Automated Ontology Evolution for an E-Commerce Recommender[1]

Elmar P. Wach

STI Innsbruck/ University of Innsbruck
Elmar/P/Wach eCommerce Consulting
Technikerstraße 21a/ Hummelsbüttler Hauptstraße 43
6020 Innsbruck/ 22339 Hamburg
elmar.wach@sti2.at/ wach@elmarpwach.com

Abstract: This research proposes an automated OWL product domain ontology (PDO) evolution by enhancing an existing ontology evolution concept. Its manual activities are eliminated by formulating an adaptation strategy for the conceptual aspects of an automated PDO evolution and establishing a feedback cycle. The adaptation strategy was validated/ firstly "instantiated" by applying it to a real-world conversational content-based e-commerce recommender as use case.

1 Introduction

Recommender systems in e-commerce applications have become business relevant in filtering the vast information available in e-shops (and the Internet) to present useful recommendations to the user. As the range of products and customer needs and preferences change, it is necessary to adapt the recommendation process. Doing that manually is inefficient and usually very expensive. Recommenders based on product domain ontologies[2] (PDO) can extract questions about the product characteristics and features to investigate the user preference and eventually recommend products that match the needs of the user. By changing the PDO, such a recommender generates different questions and/ or their order. Hence, an automated adaptation of the recommendation process can be realised by automatically evolving the PDO. The high cost of the manual adaptation of the recommendation process and the underlying PDO can herewith be minimised. The research question is: How can an automated[3] PDO evolution be realised based on feedback? The present research tackles an automated process for the first time (to the best knowledge of the author).

---

[1] The research presented in this paper is funded by the Austrian Research Promotion Agency (FFG) and the Federal Ministry of Transport, Innovation, and Technology (BMVIT) under the FIT-IT "Semantic Systems" program (contract number 825061)

[2] A product domain ontology (PDO) is defined as the formal, explicit specification of a shared conceptualisation of a product description based on OWL DL; this definition is derived from [Gr93]

[3] Without human inspection

2 Related Work

Previous approaches to the topic of this research can be found in concepts for ontology evolution, e.g. [Ha05], [KN03], [Ko07], [No06], [St02], [St03], [Za09]. This research focuses on enhancing the concept of [St02], as it is the closest work to the research in this paper. They focused on the evolution process and have defined six phases consisting of capturing, representation, semantics of change (i.e. a rich description about the semantic role of an ontology entity in order to get more information for solving inconsistencies), implementation, propagation, and validation of ontology changes. This process is implemented in the KAON[4] framework and the Ontologging[5] system. Evolution strategies have been formulated defining elementary and composite changes for executing a change request and eventually deciding the evolution path. In the six phase evolution process, two steps include manual activities, namely (i) "implementation" in which the implications of an ontology change are presented to the user and have to be approved by her before execution, and (ii) "validation" in which performed changes can get manually validated. Both manual steps are eliminated with the adaptation strategy and its implementation. To automate (i), the PDO evolution is conceptualised and implemented as a complete feedback cycle. An insufficient PDO change is indicated by decreased metrics and gets revised according to the evolution strategy chosen. Hence, the PDO changes do not have to get manually approved before execution. To automate (ii), the PDO changes are predefined and application-oriented. Hence, only valid changes are executed, and nobody has to manually validate them.

3 Approach and Proposed Solution

The aim of this research is to combine the use of PDO with processing user feedback. The work focuses on how the given feedback can lead to a self-improvement of the semantic application by adapting the PDO. In this context self-improvement means that by automatically processing user feedback and evolving the PDO, the defined key performance indicators (KPI) of the application will increase. For this, a six step adaptation strategy for the conceptual aspects of an automated PDO evolution has been formulated and a feedback cycle established. The adaptation strategy answers the questions when and how to evolve the PDO by evaluating the impact of the evolution in the precedent feedback cycle and is implemented in two components constituting a new adaptation layer. The first question defines the (temporal and causal) trigger initiating the PDO change. Basically, this is receiving and transforming the feedback into ontology input (i.e. calculating Success Trends ST) and will be addressed with the feedback transformation strategy. This strategy is implemented in the Feedback Transformer component. After having transformed the different feedback types, the calculated ST are reported to the next component, i.e. Adaptation Manager.

---

[4] http://kaon.semanticweb.org
[5] European Commission project IST-2000-28293

The second question defines the changing of the PDO with annotated instances. This is evolving the PDO and will be addressed with the PDO evolution strategy. This strategy is implemented in the Adaptation Manager. Due to space limitations, the adaptation strategy is not elaborated in this paper. The strategy is used to concisely describe the application for which the automated PDO evolution should be implemented and the impacts of PDO changes on the application behaviour. The interested reader is referred to [Wa11].

The automated ontology evolution is realised by utilising an evolution heuristic and evolution strategies. Both are briefly introduced. Those are defined in the fifth step of the adaptation strategy "Decide the adequate PDO evolution". The evolution heuristic (confer section 4) defines the PDO change to be executed. [GL97] introduced the tabu search metaheuristic which is utilised in this research with the philosophy that the highest precedent ST ("greedy") defines the next PDO change to always choose the best evolution. The tabu search enhances a local search (i.e. iteratively improving a criterion in the search space) metaheuristic by using "taboos" – a solution is not executed again according to the criteria defined in the tabu list.

This research proposes to additionally formulate evolution strategies that decide the general evolution behaviour (e.g. executing the same type of PDO change or a rollback) by correlating the types of PDO changes needed to the ST calculated. The philosophy of the evolution strategies is that the development (and its strength) of the precedent ST defines the next type of PDO change to distinguish different evolution impacts. The predefined evolution strategies summarised in table 1 are considered as basic categories. They can be fine-tuned with regard to the associated types of PDO changes as well as the threshold defining the trend significance.

Table 1: Evolution strategy, Success Trend ST, and associated type of PDO change

Evolution Strategy	Decision Criteria	Type of PDO Change
Risky Evolution ("always evolve differently")	$-1 \leq ST \leq 1$	Different than before
Progressive Evolution ("learn from the past")	$0{,}2^* \leq ST \leq 1$ $0 \leq ST < 0{,}2^*$ $-1 \leq ST < 0$	Same as before Different than before Different than before or Rollback
Safe Evolution ("only revert negative trends")	$0 \leq ST \leq 1$ $-1 \leq ST < 0$	None Rollback
Rollback ("undo the PDO changes")	Manually	Rollback

* Threshold trend significance: Increase of the ST by 20 basis points between the precedent and the current feedback cycle

Each evolution strategy besides Rollback ensures an adaptive change of the PDO. By selecting a strategy in the administration interface, the business manager decides how fundamental the evolution will be.

4 Evaluation and Validation

The adaptation strategy has been validated/ "instantiated" by applying it to the use case which is a real-world conversational content-based e-commerce recommender system based on PDO that semantically describe the products offered in e-commerce applications according to GoodRelations[6]. Two feedback channels deliver implicit and explicit feedback as RDF data via separate SPARQL endpoints programmatically accessible. In a conversational approach the actions and modifications done in the adaptation layer mainly lead to a changed user dialogue in the application layer. Four types of PDO changes are defined with the following impact on the user dialogue:

- Switching individuals (i.e. properties are related to other individuals within the same class): This leads to a different clustering of the questions
- Switching datatype property ranges (i.e. properties get Boolean ranges instead of string ranges and vice versa (where applicable)): This leads to textual modifications of the questions
- Switching annotation properties label and comment (i.e. properties get different labels and comments extracted from another information source): This leads to textual modifications of the questions (and maybe a need-based sales approach instead of a technology-prone one)
- Changing annotation property priority (i.e. different priority values): This leads to a different ranking of the questions and skips the ones with low priorities

Applying the adaptation strategy could be done quite smoothly. Only minor aspects of the strategy were clarified, restructured, and reformulated. After having applied the strategy, the use case was concisely described and conceived by the ontology engineer. Moreover, the result formed the basis of the technical specification and thus the development of the adaptation layer.

Due to space limitations the "instantiation" of the adaptation strategy is not completely elaborated in this paper. In the following the evolution heuristic based on tabu search is introduced in extracts (excluding its ramp-up, for instance). The "taboos" are defined as follows:

- General tabu criterion gt: It is calculated by multiplying the two specific tabu criteria defined below; result is the number of allowed PDO changes gt; the PDO changes (e.g. switching the property weight from the individual WeightAndDimension to the individual GeneralCharacteristics) are sequentially executed and added to the tabu list
- Specific tabu criteria (specifically calculated for each type of PDO change):
  - "Allowed number of horizontal switches" sw: One (set of) ontological entity of a PDO within the same type of PDO change is switched, e.g. a PDO change of one (set of) property or (set of) individual – often there is only one switch possible like changing the individual, the property range, or the annotation properties label and comment, and the next change would revert that change. This tabu is defined as follows:

[6] www.purl.org/goodrelations

$$\text{sw} = \begin{cases} 0, \text{ case: } p = 1 \wedge c_{fix} = 0 \\ 2 + c_{fix}^2 / 2 - c_{fix}, \text{ case: } p = 1 \wedge c_{fix} = 2*k,\ c_{fix},\ k \in \mathbb{N} \setminus \{0\} \\ 1 + c_{fix} * (c_{fix} - 1) / 2, \text{ case: } p = 1 \wedge c_{fix} = 2*k - 1,\ k \in \mathbb{N} \setminus \{0\} \\ 1 + p^2 / 2 - p, \text{ case: } p > 1 \wedge p = 2*k,\ p \in \mathbb{N} \setminus \{0,1\},\ k \in \mathbb{N} \setminus \{0\} \\ p * (p - 1) / 2, \text{ case: } p > 1 \wedge p = 2*k - 1,\ p \in \mathbb{N} \setminus \{0,1\},\ k \in \mathbb{N} \setminus \{0\} \end{cases} \quad (\mathbf{1})$$

(c_{fix}: Number of fixed candidates to be switched to), p: Number of pools of sets of entities (e.g. each source for the properties is a pool like string ranges, Boolean ranges, DBpedia, or WordNet), k: Even ($c_{fix} = 2 * k$, $p = 2 * k$) or odd ($c_{fix} = 2 * k - 1$, $p = 2 * k - 1$) number of fixed candidates or pools (the case for the even c_{fix} or p equates to an Eulerian trail, the case for the odd c_{fix} or p to an Eulerian circuit)

Result is the number of allowed switches sw. In case an entity is already connected to c_{fix}, the second and third cases in (**1**) are lessen by this one "impossible" switch (i.e. $\text{sw}_{\text{fix}} = \text{sw} - 1$). In case sw is met, the PDO change with the second highest ST within the same type of PDO change is going to be executed.

- "Allowed number of vertical PDO change iterations" ch: Successive sw switches within the same type of PDO change. These formulae are omitted.

sw is calculated exemplarily for the PDO change "switching individuals" ($p = 1$): A digital camera has the sets of properties and individuals {faceDetection, Features}, {weight, WeightAndDimension}, {videofunction, GeneralCharacteristics}, {HDMI, Ports}, {opticalZoomFactor, LensFeatures}, and {touchscreen, Display}. E.g., weight could be switched to Features or GeneralCharacteristics. Hence, $c_{fix} = 2$, and sw = 2 (not connected to c_{fix} before switching) and $\text{sw}_{\text{fix}} = 1$ (connected to c_{fix} before switching). In case sw is met, the next set of the properties related to the same individual is switched.

The adaptation layer is going to be evaluated by conducting an experiment with approximately thirty ontology experts who decide the PDO changes to be executed. Eventually, the PDO resulted from this manual evolution is compared with the automatically evolved one regarding the evaluation criteria defined by [Gó01]. The adaptation layer is going to be validated by programming the layer and measuring the effects in the e-commerce recommender system. Its success is defined by the click-out rate (i.e. clicks-to-recommendations; the user follows the recommendation by clicking on the product recommended) that measures the impact of the PDO evolution.

5 Conclusion

The need for automatically updating and evolving ontologies is urging in today's usage scenarios. Having accomplished an automated ontology evolution based on user feedback can mainly have two impacts on the community. Firstly, it signals that this is feasible and thus can induce further research in this direction. Secondly, the adaptation strategy formulated can be seen as methodology and utilised in similar efforts, e.g. for developing automated feedback-driven systems. The application scenarios are semantic applications based on PDO like e-commerce recommender systems.

6 References

[GL97] Glover, F. and Laguna, M. 1997, Tabu Search, Kluwer Academic Publishers.

[Gó01] Evaluation of ontologies, International Journal of Intelligent Systems, Volume 16, pp. 391-409.

[Gr93] Gruber, T. R. 1993, Toward principles for the design of ontologies used for knowledge sharing, Formal ontology in conceptual analysis and knowledge representation, Kluwer Academic Publishers.

[Ha05] Haase, P. et al. 2005, A framework for handling inconsistency in changing ontologies, Proceedings of the 2005 International Semantic Web Conference (ISWC05), pp. 353-367.

[KN03] Klein, M.; Noy N. F. 2003, A component-based framework for ontology evolution, Proceedings of the IJCAI-03 Workshop on Ontologies and Distributed Systems.

[Ko07] Konstantinidis, G. et al. 2007, Ontology evolution: A framework and its application to RDF, Proceedings of the Joint ODBIS & SWDB Workshop on Semantic Web, Ontologies, Databases.

[No06] Noy, N. F. et al. 2006, A framework for ontology evolution in collaborative environments, Proceedings of the 2005 International Semantic Web Conference (ISWC05), pp. 544-558.

[St02] Stojanovic, L. et al. 2002, User-driven ontology evolution management, Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW '02), pp. 285-300.

[St03] Stojanovic, N. et al. 2003, The OntoManager – a system for the usage-based ontology management, LNCS 2888, pp. 858-875.

[Wa11] Automated ontology evolution for an e-commerce recommender, Proceedings of the 14th International Conference on Business Information Systems (BIS 2011), in press.

[Za09] Zablith, F. 2009, Evolva: A comprehensive approach to ontology evolution, Proceedings of the 6th European Semantic Web Conference (ESWC 2009).