HAL
open science

A Food Packaging Use Case for Argumentation

Nouredine Tamani, Patricio Mosse, Madalina Croitoru, Patrice Buche, Valérie Guillard

To cite this version:

Nouredine Tamani, Patricio Mosse, Madalina Croitoru, Patrice Buche, Valérie Guillard. A Food Packaging Use Case for Argumentation. MTSR: Metadata and Semantics Research, Karlsruhe University of Applied Sciences. DEU., Nov 2014, Karlsruhe, Germany. pp.344-358, 10.1007/978-3-319-13674-5_31 . lirmm-01089612

HAL Id: lirmm-01089612

https://hal-lirmm.ccsd.cnrs.fr/lirmm-01089612

Submitted on 2 Dec 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Food Packaging Use Case for Argumentation

Nouredine Tamani[1,4], Patricio Mosse[2], Madalina Croitoru[1,3,4], Patrice Buche[1,2,4], Valérie Guillard[2,3]

[1] INRIA, France;
[2]IATE, INRA, France;
[3]University Montpellier 2, France
[4]LIRMM, France;

Abstract. Within the framework of the European project EcoBioCap (ECOefficient BIOdegradable Composite Advanced Packaging), aiming at conceiving the next generation of food packagings, we introduce an argumentation-based tool for management of conflicting viewpoints between preferences expressed by the involved parties (food and packaging industries, health and waste management authorities, consumers, etc.). In this paper we recall briefly the principles underlying the reasoning process, and we detail the main functionalities and the architecture of the argumentation tool covering the overall reasoning steps starting from formal representation of text arguments and ending by extraction of justified preferences. Finally, we detail its operational functioning through a real life case study to determine the justifiable choices between recyclable, compostable and biodegradable packaging materials based on stakeholders' arguments.

1 Introduction

Within the framework of the European project EcoBioCap (ECOefficient BIOdegradable Composite Advanced Packaging), we have designed a Decision Support System (called DSS) whose objective is to select packaging materials according to possibly conflicting requirements expressed by the involved parties (food and packaging industries, health authorities, consumers, waste management authority, etc.). The requirements and user preferences are modeled by several ontological rules provided by the stakeholders expressing their viewpoints and expertise.

The DSS software is made of two parts, as depicted in Figure 1:

1. a multi-criteria flexible querying process [2] which takes as inputs desired preferences associated with packaging characteristics (permeability, shape, dimensions, desired shelf life, ...) and retrieves from a packaging database a ranked list of the most relevant packagings.
2. an argumentation process which aims at aggregating several stakeholders (consumers, researchers, food industry, packaging industry, waste management policy, etc.) requirements expressed as simple textual arguments, to enrich the querying process by stakeholders' justified preferences. Each argument supports/opposes a choice justified by the fact that it either meets or not a requirement according to packaging aspects (biodegradability, recyclability, transparency, ...).

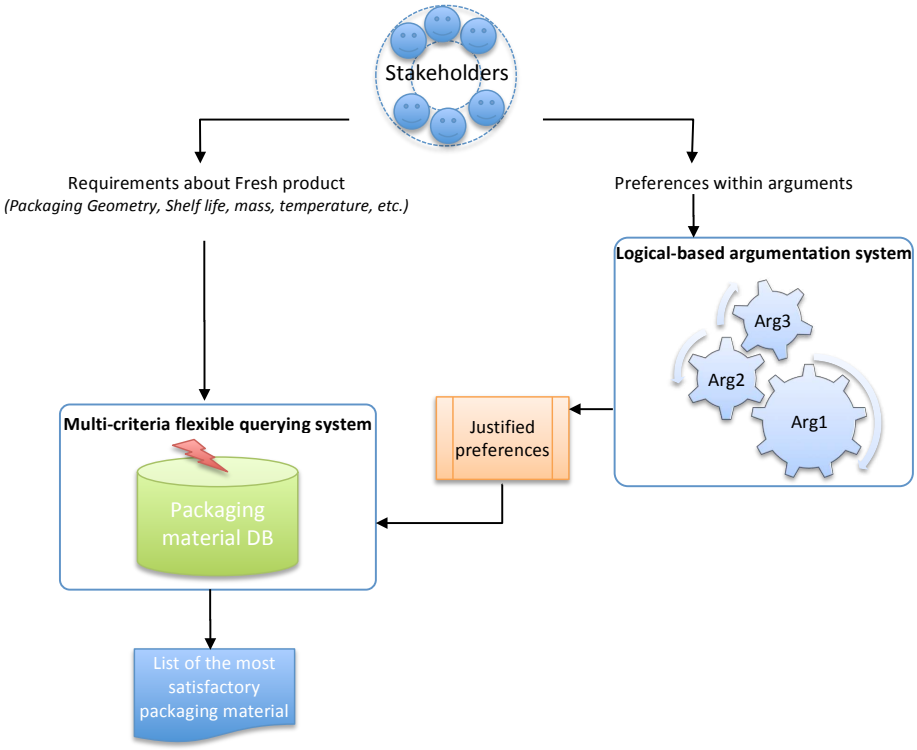

Fig. 1. Global insight of the DSS.

Thus, our approach consists of two steps: (i) aggregating possibly conflicting preferences expressed by the involved stakeholders (ii) querying a database of packagings with the resulting aggregated preferences obtained at point (i). Indeed, packagings have to be selected according to several aspects or criteria (permeance, interaction with the packed food, end of life, etc.), highlighted by the expressed stakeholders' arguments.

In this paper we detail the implementation of the argumentation system. This module has as inputs stakeholders' arguments supporting or opposing a packaging choice which could be seen as preferences combined with their justifications, and returns consensual preferences which may be candidates to enrich the bipolar querying system. From the argumentation-based software standpoint, based on the recent survey of [8] and the web site *http://www.phil.cmu.edu/projects/argument_mapping/*, argumentation tools could be divided into the two following categories:

– Software for argument expression and modeling. These software such as Araucaria [6], Argunet [7] and DebateGraph (www.debategraph.org) allow the expression of arguments in a text format and manually formalizing them as logical implications made of hypothesis and conclusions. The user can after that save the arguments as an XML file.

– Software for extension computation (an extension is defined as a conflict-free subset of arguments defending themselves against attacks computed under a considered semantics [3]) over an argumentation graph given as input, like OVA-GEN and ArguLab.

Despite the plethora of the software available in the field of argumentation, there are few argumentation software implementing an argumentation process from argument expression to extensions computation, while providing users with several Graphical User Interfaces to visualize the entire process. Unlike ArgTrust [4] in which the authors considered the uncertainty underlying the sources of the knowledge used in the argumentation framework for decision making, we introduce in this paper a real world application based on argumentation reasoning and combining a querying process, which exploits the result of the argumentation process as justified preferences expressing consensual solutions that meet the stakeholders needs and requirements.

Section 2 summarizes the main functionalities of the tool. Section 3 details the architecture of the argumentation tool. Section 4 introduces a real use case for packaging selection. Section 5 describes the implementation of the approach and Section 6 concludes the paper and sums up some future work.

2 User Requirements

We detail hereinafter the main functions of the argumentation system. After discussions and interviews with the project partners, we have identified some requirements summarized in the following functionalities:

– *Formalize text arguments*: the argumentation system should provide users with a user-friendly interface allowing them to express their arguments as text and then formalizing them as concepts and rules. Here, a concept is either a *concrete* concept defined over some attributes of the packaging material database for which values (numerical, intervals or boolean) can also be specified, or an *abstract* concept which is not related to any attribute in the database and only used in the reasoning process, as illustrated in Example 1.

  Example 1. The following text argument in favor of recyclable material *"Recyclable packagings are advised since they protect the environment according to the life cycle analysis"* can be modeled by the following concepts and logical rules:
  - `RecyclablePack`: a concrete concept corresponding to recyclable packaging materials. It is defined over the boolean attribute *Recyclable* from the packaging materials database with the value `true`,
  - `ProtectEnvPack`: an abstract concept corresponding to packaging materials which protect the environment,
  - $RecyclablePack \Rightarrow ProtectEnvPack$ is a logical rule (implication) expressing the fact that any recyclable packaging is a packaging which protects the environment,
  - $ProtectEnvPack \Rightarrow AcceptedPack$ is a logical rule expressing the fact that the decision of acceptance is attached to each packaging protecting the environment.

The system is also equipped with a function of import/export formalized arguments from/into an XML format. So, we can load already formatted concepts and rules directly in the system or obtain a local copy of the current project.

– *Process arguments*: the system should automatically compute the logical arguments obtained from the set of concepts and rules.

  Example 2 (example of a logical argument). The text argument of Example 1 is automatically translated into a logical argument made of the following three steps of reasoning:
  - Choice $C_1 = RecyclablePack$,
  - By the rule "$RecyclablePack \Rightarrow ProtectEnvPack$" we get the conclusion: C_1 is $ProtectEnvPack$,
  - By the rule "$ProtectEnvPack \Rightarrow AcceptedPack$" we get the final conclusion: C_1 is $AcceptedPack$.

  The arguments can be gathered into pros and cons with regard to some packaging alternative characteristics. Once logical arguments are built, the system computes all conflicts or attacks among them and draws the corresponding argument graph.

  Example 3 (example of conflict). The following argument *"recyclable packagings are not advised since they need to provide collect and treatment facilities, which could be very expensive"* is in conflict with the argument of Example 1.

– *Compute extensions*: an extension is a subset of non-conflicting arguments, which defined themselves against attacks, defined according to one semantics (admissible, preferred, grounded, stable, etc. see [3] for more details about semantics). The current version of the system implements different kinds of semantics. The user can compute one particular semantics or all the implemented semantics.
– *Enrich the bipolar querying*: based on the obtained extensions, the system extracts the criteria leading to either the rejection or the acceptance of some packaging types. These criteria and eventually associated values become predicates (conditions) which can be used latter as contraints or wishes to enrich the bipolar query which can be processed by the flexible querying system.

In addition to the above functions, the argumentation system must deal with interaction concerns such as:

– *multi-users*: the system must allow a real-time discussion among stakeholders. Every stakeholder has an account and a password. After logged in the system, a stakeholder can browse the current project, open one of them and join the discussion by adding or updating the expressed arguments.
– *persistence*: in the sense that the system saves in a database the ongoing or old projects (concept and rules) and already expressed concepts and rules and makes them available and accessible to the stakeholders to define quickly their own arguments.
– *informative*: the user can access to a log describing further details about the current state of the system including current users, complete description of logical arguments and conflicts.

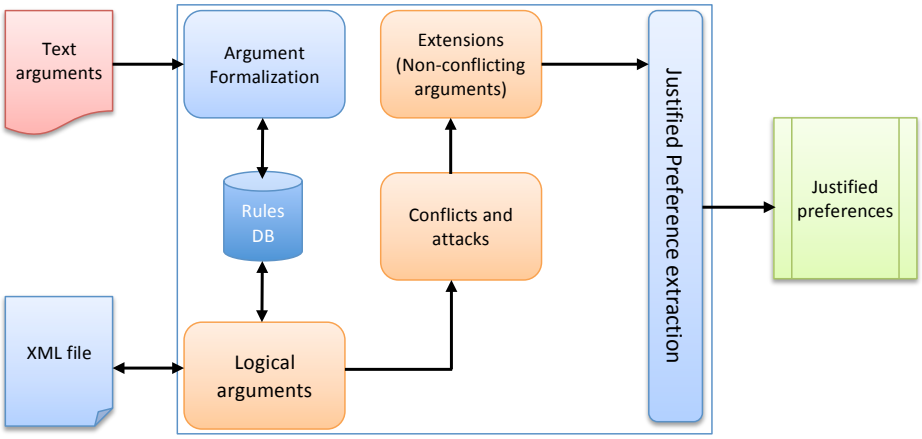

Fig. 2. The architecture of the argumentation system.

– *configurable*: regarding to the amount of information the user can face during an argumentation session, the system can be run in either *expert* or *user* mode. The former allows to display all information about the process (argument graph, attack graph, conflicts, extensions, etc.) and the latter limits the information to the most relevant one (conflicts and extensions).

3 Architecture of the argumentation system

As illustrated in Figure 2, the proposed argumentation system relies on 5 main modules, described below.

– *Argument formalization module*: this module implements a user-friendly interface for an interactive translation of text arguments into a formal representation made of concepts and rules (claims and hypothesis). The user has just to specify the part of the text argument corresponding to the claim and the part corresponding to the conclusion. Each part is modeled as a concept which could be either supported or not by the database. A graphical representation of the expressed rules is also built as the users formalize their text arguments. The formal representation obtained is finally saved in a database for a persistent storage allowing to reload argumentation projects without rebuilding all the arguments and to also reuse the already formatted rules in other projects.
– *Logical arguments*: this module receives as inputs the list of concepts and rules corresponding to text arguments. This list can be the result of the formalization module or given by the user as an XML file. As in [11, 12] and by using a derivation process, this module builds all possible arguments according to the logical process defined in ASPIC/ASPIC+ logic-based argumentation frameworks [1, 5]. This module also allows to export the argument list into an XML document.

- *Conflicts and attacks*: this module relies on the logical arguments built by the previous module. According to the negation operator, it detects all the conflicts among arguments and models them as attacks with respect to the definition of attacks introduced in [11, 12]. The output of this module is an argumentation graph made of arguments (nodes) and attacks (edges).
- *Extensions*: an extension is a subset of non-conflicting (consistent) arguments which defend themselves from attacking arguments. The computation of extensions is made under one semantics (preferred, stable, grounded, eager, semi-stable, semi-stable, ideal) as defined in [3]. This module allows the computation of one or all semantics considered. We notice that theoretically we can get empty extensions under any semantics. This situation occurs when a user expresses at least one self-defeated argument, which is not attacked by any other argument, but attacks all the others. This kind of arguments are called contaminating arguments [13]. The current version of the system detects the rules leading to such arguments and discards them before performing the process of extension computation.
- *Extraction of the justified preferences*: the computation of extensions delivers one or several extensions. In the case of several extensions, the system lets the users selecting the most suitable one according to their objectives. The selected extension is then used to extract the preferences underlying the contained concepts. These preferences are translated into a list of couples $(attribut, value)$, where $attribute$ stands for a packaging attribute as defined in the packaging database schema of the flexible querying system part of the DSS, and $value$ is the preferred value expressed for the considered attribute. More details are provided in section 5.

4 Use Case

This section details a use case employed throughout the project and in this paper for exemplification reasons. We consider the following arguments expressed by the stakeholders obtained by interviews and surveys.

1. Packaging materials with low environmental impact are preferred, low environmental impact corresponds to carbon footprint of value $[0, 10]\ kg\ CO_2$,
2. Waste management authority aims at collecting at least $75\,\%$ of recyclable packaging,
3. Consumers are unwilling to sort packaging cause of its extra taxe,
4. Life Cycle Analysis (LCA) results are not in favor of biodegradable and compostable materials,
5. Consumers are in favor of biodegradable material because they help to protect the environment,
6. Biodegradable materials could encourage people to throw their packaging in nature, causing visual pollution,
7. Micro-perforated packaging can increase the shelf life by about 20 days,
8. Multilayered byproduct made packagings allow a good permeance,
9. Biodegradable and compostable are expensive to product,
10. Consumers do not want pay an extra cost greater than 5% for a product packed with biodegradable or compostable packaging,

11. Recycling creates new jobs and encouraged by the waste management local administration.

Given the different concerns expressed by the experts, we have split the above arguments into several viewpoints. Focusing on a specific point of view or a concern helps experts to concentrate and to elicit knowledge. Such restriction is explained by (i) the huge number of arguments generated during the software testing phase, and (ii) the need for associating the packaging attributes with the argumentation process in its early stages. So, a viewpoint corresponds to one or several packaging attributes. In the current version of the system, argument splitting is made manually by the users when they express their arguments. In the above list of arguments, we can distinguish the following viewpoints:

- *end of life*: in this viewpoint, stakeholders (waste management authority, users, researchers) argue between biodegradability, compostability and recyclability of the packaging from their environmental effects. It contains arguments 1 to 6,
- *shelf life*: this viewpoint contains arguments 7 and 8, the choice is between multi-layered packagings or micro-perforated packagings,
- *economic*: this viewpoint deals with arguments expressed on the economic concerns such the extra cost of the final product, and the effect on employment. It contains arguments 9 to 11.

5 Implementation of the approach

The implementation of the approach has been done in the context of the EcoBioCap DSS. A java GXT/GWT web interface was developed and a open version is accessible on *http://pfl.grignon.inra.fr/EcoBioCapProduction/*. Hereinafter, some user interfaces are displayed showing the obtained result in the case of the viewpoint "end of life". The main interface of the system is illustrated in Figure 3 which gives access to the functions of the **Logical-based argumentation system** in Figure 1 and detailed in Figure 2. It is divided into 5 zones. Zone 1 corresponds to the task bar implementing general functions applied on projects (create, load, close, refresh, export, etc.). Zone 2 lists the text arguments by stakeholders. Zone 3 displays the extracted concepts and rules from the text arguments, they are also listed by stakeholders. Zone 4 displays the graphical representation of the formalized concepts and arguments. Zone 5 is a notification area displaying the computed conflicts and extensions.

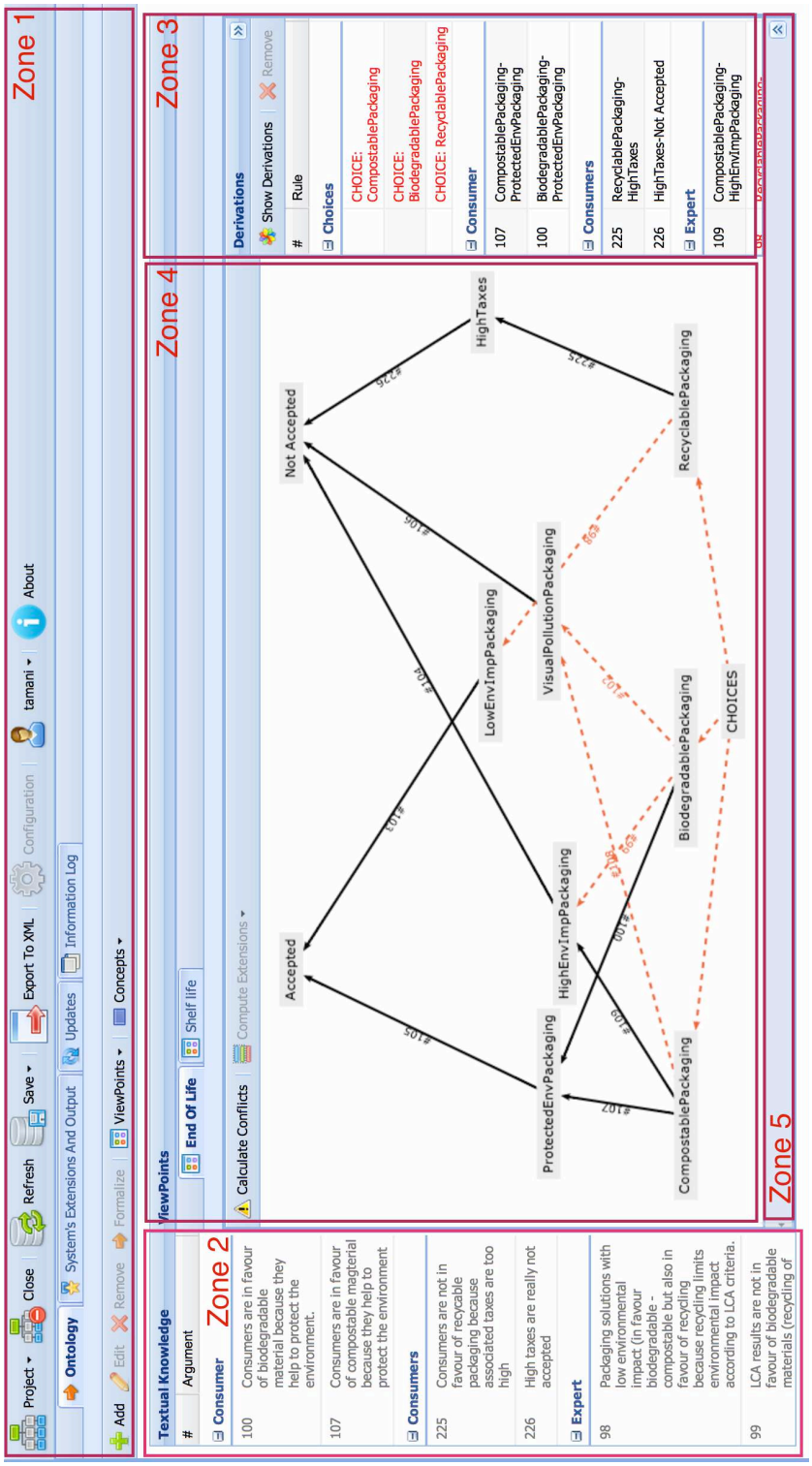

Fig. 3. Main interface of the argumentation system.

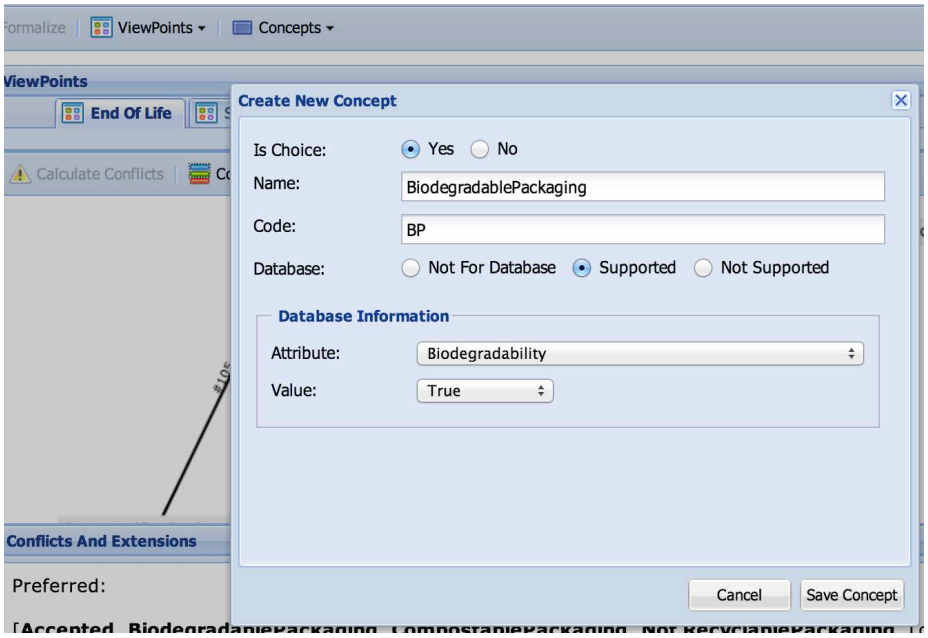

Fig. 4. Add a concept based on a defined attribute in the packaging database.

After logging in, the user can create a new project, load an existing one or import a new project from an XML file. So, in the freely available version, stakeholder' arguments can be entered as (i) an XML file, by using the *import from XML* function, or (ii) text arguments to formalize then as concepts and rules by using a dedicated user interface (Figures 4, 5, 6 and 7) guiding and helping the user during all the process of formalization (implementing the **Argument Formalization** module in Figure 2). A new concept has a name and a short code, it can be defined as either a choice or not and can be related to a packaging attribute (as in Figure 4 for `BiodegradablePackaging` corresponding to packagings having the attribute `Biodegradability = true` in the packaging database), not related to any information in the database (as in Figure 5 for the concept `HighTaxes`), or can suggest a new attribute to enrich the packaging description in the database (as in Figure 6, concept `HighEnvPackaging` suggests the new attribute `CarbonFootPrint`, with the measure unit of $Kg\ CO_2\ eq.$ to describe packaging).

Figure 7 shows the formalizing interface in which a user can specify the already created concepts as premise or conclusion to form the rule underlying the text argument. The rule is then connected to a decision (`accepted, not accepted`). The rule and its decision can be specified either strict or defeasible.

Figure 8 illustrates the obtained ontology in the case of the viewpoint *end of life* in which stakeholders argued about biodegradability, recyclability and compostability. This ontology is the input of the **Logical Arguments** module in Figure 2.

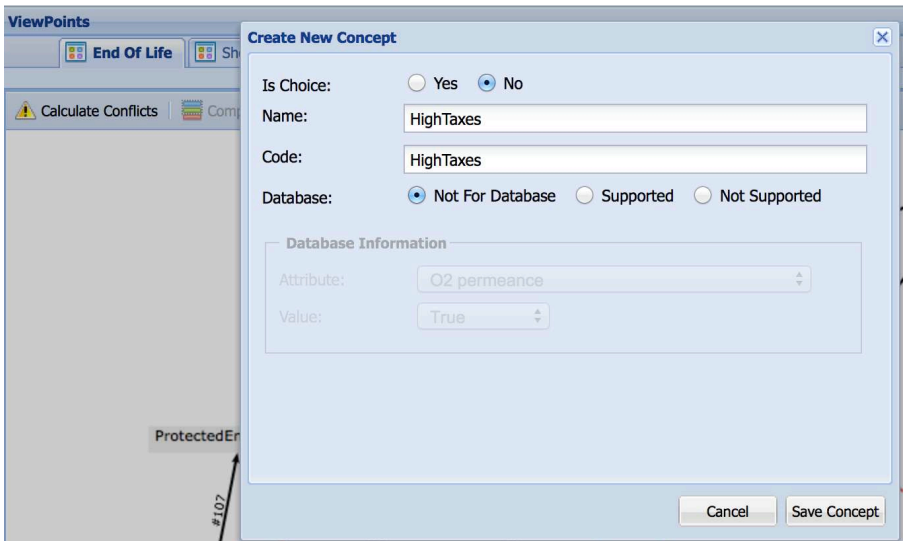

Fig. 5. Add a concept based which is not related to the database.

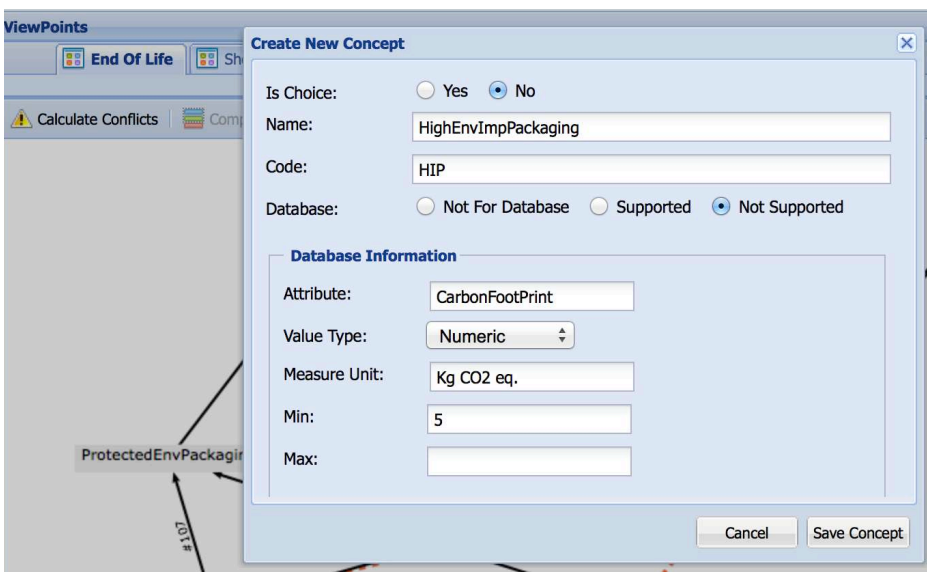

Fig. 6. Add a concept not currently supported in the packaging database but suggested for addition.

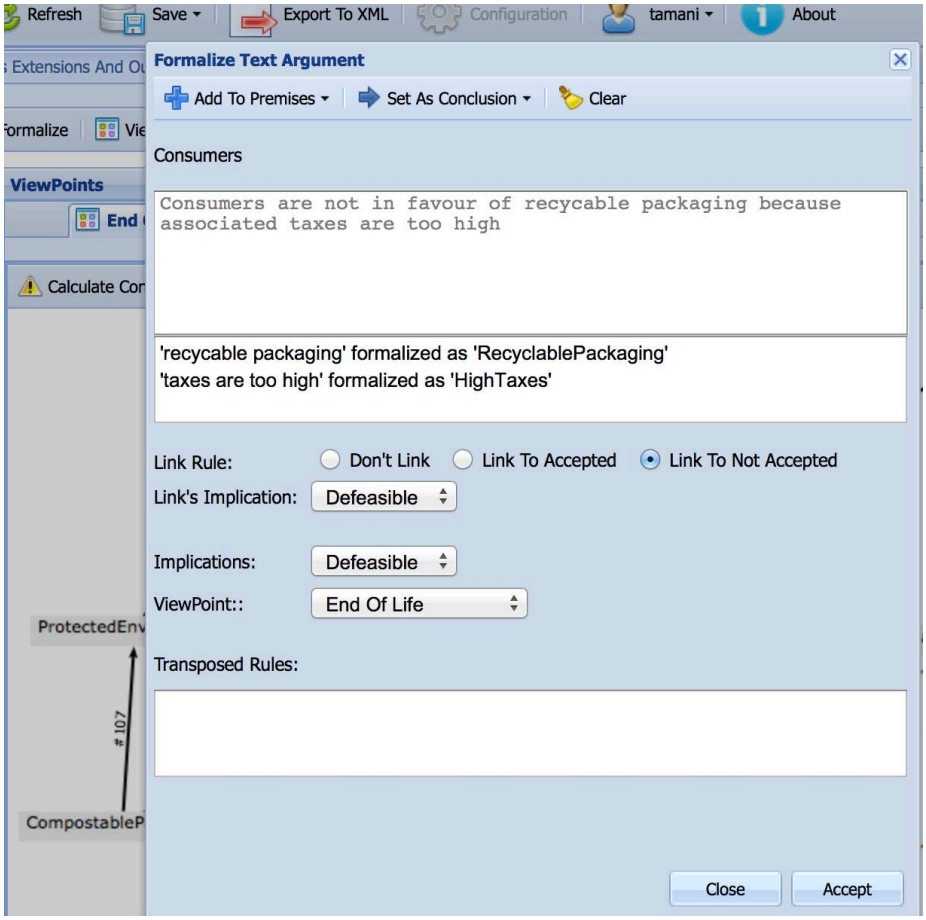

Fig. 7. Formalizing a text argument as concepts and rules.

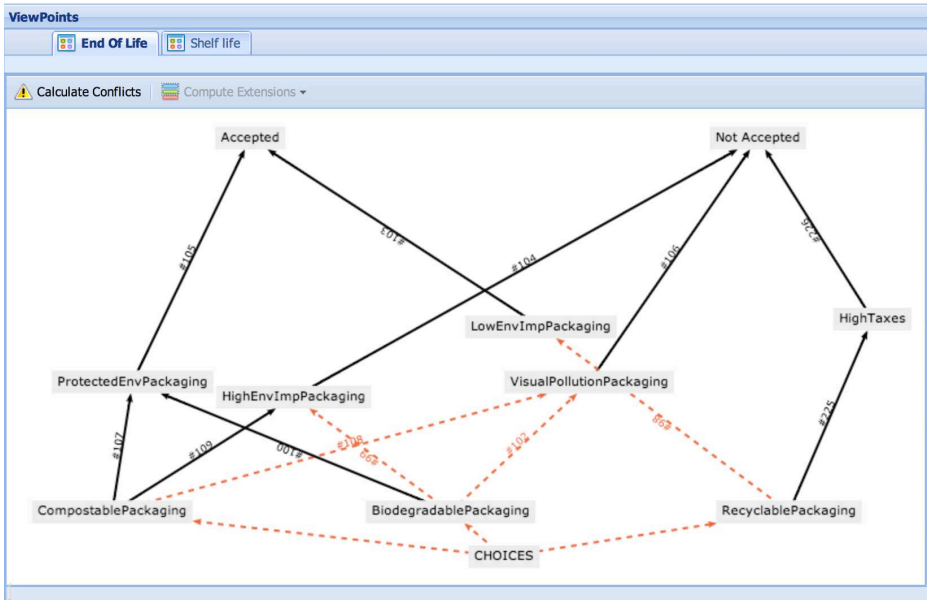

Fig. 8. Example of an ontology built upon the viewpoint *end of life*.

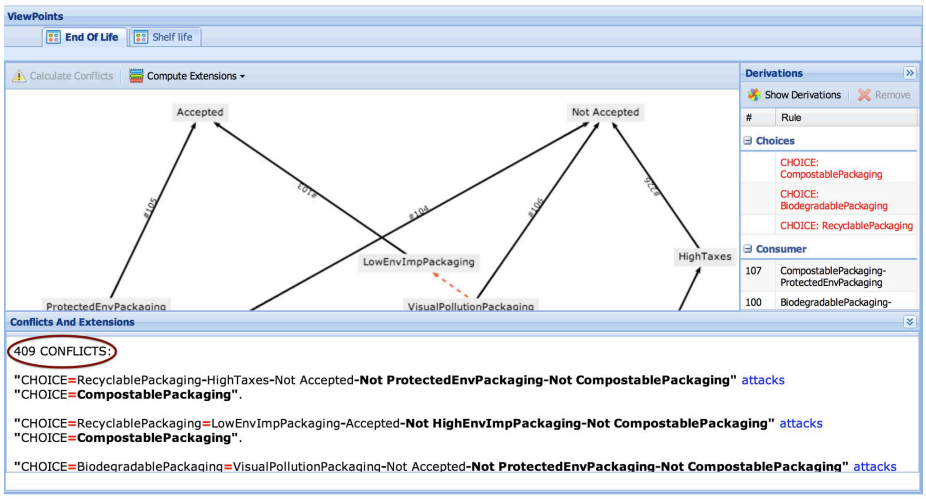

Fig. 9. Conflicts computed in the viewpoint *end of life*.

The system generates arguments and computes conflicts and attacks as depicted in Figure 9 corresponding to the result of the **Conflicts and Attacks** module in Figure 2. For the arguments of the "end of life" viewpoint, the system detected 409 conflicts. The extensions under different semantics (stable, preferred, admissible, grounded, naive) are after that computed and their contents are displayed to the user in Figure 10 (implementing the **Extensions** module in Figure 2).

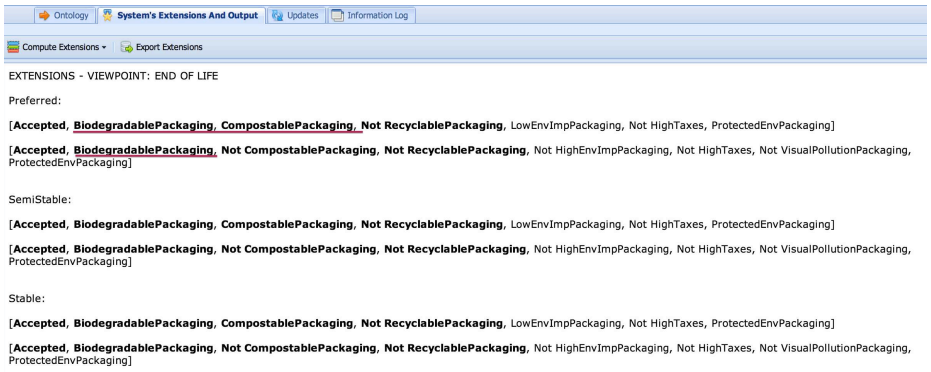

Fig. 10. Delivered extensions in the *end of life* viewpoint.

We notice in Figure 10 that the system concludes skeptically that biodegradable packagings are the most justified ones. The obtained extensions are then stored as a list of $attribute = value$ (Figure 11) to be used in the flexible querying system in addition

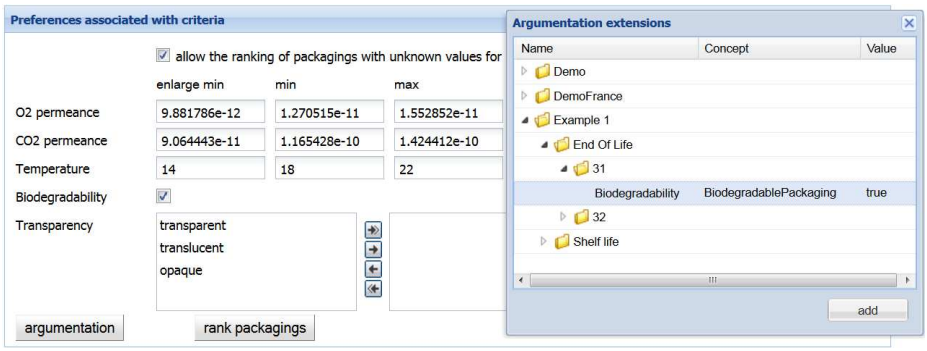

Fig. 12. Selecting the preferences associated with the *end of life* view point to complete the query with $Biodegradable = true$.

to some other parameters useful for the querying process. In the context of *end of life* viewpoint, the condition $Biodegradable = true$ is sent to the querying process to be used as a justified preference for packaging material selection.

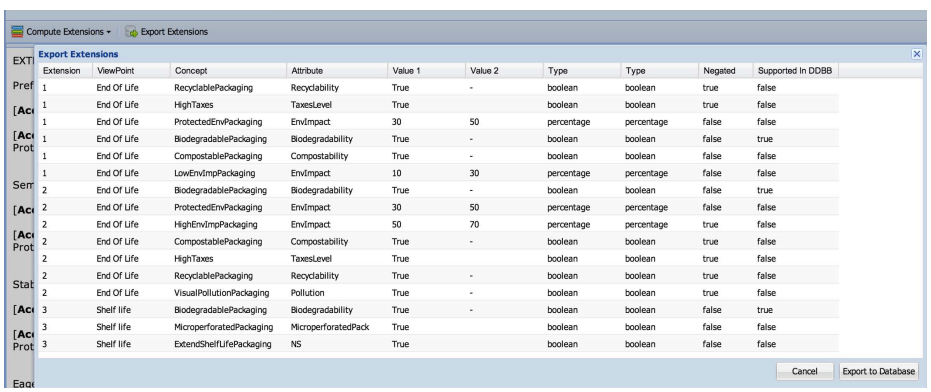

Fig. 11. Export delivered attributes to a database for the querying process.

The user can actually select the extensions, previously translated into couples $attribute = value$, from the graphical user interface of the flexible multi-criteria querying system as displayed in Figure 12. Figure 13 finally displays the result after the execution of the multi-criteria querying which takes into account the aggregated preferences about biodegradability attribute. Four packagings are delivered and listed according to their relevance to the query preferences.

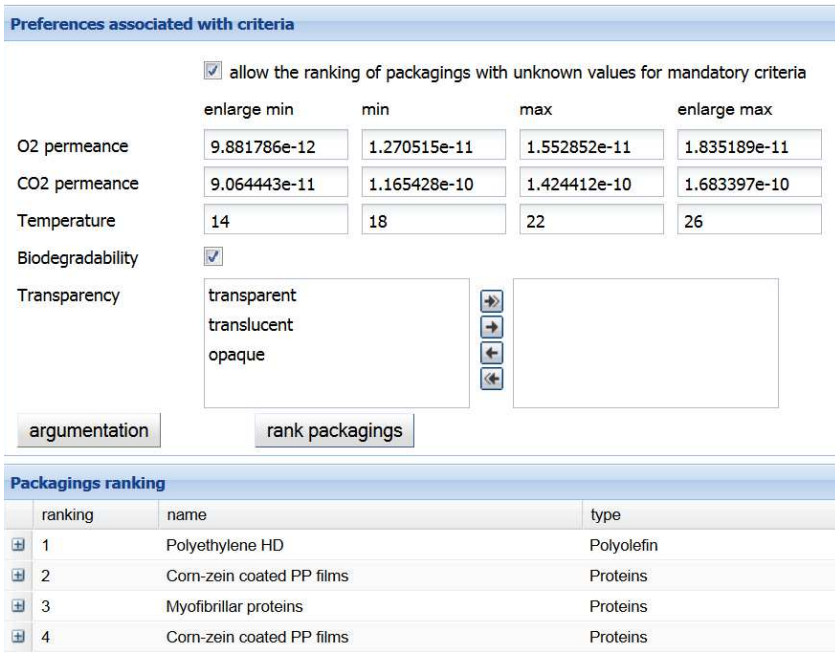

Fig. 13. The final result after running the multi-criteria querying process.

6 Conclusion

We applied in this paper an argumentation approach on a real use case from the industry, based on a combination of an ASPIC argumentation system with a DLR-Lite specifications allowing stakeholders to express their preferences and providing the system with stable concepts and subsumptions of a domain. We have proposed an argumentation system in which each criterion (attribute or aspect) is considered as a viewpoint in which stakeholders express their arguments in homogenous way. Each viewpoint delivers extensions supporting or opposing certain choices according to one packaging aspect, which are then used in the querying process. The approach is finally implemented as freely accessible web application and a demonstration of the tool can also be provided. Some feedback obtained from test users point out the difficulties to consider a rule as either strict or defeasible and expressed the need to be able to specify a sort of importance encompassing the notions of strictness and defeasibility. So, one work in progress is to extend the proposed approach to fuzziness to make it possible to deal with vague and uncertain concepts and rules [10, 9].

7 Acknowledgements

The research leading to these results has received funding from the European Community's seventh Framework Program (FP7/ 2007-2013) under the grant agreement

n°FP7-265669-EcoBioCAP project. The authors would like to thank also the partners of the project for the help provided during the argument elicitation.

References

1. L. Amgoud, L. Bodenstaff, M. Caminada, P. McBurney, S. Parsons, H. Prakken, J. Veenen, and G. Vreeswijk. Final review and report on formal argumentation system.deliverable d2.6 aspic. Technical report, 2006.
2. S. Destercke, P. Buche, and V. Guillard. A flexible bipolar querying approach with imprecise data and guaranteed results. *Fuzzy sets and Systems*, 169:51–64, 2011.
3. P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-persons games. *Artificial Intelligence*, 77(2):321–357, 1995.
4. S. Parsons, E. Sklar, J. Salvit, H. Wall, and Z. Li. Argtrust: decision making with information from sources of varying trustworthiness. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 1395–1396. International Foundation for Autonomous Agents and Multiagent Systems, 2013.
5. H. Prakken. An abstract framework for argumentation with structured arguments. Technical report, Department of Information and Computing Sciences. Utrecht University., 2009.
6. C. Reed and G. Rowe. Araucaria: Software for argument analysis, diagramming and representation. *International Journal on Artificial Intelligence Tools*, 13(04):961–979, 2004.
7. D. C. Schneider, C. Voigt, and G. Betz. Argunet- a software tool for collaborative argumentation analysis and research. In *7th Workshop on Computational Models of Natural Argument (CMNA VII)*, 2007.
8. J. Schneider, T. Groza, and A. Passant. A review of argumentation for the social semantic web. *Semantic Web*, 4(2):159–218, 2013.
9. N. Tamani and M. Croitoru. Fuzzy argumentation system for decision support. In *IPMU'14*, pages 77–86, 2014.
10. N. Tamani and M. Croitoru. A quantitative preference-based structured argumentation system for decision support. In *Fuzz-IEEE*, pages 1408–1415, 2014.
11. N. Tamani, M. Croitoru, and P. Buche. A viewpoint approach to structured argumentation. In M. Bramer and M. Petridis, editors, *The Thirty-third SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 265–271, 2013.
12. N. Tamani, M. Croitoru, and P. Buche. Conflicting viewpoint relational database querying: an argumentation approach. In L. Scerri and B. Huhns, editors, *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014)*, pages 1553–1554, 2014.
13. Y. Wu. *Between argument and conclusion. Argument-based approaches to discussion. Inference and Uncertainty*. PhD thesis, Université du Luxembourg, 2012.