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

General Terms
Experimentation

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.

Procedural knowledge (also called how-to knowledge) is the knowledge exercised in the performance of some task [Plaza 2009]. 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 [Minor et al. 2007, Leake and Kendall-Morwick 2008, Montani and Leonardi 2010, Minor et al. 2010, Bergmann and Gil 2011]. Workflow retrieval methods [Minor et al. 2007, Leake and Kendall-Morwick 2008, Montani and Leonardi 2010, Bergmann and Gil 2011] as well as automated adaptation methods [Minor et al. 2010] make use of such a representation to support Web users in problem-solving tasks based on experience. However, the manual modeling 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” [Jurafsky et al. 2000, p. 759]. The first approach, a term-based workflow extraction, is based on traditional information extraction methods [Sarawagi 2008] 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 [Cunningham et al. 2002]. The second approach, a frame-based workflow extraction, is on template filling [Ju-
Cooking recipes are a typical sample for procedural experience. Large amounts of recipes are provided by Web communities. For instance, [Group 2008]. It has been implemented based on the SUNDANCE system [Riloff and Phillips 2004], 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 of 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” [Coalition 1999]. 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. In the linguistic literature [Langer 1995, 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 [Van den Broeck 1986]. The order in that activity verbs occur in the text, corresponds with the execution order of cooking tasks [Langer 1995, 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 [Group 2008]. 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.

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 [Russell and Norvig 2010, p. 873]. Entities are typically noun phrases and comprise of one or multiple tokens [Sarawagi 2008]. 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 [Sarawagi 2008]. 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 [Russell and Norvig 2010, 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.

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. After the products onion and green pepper have been extracted, each of them is linked to the according task.

**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 [Sarawagi 2008]. 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,
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 automatons.

### 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 [Riloff and Phillips 2004].

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 [Schank 1972]. 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 [Schank 1972].

SUNDANCE allows to specify extraction patterns for the system called case frames [Fillmore 1977]. 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_Extaction = "tomato"

The case frames used in our approach have been constructed automatically by the AutoSlog system [Riloff 1993]. 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 [Rinderle-Ma 2009], 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 TAAABLE food tree and the dictionary of verbs was created by a student in half a day. 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 assessment 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>
  <in>Water</in>
  <in>herbs</in>
  <in>onion</in>
  <in>olives</in>
  <in>mushrooms</in>
  <in>garlic</in>
  <in>oregano</in>
</recipe>

3www.taaable.fr
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 dictionary 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.

6. RELATED WORK

Chun Atluri and Adam [Chun et al. 2002] 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. Asiiinwe [Asiiinwe et al. 2007] presented the SmartCAT 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 [Karlin 1988] 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. [Dufour-Lussier et al. 2010] 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 [Gil et al. 2011] 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


