (6)$$
$$\exists MV_S \text{ where } M_S(MV_S(e)) = M_S(MV_S(W))$$

Definition 7: Enriched Event

Let S be a unified subscription, e an event from E, ES the enrichment source, HV_S the enrichment view of the H_S element of S, $\oplus$ the FUSE BY operator of H_S:

$$ee \text{ is the enriched event of event } e \text{ according to } H_S \quad (7)$$
$$\Leftrightarrow ee = e \oplus HV_S(U)$$

Definition 8: Valid Enrichment

Let S be a unified subscription, e an event from E, ES the enrichment source, HV_S the enrichment view of the H_S element of the unified subscription S:

$$HV_S(U) \text{ is valid} \Leftrightarrow HV_S(U) \setminus HV_S(W) = \Phi \quad (8)$$

Definition 9: Successful Enrichment

Let S be a unified subscription, e an event from E, ES the enrichment source, HV_S the enrichment view of the H_S element of S, $\oplus$ the FUSE BY operator of H_S:

$$HV_S(U) \text{ is successful} \Leftrightarrow HV_S(U) \text{ is valid} \wedge \quad (9)$$
$$e \oplus HV_S(U) \text{ is complete with respect to } M_S$$

Definition 10: Minimal Successfully Enriched Event

Let e be and event from E and ES be the enrichment source. Let S_1, $S_2 \ldots S_n$ be a set of unified subscriptions in $\mathcal{L}$ where the matching element of all of them is the same M_S, while they vary in the enrichment elements being H_{S1}, $H_{S2} \ldots H_{Sn}$ respectively. Let HV_{S1}, $HV_{S2} \ldots HV_{Sn}$ be the set of enrichment views corresponding to the subscriptions. Let ee_1, $ee_2 \ldots ee_n$ be the enriched events of e according to the enrichment views respectively:

$$ee_k \text{ is a minimal successfully enriched event} \Leftrightarrow \quad (10)$$
$$ee_k \setminus \{ii : ii \in ee_k\} \text{ is not complete with respect to } M_S$$

An ideal event enrichment process would always turn events into minimal successfully enriched events. Ideally the areas in Figure 2 of $MV_S(W)$ and $HV_S(W)$ would be identical for at least one MV_S. Besides, the enrichment view would be valid, i.e. $HV_S(W) = HV_S(U)$. Thus, the areas A_1, B_1, A_2, B_2, AB_2, A_3, and B_3 become all empty. The definition above can be interpreted as a hard constraint, meaning that an enrichment process is considered successful for an event only if it produces a minimal successfully enriched event. This interpretation is suitable in many cases such as when the matching element M_S is a Boolean matching element.

However, there are cases where the event processing system may accept approximation. One example is when the matching element M_S is an approximate matching element. In such cases, it is suitable to adapt definition 10 to a softer interpretation, leading to Definitions 11 and 12.

Definition 11: Cost of Transformation into a Minimal Successfully Enriched Event

Let e be an event from E and ES the enrichment source ES, let S_1, $S_2 \ldots S_n$ be a set of all possible subscriptions in $\mathcal{L}$ where the matching element of all of them is the same M_S, while they vary in the enrichment elements being H_{S1}, $H_{S2} \ldots H_{Sn}$ respectively. Let ee_{m1}, $ee_{m2} \ldots ee_{mk}$ be the set of minimal successfully enriched

events of e according to the various enrichment clauses elements H_{S1}, H_{S2}… H_{Sn}. Let S be a subscription with the enrichment element H_S. Let ee be the enriched event of e according to Hs. We define the cost function $MSECost$ as follows:

$$MSECost : W \times W \rightarrow \Re^{+} \cup \{0\} \quad (11)$$

$$MSECost\ (ee, ee_{mi})\ is\ the\ min\ cost\ to\ turn\ ee\ into\ ee_{mi} \quad (12)$$

$$MSECost\ (ee_{mi}, ee_{mi}) = 0 \quad (13)$$

Definition 12: Approximately Minimal Successfully Enriched Event

Let ee be a successfully enriched event and ee_{mi} any minimal successfully enriched event:

$$ee\ is\ an\ approximately\ minimal\ successfully\ enriched\ event \Leftrightarrow Min_{ee_{mi}}(MSECost\ (ee, ee_{mi})) > 0 \quad (14)$$

4.2 Implications

This section discusses three of the potential implications of the proposed model:

- **Sharing and Re-usability of Enrichment Elements**: This stems from the core concept of recognizing enrichment routines as separate and modular declarative language elements. In deployments where a large number of producers and consumers exist, it is possible that only a small set of consumers would have the knowledge and expertise to provide well defined enrichment elements along with matching elements through unified subscriptions. Other consumers will keep writing classical matching subscriptions without specifying enrichment logic. This forms an opportunity for the event processing engine to enrich events according to the provided enrichment routines by expert users and forward the enriched events to normal users who would get more complete events.
- **Distribution of Enrichment**: When the event processing system is distributed into a set of brokers, there is an opportunity to distribute enrichment elements on the nodes to achieve an optimal overall completeness. With a suitable algebra for enrichment elements, coverage and ordering relationships can be defined for enrichment elements to avoid redundant enrichment and to account for optimized distributed enrichment plans.
- **Approximation in Event Processing Engines**: Building on the case when enrichment is done automatically by native enrichers may introduce some approximately complete events rather than fully complete events. Matching over partially complete events would need to account for the still missing information. This provides a good motivation for approximate matching in event processing systems which was investigated previously by the authors [14] based on the need for loose semantic coupling in heterogeneous systems.

These implications and others are subject to further investigation in the future.

5. A LINKED DATA INSTANTATION OF THE EVENT ENRICHMENT MODEL

This section details the implementation of the proposed model (refer to Figure 2): the event model, the enrichment source model, the matching element of subscriptions and the enrichment element along with a native enricher. The instantiation is designed for Linked Data events. Linked Data along with its core RDF graph model can be seen as a generic model for events, making the concepts applied in this instantiation also applicable in other implementations. A large amount of openly accessible Linked Data has been published on the web in the recent years making it easier to experiment with Linked Data events to study the enrichment model. Linked Data has also been used as a mechanism to link contextual data within different domains including finance, life sciences, public sector and energy [8].

5.1 Event Model

Events are instantiated as Linked Data events. Thus, an overview of Linked Data is given before proceeding.

Linked Data

Emerging from research into the Semantic Web, Linked Data proposes an approach for information interoperability based on the creation of a global information space. Linked Data leverages the existing open protocols and standards of the World Wide Web (WWW) architecture for sharing structured data on the web. The overall objective of Linked Data is to provide flexible data publishing and consumption. Berners-Lee [3] summarizes Linked Data in four principles:

1. Using URIs as names for things.
2. Using HTTP URIs so that people can look up those names.
3. When someone looks up a URI, providing useful information using standards such as RDF [17].
4. Including links to other URIs so that people can discover more things.

Event Model

An event is instantiated as a labelled directed graph. The resource description framework (RDF) is used to represent information about events using statements or triples. A statement consist of a *(subject, property, object)* triple. Subjects are references to information resources and are represented as URIs. Objects may be URIs or literal values. Properties come from various vocabularies (the RDF name of ontologies) and are represented as URIs of terms in these vocabularies. One subject may have multiple statements with the same property and different objects.

The resulting event can be represented as follows: Let E be the set of events conforming to the event model, P the set of properties, $URIs$ the set of all URIs and Lit the set of all Literals such as strings and numbers, then an event can be seen as a finite set of triples as follows:

$$e \in E \Leftrightarrow e = \{(s, p, v) : (s, p, v) \in URIs \times P \times (URIs \cup Lit)\} \quad (15)$$

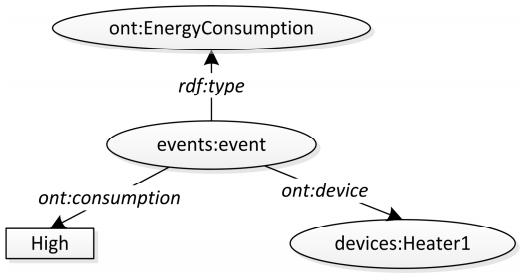

Figure. 3. An example event

A URI can be written using prefixes for clarity. For example the URI *http://www.example.com#event* can be written as *example:event* with the prefix *example* representing the part *http://www.example.com*. Figure 3 illustrates an example event where *ont* represents a prefix for the vocabulary of terms in the energy domain, *devices* a prefix for instances of devices in the environment, and *events* a prefix for all event instances.

5.2 Enrichment Source Model

The enrichment source is instantiated as a labelled directed graph. RDF is used to represent enrichment information. The enrichment source is a set of triples *(subject, property, object)* following the Linked Data principles. Let *ES* be the enrichment source, *P* the set of properties, *URIs* the set of all URIs and *Lit* the set of all Literals such as strings and numbers then:

$$ES = \{(s, p, v) : (s, p, v) \in URIs \times P \times (URIs \cup Lit)\} \quad (16)$$

Figure 4 illustrates an example enrichment source where *building* is a prefix for instances such as rooms and floors.

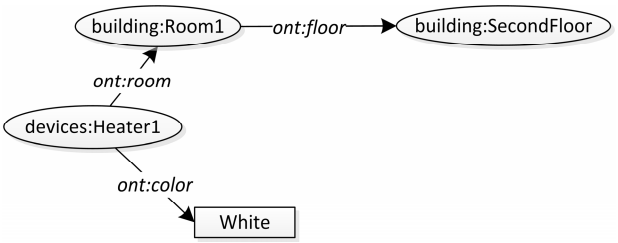

Figure. 4. An example enrichment source

The enrichment source is assumed to be accessible by dereferencing URIs associated with it. Dereferencing a URI means sending an HTTP request to its host, specifying the content type to be returned such as RDF, and finally receiving the HTTP response. The validity of a triple as required by Definition 8 is judged by its existence in the event or in the enrichment source.

5.3 Matching Element Model

The instantiation of the matching element of a subscription is a simplified version of the SPARQL patterns [23] which can contain basic graph patterns with variables. The matching element uses property paths in the place of properties to describe a regular expression of properties, or a path. The matching element is a Boolean matching element as defined in Definition 1. A matching view as defined in Definition 4 is the set of all triples that forms a solution to the graph pattern. Example 5 presents an example matching element.

Example 5: A Matching Element

The following matching element matches any event of type energy consumption whose URI has a path to the second floor URI within three nodes:

```
?event rdf:type ont:EnergyConsumption.
?event (?p){3} building:SecondFloor.
```

5.4 Enrichment Element Model

The instantiation of the enrichment element of a subscription is as follows:

- ENRICH FROM specifies the domain URI of the enrichment source.
- RETRIEVE BY specifies dereferencibility as the method for retrieval, notated as DEREF.
- FIND BY specifies how to explore the enrichment source to find complementary information. We propose a spreading activation strategy to be used by the enricher as explained in Section 5.5. The enrichment view defined in Definition 5 is the set of all triples whose subjects are activated during the spreading activation.
- FUSE BY realizes the $\oplus$ operator of the model presented in Definition 7. The RDF UNION is a suitable instantiation.

Example 6 presents a unified subscription that enriches from an enterprise Linked Data cloud, retrieves by dereferencibility, finds via a spreading activation strategy called *UniformWeightsAllAdjacent* and fuses via union. It aims at matching any event of type energy consumption whose URI has a three-links path to the *"second floor."*

Example 6: A Unified Subscription

```
ENRICH FROM <www.myenterprise.org>
RETRIEVE BY 'DEREF'
FIND BY 'Spreading Activation'
        'UniformWeightsAllAdjacent'
FUSE BY 'UNION'
{?event rdf:type ont:EnergyConsumption.
?event (?p){3} building:SecondFloor.}
```

The minimality of enriched events as defined in Definition 8 is realized by removal of triples from an enriched event. Finally, the approximation between an enriched event and a minimal successfully enriched event defined by the function *MSECost* in relations (11), (12) and (13) is realized by the cardinality of the relative complement operation '\' on sets of triples. Thus, the cost to turn an enriched event *ee* into a minimal successfully enriched event ee_m is composed of two costs:

- The cost to include all the successful enrichment triples in ee_m into *ee*. That is equivalent to $|ee_m \setminus ee|$.
- The cost to remove all unnecessary enrichment triples from *ee*. That is equivalent to $|ee \setminus ee_m|$.

The first point measures the completeness while the second measures the precision. These two measures and their combination form the basis for evaluation as shown in Section 6.

5.5 Native Enricher

The enrichment model is realized through a spreading activation algorithm [5]. Spreading activation (SA) originated in cognitive psychology as a network processing model for a supposed model of human memory. Applications of SA can be found in Artificial Intelligence, Cognitive Science, Databases, Information Retrieval, etc. The pure spreading activation model incorporates a processing technique for a generic graph data structure such as the RDF graphs. It is based on the idea of marking some nodes as active and then spreading the activation into other nodes iteratively. The way that spreading takes place and the semantics of the active nodes depends on the application. The processing is defined by a sequence of iterations that continue until a termination condition is activated. Each iteration consists of one or more pulses and a termination check [6].

Each pulse of the spreading activation consists of three stages: pre-adjustment, spreading and post-adjustment [6]. The spreading phase consists of a number of activation waves where each node

calculates activation inputs transferred to it from its neighbors, which can be done using the formula:

$$I_j = \sum_i O_i w_{ij} \quad (17)$$

Where I_j is the total input to node j, O_i is output of neighbor i and w_{ij} is a weight associated with the edge from node i to node j. When a node computes its total input I_j it calculates its output O_j as a function of I_j:

$$O_j = f(I_j) \quad (18)$$

The function can be simply a threshold function which decides if the node j is activated or not. The output of the node is in turn sent to neighboring nodes in the next pulse and so on. Activation spreads from the initially activated nodes to further nodes in the network. Pure SA may fall in a deadlock and run forever unless controlled. Constraints can be enforced in the pre-adjustment stage. Four sorts of constraints can be recognized [6]:

- **Distance Constraint**: The SA should decay as it reaches nodes far from the initially activated nodes.
- **Fan-out Constraint**: The SA should cease at nodes with very high connectivity.
- **Path Constraint**: The SA should be selective in the path it spreads in making use for example of the semantics of labels on the edges.
- **Activation Constraint**: Using various thresholds can affect the behavior of the SA.

Spreading activation within the enricher along with the Linked Data instantiation of the event and the enrichment source models can realize the enrichment model. Spreading Activation can be used to explore the enrichment source and retrieve a set of triples to be fused in the event. In order to guide SA in the enrichment source, we propose a path constraint to favor some links over the others. The path constraint that we propose is based on ranking the links connected to a spread node based on their semantic relatedness with terms in the matching element and then just follow the top two or three links. The semantic relatedness used in the experiment is a WordNet-based measure called the Path measure. More on WordNet and semantic measures can be found in [4].

6. EXPERIMENT

In order to demonstrate how to evaluate a particular instantiation of the proposed enrichment model, an experiment has been conducted in association with the Linked Data instantiation of the enrichment model described in Section 5. The experiment has been done on real-world data, namely events extracted from Wikipedia, and uses the DBpedia dataset as an enrichment source.

A set of event subscriptions is generated where each subscription conforms to the unified language instantiation in Section 5. Matching elements use the property path variables to express a path of predicates between an event and a value. The minimal successfully enriched events for each subscription are calculated in order to form a baseline to measure the effectiveness of enrichment.

The purpose of the experiment is to compare three strategies of event enrichment which vary the mechanism used by the enricher to find complementary information items in the enrichment source. The variation is expressed by different parameters to the spreading activation algorithm in the FIND BY clause of the subscription enrichment element. The three strategies are:

- **UniformWeightsAllAdjacent**: A spreading activation strategy where activation from one node spreads equally to all adjacent nodes.
- **UniformWeightsRandomAdjacent**: A spreading activation strategy where activation from one node spreads equally to a random set of adjacent nodes.
- **DifferentWeightsSemRel**: A spreading activation strategy where activation from one node spreads unequally to a set of adjacent nodes based on the semantic relatedness of the adjacency edges and the terms in the matching element of the subscription.

The key difference between the evaluated strategies is that the former two guide enrichment independently from the matching element of the subscription while that last strategy actually benefits from the fact that enrichment logic and matching logic exist together in the unified subscription. The last strategy guides the enrichment algorithm according to semantic relatedness between the terms in the matching element and terms on the links in the enrichment source. Thus, the last strategy, if confirmed to perform better than the other two, proves that a unified subscription with enrichment and matching together unified and native to the event processing engine is a beneficial approach to event enrichment.

It is worth mentioning that the objective is not to investigate the best approach for enrichment in the particular Linked Data instantiation but rather to demonstrate how evaluation can be conducted. Investigating the best performing enrichment strategies for Linked Data events is indeed an important future direction.

6.1 Event Set and Enrichment Source

The event set used in this experiment is a structured representation of events in Wikipedia[1]. DBpedia [2] is a community project to extract structured information from Wikipedia. DBpedia is one of the efforts under the Linked Open Data initiative which targets the publication of structured data on the web according to the Linked Data principles [3]. The data model used to represent DBPedia data is RDF. The event set used for this experiment is a subset of the current version the English DBpedia[2]. It contains all resources of type `dbpedia-owl:Event`. Each event is a triple of the form `<eventURI, rdf:type, dbpedia-owl:Event>`.

The size of the event set is around 24,000 events. Examples of various event types found in the event set are: *"Football Match"*, *"Race"*, *"Music Festival"*, *"Space Mission"*, *"Election"*, *"10th-century BC Conflicts"*, *"Academic Conferences"*, *"Aviation Accidents And Incidents In 2001"*, etc.

The enrichment source is the set of all triples that are stored on the online DBpedia and can be retrieved by looking up DBpedia resource URIs. Events are played sequentially and pushed to the native enricher which searches the enrichment source for complementary information, fuses it with the events and forwards them to the event matcher.

[1] http://www.wikipedia.org/

[2] http://downloads.dbpedia.org/3.8/en/. Last modified on the 1st of August 2012. Accessed on 25th of February 2013.

6.2 Unified Subscription Set

The subscription set consists of four subscriptions. The matching element of subscriptions was automatically generated using the following method:

1. We start by the seed URI of the 1966 FIFA World Cup Final *http://dbpedia.org/resource/1966_FIFA_World_Cup_Final* and retrieve resources linked to it to build a path-shaped graph of 4-triples long. Figure 5 shows the resulting full path-shaped graph.
2. For the first subscription, we pick the first triple and consider it as the matching element.
3. For the second subscription, we pick the first two triples and construct a matching element as defined in Section 5 using the two terminal URIs of the two-triples long path as subject and object and a property path variable in between.
4. We repeat the last step for subscriptions 3 and 4.

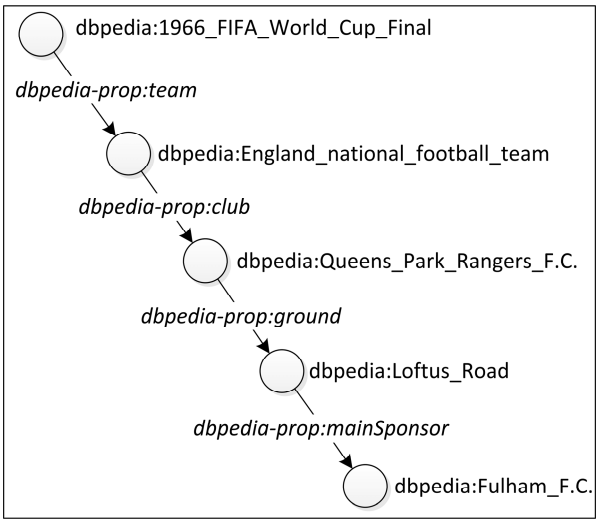

Figure 5. The base path-shaped graph used to generate the matching elements of the subscriptions

The resulting matching elements are shown in Table 1. Subscriptions range in complexity with respect to the length of the property path in their matching elements with the most complex subscription being the one with the longest property path. To form the final unified subscriptions, each matching element is concatenated with an enrichment element which consists of the four clauses ENRICH FROM, RETREIVE BY, FIND BY and FUSE BY. The evaluated three strategies are passed as parameters to the FIND BY operator.

Table 1. Matching elements of the unified subscription set

ID	Matching Element
1	?event rdf:type dbpedia-owl:Event. ?event (?p){1} dbpedia:England_national_football_team.
2	?event rdf:type dbpedia-owl:Event. ?event (?p){2} dbpedia:Queens_Park_Rangers_F.C..
3	?event rdf:type dbpedia-owl:Event. ?event (?p){3} dbpedia: Loftus_Road.
4	?event rdf:type dbpedia-owl:Event. ?event (?p){4} dbpedia: Fulham_F.C..

6.3 Minimal Successfully Enriched Events Construction

In order to generate the event data that can be considered a minimal successfully enriched event with respect to each subscription, the following methodology has been used:

For each matching element of a subscription, a SPARQL [23] query is formed and executed against the DBpedia online SPARQL API. The query uses optional joins and filters to match all the events in DBpedia with all possible cases of their associated values or predicates. Example 7 shows the generated query for subscription 3.

Example 7: A Generated SPARQL Query

```
SELECT DISTINCT ?event ?team ?club
WHERE
  {?event a dbpedia-owl:Event .
    OPTIONAL
      {?event  dbpedia-prop:team  ?team .
        FILTER (!isLiteral(?team))
        OPTIONAL
          {?team dbpedia-prop:team  ?club .
            FILTER (!isLiteral(?club) )}}}
```

When the SPARQL queries are executed, the result contains all the events with possible values for the specified path. These events with their associated data are minimally complete as a matching decision can be made upon them for the specified subscription.

6.4 Evaluation Criteria

Given a subscription S and an event e. Let ee be the enriched event of e according to S. Let e_m be the closest minimal successfully enriched event to ee according to relations (11), (12) and (13) and their instantiation in Section 5.4. We define the following metrics for evaluating the effectiveness of the enrichment approach:

$$Completeness = \frac{|ee \cap e_m|}{|e_m|} \tag{19}$$

$$Precision = \frac{|ee \cap e_m|}{|ee|} \tag{20}$$

$$F_5\ Score = \frac{(1+5^2)*Precision*Completeness}{5^2*Precision + Completeness} \tag{21}$$

The intersection is realized via an intersection between the set of triples that form each graph ee and e_m. The cardinality of events here is realized through the number of triples in the set that corresponds to each graph. The *F-Score* is a composite measure which is useful to summarize the effectiveness of an enrichment approach in one number for a subscription rather than two numbers. We argue that completeness and precision are not equally important. To evaluate an enrichment approach based on information completeness, the completeness measure should be given more weight. That is why the $F_5 Score$ is chosen in this evaluation. Depending on the application domain and constraints, other weighting may be considered.

6.5 Results

Figure 6 illustrates the combined F_5Score achieved by each enrichment approach for each subscription and averaged on events. The chart shows the superiority of the semantic relatedness-based approach and confirms the hypothesis that an enrichment approach which makes benefit from the enrichment logic unified with the matching logic is more effective than enrichment that is only based on enrichment logic.

There is also a trend showing that the enrichment effectiveness decreases for more complex subscriptions. The decreasing effectiveness is due to the fact that a longer property path requires more spreading to reach relevant triples while spreading may fade before that. From an empirical perspective, this raises the issue that the evaluation of an enrichment approach shall factor in the effect of the several types and complexities of subscriptions in the results. It is noticeable that the trend is sometimes broken on subscriptions 2 and 4 for some approaches. This is due to the small sample of subscriptions that represent each complexity level used in the experiment. A larger number of subscriptions is supposed to make the trend more apparent.

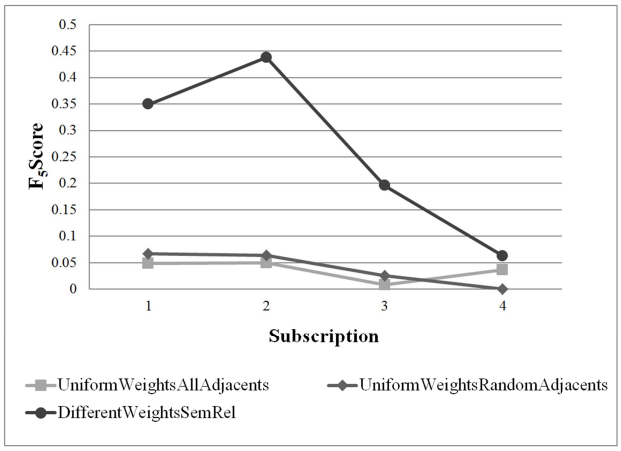

Figure 6. The combined F_5Score achieved by the enrichment approaches for each subscription

7. RELATED WORK

Related work to the event enrichment problem can be found in the event processing and middleware community as well as work in the database community on incompleteness. Hinze et al. [15] recognizes event enrichment among the features most required by event-enabled applications. Nevertheless, event enrichment is still widely addressed by ad-hoc dedicated agents which are tailored specifically to some situations. This is reflected in the set of enterprise integration patterns presented in [16]. Such approaches are non-native to the event processing paradigm where enrichment behaviour is pushed to the end user and less integrated with the rest of the features of event processing engines.

There are several research efforts to address the challenges of integrating background knowledge bases with event streams. Teymourian et al. [25] describes an approach based on a SPARQL-based query language where queries refer to event streams as well as the knowledge base. The authors recognize a set of categories of queries according to which they propose a set of execution plans. Similarly, Le-Phuoc et al. [18] proposes an approach to unify streams and background knowledge using Linked Data. They investigate methods to optimize the continuous queries over the resulting dynamic Linked Data based on cost-based optimization within time windows. Both approaches form good examples of efforts to address the need for using background knowledge bases with event-based systems. However, some assumptions are already made in these approaches such as the access mechanism to the knowledge base, the data models and the feasibility of using join operators. These assumptions may not hold in some situations and thus the event enrichment problem is not addressed natively within the event-processing paradigm.

While enrichment is generally understood as a process to fuse data from an external source with the events, there has been some work which tackled other aspects of enrichment. Petrovic et al. [22] proposes an approach to semantically match events with subscriptions. At one stage of the proposed system, events get enriched with synonyms of the terms that are used within the events. Such enrichment is interesting as it shows the semantic dimension of the problem. However, the approach does not tackle the enrichment problem in the general terms.

Some work from the database community identifies the problem of incomplete databases and incomplete queries. While the proposed approaches are more attached to databases in general and the relational model in particular, it still gives good insight on the problem. Some focus on missing tuples and missing values such as [12]. Some are more aligned to the query answering perspective such as [19] and [24]. While other works focus on improving the quality of incomplete databases [21].

8. CONCLUSIONS

This paper discussed the information incompleteness problem in event processing systems due to the decoupling principle. The dimensions of event incompleteness have been discussed along with challenges to overcome the incompleteness issue. A model for event enrichment has been proposed. The model is based on unifying enrichment within the event consumer logic and a native enricher that tackles incompleteness before matching. To validate the model, an instantiation using Linked Data events and a Linked Data cloud as an enrichment source has been discussed. The instantiation proposes spreading activation as a potential enrichment approach. Various strategies of spreading activation have been evaluated using a set of Wikipedia events and DBpedia as the enrichment source. Evaluation has been done using a composite completeness and precision measure and it showed a superiority of the spreading activation strategy that is based on semantic relatedness over the other two approaches. This indicates the benefit of the unified subscription model.

The proposed model has implications on various aspects of event processing. Namely, sharing and re-usability of enrichment elements which may help improve the overall information completeness of events, distribution of enrichment which can improve the overall enrichment time, and approximation in event processing systems.

Future work includes further investigation of the aforementioned implications, investigating proper optimization approaches for the native enricher such as caching and indexing is of potential importance in real-time situations. The same also applies to optimizing the precision aspect of enrichment where higher precision means less unnecessary retrieval operations and thus a better performance. Additionally, improvement of enrichment approaches and algorithms for specific instantiations is important to improve the completeness, precision and time performance.

9. ACKNOWLEDGMENTS

This work has been funded by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2).

10. REFERENCES

1. Atzori, L., Iera, A., and Morabito, G. The internet of things: A survey. *Computer Networks 54*, 15 (2010), 2787–2805.
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. DBpedia: A Nucleus for a Web of Open Data. *The Semantic Web 4825*, (2007), 722–735.
3. Berners-Lee, T. Linked Data- Design Issues. 2006. http://www.w3.org/DesignIssues/LinkedData.html.
4. Budanitsky, A. and Hirst, G. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics 32*, 1 (2006), 13–47.
5. Collins, A.M. and Loftus, E.F. A spreading-activation theory of semantic processing. *Psychological review 82*, 6 (1975), 407.
6. Crestani, F. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review 11*, 6 (1997), 453–482.
7. Curry, E., Hasan, S., and O'Riain, S. Enterprise Energy Management using a Linked Dataspace for Energy Intelligence. *Second IFIP Conference on Sustainable Internet and ICT for Sustainability*, IEEE (2012).
8. Curry, E., O'Donnell, J., Corry, E., Hasan, S., Keane, M., and O'Riain, S. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics 27*, 2 (2012), 206–219.
9. Curry, E. Message-Oriented Middleware. In Q.H. Mahmoud, ed., *Middleware for Communications*. John Wiley and Sons, Chichester, England, 2004, 1–28.
10. Etzion, O. and Niblett, P. *Event Processing in Action*. Manning Publications Co., 2010.
11. Eugster, P.T., Felber, P.A., Guerraoui, R., and Kermarrec, A.M. The many faces of publish/subscribe. *ACM Computing Surveys (CSUR) 35*, 2 (2003), 114–131.
12. Fan, W. and Geerts, F. Capturing missing tuples and missing values. *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, (2010), 169–178.
13. Hasan, S., Curry, E., Banduk, M., and O'Riain, S. Toward Situation Awareness for the Semantic Sensor Web: Complex Event Processing with Dynamic Linked Data Enrichment. *The 4th International Workshop on Semantic Sensor Networks 2011 (SSN11)*, (2011), 60–72.
14. Hasan, S., O'Riain, S., and Curry, E. Approximate Semantic Matching of Heterogeneous Events. *6th ACM International Conference on Distributed Event-Based Systems (DEBS 2012)*, ACM (2012), 252–263.
15. Hinze, A., Sachs, K., and Buchmann, A. Event-based applications and enabling technologies. *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems - DEBS '09*, (2009), 1.
16. Hohpe, G. and Woolf, B. *Enterprise integration patterns: Designing, building, and deploying messaging solutions*. Addison-Wesley Professional, 2004.
17. Klyne, G. and Carroll, J.J. Resource Description Framework (RDF): Concepts and Abstract Syntax. 2004. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.
18. Le-Phuoc, D., Dao-Tran, M., Xavier Parreira, J., and Hauswirth, M. A native and adaptive approach for unified processing of linked streams and linked data. *The Semantic Web--ISWC 2011*, Springer (2011), 370–388.
19. Levy, A. Obtaining complete answers from incomplete databases. *Proceedings of the International Conference on Very Large Data Bases*, (1996), 402–412.
20. OECD. Machine-to-Machine Communications: Connecting Billions of Devices. *OECD Digital Economy Papers No. 192*, 2012.
21. Parssian, A., Sarkar, S., and Jacob, V.S. Assessing information quality for the composite relational operation join. *Proc. Seventh Int'l Conf. Information Quality*, (2002), 225–237.
22. Petrovic, M., Burcea, I., and Jacobsen, H.-A. S-ToPSS: semantic Toronto publish/subscribe system. *Proceedings of the 29th international conference on Very large data bases - Volume 29*, VLDB Endowment (2003), 1101–1104.
23. Prud'Hommeaux, E. and Seaborne, A. SPARQL query language for RDF. *W3C working draft 4*, January (2008).
24. Razniewski, S. and Nutt, W. Checking query completeness over incomplete data. *Proceedings of the 4th International Workshop on Logic in Databases*, (2011), 32.
25. Teymourian, K., Rohde, M., and Paschke, A. Fusion of background knowledge and streams of events. *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, ACM (2012), 302–313.