Extraction of Procedural Knowledge from the Web

A comparison of two workflow extraction approaches.

Pol Schumacher
pol.schumacher@uni-trier.de

Mirjam Minor
minor@uni-trier.de

Kirstin Walter
walt4701@uni-trier.de

Ralph Bergmann
bergmann@uni-trier.de

University of Trier
Department of Business Information Systems II
54286 Trier, Germany

ABSTRACT

User generated Web content includes large amounts of procedural knowledge (also called how to knowledge). This paper is on a comparison of two extraction methods for procedural knowledge from the Web. Both methods create workflow representations automatically from text with the aim to reuse the Web experience by reasoning methods. Two variants of the workflow extraction process are introduced and evaluated by experiments with cooking recipes as a sample domain. The first variant is a term-based approach that integrates standard information extraction methods from the GATE system. The second variant is a frame-based approach that is implemented by means of the SUNDANCE system. The expert assessment of the extraction results clearly shows that the more sophisticated frame-based approach outperforms the term-based approach of automated workflow extraction.

Categories and Subject Descriptors

H.3.4 [**Web 2.0**]: World Wide Web (WWW)—*experience Web, experience reuse*; I.2.4 [**Knowledge Representation Formalisms and Methods**]: [Information extraction, Workflow extraction]; H.4 [**Information Systems Applications**]: Workflow management

Keywords

information extraction, procedural knowledge, workflow, workflow extraction

1. INTRODUCTION

People constantly learn from their experiences, but since their experiential scope is always limited, people ask about the experiences of friends, relatives, and colleagues in order to expand their knowledge or to solve their problems. Through the Web common people have found means to express individual experiences and observations of facts, events, and situations, building a huge body of "practical world knowledge" (human experiences). We argue that this kind of content we refer to as experience is possibly the most valuable asset in Internet communities.

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

Procedural knowledge (also called *how-to* knowledge) is the knowledge exercised in the performance of some task [17]. Similar to a plan, it describes how to do a certain thing or to achieve a certain goal through a sequence of steps. In forums numerous records of instances of such procedural experiential knowledge is found for example in question-answer pairs. Concrete example domains are 'do-it-yourself' instructions for home repair tasks or installation and usage guidelines of software. For example, a (non-professional) computer user may provide her experience on how to make business cards with a computer. The steps of the described procedure are how to select a template, how to insert a logo and fill in the text, and, finally, how to print and cut the business cards. Also cooking recipes as published in cooking forums can be considered as procedural experiential knowledge about how to prepare a certain dish, involving a sequence of preparation and cooking steps.

One formal representation of procedural experiential knowledge by workflows has been recently introduced to enable reuse of experiential content [15, 13, 16, 14, 2]. Workflow retrieval methods [15, 13, 16, 2] as well as automated adaptation methods [14] make use of such a representation to support Web users in problem-solving tasks based on experience. However, the manual modelling effort for experts to acquire workflows from Internet communities is considerable.

To overcome this problem, this paper proposes novel, automated methods for workflow extraction from semi-structured web content. The main focus of the paper is a formative investigation of two different methods for workflow extraction. Both methods employ information extraction techniques, i.e. techniques that "turn[...] the unstructured information embedded in texts into structured data" [10, p. 759]. The first approach, a *term-based workflow extraction*, is based on traditional information extraction methods [22] for tokenization, sentence-splitting, part-of-speech tagging, and semantic tagging by means of hand-crafted rules on terms. It has been implemented in Java integrating and configuring the GATE system [5]. The second approach, a *frame-based workflow extraction*, is on template filling [10, p. 761]. It has been implemented based on the SUNDANCE system [19], which activates frame structures with multiple slots using linguistic patterns. The information distilled by Sundance can be regarded more complex than the information distilled by GATE. However, the final workflows resulting from post-processing the output of both the particular term-based or the frame-based approach have the same structure. The

evaluation is carried out to compare the two approaches by getting expert feedback.

The remainder of the paper is organized as follows. Section 2 introduces the formal representation of procedural experience. The third section describes the two approaches and the extraction process. Section 4 presents the formative evaluation. It is followed by a discussion of the results. The paper concludes with a review of related work and some concluding remarks.

2. FORMAL REPRESENTATION OF PROCEDURAL EXPERIENCE

Traditionally, workflows are "the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules" [4]. We take a broader view of workflows as we do not restrict their use towards the automation of a business process. We use the term to describe any flow of activities (not necessarily involving several participants) in problem-solving.

Workflows consist of a set of *activities* (also called *tasks*) combined with *control flow structures* like sequences, parallel or alternative branches, and loops. Tasks and control-flow structures form the *control flow*. In addition, tasks exchange certain *products*, which can be physical matter (such as cooking ingredients or a physical business card) or information. Tasks, products and relationships between the two of them form the *data flow*. Furthermore, a task may require or consume subsidiary resources (e.g. cooking equipment or a software tool), and have constraints (e.g. a duration). On the following, we call such additional information on a task *task facets*. Procedural experiential knowledge as expressed by users in Internet communities usually does not provide completely all the details of such a workflow representation. The workflow is usually only partially described by text, but all the mentioned representational issues may occur.

Cooking recipes are a typical sample for procedural experience. Large amounts of recipes are provided by Web communities. [1] In the linguistic literature [12, p. 269] cooking recipes are regarded a special form of how-to instructions. The characteristics of the language used to describe recipes are the following. English cooking recipes have a great uniformity of directive structures, which are almost exclusively imperatives [24]. The order in that activity verbs occur in the text, corresponds with the execution order of cooking tasks [12, p. 269]. We observed these characteristics also in other domains for how to instructions. An additional benefit of choosing cooking as a sample domain is that there is an annual scientific competition called computer cooking contest [2]. This contest provides us with the opportunity to compare our results with those from other research groups and to learn from each other. Figure 1 depicts a sample workflow for cooking rocket pasta (see Listing 4). The graphical notation of the workflow is an extension of Business Process Modelling and Notation [9]. Tasks are represented by round cornered rectangles. Links between two tasks represent sequences of the control flow. Rectangles illustrate the products, which are consumed. Task facets are depicted by ellipses. Products and task facets are connected by an arrow to their corresponding task. The sample recipe starts with the activity `saute`, which has `onion` and `green pepper` as input and `in a large skillet` and `until tender` as facets. This is followed by further activities like adding more ingredients, boil water, etc.

[1] For instance, www.allrecipes.com provides 44,000 recipes

[2] www.computercookingcontest.net

3. EXTRACTION PROCESS FOR WORKFLOWS

Information extraction methods are employed to create the workflow representation described above. In general, the process of information extraction acquires knowledge by skimming a text and looking for entities and for relationships among those [21, p. 873]. *Entities* are typically noun phrases and comprise of one or multiple tokens [22]. A form of an entity is the named entity like names of persons, locations, or product names. Named entity recognition is the task of locating named entities in unstructured text. A *relationship* is defined as a relation between two entities [22]. An example for a relation would be the *"part of"* relation. In a limited domain, information extraction can be performed with a high accuracy of the results [21, p. 873]. In order to apply the information extraction steps for the topic of workflow extraction, the extraction goals have to be specified more in detail. The goals are common for both workflow extraction approaches to be compared. Entities to be extracted are tasks, products, and task facets. Relationships to be extracted are assignment of facets to tasks, input and output relationships between products and tasks as well as the order of tasks. Each workflow task is instantiated and filled with the corresponding relationships during the extraction process for the products, the facets, and the control flow structures.

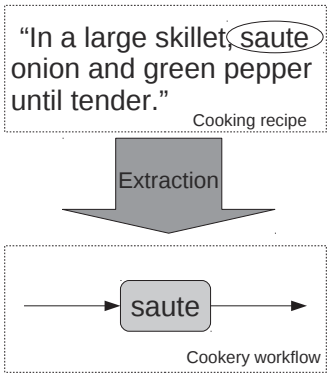

Figure 2: Extraction of an activity

Extraction of tasks: A workflow task corresponds to a single activity, which is mainly defined by a activity verb. Refer to Figure 2 for an example of extraction: the activity verb *saute* is extracted and forms a task in the resulting workflow part.

Extraction of products: The inputs of each task are described by the products of the workflow description 3. After the products *onion* and *green pepper* have been extracted, each of them is linked to the according task.

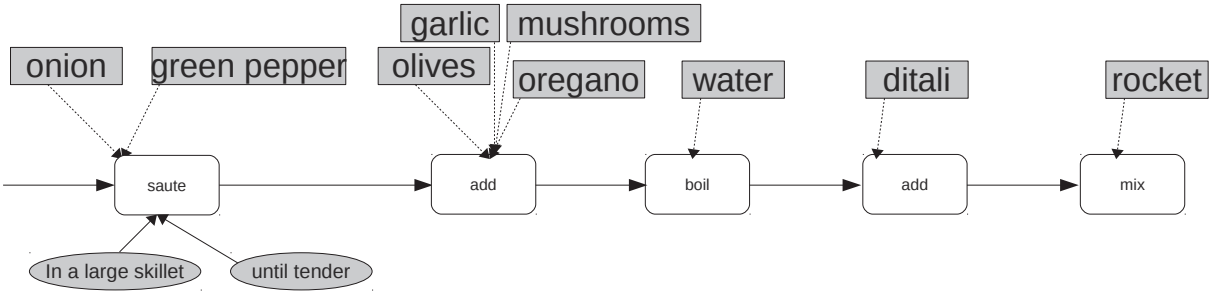

Figure 1: Sample workflow for cooking rocket pasta with vegetable.

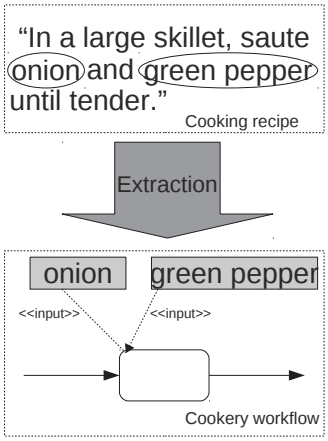

Figure 3: Extraction of products

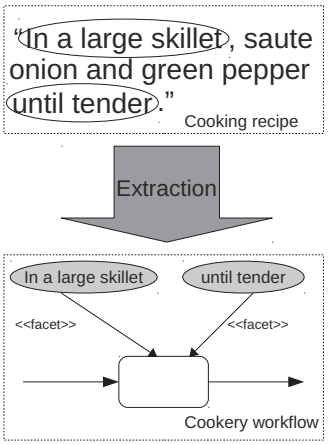

Figure 4: Extraction of task facets

Extraction of task facets Frequently, workflow tasks have additional information in the original description. In the visualized case (Figure 4), the extraction process gathered: *In a large skillet* and *until tender*. Subsequently, these task facets will be linked to the workflow task.

Extraction of the control flow: In addition, the workflow description gives a particular order of the activities. This order must be reflected in the control flow of the workflow. In the example of Figure 5, a sequential order for the two instructions is given, hence the matching tasks are ordered sequentially. For pragmatic reasons, we restrict our current implementation to extracting sequential orders only. In future work we are going to ponder on parallel or disjunctive control flows as well.

The two approaches described in the following sub-sections are based on a three-phase extraction process (see Figure 6).

In the first phase, the linguistic analysis is performed followed by the task extraction phase (including related products and facets) and, finally, the workflow build phase. Phase one begins with the tokenization, during which a stream of characters is separated into groups of characters, which are usually simple words. This is followed by the detection of the sentence borders called sentence-splitting. The first phase ends with the tagging of the part of speech. For this, the part of speech tagger assigns to each word a grammatical category coming from a fixed set of tags [22]. In the second phase, the activities, products, and task facets are extracted. To extract these elements, domain specific dictionaries of terms are used. One contains a set of products while the other one contains a set of activity verbs. In addition to the dictionaries, extraction rules are used. The frame-based approach uses generic rules while the term-based approach uses domain-specific rules (see below for details). In the last phase, we integrate the workflow tasks including the extracted elements (activities, products, task facets). For this, the single workflow tasks are placed at the correct position in the workflow.

3.1 Term-based approach

In the term-based approach we use the GATE framework. The first phase is entirely handled by GATE and therefore we will concentrate our description on the second phase, which does the main work. The input for this phase is the text annotated by GATE. First, a gazetteer is used to establish product and activity candidates. A gazetteer is basically

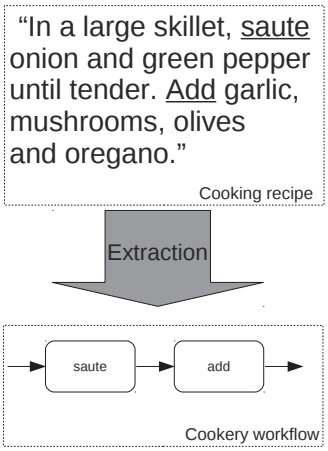

Figure 5: Extraction of task order

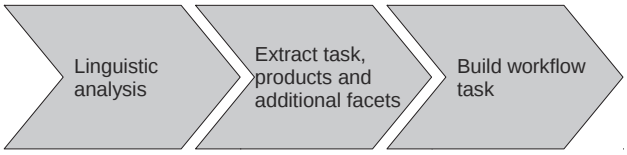

Figure 6: Extraction phases

a list with known entities and there categories. In our approach, a domain-specific dictionary of ingredient names and cookery verbs has been integrated with the general gazetteer that is included in GATE. In the future, we are going to replace the dictionary by a domain-specific ontology. GATE checks for every token if it is part of the gazetteer. In case a matching token is found, the token is tagged with the corresponding category. Then, a set of extraction rules is applied. These rules handle special cases, for instance, the verb "heat" that occurs also as noun. They are used to refine the part of speech tagging and the tagging that was produced by the gazetteer. Another set of rules is used to extract the facets of the task. The extraction rules are defined in the JAPE language. The JAPE (Java Pattern Annotations Engine) language was developed for the GATE framework. It allows the definition of transducers which operate with regular expressions at the level of text annotations. A transducer is a finite-state automaton which reads a stream of characters from an input-track. The characters are processed and written to an output-track. A JAPE transducer operates at the level of text annotations; the original text remains unchanged.

Listing 1: JAPE rule used by GATE

```
Rule: TimeSpec(
        ({Token.kind == number})
        ({Token.string == "minutes"})
): time-->
:time.Time = {rule = "Time"}
```

Listing 1 shows a sample JAPE rule. The rule has a left and right side divided by an arrow. The left hand side defines an annotation pattern. The structure of this pattern is similar to the structure of a regular expression. The right hand side of the rule defines the action that will be performed if a matching pattern was found. In our sample, the pattern matches if a stream of characters was found which contains two tokens. The property `kind` of the first token must be `number`. The property `string` of the second token must equal the value `minutes`. Then the two tokens are grouped in a temporary variable with the name `time`. Due to this grouping it is possible to add the new annotation (`Time`) to the group of tokens.

The JAPE rules are hand-crafted and domain-specific.

After we identified the activities, products and facets we instantiate the task template and fill the slots with the corresponding information. Finally, the task is positioned at the correct place of the workflow. Due to the language characteristics of cooking recipes (compare the discussion above), the order of activities in the workflow-description and the task-order in the extracted workflow are identical. Hence, this step can be performed quite straightforward. The execution of all phases is controlled by finite-state automata.

3.2 Frame-based approach

The frame-based approach makes use of the information-extraction-software SUNDANCE (Sentence UNDerstanding ANd Concept Extraction) developed by Ellen Riloff [19].

SUNDANCE performs the first phase of the extraction process described in Figure 6 in a quite similar manner than the term-based approach. The second phase, however, differs significantly from the term-based approach. SUNDANCE assigns the syntactic roles, creates and applies the so-called case frames as described below.

SUNDANCE has been inspired by the conceptual dependency theory published by Roger C. Schank [23]. The theory aims at illustrating how people think and process information. It is based on three kinds of concepts, the nominal, the action, and the modifier. A sentence or a thought can be represented by these three types of concepts. Nominals are those things that can be thought of without the need for relating them to other concepts. It usually used to represent things or people. The concept of an action is what a nominal is doing. These actions are usually represented by a verb. The last type of concept is the modifier, which specifies properties of a nominal or an action. A sentence is built of one or more concepts and a concept may depend on another one to get the whole meaning [23].

SUNDANCE allows to specify extraction patterns for the system called case frames [7]. These patterns are similar to the concepts described in the concept dependency theory. The SUNDANCE parser assigns syntactic roles (subject, direct object, and indirect object) to text snippets based on a heuristic. Then the SUNDANCE information extraction engine tries to fill a case frame as follows. Each case frame specifies a trigger phrase. If the trigger phrase is detected in a sentence the according case frame is activated. This means that the activation functions of the frame try to match the specified linguistic patterns with the syntactic roles. A slot specifies a syntactic role whose content is to be extracted from the text.

Listing 2: Case frame used by sundance

```
Name: ActVp_<dobj>:COOK
Anchor: VP(COOK)
Act_Fcns: active_verb_narrow_p(VP(COOK) )
Slot: dobj
```

Listing 2 shows an example for a case frame. The case frame begins with the name (*Name*) ActVp_<dobj>:COOK. The *Anchor* defines the trigger phrase. The sample case frame will be activated when a verb phrase with the word *cook* is found. In addition it is possible to define tests that must be true to activate the case frame, typically they look for types of syntactic construction. In our sample, the tests check whether the verb *cook* is head word of an active voice verb phrase. The last element of the case frame is the slot (*Slot*). It defines the syntactic role of the element that should be extracted. The result of applying the sample case frame of Listing 2 to the sentence :*"Cook tomato and empty the kitchen table."* is shown in Listing 3. The case frame was triggered by the word *COOK* and the slot was filled with the information *tomato*.

Listing 3: Sample extraction result

```
CaseFrame: ActVp_<dobj>:COOK
Trigger(s): (COOK)
DOBJ_Extraction = "tomato"
```

The case frames used in our approach have been constructed automatically by the AutoSlog system [18]. We restricted the set of case frames created from AutoSlog to those of special grammatical types. This was possible due to the uniform language style of procedural knowledge. The heuristics applied in the creation step are domain independent except for two domain-specific dictionaries for the products and for the task verbs.

In the context of workflow extraction we consider the trigger verb phrase as a task and the direct object as a product. In a similar way we extract task facets.

In the last phase we use the filled case frames to build the workflow in the same manner than in the term-based approach.

4. FORMATIVE EVALUATION

In this section we describe our formative experiments. The two extraction approaches have been compared to evaluate the following hypothesis: *The quality of extraction results gained from the frame-based approach is better than that from the term-based approach.*

We mean by quality the adequacy of the created workflow w.r.t. the procedural model that a domain expert has in mind after having read the according recipe. This is an empirical quality criterion, which can be measured only based on assessments from experts. In future work, a quality measure based on execution traces of the created workflow could be developed that computes the compliance of simulated execution traces of the extracted workflow with the workflow that has been modelled by the domain expert. For instance, the compliance measure introduced by Rinderle-Ma [20], which focuses on tasks and their products, could be extended for task facets. A simple computation of workflow isomorphisms is not appropriate as multiple adequate workflows might exist for the same text.

4.1 Experimental method

To prove our hypothesis, we employed both approaches for the same set of data and compared the results. We randomly chose a set of input data from the cooking domain.

Prototypical implementations of the term-based approach and the frame-based approach were employed to extract workflows from this data. To make the test comparable, the two prototypes used the same dictionary of cooking verbs and ingredients. The dictionary of ingredients was derived from the TAABLE food tree and the dictionary of verbs was created by a student in half a day.[3] After the extraction process, we asked an expert to assess the created workflows by scores from 0 (completely wrong) to 10 (perfect match). This empirical asessment provides evidence for or against our hypothesis. Traditional, quantitative methods for quality assessment based on measuring precision and recall, for instance, have not been applicable for several reasons. A corpus of extracted workflows that could serve as a baseline data for such quantitative measures is not available so far. Furthermore, it is not clear whether multiple solutions of workflows extracted from the same text exist and how they would be comparable. The quality assessment of the results by experts is feasible way feasible for this formative evaluation.

4.2 Setup and execution

We crawled about 25,000 different recipes from freely available sources on the Web in HTML-format. 40 recipes were randomly selected. These recipes were preprocessed and thereby transformed into a simple XML-file. Listing 4 depicts the XML-representation of the sample pasta recipe from which the workflow in Fig. 1 has been extracted. It includes an XML element `<ti>` for the recipe title, one for each ingredient (`<in>`) and an element `<prep>` for the preparation description. This preprocessing was done manually, though it is only a small step to perform it automatically by a crawling script.

Listing 4: Sample recipe in XML format

```
<recipe>
    <ti>Rocket Pasta with vegetable</ti>
    <in>Ditali</in>
    <in>rocket</in>t
    <in>Water</in>
    <in>herbs</in>
    <in>onion</in>
    <in>olives</in>
    <in>mushrooms</in>
    <in>garlic</in>
    <in>oregano</in>
    <in>green pepper</in>
    <prep>In a large skillet, saute onion
        and green pepper until tender.
        Add garlic mushrooms, olives
        and oregano. Boil water. Add ditali.
        Mix the cooked vegetable with the
        ditali and the rocket.</prep>
</recipe>
```

The dictionaries for both approaches have been populated by the same entries: 95 verb entries and 2781 entries for in-

[3] www.taaable.fr

gredients have been specified by the authors. Additionally, nearly 20 domain-specific extraction rules have been specified in JAPE for the term-based approach. The frame-based and the term-based approach have been performed for each of the 40 recipes. The results of both approaches have been assessed by the expert.

5. RESULTS AND DISCUSSION

Figure 7 shows the distribution of results obtained through the evaluation of the 40 extraction results by the human expert. The frame-based extraction approach outperforms the term-based approach in every recipe. Table 1 presents a summary of the assessment scores. The term-based approached achieved a mean score of 1.3 only while the frame-based approach achieved a much higher score of 6.0 on average. This shows clearly that there is a huge difference of the quality of the results. The main reason for the large difference was the handling of unknown tokens. The term-based approach was unable to extract tokens which were not part of the dictionary. The frame-based approach was able to extract unknown ingredients, yet it was very dependant on the verbs. In the case, that a verb was not in the dictionary it failed to extract the task. Considering that the number of verbs is a lot smaller than the number of different ingredients, this problem is minor.

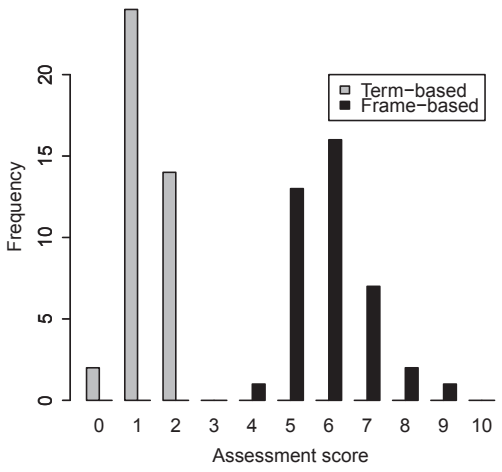

Figure 7: Distribution of assessment scores

Another difference between the approaches was the quantity of task facets which have been extracted. The term-based approach extracted 135 task facets while the frame-based approach extracted 417 task facets. This result does not allow to draw conclusions on the quality of the approaches. However, it indicates that the frame-based approach might have achieved a higher recall than the term-based approach.

For a transfer of the term-based approach to a new domain, it is necessary to populate a new dicionary and to code new extraction rules. For the transfer of the frame-based approach, only the new dictionary is required. The latter seems to be transferable to a new domain with lower effort than the first one.

Table 1: Summary of assessment scores

	Term-based	Frame-based
mean	1.3	6.0
worst	0	4
best	2	9

6. RELATED WORK

Chun Atluri and Adam [3] presented a knowledge-based approach for the automatic generation of workflows in the inter-agency domain. It is based on a composition algorithm that evaluates compositional rules against a user profile. These rules are given in several ontologies. Asiimwe [1] presented the SmartCAT-T tool which is able to extract knowledge-rich hierarchically structured cases from semi-structured textual reports. They extracted a set of concept, though they did not produce any temporal ordering of these concepts. SEAFACT [11] is a natural language interface to computer-generated animation system. It allows the user to specify a cooking task using natural language. The system then aims at animating the described task. The difference to our work is that SEAFACT concentrates on the extraction of one task instead of an entire workflow. Dufour-Lussier et al. [6] extract a tree representation of recipes from text. In contrast to our approach, the focus is on food components that are transformed by cooking activities to other food components. The TAAABLE ontology is applied to resolve references in the text by matching sets of food components, e.g. “blueberry” and “raspberry” with “fruits”. The TellMe [8] system allows the user to define procedures trough utterances in natural language that are interpreted by the system and transformed to formal workflow steps. In comparison to our system, the extraction process of the TellMe system is interactive; the user might get feedback from the system and can specify his input.

7. CONCLUSION

A method that supports reuse of procedural knowledge of semi-structured data was presented in this paper. We illustrated two different approaches to extract this knowledge. Additionally, we developed two prototypes implementing these approaches. The prototypes were assessed by a human expert. The assessment score showed that the frame-based approach outperforms the term-based approach with respect to the quality of the extracted workflows. Another aspect that was discussed was the work that must be done to switch to another domain in future work. We are planning to explore new domains like software installation guides or medical notes.

8. ACKNOWLEDGEMENTS

This work was partially funded by the German Research Foundation, project number BE 1373/3-1.

9. REFERENCES

[1] S. Asiimwe, S. Craw, B. Taylor, and N. Wiratunga. Case authoring: from textual reports to knowledge-rich cases. *Case-Based Reasoning Research and Development*, pages 179–193, 2007.

[2] R. Bergmann and Y. Gil. Retrieval of semantic workfows with knowledge intensive similarity measures. In *Case-Based Reasoning. Research and Development, 19th International Conference on Case-Based Reasoning, ICCBR 2011*, volume 6880 of *Lecture Notes in Computer Science*. Springer, 2011. The original publication is available at www.springerlink.com.

[3] S. Chun, V. Atluri, and N. Adam. Domain knowledge-based automatic workflow generation. In *Database and Expert Systems Applications*, pages 778–838, 2002.

[4] W. M. Coalition. Workflow management coalition glossary & terminology. 1999.

[5] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: an architecture for development of robust HLT applications. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 168–175, 2002.

[6] V. Dufour-Lussier, J. Lieber, E. Nauer, and Y. Toussaint. Text adaptation using formal concept analysis. *Case-Based Reasoning. Research and Development*, pages 96–110, 2010.

[7] C. Fillmore. The case for case reopened. *Syntax and semantics*, 8(1977):59–82, 1977.

[8] Y. Gil, V. Ratnakar, and C. Frtiz. Tellme: Learning procedures from tutorial instruction. In *Proceedings of the 15th international conference on Intelligent user interfaces*, page 227–236, 2011.

[9] O. M. Group. Business process modeling notation, v1.1. 2008.

[10] D. Jurafsky, J. Martin, A. Kehler, K. Vander Linden, and N. Ward. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*, volume 163. MIT Press, 2000.

[11] R. Karlin. SEAFACT: semantic analysis for animation of cooking tasks. *Technical Reports (CIS)*, page 734, 1988.

[12] G. Langer. *Textkohärenz und Textspezifität*. 1995.

[13] D. B. Leake and J. Kendall-Morwick. Towards Case-Based support for e-Science workflow generation by mining provenance. In *Advances in Case-Based Reasoning, 9th European Conference, ECCBR 2008, Trier, Germany, September 1-4, 2008. Proceedings*, pages 269–283, 2008.

[14] M. Minor, R. Bergmann, S. Görg, and K. Walter. Towards Case-Based adaptation of workflows. In S. Montani and I. Bichindaritz, editors, *Case-Based Reasoning. Research and Development, 18th International Conference on Case-Based Reasoning, ICCBR 2010, Alessandria, Italy, July 19-22, 2010. Proceedings*, LNAI 6176, pages 421–435. Springer, 2010. The original publication is available at www.springerlink.com.

[15] M. Minor, A. Tartakovski, and R. Bergmann. Representation and structure-based similarity assessment for agile workflows. In *Case-Based Reasoning Research and Development*, pages 224–238, 2007.

[16] S. Montani and G. Leonardi. A Case-Based approach to business process monitoring. *Artificial Intelligence in Theory and Practice III*, 331/2010:101–110, 2010.

[17] E. Plaza. On reusing other people's experiences. *Künstliche Intelligenz*, 9(1):18–23, 2009.

[18] E. Riloff. Automatically constructing a dictionary for information extraction tasks. In *Proceedings of the National Conference on Artificial Intelligence*, pages 811–811, 1993.

[19] E. Riloff and W. Phillips. An introduction to the sundance and autoslog systems. Technical report, Technical Report UUCS-04-015, School of Computing, University of Utah, 2004.

[20] S. Rinderle-Ma. Data flow correctness in adaptive workflow systems. In *Emisa Forum*, volume 29, pages 25–35, 2009.

[21] S. Russell and P. Norvig. *Artificial intelligence: a modern approach*. Prentice hall, 2010.

[22] S. Sarawagi. Information extraction. *Foundations and trends in databases*, 1(3):261–377, 2008.

[23] R. Schank. Conceptual dependence: A theory of natural language understanding. *Cognitive psychology*, 1972.

[24] R. Van den Broeck. Contrastive discourse analysis as a tool for the interpretation of shifts in translated texts. *Interlingual and Intercultural Communication: Discourse and Cognition in Translation and Second Language Acquisition Studies. Tübingen: Gunter Narr*, pages 37–47, 1986.