Simple Life-Events Ontology in SU(M)O-KIF

Boštjan Berčič, Mirko Vintar

University of Ljubljana, Faculty of Administration, Gosarjeva 5, 1000 Ljubljana, Slovenia
{bostjan.bercic , mirko.vintar}@fu.uni-lj.si

Abstract. This paper presents a proof-of-concept research into building an ontology of life-events in public administration and integration of this domain ontology with standard upper ontology. SUMO was chosen to be the upper ontology and SUO-KIF the language of both upper and domain ontology.

1 Introduction

Various approaches have been tried so far in the modelling of life-events. Some of them stressed the procedural nature of life-events, others defined activities as main contents of particular life-events without recourse to their intertemporal relations (see for example LEAP 2, eGOV3, ILEP 4 for different approaches). Our approach consisted of mapping a set of activities contained in the specific life-event instance to a procedural image, thus defining sequential, causal and intertemporal relations between them (see 4) .

Models of different life-events share a number of common terms, such as institutions, administrative procedures, rights and duties etc. It would therefore be of great help if a standard vocabulary (ontology) of terms that make up life-events would exist. On the other hand, not only elements which form part of life-events, but life-events themselves could be organized into an ontology of life-events. Such an ontology would consist of all known life-events (i.e. life-events considered so far in various projects) hierarchically organized into a tree with type-subtype and instance-type relations.

2 SUMO and SUO-KIF

To use pre-existing general knowledge about the world, one has to make use of some top level ('upper') ontology and connect it to life-events ontology. Life-events ontology, being a domain ontology, should specialize some standard upper ontology such as SUO, SUMO, OpenCyc, etc.

Alongside with various ontological commitments, standard upper ontology also provides a formal language for the description of types and instances of life-events. Many upper ontologies use first-order logic in a more or less restricted way to express its concepts (Ontolingua and SU(M)O use KIF, Cyc uses CycL, all are dialects of first-order logic).

We have chosen to use SUMO 5 (Sugested Upper Merged Ontology) as an upper ontology for our sample life-events ontology with its SUO-KIF 6 development language. Having said this, our goal in this paper (and corresponding research) was to:

- build a simple life-events ontology with SUO-KIF language,
- connect our life-events ontology to SUMO (connecting concepts from both ontologies in appropriate places, i.e. appending concepts from life-events ontology to lower parts of SUMO ontology and deriving life-event ontology concepts from SUMO concepts).

This simple life-events ontology serves us as a proof-of-concept for the latter building of larger life-events ontology.

3 Sample Life-Events Ontology

We start with the definition of LifeEvent class which will be the root element in our ontology. Since it denotes an intentionally carried out process, we append it to (derive it from) IntentionalProcess class in SUMO, which is a subclass of Process and denoting generic processes. We also define that LifeEvent class be exhaustively and in a mutually disjoint way covered by two subclasses: BusinessRelatedLifeEvent and CitizensRelatedLifeEvent. We also define superclasses for both.

```
(subclass LifeEvent IntentionalProcess)
(partition LifeEvent BusinessRelatedLifeEvent CitizensRelatedLifeEvent)
(subclass CitizensRelatedLifeEvent LifeEvent)
(subclass BusinessRelatedLifeEvent LifeEvent)
```

We could next define a few subclasses for citizens- and business-related life-event classes (see appendix). We limit ourselves here to citizens-related life-events and define four subclasses: family, house, education and employment related classes of life-events.

Note the absence of the partition relation here; partition relation partitiones subclasses of a given class disjointly and exhaustively. In our example this means that we would neither like this partition to be exhaustive (maybe we will want to add new subclasses in the future, such as e.g. HealthRelatedLifeEvent), nor to be necessarily disjoint subclasswise (for example, there is usually expected to be some overlap in family- and house-related life-events as well as in family- and education-related life-events).

On the other hand, we expect that house-related life-events and education-related life-events as well as house-related life-events and employment-related life-events will be respectively mutually disjoint. We write this as:

```
(disjoint HouseRelatedLifeEvent EducationRelatedLifeEvent)
(disjoint HouseRelatedLifeEvent EmploymentRelatedLifeEvent)
```

But since we cannot reasonably expect education- and employment-related life-events to be mutually disjoint, we don't define disjoint relation here.

We go on by defining some more layers of subclasses. Consult appendix for detailed description.

In our research we model life-events for what they mean in terms of administrative porcedures that have to be initiated and performed. Life-events thus become description of various activities that have to be caried out in order to solve problems posed by them (e.g. problem of obtaining birth certificate after birth of a child). Life-events which we derive from class Process in SUMO can thus be subdivided in subprocesses. In SUMO they are defined with subProcess relation whose signature is:

(domain subProcess 1 Process)
(domain subProcess 2 Process).

First we define types of these subprocesses to be IntentionalProcesses (see apendix). Then we define subprocesses for BirthLifeEvent and ChanceOfDomicileLifeEvent:

(subProcess BirthCertificateApplicationProcess BirthLifeEvent)
(subProcess PassportChangeApplicationProcess BirthLifeEvent)
(subProcess ChangeOfAdressApplicationProcess ChanceOfDomicileLifeEvent)
(subProcess ChangeOfAdressPersonalIDApplicationProcess ChanceOfDomicileLifeEvent)
(subProcess ChangeOfAdressDrivingLicenceApplicationProcess ChanceOfDomicileLifeEvent).

Finally we take one of those subprocesses (say BirthCertificateApplicationProcess) and define it in detail in terms of case roles involved (who is the the agent of action, what kind of entity is in the patient role, who is the beneficiary etc). SUMO possesses several processes defined in terms of case roles so we first try to map our BirthCertificateApplicationProcess to some SUMO concept. Since 'apply' does not exist in SUMO, we try with mapping of english words from WordNet to SUMO concepts, as available in SUMO Search Tool at 1. Several mappings are possible, the most semantically appropriate is the fourth one: Requesting, asking for something. We then check SUMO internal definition of Requesting process, which is defined as "a request expresses a desire that some future action be performed. For example, the 5th Battalion requested air support from the 3rd Bomber Group. Note that this class covers proposals, recommendations, suggestions, etc." This seems fine so we choose Requesting as a substitute for applying.

We therefore define BirthCertificateApplicationProcess to be a subclass of Requesting process. Next we have to choose appropriate case (thematic) roles for Requesting process (agent, patient, etc.). We have in mind a sentence like :« Upon a birth of a child, a citizen requests from appropriate governmental body that it issue him a birth certificate.« We define agent of this process to be of type Human , a special kind of Agent in SUMO concept hierarchy. This agent is then connected to the requesting process with agent thematic relation. A governmental body, which is the addressee or recepient in this case, is linked to the requesting process with destination case role. We also define this governmental body to be an instance of Government type.

(subclass BirthCertificateApplicationProcess Requesting)
(agent BirthCertificateApplicationProcess ?HUMAN)
(instance ?HUMAN Human)
(destination BirthCertificateApplicationProcess ?ADMINISTRATIVE_UNIT)
(instance ?ADMINISTRATIVE_UNIT Government)

The gist of the sentence is the request that government should issue birth certificate. Issuing a birth certificate is a process on its own and should be modelled as such.

First question to be answered is how is this new process related to the process of Requesting. Since issuing a certificate is a theme of the request process, it must be modelled using the patient case role. Issuing process is then described using Formula class as a second argument to patient role (Formula class denotes a syntactically well-formed formula in the SUO-KIF knowledge representation language, so a process description as a well-formed formula suits this definition).

Next question to be answered is which process in SUMO ressembles most certificate issuing activity (Issuing concept does not exist in SUMO). We use SUMO Search Tool again and try to find the most appropriate mapping of verb issue. There are again various options (Publication, Giving, Declaring, Motion,Writing). After a bit of search in SUMO we choose Writing as the most appropriate one because Writing results in class Text which has Certificate as one of its subclasses. So we write:

```
(patient BirthCertificateApplicationProcess
        ( and (subclass IssueBirthCertificate Writing)
              (agent IssueBirthCertificate ?ADMINISTRATIVE_UNIT)
                 (instance ?ADMINISTRATIVE_UNIT Government)
                 (result IssueBirthCertificate BirthCertificate)
                 (subclass BirthCertificate Certificate)))
```

Finally we mustn't forget to declare this formula as an instance of Formula class.

```
(instance (and (instance IssueBirthCertificate Writing)
                  (agent IssueBirthCertificate ?ADMINISTRATIVE_UNIT)
                     (instance ?ADMINISTRATIVE_UNIT Government)
                     (result IssueBirthCertificate BirthCertificate)
                  (instance BirthCertificate Certificate)) Formula)
```

These relations are now depicted on figure 1.

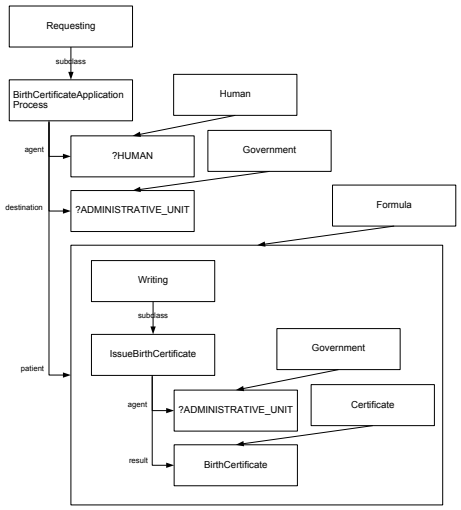

Fig. 1. Specification of BirthCertificateApplicationProcess

4 Integration with SUMO

We can now make connection between SUMO and our domain ontology explicit. Figure 2 presents upper levels of SUMO (beginning from root at the top) and conection points to our life-events ontology in lower levels of SUMO. Connection points are shadowed.

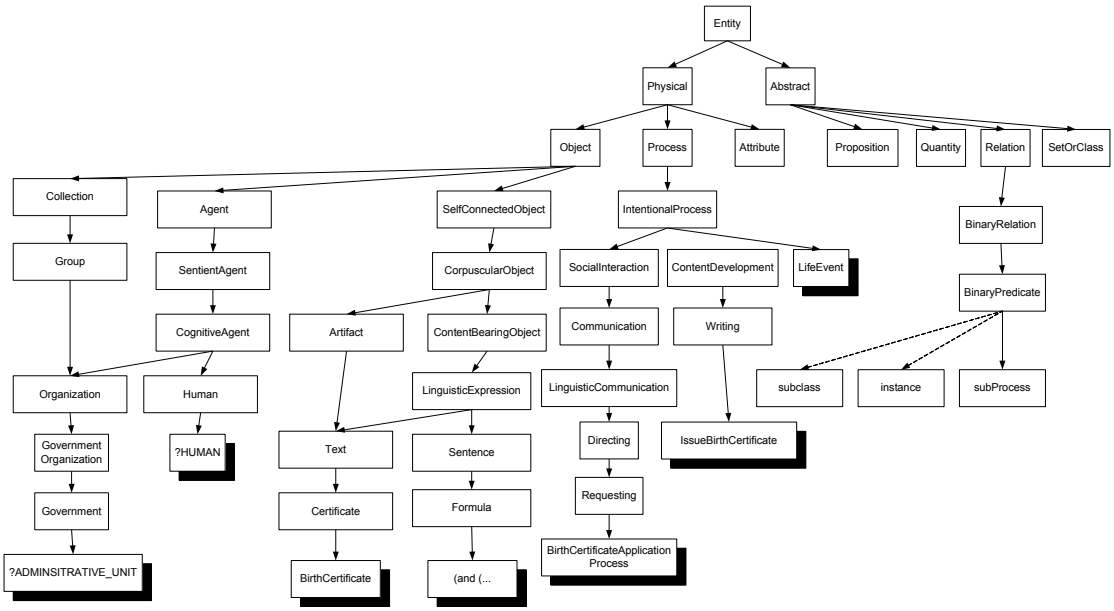

Fig. 2 SUMO and connection points to life-events ontology

In short, SUMO has Entity class as its root element. Entity is partitioned (exhaustively and mutually disjoint subclasswise) into Physical and Abstract classes. Physical is partitioned into Object and Process classes, Abstract is partitioned into Attribute, Proposition, Quantity, Relation and SetOrClass classes. Connecting point for LifeEvent class is SUMO IntentionalProcess class (life-events are processes and they are carried out intentionally). Connecting point for government institutions is Government class which is a (multiple) descendant from Object (one through Agent class and the other through Collection class). Citizen is connected to SUMO under Human class (which is itself subclass of CognitiveAgent and therefore subclass of Object). Documents (like birth and death certificates) are connected to SUMO under Certificate class (which is subclass of Text which is in its turn subclass of Artifact, a product of the process of Making). BirthCertificateApplicationProcess is defined as a subclass of Requesting, which is a special kind of Communication and therefore also IntentionalProcess. The same goes for IssueBirthCertificate process, which is Writing and as such Content Development and ultimately also IntentionalProcess.

5 Conclusions

Life-events can be (and historically have been) modeled using different modelling languages and ontological commitments. We used first-order logic based SUO-KIF language for the description of life-events in this paper for two reasons. First was our wish to be able to express life-events matter as detailed and as succint as possible. Second, using SUO-KIF we were able to merge our simple life-events ontology with

SUMO upper ontology. This gave us the benefit of not having to define every bit of information for our life-events from scratch and allowed us to rely on predefined general concepts from SUMO. Our future work will be to refine this life-events ontology and to translate some of our models to first-order logic based formalism and connect them to this ontology. We hope that such models of life-events, defined within a common ontological framework, will be both more expressive and more exact that the old ones.

References

1. http://128.136.11.20:8080/sigma/skb.jsp?skb=SUMO_skb
2. LEAP Life-Events Access Project, available at www.leap.gov.uk
3. E. Tambouris, E. Spanos, G. Kavadias (Editors), Services and Process models functional specifications, Deliverable of IST Project IST-2000-28471 An Integrated Platform for Realising Online One-Stop Government (eGOV),2001.
4. M. Vintar, A. Leben, The concepts of an Active Life-Event Public Portal, Proceedings of the First International Conference,EGOV 2002, Aix-En-Provence, France,September 2002,pp.383-390.
5. SUMO , available at http://ontology.teknowledge.com/
6. SUO_KIF, available at http://suo.ieee.org/suo-kif.html
7. Berčič B., Vintar M.: Ontologies, web services, and intelligent agents : ideas for further development of life-event portals, Lecture notes in computer science n.2739, 2003, Springer Verlag ,pp. 329.-334

APPENDIX A: SIMPLE LIFE-EVENTS ONTOLOGY

```
(subclass LifeEvent IntentionalProcess)
(partition LifeEvent BusinessRelatedLifeEvent CitizensRelatedLifeEvent)
(subclass CitizensRelatedLifeEvent LifeEvent)
(subclass BusinessRelatedLifeEvent LifeEvent)

(subclass FamilyRelatedLifeEvent CitizensRelatedLifeEvent)
(subclass HouseRelatedLifeEvent CitizensRelatedLifeEvent)
(subclass EducationRelatedLifeEvent CitizensRelatedLifeEvent)
(subclass EmploymentRelatedLifeEvent CitizensRelatedLifeEvent)

(disjoint HouseRelatedLifeEvent EducationRelatedLifeEvent)
(disjoint HouseRelatedLifeEvent EmploymentRelatedLifeEvent)

(subclass BirthLifeEvent FamilyRelatedLifeEvent)
(subclass ChangeOfDomicileLifeEvent FamilyRelatedLifeEvent)
(subclass ChangeOfDomicileLifeEvent HouseRelatedLifeEvent)

(subclass BirthCertificateApplicationProcess IntentionalProcess)
(subclass PassportChangeApplicationProcess IntentionalProcess)
```

(subclass ChangeOfAdressApplicationProcess IntentionalProcess)
(subclass ChangeOfAdressPersonalIDApplicationProcess IntentionalProcess)
(subclass ChangeOfAdressDrivingLicenceApplicationProcess IntentionalProcess)

(subProcess BirthCertificateApplicationProcess BirthLifeEvent)
(subProcess PassportChangeApplicationProcess BirthLifeEvent)
(subProcess ChangeOfAdressApplicationProcess ChanceOfDomicileLifeEvent)
(subProcess ChangeOfAdressPersonalIDApplicationProcess ChanceOfDomicileLifeEvent)
(subProcess ChangeOfAdressDrivingLicenceApplicationProcess ChanceOfDomicileLifeEvent).

(subclass BirthCertificateApplicationProcess Requesting)
(agent BirthCertificateApplicationProcess ?HUMAN)
(instance ?HUMAN Human)
(destination BirthCertificateApplicationProcess ?ADMINISTRATIVE_UNIT)

(instance ?ADMINISTRATIVE_UNIT Government)

(patient BirthCertificateApplicationProcess
(and (subclass IssueBirthCertificate Writing)
(agent IssueBirthCertificate ?ADMINISTRATIVE_UNIT)
(instance ?ADMINISTRATIVE_UNIT Government)
(result IssueBirthCertificate BirthCertificate)
(subclass BirthCertificate Certificate)))

(instance (and (subclass IssueBirthCertificate Writing)
(agent IssueBirthCertificate ?ADMINISTRATIVE_UNIT)
(instance ?ADMINISTRATIVE_UNIT Government)
(result IssueBirthCertificate BirthCertificate)
(subclass BirthCertificate Certificate)) Formula)