Licentia: a Tool for Supporting Users in Data Licensing on the Web of Data

Cristian Cardellino[1], Serena Villata[1], Fabien Gandon[1],
Guido Governatori[2⋆], Ho-Pun Lam[2], and Antonino Rotolo[3]

[1] INRIA Sophia Antipolis, France - firstname.lastname@inria.fr
[2] NICTA Queensland Research Laboratory
firstname.lastname@nicta.com.au
[3] University of Bologna
antonino.rotolo@unibo.it

Abstract. Associating a license to data is a fundamental task when publishing data on the Web. However, in many cases data producers and publishers are not legal experts, and they usually have only a basic knowledge about the possible constraints they want to ensure concerning the use and reuse of their data. In this paper, we propose a framework called Licentia that offers to the data producers and publishers a suite of services to deal with licensing information. In particular, Licentia supports, through a user-friendly interface, the users in selecting the license that better suits their needs, starting from the set of constraints proposed to regulate the terms of use and reuse of the data.

1 Introduction

In order to ensure the high quality of the data published on the Web of Data, part of the self-description of the data should consist in the licensing terms which specify the admitted use and re-use of the data by third parties. This issue is relevant both for data publication as underlined in the "Linked Data Cookbook"[1] where it is required to specify an appropriate license for the data, and for the open data publication as expressing the constraints on the reuse of the data would encourage the publication of more open data. The main problem is that data producers and publishers often do not have extensive knowledge about the existing licenses, and the legal terminology used to express the terms of data use and reuse. To address this open issue, we present Licentia, a suite of services to support data producers and publishers in data licensing by means of a user-friendly interface that masks to the user the complexity of the legal reasoning process. In particular, Licentia offers two services: *i)* the user selects among a pre-defined list those terms of use and reuse (i.e., permissions, prohibitions, and obligations) she would assign to the data and the system returns

⋆ NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

[1] http://www.w3.org/2011/gld/wiki/Linked_Data_Cookbook

the set of licenses meeting (some of) the selected requirements together with the machine readable licenses' specifications, and *ii)* the user selects a license and she can verify whether a certain action[2] is allowed on the data released under such license. Licentia relies on the dataset of machine-readable licenses (RDF, Turtle syntax, ODRL vocabulary[3] and Creative Commons vocabulary[4]) available at `http://datahub.io/dataset/rdflicense`. We rely on the deontic logic presented by Governatori et al. [2] to address the problem of verifying the compatibility of the licensing terms in order to find the license compatible with the constraints selected by the user. The need for licensing compatibility checking is high, as shown by other similar services (e.g., Licensius[5] or Creative Commons Choose service[6]). However, the advantage of Licentia with respect to these services is twofold: first, in these services compatibility is pre-calculated among a pre-defined and small set of licenses, while in Licentia compatibility is computed at runtime and we consider more than 50 heterogeneous licenses; second, Licentia provides a further service that is not considered by the others, i.e., it allows to select a license from our dataset and verify whether some selected actions are compatible with such license.

2 Licentia: services for supporting data licensing

Licentia is implemented as a Web service [7]. It is written as a Play Framework application in Scala and Java, using the Model-View-Controller architecture, and powered with SPINdle [3] Java library as background reasoner. The architecture of the Web service is shown in Fig. 1. The workflow is defined by three steps: selection and specification of licensing conditions, reasoning and incompatibility checking, and process and return of results.

Selection and specification of conditions. Using the web interface form, the data producer and publisher specifies a set of licensing conditions she wants to associate to the data. These conditions are divided into three categories: permissions (e.g., Distribution), obligations (e.g., Attribution) and prohibitions (e.g., Commercial Use). The chosen set of conditions is taken by a server side controller, which also gets, through a SPARQL endpoint from a RDF triplestore server containing our licenses repository, a list of the stored licenses and their corresponding conditions. This data is delivered to a module that handles the information and formalizes it into defeasible logic rules for process in SPINdle[8] – a modular and efficient reasoning engine for defeasible logic and modal defeasible

[2] In the interface, we adopt the terminology and the rights of the ODRL vocabulary.
[3] `http://www.w3.org/ns/odrl/2/`
[4] `http://creativecommons.org/ns`
[5] `http://oeg-dev.dia.fi.upm.es/licensius/`
[6] `https://creativecommons.org/choose/`
[7] A demo video of Licentia showing the finding licenses service is available at `http://wimmics.inria.fr/projects/licentia/`.
[8] `http://spin.nicta.org.au/spindle/index.html`

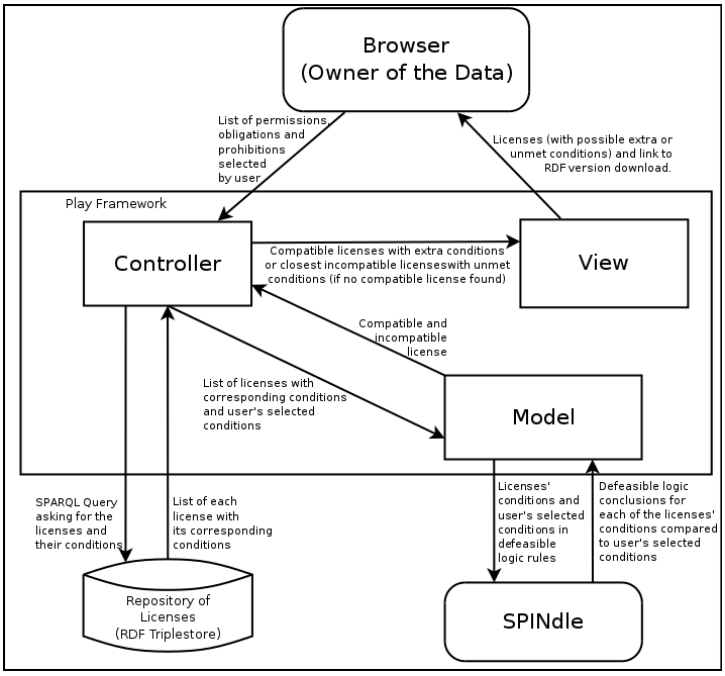

Fig. 1: Licentia Web service architecture.

logic [3]. Licentia is based on the logic and the licenses compatibility verification process proposed by Governatori et al. [2]. The module translates the licenses and the set of conditions from the RDF specification to defeasible logic rules so that SPINdle can reason over them. The module considers every single license in the repository and compares them to the list of conditions selected by the user.

Reasoning and incompatibility checking. SPINdle returns a set of conclusions for each license's conditions compared to the set of conditions selected by the user. From these conclusions, the module gets the set of incompatible conditions chosen by the user with respect to each license: all defeasible non provable rules are incompatible conditions. If this list is empty, then the license is compatible to the set of conditions the user selected. After the module gets all the conclusions for each license, it has two partial results: one containing compatible licenses, and one containing incompatible ones. If the set of compatible licenses is non empty, the module divides this set in two parts: one with those licenses containing the complete set of user's conditions, and the other with those licenses that do not contain all the user's conditions (but still are compatible), highlighting those user's conditions that are not explicitly mentioned in such license. If the set of compatible licenses is empty, the module returns the set of incompatible licenses along with a list highlighting for each license what are the user's conditions that are incompatible with the license.

Process and return of results. If a set of compatible licenses is returned, the controller provides a view listing all the licenses that are compatible and contain the user's selected conditions on the top of the page. Secondly, it returns a list of all other compatible licenses that do not share all of the user's conditions, in ascending order by the number of not contained conditions, highlighting each of the user's conditions that are not explicitly defined in the license. If the set of incompatible licenses is returned, the controller filters all licenses not matching any of the conditions the user selected, keeping those licenses containing at least one of the conditions chosen by the user. If the filtered set is non empty, the system shows a message stating there is no license in the repository compatible with the selected conditions, but there exist some licenses that meet *some of* the conditions. Such licenses are thus listed in ascending order by number of incompatible conditions, highlighting every unmet condition. In any case, the list provides a link to the legal specification of the license as well as a link to a downloadable RDF version of the license. If no compatible license is found then a disclaimer is shown. Note that the evaluation of the performances of the SPINdle[9] reasoning module to verify the compatibility of licensing terms has been presented in [2].

3 Future Perspectives

In this demo, we present the Licentia tool that proposes a set of services for supporting users in data licensing. We are currently finalizing the second service (verification of compatible actions with respect to a specific license), and we are increasing the number of considered machine readable licenses. We plan to extend Licentia by integrating an improved version of the generator of RDF licenses specifications from natural language texts introduced in [1]. Finally, a user evaluation should not be underestimated in order to improve the usability of the user interface.

References

1. Cabrio, E., Aprosio, A.P., Villata, S.: These are your rights - a natural language processing approach to automated rdf licenses generation. In: ESWC. Lecture Notes in Computer Science, vol. 8465, pp. 255–269. Springer (2014)
2. Governatori, G., Rotolo, A., Villata, S., Gandon, F.: One license to compose them all - a deontic logic approach to data licensing on the web of data. In: ISWC. Lecture Notes in Computer Science, vol. 8218, pp. 151–166. Springer (2013)
3. Lam, H.P., Governatori, G.: The making of SPINdle. In: Proceedings of RuleML, LNCS 5858. pp. 315–322. Springer (2009)
4. Maher, M.J., Rock, A., Antoniou, G., Billington, D., Miller, T.: Efficient defeasible reasoning systems. International Journal of Artificial Intelligence Tools 10, 483–501 (2001)

[9] SPINdle has been experimentally tested against the benchmark of [4] showing that it is able to handle very large theories, indeed the largest theory it has been tested with has 1 million rules.