Model-aware Wiki Analysis Tools: the Case of HistoryFlow

Oscar Díaz
oscar.diaz@ehu.es

Gorka Puente
gorka.puente@ehu.es

ONEKIN Research Group
University of the Basque Country
San Sebastián, Spain
www.onekin.org

ABSTRACT

Wikis are becoming mainstream. Studies confirm how wikis are finding their way into organizations. This paper focuses on requirements for analysis tools for corporate wikis. Corporate wikis differ from their grow-up counterparts such as *Wikipedia*. First, they tend to be much smaller. Second, they require analysis to be customized for their own domains. So far, most analysis tools focus on large wikis where handling efficiently large bulks of data is paramount. This tends to make analysis tools access directly the wiki database. This binds the tool to the wiki engine, hence, jeopardizing customizability and interoperability. However, corporate wikis are not so big while customizability is a desirable feature. This change in requirements advocates for analysis tools to be decoupled from the underlying wiki engines. Our approach argues for characterizing analysis tools in terms of their abstract analysis model (e.g. a graph model, a contributor model). How this analysis model is then map into wiki-implementation terms is left to the wiki administrator. The administrator, as the domain expert, can better assess which is the right terms/granularity to conduct the analysis. This accounts for suitability and interoperability gains. The approach is borne out for *HistoryFlow*, an IBM tool for visualizing evolving wiki pages and the interactions of multiple wiki authors.

Categories and Subject Descriptors

D.2.2 [**Software Engineering**]: Design Tools and Techniques; D.2.12 [**Software Engineering**]: Interoperability; H.5.3 [**Information Interfaces and Presentation**]: Group and Organization Interfaces—*Collaborative computing*

General Terms

Human Factors, Management, Design

Keywords

MDE, Information visualization, Wiki, interoperability, analysis tools, web 2.0, collaboration, visualization, model-driven

WikiSym '10 July 7-9, 2010, Gdańsk, Poland

1. INTRODUCTION

Methods for wiki analysis have been proposed along four perspectives [15]: social perspective (who knows who), knowledge perspective (who knows what), information perspective (what refers to what), and temporal perspective (what was done before). To this end, distinct tools are available: *Sonivis* [9], *Wiki Explorator* [11], *Sioc MediaWiki* [8], *WikiTracer* [12], *HistoryFlow* [29, 30], to name a few. Most tools focus on large wikis such as *Wikipedia* where efficient handling of large bulk of data is key. This forces a tight coupling between the tool and the underlying wiki engine (e.g. *MediaWiki*) so that the terms of analysis (e.g. nodes, edges) tend to be coupled to how these notions are realized in the wiki (e.g. wiki pages, wiki users). Hence, it is not uncommon for tools to bound domain concepts to wiki notions. If the tool binds nodes to wiki articles then, analysts are prevented from using this tool to study other wiki concepts where nodes could have stood for other notions (e.g. wiki users, revisions, etc).

This binding can be justified on an efficiency basis, specially when facing large wikis. However, not all wikis are so large. Indeed, wikis are being massively adopted in small-and-medium organizations according to the Intranet 2.0 Global Survey report [7]: 46% of the respondent companies were using wikis. Our premise is that this new crop of wikis is characterized by (1) being of a smaller size that their Wikipedia-like predecessors, and (2) analysis requirements being domain-specific. These premises would imply the removal of our previous conjecture for large wikis, and hence, a pressure to make explicit the analysis model for these tools (rather than being bound to its realization in a concrete wiki engine).

This paper advocates for making wiki analysis tools "model aware". That is, tools are characterized in terms of their abstract analysis models (e.g. a graph model, a contributor model, a collaboration model). How this analysis model is then map into wiki-implementation terms is left to the wiki administrator who, as the domain expert, can better assess which is the right granularity to conduct the analysis. This approach is illustrated for *HistoryFlow* [29, 30].

HistoryFlow is a tool for visualizing dynamic, evolving documents and the interactions of multiple collaborating authors. In its current implementation, *HistoryFlow* is being used to visualize the evolutionary history of wiki pages on *Wikipedia*. This assumption is engineered in the tool itself. For instance, you can feed *HistoryFlow* with the URL of a *Wikipedia* page to visualize the evolution of this page's content. The subject of analysis is then a wiki page, specifically, how users make a wiki page evolve.

However, other useful scenarios can be envisaged for *HistoryFlow*. Let's consider the use of wikis for collaboratively capturing system requirements [20]. Wikis can collect software requirements from

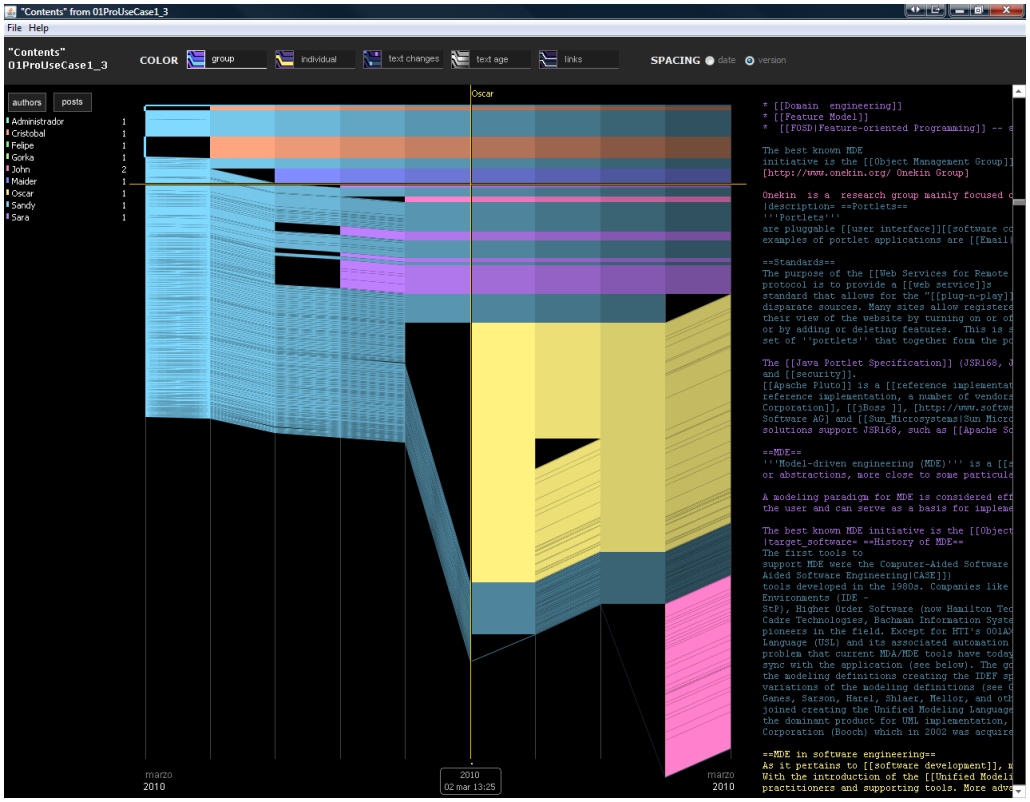

Figure 1: Standard case: useCase x stakeholder. These dimensions match the granularity level at which content is captured at the wiki. That is, wiki pages stand for use cases whereas wiki users represent stakeholders.

the distinct stakeholders in different projects. Stakeholders become wiki users whereas use cases (a common mechanism for capturing system requirements) are realized as wiki pages. In this scenario, *HistoryFlow* can serve to analyze the evolution and contribution of the distinct stakeholders to the use cases. Nevertheless, the usefulness of *HistoryFlow* does not stop here. It can also be of interest to monitor the evolution of the whole project (which encompasses a set of wiki pages), and not just its individual use cases (a single wiki page). Likewise, the *contribution* can be assessed based on the stakeholders' departments rather than on a stakeholder basis (e.g. to see the engagement of the department in a project). What has changed is the abstraction level at which the analysis is conducted. Rather than thinking in terms of wiki users and wiki pages, *HistoryFlow* becomes a tool for visualizing *"contributors"* and *"subject of contributions"*. The most straightforward mapping between these abstract notions and the wiki realization is the one natively supported by *HistoryFlow*: *contributor = wiki user; subject of contribution = wiki page*. However, wiki users and wiki pages could not always provide the right granularity for the analysis. Coarser-grained or finer-grained analysis could be needed. And *HistoryFlow* could have been used in those cases as long as the terms of the analysis would have not been hardwired.

This approach implies a trade-off between efficiency and suitability. The former refers to the indirection cost: wiki data is first converted into a model, and next, this model feeds the tool. For large wikis, this cost is prohibitive. Notice however, that most wikis are by far less large than *Wikipedia*, and hence, this cost could pay-off in terms of both suitability and interoperability. The former refers to the appropriateness of the tool's functions. Model-aware tools facilitate to tune the model for the analysis at hand. Additionally, interoperability refers to the ability to exchange and use information from different systems to enable them to operate effectively together. In our context, interoperability permits domain experts to easily switch between analysis tools so that the very same wiki data can be visualized and measured through distinct tools.

The rest of this paper is organized as follows. Section 2 outlines *HistoryFlow* as a representative of wiki analysis tools. Section 3 describes the approach followed to make it model aware. The claimed advantages of suitability and interoperability are next addressed in Sections 4 and 5, respectively. Section 6 recaps the trade offs of this approach. Related work and conclusions end the paper.

2. THE CASE OF HISTORYFLOW

History Flow [29, 30] is a tool for visualizing evolving wiki pages and the interactions of multiple wiki authors. As an example, consider the use of wikis for collaboratively capturing system requirements [20]. Wikis can collect software requirements from the distinct stakeholders in different projects. Stakeholders become wiki users whereas use cases (a common mechanism for capturing system requirements) are realized as wiki pages. In this scenario, *HistoryFlow* can serve to analyze the evolution and contribution of the distinct stakeholders to use cases.

Figure 1 shows the *HistoryFlow* visualization. Vertical axes stand for page versions whereas colors represent stakeholders (i.e. wiki users). *HistoryFlow* "connects text that has been kept the same between consecutive versions; in other words, it connects corresponding segments on the lines representing versions. Pieces

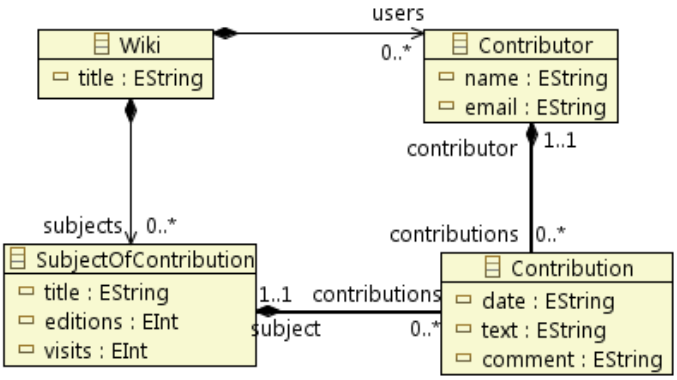

Figure 2: The *HistoryFlow* analysis (meta) model

of text that do not have correspondence in the next (or previous) version are not connected and the user sees a resulting "gap" in the visualization; this happens for deletions and insertions". This rendering permits to conduct a two-dimensional analysis, where the granularity is set as *pages x authors*. (hereafter referred to as the **"base granularity"**). This base granularity is favored by *HistoryFlow* in the sense that is engineered in the tool itself (e.g. you can feed *HistoryFlow* with the URL of a *Wikipedia* page to visualize the evolution of this URL's content). This partially stems from analysis being conducted for large wikis (*Wikipedia* is a case in point) where tight coupling between the wiki engine and the analysis tool accounts for efficiency.

However, wikis are being used for intra-organization collaborative content production [31]. For the purpose of this work, the implications are twofold. First, this scenario exhibits much smaller wikis (at least compare with *Wikipedia*), and hence, relaxing efficiency demands. Second, analysis is conducted in domain-specific terms and so should it be the companion tools. For instance, on our sample case of wikis being used for requirement gathering, it can be of interest to monitor the evolution of the whole project (which encompasses a set of wiki pages) and not just its individual use cases (a single wiki page). Likewise, the *contribution* can be assessed based on the stakeholders' departments rather than on a stakeholder basis (e.g. to see the engagement of departments in distinct projects).

Unfortunately, the terms of analysis tend to be hard-coded in the tool itself, making very cumbersome to use these tools for scenarios other than the base granularity. Therefore, rather than describing *HistoryFlow* analysis in terms of wiki pages and wiki users it would more convenient to abstract them into *"subject of contribution"* and *"contributor"*, respectively, leaving to the user (which after all is the domain expert) to decide both what is the *subject of contribution*, and who is the *contributor* for the analysis at hand. This basically involves making explicit the analysis model, and let the user do the mapping between wiki artefacts (articles, categories, templates, etc) and the primitives of this analysis model. In so doing, *HistoryFlow* becomes a tool for visualizing *"contributors"* and *"subject of contributions"* rather than *"wiki users"* and *"wiki pages"*. Next section looks at the details.

3. MAKING HISTORYFLOW MODEL AWARE

Broadly, analysis tools encompass three main endeavors: data extraction + analysis algorithm + fancy interfaces. Distinct benefits

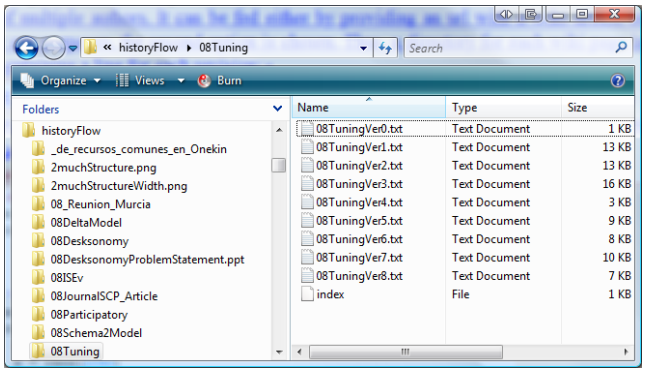

Figure 3: *HistoryFlow* directory structure*:* each wiki article accounts for a directory which contains a text file for each revision.

can be obtained by externalizing data extraction out of analysis tools so that the terms of analysis are not hardwired to their wiki realization. So far, *HistoryFlow* focuses on visualizing how wiki users contribute to wiki pages. However, this useful tool can be leveraged by moving from wiki users and wiki pages to the more abstract notions of *"contributors"* and *"the subject of contributions"*, respectively. Who is a *contributor* and what is a *contribution* is not hardwired into *HistoryFlow* but decided by the analyst. The tool is then described in terms of an "analysis model".

Figure 2 shows the analysis model for *HistoryFlow*. The model makes explicit the terms in which *HistoryFlow* conducts the analysis: the *contributor* and the *subject of contribution*. Ideally, the analysis model should be described along a standard such as UML or ECORE [27] (the Eclipse realization of OMG's MOF [23]). This would facilitate the interoperability with other tools. However, this is seldom the case. Either the tool directly accesses the wiki database or at best, it provides some input parameters. For instance, besides providing the URL of a *Wikipedia* article, *HistoryFlow* also admits to be fed from a set of files containing page histories. Figure 3 provides an example. Each wiki article to be analyzed accounts for a directory. This directory contains an index, and a *.txt* file for each revision made to this article. The index file holds an entry for each revision along the following format:

File name [tab] Author [tab] Comment [tab]
YYYY–MM-DD HH:mm: ss

```
1  main do
2    HistoryFlow::Wiki.all_objects.each do |g|
3      g.subjects.each do |sub|
4        $count = 0
5        $contribution = ""
6        sub.contributions.each do |con|
7          compose_file(Helper.normalize(sub.title) + '/' +
8            con.subject.title + 'Ver' + $count.to_s + '.txt') do
9            apply_rule :contributions, con
10         end
11         $count += 1
12       end
13       compose_file(Helper.normalize(sub.title) + '/' + 'index') do
14         apply_rule :index, sub
15         ...
16 rule 'contributions' do
17   from HistoryFlow::Contribution   do
18     text do
19       println self.text
20       date = Time.parse(self.date).strftime('%Y-%m-%d %H:%M:%S')
21       # File name [tab] Author [tab] Comment [tab] YYYY-MM-DD HH:mm:ss
22       if self.contributor.nil?
23         contributor = 'anonymous'
24       else
25         contributor = self.contributor.name
26       end
27       $contribution = $contribution + self.subject.title +
28       "Ver#{$count}.txt\t#{contributor}\t#{self.comment}\t#{date}\n"
29       ...
30 rule 'index' do
31   from HistoryFlow::SubjectOfDiscussion do
32     text do
33       println $contribution
```

Figure 4: Mapping analysis primitives down to *HistoryFlow* files (partial view) using *RubyTL*.

We can use this back door to feed *HistoryFlow* but leaving open what directories and files stand for. Better said, directories are the realization of the abstract notion *"subject of contribution"* as defined in the ECORE model at Figure 2. However, what is a *subjectOfContribution* is no longer in the realms of *HistoryFlow* but it is for the domain expert to decide. The domain expert extracts the data himself, and builds the *HistoryFlow* model. Next, this model is mapped into the internal data structures of *HistoryFlow* (i.e. directories and files). It is most important to note that this transformation process should be transparent to the expert himself. Mapping models to internal data structures should be encapsulated as part of the tool[1] .

The good news is that transformation languages exist that facilitate this task. *MOFScript* [22] or *RubyTL* [16] are two examples. Figure 4 shows the mapping from the *HistoryFlow* model down to *HistoryFlow* directories. The enactment of this transformation generates the directories which are then consulted by the *HistoryFlow* engine. Basically, the transformation generates (see figure 4) a directory for each *"subjectOfContribution"* and a *.txt* file for each *"contribution"* (lines 1-9). The content of the index file is generated in the *'contributions'* rule (lines 16-29) and stored in the *"$contribution"* variable (lines 27-28). The index file is populated in the *'index'* rule (lines 30-33) with the information of the previous rule.

From now on, *HistoryFlow* becomes model aware, i.e. it can be fed through an *ECORE* model. This accounts for suitability and interoperability. Next sections delve into the details.

4. MODEL AWARENESS ACCOUNTS FOR SUITABILITY

According to ISO9126, suitability is a functionality characteristic

[1]As we have no access to the internals of *HistoryFlow*, this process is now explicit to end users who need to enact the transformation themselves before using *HistoryFlow*. However, this shows the feasibility of *HistoryFlow* to be directly fed through models.

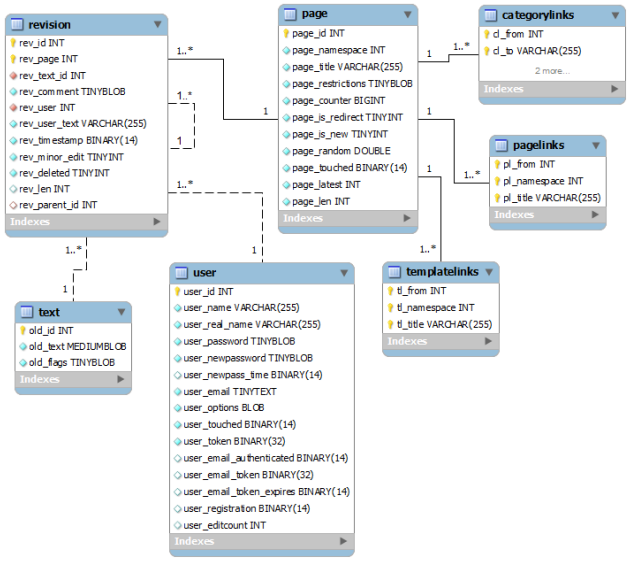

Figure 5: Partial view of *MediaWiki* database schema.

that refers to the appropriateness (to specification) of the functions of the software. Models abstract from the concrete realization of the model in a given setting. This permits users to map the analysis model of the tool to their own terms. The mapping natively provided by *HistoryFlow* is for *"contributor"* and *"the subject of contribution"* to be realized as *"wiki user"* and *"wiki page"*, respectively (referred to as the *base granularity*). However, other granularities are now possible and this enhances *HistoryFlow* suitability. Some examples follow.

Coarse-grained analysis. Wiki users can be grouped into meaningful dimensions for the analysis, e.g. based on their department, job status, income, background, etc. The contributor is no longer a user but a department (e.g. *user@marketing*, *user@sales*, etc) Likewise, wiki pages can conform to higher order notions, e.g. if wiki pages stand for use cases, then the analysis could be conducted in a project basis where a set of wiki pages is analyzed together. The *subject of contribution* is no longer a wiki page but a project realized as a set of wiki pages (e.g. *page@project1*, *page@project2*, etc). Figure 6 shows a *HistoryFlow* visualization where user contributions are grouped based on the user's company. Notice that this company-based figure is shaped like the user-based one in Figure 1 since the evolution of the content is the same. However, color strips are just three, providing a quick overview of the engagement of each company in terms of their employees' participation.

Fine-grained analysis. The wiki page can be too coarse grained. In some cases, wiki pages exhibit some common structure that can be explicitly captured through wiki templates. Commonly cited advantages of wiki templates include: enforcing a uniform layout, ensuring that all available relevant information is provided, lack of information is made explicit, and easing comparison of wiki pages. For our running example, wiki templates have been proposed to collect information of the use cases [10]. The semantics of use cases is captured in terms of impact (in terms of audience or improvement of a series of aspects), required existing standards, limitations know at this stage, expected problems for implementation or deployment, or tools expected to implement these recommendations. These template sections can then become the focus of analysis rather than the whole use case (i.e the wiki

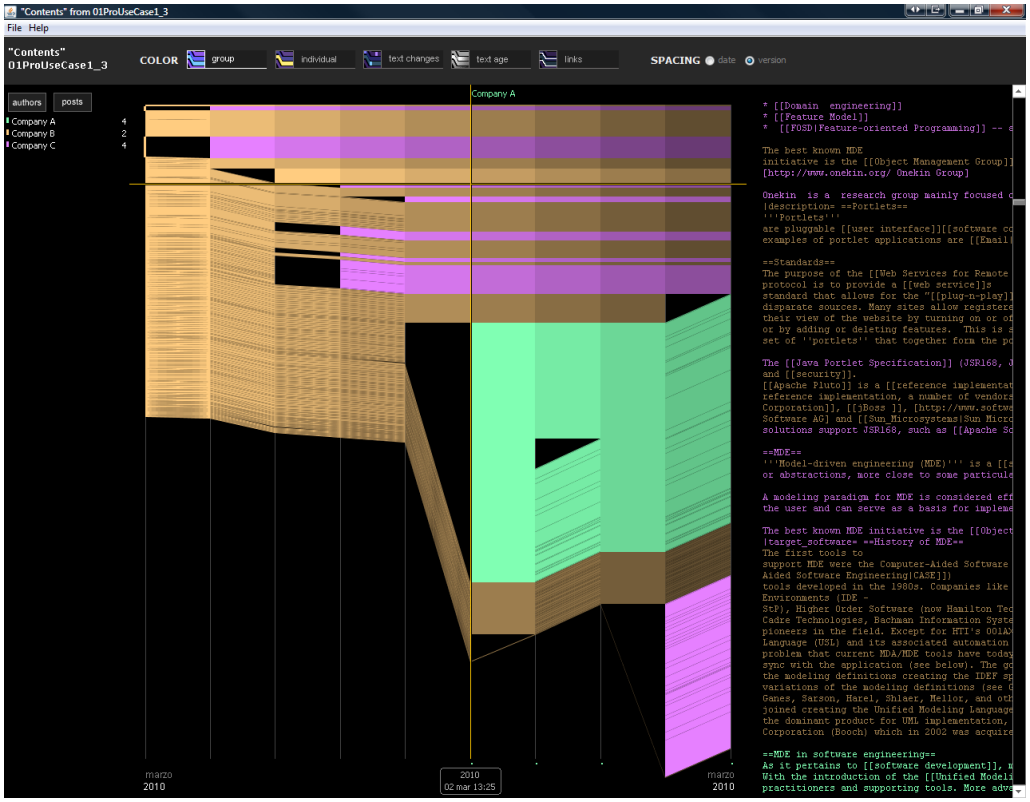

Figure 6: Coarse grained granularity: *usecase x company*.

page). The analyst may be interested in assessing the contribution of the "impact" section. That is, the *subjectOfContribution* is no longer the wiki page but template sections (e.g. *page#impact*, *page#targetAudience*).

The bottom line is that by making explicit the analysis model, domain experts are able to tune the tool analysis (meta)model for their own purposes. But this is not a free lunch. Model awareness implies a stronger user involvement. Previously, *HistoryFlow* digs directly into the wiki database. Now, HistoryFlow is fed through an analysis model. This implies that the domain expert needs to construct such an analysis model from the wiki database. This is the topic of the next subsection.

4.1 Coming Up with the Analysis Model

The wiki database provides the raw material to build up the analysis model. Figure 5 shows the database schema for *MediaWiki*. *"The contributor"* can be obtained from a single tuple in the *user* table or be subject to a more elaborate process. This process is far from simple and this very fact vindicates the claim of separating concept extraction from concept analysis, both complex undertakings on their own right. So far, most tools embed both processes. The good news is again that dedicated transformation languages exist from model extraction (a.k.a. model harvesting). *Schemol* [17] is a case in point.

Figure 7 shows a *Schemol* transformation that obtains a *HistoryFlow* model out of *MediaWiki* database tuples. This transformation accounts for "the base granularity". The transformation starts from the top rule (line 1). This rule generates the *"wiki"* element of the *HistoryFlow* model. This element is related with *"contributor"* and *"subjectOfContribution"* through the *"users"* and *"subjects"* relationships, respectively (see Figure 2). This rule generates such links out of the tuples in the *"page"* and *"user"* tables (lines 6-7). These assignments implicitly cause the triggering of the *"subjectOfContribution"* and *"contributor"* rules. The former leads to the creation of distinct *"subjectOfContribution"* elements from the attributes of the *"page"* and *"revision"* tables (lines 15-18). The filter restrict the mapping to those pages whose name space is 0 which stands for articles (rather than categories, templates or discussion pages) (line 12-13). Notice how the population of *"contributions"* implicitly leads to create *"contribution"* elements out of the *"revision"* table (line 20-27). In this way, primitives of analysis are obtained from *MediaWiki* tuples. Maintainability wise, the important point is that now this mapping is explicit rather than being hard-coded in the tool itself.

Figure 8 shows a *Schemol* transformation for a coarser granularity: articles are grouped based on the project they belong to, whereas users are clustered based on their companies. Rule *"SubjectOfContribution"* identifies the membership of projects based on the project prefix (line 15-21). Rule *"Contributor"* generates just three elements for companies *A, B* and *C* where membership of wiki users is explicitly stated (lines 31-37). As for *"contributions"*, versions in articles on a user basis, should now be aggregated to obtain versions in projects caused by companies. That is, contributions made on articles belonging to the same project and conducted by users of the same company, should now be regarded as *contributions* on the very same *subject of contribution.* This requires some string processing before article-user contributions (as stored on the *MediaWiki* database) can become project-company contributions on the *HistoryFlow* model. It is this complexity (and its flexibility companion) that vindicates the fact of moving data extraction outside the tool realm. The discussion section will delve into this issue.

```
rule 'wiki'
  from Database db
  to HistoryFlow::Wiki wiki
  mapping
    wiki.title = "HistoryFlow";
    wiki.subjects = db.@page;
    wiki.users = db.@user;
end_rule
rule 'subjectOfDiscussion'
    from wikidb::page page
    to HistoryFlow::SubjectOfContribution subj
    filter
      page.page_namespace == 0
    mapping
        subj.title = page.page_title;
        subj.visits = page.page_counter;
        subj.editions = page.@revision.count();
        subj.contributions = page.@revision;
end_rule
rule 'contribution'
    from wikidb::revision revi
    to HistoryFlow::Contribution cont
    mapping
        cont.date = revi.rev_timestamp;
        cont.text = revi.rev_text_id.old_text;
        cont.comment = revi.rev_comment;
        cont.contributor = revi.rev_user;
end_rule
rule 'contributor'
    from wikidb::user user
    to HistoryFlow::Contributor cont
    mapping
        cont.name = user.user_name;
        cont.email = user.user_email;
end_rule
```

Figure 7: *Schemol* transformation for the base granularity: *"article"* and *"wiki user"* becomes *"the subjectOfContribution"* and *"the contributor"*, respectively.

```
rule 'wiki'
  from Database db
  to HistoryFlow::Wiki wiki
  mapping
    wiki.title = "HistoryFlow";
    wiki.subjects = db.@page;
    wiki.users = db.@user;
end_rule
rule 'subjectOfDiscussion'
    from wikidb::page page
    to HistoryFlow::SubjectOfDiscussion subj
    filter
        page.page_namespace == 0
    mapping
        if page.title.startswith("01Pro")
            subj.title ="01Pro";
        else if page.title.startswith("02Pro")
            subj.title = "02Pro";
        else if page.title.startswith("03Pro")
            subj.title = "03Pro";
        ...
        subj.contributions = page.@revision;
end_rule
rule 'contribution'
  ...
end_rule
rule 'contributor'
    from wikidb::user user
    to HistoryFlow::Contributor cont
    mapping
        if user.user_name in ("Oscar", "Felipe", "Administrador", "Gorka")
            cont.name ="Company A";
        else if user.user_name in ("Sandy", "Salva", "Maider")
            cont.name = "Company B";
        else if user.user_name in ("Cristobal", "Sara", "John")
            cont.name = "Company C";
        ...
```

Figure 8: *Schemol* transformation for a coarser granularity: *"project"* and *"company"* becomes *"the subjectOfContribution"* and *"the contributor"*, respectively.

5. MODEL AWARENESS ACCOUNTS FOR INTEROPERABILITY

Interoperability refers to the ability to exchange and use information from different systems to enable them to operate effectively together. In our context, interoperability permits domain experts to easily switch between analysis tools so that the very same data can be visualized and measured through distinct tools. In the current panorama where measures and visualization of wiki content and collaboration are intensively being explored, interoperability permits to join forces among tools. Rising the level of abstraction (through models) is a well-known approach to facilitate interoperability. This section provides an example where the *HistoryFlow* model can now be used to feed other tools, in particular *Cytoscape* [26].

Cytoscape is a software for visualizing molecular interaction networks and biological pathways. Plugins are available to extend the base functionality with fancy utilities[2]. One of these plugins is *NetMatch [19]. NetMatch* allows searching biological networks for subcomponents matching a given network pattern.

Even though *Cytoscape* was designed for biological research, it can be used for other kinds of complex networks. Wikis can be complex networks. We can then map the *HistoryFlow* model to the *NetMatch* graph model so that the very same data can be visualized both *à la HistoryFlow* and *à la Cytoscape*. Such mapping is described in Table 1. Specifically, *"subjectOfContribution"*, *"contributor"* and *"contribution"* become nodes of different types depicted as ellipses, diamonds and triangles, respectively. Model relationships are turned into edges. The mapping is straightforward, and it takes less than an hour to write the *RubyTL* program counterpart.

[2]http://chianti.ucsd.edu/cyto_web/plugins/index.php

Now, the *HistoryFlow* model can be visualized along *Cytoscape*. Specifically, we can enjoy the benefits of using *NetMatch* to detect collaboration patterns among contributors. Figure 9 (left handside) depicts a *NetMatch* query that looks for *subjectOfContributions* where Oscar, Maider and Gorka have contributed at any time. The answer is shown at the right handside in Figure 9. A second example is depicted in Figure 10. Now, *subjectOfContributions* demand not only the common collaboration of Oscar, John and Gorka but this collaboration being achieved sequentially. The relationship between contributions indicate this temporal-ordering constraint. The answer is shown as a subgraph in the right handside. The expert can now click on any of these subgraphs to obtain more details about the context and time where this pattern arises.

6. DISCUSSION

We advocate for externalizing data extraction from wiki analysis tools by using a model-driven approach. Models should be the primary representation for tools. Since they are not yet the primary representation, an extra initial effort is needed to inject and extract the tool data into a model. Figure 11 depicts the architecture. Wiki data is moved to the target tool (e.g. *HistoryFlow*) by passing through a pivot metamodel (i.e. the analysis model) enabling a star architecture of transformations. Order of n transformations (between the wiki engine and the pivot) are thus sufficient for transforming to n tools, compared to the order of n^2 transformations required in traditional point-to-point *ad-hoc* data extraction.

The approach decouples analysis model processing (e.g. visualization) from analysis model population (e.g. how the model is obtained). This accounts for the following benefits. First, it facilitates separation of concerns. While coding analysis algorithms requires a sound mathematical background, data extraction demands programming skills. Second, decoupling augments tool suitability since the analysis tool can now be used in

Graph element	(Meta)Model element
Node	Contributor
Node	SubjectOfContribution
Node	Contribution
Edge	Contributor to Contribution
Edge	Contribution to Contribution
Edge	SubjectOfContribution to SubjectOfContribution
Edge	Contribution to SubjectOfContribution

Table 1: Mapping between the (meta)model elements and the graph elements

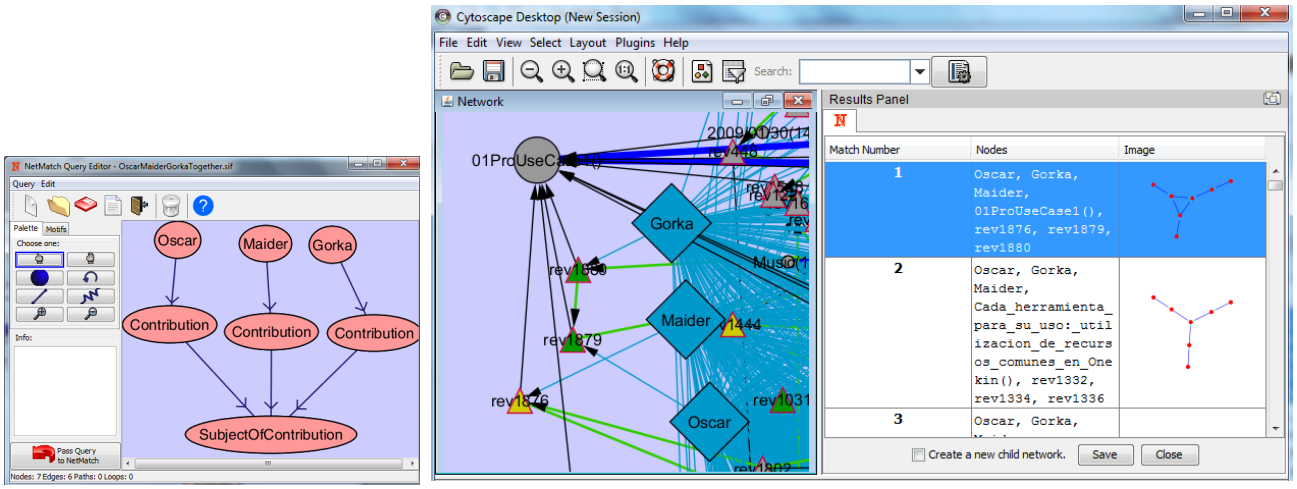

Figure 9: *NetMatch* pattern query: in which *subjectOfContributions* have Oscar, Maider and Gorka work at anytime together?

scenarios other than those initially thought. Third, it enhances tool interoperability by allowing the same model to be used by distinct tools, or be extracted from wiki engines with different database schemas.

On the downside, there exists an efficiency penalty due to the indirection caused by first moving to a model rather than directly obtaining the data from the wiki database. Notice that this cost is at transformation time and not while working with the analysis tool. Even though, this cost is enormous in the case of large wikis such as *Wikipedia*. However, corporate wikis are not so large, and this additional cost can be compensated by the gains in interoperability and suitability.

Finally, we see as the main drawback, the additional effort required for the domain expert to define appropriate mappings from the wiki database to the analysis model (e.g. coding the *Schemol* rules). So far, using *HistoryFlow* is just a question of providing the URL of the article of interest. This is in contrast with the effort of coding *Schemol* rules. But this is the penalty to be payed for having the freedom to decide the subject of analysis. In the medium run, mechanisms similar to those available for *OnLine Analytical Processing (OLAP)* tools can alleviate this situation (e.g. graphical tools, wizards, etc).

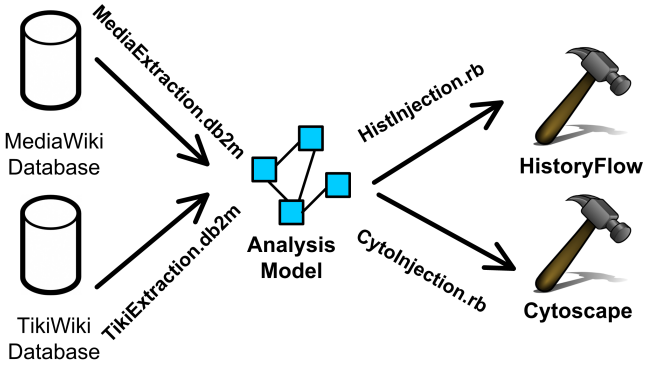

Figure 11: Analysis models as pivots to decouple wiki engines from wiki analysis tools. .

7. RELATED WORK

On the effort to depart from concrete wiki engines, our work aligns with that of Ferreira et al. [18] where a wiki metamodel is introduced for describing the structure of wikis for Requirements Engineering (RE) where distinct "semantic links" are introduced along the so-called semantic wikis [25]. Our work contributes to

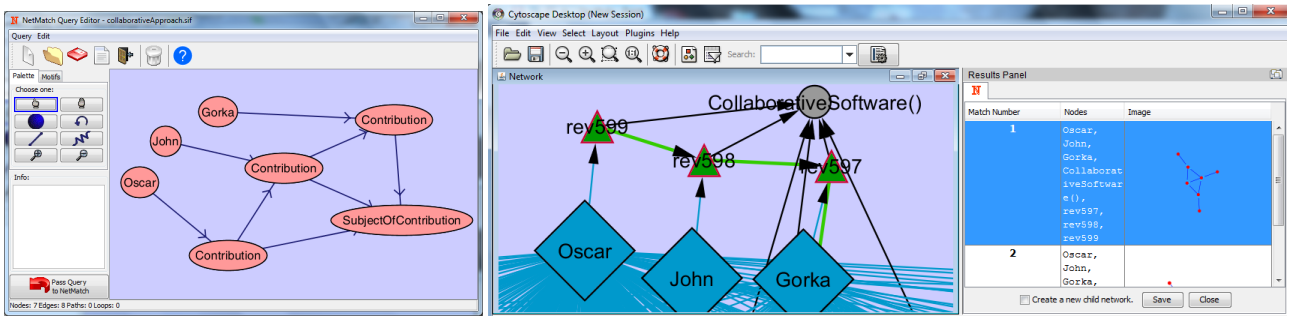

Figure 10: *NetMatch* pattern queries: in which subjectOfContributions have Oscar, John and Gorka work sequentially together?

these efforts by providing a case study on the benefits of abstracting for specific wiki engines, and the need for wiki standards that permit a market of wiki utilities. Similarly, all work on describing wikis as graphs (as an analysis model) is also of interest. Distinct authors abstract wikis as graphs where e.g. nodes correspond to wiki pages, and directed edges stand for hyperlinks among them [13, 14, 32, 21]. Using this formalism, studies have been conducted to provide different measures for the *Wikipedia* [32], to see the temporal evolution of wikis [14], or the structural similarities of complex networks using wikis as a test case [21]. Notice however that even here the syntax to express graphs can vary. Popular graph specification languages include the *Graph Description Language* (*.gdl*) [4], Graph Exchange Language (*.gxl*) [6], *GraphML* (*.graphml*) [5] and *DOT* (*.dot*) [3].

On the other hand, the Web Engineering community is actively supporting model-driven practices. There exists a series of workshops on this topic (see *http://mdwe2009.pst.ifi.lmu.de/*), and different experiences have been reported (refer to [24] for an overview). Our work contributes to this area by abstracting away into a platform-independent model (i.e. analysis model), and addressing the extraction of analysis models out of wiki databases. Furthermore, the MDE community is striving for getting interoperability through the use of models as main assets[1].

The ATHENA Model-Driven Interoperability (MDI) Framework [2] provides guidelines about how MDE approaches can be applied in achieving interoperable enterprise software systems. As Sun et al. [28] states "each specialized tool contributes to a crucial step in the development process", likewise the co-existence of distinct tools (e.g. *HistoryFlow*, *Cytoscape*) will more likely be necessary to assess the wiki collaboration process. Sun et al. [28] also highlights benefits learned from applying model transformation to the tool interoperability problem, such as separation of concerns across the integration process and adaptability and extensibility in defining new tools, among others.

8. CONCLUSIONS

Corporate wikis tend to be smaller than their *Wikipedia*-like predecessors. Analyzing the use and content of these corporate wikis can shed light on how the organization works. However, the terms of analysis tend to be domain dependent. Wiki articles and wiki users can stand for different realities within the organization. And analysis is to be conducted in terms of the domain, rather than in terms of how this domain is captured by the wiki platform. To this end, this paper introduces an indirect approach where the analysis tool is characterized through an analysis model. Then, it is up to the wiki administrator to map wiki terms into domain primitives. The approach is illustrated for *HistoryFlow*.

Future work includes to enrich analysis with data coming from other sources besides the wiki itself. Unlike *Wikipedia*-like cases, corporate wikis are not isolated islands but part of the Information System infrastructure of the organization. Multi-dimensional analysis of wiki data could then become possible.

Acknowledgments

This work is co-supported by the Spanish Ministry of Education, and the European Social Fund under contract TIN2008-06507-C02-01/TIN (MODELINE), and the University of the Basque Country under contract U07/09. Puente has a doctoral grant from the Spanish Ministry of Science & Education.

9. REFERENCES

[1] Architecture-Driven Modernization (ADM). Online at adm.omg.org.

[2] ATHENA (MDI) Framework. Online at www.modelbased.net/mdi/index.html.

[3] DOT. Online at www.graphviz.org/doc/info/lang.html.

[4] GDL. Online at www.aisee.com/gdl/nutshell.

[5] GraphML. Online at graphml.graphdrawing.org/index.html.

[6] GXL. Online at xml.coverpages.org/gxl.html.

[7] Intranet 2.0 Global Survey. Online at intranetblog.blogware.com/blog/archives/2009/5/15/4187339.html.

[8] Sioc MediaWiki. Online at ws.sioc-project.org/mediawiki/.

[9] Sonivis. Online at www.sonivis.org/.

[10] W3C Use Case Template. Online at www.w3.org/egov/wiki/Use_Case_Template.

[11] Wiki Explorator. Online at www.kinf.wiai.uni-bamberg.de/mwstat/.

[12] Wiki Tracer. Online at wikitracer.com/.

[13] A. Z. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A.Tomkins, and J. L. Wiener. Graph Structure in the Web. *Computer Networks*, 2000.

[14] L. Buriol, C. Castillo, D. Donato, S. Leonardi, and S. Millozzi. Temporal Analysis of the Wikigraph. In *Web Intelligence*, USA, 2006. IEEE Computer Society.

[15] K. M. Carley. Dynamic Network Analysis. In *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*, 2004.

[16] J. S. Cuadrado, J. G. Molina, and M. M. Tortosa. RubyTL: A Practical, Extensible Transformation Language. In *ECMDA-FA*, 2006.

[17] Oscar Díaz, Gorka Puente, Javier Luis Cánovas Izquierdo, and Jesús García Molina. Harvesting Models From Web 2.0 Application Databases. *Software and Systems Modeling (SOSYM)*, Model-Driven Web Engineering (Theme Issue), 2010. Pending Revision.

[18] D. Ferreira and A. Rodrigues da Silva. An Enhanced Wiki for Requirements Engineering. In *SEAA*, 2009.

[19] A.o Ferro, R. Giugno, G. Pigola, A. Pulvirenti, D. Skripin, G. D. Bader, and D. Shasha. Netmatch: a cytoscape plugin for searching biological networks. *Bioinformatics*, 2007.

[20] Panagiotis Louridas. Using wikis in software development. *IEEE Softw.*, 2006.

[21] A. Mehler. Structural Similarities of Complex Networks: a Computational Model by Example of Wiki Graphs. *Applied Artificial Intelligence*, 2008.

[22] J. Oldevik, T. Neple, R. Grønmo, J. Ø. Aagedal, and Arne-Jørgen Berre. Toward standardised model to text transformations. In *ECMDA-FA*, 2005.

[23] OMG. MetaObject Facility (MOF) Specification, January 2006. Online at www.omg.org/spec/MOF/2.0/PDF/f.

[24] G. Rossi, O. Pastor, D. Schwabe, and L Olsina, editors. *Web Engineering: Modelling and Implementing Web Applications*. Springer, 2008.

[25] S. Schaffert, F. Bry, J. Baumeister, and M. Kiesel. Semantic Wikis. *IEEE Software*, 2008.

[26] P Shannon, A Markiel, O Ozier, N S Baliga, J T Wang, D Ramage, N Amin, B Schwikowski, and T Ideker. Cytoscape: a Software Environment for Integrated Models of Biomolecular Interaction Networks. *Genome Research*, 13, 2003.

[27] Dave Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. *EMF: Eclipse Modeling Framework*. Addison-Wesley, 2008.

[28] Y Sun, Z Demirezen, Fc Jouault, R Tairas, and J Gray. A Model Engineering Approach to Tool Interoperability. In *SLE*, 2008.

[29] F. B. Viégas, M. Wattenberg, J. Kriss, and F. van Ham. Talk before you type: Coordination in wikipedia. In *HICSS*, 2007.

[30] F. B. Viégas, M.n Wattenberg, and K. Dave. Studying Cooperation and Conflict Between Authors With *History Flow* Visualizations. In *CHI*, 2004.

[31] Christian Wagner and Ann Majchrzak. Enabling customer-centricity using wikis and the wiki way. *J. Manage. Inf. Syst.*, 06-7.

[32] T. Zesch and I. Gurevych. Analysis of the Wikipedia Category Graph for NLP Applications. In *NAACL-HLT*, 2007.