Automated Ontology Evolution as a Basis for Adaptive Interactive Systems

Elmar P. Wach
STI Innsbruck, University of Innsbruck/
Elmar/P/Wach eCommerce Consulting
Technikerstraße 21a, 6020 Innsbruck,
Austria/ Hummelsbüttler Hauptstraße
43, 22339 Hamburg, Germany
+49 172 713 6928

elmar.wach@sti2.at/
wach@elmarpwach.com

ABSTRACT

The research presented in this paper aims at realising an automated ontology evolution process based on feedback without a human inspection. For that, a generic adaptation strategy consisting of a feedback transformation strategy and an ontology evolution strategy is formulated. It decides when and how to evolve by evaluating the impact of the evolution in the precedent feedback cycle. These strategies are implemented in a feedback transformer component and an adaptation manager component respectively, constituting a new adaptation layer. The adaptive ontology is evaluated with an experiment and validated with a real-world conversational content-based e-commerce recommender system as use case.

Categories and Subject Descriptors

G.2.2 [**Discrete Mathematics**]: Graph Theory – *graph algorithms, graph labeling*. H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *relevance feedback*. H.3.5 [**Information Storage and Retrieval**]: Online Information Services – *commercial services, web-based services*. I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods – *representations (procedural and rule-based), semantic networks*. I.2.6 [**Artificial Intelligence**]: Learning – *concept learning, knowledge acquisition*. K.4.3 [**Computers and Society**]: Organizational Impacts – *automation*

General Terms

Algorithms, Management, Measurement, Design, Experimentation, Standardization, Languages.

Keywords

Ontology Evolution, Ontology Versioning, Recommender Systems, Self-Adapting Information Systems, Algorithms.

SEMAIS'11, February 13, 2011, Palo Alto, CA, USA

1. INTRODUCTION

Today, the user in the Internet gets overflowed with information and products that she should purchase. Not only becomes it difficult for her to take the right buying decision, but also don't match many search results her needs. Hence, recommender systems in e-commerce applications have become business relevant in filtering the vast information available in the Internet (and e-shops) to present useful search results and product recommendations to the user.

As the range of products and customer needs and preferences change – and they will change even more frequently – it is necessary to adapt the recommendation process. Doing that manually is inefficient and usually very expensive.

Therefore, this research proposes an automated adaptation of the recommendation process by utilising semantic technology and processing user feedback.

The shortcomings of a manual adaptation of the recommendation process based on user feedback are aimed to be solved with a system based on product domain ontologies (PDO) modelling the products offered in the e-commerce application and automatically evolving with processing user feedback. As the PDO describes the products formally, it offers a higher computability than conventional product descriptions and, hence, facilitates automated processing of information.

In order to get the system user-driven, user feedback is gathered by unobtrusively monitoring user needs. The more information is available from a user, the better the adaptation to her needs can be. Hence, implicit and explicit feedbacks provided via feedback channels are evaluated. Implicit feedback is given by the user as a side-effect of her usage behaviour, e.g. by clicking on the product recommended. Explicit feedback could be provided by answering questions about her satisfaction with the application. As this effort cannot be expected from a user, an alternative is to extract feedback from the Web that could also deliver new information and aspects about the products offered. In order to focus this research on developing an automated ontology evolution, the feedback is assumed to be given.

On a more abstract level, this research aims at realising an automated ontology evolution process based on feedback without a human inspection.

Topics of the SEMAIS 2011 workshop related to this research:

- What are the major technical challenges for developing or generating user interfaces based on semantic models? This paper aims to answer the above question with a generic approach.
- For which kind of systems or applications are semantic models particularly useful?
  The use case in this paper is a recommender; for which other systems or applications can it be useful?
- Additional question: Which ontological information and its changes (properties, etc.) are requested by adaptive interactive systems?

2. RELATED WORK

Previous approaches to the topic of this research can be found in concepts for ontology evolution like formulated frameworks for ontology evolution, e.g. [6], [7], [8], [14], [16], [18]. Due to the specific challenges of the present research like the automated ontology evolution process, none of the identified frameworks can be completely used as basis, e.g. all of the frameworks include a step for the human inspection of the ontology changes before they are executed. The closest work to the research in this paper is [16] – 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. The research in this paper proposes an extension of [16] towards an automated ontology evolution by developing a generic adaptation strategy and further introducing a complete feedback cycle based on the ontology usage that eliminates the implementation and validation steps of above – an ontology change needs those manual steps no longer, as an insufficient change would be alerted by a negative feedback and get corrected automatically.

The approaches to the identified recommender systems [1], [2], [4], [11], [12], [13] research the impact on the recommendation result by using the different recommender types (i.e. content-based filtering, collaborative filtering, hybrid approaches) and mostly utilising domain and user ontologies, whereas the feedback gets processed in the latter one. None of them combines an e-commerce domain ontology with the processing of implicit and explicit user feedbacks.

3. ADAPTATION STRATEGY

For realising an automated ontology evolution, a generic adaptation strategy consisting of a feedback transformation strategy and an ontology evolution strategy is formulated. It decides when and how to evolve by evaluating the impact of the evolution in the precedent feedback cycle. The first question defines the (temporal and causal) trigger initiating the ontology change. Basically, this is receiving and transforming the feedback into ontology input and will be addressed with a feedback transformation strategy (confer chapter 3.1).

The second question defines the changing of the ontology including instance data. This is denoted by ontology evolution referring to the activity of facilitating the modification of an ontology by preserving its consistency [19]. This will be addressed with an ontology evolution strategy (confer chapter 3.2) considering also how identified conflicts can be solved, e.g. when moving a sub-concept.

By following the principles of adaptive systems [3], the adaptation strategy is implemented in a new adaptation layer consisting of components in which the user feedback gets transformed (i.e. Feedback Transformer) and the respective actions are decided and initiated (i.e. Adaptation Manager).

3.1 Feedback Transformation Strategy

In order to automatically process feedback, i.e. transforming it into ontology input, an adequate feedback transformation strategy has to be formulated and implemented. It has to allow for different feedback channels as well as different kinds of feedback. This strategy is implemented in the feedback transformer component depicted in figure 1. In the Feedback Transformer the ontology affected by the feedback reported is identified, the feedback is analysed and transformed, and eventually get related to the precedent feedback.

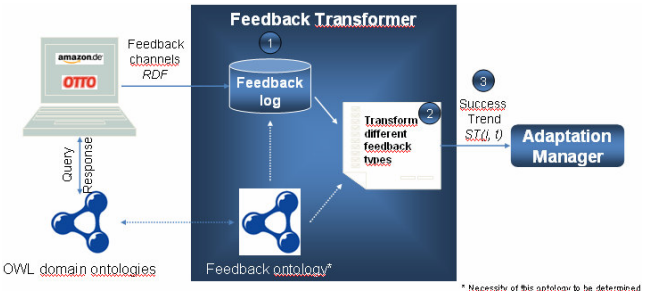

Figure 1. Conceptual architecture of the feedback transformer component

Basically, the strategy comprises the following steps:

1. Gather feedback from the different channels
2. Transform different feedback types
1. Report transformed feedback to the next component

Ad 1. Each feedback channel provides user feedback as RDF triples at separate SPARQL endpoints. The RDF triples are retrieved by the Feedback Transformer and captured in a semantic feedback log as instances of the feedback ontology (confer next paragraph).

Ad 2. The feedback ontology is a prerequisite for the meaningful analysis of the feedback [17]. In the present research, it models the feedback at the product level and additionally contains all product names of the product ontologies. The structure of the feedback ontology enables reasoning about a product and its ratings including the historical development as well as identifying properties and relations to be newly added to the product ontology. Accordingly, we distinguish between the three feedback types "KPI[1] trend", "product rating", and "new property". The root concept is "Feedback". Its hierarchy consists of the sub-concepts "KPI trend", "product rating", and "new property". Appropriate relations like "previousRating" model the history of the ratings.

[1] Key Performance Indicator, measured in the application layer

The first two feedback types are converted by either a simple transformation or a feedback evaluation algorithm to values in the range [+1…-1] relating the current transformed feedback to the one in the precedent cycle.

For the feedback type "product rating" the RDF feedback includes the product name and rating but no new potential property. The feedback is transformed with a feedback evaluation algorithm. In the first step, the impact of the ontology evolution on the KPI (e.g. conversion rate and click-out rate) is calculated for each product and feedback channel. In the next step, all feedback channels are aggregated at the product level. Finally, a trend metric is calculated relating the current transformed feedback to the one in the precedent cycle.

For the feedback type "new property" the RDF feedback includes the product name and a new potential property to be eventually added to the product ontology, e.g. information like aspects or relevant features of a product. This feedback type is not covered by the feedback evaluation algorithm. A new sub-property for the aspect/ feature is created in the feedback ontology and its count gets related to the count of all properties in the respective PDO. When reaching a defined threshold, the new property is added to the respective PDO.

The semantic feedback log captures the exact sequence of the reported feedbacks. Each feedback is associated with the respective product (i.e. the RDF feedback contains the corresponding product name) and represented as instances of the sub-concepts of "Feedback". These instances contain the product name, feedback channel, date and time of the feedback, rating, and the certainty of the rating as well as the number of properties contained in the product ontology. The log allows the analysis of the feedback development.

Ad 3. After having transformed the different feedback types, the calculated metrics relating the current feedback to the feedback in the precedent cycle are reported to the next component, i.e. the Adaptation Manager.

3.2 Ontology Evolution Strategy

The ontology evolution strategy defines how the PDO change. It associates the transformed feedback values to evolution actions and ensures a consistent new version of a PDO. This strategy is implemented in the adaptation manager component depicted in figure 2. In the Adaptation Manager the structure of the respective ontology get dynamically analysed with SPARQL SELECT statements and the ontology changes (e.g. switching individuals, switching annotation property labels and comments, changing annotation property priorities, adding new properties) are executed with SPARQL CONSTRUCT rules according to predefined evolution strategies.

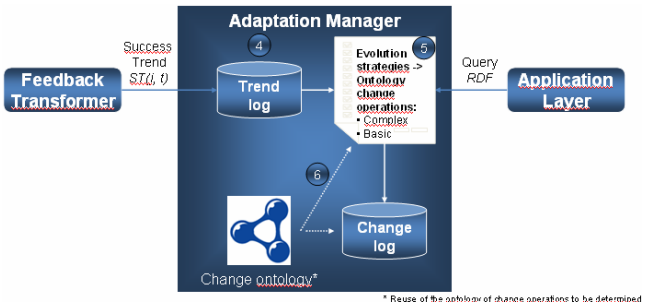

Figure 2. Conceptual architecture of the adaptation manager component

Basically, the strategy comprises the following steps:

4. Gather feedback trends
5. Associate ontology changes with evolution strategies
6. Ensure a consistent ontology evolution

Ad 4. In each feedback cycle the transformed feedback gets reported to the Adaptation Manager. The feedback is based on the product level. Each reported feedback is captured in a trend log at the product level.

Ad 5. The central task of the ontology evolution strategy and the Adaptation Manager is to choose the right evolution, i.e. ontology changes, for the transformed feedback.

[9] introduced a meta-ontology for the ontology evolution enabling representation, analysis, realisation, and sharing of ontological changes. Each possible change is represented as a concept in that evolution ontology having an evolution log as instance capturing the changes. A central element in the framework of [7] are a change log and an ontology of change operations for OWL describing basic ontology change operations[2] and complex change operations composed of multiple basic operations. This research aims at utilising the ontology of change operations sketched above.

Derived from user scenarios, evolution strategies are defined reflecting different behaviours and associating ontology changes, namely:

- Risky Evolution ("always evolve differently"): Regardless of the feedback trend between two consecutive feedback cycles, other complex ontology change operations are executed
- Progressive Evolution ("learn from the past"): Depending on the leap of the trend, same or different complex ontology change operations are executed; in case of a negative trend, it is optional to either do a different complex ontology change operation or a rollback; additionally, with a threshold indicating the increase of the trend between the current and the precedent cycle the "risk" of the evolution can be adjusted and the strategy tuned towards the Risky Evolution (with a higher threshold)
- Safe Evolution ("only revert negative trends"): In case of a negative trend, a rollback is executed

[2] Basic ontology change operations modify only one specific feature of an OWL ontology

- Rollback ("undo the ontology changes"): Reverts the ontology changes from the precedent feedback cycle and is based on any reason or decision of the manager; it is executed only once but can be manually chosen multiple times

Ad 6. After having chosen the ontology change operations to be executed, the ontology has to evolve depending on rules and by retaining its consistency to finally provide its knowledge to the application layer.

The existing research about ontology evolution is based on the work about data schema evolution but focuses on the specific needs of ontologies, e.g. [10], [15], [16].

To execute ontology changes, an ontology evolution algorithm has to be formulated. The following prerequisites have to be respected:

- The basic and complex ontology change operations have to be defined formally
- It has to be defined when an ontology is inconsistent, i.e. an ontology consistency model has to be formulated; the preconditions and postconditions of the change operations have to be checked before execution
- The options for a consistent ontology evolution have to be identified and the "best" evolution path chosen; in the present research the belief revision principle of minimal change will be followed [8]; eventually, the ontology evolution algorithm can be formulated

When evolving the ontology, it has to be clear how the ontology has been evolved over time, i.e. the different ontology evolutions have to be versioned. In the context of this research this is of paramount importance, for (i) the ontology changes in the current feedback cycle are derived from the changes in the precedent cycle and (ii) an undoing of the changes in the precedent feedback cycle, i.e. a rollback, has to be realisable.

The preferred concept of ontology versioning is change-based versioning (i.e. each state gets its own version number and additionally stores information about the changes made), because it facilitates change detection, integration, conflict management [9], and it allows the interpretation how ontology changes influence the KPI. A change-based versioning can be best realised by tracking the ontology changes in a semantic log [9].

The change ontology models the applicable changes and meta-information and provides the semantics of all possible ontology changes. The root concept is "Change". Its hierarchy consists of the sub-concepts "complex ontology change operations" and "basic ontology change operations". Appropriate relations like "previousChange" model the history of the ontology changes and construct the sequence of the required changes. The structure of the change ontology enables reasoning about changes including their historical development.

The semantic change log captures the exact sequence of the ontology changes executed. Each change is represented as instances of the sub-concepts of "Change". The log allows the analysis of the change development including realising a rollback.

The whole adaptation strategy and its implementation via the components Feedback Transformer and Adaptation Manager allow eliminating both manual steps in the six phase evolution process of [16]:

- Phase "Implementation" (ontology changes are manually approved before execution): Nobody has to do that, as the ontology evolution is seen as a complete feedback cycle – an insufficient ontology change is indicated by decreased KPI and gets revised according to the evolution strategy chosen
- Phase "Validation" (performed changes can get manually validated): As the ontology changes are predefined, only valid changes are executed, and nobody has to validate them

4. EVALUATION AND VALIDATION

The automatically evolved ontology is going to be compared with a manually evolved one by setting up and evaluating an experiment with ontology experts. Those analyse the feedbacks delivered and decide the ontology changes to be executed. Eventually, the ontology resulted from this manual evolution is compared with the automatically evolved one regarding the evaluation criteria consistency, completeness, conciseness, expandability, and sensitiveness [5].

The validation of this research is done with a use case by utilising a real-world conversational content-based e-commerce recommender system and two feedback channels – the Web application and information extracted from Linked Open Data. As the recommender is already used in live e-commerce applications, the evaluation of the system adaptations is a real-world scenario.

The recommender is based on PDO that semantically describe the products offered in e-commerce applications according to the GoodRelations ontology.[3]

The success of such a system is usually defined by analysing KPI like the achieved conversion rate (i.e. customers-to-recommender users ratio) or click-out rate (i.e. clicks-to-recommendations ratio).

The evaluation scenario is to test and evaluate the impact of the ontology evolution by utilising the formulated evolution strategies, i.e. Risky Evolution, Progressive Evolution, and Safe Evolution.

The impact of the ontology evolution will be analysed and evaluated with regard to the respective KPI at the application level after each to be defined number of accomplished recommendation processes and reported to the ontology.

According to the respective results and feedbacks reported, the ontology evolves. The ontological knowledge is provided to the application layer, and eventually adapted recommendations are presented to the customer. The feedback circle of the automated system concludes with re-evaluating the KPI after having again reached the defined number of recommendation processes.

The intended results are a highly adaptive system and eventually better recommendations given to the user leading to an increase of the defined KPI. The expected business impacts are a higher

[3] www.purl.org/goodrelations

customer satisfaction and loyalty and eventually increased revenue for the provider of the application.

This evaluation procedure will be executed for all three evolution strategies and evaluated analogously.

An interesting result of the evaluation scenario would be that one of the three evolution strategies leads to a higher increase of the KPI.

In case a predominant evolution strategy is identified, it can be interpreted that the historic development of changing the ontology (i.e. doing the same change again versus doing a different change) has a significant influence on the customer satisfaction. Though, this can in the case of same changes only be valid within a realisable frame, e.g. it is not possible to move up a sub-concept in the concept hierarchy infinitely times.

5. CONCLUSION

The need for automatically updating and evolving ontologies is urging in today's usage scenarios. The present research tackles an automated process for the first time (to the best knowledge of the author). The reason for that can be found in the ontology definition "formal, explicit specification of a shared conceptualisation". "Shared" means the knowledge contained in an ontology is consensual, i.e. it has been accepted by a group of people. Entailed from that, one can argue that by processing feedback in an ontology and evolving it, it is no longer a shared conceptualisation but an application-specific data model. On the other hand, it is still shared by the group of people who are using the application. It may even be argued that the ontology has been optimised for the usage of that group (in a specific context or application) and, hence, is a new way of interpreting ontologies: They can also be a specifically tailored and usage-based knowledge representation derived from an initial ontology – an ontology view, preserving most of the advantages like the support of automatically processing information. Thus, this changed way of conceiving ontologies could facilitate the adoption and spread of using this powerful representation mechanism in the real world, as it is easier to accomplish consensus within a smaller group of people than a larger one.

6. ACKNOWLEDGMENTS

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*).

7. REFERENCES

[1] Aktas, M. S., Pierce M., Fox, G. C., Leake D. 2004. A Web based conversational case-based recommender system for ontology aided metadata discovery, *Proceedings 5th IEEE/ ACM International Workshop on Grid Computing*, pp. 69-75.

[2] Blanco, Y. et al. 2005. AVATAR: An approach based on semantic reasoning to recommend personalized TV programs, *Proceedings 14th International conference on World Wide Web*, pp. 1078-1079.

[3] Broy, M. et al. 2009. Formalizing the notion of adaptive system behavior, *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC '09)*, pp. 1029-1033.

[4] Drachsler, H. et al. 2008. Effects of the ISIS recommender system for navigation support in self-organised learning networks, *Proceedings of Special Track on Technology Support for Self-Organised Learners*, pp. 106-124.

[5] Gómez-Pérez, A. 2001. Evaluation of ontologies, International Journal of Intelligent Systems, Volume 16, pp. 391-409.

[6] 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.

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

[8] 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*.

[9] Mädche, A. et al. 2002. Managing multiple ontologies and ontology evolution in Ontologging, *Proceedings of the IFIP 17th World Computer Congress – TC12 Stream on Intelligent Information Processing*, pp. 51-63.

[10] Mädche, A. et al. 2003. Managing multiple and distributed Ontologies on the Semantic Web, *The VLDB Journal – The International Journal on Very Large Data Bases*, Volume 12, Issue 4, pp. 286-302.

[11] Maidel, V., Shoval, P., Shapira, B., Taieb-Maimon, M. 2008. Evaluation of an ontology-content based filtering method for a personalized newspaper, *Proceedings of the 2008 ACM conference on Recommender systems*, pp. 91-98.

[12] Middleton, S. E., De Roure, D. C., Shadbolt, N. R. 2001. Capturing knowledge of user preferences: Ontologies in recommender systems, *Proceedings 1st international conference on Knowledge capture*, pp. 100-107.

[13] Middleton, S. E., Shadbolt, N. R., De Roure D. C. 2003. Capturing interest through inference and visualization: Ontological user profiling in recommender systems, *Proceedings 2nd international conference on Knowledge capture*, pp. 62-69.

[14] 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.

[15] Plessers, P. 2006. *An approach to Web-based ontology evolution*, Ph.D. Thesis, Department of Computer Science, Vrije Universiteit Brussel.

[16] 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.

[17] Stojanovic, N. and Stojanovic, L. 2002. *Usage-oriented evolution of ontology-based knowledge management systems*, LNCS 2519, pp. 1186-1204.

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

[19] Suárez-Figueroa, M. C. and Gómez-Pérez, A. 2008. Towards a glossary of activities in the ontology engineering field, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC '08)*.