

Applying Semantic Web Techniques to Reservoir Engineering: Challenges and Experiences from Event Modeling

Tao Zhu*, Amol Bakshi[†], Viktor K. Prasanna[‡], and Karthik Gomadam*

*Dept. of Computer Science, University of Southern California, Los Angeles, CA 90089

Email: {taozhu,gomadam}@usc.edu

[†]Ming Hsieh Dept. of Electrical Engineering, University of Southern California, Los Angeles, CA 90089

Email: prasanna@usc.edu

[‡]Chevron Corporation, Houston, TX 77002

Email: Amol.Bakshi@chevron.com

Abstract—In reservoir engineering domain experts deal with many tasks and operations, ranging from reservoir simulation to well maintenance scheduling, with the goal of maximizing oil production. These operations involve different applications, and generate a large number of events at the time of their execution. Understanding these events and the relationships among them can aid the domain experts in decision making. The task of aggregating and analyzing the event information that is generated by different applications can be difficult and time consuming. In this paper, we discuss our approach on modeling events in reservoir engineering using Semantic Web techniques. We create an extensible event ontology using the Web Ontology Language (OWL) to model events in reservoir engineering. Event relationships are encoded using the Semantic Web Rule Language (SWRL). The work presented in this paper is an initial step towards a vision of integrated field event management, that uses semantic web technologies to create an efficient platform for event analysis.

Keywords-Event Correlation, Event Ontology, Integrated Asset Management, Oilfield Industry, Rule Based Reasoning.

I. INTRODUCTION

Reservoir engineering is a branch of petroleum engineering that aims to maximize the economic recovery of hydrocarbons from the subsurface. Information technologies have been applied to reservoir engineering to improve the efficiency of the different reservoir engineering tasks and processes. The IAM project (Integrated Asset Management [1] [2]) integrates information and application of various tasks in reservoir simulation and real time surveillance to enable fast and efficient decision making.

The ability to identify and analyze events is central to realizing robust real time decision making capabilities. This is because, reservoir engineering involves various tasks such as reservoir simulation, reservoir surveillance, well testing, and analysis of reservoir fluids. The execution of these tasks generates a number of events which record key operations and dynamic status of resources (wells, field equipments, etc.) in a reservoir. Examples of these events include starting or shutting in a well and important changes to well temperature or pressure. More often than not, events that occur at a

resource may have potentially effects across other resources in a reservoir. For example, water being injected into an injection well may travel under the ground and can affect other wells in the reservoir. Reviewing event information occurring at different locations and understanding their possible causes and potential impacts can aid in better decision making. For example, a domain expert who knows that the oil production of a well declines dramatically because of a pump exception can make a decision to either repair or replace the pump. In another scenario, if the oil production of a well reduces because of pressure depletion, the solution might be to inject water into an injection well so as to improve the production.

Addressing the problem of identifying and reacting to events that occur across an asset requires the ability to aggregate and understand the events that have been detected. Event information comes from multiple sources and is stored in heterogeneous formats, including historical databases which store data collected from the reservoir, legacy files (MS Excel files, Spreadsheets) used by domain experts to report field operations, and logs generated by software applications. This heterogeneity in event representation poses a challenge towards event aggregation and understanding. Currently, this challenge is addressed by a time consuming process involving a group of domain experts. This manual approach presents a roadblock towards realizing efficient real-time decision making abilities.

One approach to address the challenges posed by heterogeneity, is to create a uniform model for event description. In this paper, we discuss our approach to model events in the domain of reservoir engineering using Semantic Web techniques. Semantic Web is “an extension of the current web in which information is given well-defined meaning, enabling computers and people to work in cooperation” [3]. Semantic Web techniques are a set of languages and standards proposed to accomplish this vision. The Web Ontology Language (OWL) [4] is a family of languages for authoring ontologies, which are a formal representation of domain concepts and their relationships. The Semantic

Web Rule Language (SWRL) [5] is a proposed standard for creating rules based on ontologies.

The main contributions of this paper include:

- We design an OWL ontology for representing events in reservoir engineering, which describes the major types of events in reservoir engineering, features of these events, and their relationships between each other.
- We show that rules defined using SWRL based on our event ontology can represent meaningful domain knowledge about event relationships. The rules encode logic of discovering implicit event instances or relations between events from existing events and domain status.

The work presented in this paper is an initial step towards a vision of integrated field event management. The objective of integrated field event management is to facilitate efficient event analysis by enabling the sharing of event information among different organizations and people. The event information captured from various sources can be used to populate event instances based on the event ontology. Rules of event correlation defined by domain experts can be applied to these event instances to infer implicit event instances and relationships between events. Event information can be correlated better and can be made more accessible.

The rest of the paper is organized as follows: in section II we summarize the related work. In section III we describe our event ontology. Examples of rules defined using SWRL are introduced in section IV. We present the conclusions of this work and discuss our future work in section V.

II. RELATED WORK

Feigenbaum et. al [6] present a discussion of the application of semantic web technologies to data interchange in the oilfield industry. AKSIO [7] is an oilfield knowledge management project which uses RDF and OWL for semantic annotation of experiences and supports contextual ontology driven retrieval of content. SmartWellOnto [8] is a preliminary version of an ontology for smart wells, which is expected to be used to build a knowledge base system for optimization of the oil/gas reservoir. It describes the relationships between field entities (well, reservoir, completion, etc.). Existing work in the IAM project [9] also includes the design of a domain ontology for field entities in a reservoir. While these work mainly focuses on modeling of domain entities and their relationships, we are working on modeling the events in the domain.

Semantic Web technologies have been used to define event models both at general and domain-specific levels. Upper level ontologies containing events description have been proposed in [10] and [11]. These models provide a high level abstraction of events, while more detailed ontologies have been proposed for specific domains. For instance, in bioinformatics, an event ontology has been designed to support detection of infectious disease outbreaks from the analysis of natural language text [12]. Ontologies for event definition

have been defined for soccer [13] and broadcast news [14] to enhance video annotation and retrieval capabilities exploiting temporal and spatial relationships among events occurring in a video. In these two approaches Semantic Web technologies have been used for their expressiveness and for reasoning capabilities, which allow automatic specification of events categories.

Our event model belongs to domain-specific ontologies. It has been designed to facilitate event representation and analysis in reservoir engineering. To the best of our knowledge, our work is the first attempt to define an event ontology in this domain.

Event correlation is an important topic in several research fields such as network management and event-based systems. Various strategies have been developed to encode knowledge about event relationships and leverage it to correlate events efficiently. For example, S. Yemini et al. [15] propose an approach based on hamming code to correlate the observed events in network transactions to identify and localize underlying problems. They encode knowledge of event relationships using hamming code, which is not intuitive or flexible. Researchers in network management have also tried using rule based reasoning to perform event correlation. Most of these approaches use non-standard rule formats that are not compatible with semantic web standards.

III. EVENT MODELING

In this section we discuss our approach for creating an event model for the domain of reservoir engineering using OWL. Our design philosophy is based on the two rules of thumb for creating semantic models:

- **Reusability:** The ontology should be reusable by various applications of event management in reservoir engineering, rather than being restricted to a specific use case.
- **Extensibility:** The ontology must allow users to create newer domain models by extending a core subset.

To ensure these features, we design our event ontology in a way which is 1) modular and 2) using a two-layered framework. In the following, we introduce the main classes and properties defined in the event ontology and provide examples to illustrate them, instead of enumerating all classes and properties we have defined in the ontology.

A. Core Event Ontology

We build our event ontology in a modular way. We import the OWL-Time ontology [16] into our event ontology to describe the temporal feature of an event. The OWL-Time ontology defines a set of temporal concepts, such as a time point, a time interval, etc.,. OWL-Time also describes relationships between different temporal instances, such as “before” and “after”. We define two relationships between events and field entities: 1) *associated field objects that an event occurs at* and 2) *relevant field objects of an event* and

import their definitions from an existing domain ontology [9]. This ontology describes elements of the oilfield and their relationships. The modular nature of our model helps in ontology maintenance. The ability to extend a part of the model based on requirements is another advantage of the modular design.

To make the event ontology extensible, we design the ontology to be a two layered framework. The core event ontology has been designed using general events concepts and properties. This core model is extended with hierarchies of domain specific events (Figure 1)(only part of the classes and properties are shown).

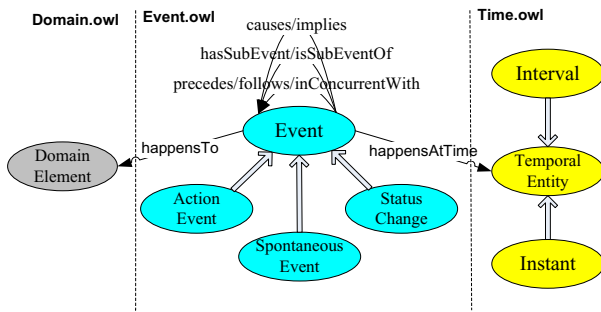


Figure 1. The Core Event Ontology

As shown in Figure 1, the property “happensAtTime” is used to describe the temporal feature of an event, whose range is the “TemporalEntity” class defined in the Time ontology. The time of occurrence of an event can be either a time point (“Instant”) or a time period (“Interval”), depending on the nature of the event and the way it is described. For example, we can model an instant production rate decline can have a time point as the “TemporalEntity”, while a gradual decline in production can have a time period as the “TemporalEntity”. The property “happensTo” describes the spatial location of the event (wells, surface facilities, reservoirs, etc.). For example, the “happensTo” property of the “WellShutIn” event would be the well that was shut-in. The range of the “happensTo” property is the “DomainElement” class defined in the domain ontology.

We define three kinds of event relationships in the core model.

- 1) Sequential relationships (*precedes / follows / isConcurrentWith*),
- 2) Organizational relationships (*isSubEventOf / hasSubEvent*), and
- 3) Cause-and-effect relationships (*causes, implies*)

We make an observation here that the sequential relationships between events are not necessarily the same as the temporal relationships defined in the OWL-Time ontology. Sequential relationships also contain logical dependencies between events, such as the execution of two consecutive tasks in a workflow. If event *A* precedes event *B*, then *A* must happen before *B*, but not vice-versa (i.e. if *A*

happens before *B*, then *A* precedes *B*). For example, a well “W1” starts producing before another well “W2” while “W1” and “W2” are in different reservoirs. These two events are independent (or to our limited knowledge of the asset are independent) and do not have sequential relationships between each other.

B. Modeling of Domain Specific Event Types

We categorize the events in reservoir engineering into four major types - *action events, status change, spontaneous events and composite events*. We model these types of events using OWL classes and properties.

Action Events are operations performed by people in different roles (facility operators, production engineers, etc.) or controlled by certain applications. For example, a well was shut-in by production engineers, water was injected into a well by facility operators, etc.

The “ActionEvent” class inherits the following two properties from the “Event” class: 1) “happensAtTime” to describe the time it was performed, and 2) “happensTo”, to describe the field object that this operation is performed to. The “happensTo” is classified into different sub-properties specific to different sub-classes of “ActionEvent” (“happensToWell”, “happensToCompletion”, etc.). For example, a “wellStartsProducing” event should happen to a well.

Status Change describes the transition of a domain status from one state to another. A domain status is usually represented by a set of parameters, including pressure on different equipments (tubing, casing, separator, etc.), temperatures, water/oil/gas production of wells, gas oil ratio, etc.. The parameter values measured from the field contain useful information about the surface facilities and subsurface components. For example, the gas oil ratio (GOR) represents the number of cubic feet of natural gas produced with a barrel of oil ([17]). In some situations, the GOR value tells the pressure in a reservoir, which is a major force in moving the oil to the boreholes in the wells. A high GOR value indicates a possible depletion of the pressure, which means a great percentage of the oil may not be recoverable.

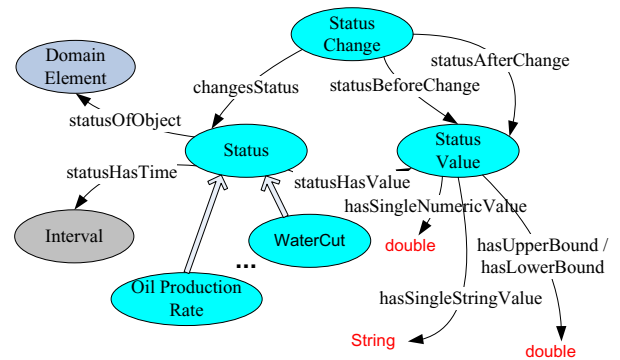


Figure 2. StatusChange

A domain status (*< type, value >*) is described using two

classes, the “Status” class representing the type / category of the status and the “StatusValue” class representing the value, which are associated using the property “statusHasValue”. These two classes are further classified into sub-classes to represent different statuses. A domain status is associated with specific domain objects using the property “statusOfObject”.

Another important property of the status concept is its temporal feature. A status is valid only for a period of time. We use a property “statusHasTime” pointing to a time interval defined by the OWL-Time ontology to represent the status’ valid time.

The “StatusChange” event class has a property “changesStatus” whose range is the “Status” class. This property describes the type of status that has been changed. The “StatusChange” event also has a property pair “statusBeforeChange” and “statusAfterChange”. The range of both the properties is the “StatusValue” class and they denote what change has occurred to the status.

Spontaneous Events describe natural phenomena that occurs without being *directly* controlled by human or programs. For example, the “WaterBreakthrough” event means an excessive amount of water, which previously is isolated or separated from production, gains access to a producing well bore ([18] [17]). Water breakthrough is a spontaneous process occurring in the subsurface of the reservoir, though it might be initiated or caused by some manual operations, such as injecting water into a well.

An important subclass of “SpontaneousEvents” is “Exception” events. “Exception” events are unexpected errors or problems occurring in the domain. An exception can either be observed and reported by certain surveillance tools, or inferred by a set of symptoms. Examples of exceptions include “a well was shut down unexpectedly”, “the gas oil ratio of a producing well exceeds expected range”, etc.

Composite Events describes events that involve multiple other events of the above types. A composite event represents the situation that *multiple correlated* events together mean a single event in a larger scope. On the other hand, the concept of composite event allows one to describe an event in a finer granularity, by breaking it down into its constituent events. In this sense, every event can be a composite event when it is to be analyzed in a more detailed granularity.

An example of a composite event in the domain of reservoir engineering is the “workover” event (Figure 3). “Workover” is the process of performing major maintenance or remedial treatments on an oil or gas well, for the purpose of restoring, prolonging or enhancing the production of hydrocarbons ([17]). A “workover” process usually consists of several operations performed in a specific sequence, including shutting-in a well, then replacing or closing certain subsurface components (tubing or completions), and finally restarting the well to produce.

We define the property “hasSubEvent” of the “Event”

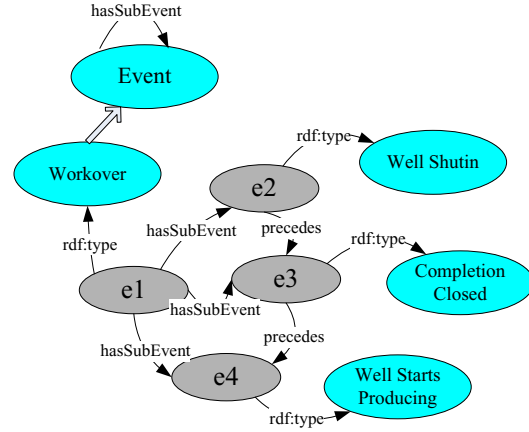


Figure 3. Example of Composite Events

class to represent its subsumption relationships with its sub-events. So every event instance that has the “hasSubEvent” property is a composite event. The sub-events may be of different types and may have temporal/logical relationships between each other.

IV. RULES FOR EVENT CORRELATION

In this section we describe techniques for encoding relationships using SWRL rules.

A. Overview of Rule Based Event Correlation

Using rules to encode relationships enables automatic event correlation. SWRL (Semantic Web Rule Language) is grounded in horn logic. SWRL is a proposed specification of rule languages for semantic web. In examples discussed in this section, the predicates in unary terms (e.g. *WellShutIn* in *WellShutIn(?e)*) represents classes defined in the ontology. Predicates (e.g. *happensTo*) in the binary terms (e.g. *happensTo(?e,?w)*) represents the properties. The prefix represents the namespace of the predicates, i.e. the ontology in which the class or property is defined.

We first introduce the main tasks of event analysis. Later in the section, we illustrate the technique for encoding domain knowledge using rules using examples. A rule can be used for multiple tasks and we describe the association between rules and tasks.

The main tasks involved in event analysis for reservoir engineering are:

- **Event grouping (Rule Example 4):** group a set of events that together represent a composite event in semantic level.
- **Interpreted event detection (Rule Example 3):** discover events that are not explicitly recorded by interpreting a set of domain status and events.
- **Cause-and-effect relationship discovery (Rule Example 1, 2, 3):** discover the causal relationship between events.

B. Rule Examples

Rule Example 1: Pressure Decline

event : *WellPressureDecline*(?e1) \wedge
event : *OilProductionRateDecline*(?e2) \wedge
event : *happensTo*(?e1, ?w) \wedge
event : *happensTo*(?e2, ?w) \wedge
domain : *wellRecoveryStrategy*(?w, 'PRIMARY') \wedge
event : *happenAtTime*(?e1, ?t1) \wedge
event : *happenAtTime*(?e2, ?t2) \wedge
time : *before*(?t1, ?t2) \Rightarrow *event* : *causes*(?e1, ?e2)

Example 1 encodes logic pertaining to the recovery stages of a well. If a well is its primary recovery stage, decline of well pressure is the primary factor that causes a decline in production rate. However, this may not be the case in case of a secondary recovery state.

Rule Example 2: Water Injection

event : *WaterInjection*(?e1) \wedge
event : *WaterBreakthrough*(?e2) \wedge
event : *happensTo*(?e1, ?w1) \wedge
event : *happensTo*(?e2, ?w2) \wedge
domain : *wellBelongsToWellGroup*(?w1, ?g) \wedge
domain : *wellBelongsToWellGroup*(?w2, ?g) \wedge
event : *waterCutoffWell*(?wc, ?w2) \wedge
event : *statusHasValue*(?wc, ?value) \wedge
event : *hasSingleNumericValue*(?value, ?v) \wedge
GreaterThan(?v, MAXWaterCut) \wedge
event : *happenAtTime*(?e1, ?t1) \wedge
event : *happenAtTime*(?e2, ?t2) \wedge
time : *before*(?t1, ?t2) \Rightarrow *event* : *causes*(?e1, ?e2) \wedge
event : *WaterCutExceedsLimit*(?e3) \wedge
event : *happensTo*(?e3, ?w2) \wedge *event* : *implies*(?e3, ?e2)

Rule example 2 establishes causal relationship between a water injection and a water breakthrough event. As the reservoir pressure depletes, water may be injected to a well to recover more fluids (oil or gas) from the reservoir. The injected water will travel in the subsurface, and may break through the barriers between formations to enter a production well. This condition is described as a water breakthrough ([18] [17]). A possible effect of water breakthrough is the amount of water produced from the well rises dramatically and exceeds the water cut limit. As described in the rule, if a water breakthrough is detected in a production well after water is injected into an injection well, and if the two wells belong to the same well group (which means there is a possibility for the water to travel from the injection well to the production well), the water injection is the cause of the water breakthrough. Water produced in excess of the measured water cut limit is an effect that implies the occurrence of a water breakthrough.

Rule Example 3: Pump Exception

event : *PumpEfficiencyFall*(?e1) \wedge
event : *OilProductionRateDecline*(?e2) \wedge
event : *happensTo*(?e1, ?p) \wedge

event : *happensTo*(?e2, ?w) \wedge
domain : *wellHasPump*(?w, ?p) \wedge
event : *WORofWell*(?wor, ?w) \wedge
event : *statusHasValue*(?wor, ?value) \wedge
event : *hasSingleNumericValue*(?value, ?v) \wedge
LessThan(?v, MAXWOR) \wedge
event : *measurementQuality*(?wor, 'HIGH') \wedge
event : *happenAtTime*(?e1, ?t1) \wedge
event : *happenAtTime*(?e2, ?t2) \wedge
time : *before*(?t1, ?t2) \Rightarrow
event : *PumpException*(?e5) \wedge
event : *happenTo*(?e5, ?p) \wedge *event* : *causes*(?e1, ?e2)

In exception driven surveillance, exceptions are interpreted from a set of dynamic domain status and recent action events. Rules can be defined to represent the logic for detecting exceptions based on our ontology.

Rule example 3 illustrates the detection of a pump exception. A decline in oil production is a possible effect of reduction in pump efficiency. In a situation where the water to oil ratio is normal and the measurement accuracy of well parameter values is high, we can relate the decline in production to a reduction in pump efficiency.

Rule Example 4: Workover

event : *WellPlannedAsWorkoverCandidate*(?e1) \wedge
event : *WellShutin*(?e2) \wedge
event : *CompletionClosed*(?e3) \wedge
event : *WellStartsProducing*(?e4) \wedge
event : *happensTo*(?e1, ?w) \wedge
event : *happensTo*(?e2, ?w) \wedge
event : *happensTo*(?e3, ?c) \wedge
domain : *wellHasCompletion*(?w, ?c) \wedge
event : *happenAtTime*(?e1, ?t1) \wedge
event : *happenAtTime*(?e2, ?t2) \wedge
event : *happenAtTime*(?e3, ?t3) \wedge
event : *happenAtTime*(?e4, ?t4) \wedge
time : *before*(?t1, ?t2) \wedge
time : *before*(?t2, ?t3) \wedge
time : *before*(?t3, ?t4) \Rightarrow *event* : *Workover*(?e5) \wedge
event : *happenTo*(?e5, ?w) \wedge
event : *hasSubEvent*(?e5, ?e2) \wedge
event : *hasSubEvent*(?e5, ?e3) \wedge
event : *hasSubEvent*(?e5, ?e4) \wedge
event : *precedes*(?e2, ?e3) \wedge
event : *precedes*(?e3, ?e4) \wedge
event : *precedes*(?e1, ?e5)

Example 4 illustrates the representation of a set of well maintenance events, into a single group called a “workover” event. Workovers are expensive and are usually planned in advance. A well in which a workover is to be performed, is identified as a workover well. The workover process involves shutting-in the well, performing the maintenance tasks and restarting the well again.

The maintenance tasks performed during the workover process may vary between different well conditions and

workover plans. In rule example 4, we consider the situation of performing maintenance by closing a completion of the well. Actually multiple rules can be created for the different alternatives available for maintenance tasks in the workover process.

V. CONCLUSION AND FUTURE WORK

In this paper we described our approach to define an event ontology for the reservoir engineering domain. Based on the event ontology, we demonstrated an approach to use rules defined using SWRL to encode domain knowledge about event relationships. The event model and the rules can be used in achieving automatic event correlation. We also observe here that the rules we express in the domain of reservoir engineering, do not require non-monotonicity or negation as failure, the two commonly observed shortcomings of SWRL.

We are currently developing a prototype system that captures event information from various legacy files to populate event instances. Rules will be defined and applied to these instances using a rule based reasoner for automatic event correlation.

Our future work includes the design and development of uniform interfaces for event publishing to accommodate different event detection strategies, user friendly interfaces for domain experts to create and edit rules, and interfaces for querying and visualization of event information.

ACKNOWLEDGMENTS

This research was funded by CiSoft, (Center for Interactive Smart Oilfield Technologies), a Center of Research Excellence and Academic Training and a joint venture between the University of Southern California and Chevron. We thank the researchers in Chevron for sharing valuable domain knowledge. We also would like to thank Ramakrishna Soma for his prior work on the design of the domain ontology and general guidance on applying semantic web technologies to reservoir engineering.

REFERENCES

- [1] R. Soma, A. Bakshi, A. Orangi, V. K. Prasanna, and W. Da Sie, "A service-oriented data-composition architecture for integrated asset management," in *Intelligent Energy Conference and Exhibition*, Amsterdam, Netherlands, 2006.
- [2] T. Unneland and M. Hauser, "Real-Time asset management: From vision to engagement - an operator's experience," in *SPE Annual Technical Conference and Exhibition*, Dallas, Texas, USA, 2005.
- [3] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific American*, vol. 284, no. 5, pp. 34–43, 2001.
- [4] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein, "Owl web ontology language reference," <http://www.w3.org/TR/owl-ref/>, February 2004. [Online]. Available: <http://www.w3.org/TR/owl-ref/>
- [5] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosz, and M. Dean, "Swrl: A semantic web rule language combining owl and ruleml," <http://www.w3.org/Submission/SWRL/>, May 2004. [Online]. Available: <http://www.w3.org/Submission/SWRL/>
- [6] L. Feigenbaum, B. Szekely, and L. McCullough, "Cambridge semantics: Position paper on the semantic web in the oil and gas industry," in *W3C Workshop on Semantic Web in Oil and Gas Industry*, Houston, Texas, USA, 2008.
- [7] D. Norheim and R. Fjellheim, "Aksio - active knowledge management in the petroleum industry," in *Third European Semantic Web Conference*, Budva, Montenegro, 2006.
- [8] M. Oprea, M. Marcu, and M. P. Coloja, "Smartwellonto: An ontology for smart wells," in *International Multi-Conference on Computing in the Global Information Technology*, Bucharest, Romania, 2006.
- [9] R. Soma, A. Bakshi, and V. K. Prasanna, "A semantic framework for integrated asset management in smart oilfields," in *CCGRID '07: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 119–126.
- [10] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltramari, and L. Schneider, "The wonderweb library of foundational ontologies," <http://www.loa-cnr.it/Papers/DOLCE2.1-FOL.pdf>, May 2003. [Online]. Available: <http://www.loa-cnr.it/Papers/DOLCE2.1-FOL.pdf>
- [11] C. Lagoze and J. Hunter, "The abc ontology and model," in *DCMI '01: Proceedings of the International Conference on Dublin Core and Metadata Applications 2001*. National Institute of Informatics, Tokyo, Japan, 2001, pp. 160–176.
- [12] A. Kawazoe, C. H., M. Shigematsu, and N. Collier, "Structuring an event ontology for disease outbreak detection," *BMC Bioinformatics*, 2008. [Online]. Available: <http://dx.doi.org/10.1186/1471-2105-9-S3-S8>
- [13] M. Bertini, A. Del Bimbo, C. Torniai, C. Grana, and R. Cucchiara, "Dynamic pictorial ontologies for video digital libraries annotation," in *MIR '07*. New York, NY, USA: ACM, 2007, pp. 47–56.
- [14] M. Bertini, A. Del Bimbo, and G. Serra, "Learning ontology rules for semantic video annotation," in *Proceeding of the 2nd ACM workshop on Multimedia semantics*. New York, NY, USA: ACM, 2008.
- [15] S. A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie, "High speed and robust event correlation," *Communications Magazine, IEEE*, vol. 34, pp. 82–90, 1996. [Online]. Available: <http://dx.doi.org/10.1109/35.492975>
- [16] J. R. Hobbs and F. Pan, "An ontology of time for the semantic web," *ACM Transactions on Asian Language Information Processing (TALIP)*, vol. 3, no. 1, pp. 66–85, 2004.
- [17] "Schlumberger oilfield glossary," <http://www.glossary.oilfield.slb.com/>.
- [18] "Wikipedia," <http://www.wikipedia.org/>.