# The Snoopy Concept: Fighting Heterogeneity in Semistructured and Collaborative Information Systems by using Recommendations

Wolfgang Gassler, Eva Zangerle, Günther Specht
*Databases and Information Systems*
*Institute of Computer Science*
*University of Innsbruck, Austria*
*Email: firstname.lastname@uibk.ac.at*

## ABSTRACT

*The collaborative creation and manipulation of semistructured data imposes the major problem of structure heterogeneity. The more users enter information, the more heterogeneous the structure of information becomes. This proliferation of the schema has a significantly negative impact on the performance of querying facilities as structured, unified access of data is no longer possible. In this paper we present the Snoopy Concept, a novel approach for collaborative, semistructured information systems within an online environment. It deals with structure heterogeneity by incorporating the user in the alignment process of data already during the insertion. This is accomplished by providing the users with useful recommendations how to structure information. Furthermore, the system encourages users to enter more information as it points users to missing bits of information.*

**KEYWORDS:** semistructured data, heterogeneity, RDF, recommendations, schema proliferation, collaboration

## 1. INTRODUCTION

Collaboration has been lifted to a new level throughout the last years as online collaboration, social networks and mass intelligence have become immensely popular and important. Wiki systems are the most important type of collaborative, online information systems. Their flexibility and simple usage paved the way for their current popularity [12]. The big disadvantage of common wiki systems is the lack of machine-readable access to information which results in the lack of complex search capabilities. Traditional wiki systems only support full-text search which is not feasible for complex queries such as "Which Universities have more than 10.000 students and have a female rector?" Weikum *et al.* [20] observed that modern information systems have to be able to support both structured and unstructured data to combine the advantages and be able to answer such complex structured questions.

Semistructured information systems try to combine these two models while retaining the advantages by supporting structured information without having to specify a fixed schema. Most of such systems are based on RDF, which consists of triples containing a *subject*, a *property* and a *value*. Infoboxes within Wikipedia articles are perfect examples of such semistructured data, which can also be extracted as RDF triples [3]. The schemata of infoboxes in Wikipedia are predefined and maintained by the committed community. However Wu and Weld [22] showed that Wikipedia's template-based infoboxes are divergent and noisy. They detected a huge amount of semantically duplicated templates, synonymous attributes in the schemata and a long-tail distribution of template usage. These facts imply that even with the support of a huge committed community, the proliferation of schemata within a mass collaboration system cannot be prevented. Hence, collaborative information systems which do not restrict the schema of information at all would result in a rapid growth of proliferation of substructures and schemata. When storing semantically similar information using very heterogeneous schemata, the knowledge is no longer searchable by using structured access and structured queries. For example a user who searches for the value of *numberOfStudents* cannot find information

which was stored using the properties *students*, *numberStudents* or *num_students*. Therefore, especially in collaborative information systems, the creation of a common schema without restricting the domain, type or amount of information is needed.

In this paper we introduce the Snoopy Concept which enables collaborative, semistructured, online information systems to exploit the extensive knowledge of the collaborating users. The concept proposes to support the user during the insertion process to align information to a homogeneous structure by a self-adapting and self-learning recommendation engine. The recommendations contribute to a very homogeneous schema and simultaneously increase the quality and quantity of stored knowledge.

The rest of the paper is organized as follows. Section 2 outlines the main idea behind the Snoopy concept. Subsequently, Section 3 explains how recommendations can contribute to the creation of a common schema and which benefits arise from this fact. Section 4 provides detailed information about the implemented prototype of the Snoopy Concept. Section 5 features the evaluation and the respective results of our approach. Section 6 covers the related work and Section 7 concludes the paper.

## 2. BASIC SNOOPY CONCEPT

The Snoopy Concept is based on the semistructured data model. Semistructured data features a structure but its schema is not known in advance. Essentially, the Snoopy Concept proposes to model information and knowledge as property-value pairs. Therefore, users are able to store information about a certain subject as property-value pairs, e.g. the number of students at a certain university could be stored as *numberOfStudents: 20.000*. Storing information about a certain subject, e.g. the University of Innsbruck, forms a so-called collection and could be structured as follows:

*Collection: University of Innsbruck*
*country: Austria*
*numberOfStudents: 21,001*
*numberOfFaculties: 15*
*established: 1669*

By using collections, all information is stored as triples similar to RDF triples [13]. These triples consist of the subject name, the property and the respective value as e.g. *<University of Innsbruck> <numberOfStudents> <20.000>*. By using such triples to represent information, all information stored is machine-readable and therefore can e.g. further be used for automatic reasoning tasks and complex structured search facilities.

It is important to note that within semistructured systems, users can arbitrarily choose the properties used for storing information. This fact is very beneficial as it provides a huge amount of flexibility to the users of the system while at the same time - due to the property-value format - still features a certain amount of structure. However, the schema within such information systems tends to become more and more heterogeneous as more and more information is entered by multiple users. The reason for such a *proliferation* of the schema is the fact that users choose different property names for semantically equivalent properties, e.g. one user uses *numberOfStudents* and another user may choose *noStudents*. Furnas *et al.* [9] already found in the 1980s that the chance of two humans choosing the same term for the same object is only 20%. This fact is crucial in online, mass-collaboration information systems, as there are thousands of different users who come from different social levels, backgrounds and edit information in different domains and contexts. The resulting heterogeneous schema has a serious negative impact on the performance of the information system as search capabilities are very limited if no common structure is incorporated within the stored information. This means that users who are searching for the property *noStudents* are not able to find all semantically equivalent entries, as many other users used the property *numberOfStudents*. Thus, valuable information is lost for the user as it is not accessible by common search facilities.

### 2.2. Main Benefits of the Snoopy Concept

In order to avoid a proliferation of schemata, the Snoopy Concept is focused on supporting the users in the creation of a common, homogeneous structure (properties). This is realized by incorporating the user into the alignment process. The user has very valuable knowledge about the information he is about to insert. Therefore, the Snoopy concept aims at exploiting this knowledge for the alignment of information by suggesting highly suitable structures to the user during the insertion process. Additional recommendations enable the user to insert information as simply and efficient as possible. By using the user-centric approach the following benefits can be achieved:

- Avoid proliferation of structures
- Avoid synonyms in the system
- Semantic refinements by resolving homonyms
- Exploit user's extensive and valuable knowledge

- Increase the quantity of information contained in the system
- Increase the quality of information in the system

All recommendations aim at supporting the user, exploiting the knowledge of the user and therefore "snooping" as much information as possible. The underlying measures and approaches of the Snoopy Concept to be able to provide these benefits are discussed in the following section.

## 3. RECOMMENDATIONS

The key enabler for a common schema within an information system based on the Snoopy concept is a recommender system [15]. Essentially, a recommender system is a system which analyzes all information stored within the system to subsequently provide its users with useful recommendations. Traditionally, recommender systems are used in online shops where clients are pointed to further products. Another scenario is the recommendation of movies for the users of a movie database. Such recommendations are usually computed by applying similarity measures to the stored data in order to either find items similar to those the user bought before or to find users with similar preferences to further deduce item recommendations.

In the context of the Snoopy Concept, the recommender system suggests suitable structures the user might want to use. Also, not only properties (structures) are recommended to the user. The Snoopy Concept also proposes to recommend values, links, types and input formats to the user. These recommendations and their benefits are described in detail in the next sections.

### 3.1. Structure Recommendations

The term of structure recommendations refers to the recommendation of additional properties during the insertion process and is the most important feature of the Snoopy Concept. This feature significantly contributes to a common and homogeneous schema within the system.

Consider the example of a user who e.g. specifies a subject consisting of the properties *numberOfFaculties* and *numberOfStudents*. The system computes a set of appropriate properties which occur on collections with a similar structure and recommends this set of properties to the user. To contribute to a common structure, an additional ranking mechanism is proposed. This ranking assures that more popular properties are preferred in the list of recommendations in order to efficiently use the limited visual space in a user interface as well as the limited cognition of the user.

In the mentioned example within the domain of universities e.g. the properties *rector*, *established*, *location*, *website*, etc. are recommended to the user. These recommendations are computed on the fly and are based on the just entered properties by the user. Any additional specified property or accepted recommended property during the insertion process results in the recomputation and refinement of all structure recommendations for the current collection.

The common schemata in the system are very dynamic, as they are based on all stored collections and therefore are influenced by every newly stored collection and their properties. Every newly stored property is automatically taken into the set of possible property recommendations and can influence further recommendations to other users who want to enter information about a similar collection. This flexibility and the fact that users are free to modify the recommended structure prevent the system from creating a completely unified and aligned schema. Thus, the user is guided to a common schema without restricting him in his way of structuring information or extending existing schemata. Therefore, the Snoopy Concept does not require any schema matching [16] after the insertion of data. The alignment is done by the user with his extensive knowledge and is therefore always more powerful than any automated alignment algorithm.

Furthermore, the recommendations of structures increase the quantity of information as the recommendations indicate "missing" bits of information. In the mentioned example of the domain of universities, the system could recommend the property rector. By providing such additional property recommendations, the user is encouraged to enter more information than he originally intended to insert and the valuable knowledge of the user is once more exploited. Such information gathered by the "snooping" process would be lost without recommendations and cannot be completed by any machine afterwards and therefore enhance the information system dramatically.

### 3.2. Avoiding Synonyms by Recommendations

During the specification of further content, the user is supported by kind of an intelligent auto-completion feature. The system suggests suitable properties to the user which have already been used within the information system. Consider e.g. a user who started entering the property *number*. The system subsequently suggests all previously used properties in the system which are related to the term *number*. In this case, the system would suggest

*numberOfStudents* and *numberOfFaculties*. In most cases the user accepts such a recommended property if it is suitable in the respective context. In this example the user would be prevented to insert a new property, such as *numberFaculties*. A more severe challenge in information systems is the coping with syntactically different synonyms as it cannot be solved by string-based matching approaches. Consider the example of a user entering *Faculty*, a string based matching approach would not recommend the already entered property *Academic Staff*. By using a thesaurus, *Faculty* can be matched with the semantically equivalent, already existent property *Academic Staff* which can then be suggested to the user.

### 3.3. Semantic Refinement & Value Recommendations

The guidance of the user is not limited to properties, also possible values can be suggested. This mechanism has the advantage of providing the user with values in an already aligned form, which prevents the user from entering synonyms of already existing values. This can be achieved by using the same mechanisms used for the recommendation of properties previously described.

Furthermore, not only the value itself can be recommended but the system can additionally suggest semantic links between values and other subjects. This measure copes with the common challenge in preserving semantic "correctness" of homonyms. If the user e.g. specifies the city *Freiburg* as a twin city of *Innsbruck*, it is not clear whether the user refers to *Freiburg in Germany* or *Freiburg in Switzerland*. Within the Snoopy concept, this problem is resolved by recommending possible semantic links from the entered value *Freiburg* to already existent, semantically equivalent collections (e.g. *Freiburg, Germany* and *Freiburg, Switzerland*). The user is then able to specify the meaning just by accepting the appropriate link-recommendation. As the user has extensive knowledge about the content to be inserted, he can provide more semantic information than any automated extraction process can do afterwards. Using these measures, homonyms are further semantically enhanced by humans, which leads to a high confidence of semantic data in the system.

### 3.4. Validation & Recommendations of Types

Furthermore, the concept proposes a validation process, which includes determining a data type for each newly entered value, e.g. the value for the property *numberOfStudents* is asserted to be an integer value. Vice versa, if a property is added that already exists, has a data type assigned and is used by the majority of property instances, the user is prompted to enter values according to this data type. The detection of data types, especially in the case of numeric types, is very important as it is crucial for queries based on numeric evaluation. E.g. the query "List all universities having more than 10,000 students" is only possible if the value of the property *numberOfStudents* is stored as a numeric value. Furthermore, also other data types like e.g. date, time, HTML, file, image, audio or video are possible and lead to special behavior according to the data type (e.g. date picker, calendar views, content-based image search, mp3 metadata search, etc.). Additionally, the syntactic correctness is validated according to the respective data type (e.g. correct date format).

All these measures enable the user to enter information fast and efficiently by just accepting recommendations while at the same time additional, semantically equivalent properties and values are avoided and the amount of unified information and the confidence of semantic data in the system are increased.

## 4. PROTOTYPE

The Snoopy Concept was implemented in a first prototype called "SnoopyDB". A screenshot of the SnoopyDB prototype can be seen in Figure 1. This figure shows a user entering information about the University of Innsbruck. The user already entered four property-value pairs about the foundation year, the founder, the number of professors and the official website of the University of Innsbruck. The three additional rows displayed in grey font mark the properties which were recommended by the system. The value fields corresponding to these properties already contain exemplary values. This way the user can immediately recognize that for example the value of the property *employees* is normally entered as a numeric value. The box on the right side of the screenshot contains further suitable properties for the current subject. These properties can easily be added to the input form by clicking on the arrow icon. The screenshot also shows how the system automatically detects the data type of the entered information. In line 4, a link for the website of the university is created. The system automatically detected that a big percentage of the values belonging to the property *website* were stored as links and therefore, the system suggests the insertion of a link to the user. In this case, the user accepted this suggestion and entered the URL of the official website of the University of Innsbruck. The underlying algorithms and implementation of SnoopyDB are described in the following section.

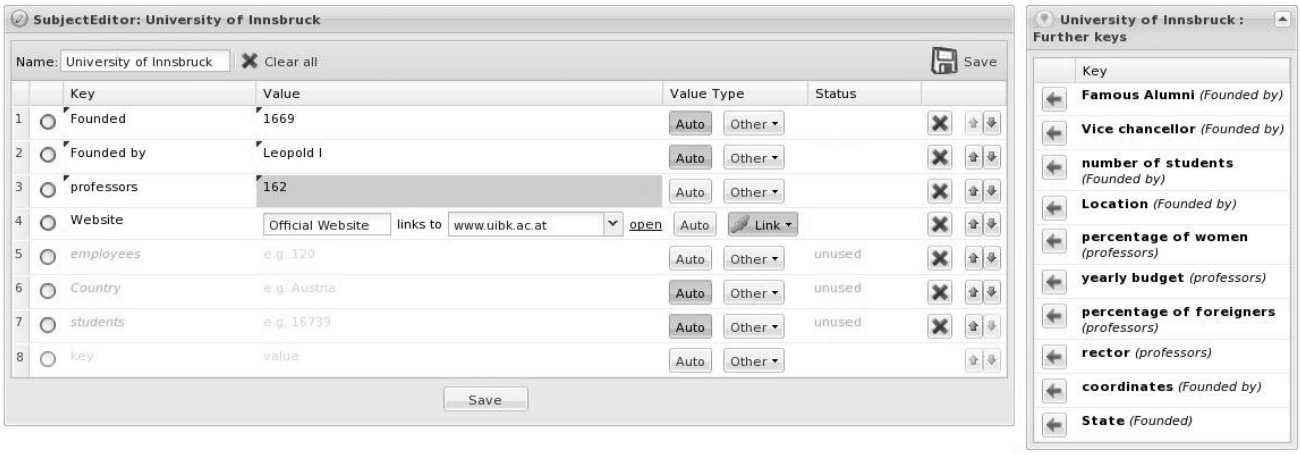



**Figure 1. Screenshot Of The SnoopyDB Prototype**

### 4.1. Recommendation Algorithm

The SnoopyDB approach is based on a recommendation algorithm (see Algorithm 1), which takes all collections (subjects) and properties occurring on these subjects into account. Formally, the set of properties occurring within the whole system can be denoted as $P = \{p_1, p_2, p_3, ..., p_n\}$ and the set of all subjects occurring can respectively be denoted as $S = \{S_1, S_2, S_3, ..., S_m\}$. The properties belonging to a certain Subject $S_i$ can be identified as $P_{Si}$.

**Algorithm 1:** Recommendation Candidate Computation

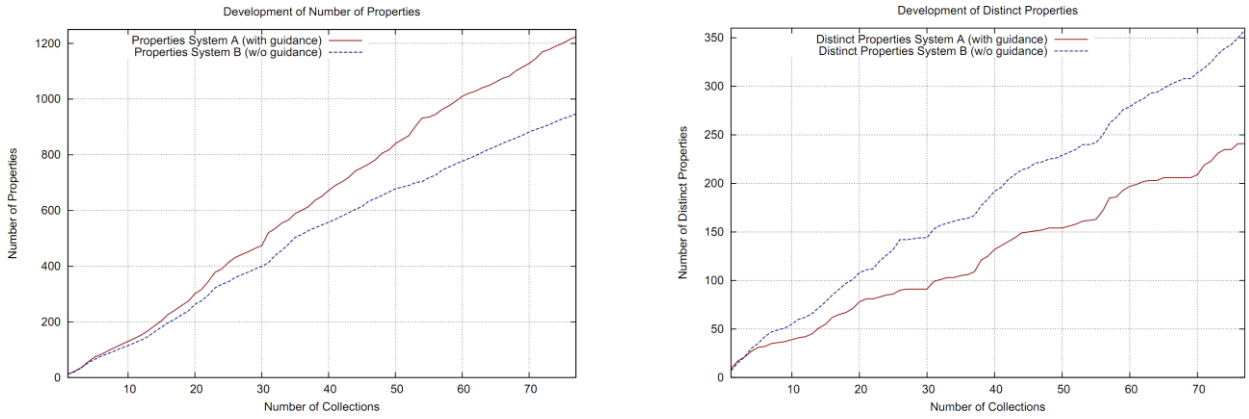
**Input**: $\mathcal{P}_{S_i}, \mathcal{R}$
**Output**: set $\mathcal{C}$ of all recommendation candidates for $S_i$
$\mathcal{C} \leftarrow \varnothing$
$\mathcal{T} \leftarrow \varnothing$
**foreach** $p_i \in \mathcal{P}_{S_i}$ **do**
  **foreach** $(p_a, p_b, c) \in \mathcal{R}$ **do**
    **if** $(p_a == p_i \wedge p_b \notin \mathcal{P}_{S_i})$ **then**
      $\mathcal{T} \leftarrow \mathcal{T} \cup \{(p_a, p_b, c)\}$
    **end**
  **end**
**end**
**foreach** $(p_a, p_b, c_i) \in \mathcal{T}$ **do**
  **if** $(\exists (p_x, c) \in \mathcal{C},\ where\ p_x == p_b)$ **then**
    $\mathcal{C} = \mathcal{C} \backslash \{(p_x, c)\}$
    $\mathcal{C} = \mathcal{C} \cup \{(p_x, c + c_i)\}$
  **else**
    $\mathcal{C} = \mathcal{C} \cup \{(p_b, c_i)\}$
  **end**
**end**
**return** $\mathcal{C}$

Based on these definitions, recommendations can be computed by firstly determining all pairs of properties ($p_a$, $p_b$) occurring on the same subject. If a user e.g. specified the properties *name*, *location* and *numberOfStudents* on the same collection, the pairs (*name, location*), (*name, numberOfStudents*) and (*location, numberOfStudents*) are formed. These pairs are computed for all subjects within the system and subsequently stored together with the total number of occurrences of the respective pair of properties within the whole system. This set of rules (pairs) is denoted by R and serves as input for the computation of recommendations during the insertion process. If a user enters new information about a certain subject, the properties already contained in this subject also serve as input for the computation of recommendations. For each property $p_i$ occurring on the input subject $S_i$, all triples where $p_i$ is contained within the property pair, are detected. These triples basically form the set of recommendation candidates. After having detected these recommendation candidates, the most important and therefore the most useful recommendations for the user have to be extracted. This is accomplished by determining the most popular properties within the recommendation candidates. Therefore, the number of occurrences of each recommendation candidate is summed up. These properties are subsequently ranked by their popularity and the top-*k* items are recommended to the user.

## 5. EVALUATION

The performance of the presented algorithm itself on very large semistructured datasets without user-interaction was shown in [23]. The following evaluation of the Snoopy Concept is based on the SnoopyDB prototype and was focused on the user-interaction regarding the acceptance of recommendations and the provided support in general. Furthermore, we investigated the implications of the provided mechanisms concerning the amount of entered information and the homogeneity of schemata within the system.

The evaluation of such an interactive information system cannot be conducted artificially because the interaction of users with the system cannot be simulated. This is due to




**(a)** **(b)**

**Figure 2. Evaluation Results: (a) Total Number Of Properties, (b) Total Number Of Distinct Properties**

the fact that the performance can only be determined if users accept or decline recommendations and use the autocompletion feature. Furthermore, the user has extensive domain knowledge about the data he is about to store. Therefore, we chose to conduct a user-centric evaluation. A crucial factor for the acceptance of recommendations is the fast computation of recommendations and the resulting response time of the system. Top-*k* recommendations can be computed on large datasets [23] with millions of rules within less than a fraction of a second on commodity hardware and therefore does not prevent the user from a fast creation of content. The experiments were conducted on two different prototypes:

- *System A*: a SnoopyDB prototype implementing all the recommendation and guidance features described in this paper.
- *System B*: a SnoopyDB prototype implementing no support and guidance at all. This system essentially enables its user to store collections based on property-value pairs.

The evaluation was based on a small basic dataset created in [10] consisting of entries originating from two domains (cities and musicians). In order to be able to compare the two systems and to practically show the benefits of the Snoopy Concept, we asked test users to enter information into both of the systems. The entered information was limited to two additional domains: universities and the motor vehicle industry (car models). In the first run, users had to enter information about an arbitrary university in System B and then in System A. In a second step, users were instructed to enter a second collection concerned with the motor vehicle industry. During the insertion process, we logged every single action the users took. Afterwards we analyzed the resulting logs and the data created within the two systems.

## 5.1. Results

In total, 24 test users (2/3 being computer scientists/students and 1/3 being standard computer users) took part in our experiments and 78 collections were created in both System A and System B. The analysis of the collections showed that within System A, a total amount of 1,247 property instances (property-value pairs) were entered whereas System B only contained 959 property instances which amounts to 31% more information within the system, which can be seen in Figure 2(a). This fact can be led back to the fact that by making structure recommendations, the users are encouraged to enter more information than they originally intended to. As for the number of distinct properties within the respective systems, System A contained 250 distinct properties whereas System B contained 368 distinct properties, shown in Figure 2(b). This implies that by facilitating the Snoopy Concept, a 33% less heterogeneous schema can be reached.

An important result of the evaluations was that the introduction of the new domains did not result in a dramatic increase of newly added properties. This fact implies that most of the properties were reused - even if the collections originated from different domains like e.g. universities and motor vehicle industry in our experiments.

The results of the analysis of the detailed log, which tracked all user actions on collections concerning the new domains are explained in the following paragraphs. The log consisted of 35 sessions, which are characterized by the respective user and framed by the “open collection” action and the “save collection” event. There are more events than newly added collections as some users preferred to build up collections in multiple steps which results in multiple sessions. We analyzed each session and the newly added property-value pairs in the respective

session. As a result, we encountered that 22% of all recommended properties (structure recommendations) were accepted by the user. This value of 22% is lower than the calculated precision value of 38% in [23]. However, the precision was calculated on a very large, template-based and therefore more homogeneous dataset and did not include any user interaction. In contrast to the precision value in [23], the acceptance rate cannot verify the correctness of the recommendations. Rather more, it describes the suitability assessed by the user in the according session resp. collection. The suitability is also emphasized by the fact that 49% of all newly added properties were inserted by accepting recommended properties. Furthermore, 23% of all newly added properties and 17% of all newly added values were inserted by accepting autocompletion recommendations regarding properties or values. Even the simple measure of suggesting the correct spelling of entered information (by using a dictionary service) was accepted in 37% and results in a further avoidance of synonyms resp. incorrect information. The evaluation of logs and the data contained in all collections shows that recommendations proposed by the Snoopy Concept contribute to a homogeneous structure within a collaborative, semistructured information system.

## 6. RELATED WORK

Many different research areas are closely related to the SnoopyDB approach. However, we identified the areas of collaborative online systems, recommender systems and semistructured information as the most important fields of research which are directly related to the Snoopy concept. The main research area which is covered by SnoopyDB are (mass) collaboration systems. Wikipedia is one of the most popular examples of collaborative information systems. A large part of Wikipedia pages feature so-called infoboxes which basically are tabular summaries of the page content. This tabular structure essentially consists of properties and according values. Therefore, Wikipedia also has to deal with the problem of schema heterogeneity. Thus, various research projects are concerned with the creation of structured knowledge within Wikipedia. The Kylin/KOG System [22] is a part of the Intelligence in Wikipedia project and automatically verifies semantically enhanced data by explicit community feedback. The DBPedia project [3] extracts structured information from Wikipedia infoboxes and stores it as RDF-triples. YAGO [17] is also based on Wikipedia data and tries to semantically enhance this data. Nevertheless, all these approaches do not provide any support or recommendations regarding the structure and schema. The paper by Doan *et al.* [8] provides a very overview about current mass collaboration systems on the web. Ontowiki [4] is an online collaboration system which is also based on RDF data. Users are able to create structured knowledge bases, navigate through the data and visualize the stored data. Ontowiki provides its users with a basic autocompletion feature during the insertion of information. However, any further guidance contributing to the structuring of data is not available to the users of Ontowiki. Semantic Wikipedia [19] extends MediaWiki, the Wiki software used by Wikipedia by adding typed links between Wikipedia entries as well as attributes and types. However, the user is not guided in the process of specifying this additional semantics. Cimple/DBLife [7] presents an approach to build a structured community portal from already existing community sources where users can subsequently add more structured and also unstructured information in the form of Wiki pages. ExDB [6] extracts information from the web, adds structure and is then able to query this data in a structured manner. Another approach is freebase [5] which basically lets users build schemata about certain topics. These collections can then be linked based on a graph structure of all collections. Freebase also provides autocompletion. The authors stated in [18] that a fast autocompletion features significantly cut down duplicate entries. Google Fusion Tables [11] allow its users to efficiently collaboratively manipulate tabular data, connect these tables and visualize the table data. However, both approaches only support predefined schemata and do not allow for dynamic, self-adapting schemata.

As for the recommender system within SnoopyDB, many publications are focused around collaborative filtering. [15] and [1] provide a good overview about the different approaches and the state-of-the-art. The computation of recommendations in the case of SnoopyDB approach is mainly concerned with the detection of substructures within semistructured information, which is covered in [14] and [2]. However, traditional approaches for recommender systems like e.g. collaborative filtering are hardly scalable for huge amounts of data. As SnoopyDB features a very efficient algorithm for the computation of recommendations it scales well even in high-volume information systems.

## 7. CONCLUSION

In this paper, we presented the Snoopy Concept for collaborative, semistructured online information systems. The concept aims at creating a common schema by providing the user with suitable recommendations for properties and values. The benefit of such an approach is that an aligned, homogeneous schema is created and maintained and that users tend to enter more information into the system. We have shown the broad acceptance of

the recommended items (49% of properties were reused) which resulted in the fact that users entered more information (31% more data). Furthermore, the information was aligned (32% less distinct properties), featured a common schema and increased the amount of data and its confidence by "snooping" as much information as possible.

## REFERENCES


[1] G. Adomavicius and A. Tuzhilin. "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions". *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.

[2] T. Asai, K. Abe, S. Kawasoe, H. Arimura, H. Sakamoto, and S. Arikawa. "Efficient substructure discovery from large semi-structured data", Second SIAM International Conference on Data Mining, Arlington, VA, USA, pages 158–174, 2002.

[3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. "Dbpedia: A nucleus for a web of open data" *Lecture Notes in Computer Science*, 4825:722, 2007.

[4] S. Auer, S. Dietzold, and T. Riechert. "OntoWiki–A tool for social, semantic collaboration". The Semantic Web-ISWC 2006, Athens, GA, USA, pages 736–749, 2006.

[5] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. "Freebase: a collaboratively created graph database for structuring human knowledge", 2008 ACM SIGMOD international conference on Management of data, Vancouver, CA, pages 1247–1250, 2008.

[6] M. J. Cafarella, C. Re, D. Suciu, and O. Etzioni, "Structured querying of web text data: A technical challenge", Conference on Innovative Data Systems Research, Asilomar, CA, USA, pages 225–234, 2007.

[7] P. DeRose, W. Shen, F. Chen, A. Doan, and R. Ramakrishnan, "Building structured web community portals: a top-down, compositional, and incremental approach", 33rd international conference on Very large data bases, Vienna, Austria, pages 399–410, 2007.

[8] Anhai Doan, Raghu Ramakrishnan, and Alon Y. Halevy. "Mass collaboration systems on the World Wide Web", *Communications of the ACM*, 2010.

[9] G.W. Furnas, T.K. Landauer, L.M. Gomez, and S.T. Dumais, "The vocabulary problem in human-system communication", *Communications of the ACM*, 30(11):971, 1987.

[10] W. Gassler, E. Zangerle, M. Tschuggnall, and G. Specht, "SnoopyDB Narrowing the Gap between Structured and Unstructured Information using Recommendations", 21st ACM Conference on Hypertext and Hypermedia, Toronto, Ontario, Canada, pages 271–272, June 2010.

[11] Google Inc, "Google Fusion tables", January 2011, Available: http://www.google.com/fusiontables.

[12] B. Leuf and W. Cunningham, THE WIKI WAY: QUICK COLLABORATION ON THE WEB, Addison-Wesley, Boston, 2001.

[13] B. McBride. "The resource description framework (rdf) and its vocabulary description language rdfs", HANDBOOK ON ONTOLOGIES, pages 51–66, 2004.

[14] T. Miyahara, Y. Suzuki, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda, "Discovery of frequent tag tree patterns in semistructured web documents". *Advances in Knowledge Discovery and Data Mining*, pages 341–355, 2002.

[15] P. Resnick and H.R. Varian, "Recommender systems", *Communications of the ACM*, 40(3):58, 1997.

[16] P. Shvaiko and J. Euzenat, "A survey of schema-based matching approaches*", Journal on Data Semantics*, 4:146–171, 2005.

[17] F. Suchanek, G. Kasneci, and G. Weikum, "Yago: A large ontology from wikipedia and wordnet", Web Semantics: Science, Services and Agents on the World Wide Web,. World Wide Web Conference 2007, Semantic Web Track, 6(3):203 – 217, 2008.

[18] J. Taylor, "The stone soup of data", Panel at WWW 2007, Januar 2011, Available: http://km.aifb.uni-karlsruhe.de/ws/ckc2007/StoneSoup-www2007.pdf.

[19] M. Voelkel, M. Kroetzsch, D. Vrandecic, H. Haller, and R. Studer, „Semantic Wikipedia". 15th international conference on World Wide Web, pages 585–594, Edinburgh, Scotland, 2006.

[20] G. Weikum, G. Kasneci, M. Ramanath, and F. Suchanek., "Database and information-retrieval methods for knowledge discovery", *Communication of the ACM*, 52(4):56–64, 2009.

[21] D.S. Weld, F. Wu, E. Adar, S. Amershi, J. Fogarty, R. Hoffmann, K. Patel, and M. Skinner, "Intelligence in Wikipedia", Twenty-Third Conference on Artificial Intelligence (AAAI'08), Chicago, IL, USA, 2008.

[22] F. Wu and D. S. Weld., "Automatically refining the wikipedia infobox ontology", 17th international conference on World Wide Web, pages 635–644, New York, NY, USA, 2008.

[23] E. Zangerle, W. Gassler, and G. Specht, „Recommending structure in collaborative semistructured information systems", Fourth ACM conference on Recommender systems, pages 261–264, Barcelona, Spain, 2010.