The Reporting Event Ontology Design Pattern and Its Extension to Report News Events

Ewa Kowalczuk and Agnieszka Ławrynowicz

Faculty of Computing, Poznan University of Technology, Poznań, Poland

Abstract. This article describes an ontology design pattern for modelling events as elements of discourse. Information provided about an event is not treated as universal truth, but rather as a statement of a particular agent, based on defined sources. We demonstrate how this pattern can be used for modelling historical debates. We also propose specialisation of the pattern, which models presentation of the current events in mass media for the purposes of citizen control.

Keywords: ontology design patterns, ODP, event-based model, discourse modelling, provenance, media control

1 Introduction

Our pattern proposition responds to the emerging need for storing and investigating not only information about a particular event, but also the provenance of the information and circumstances of its provision. Properties attributed to an event are not stored as facts, but as a narrative of a particular agent, which could differ from the narratives of other agents. The event reports, their content and attributes can be subsequently the subject of in-depth analysis.

Analysis of current events reporting and the reports consequences is one of the important areas of Media Studies, exemplified by surveys described in [1] and [2]. Recently, Machine Learning and Natural Language Processing techniques have been employed for automation of news messages analysis [3,4]. Although a more thorough validation of the methods used should be performed before drawing any sociological conclusions from them, these studies constitute an important step towards large-scale analyses of news content.

Such analyses might benefit from a pattern for storing the events as narratives of agents. However, not all existing event ontologies address the subjectivity of event properties. The well-established CIDOC Conceptual Reference Model (CRM) [5] provides a rich set of event properties, but does not allow to mark the level of property value probability or attribute a property assignment to a particular agent. Nor does the Linking Open Descriptions of Events ontology (LODE) [6], although it allows for linking events to media objects presenting them and thus denoting sources.

The IPTC NewsML-G2 controlled vocabulary [7] allows to define whether an event and its time interval are confirmed or not. IPTC rNews model [8] introduces a property rnews:accountablePerson for linking to a person responsible for a particular news item (as a whole).

The formalisms most closely related to our pattern proposal are Simple Event Model (SEM) [9] and BBC Storyline Ontology [10]. SEM introduces a special subclass of sem:Constraint, called sem:View, allowing to mark some attributed property as a belief (point of view) of a particular sem:Authority. The property assignment is constant: there is no means of representing the fact that a view changed over time or denoting other circumstances of the property assignment act, for example timestamp (this could be achieved by extending an ontology).

BBC Storyline Ontology introduces a notion of nsl:Storyline, which is used to denote the editorial perspective of an event or a group of events. It can be attributed to a specific owner. Storylines have generally a larger span than a single event. They can include nsl:StorylineSlots: real world events or inner storylines.

2 The Reporting Event Ontology Design Pattern

2.1 Intent

The intent of the pattern is to allow for modelling situations in which the knowledge about an event cannot be treated as certain. It is particularly useful for cases in which two or more agents provide different, contradictory information about the same event. It can be also used for modelling situation in which a single agent provided contradictory information about the same event at different points in time. The pattern allows for stating different circumstances of an act of the information provision.

2.2 Competency Questions

The pattern was designed to allow a completed ontology to answer the following questions:

- What characteristics (e.g. date, participants, cause) is an actual event said to have?
- Which agent made a statement about an actual event?
- On which sources these statements were based?
- What were the circumstances of providing information about an actual event?

2.3 Pattern Description

Figure 1 depicts our proposition of the **Reporting Event** ontology design pattern[1]. Yellow boxes represent classes included in the pattern. White boxes represent classes of existing patterns which are embedded in our pattern, in accordance with **Types of Entities**[2] pattern. The **Types of Entities** pattern, as well as

[1] `http://ontologydesignpatterns.org/wiki/Submissions:ReportingEvent`
[2] `http://ontologydesignpatterns.org/wiki/Submissions:Types_of_entities`

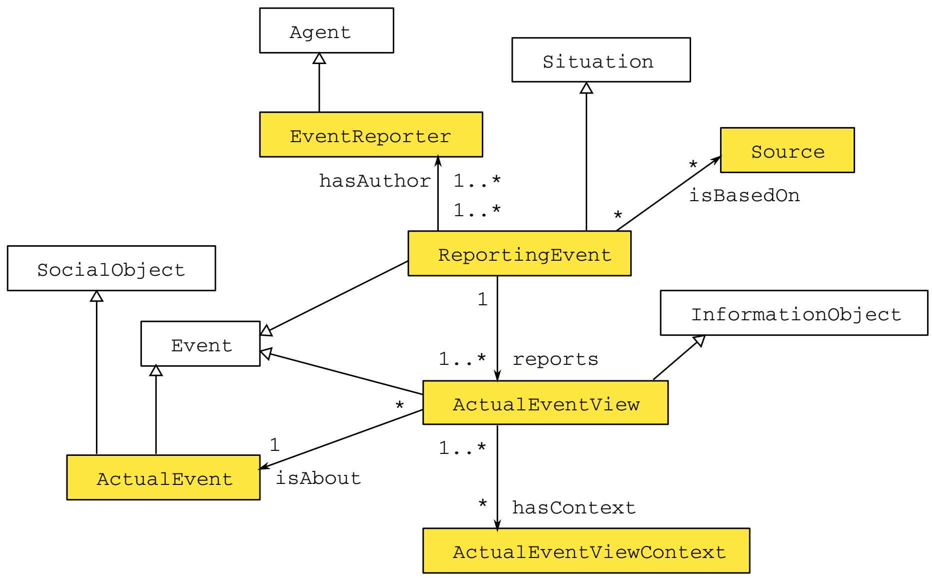

Fig. 1. Reporting Event ontology design pattern.

the most of the other patterns we reused, was listed in NeOn project deliverable [11] and included in DOLCE+DnS Ultralite ontology [12].

There are in total 3 different types of events included in the pattern. The most important one is ActualEvent, which is the *physical* event – event that actually happened or is said to have happened. All the circumstances of the event (contexts) are not directly linked to the ActualEvent. They are attached to the ActualEventView instance, which constitutes a view of the actual event included in a particular description (report, statement).

By making the ActualEventView an Event, we can use all properties that could be used in case of an ordinary event, like place, time interval, cause and consequences. They are grouped in a class ActualEventViewContext and are attached by hasContext property or its specialisations.

As ActualEventView provides an information about the ActualEvent instance, it can be viewed as an InformationObject. It is connected to an ActualEvent instance using the isAbout property. This fragment of our pattern embeds the **Intention Extension**[3] pattern.

The third event included in our pattern is the act of reporting the actual event – the ReportingEvent. Its result is an ActualEventView (linked using property reports). By modelling the act of reporting as an event, we can expose its different circumstances, for example the timestamp at which the particular reporting took place. This timestamp would probably differ from the timestamp specified for the ActualEventView. The act of reporting usually takes place later than

[3] `http://ontologydesignpatterns.org/wiki/Submissions:IntensionExtension`

the event reported (the exceptions are live transmissions and announcements of future events). Because the ReportingEvent defines different circumstances of the ActualEventView provision, it can also be viewed as the specialisation of the Situation class from the **Situation** design pattern[4]. The ReportingEvent can be also treated as an activity, performed by a certain agent (person or organisation) – EventReporter, utilising defined sources.

2.4 Example Usage: Historical Debate

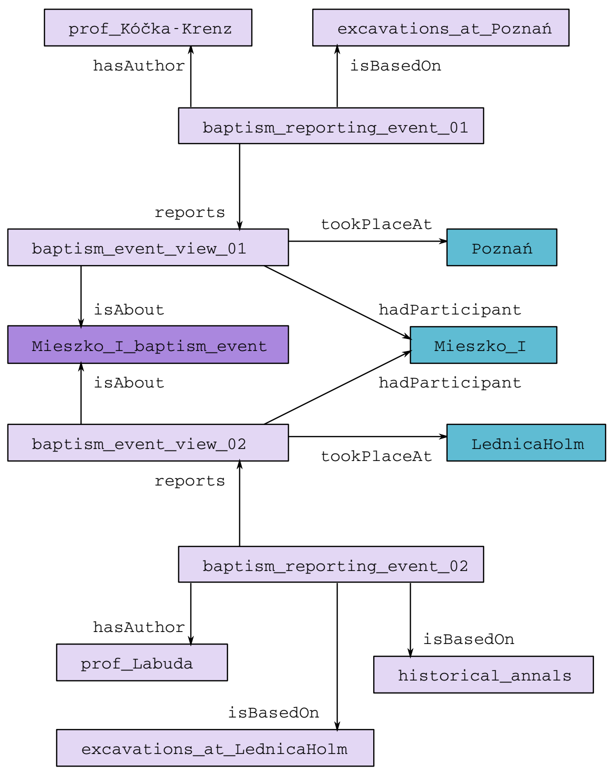

Fig. 2. Using the **Reporting Event** pattern to model different opinions of historians.

Figure 2 shows an example usage of the **Reporting Event** pattern. We model the situation in which two History professors have different opinion about the same actual event. The exemplary actual event is the baptism of prince Mieszko I, which inaugurated the christianisation of Poland. The baptism exact place is subject to historical debate. According to Prof. Hanna Kóčka-Krenz the likely

[4] `http://ontologydesignpatterns.org/wiki/Submissions:Situation`

place was Poznań. She bases her claim on Poznań excavations that she conducted. However, late Prof. Labuda pointed to Lednica Holm. He also based his claims on the local excavations, but also on historical annals.

Figure 2 presents only instances of classes (violet boxes) and properties that link them. The central element of the example ontology is an ActualEvent: Mieszko_I_baptism_event (denoted in darker violet colour). We do not attach any hasContext properties to the event itself, as it would put us in the same position as other agents making claims about the event. Instead, we link the event to the corresponding ActualEventView, which constitutes an overall picture of the actual event in the eyes of a particular researcher. The event baptism_reported_event_01 is linked to Poznań by property tookPlaceAt (which is a subproperty of hasContext). The event baptism_reported_event_02 is linked to LednicaHolm instead.

Some convictions about the actual event are common to both researchers (for example the fact that Mieszko was present during his own baptism). The property hadParticipant is also a subproperty of hasContext. This shows that we can describe different circumstances of an actual event, mentioned in a given event report.

The events baptism_reporting_event_01 and baptism_reporting_event_02, of type ReportingEvent, model the act of making a statement about the actual event. This activity is performed by defined authors and is based on the particular sources.

2.5 Pattern Formalisation

The pattern can be formalised in the Description Logics [13] as follows:

$\mathsf{ActualEvent} \sqsubseteq \mathsf{Event}$
$\mathsf{ActualEvent} \sqsubseteq \mathsf{SocialObject} \sqsubseteq \mathsf{Object}$

$\mathsf{ActualEventView} \sqsubseteq \mathsf{Event}$
$\mathsf{ActualEventView} \sqsubseteq \mathsf{InformationObject} \sqsubseteq \mathsf{Object}$
$\mathsf{ActualEventView} \sqsubseteq\ = 1\ \mathsf{isAbout.ActualEvent}$
$\mathsf{ActualEventView} \sqsubseteq \forall \mathsf{isAbout.ActualEvent}$
$\mathsf{ActualEventView} \sqsubseteq\ = 1\ \mathsf{reports}^{-1}\mathsf{.ReportingEvent}$
$\mathsf{ActualEventView} \sqsubseteq \forall \mathsf{hasContext.ActualEventViewContext}$

$\mathsf{ActualEventViewContext} \sqsubseteq \exists \mathsf{hasContext}^{-1}\mathsf{.ActualEventView}$

$\mathsf{ReportingEvent} \sqsubseteq \mathsf{Event}$
$\mathsf{ReportingEvent} \sqsubseteq \mathsf{Situation}$
$\mathsf{ReportingEvent} \sqsubseteq \exists \mathsf{reports.ActualEventView}$
$\mathsf{ReportingEvent} \sqsubseteq \forall \mathsf{reports.ActualEventView}$
$\mathsf{ReportingEvent} \sqsubseteq \exists \mathsf{hasAuthor.EventReporter}$
$\mathsf{ReportingEvent} \sqsubseteq \forall \mathsf{isBasedOn.Source}$

$\mathsf{EventReporter} \sqsubseteq \mathsf{Agent}$
$\mathsf{EventReporter} \sqsubseteq \exists \mathsf{hasAuthor}^{-1}.\mathsf{ReportingEvent}$

$\mathsf{Source} \sqsubseteq (\mathsf{Event} \sqcup \mathsf{Object})$

$\mathsf{reports} \sqsubseteq \mathsf{isSettingFor}$

The pattern was also formalised in OWL2, as a reusable ontology fragment[5]. It imports and specialises the embedded patterns ontologies as well as PROV-O vocabulary [14] (described below). To enhance the interoperability of the pattern, we decided to introduce only general, universal properties, like owns, which usually don't have strict domain and range restrictions. Instead, we narrow down the classes definitions using the subClassOf restrictions. This approach results in more costly reasoning [15], but it might also bring more interesting and complex reasoning results, including the detection of non-trivial inconsistencies, which are particularly interesting for our provisioned pattern usage.

2.6 Related Ontologies

We believe that proposition of an ontology design pattern should be accompanied, if possible, with recommendations of good-quality and widely used ontologies, which can constitute a proper base for some of the newly created ontology components.

The **Reporting Event** ontology design pattern was modelled to allow for reuse of the elements of the PROV-O vocabulary [14]. The PROV-O ontology was imported into OWL formalisation of the pattern[6]. Specialisations of PROV-O classes and properties were created. Class ActualEventView is a subclass of prov:Entity. It is created in prov:Activity, which in our case is ReportingEvent. Property reports, which links ReportingEvent and ActualEventView, is a specialisation of $\mathsf{prov{:}wasGeneratedBy}^{-1}$. The activity is linked to prov:Agent, in our case EventReporter. The property hasAuthor is specialization of prov:wasAssociatedWith. We propose two types of sources: SourceObject (subclass of prov:Entity) and SourceActivity (subclass of prov:Activity). They can be linked to ReportingEvent using properties isBasedOnSourceObject (subproperty of prov:used) and isBasedOnSourceActivity (subproperty of prov:wasInformedBy), respectively.

An ontology using our desing pattern would also benefit from utilising a well-established ontology designed to describe different properties of events, for example CIDOC CRM [5]. As it was mentioned in the introduction, CIDOC CRM treats facts stated about events as certain. However, it still can be used to attach different context properties to ActualEventViews objects. The example provided above utilises properties cidoc:P11_had_participant and cidoc:P7_took_place_at, which in our case inherit from hasContext property.

[5] http://www.semantic.cs.put.poznan.pl/ont/reportingevent.owl
[6] http://www.semantic.cs.put.poznan.pl/ont/reportingevent.owl

3 The News Reporting Event Ontology Design Pattern

3.1 Intent

The **News Reporting Event** pattern can be used for modelling situations in which we are not certain that a particular actual event has the properties which were described in a news message. We want to define the properties of an actual event which were reported (time, place, actors, subevents, cause, effect etc.), but not to treat them as universal, verified knowledge. The pattern also allows to define who is responsible for a particular description of an event and how this description is dealt with.

3.2 Competency Questions

In addition to the competency questions defined for pattern **Reporting Event**, the **News Reporting Event** pattern allows to answer the following questions:

- What aspects of an actual event were presented in the news message?
- Who reported an actual event? Which news provider they represented?
- When was a certain actual event reported for the first time?
- What actual events are presented in a certain medium/by media of a certain news provider?
- How was an actual event presented?

3.3 Pattern Description

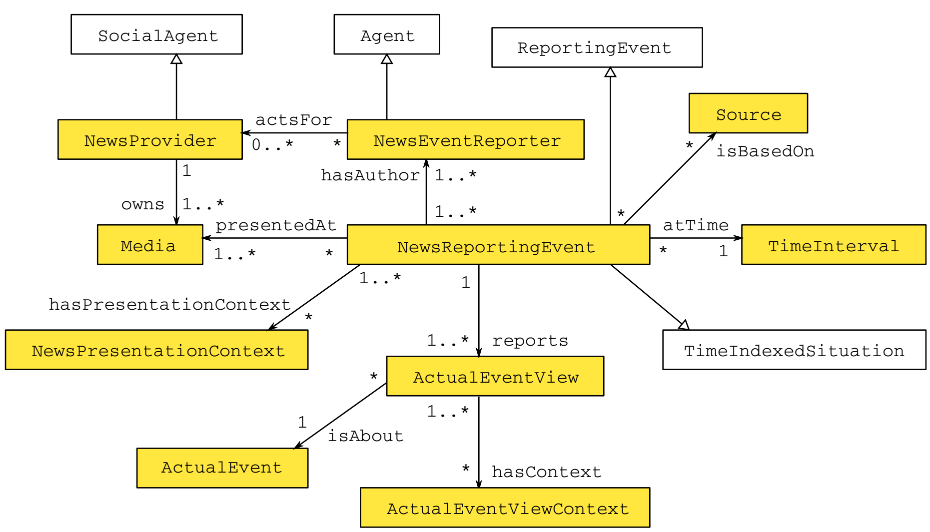

Fig. 3. The **News Reporting Event** ontology design pattern.

The **News Reporting Event** pattern[7] extends the **Reporting Event** pattern by specifying the primary properties of the specialisation of the ReportingEvent class. The specialisation, NewsReportingEvent, denotes the act of providing a unit of information (ActualEventView) to the general public.

The act utilises a certain Media (TV station, radio station, newspaper, website). A Media is owned by a certain SocialAgent – NewsProvider. This agent takes partial responsibility for the content provided, but its responsibility differs from the one of the NewsEventReporter – a Person that directly reports the ActualEvent. To present this fact, we reuse the pattern **Acting For**[8].

We also include the pattern **Time-indexed Situation**[9] which allows us to mark the ActualEventView with a TimeInterval at which it was presented to the general public. It can range from a few seconds – at which a particular fragment of a TV programme was aired – to a few days – time at which an article was displayed in the current events section of a website. The class NewsReportingEvent inherits from class TimeIndexedSituation. Information whether material was marked with red font, announced in the beginning of a programme etc., can be represented using objects of class NewsPresentationContext.

3.4 Example Usage: Presentation of Social Unrests

Included example shows two different media reporting a single event (bus drivers protest). This is evident by two instances of class NewsEvent – protest_news_event_01 and protest_news_event_02 – linking to the same ActualEvent – bus drivers protest. We do not state any other facts about the actual event, and only view it in the perspective of two varying reports. First report (protest_reporting_event_01) claims that the protest was caused by the malfunction of buses and included a brutal police intervention. Second report (protest_reporting_event_02) does not mention the intervention, but includes destruction of public property. It adds another cause of protest, namely planned reduction of social benefits. It can be observed in the figure that not all event views are linked to all instances of class ActualEventViewContext (denoted in blue).

From the presented ontology fragment we cannot conclude any certain facts about the event (or even that it actually happened). We can only notice similarities and differences in the news messages. We can also observe that for one medium the message was extremely important (it was included in prime time as first news programme material), while for the other it was much less important (it was presented at the bottom of the webpage).

In general, the listed observations might constitute a significant information for anyone interested in analysis of media coverage, for example a watchdog organisation which controls the public media objectivity.

---

[7] http://ontologydesignpatterns.org/wiki/Submissions:NewsReportingEvent
[8] http://ontologydesignpatterns.org/wiki/Submissions:ActingFor
[9] http://ontologydesignpatterns.org/wiki/Submissions:TimeIndexedSituation

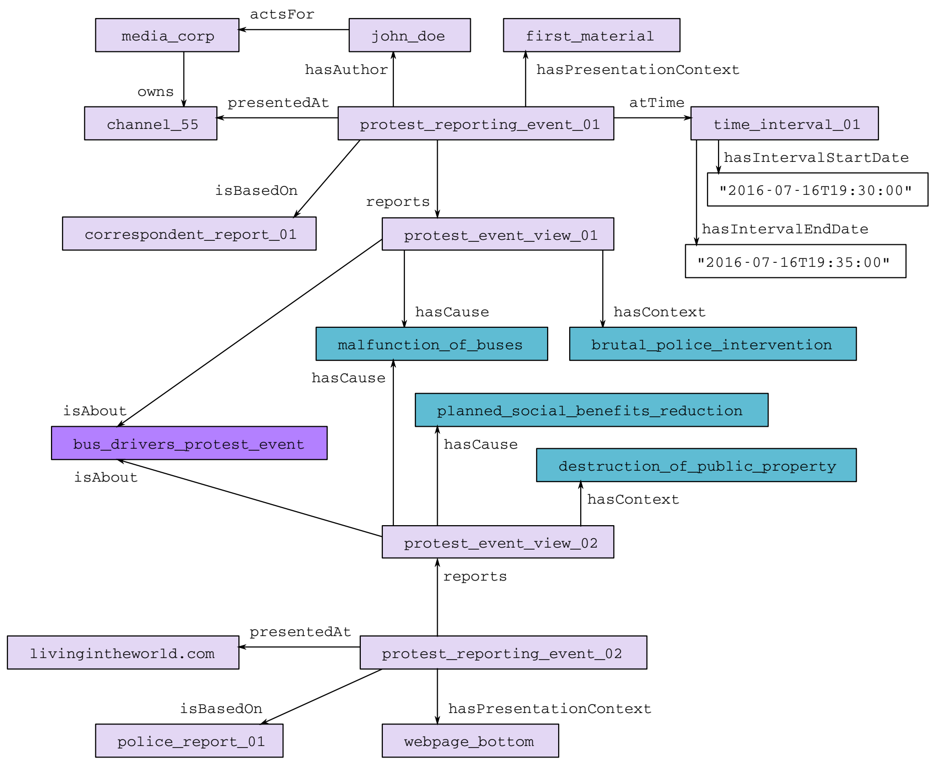

Fig. 4. Example usage of **News Reporting Event** pattern: single event reported differently in different media.

3.5 Pattern Formalisation

The addition to the formalisation of the **Reporting Event** and the embedded patterns the formalisation of the pattern **News Reporting Event** is as follows:

$$\mathsf{NewsReportingEvent} \sqsubseteq \mathsf{ReportingEvent}$$
$$\mathsf{NewsReportingEvent} \sqsubseteq \mathsf{TimeIndexedSituation}$$
$$\mathsf{NewsReportingEvent} \sqsubseteq \exists \mathsf{hasAuthor}.\mathsf{NewsEventReporter}$$
$$\mathsf{NewsReportingEvent} \sqsubseteq \exists \mathsf{presentedAt}.\mathsf{Media}$$

$$\mathsf{NewsEventReporter} \sqsubseteq \mathsf{EventReporter}$$
$$\mathsf{NewsEventReporter} \sqsubseteq \exists \mathsf{hasAuthor}^{-1}.\mathsf{NewsReportingEvent}$$

$$\mathsf{NewsProvider} \sqsubseteq \mathsf{SocialAgent}$$
$$\mathsf{NewsProvider} \sqsubseteq \exists \mathsf{owns}.\mathsf{Media}$$

$$\mathsf{Media} \sqsubseteq\ = 1\ \mathsf{owns}^{-1}.\mathsf{NewsProvider}$$

$\mathsf{NewsPresentationContext} \sqsubseteq \exists \mathsf{hasPresentationContext}^{-1}.\mathsf{NewsReportingEvent}$

$\mathsf{hasPresentationContext} \sqsubseteq \mathsf{isSettingFor}$

The pattern was also formalised in OWL2[10].

4 Discussion

What might seem unintuitive or even clearly wrong about the ActualEventView is its dual nature: it inherits both from the InformationObject and the Event. However, we believe that it accurately captures our intent: the ActualEventView can posses all features which are normally attributed to real world events – we can capture all properties defined by the EventReporter within an adopted event model. On the other hand, we can treat the ActualEventView as a piece of information, InformationObject, provided by an EventReporter during a ReportingEvent.

We intent to use the **News Reporting Event** pattern as the main building block in our ontology for documenting the content of television news programmes. The foreseen aim of the ontology is supervision of news media. We would like to analyse what aspects of the actual events are presented. Are some events not mentioned at all? Are some circumstances of events commonly ignored? Does it differ from one media to another? We would like to initially curate the data manually, in order to create a baseline for automatic knowledge base population and data analysis. Automatic processing of data will face many challenges. One of them is instance matching. If one media is discussing „destruction of public property" and another „acts of vandalism", is this the same aspect of a particular event?

Another important issue, which should be taken into consideration, is the ability to use the pattern for reasoning. In the ideal case, the reasoning process should result in detection of inconsistencies between different reports of the same ActualEvent or even arrive at the correct properties that can be attached directly to the ActualEvent. However, it might be very hard to achieve these tasks utilising a Description Logics model, as they include the Open World Assumption [13]: we cannot definitely say that protest_news_event_01 did not mention destruction_of_public_property – it is unknown. We could partially close the model by introducing a property doesNotHaveContext, but the consequences on reasoning validity and performance are yet to be analysed.

For simple cases, like analysis of where the event took place, one might want to merge all instances of ActualEventView into a single event and look for inconsistencies within it. This would require designing of strong, formal rules, like $\mathsf{ActualEventView} \sqsubseteq\ = 1\ \mathsf{tookPlaceAt.Place}$. These rules might still not work in real world examples: if one of the event specified Warsaw and another one Poland, or if event happened at few places simultaneously, like wide-spread protests.

[10] `http://www.semantic.cs.put.poznan.pl/ont/newsreportingevent.owl`

Another interesting approach to solve the inference problem, would be to reuse constraints/views idea from the Simple Event Model. In this case every property type would be defined separately using a specific subclass of sem:View, which could have some additional restrictions defined. Nevertheless, the problems described above might still occur in this case.

It should be also noted that the **News Reporting Pattern** is not an universal solution for storing any kind of the news media content, like narratives and descriptions of social phenomena. One could attempt to squeeze such descriptions into the event-based model, by creating a long-term events, but it might be not a satisfactory solution. Nevertheless, the general idea behind our pattern, that a certain statement is treated not as an universal fact, but as a view of a defined agent, could be reused to create an analogical pattern for storing descriptions and opinions.

5 Summary

We proposed a new ontology design pattern and showed how it can be used for modelling cases where the knowledge about a particular event is not certain and originates from one or more agents whose reports might differ. We also presented a specialisation designed for news messages, which could be utilised for computational sociology studies and citizen control of mass-media.

Our design patterns reuse some of the existing patterns (other patterns can be embedded if needed) and gather some of the phenomena included in the existing formalisms related to events, news reporting and provenance. Our patterns allow for attaching any of the properties which are normally directly attributed to an event to the special event-view objects and to state any circumstances of the event report, like time interval and utilised sources.

Acknowledgement

Agnieszka Ławrynowicz acknowledges the support from the project "Ontological foundations for developing historical geographic information systems" (2b H15 0216 83) funded by the National Humanities Development Program. The authors thank Karl Hammar and two other anonymous reviewers for constructive feedback on the paper.

References

1. McClure, J., Velluppillai, J.: The Effects of News Media Reports on Earthquake Attributions and Preventability Judgments: Mixed Messages About the Canterbury Earthquake. Australasian Journal of Disaster & Trauma Studies **1** (2013) 27–35
2. Lugalambi, G.W., Nyabuga, G.M., Wamala, R., Yeboah, A.A., Lambiv, G.T., Lukanda, I.N., Sebaana, H.N., Siro, A.K., Phakisi, N., Erastus, E.: Media Coverage of Science and Technology in Africa (2011)

3. Ali, O., Flaounas, I., De Bie, T., Mosdell, N., Lewis, J., Cristianini, N.: Automating News Content Analysis: An Application to Gender Bias and Readability. Workshop on Applications of Pattern Analysis (WAPA). JMLR: Workshop and Conference Proceedings (2010) 36–43
4. Van Atteveldt, W., Sheaferm, T., Shenhav, S., Fogel-Dror, Y.: Clause Analysis: Using Syntactic Information to Enrich Frequency-based Automatic Content Analysis. In: Symposium New Frontiers of Automated Content Analysis in the Social Sciences. (2015)
5. Le Boeuf, P., Doerr, M., Ore, C.E., Stead, S., eds.: Definition of the CIDOC Conceptual Reference Model. Version 6.2.1. (2015) http://new.cidoc-crm.org/sites/default/files/cidoc_crm_version_6.2.1.pdf.
6. Shaw, R., Troncy, R., Hardman, L.: LODE: Linking Open Descriptions of Events. In: Proceedings of the 4th Asian Conference on The Semantic Web. ASWC '09, Berlin, Heidelberg, Springer-Verlag (2009) 153–167
7. IPTC: IPTC NewsML-G2 standard https://iptc.org/standards/newsml-g2/.
8. IPTC: IPTC rNews standard https://iptc.org/standards/rnews/.
9. van Hage, W.R., Malaisé, V., Segers, R.H., Hollink, L., Schreiber, G.: Design and use of the Simple Event Model (SEM). Web Semantics: Science, Services and Agents on the World Wide Web **9**(2) (2011)
10. Rissen, P., Lippell, H., Chadburn, M., Leitch, T., Brickley, D., Smethurst, M., Cevey, S.: BBC Storyline Ontology http://www.bbc.co.uk/ontologies/storyline.
11. Presutti, V., Gangemi, A., David, S., de Cea, G.A., Suarez-Figueroa, M.C., Montiel-Ponsoda, E., Poveda, M.: A Library of Ontology Design Patterns: Reusable Solutions for Collaborative Design of Networked Ontologies. NeOn Deliverable D2. 5.1, Institute of Cognitive Sciences and Technologies, CNR (2008)
12. Gangemi, A.: The DOLCE+DnS Ultralite Ontology http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite.
13. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, New York, NY, USA (2003)
14. McGuinness, D., Lebo, T., Sahoo, S.: PROV-O: The PROV Ontology. W3C recommendation, W3C (April 2013) http://www.w3.org/TR/2013/REC-prov-o-20130430/.
15. Hammar, K. In: Ontology Design Pattern Property Specialisation Strategies. Springer International Publishing, Cham (2014) 165–180