IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 18, 2017, accepted September 26, 2017, date of publication October 9, 2017,
date of current version November 7, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2761232

A Malaria Analytics Framework to Support Evolution and Interoperability of Global Health Surveillance Systems

JON HAËL BRENAS[1], MOHAMMAD SADNAN AL-MANIR[2], CHRISTOPHER J. O. BAKER[2,3], AND ARASH SHABAN-NEJAD[1]

[1]Oak Ridge National Laboratory, Center for Biomedical Informatics, Department of Pediatrics, University of Tennessee Health Science Center, Memphis, TN 381032893, USA
[2]Computer Science Department, University of New Brunswick, Saint John, NB E2L 4L5, Canada
[3]IPSNP Computing Inc., Saint John, NB E2L 4S6, Canada

Corresponding author: Arash Shaban-Nejad (ashabann@uthsc.edu)

This work was supported by the Bill and Melinda Gates Foundation.

ABSTRACT Malaria is a leading cause of death in Africa. Many organizations, NGO's, and government agencies are collaborating to prevent, control, and eliminate malaria. In order to succeed in these shared goals, an integrated, consistent knowledge source to empower informed decision-making is required. Malaria surveillance is currently performed using dynamic, interconnected, systems which require rapid data exchange between different platforms. An important challenge these systems must overcome is the occurrence of dynamic changes in one or more interacting components, which can lead to inconsistencies and mismatches between components of the infrastructure. In this paper, we present our efforts toward the design and development of the semantic interoperability and evolution for malaria analytics platform, with the goal of improving data and semantic interoperability for dynamic malaria surveillance and to support the integration of data across multiple scales. The long term target is to deliver transparent and scalable tools for decision making for malaria elimination. Our analysis is focused on sentinel sites in selected African countries, including Uganda and Gabon.

INDEX TERMS Interoperability, change management, malaria surveillance, graph transformation, web services, semantics.

I. INTRODUCTION

Malaria is an infectious disease and one of the top causes of death in low-income developing countries (LIDCs) [1]. According to a 2016 WHO World Malaria Report [2], combining data from reports from 91 endemic countries, there were 212 million new cases of malaria in 2015, and an estimated 429,000 malaria deaths, worldwide. African countries accounted for almost 90% of global cases of malaria and 92% of malaria deaths (mostly young children in Sub-Saharan Africa). The malaria transmission season generally coincides with the planting and/or harvesting season in African countries when even a short period of ailment imposes a tremendous cost burden on the world's most economically challenged countries [3]. It is estimated that in highly endemic countries, malaria is responsible for a decrease in economic growth by more than 1% per year [3].

Malaria is caused by parasitic micro-organisms of the Plasmodium species (e.g. *Plasmodium falciparum* and *Plasmodium vivax*). The parasite is transmitted person-to-person, through an intermediate host/vector. The mosquito species Anopheles gambiae is the vector that is primarily responsible for malaria proliferation in Africa [4]. Many factors contribute to and influence the way malaria is transmitted. These include a range of environmental factors such as the location of mosquito vectors (vector ecology), weather cycles and climate change, deforestation, international travel [5], population growth, human movements, overuse of malaria drugs, housing, urbanization and several socioeconomic aspects [6] (e.g. poverty and the deterioration of public health infrastructures) [7].

In recent years, there have been growing investments in malaria control and research programs; the total funding for malaria control and elimination in 2015 was estimated to be $2.9 billion. However, this is still short of the $6.4 billion annual stipulated by the WHO Global Technical Strategy (GTS) for malaria [2], a target set to be achieved

by 2020. The goals of the GTS are, (i) to achieve a 90% reduction in malaria incidence and mortality rates compared with 2015, (ii) the elimination of malaria from at least 35 high-transmission malaria countries (mostly low-income developing countries), and (iii) the prevention of recurrence of malaria in all countries that are currently malaria-free [2]. All of these targets are to be achieved by 2030 and several organizations, partners, and stakeholders at national, regional, continental and global levels must work together to make the malaria elimination and eradication agenda a success.

In order to make timely decisions about where to locate malaria vectors and parasites and how to prevent the reoccurrence of malaria, an integrated real-time surveillance program is required [8]. This must be performed by a dedicated digital infrastructure using core information retrieval, data and knowledge management methodologies designed to address the unique challenges specific to global population health and epidemiology. These challenges include the following obstacles, (i) malaria data today are scattered across different countries, laboratories, and organizations in different heterogeneous data formats and repositories, (ii) a diversity of access methodologies make it difficult to retrieve all relevant data in a timely manner (iii) the absence of rich metadata on existing data and repositories limits the discoverability and reusability of data. Overall the current processes for discovering, accessing, and reusing the malaria specific data are inefficient, labour-intensive and error prone.

Furthermore, data about malaria must be integrated and interpreted in the context of existing knowledge models that describe the biology of malaria. Typically the knowledge models, also known as ontologies, have to be built in consultation with subject matter experts who have to manage multiple versions since their underlying knowledge and understanding of malaria is constantly changing. This further complicates interpretations and inferences that can be derived from surveillance data.

Likewise, malaria transmission and prevention are dynamic processes, therefore requiring formal mathematical malaria transmission models, which are necessarily quite complex [6], [9]. These mathematical models are often composed of several interacting elements, some even hidden, represented in dynamic inter-connected complex systems. Part of this complexity is due to topographical and climatic variations as well as human mobility [9].

Overall our knowledge about malaria and appropriate preventive measures becomes more comprehensive and therefore we expect many changes in existing malaria data management systems, data collection standards, and data stewardship over the next several years. Specifically, data and knowledge integration often result in changes such as extension, specialization, or adaptation in one or more data sources. Collectively these changes will make it more difficult to perform accurate data analytics or achieve reliable estimates of important metrics, such as infection rates.

Consequently, there is a critical need to rapidly assess the integrity of data and knowledge infrastructures that are depended on to support surveillance tasks. Reactive mechanisms to facilitate updates, fix errors, reclassify taxonomies, add/remove concepts, attributes, relations, and instances are required. Surveillance infrastructures currently in place today have yet to adequately address the core issues of ontology evolution, system interoperability, and semantic data integration.

A. RESEARCH STRATEGY

We present our efforts toward the development of the Semantic Interoperability & Evolution for Malaria Analytics (SIEMA) framework. Our objectives are to introduce a framework in which access to distributed data repositories containing malaria-related data is robust and consistent and to facilitate uninterrupted dynamic surveillance queries across multiple resources. Essential features of the infrastructure are; semantic interoperability among distributed resources, dynamic ad-hoc query access and a framework in which the multiple components of a surveillance infrastructure can be monitored for changes that make data access unreliable. Core contributions of our design lie in the use of:

1) Domain ontologies to capture knowledge for malaria control programming and to align, merge and integrate different models;
2) Semantic web services to ensure discoverability and interoperability of data retrieval and data transformation resources;
3) A series of software agents to monitor and report changes and evolution in distributed data resources, service descriptions, ontologies, and registries;
4) Graph Transformation rules to describe, verify and manage evolution and changes in the existing data sources and ontologies.

B. STATE-OF-THE-ART OF MALARIA SURVEILLANCE SYSTEMS AND DATA SOURCES

Malaria surveillance systems aim to assist public health practitioners and decision makers to (i) identify the regions or populations affected by malaria; (ii) identify trends in malaria morbidity and mortality and (iii) evaluate preventive or therapeutic malaria interventions and programs [10].

Malaria data is currently stored in distributed databases in different levels, locally and globally, and in various levels of granularities. Recent advances in knowledge and technology allow researchers to collect data from intelligent disease monitoring systems worldwide. The Scalable Data Integration for Disease Surveillance (SDIDS) [11] is an example of an application that enables the integration and analysis of malaria data across multiple scales to support global health decision-making. Africa Health Observatory (AHO) and real-time Strategic Information System (rSIS) [12] are other examples of surveillance systems that together aim at (i) monitoring and facilitating the prediction of events and early-warning systems (ii) sustaining the monitoring and evaluation of health reforms and priority health programs, (iii) enabling the generation and sharing of evidence for policy and decision-

making, and (iv) establishing and maintaining networks and communities of practice for the translation and application of evidence and knowledge sharing [12].

In Tanzania, an integrated mobile health system combining Coconut Surveillance [13] and Zanzibar's Malaria Case Notification (MCN) System [14] provide analysis of the geo-location of malaria cases and generates reports to health practitioners through SMS. The Swaziland national malaria surveillance program [15] is another example. Guintran *et al.* [16] and Ohrt *et al.* [17] provide non-exhaustive lists of different African malaria surveillance programs and systems.

There are several malaria data sources (e.g. databases and ontologies) and systems that we analyze and use in our project. The list of resources is as follows; Mapping Malaria Risk in Africa (MARA) [18], [19], is an open-access Web-based platform designed to extract and display raw malariometric data, with an emphasis on prevalence data. MARA represents data related to decades of malaria research in Africa; VecNet [20], provides an interface to model the impact of interventions on malaria transmission which is supported by a repository of integrated data from disparate sources; Global Malaria Mapper [21], [22], is a free platform that allows various stakeholders to create maps showing a range of themes (e.g. epidemiological profiles for countries and regions, reported cases, mortalities, and scale-up of interventions within a given geographical area); the Malaria Atlas Project [23], [24], maps the current parasite prevalence dataset provided by the USAID-funded MEASURE Demographic and Health Surveys (MEASURE DHS) repository [25]; VectorBase [26], [27], is an integrated database of vector information; Zambia's District Health Information Software 2 (DHIS2) [28], is an open source software platform for reporting, analysis, and dissemination of data for different health conditions, health program monitoring, and evaluation. Additionally there are ontologies such as the Ontology for Vector Surveillance and Management (VSMO) [29], Malaria Ontology (IDOMAL) [30], [31], Mosquito Insecticide Resistance Ontology (MIRO) [32], [33], HealthMap [34], [35], and other regional/local databases for malaria and health metrics across the region.

Our proposed platform aims to provide a mechanism to manage the evolution of these and other similar data sources and improve the interoperability of the malaria surveillance systems.

II. THE SIEMA ARCHITECTURE

The SIEMA architecture follows a standard multi-tier information system design. Fig. 1 shows SIEMA's main components which are a presentation tier, a query tier, a service tier and a data access tier. Within the architecture there is considerable use of, and adherence to, data and knowledge management standards recommended by the World Wide Web Consortium (W3C) [36], for different tasks; for example HTTP [37] for the server side HTTP request to establish communication channels between the components, Resource Description Framework (RDF) [38] graphs for representation of the data in a relational database, Web Ontology Language (OWL) [39] for representing the components in ontologies and SPARQL [40] for querying purposes. Additionally, we use the Semantic Automatic Discovery and Integration (SADI) [41] design patterns for modeling the web services I/O.

A. PRESENTATION TIER

The presentation tier aims to provide resources for users to view global infrastructure changes and allow users to respond using an interactive environment. This tier comprises two main elements: a reporting tool in the form of a dashboard, and a semantic query interface. The first element, the dashboard presents a collection of widgets. Each widget presents the status of some predefined metrics in the form of numbers or textual information. Specifically, the following infrastructure components are monitored for changes that are displayed in the dashboard widgets; domain ontologies, service ontologies, and databases. Additionally, dependencies among the system components are monitored constantly to analyze changes occurring in one component (e.g. a data source) and the potential effects they might have on the rest of the system. The dashboard also allows users to explore the metrics further to pinpoint the reasons behind the failure of an operational service.

The second element, the query interface is a graphical query gateway that permits users to construct queries using domain specific keywords and generate a semantic graph of nodes and edges. The graph is translated into the query language SPARQL used by the query engine in the Query Tier. This type of query composition process allows non-technical users to compose queries without learning the syntax and complexity of the SPARQL and still define complex queries to search for relevant malaria data and information.

B. QUERY TIER

The query tier leverages a query engine designed to receive a query in SPARQL syntax. The query engine matches predicates in the SPARQL query to predicates that describe the functionality of deployed SADI semantic web services, hosted in a registry. Given that complex queries must recruit multiple services to deliver the requested data the engine also performs query planning, designing and running a bespoke workflow for each query. The query engine can be used to query multiple distributed sources of data and return the outputs, as if the user had queried a single database. Two existing query engines operate on SADI services, SHARE [42] and HYDRA [43], [44].

C. SERVICE TIER

The service tier comprises two components (i) one or more registries for hosting of service descriptors (ii) the service descriptors in the form of service ontologies that represent descriptions of expected Web service inputs, and the provisional outputs. HYDRA and SHARE are able to discover and

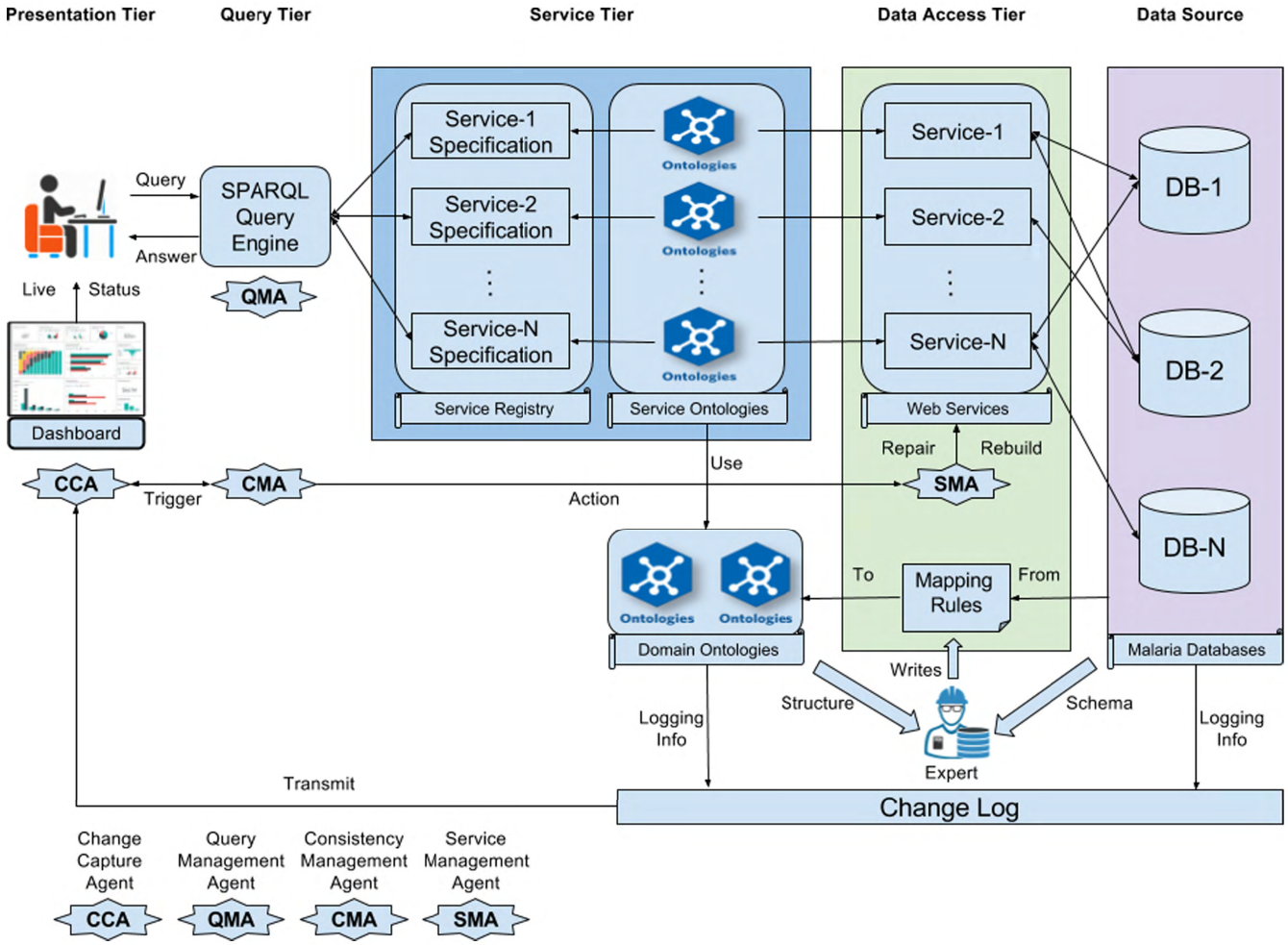

FIGURE 1. A diagram representing the architecture of the SIEMA Infrastructure.

interpret service ontologies deployed in a registry. A brief description of each of the registry and services is given below.

1) SERVICE REGISTRY

Every deployed SADI semantic web service is hosted in a registry. Each service has a unique endpoint which responds to HTTP GET with a service interface document containing the input and output definitions expressed in OWL classes with object and/or data properties restriction. Services are discovered based on the properties and the types of input. Invocation of a service is performed through plain HTTP POST of RDF data to the service endpoint. For each service, the registry also contains auxiliary information such as the description of the service's functionality, information about its creator, and a unique name, all of which are based on the myGrid ontology [45] for describing functions of web services and their parameters.

2) SERVICE ONTOLOGIES

Input and output of each service are defined in the service ontology as an OWL class expression along with the related data/object property restrictions. The input RDF data is classified against the input OWL class. After the service is executed, this RDF data is enriched with properties attached to instances or literals such that the enrichment is classified against the output OWL class. This enrichment preserves the SADI principle that the URI of the input OWL class instance is the same as the URI of the output OWL class instance, having a common node as root.

D. DATA ACCESS TIER

This tier provides access to data using Semantic Web services. Each service is built based on the SADI framework. SADI services only consume and produce data modeled in RDF format. As a result, each service must read input values from RDF data and write the results into RDF data, before and after executing the services, respectively. Just like a typical database query service, a SADI service may include database-specific SQL queries.

This tier also includes mapping rules, which are expressive rules to map relational database schemas to the domain ontologies. Expressive rule languages such as

TABLE 1. A snapshot of the table species bionomics in VecNet.

Id	Species	Form	Vector Status	Daily Adult Survival Rate	Larval Survival Rate	Indoor Feeding Rate
1	An. arabiensis	Not available	Dominant	78 - 83	49 - 87	37 - 74
2	An. funestus	ss	Dominant	63 - 97	49 - 61	49 - 89
3	An. gambiae	ss	Dominant	77 -95	60 - 80	45 - 67
4	An. melas	Not available	Dominant	Parity only	80	50 - 56
5	An. merus	Not available	Dominant	Parity only	46 - 79	40
6	An. moucheti	Not available	Dominant	88	No data	30 - 83
7	An. nili	ss	Dominant	90 - 95	No data	31 - 88
8	An. farauti	Not available	Not available	74 - 76	No data	No data

Rule Interchange Format (RIF) [46], and Positional-Slotted Object-Applicative (PSOA) [47] are used for this purpose.

E. ROLE OF AGENTS IN SIEMA

1) CHANGE CAPTURE AGENT (CCA)

The *CCA* checks various data sources in order to detect and identify changes. It does so by using two different mechanisms:

a) The *CCA* keeps track of queries used and compares their answers over time. If the same query gives different answers at different times, it indicates that some part of the data has been modified.

b) The *CCA* also looks at change logs that are created by tools like RacerPro [48] or Protégé [49] used for ontology reasoning and editing. When a new entry is made in a change log, the *CCA* flags that some change has occurred.

2) CONSISTENCY MANAGEMENT AGENT (CMA)

The role of the *CMA* is to keep an eye on the consistency of the system. It sends and receives information to and from the *CCA* to know the status of the services. Based on the output generated by the *CCA*, if the *CMA* determines that one or more services in the *Data Access Tier* are prone to malfunction, it sends a signal to the *SMA*, which then prompts to rebuild the affected service with updated information.

3) SERVICE MANAGEMENT AGENT (SMA)

The task of the *SMA* is to build SADI services. Currently, query services can be built automatically [50]. The *SMA* will build a SADI service only if the *Action* signal is transmitted from the *CMA*.

III. GRAPH AND TRANSFORMATIONS

In this section, we introduce the underlying formalism that we use to represent the malaria resources (e.g. ontologies, databases), the related data and their evolution. Simply put, a graph is composed of a set of nodes that represent individual elements that we aim to represent, and of a set of edges, that represent the relationships between the individuals. Graph transformations allow creating a new graph from an existing one by applying rules. Graph transformations can take several forms, the most popular being the algebraic approach [51], [52] and algorithmic methods that make use of actions. In this paper, we choose a more algorithmic approach [53], [54].

A. DATA REPRESENTATION

One of the most important problems SIEMA is trying to address is the interoperability between various data sources with different lexicons, semantics, and languages. The interoperability is achieved operationally by using semantic web services along with an abstract formal language. The abstract language enables the users to ignore the specifics of each source language and concentrate on the information they contain.

We follow the standards to represent data (RDF) and ontologies (OWL). Even for sources that do not typically follow the RDF standard (for instance, databases), the conversion to graph representation [55] is always an option. Example 1 shows the translation of data coming from a table in VecNet database into a graph. The abstraction of the data as a graph is only used as an intermediate device on which it is easier to reason. It is easier to define an action on the graph, say find the ages of all patients, than to define the specific action required for each source.

Example 1: Table 1 is a snapshot at a table in VecNet. Fig. 2 contains some of the information of Table 1 represented as a graph. There are many ways to interpret a table depending on the language that is used. In this case, we assume that circles are ontological concepts (e.g. `1` *is the instance of the concept* Anopheles *corresponding*

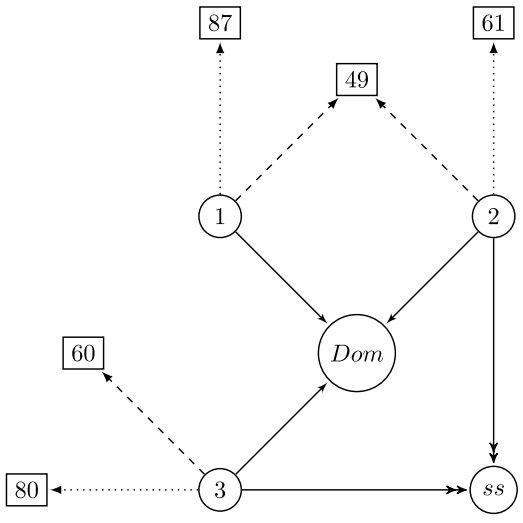

FIGURE 2. A graph representation of part of the table Species Bionomics in VecNet.

to the row with id 1 *while* Dom *is the instance Dominant of the concept* VectorStatus*) while rectangles are data values, that is to say represent actual values not concepts (e.g.* 87 *represents the integer 87). Edges represent relations between nodes. For instance, plain arrows represent the relation* ***HasVectorStatus****, double-headed arrows represent* ***HasForm****, dashed arrows represent* ***HasLarvalSurvivalRateMinimum*** *and dotted arrows represent* ***HasLarvalSurvivalRateMaximum****. The edge between* 2 *and* Dom *is thus interpreted as* ***HasVectorStatus****(*2, Dom*) which is the same information that can be found in the table at the intersection of the Row* 2 *and the column* Vector Status.

The formal definition we use slightly differs from the popular, and less expressive, definition. In addition to the sets of nodes and edges, we define functions that label nodes and edges with attributes. These attributes are formulae from a logic. This is why we call them *logically decorated graphs*. For the rest of this paper, graphs should be interpreted as logically decorated graph.

Definition 2 (Logically Decorated Graph): Let $\mathcal{L}$ *be a logic (set of formulae). A graph alphabet is a pair* $(\mathcal{C}, \mathcal{R})$ *of sets of elements of* $\mathcal{L}$*, that is* $\mathcal{C} \subseteq \mathcal{L}$ *and* $\mathcal{R} \subseteq \mathcal{L}$*.* $\mathcal{C}$ *is the set of node formulae or concepts and* $\mathcal{R}$ *is the set of edge formulae or roles. Subsets of* $\mathcal{C}$ *and* $\mathcal{R}$*, respectively named* $\mathcal{C}_0$ *and* $\mathcal{R}_0$*, contain basic (propositional) concepts and roles respectively. A logically decorated graph* G *over a graph alphabet* $(\mathcal{C}, \mathcal{R})$ *is a tuple* $(N, E, \Phi_N, \Phi_E, s, t)$ *where* N *is a set of nodes,* E *is a set of edges,* Φ_N *is the node labeling function,* $\Phi_N : N \to \mathcal{P}(\mathcal{C})$*,* Φ_E *is the edge labeling function,* $\Phi_E : E \to \mathcal{P}(\mathcal{R})$*,* s *is the source function* $s : E \to N$ *and* t *is the target function* $t : E \to N$*.*

First we need to define the logic we use to decorate the graphs. Several different types of logic have been proposed to describe graphs [56]–[59]. Generally, the choice of the logic depends on various factors, most importantly to the underlying representation needed for a given problem and the types of inferences that need to be drawn. In this study, we use first order logic, because first of all it is arguably one of the most well-known and used types of logics and also it is expressive enough to represent the properties that we need.

Choosing the right logic for a specific application is no trivial task. Different applications may require to express properties of higher order and to use highly expressive axioms (e.g. restrictions). On the other hand, many problems in expressive logics are undecidable, which limits their usability. One may thus have to accept a trade-off between expressivity and computational efficiency.

B. DATA MODIFICATION

Graphs are used to represent the data and they allow us to work with an abstract structure that allows us to forget about sources' characteristics. Yet, the fact is that malaria surveillance data sources are bound to evolve. New knowledge is obtained, old data become obsolete, insects develop new resistances to pesticides, people move, new drugs are created and old ones are dropped, climate and socioeconomic situations change, and so forth. In the same way, people may decide to structure data differently or new troves of data are created or become available or relevant.

The most common change is the addition or deletion of data, tables or the modification of the schemas in a database or a data-warehouse. Additions and deletions occur in ontologies as well, to keep the knowledge current.

Example 3: Fig. 3 shows a fragment of the IDOMAL ontology in graph representation. It shows that the exact same concept (e.g. Anophelinae) can be defined in several different ontologies without any connection between the definitions. To be more precise, it is possible for an element to be considered as Anophelinae according to one definition and not the other

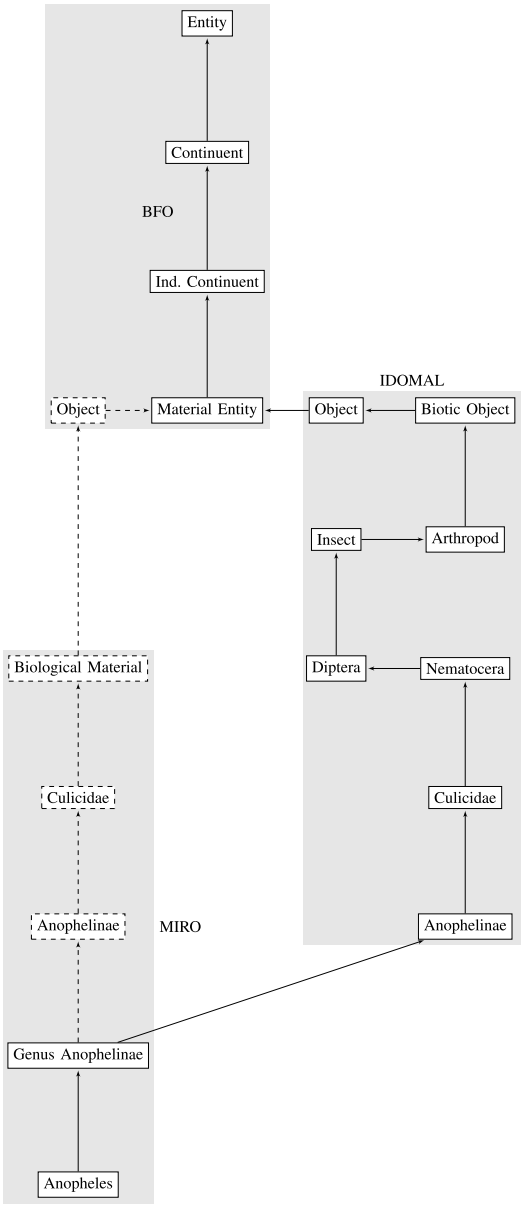

FIGURE 3. A graph representation of part of the ontology IDOMAL. Plain arrows represent the relation *is_a* that is defined in IDOMAL; dashed nodes represent classes defined in MIRO but not in IDOMAL; dashed arrows represent the relation is_a that is defined in MIRO.

despite the fact that both are expected to represent the same concept. In this example, when integrating IDOMAL and MIRO, since the concepts Anophelinae in both ontologies are equivalent, they can be merged into one. On the other hand, the class Insect in IDOMAL does not seem to have an equivalent in MIRO. The concept Biological Material in MIRO represents dead or live insects. It is, however, possible to have Insects, defined in IDOMAL, that are not Biological Materials, defined in MIRO, and vice-versa because their definitions are not related even though the concepts they represent are.

As we decided to model data sources as graphs, we consider all the changes as being graph transformations. We use an algorithmic representation to define graph transformations. In this research we use the algorithmic approach defined in [60]. The most basic components of graph transformation that we consider are elementary actions. These are actions that modify a graph in a predefined way. They allow to:

- Add (or remove) a label to (or from) a node or an edge;
- Create or delete nodes and edges;
- Redirect incoming or outgoing edges from a node to another; or
- Clone a node or to merge two existing ones.

Based on different situations many such elemental actions can be defined, subject to checking the theorems afterward to see if they still hold when such elemental actions are allowed. For the sake of brevity, here we only define a few of such actions.

Definition 4 (Elementary action, action): An elementary action, say a, may be of the following forms:

- *a node addition $add_N(i)$ (resp. node deletion $del_N(i)$) where i is a new node (resp. an existing node). It creates the node i. i has no incoming nor outgoing edge and it is not labeled with any basic concept $\Phi_N(i) \cap \mathcal{C}_0 = \emptyset$ (resp. it deletes i and all its incoming and outgoing edges).*
- *a concept addition $add_C(i, c)$ (resp. concept deletion $del_C(i, C)$) where i is a node and c is a basic concept (a proposition name) in $\mathcal{C}_0$. It adds the label c to (resp. removes the label c from) the labeling of node i.*
- *a role addition $add_R(i, j, r)$ (resp. role deletion $del_R(i, j, r)$) where i and j are nodes and r is an basic role (edge label) in $\mathcal{R}_0$. It adds the label r to (resp. removes the label r from) the labeling of the edge represented by the pair (i, j).*
- *a node merging $mrg(i, j)$ where i and j are different nodes. It is the elementary action that merges i and j. It redirects all edges coming from (resp. going to) j toward i and adds the labels labeling j to the labeling of i. It also removes j.*

An action, say α, is a sequence of elementary actions of the form $\alpha = a_1; a_2; \ldots; a_n$. The result of performing α on a graph G is written $G[\alpha]$. $G[a; \alpha] = (G[a])[\alpha]$ and $G[\epsilon] = G$ with ϵ being the empty sequence.

The result of performing the elementary action α on a graph $G = (N^G, E^G, \Phi_N^G, \Phi_E^G, s^G, t^G)$, written $G[\alpha]$, produces the graph $G' = (N^{G'}, E^{G'}, \Phi_N^{G'}, \Phi_E^{G'}, s^{G'}, t^{G'})$ defined as:

- If $\alpha = add_C(i, c)$ then:
  - $N^{G'} = N^G$
  - $E^{G'} = E^G$
  - $\Phi_N^{G'}(n) = \begin{cases} \Phi_N^G(n) \cup c & \text{if } n = i \\ \Phi_N^G(n) & \text{if } n \neq i \end{cases}$
  - $\Phi_E^{G'} = \Phi_E^G$
  - $s^{G'} = s^G$
  - $t^{G'} = t^G$
- If $\alpha = del_C(i, c)$ then:
  - $N^{G'} = N^G$
  - $E^{G'} = E^G$
  - $\Phi_N^{G'}(n) = \begin{cases} \Phi_N^G(n) \backslash c & \text{if } n = i \\ \Phi_N^G(n) & \text{if } n \neq i \end{cases}$
  - $\Phi_E^{G'} = \Phi_E^G$
  - $s^{G'} = s^G$
  - $t^{G'} = t^G$
- If $\alpha = add_R(i, j, r)$ then:
  - $N^{G'} = N^G$,
  - $\Phi_N^{G'} = \Phi_N^G$
  - $E^{G'} = E^G \cup e$ where e is a new element
  - $\Phi_E^{G'}(e') = \begin{cases} r & \text{if } e' = e \\ \Phi_E^G(e') & \text{if } e' \neq e \end{cases}$
  - $s^{G'}(e') = \begin{cases} i & \text{if } e' = e \\ s^G(e') & \text{if } e' \neq e \end{cases}$
  - $t^{G'}(e') = \begin{cases} j & \text{if } e' = e \\ t^G(e') & \text{if } e' \neq e \end{cases}$
- If $\alpha = del_R(i, j, r)$ then:
  - $N^{G'} = N^G$
  - $\Phi_N^{G'} = \Phi_N^G$
  - $E^{G'} = E^G \backslash \{e | s^G(e) = i \wedge t^G(e) = j \wedge \Phi_E^G(e) = r\}$
  - $\Phi_E^{G'}$ is the restriction of Φ_E^G to $E^{G'}$
  - $s^{G'}$ is the restriction of s^G to $E^{G'}$
  - $t^{G'}$ is the restriction of Φ_E^G to $E^{G'}$
- If $\alpha = add_N(i)$ then
  - $N^{G'} = N^G \cup i$ where i is a new node
  - $\Phi_N^{G'}(n') = \begin{cases} \emptyset & \text{if } n' = n \\ \Phi_N^G(n') & \text{if } n' \neq n \end{cases}$
  - $E^{G'} = E^G$
  - $\Phi_E^{G'} = \Phi_E^G$
  - $s^{G'} = s^G$
  - $t^{G'} = t^G$
- If $\alpha = del_N(i)$ then:
  - $E^{G'} = E^G \backslash \{e | s^G(e) = i \vee t^G(e) = i\}$
  - $N^{G'} = N^G \backslash i$
  - $\Phi_N^{G'}$ is the restriction of Φ_N^G to $N^{G'}$
  - $\Phi_E^{G'}$ is the restriction of Φ_E^G to $E^{G'}$
  - $s^{G'}$ is the restriction of s^G to $E^{G'}$
  - $t^{G'}$ is the restriction of Φ_E^G to $E^{G'}$

- If $\alpha = mrg(i, j)$ then:
  - $N^{G'} = N^G \backslash \{j\}$
  - $E^{G'} = E^G$
  - $\Phi_N^{G'}(n) = \begin{cases} \Phi_N^G(i) \cup \Phi_N^G(j) & \text{if } n = i \\ \Phi_N^G(n) & \text{otherwise} \end{cases}$
  - $\Phi_E^{G'}(e) = \Phi_E^G(e)$
  - $s^{G'}(e) = \begin{cases} i & \text{if } s^G(e) = j \\ s^G(e) & \text{otherwise} \end{cases}$
  - $t^{G'}(e) = \begin{cases} i & \text{if } t^G(e) = j \\ t^G(e) & \text{otherwise} \end{cases}$

Elementary actions in and of themselves are not enough to describe the complex changes that we are interested in. In particular, all elementary actions work on nodes that need to be provided. On the other hand, one often wants to apply the same change to all elements of a table in a database or a sub-taxonomy in an ontology with a given property without having to enumerate them. For instance, one might want to flag all patients that have had malaria in the past as possible healthy carriers without having to actually name every single one of them. In order to tackle this problem, we introduce the notion of *logically decorated rewriting systems*. These are extensions of graph rewriting systems defined in [53] where graphs are attributed with formulas from a given logic. The left-hand sides of the rules are thus logically decorated graphs whereas the right-hand sides are defined as sequences of elementary actions.

Definition 5 (Rule, Logically Decorated Rewriting System): A rule ρ is a pair (LHS,α) where LHS, called the left-hand side, is an attributed graph with formulae as attributes and α, called the right-hand side, is an action. Rules are usually written LHS $\rightarrow$ α. A logically decorated rewriting system, LDRS, is a set of rules.

The fact that the left-hand side of a rule is an attributed graph, and that it can contain nodes labeled with formulae, is important. Indeed, these formulae can express reachability (closure of a program), conditions on the number of neighbors (counting quantifiers), non-local properties (universal quantifiers) and so forth depending on the chosen logic.

Example 6: Fig. 4 shows examples of rules that could be applied to a knowledge base consistent with the IDOMAL ontology. The first rule, ρ_0, looks for a Process *and a* Duration *and adds the fact that the process lasts for the duration.*

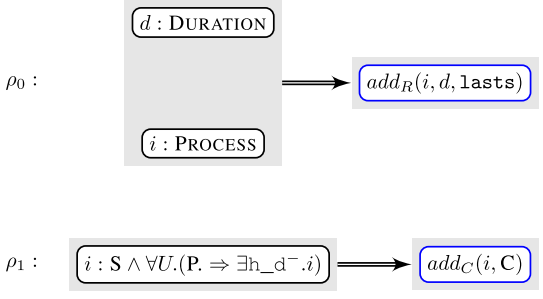

FIGURE 4. Examples of rules: ρ_0 adds a duration to a process; ρ_1 makes a Symptom that happens_during every Process a Constant.

ρ_1 looks for a Symptom *that* ***happens_during*** *every* Process*, that is such that every* Process *has an incoming edge labeled with* ***happens_during*** *coming from the* Symptom*, and labels it as* Constant.

Rewrite rules and logically decorated rewriting systems offer much more flexibility than elementary actions. They allow applying the change to elements that satisfy conditions instead of named elements, which is to modify elements knowing their properties and not their identities. In order to find the elements that the left-hand side can be mapped to, we define the notion of "Match".

Definition 7 (Match): A match h between a left-hand side LHS and a graph G is a pair of functions $h = (h^N, h^E)$, with $h^N : N^{LHS} \rightarrow N^G$ and $h^E : E^{LHS} \rightarrow E^G$ such that:

1) $\forall n \in N^{LHS}, \forall c \in \Phi_N^{LHS}(n), h^N(n) \models c$
2) $\forall e \in E^{LHS}, \Phi_E^G(h^E(e)) \models \Phi_E^{LHS}(e)$
3) $\forall e \in E^{LHS}, s^G(h^E(e)) = h^N(s^{LHS}(e))$
4) $\forall e \in E^{LHS}, t^G(h^E(e)) = h^N(t^{LHS}(e))$

The third and the fourth conditions are classical and highlight the fact that the source and target functions and the match have to agree. The first condition says that for every node n of the left-hand side, the node, $h(n)$, to which it is associated in G has to satisfy every concept that n satisfies. This condition clearly expresses additional negative and positive conditions, which are added to the "structural" pattern matching. The second condition ensures that the match respects edge labeling as well.

Definition 8 (Rule Application): A graph G rewrites to graph G' using a rule $\rho = (LHS, \alpha)$ iff there exists a match h from LHS to G. G' is obtained from G by performing actions in $h(\alpha)$.[1] Formally, $G' = G[h(\alpha)]$. We write $G \rightarrow_\rho G'$ or $G \rightarrow_{\rho,h} G'$.

When considering a system of rewrite rules, it is natural to consider the order in which the rules are applied. Confluence, the property that indicates there is only one possible result, of graph rewriting systems is not always easy to establish. For instance, orthogonal graph rewrite systems[2] are not always confluent (see [53] for examples) even though orthogonal term rewrite systems are. So we use the notion of rewrite strategies to control the use of possible rewrite rules. Informally, a strategy specifies the application order of different rules. It does not point to where the matches are nor does it ensure the unicity of the reduction outcome.

Definition 9 (Strategy): Given a graph rewriting system $\mathcal{R}$, a strategy is a word of the following language defined by s:

$s :=$	ϵ	*(Empty strategy)*	ρ	*(Rule)*
	$s; s$	*(Composition)*	$s \oplus s$	*(Choice)*
	s^*	*(Closure)*		

where ρ is any rule in $\mathcal{R}$.

We write $G \Rightarrow_{\mathcal{S}} G'$ when G rewrites to G' following the rules that are given by the strategy $\mathcal{S}$.

[1] $h(\alpha)$ is obtained from α by replacing every node name, n, of *LHS* by $h(n)$.

[2] Orthogonal graph rewrite systems are systems in which there cannot be more than one rule that can be applied to a given subgraph at any step.

Intuitively, the strategy ρ consists of applying the rule ρ once. The strategy ϵ does nothing. The composition $s_0; s_1$ applies the strategy s_0 and then the strategy s_1 while the choice $s_0 \oplus s_1$ non-deterministically applies either s_0 or s_1. The closure s^* applies a strategy for as long as possible.

We have to mention that using the closure forces us to be cautious while designing transformation rules. It is indeed easy to devise strategies that are obviously non-terminating. Even though non-terminating strategies are not forbidden per se, they do not usually represent the changes that a surveillance expert or ontology manager has in mind as one would expect the changes we enact to have an end that is a final state. Example 10 contains an example of such a non-terminating strategy.

Example 10: The rules presented in Fig. 4 may seem sensible and one may want to apply the strategy ρ_0^* to give to each Process *a* Duration *but an application of such a strategy, provided there exist at least one* Process *and at least one* Duration*, would never stop as ρ_0 does not preclude the creation of an already existing edge. An alternative rule, ρ_0', that would yield a terminating strategy is given in Fig. 5. Applying the strategy $\rho_0'^*$ terminates, provided there is a finite number of* Processes*, as it can be applied, at most, once by* Process*.*

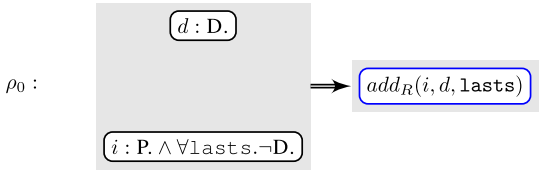

FIGURE 5. Examples of rules: ρ_0 adds a duration to a process that does not already have one.

Let us now give an example of transformation with a slightly more complex strategy.

Example 11: One of the most common changes in ontologies is the identification of concepts defined in different ontologies but that are equivalent. For instance, as shown in Fig. 3, both IDOMAL and MIRO define a concept Culicidae. Assuming that the graph we consider describes both ontologies (and, in particular, contains a node named MIRO:Culicidae *and another node named* IDOMAL:Culicidae*) and contains some data about Culicidae (that is there exist a label* MIRO:Culicidae *and a label* IDOMAL:Culicidae*), one could apply the strategy $\rho_0; \rho_1^*$ to update the graph. Applying ρ_0 merges the definitions in the two ontologies while ρ_1^* relabels the nodes that are affected by the change. This example is illustrated in Fig. 6.*

In the following, because the transformations we use as examples are designed to be simple and intuitive, one rule will be sufficient most of the times. In the following, we will define rules, and strategies when applicable, with examples.

IV. CASE STUDY

Now that graphs and their transformations have been formally defined, we look at changes in the context of malaria data

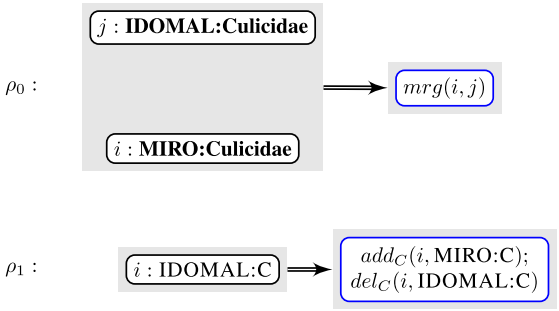

FIGURE 6. Rules that redefine *IDOMAL:Culicidae* as *MIRO: Culicidae* and make the appropriate changes to the knowledge base.

sources. Changes can come in many varieties affecting different components of a malaria surveillance infrastructure. However, it does not mean that these changes are completely isolated with no interaction with each other. We examine our model by analyzing standalone restricted changes first. In some cases, changes may affect the consistency of the captured knowledge and in turn may threaten the validity of the logically inferred knowledge. Example 12 presents such a case. Here, we do not consider changes that cause inconsistency (in such cases, the focused should be on the repair rather than querying).

Example 12: Let us assume that our domain ontologies contain an axiom stating that only Biological Material *entities have a* ***Lifespan*** *and an axiom stating that a* Geographical Location *is not a* Biological Material*. We also assume that in our database there is a table that states* Uganda *is a* Geographical Location *whose* ***Capital*** *is* Kampala*. If the database evolves in a way, say by applying the rule presented in Fig. 7, that adds the assertion that* Uganda *has a* ***Lifespan****, the knowledge becomes inconsistent, since it is possible to infer that* Uganda *is a* Geographical Location*, as this is an assertion in the database, and that* Uganda *is not a* Geographical Location*, as it has a* ***Lifespan*** *making it a* Biological Material *that cannot be a* Geographical Location*.*

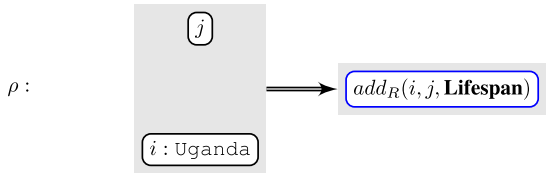

FIGURE 7. A rule that adds a **Lifespan** to Uganda.

We classify changes by specifying what part of the knowledge can be changed.

A. TARGET OF CHANGE

"Data" is the most frequently changing element in a knowledge base. Some pieces of data are added to databases while others are discarded. This is the most common change and one that happens during the development phase as well as during its operation.

Example 13: Let us assume that we have a database containing information about Geographical location. *In particular, it keeps track of the current* ***Weather*** *and* ***Temperature****. In order to do that, captors have been used to return a value with a given periodicity. Each time a new set of measurement is produced, the database has to be updated: the old values are discarded and the new ones are inserted. One of the rules that are used could be the one from Fig. 8 that updates the* ***Temperature*** *of a* Geographical location.

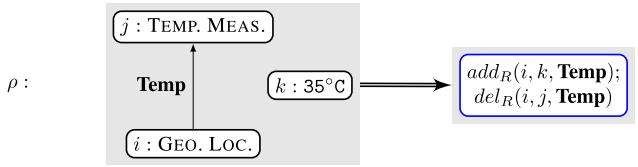

FIGURE 8. A rule that updates the **Temperature** of a ***Geographical location*** to 35°C.

Domain ontologies are also subject to change. Many domain ontologies are under active development reusing parts of existing domain ontologies and creating new ones. When working with different interconnected domain ontologies it is not unusual to find redundancy and heterogeneity (e.g. concepts that are represented in several domain ontologies in different ways, structurally and semantically). Improving the interoperability between these ontologies requires changes in one or multiple components.

Example 14: Fig. 3 shows a fragment of the IDOMAL ontology in graph representation. In order to improve the interoperability between MIRO and IDOMAL, one could apply the rule ρ_0 *of Fig. 6.*

Also, database schemas change when new kinds of information are added or deleted, for instance by creating a new table replacing what was a column in a previous table. This has a deep impact on the related ontologies that interpret the data since they have to be able to represent the new information. Thus, changes in the schema usually force changes in the related ontologies, or at least in the rules interpreting the database as an instance of the ontology.

Example 15: Let us assume that the data contains a table where Geographical Location*s are linked to their* Country, Name, Gaul Code *and to the estimated* Percentage of Households with *ITN and another one where* Geographical Location *are linked to their* Country, Name, Gaul Code *and to the estimated* Percentage of Population Protected by *IRS. There is obviously many overlaps between the two tables. To reduce the cost of storing the database, it makes sense to keep only one table with all the information about* Geographical Location*s. Applying the rule of Fig. 9 with the strategy* ρ^* *would yield that result.*

After introducing a few different types of changes, we now describe how to handle them; more specifically how to detect the change and how to consistently manage their effects. We focus first on the detection.

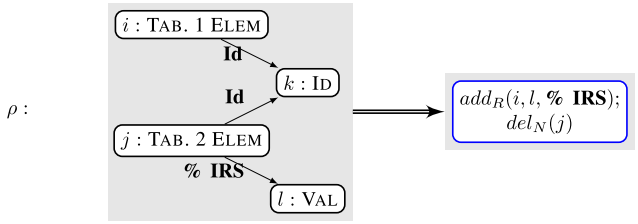

FIGURE 9. A rule that updates tables by moving the content of a column from one to the other.

B. CHANGE DETECTION

Before being able to update the infrastructure to handle the changes, it is crucial to actually detect such changes. This task is performed by the Change Capture Agent. How to detect those changes depends mostly on what has changed. In the case of a change in the data, a change is detected when a query returns a result that differs from the one it used to return. In order to detect that kind of change, one has to store the result of previous queries and check whether the new result is the same or not. The infrastructure is thus extended by the addition of a triple-store that contains a query, its timestamp, and its result. This allows tracking the evolution of data in the infrastructure.

Example 16: Assume that the data contains a table with various Ugandan cities and their population. We query by asking for the list of all cities with more than 200,000 inhabitants. The result, in 2011, is only `Kampala`*. This result is stored in the triple-store as {Cities with a population greater than 200,000, 2011, (*`Kampala`*)}. The query is run again in 2014 and the result is now (*`Kampala`*,* `Nansana`*,* `Kira`*). A new triple is added, namely {Cities with a population greater than 200,000, 2014, (*`Kampala`*,* `Nansana`*,* `Kira`*))}. It is then compared to the previous one. The set of answers has changed and thus a change has occurred.*

Changes in the domain ontology can also be detected using the stored triples. It is, however, much less efficient as modifications of the domain ontologies affect services more than their results. This means a service may return the same result even though the data changed but the service did not notice the change.

Example 17: Assume that a query has been produced that returns the ***name****s of all* Culicidae *from IDOMAL, a domain ontology. IDOMAL is modified by adding the fact that the concept* Culicinae*, defined in the domain ontology VectorBase CV [61], is a subconcept of* Culicidae*. This means all* Culicinae*s are also* Culicidae*s. If the same query as before is used, even assuming that there exists a service that lists the* ***name****s of all* Culicinae*s, they will not be considered because the query engine does not know how to use the service.*

Moreover, the impact of the change is more profound when the domain ontologies change. Indeed, as the domain ontology is used to create the services, if the domain ontology changes, the language and axioms used in the query may no longer represent valid knowledge; thus the query completely fails because it no longer conveys a meaning.

In that case, the Consistency Management Agent (CMA) detects the inconsistencies and requests an update to resolve it. CMA also indicates whether a change has occurred in the domain ontology or in the database schema. After detecting the changes we need to be able to identify and classify them as well. To do so, we use change logs that keep track of the changes. The change logs stored historical data expressed in a uniform language, which is the same for all kind of changes. This makes it possible to parse the logs and update the service ontologies, the web services, the queries and, possibly, the translation rules accordingly.

C. HANDLING CHANGE

Once a change has been detected and identified, it is possible to update the infrastructure to cope with the change using the Service Management Agent. The magnitude and scope of change and its target location determine how it is handled. If it only affects the data, e.g. new pieces of information are added or deleted, without changing the database schema, the change management will be relatively easy, as it only changes the results of the queries. On the other hand, when the underlying domain ontologies used by web services are modified, the services become inconsistent with the ontologies. Therefore, the service ontologies need to be rebuilt to be once again consistent with the domain ontologies.

Example 18: Let us consider IDOMAL as our domain ontology. We are interested in the ***Name****s of all* Culicidae *and that there exists a service that takes as input an* IDOMAL:Culicidae *and outputs its* ***Name****. At that point, we realize that some tables are interpreted as being* MIRO:Culicidae *and we decide that they represent the same concept and thus merge them. A new service may have to be created that takes as input a* MIRO:Culicidae *and outputs its* ***Name****. Both services are then used to answer the query.*

When we modify a database schema the change management process becomes more challenging, as one has to redefine the translation rules between the schema and the domain ontologies. This is not easy to automate as some changes may be interpreted in different ways and the modified element in the database (e.g. a column) can either correspond to several different concepts defined in the domain ontologies or none.

Example 19: Let us assume that the database has a table containing information about Culicidae*. A new column* ***exhibitsTrait*** *is added whose value is a string.*

Assuming data is added to the current knowledge, there may be no specific term in the domain ontologies that represent that new relation or concept or they may be many different terms, such as synonyms. This makes the automatic interpretation very difficult, therefore requires a partial human supervision and guidance to interpret and manage new pieces of data semi-automatically.

Example 20: Let us assume that we add the following elements: `Anopheles Merus` *is* exophilic, `Anopheles Melas` *is a* saltwater species, `Anopheles Funestus` *is* strongly anthropophilic *and* `Anopheles Kerteszia` *is* `living in bromeliads`*. The first one is rather easy to interpret as IDOMAL contains a concept for* exophily*. The second one requires more thinking as* saline lake, saline water *and* saline wedge estuary *are imported from the Environment Ontology [62]. A solution would be to consider the addition of* `Anopheles Melas` ***lives_in*** `place` *where* `place` *belongs to one of the three salty concepts instead of adding a concept assertion about* `Anopheles Melas`*. The third is still harder. The concept* anthropophilic *could be used. However, it is a qualitative concept compared to the quantitative* strongly anthropophilic*. Finally, the last one corresponds to no known concept in any domain ontology we use and is thus impossible to accurately interpret.*

D. UTILIZING CHANGE

Up to this point, a change was considered as some sort of obstacle. It is, however, possible to consider change as a natural and integral part of any live and dynamic data source. Indeed, as we store previous results of queries, we can timestamp and store different versions of domain ontologies or databases and use them for ad-hoc and agile querying.

Example 21: Assume we store in the triple-store the result of multiple queries of the form: "What is the weather like in City *today?" where* City *covers a range of various cities. Then we might be interested in a query that gives us the list of all cities in which the weather has been sunny at least one-third of the time during the past week.*

This approach is not, in and of itself, much different from the querying of distributed sources. The main problem is that the size of queried data is much bigger.

E. CHANGE CLASSIFICATION

It is possible to classify changes according to their impact on changing the definition of services and mapping rules. The table in Table 2 lists examples in both categories. Simply put, when the change only affects the data, by adding or removing elements, the change is non-critical and the infrastructure is left unchanged. On the other hand, when the structure of the data source is modified, be it in the domain ontologies or the database schema, the change is critical. The classification presupposes that the concepts and relations used in the non-critical cases existed before the change. If new concepts and relations are introduced, the change is always critical.

TABLE 2. A classification of the examples presented in this paper depending on whether they are critical or not.

Non-critical	Critical
Example III.3	Example III.5: ρ_0
Example III.4	Example IV.4
Example III.5: ρ_1	Example IV.6
Example IV.2	Example IV.7
Example IV.5	Example IV.8

F. USE CASE SCENARIO

We now walk through an example to demonstrate how the various components in the SIEMA infrastructure act together

```
PREFIX idomal: < http://purl.obolibrary.org/obo/idomal.owl#>
SELECT ?name ?indoorFeedingRate
WHERE {
?insect idomal:is_a idomal:Culicidae
?insect idomal:has_indoorFeedingRate ?indoorFeedingRate
?insect idomal:has_species ?name
}
```

FIGURE 10. A SPARQL query that returns the indoor feeding rates and the names of all culicidae.

to detect identify, classify changes and consistently update the data sources accordingly and to facilitate their interoperability. In our example, we run the SPARQL query shown in Fig. 10 before any change occurs. It returns the list of the Culicidae species and their indoor feeding rate. It is run against the vector base table shown in Table 1. We consider IDOMAL as our domain ontology. We assume that there exist rules that map each row in the table to a Culicidae. Two services are defined to make this query possible:

1) One takes a Culicidae (represented by its Id) as input and returns its species (e.g. if the input is 2, it will return `An. Funestus`)
2) The other takes a Culicidae as input and returns an indoor feeding rate (e.g. if the input is 5, it will return `40`).

In both cases, the input is a Culicidae as defined by IDOMAL.

Assume now that the change presented in Example 11 is applied, and the rows of the table are now interpreted as Culicidae according to the definition of MIRO instead of the one of the IDOMAL. The architecture goes through several steps to handle this change.

1) When the query is run again, it returns no result at all as there no longer is any Culicidae according to IDOMAL's definition, all the elements in the table being interpreted as Culicidae according to MIRO's definition.
2) The Change Capture Agent detects that something changed because the result of the query is different.
3) The Consistency Management Agent then checks the consistency of the service ontology with the domain ontologies. No problem is detected because the problem is not that the IDOMAL definition of Culicidae has been removed. The definition is still part of the ontology but it merged with the one from MIRO. The problem is that the concept is not used in the tables anymore.
4) By looking at the change logs, the exact modification is detected.
5) The Service Management Agent asks for the update of all services using the IDOMAL definition of Culicidae to now use the definition from MIRO. The services are now defined in the same way except that the inputs are Culicidae as defined by MIRO.
6) When the services are updated, the query is run again and it now returns the same result as before the change occurred.

V. EVALUATION

As per the functional requirements, a robust surveillance system should be capable of detecting and identifying changes anticipated in its components. In the event of a change in the system, it should be able to provide warnings and advisory messages to prevent the services from malfunctioning or even rebuild and redeploy the services if necessary. The evaluation of SIEMA will be focused on two important tasks: i) how accurately the system can capture and classify the anticipated changes? and ii) how well does the system manage these changes and control their impacts?

The entire system should function efficiently as a single unit and work reliably at all times and under varying conditions.

The reliability of the system depends on the degree of interoperability between its individual components. The number of system failures, as well as false alarms, and misclassification are also good measures of the system's reliability Our evaluation plan for SIEMA is centered over two criteria; first, the accuracy of identifying changes and second, the accuracy of algorithms determining the effects of the changes. The ability to accurately identify different types of change in the system can be evaluated by comparing the information from the log history with the new upcoming data. These changes are generally anticipated in the domain ontologies as well as in the relational databases. At this stage in our experiments, a change can be as simple as the addition of new concepts and/or relations or deletion of existing concepts and/or relations in domain ontologies. Changes in the database could be in the form of a new data type, or even rearranging the order of attributes of a table.

A unified view of these changes would be presented to the user on multiple widgets in a dashboard window [63]. The different algorithms used by the *Change Capture Agent* (CCA), *Consistency Management Agent* (CMA), and *Service Management Agent* (SMA) together will monitor changes and determine the impact of changes on the current services. Therefore, the accuracy of these algorithms needs to be evaluated to ensure that correct interpretation is made for advisory and control decisions. In our preliminary experiments, we discovered that determination of the impact can be more complex when there is a change in the relational database compared to the changes in the domain ontologies. The reason is that the specifications of web services keep both syntactic and semantic relations within the ontologies during all stages of creation, deployment, discovery, composition, and execution while the database is accessed only during the execution phase. A workaround is being considered for determining the impact of the change in a database on the deployed services. This examines the explicit semantic mapping of relational databases and domain ontologies and their relations with the semantics of the deployed services.

VI. DISCUSSION AND CONCLUSION

In this article, we have discussed the importance of change management to maintain interoperability between different

malaria data sources for surveillance purposes. We then introduced SIEMA, a web-based platform that facilitates change management and maintains interoperability between different dynamic malaria data sources.

Considering that both the data and schema evolve over time, we proposed a formal methodology based on graph transformation to detect, and identify changes and illustrated this through a series of examples. There are still some limitations and challenges that need to be addressed. For example, in a dynamic distributed environment it is important to be able to manage asynchronous accesses and changes since not all data repositories are available at all times. A further challenge is the management of heterogeneous data sources consisting of data with various degrees of granularity.

In addition, approaching change from the end-user perspective is far from being the only option. Particularly in the case of structural or ontological changes, evolution is done with a purpose and should not be considered as a hindrance. Being able to improve ontologies and database schemas while maintaining consistency is an important issue that deserves further consideration. There are two aspects to this approach: one is using verification of graph transformations [54] to prove that the changes will yield a knowledge base that meets our expectations and the other one is proposing repair [64] solutions to solve inconsistencies when they are unavoidable. Another important step is to establish a communication channel, for the agents presented in our infrastructure, to facilitate dialectical change [65] management.

As future work, we will develop algorithms to express more complex types of changes [66] as well as concurrent changes in the system. Also, we will be providing the robust implementation of the dashboard that will be used to detect changes. Finally, we will expand our system to ensure language (e.g. French and English) interoperability between different surveillance system implemented in different African countries (e.g. Uganda and Gabon). Besides malaria surveillance, SIEMA can be generalized to maintain interoperability and evolution in other domains as well.

REFERENCES

[1] World Health Organisation. (2015). *The Top 10 Causes of Death*. Accessed: Jun. 28, 2017. [Online]. Available: www.who.int/mediacentre/factsheets/fs310/en/index1.html

[2] *World Malaria Report 2016*, WHO, Geneva, Switzerland, 2016.

[3] S. Awash, B. Spielman, A. Tozan, Y. Schapira, and A. Teklehaimanot, "Coming to grips with malaria in the new millennium: UN millennium project task force on HIV/AIDS," in *Malaria, TB, and Access to Essential Medicines Working Group on Malaria*, 1st ed. Geneva, Switzerland: Earthscan Publications, Ltd., 2005.

[4] C. Bass, M. S. Williamson, C. S. Wilding, M. J. Donnelly, and L. M. Field, "Identification of the main malaria vectors in the Anopheles gambiae species complex using a TaqMan real-time PCR assay," *Malaria J.*, vol. 6, p. 155, Nov. 2007. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2213665/

[5] A. J. McMichael *et al.*, Eds., *Climate Change and Human Health—Risks and Responses*. Geneva, Switzerland: WHO, 2003.

[6] J. Sachs and P. Malaney, "The economic and social burden of malaria," *Nature*, vol. 415, no. 6872, pp. 680–685, 2002. [Online]. Available: http://dx.doi.org/10.1038/415680a

[7] "IPCC third assessment report," Intergovernmental Panel Climate Change, Tech. Rep., 2001.

[8] Office of Public Health Scientific Services (OPHSS). National *Notifiable Diseases Surveillance System (NNDSS), Data Collection and Reporting*. Accessed: Aug. 2, 2017. [Online]. Available: https://wwwn.cdc.gov/nndss/data-collection.html

[9] J. Liu, B. Yang, W. K. Cheung, and G. Yang, "Malaria transmission modelling: A network perspective," *Infectious Diseases Poverty*, vol. 1, no. 1, p. 11, Nov. 2012. [Online]. Available: http://dx.doi.org/10.1186/2049-9957-1-11

[10] World Health Organization. *Disease Surveillance for Malaria Control: An Operational Manual 2012*. Accessed: Sep. 14, 2017. [Online]. Available: https://apps.who.int/iris/bitstream/10665/44851/1/9789241503341_eng.pdf?ua=1

[11] K. Zinszer *et al.*, "Integrated disease surveillance to reduce data fragmentation—An application to malaria control," *Online J. Public Health Informat.*, vol. 7, no. 1, p. e181, 2015. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4512354/

[12] *Africa Health Observatory and real-time Strategic Information System*. Accessed: Jun. 27, 2017. [Online]. Available: http://www.afro.who.int/en/ghana/country-programmes/4989-aho-a-rsis.html

[13] RTI International. *Coconut Surveillance: Open-Source Mobile Tool Designed to Take on Malaria, Other Infectious Diseases*. Accessed: Sep. 14, 2017. [Online]. Available: https://www.rti.org/impact/coconut-surveillance

[14] K. Eeshan, *Performance Evaluation of Zanzibar's Malaria Case Notification (MCN) System: The Assessment of Timeliness and Stakeholder Interaction*. Accessed: Sep. 14, 2017. [Online]. Available: https://dukespace.lib.duke.edu/dspace/handle/10161/10026

[15] M. S. Hsiang *et al.*, "Surveillance for malaria elimination in Swaziland: A national cross-sectional study using pooled PCR and serology," *PLoS ONE*, vol. 7, no. 1, p. e29550, 2012.

[16] J.-O. Guintran, C. Delacollette, and P. Trigg, "Systems for the early detection of malaria epidemics in Africa: An analysis of current practices and future priorities," World Health Org., Geneva, Switzerland, Tech. Rep. WHO/HTM/MAL/2006.1115, 2006.

[17] C. Ohrt, K. W. Roberts, H. J. W. Sturrock, J. Wegbreit, B. Y. Lee, and R. D. Gosling, "Information systems to support surveillance for malaria elimination.," *Amer. J. Tropical Med. Hygiene*, vol. 93, no. 1, pp. 145–152, Jul. 2015.

[18] D. Le Sueur *et al.*, "An atlas of malaria in Africa," *Africa Health*, vol. 19, no. 2, pp. 23–24, Jan. 1997.

[19] *Mapping mAlaria Risk in Africa*. Accessed: Jul. 2, 2017. [Online]. Available: http://www.mara-database.org//login.html

[20] *VecNet*. Accessed: Jul. 2, 2017. [Online]. Available: https://dw.vecnet.org/datawarehouse/lookuptables/

[21] World Health Organisation. (2013). *Global Malaria Mapper*. Accessed: Jun. 27, 2017. [Online]. Available: www.who.int/malaria/publications/world_malaria_report/global_malaria_mapper/en/

[22] *Global Malaria Mapper*. Accessed: Jul. 2, 2017. [Online]. Available: http://www.worldmalariareport.org/

[23] S. I. Hay and R. W. Snow, "The malaria atlas project: Developing global maps of malaria risk," *PLoS Med.*, vol. 3, no. 12, pp. 1–5, 12 2006. [Online]. Available: https://doi.org/10.1371/journal.pmed.0030473

[24] *The Malaria Atlas Project*. Accessed: Jul. 2, 2017. [Online]. Available: http://www.map.ox.ac.uk/

[25] *The USAID Measure DHS Website, Data Downloads*. Accessed: Jun. 27, 2017. [Online]. Available: http://www.measuredhs.com/Data/

[26] D. Lawson *et al.*, "VectorBase: A home for invertebrate vectors of human pathogens," *Nucl. Acids Res*, vol. 35, p. D503-5, Jan. 2007.

[27] *VectorBase*. Accessed: Jun. 2, 2017. [Online]. Available: https://www.vectorbase.org/

[28] *The DHIS 2 Web Site*. Accessed: Jun. 27, 2017. [Online]. Available: https://www.dhis2.org/

[29] S. Lozano-Fuentes, A. Bandyopadhyay, L. G. Cowell, A. Goldfain, and L. Eisen, "Ontology for vector surveillance and management.," *J. Med. Entomol.*, vol. 50, no. 1, pp. 1–14, Jan. 2013.

[30] P. Topalis, E. Mitraka, V. Dritsou, E. Dialynas, and C. Louis, "IDOMAL: The malaria ontology revisited," *J. Biomed. Semantics*, vol. 4, no. 1, p. 16, Sep. 2013. [Online]. Available: http://dx.doi.org/10.1186/2041-1480-4-16

[31] *Malaria Ontology*. Accessed: Jul. 2, 2017. [Online]. Available: https://bioportal.bioontology.org/ontologies/IDOMAL

[32] E. Dialynas, P. Topalis, J. Vontas, and C. Louis, "MIRO and IRbase: IT tools for the epidemiological monitoring of insecticide resistance in mosquito disease vectors," *PLoS Neglected Tropical Diseases*, vol. 3, no. 6, pp. 1–9, 2009. [Online]. Available: https://doi.org/10.1371/journal.pntd.0000465
[33] *Mosquito Insecticide Resistance Ontology*. Accessed: Jul. 2, 2017. [Online]. Available: https://bioportal.bioontology.org/ontologies/MIRO
[34] C. C. Freifeld, K. D. Mandl, B. Y. Reis, and J. S. Brownstein, "HealthMap: Global infectious disease monitoring through automated classification and visualization of Internet media reports," *J. Amer. Med. Inf. Assoc.*, vol. 15, no. 2, pp. 150–157, Mar./Apr. 2008.
[35] *HealthMap*. Accessed: Jul. 2, 2017. [Online]. Available: https://www.healthmap.org/
[36] World Wide Web Consortium. *The World Wide Web Consortium (W3C)*. Accessed: Jul. 2, 2017. [Online]. Available: https://www.w3.org
[37] *HyperText Transfer Protocol*. Accessed: Jul. 2, 2017. [Online]. Available: https://www.w3.org/Protocols/
[38] *Ressource Description Framework*. Accessed: Jul. 2, 2017. [Online]. Available: https://www.w3.org/2001/sw/wiki/RDF
[39] *Web Ontology Language*. Accessed: Jul. 2, 2017. [Online]. Available: https://www.w3.org/2001/sw/wiki/OWL
[40] *SPARQL Query Language for RDF*. Accessed: Jul. 2, 2017. [Online]. Available: https://www.w3.org/TR/rdf-sparql-query/
[41] M. D. Wilkinson, B. Vandervalk, and L. McCarthy, "The semantic automated discovery and integration (SADI) Web service design-pattern, API and reference implementation," *J. Biomed. Semantics*, vol. 2, no. 1, p. 8, 2011. [Online]. Available: http://dx.doi.org/10.1186/2041-1480-2-8
[42] B. P. Vandervalk, E. L. McCarthy, and M. D. Wilkinson, *SHARE: A Semantic Web Query Engine for Bioinformatics*. Berlin, Germany: Springer, 2009, pp. 367–369. [Online]. Available: https://doi.org/10.1007/978-3-642-10871-6_27
[43] *HYDRA*. Accessed: Oct. 1, 2017. [Online]. Available: http://ipsnp.com/hydra/
[44] A. Riazanov *et al.*, "Semantic querying of relational data for clinical intelligence: A semantic Web services-based approach," *J. Biomed. Semantics*, vol. 4, no. 1, p. 9, Mar. 2013.
[45] K. Wolstencroft *et al.*, "The ^{my}Grid ontology: bioinformatics service discovery," *Int. J. Bioinf. Res. Appl.*, vol. 3, no. 3, pp. 303–325, 2007. [Online]. Available: https://doi.org/10.1504/IJBRA.2007.015005
[46] H. Boley and M. Kifer. *RIF Basic Logic Dialect (Second Edition)*. Accessed: Jul. 2, 2017. [Online]. Available: https://www.w3.org/TR/2013/REC-rif-bld-20130205/
[47] H. Boley, *A RIF-Style Semantics for RuleML-Integrated Positional-Slotted, Object-Applicative Rules*. Berlin, Germany: Springer, 2011, pp. 194–211. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-22546-8_16
[48] V. Haarslev, K. Hidde, R. Möller, and M. Wessel, "The RacerPro knowledge representation and reasoning system," *Semantic Web J.*, vol. 3, no. 3, pp. 267–277, 2012.
[49] M. A. Musen, "The Protégé project: A look back and a look forward," *AI Matters*, vol. 1, no. 4, pp. 4–12, 2015. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4883684/
[50] M. S. Al Manir, A. Riazanov, H. Boley, A. Klein, and C. J. O. Baker, "Valet SADI: Provisioning SADI Web services for semantic querying of relational databases," in *Proc. 20th Int. Database Eng. Appl. Symp.*, New York, NY, USA, 2016, pp. 248–255. [Online]. Available: http://doi.acm.org/10.1145/2938503.2938543
[51] R. Heckel, "Graph transformation in a nutshell," *Electron. Notes Theor. Comput. Sci.*, vol. 148, no. 1, pp. 187–198, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S157106610600048X
[52] M. Minas and H. J. Schneider, *Graph Transformation by Computational Category Theory*. Berlin, Germany: Springer, 2010, pp. 33–58. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-17322-6_3
[53] R. Echahed, "Inductively sequential term-graph rewrite systems," in *Proc. 4th Int. Conf. Graph Transf. (ICGT)*, vol. 5214. 2008, pp. 84–98.
[54] J. H. Brenas, R. Echahed, and M. Strecker, "Proving correctness of logically decorated graph rewriting systems," in *Proc. 1st Int. Conf. Formal Struct. Comput. Deduction (FSCD)*, Jun. 2016, pp. 14:1–14:15. [Online]. Available: http://dx.doi.org/10.4230/LIPIcs.FSCD.2016.14
[55] R. De Virgilio, A. Maccioni, and R. Torlone, "Converting relational to graph databases," in *Proc. 1st Int. Workshop Graph Data Manage. Exper. Syst.*, New York, NY, USA, 2013, pp. 1:1–1:6. [Online]. Available: http://doi.acm.org/10.1145/2484425.2484426
[56] J. H. Brenas, R. Echahed, and M. Strecker, "C2PDLS: A combination of combinatory and converse PDL with substitutions," in *Proc. SCSS*, Gammarth, Tunisia, 2017, pp. 29–41. [Online]. Available: http://www.easychair.org/publications/paper/C2PDLS_A_Combination_of_Combinatory_and_Converse_PDL_with_Substitutions
[57] C. Areces and B. ten Cate, "Hybrid logics," in *Studies in Logic and Practical Reasoning*, vol. 3. Amsterdam, The Netherlands: Elsevier, 2007, pp. 821–868.
[58] B. Courcelle, "The monadic second-order logic of graphs. I. Recognizable sets of finite graphs," *Inf. Comput.*, vol. 85, no. 1, pp. 12–75, 1990. [Online]. Available: http://dx.doi.org/10.1016/0890-5401(90)90043-H
[59] P. Balbiani, R. Echahed, and A. Herzig, "A dynamic logic for termgraph rewriting," in *Proc. 5th Int. Conf. Graph Transf. (ICGT)*, vol. 6372. 2010, pp. 59–74.
[60] J. H. Brenas, R. Echahed, and M. Strecker, "Ensuring correctness of model transformations while remaining decidable," in *Proc. 13th Int. Colloquium Theor. Aspects Comput. (ICTAC)*, 2016, pp. 315–332. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46750-4_18
[61] *VectorBase CV*. Accessed: Jul. 2, 2017. [Online]. Available: www.vectorbase.org/downloadinfo/ontologyvbcv0162017-06obogz
[62] *ENVironment Ontology*. Accessed: Jul. 2, 2017. [Online]. Available: https://bioportal.bioontology.org/ontologies/ENVO
[63] J. H. Brenas, M. S. Al-Manir, C. J. O. Baker, and A. Shaban-Nejad, "Change management dashboard for the SIEMA global surveillance infrastructure, " in *Proc. Int. Semantic Web Conf.*, 2017, pp. 1–4.
[64] M. Bienvenu, C. Bourgaux, and F. Goasdoué, "Query-driven repairing of inconsistent DL-lite knowledge bases (extended abstract)," in *Proc. 29th Int. Workshop Description Logics*, Apr. 2016, pp. 1–4. [Online]. Available: http://ceur-ws.org/Vol-1577/paper_5.pdf
[65] K. J. Holsti, *The Problem of Change in International Relations Theory*. Vancouver, BC, Canada: Univ. British Columbia, 1998.
[66] A. Shaban-Nejad and V. Haarslev, *Bio-Medical Ontologies Maintenance and Change Management*. Berlin, Germany: Springer, 2009, pp. 143–168. [Online]. Available: https://doi.org/10.1007/978-3-642-02193-0_6

JON HAËL BRENAS received the double master's degree in mathematical modeling and digital imagery specialized in modeling, computation, and simulation from ENSIMAG, Grenoble, France, and in computer engineering from the Politecnico di Torino, Torino, Italy, and the Ph.D. degree from the University Grenoble-Alps, Grenoble, France. His doctoral dissertation focus was on Verification of Graph Transformation. He is currently a Post-Doctoral Fellow with the Oak Ridge National Laboratory, Center for Biomedical Informatics, Department of Pediatrics, University of Tennessee Health Science Center, Memphis, TN, USA. His research interests lie in graphs and graph transformations application to biomedical and health informatics.

MOHAMMAD SADNAN AL-MANIR received the M.Sc. degree in European Masters Program in Computational Logic from the Vienna University of Technology, Austria, and the Free University of Bozen-Bolzano, Italy. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Applied Statistics, University of New Brunswick, Saint John, Canada. Some of his recent work include but not limited to data federation in agriculture, text-mining pipeline for scientific literatures in Canadian Rivers Institute, API for Positional-Slotted, Object-Applicative RuleML, and automatic generation of semantic web services. His research interests involve semantic technologies, text-mining, semantic web services, and rule languages.

CHRISTOPHER J. O. BAKER received the Ph.D. degree. He is currently a Full Professor and the Chair with the Department of Computer Science, University of New Brunswick, Saint John, Canada. He has core expertise in Open Data Integration and Interoperability and serves on the advisory board of the Canadian Institute for Cybersecurity. In 2016, he was a finalist for the Canadian Open Data Leader of the year and was invited as a Speaker at the Annual Meeting of Agricultural Chief Scientists of G20 States (MACS-G20) on Linked Open Data in Agriculture in 2017.

ARASH SHABAN-NEJAD received the M.Sc. and Ph.D. degrees in computer science from Concordia University, Montreal, and the M.P.H. degree from the University of California at Berkeley. He is currently an Assistant Professor with the OAK-Ridge National Laboratory, Center for Biomedical Informatics, Department of Pediatrics, University of Tennessee Health Science Center, Memphis, TN, USA. He was a Post-Doctoral Fellow of the McGill Clinical and Health Informatics Group, McGill University. Additional training was accrued at the Harvard School of Public Health. His primary research interest is Clinical and Population Health Intelligence, Epidemiologic Surveillance, and Big-Data Semantic Analytics using tools and techniques from Artificial Intelligence, Knowledge Representation, and Semantic Web. He is the Principal Investigator of a Global Health and Development Research Project for malaria elimination, funded by the Bill & Melinda Gates Foundation.