CKR:Live Demo: Using Contexts and Exceptions for Representing Evolving Knowledge States

Loris Bozzato, Luciano Serafini, and Gaetano Calabrese

Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy

{bozzato,serafini,calabrese}@fbk.eu

Abstract. Context representation of (description logic based) knowledge has recently gained interest in the Semantic Web community and a number of logic based solutions have been proposed. In this regard, in our previous works we have introduced the DL based Contextualized Knowledge Repository (CKR) framework: we recently proposed an extension of CKR with the possibility to represent and reason with defeasible axioms and exceptions in context dependent axioms. The goal of this demo is to demonstrate the use of contexts and exceptions in representing evolving situations: the demo visualizes the evolution of a soccer match by showing the (possibly non-monotonic) changes in the knowledge across different events composing the match.

1 Introduction

The interest in context representation applied to Semantic Web data (and, more in general, in description logic knowledge bases) has led to the proposal of a number of logic based solutions in recent years, like e.g. [4,5,6,7,8,9]. In this direction, in our previous works we have introduced the DL based *Contextualized Knowledge Repository (CKR) framework* [6,2,3]. In the latest formulation of CKR presented in [1], we introduced an extension of the framework with the possibility to reason with defeasible axioms and their *justifiable exceptions* in context dependent axioms.

In this demo, we demonstrate the use of contexts and local exceptions of the CKR framework in representing a scenario evolving across several subsequent knowledge states. We chose to model the example of the evolution of a soccer match: this allows us to expose the potentialities of our framework inside a simple and well-understood scenario, which, on the other hand, provides a rich set of evolving information.

Goal of this demo is to show in practice the use of contexts and their relations for interpretation of local knowledge and its propagation in an event structure. Moreover, we show how a combination of knowledge import axioms and exceptions allows for modelling a non-monotonic evolution of knowledge across states. From the point of view of the implementation, the architecture of the demo system demonstrates how to use the *CKR datalog rewriter* (CKR*ew*) and the reasoning implemented by its datalog translation [1] to compute inferences on an input CKR with defeasibility and integrate this reasoning in a pipeline for managing contextualized OWL/RDF data.

In the following sections, we first briefly summarize the structure and basic ideas of CKR with justifiable exceptions (Section 2), then we present the architecture of the system implemented for this demo (Section 3). Finally, we present the formalization of the soccer match scenario (Section 4) and we give an overview for the usage of the demo web interface (Section 5).

2 CKR with Justifiable Exceptions

We provide an informal summary of the elements of the CKR framework with justifiable exceptions: for a formal description and complete definitions, we refer to [3,1].

A CKR is a two layered structure:

1. the upper layer consists of a knowledge base $\mathfrak{G}$, called *global context*, containing (a) *meta-knowledge*, i.e. the structure and properties of contexts, and (b) *global* (context-independent) *object knowledge*, i.e., knowledge that applies to every context;
2. the lower layer consists of a set of *(local) contexts* that contain locally valid facts and can refer to what holds in other contexts.

The meta-knowledge of a CKR is expressed in a DL *meta-language* $\mathcal{L}_\Gamma$ containing the elements that define the contextual structure: for example, it contains sets of symbols for context names, module names and context classes. Intuitively, context classes represent named classes of contexts, while knowledge modules are pieces of local knowledge associated to a single context or a context class.

The knowledge inside contexts of a CKR is expressed via a DL language $\mathcal{L}_\Sigma$, called *object-language*. The expressions of object language are evaluated locally to each context: every context can interpret each symbol independently. To access the interpretation of symbols inside a specific context or context class, we allow in local object knowledge *eval expressions* of the form $eval(X, \mathsf{C})$, where X is a concept or role expression of $\mathcal{L}_\Sigma$ and C is a concept expression of $\mathcal{L}_\Gamma$. Intuitively, $eval(X, \mathsf{C})$ can be read as *"the interpretation of X in all the contexts of type* C*"*. The global object knowledge in $\mathfrak{G}$ can contain statements $\mathrm{D}(\alpha)$ with $\alpha \in \mathcal{L}_\Sigma$: this states that α is a *defeasible axiom*. Intuitively, in the local contexts, α is applied to all its local instantiations except for those generating a contradiction. In other words, a local "overriding" for some instances of α is tolerated. E.g., $\mathrm{D}(A \sqsubseteq B) \in \mathfrak{G}$ states that *"in general, every A is a B"*: in a local context, the axiom might be contradicted by assertions $S = \{A(e), \neg B(e)\}$ that override the axiom for the "exceptional" individual e.

The semantics of CKR extends the model-based semantics of DL knowledge bases to the two layered structure of the framework and provides the presented interpretation of defeasible axioms and their local exceptions. Intuitively, a CKR interpretation $\mathfrak{I}$ provides DL interpretations $\mathcal{M}$ for $\mathfrak{G}$ and $\mathcal{I}(\mathsf{c})$ for each context c. In the case of strict axioms, models of a CKR verify the contents of global and local modules associated with contexts; global object knowledge is propagated to local contexts. For defeasible axioms, we associate to every context c a set $CAS(\mathsf{c})$ of its *clashing assumptions* $\langle \alpha, \mathbf{e} \rangle$, i.e. local exceptional instances $\mathbf{e}$ of a defeasible axiom α: α is then applied to its local instances except for the ones in $CAS(\mathsf{c})$. An assumption is *justified* if there exists a set S of local assertions showing the local unsatisfiability of the axiom instance (cfr. example above): in CKR models, we require that all clashing assumptions are justified.

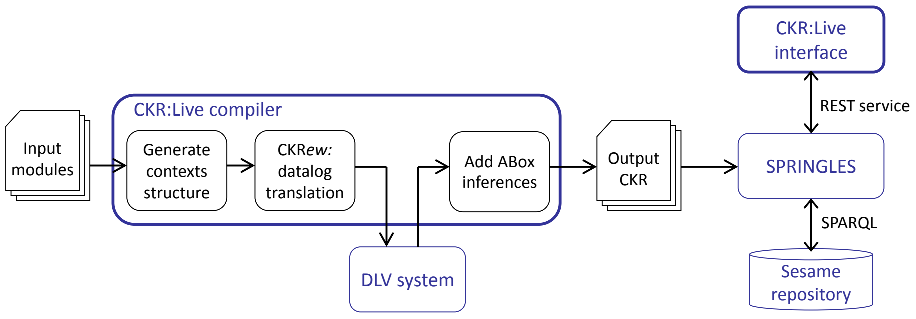


Fig. 1. CKR:Live demo architecture and compilation process

Reasoning in CKRs with defeasibility has been formalized in the form of a translation to datalog: the resulting datalog program is interpreted under answer set semantics, which provides the non-monotonic reading needed for the interpretation of defeasible axioms and exceptions. The datalog translation has been implemented in a CKR datalog rewriter (CKR*ew*) prototype, available at `http://ckrew.fbk.eu/`.

3 CKR:Live Demo System Architecture

The demo system is composed by two components: a *compiler*, which prepares the input RDF files and computes the inferences, and an *interface*, which visualizes the results of the inferences in terms of the presented demo scenario.

The architecture and process of the system is presented in Figure 1. The *compiler* first loads a set of input files containing the ABox information for the global and local modules (i.e. the global and event specific knowledge for the considered soccer match). The compiler then generates the CKR structure by linking each input local module to its intended context and adds a set of fixed and context dependent TBox axioms defining the scenario (see next section) to the knowledge bases of local contexts. The CKR so generated is then passed to CKR*ew*, which computes its datalog translation. By interfacing with the *DLV solver*[1], the resulting answer sets of the datalog translation are computed: the newly inferred ABox facts are then extracted from the computed models and explicitly added to the generated CKR. Finally, the CKR is saved to RDF files.

Such RDF files are then loaded via *SPRINGLES* [3] as a single Sesame repository.[2] The CKR:Live demo web *interface* uses a REST interface to communicate with the SPRINGLES platform: each update to the web interface is reflected by a SPARQL query (targeting the currently visualized context) implemented as a REST service. As we present in Section 5, the web interface provides an intuitive visualization of the evolution across the states of the represented system: it gives the possibility to select specific events and visualize the changes in their associated knowledge.

[1] `http://www.dlvsystem.com/dlv/`

[2] In this system, SPRINGLES is used only for loading the input RDF data as different named graphs and exposing a REST interface: no further inference is applied to the loaded data.

4 Scenario Description: Evolution of a Soccer Match

In the example scenario presented in this demo, our goal is to reason on the evolution of knowledge in the development of a soccer match. In fact, while some information is stable during the match (e.g. the number and playing position of a player), some other changes during the *events* taking place in the course of the match (e.g. the number of yellow cards assigned to a player). Moreover, most of the information increases monotonically (e.g. scored goals), but some other knowledge can be retracted across events: in our example, this is the case of player substitutions, in which a player previously in game is exchanged by a team mate previously waiting at the bench.

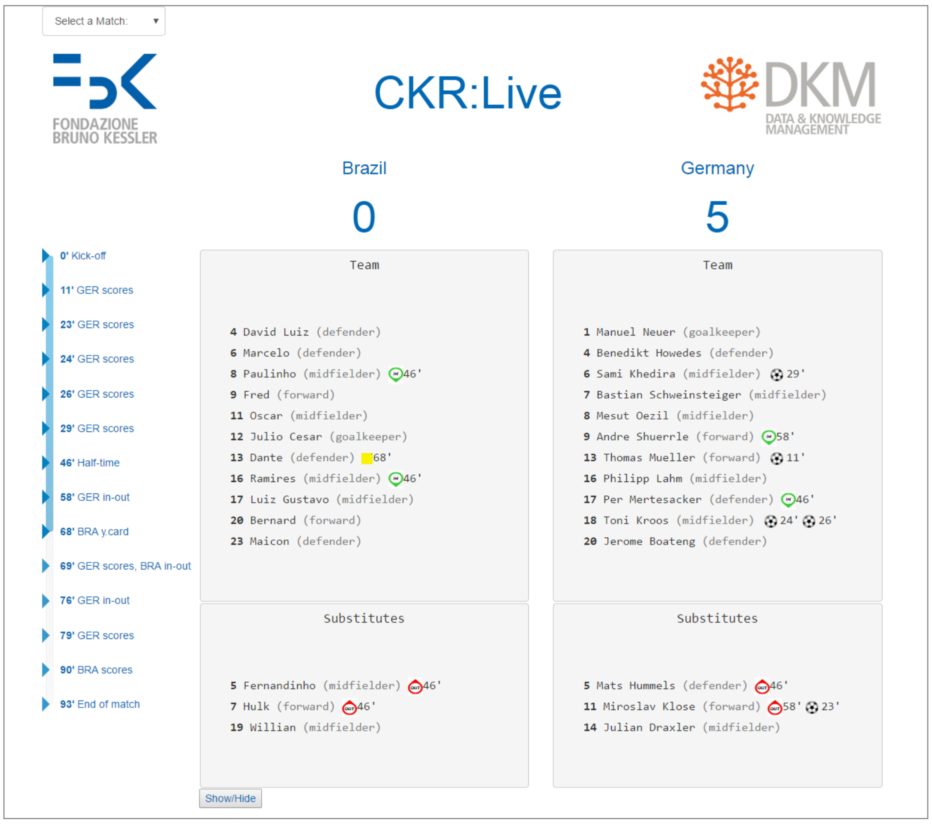
We can now show how we represented this scenario using a CKR. The stable information is contained as facts in the global context as global object knowledge: this consists of the name of the host and home teams and the name, position, team membership and number of each player. Then, local contexts represent events happening during the match: we consider goals, substitutions, card bookings, kick-off, half-time and end of the match. Events are ordered by their time of happening in the match: in other words, they represent the match evolution as a discrete succession of states. In the local knowledge of each event, the new information added by the event happening is specified as ABox assertions: for example, in the case of a goal, the scoring player and team are recorded. In particular, in the first state s_0 corresponding to the kick-off, we store the information about the initial team formations by classifying each Player as $\mathit{PlayingNow}$ (currently in game) or $\mathit{notPlayingNow}$ (currently in the reserves).

The local information of a state needs to preserve the (unchanged) information of its predecessor state: this is achieved by means of *eval* axioms. In the case of monotonic information, using *eval* we can "import" the instances of a predicate from the previous state as members of the same predicate in the current state: for example, in context s_2 we import previous goals using the axiom $\mathit{eval}(\mathit{GoalNow}, \{s_1\}) \sqsubseteq \mathit{GoalNow}$. In the case of non-monotonic information (i.e. substitutions), we use defeasible axioms: in the global context, we include the axioms:

$$\mathrm{D}(\mathit{Player} \sqsubseteq \mathit{Unchanged}) \tag{1}$$

$$\mathit{PlayingBefore} \sqcap \mathit{Unchanged} \sqsubseteq \mathit{PlayingNow} \tag{2}$$

The defeasible axiom (1) states that *"Players are generally unchanged (from previous state)"*, while the strict axiom (2) asserts that *"If a player is unchanged and he was playing before, then he still plays now"*. In local contexts, knowledge about $\mathit{PlayingBefore}$ is obtained again using *eval* axioms: for example, in state s_2 we can specify that $\mathit{eval}(\mathit{PlayingNow}, \{s_1\}) \sqsubseteq \mathit{PlayingBefore}$. In contexts representing substitutions, we can create an exception to the defeasible axiom by asserting that the substituted player is $\mathit{Changed}$ (with $\mathit{Changed} \sqcap \mathit{Unchanged} \sqsubseteq \bot$) and $\mathit{notPlayingNow}$: the first assertion defines an exception to (1) which is then not applied to the player, thus negating the local application of (2). The same mechanism is used for the "defeasible propagation" of information about players currently at the bench (i.e. $\mathit{notPlayingNow}$ instances).



CKR:Live

Brazil 0 — Germany 5

0' Kick-off
11' GER scores
23' GER scores
24' GER scores
26' GER scores
29' GER scores
46' Half-time
58' GER in-out
68' BRA y.card
69' GER scores, BRA in-out
76' GER in-out
79' GER scores
90' BRA scores
93' End of match

Brazil — Team

4 David Luiz (defender)
6 Marcelo (defender)
8 Paulinho (midfielder) 46'
9 Fred (forward)
11 Oscar (midfielder)
12 Julio Cesar (goalkeeper)
13 Dante (defender) 🟨68'
16 Ramires (midfielder) 46'
17 Luiz Gustavo (midfielder)
20 Bernard (forward)
23 Maicon (defender)

Brazil — Substitutes

5 Fernandinho (midfielder) 46'
7 Hulk (forward) 46'
19 Willian (midfielder)

Germany — Team

1 Manuel Neuer (goalkeeper)
4 Benedikt Howedes (defender)
6 Sami Khedira (midfielder) ⚽ 29'
7 Bastian Schweinsteiger (midfielder)
8 Mesut Oezil (midfielder)
9 Andre Shuerrle (forward) 58'
13 Thomas Mueller (forward) ⚽ 11'
16 Philipp Lahm (midfielder)
17 Per Mertesacker (defender) 46'
18 Toni Kroos (midfielder) ⚽ 24' ⚽ 26'
20 Jerome Boateng (defender)

Germany — Substitutes

5 Mats Hummels (defender) 46'
11 Miroslav Klose (forward) 58' ⚽ 23'
14 Julian Draxler (midfielder)

Fig. 2. Screenshot of CKR:Live demo online web interface

5 Online Demo Overview

An online version of the demo is available at `http://bit.ly/ckrlive17`.

In the online demo, we demonstrate the usage of the system on two example matches: *Brazil vs. Germany*, semi-final of 2014 FIFA World Cup[3], and *Italy vs. Spain*, from round of 16 of UEFA Euro 2016[4]. We chose these matches as they provide a fair number of events and event types. Information about the matches has been extracted from the official website of the competitions.

Using the menu at the top of the page, the user can choose one of the represented matches. At the left of the page, a slider gives the possibility to select a state and scroll across the events composing the match: a short descriptive label is associated to each event marker. At the center of the page, all of the knowledge associated to the current

[3] `http://www.fifa.com/worldcup/matches/round=255955/match=300186474/`

[4] `http://www.uefa.com/uefaeuro/season=2016/matches/round=2000744/match=2018002/`

state is visualized: at the top it is shown the current score for each team, while at the center it is shown the current team formations (divided by currently in-game players and substitutes). Each player is listed with his number, name and position: whenever he scores a goal, receives a yellow/red card warning or enters/exits the field in a substitution, an icon and time of happening appear besides its information.

By moving the slider across states, it is possible to note the changes in knowledge contents at each event. For example, consider the first match (*Brazil vs. Germany*): at the initial state, the initial team formation is shown. The first event of the example match is a goal: one can note that its information is (monotonically) preserved when passing to next states. On the other hand, in the event of a substitution, previously in-game players are replaced by reserve players: note for example the case of the *half-time* event in the shown match, in which both teams had multiple player substitutions. Clearly, if a player has not been involved in a substitution, then its position in-game (or at the bench) is preserved to the subsequent states.

By clicking on the *Show/Hide* button, it is possible to have some insights on the underlying data retrieved from the CKR repository: a text box shows the raw results (as JSON records) of the call to the SPRINGLES REST service, corresponding to a SPARQL query about players information inside the currently shown context.

References

1. Bozzato, L., Eiter, T., Serafini, L.: Contextualized knowledge repositories with justifiable exceptions. In: Bienvenu, M., Ortiz, M., Rosati, R., Simkus, M. (eds.) 27th International Workshop on Description Logics (DL2014), Vienna, Austria, July 17-20, 2014. CEUR Workshop Proceedings, vol. 1193, pp. 112–123. CEUR-WS.org (2014)
2. Bozzato, L., Ghidini, C., Serafini, L.: Comparing contextual and flat representations of knowledge: a concrete case about football data. In: Benjamins, V.R., d'Aquin, M., Gordon, A. (eds.) Proceedings of the 7th International Conference on Knowledge Capture (K-CAP 2013), 2013. pp. 9–16. ACM (2013)
3. Bozzato, L., Serafini, L.: Materialization Calculus for Contexts in the Semantic Web. In: Eiter, T., Glimm, B., Kazakov, Y., Krötzsch, M. (eds.) DL2013. CEUR-WP, vol. 1014, pp. 552 – 572. CEUR-WS.org (2013)
4. Khriyenko, O., Terziyan, V.: A framework for context sensitive metadata description. Int. J. Metadata, Semantics and Ontologies 1(2), 154–164 (2006)
5. Klarman, S.: Reasoning with Contexts in Description Logics. Ph.D. thesis, Free University of Amsterdam (2013)
6. Serafini, L., Homola, M.: Contextualized knowledge repositories for the semantic web. J. of Web Semantics 12, 64–87 (2012)
7. Straccia, U., Lopes, N., Lukácsy, G., Polleres, A.: A general framework for representing and reasoning with annotated semantic web data. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI 2010), Special Track on Artificial Intelligence and the Web (July 2010)
8. Tanca, L.: Context-based data tailoring for mobile users. In: Datenbanksysteme in Business, Technologie und Web (BTW 2007), Workshop Proceedings. pp. 282–295. Verlagshaus Mainz (2007)
9. Udrea, O., Recupero, D., Subrahmanian, V.S.: Annotated RDF. ACM Transactions on Computational Logic 11(2), 1–41 (2010)