Solution Mining for Specific Contextualised Problems: Towards an Approach for Experience Mining

Christian Sauer[*]
School of Computing and Technology
University of West London
London W5 5RF, United Kingdom
christian.sauer@uwl.ac.uk

Thomas Roth-Berghofer
School of Computing and Technology
University of West London
London W5 5RF, United Kingdom
thomas.roth-berghofer@uwl.ac.uk

ABSTRACT

In this paper we describe the task of automated mining for solutions to highly specific problems. We do so under the premise of mapping the split view on context, introduced by Brézillon and Pomerol, onto three different levels of abstraction of a problem domain. This is done to integrate the notion of activity or focus and its influence on the context into the mining for a solution. We assume that a problem's context describes key characteristics to be decisive criteria in the mining process to mine successful solutions for it. We further detail on the process of a chain of sub problems and their foci adding up to a meta problem solution and how this can used to mine for such solutions. Through a guiding example we introduce basic steps of the solution mining process and common aspects we deem interesting to be analysed closer in upcoming research on solution mining. We further examine the possible integration of these newly established outlines for automatic solution mining for highly specific problems into a SEASALT^{exp}, a currently developed architecture for explanation-aware extraction and case-based processing of experiences from Internet communities. We thereby gained first insights in issues occurring while trying to integrate automatic solution mining.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

Keywords

Experience Web, Experience Mining, Solution Mining, Context

1. INTRODUCTION

The term Experience Web or Web of Experience was introduced in the context of research on knowledge extraction from the Internet in 2009. The basic idea is that user generated content available on the Internet is basically representing a huge repository of personal as well as technical experience provided by the users [18]. Experience available from the Web is already mined and used by such approaches as Opinion Mining, mining common sense knowledge into huge databases [9, 11], the construction of ontologies of general knowledge [9] and extracting knowledge for given application domains to model expert systems for the given domain [3, 10]. All of the mentioned approaches are well established and used with great effect. However, common problems these mining approaches face are the diversity of the knowledge representations used within the user generated content as well as identifying worthwhile sources of experience in the available data [15, 24]. These problems are moderated if the mining operation aims at general or common knowledge, reducing the need of finding domain specific sources or is supported by using highly organised knowledge sources deploying recent technologies and standards of the Semantic Web such as RDF, or Linked (Open) Data (see, e.g., [13, 25] for examples of knowledge extraction aided by Semantic Web technologies) reducing the effort of adapting the mining approach to the different forms of knowledge representation present in the mining sources.

A specific problem that not yet benefited from the above mentioned facilitations is the task of searching for a solution to a very specific problem. One such problem as for example could be a programmer seeking a solution to only one function she is trying to implement in a broader but specific project context. As of now a developer has to rely on general information sources like books or API documentations and piece together her solution to the problem she is facing. If she would try to benefit from the Experience Web in her context her task is supported but not yet provided by the use of search engines due to their limitation to fact finding for given problems such as with Wolfram Alpha[1]. It is still a manual task to find a worthwhile solution to such specific problems from the experience available on the Internet.

The intent of this paper is to provide insight in the process of mining solutions to highly specific problems in a defined context. This is to be accomplished by analysing a use case of a series of searches in a real world application development. The broader aim of providing such insight into the solution search is to provide a first approach to the definitions of the steps and sub-tasks involved in searching for a solution to a contextualised specific problem and to encourage the cross-fertilisation of ideas regarding the improvement of the identified tasks and thus formulate a new research approach to this problem and highlight a number of key technical challenges that must be solved. The specific aim of the analysis is determining the possibility to integrate an automated form of solution mining for contextualised specific

[*]Corresponding author.

[1]`http://www.wolframalpha.com`

problems into the SEASALT^{exp} architecture [20, 21] where this task is required to be performed in an automated way in the knowledge acquisition step of the architecture and thus needs the identified key technical challenges of the task resolved (see section 5).

The rest of the paper is structured as follows: Section 2 gives an overview of the Experience Web regarding the different forms of knowledge representations present there and an outline of the problems induced by these varying representations for knowledge mining. Section 2 also elaborates on related work in experience mining and its use so far. In section 3 we introduce the problem of mining solutions to contextualised specific problems and present the use case on which the analysis of the solution mining process is built. Section 4 then presents sub tasks and their respective challenges encountered during the process of solution mining. We further propose a first attempt to a process model describing the task of mining for solutions for contextualised specific problems. In section 5 we discuss possibilities to integrate automated forms of the sub-tasks identified in section 4 into the SEASALT^{exp} architecture. Section 6 summarises the paper and provides an outlook on next steps.

2. RELATED WORK

One of the fastest growing kinds of data with respect to volume on the Web is user generated content (UGC). This content is mostly in the form of semi-structured texts. UGC often contains artifacts of user experiences, expressed explicitly or implicitly [18]. Extracting this information, often describing experiences, is still a task not easily accomplished by a machine. This is mainly due to the fact that most of it is unsystematic and thereby hard to retrieve efficiently and thus cannot be mined and reused easily [4]. The Web 2.0 and its forms of user communities becoming more user-friendly results in more and more users participating in one of the many forms of web communities Web 2.0 offers [5]. Following the idea of the Experience Web [29] one has to ask how experience is at the current time mined and reused and thus how the Experience Web is being exploiting at the moment and which approaches are used for this. Figure 1 provides an overview of the contents forming the Experience Web where the Content is the basic layer, its usage forms the second layer of mainly opinions and the top layer represents actual experience present within the web.

Current exploits of the Experience Web are for example Opinion Mining. As Gordon and others [9, 17, 11], point out a currently well-established approach to harvest experience from the Experience Web aims at personal web logs stories and the extraction of experience from the stories present there. This approach of extraction from web-log stories is also widely employed in the area of Opinion Mining (see [16] for a survey of the approaches to opinion mining). Another area where experience mining is used is the extraction of common-sense knowledge from the UGC available from the Web 2.0 [9, 17, 23]. These approaches aid the field of common-sense reasoning for example in that way as to build up a large database of common-sense experience to allow reasoning automated upon this [9].

Aiming also to support common-sense reasoning but also for more specific tasks is work to extract situation ontologies from the available user experiences on the Experience Web. This approach also helps modelling context aware systems as it involves the storage of situational context in the given

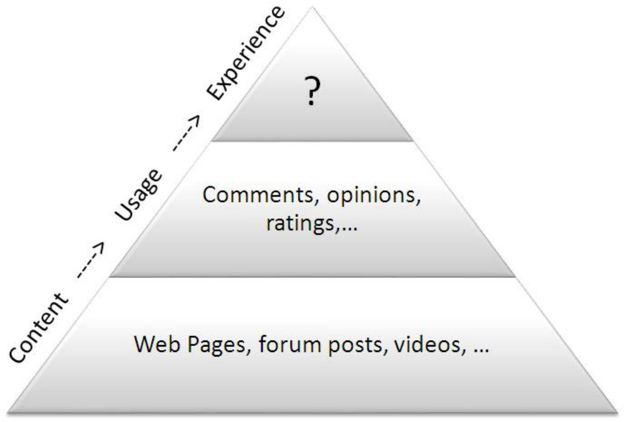
Content
Usage
Experience
?
Comments, opinions, ratings,...
Web Pages, forum posts, videos, ...

Figure 1: The Experience Web [18]

ontologies [12]. These approaches of extracting and using experience from the Experience Web are to be seen at the 'broad' or most abstract end of the spectrum of possible uses of experience mining or experience extraction. The 'fine' or most concrete end of the spectrum is given by extracting knowledge from the Experience Web to model domain specific knowledge for very specific domains given. This extraction of very specific knowledge for domains is helping to model these domains in, for example, knowledge based systems intended for decision support or planning tasks. However this approach does not try to mine for solutions to a specific problem directly, but aids to model the domains for the later automated generation or retrieval of a solution in a separate knowledge based system [26, 3, 10, 19, 14].

Despite all of the above mentioned approaches many of these experiences are still accessible only to certain approaches of knowledge extraction. Either the approaches are aiming at very broad, respectively highly abstract or common-sense knowledge or are aiming at the very specific ones, handling domain specific knowledge artefacts in a precisely defined domain. Both approaches, aiming for abstract or aiming for specific knowledge, benefit from the use of Semantic Web technologies. Using these technologies, such as RDF[2] or Open Data standards, within the web communities composing the Experience Web aids to provide a coherent and machine readable form of knowledge representation which in turns eases the effort necessary to extract the experiences present in these web communities. Thus a current research topic in the context of this paper is the analysis of knowledge representations used in web communities, the questions of how to formalise knowledge present and/or generated from the extraction process and how Semantic Web technologies and there strict use can aid the tasks of extracting experience from the Experience Web.

3. SCENARIO AND PROBLEM DESCRIPTION

The following problem is often encountered in real world programming assignments: The development of a Java based application deploying a rich GUI to generate and interact

[2]`http://www.w3.org/TR/rdf-concepts`

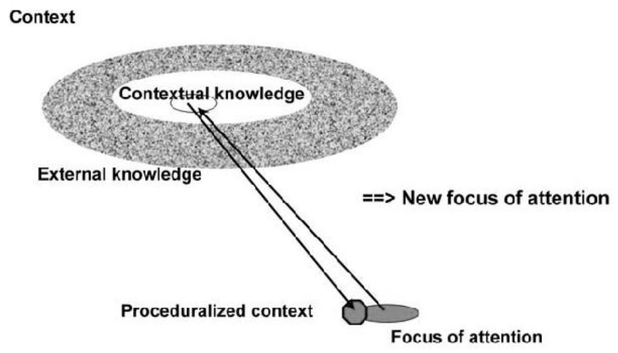

Figure 2: The three forms of context [6]

with a complex 3-dimensional scene, which was generated with the use of the Java3D library[3].

The basic assumption within the scenario is further that the developer is already experienced with the Java programming language but has only minor knowledge of the specific use of the functionalities the Java3D library provides. Said application is being implemented following the rapid prototyping approach, where the design of the overall application and its features are known at the start of the implementation. The actual implementation starts with a minimal runnable prototype implementing the most basic features, for example to display a single 3D cube in a window. From there on selected features of the overall application are seen as sub problems that are to be implemented in successive iterations, growing into a series of prototypes with an increasing number of features of the overall application.

So basically the main problem, the development of the application is broken down into a series of sub problems, each aiming at implementing a feature of the overall application. These sub problems are all bound to the context of achieving the goals given for the overall application as well as to the sub contexts of the neighbouring sub problem, i.e., previous prototypes and next prototype to be implemented.

Brézillon and Pomerol [7] define context as ‘what constrains problem solving without intervening in it explicitly.’ Further we follow their consideration of three kinds of context within the examined scenario: *external knowledge*, *contextual knowledge* and *proceduralised context*. Figure 2 provides an overview of Brézillon and Pomerol’s view of context. They assume that at a given focus of attention the context can be distinguished into a part of the context that is relevant to the given focus and one that is not. The non-relevant part of the context is called external knowledge while the relevant part is called contextual knowledge [6]. The relevant knowledge or contextual knowledge thus depends upon the current focus of attention, given in our scenario by the implementation of a sub problem. A part of it will always be invoked, assembled structured and situated according to the given focus, resulting into a proceduralised context which in our scenario would be the implementation of a sub problem [6]. After the proceduralised context is achieved the focus then can be shifted to a new focus of attention e.g. a new sub problem to be solved.

[3]`http://java3d.java.net`

So the three forms of context identified by Brézillon and Pomerol can be translated into the scenario described in this paper in the following way:

- The *external knowledge* is the general programming context and can be mined for example from within API documentations and/or general programming knowledge.
- The *contextual knowledge* is the context(s) relevant for the actual focus which is given by the implementation step at hand, aiming at the implementation of a single feature of the overall application.
- The *proceduralised context* can be seen as a ‘chunk of knowledge’ and is defined within our scenario as the very specific constrains defining the solution the developer searches for at his current task-focus.

Keep in mind that the proceduralised context can be seen as a chunk of, also contextual, knowledge [6], which can be transferred back to the surrounding contextual or even general knowledge after it once was defined/used [27, 6]. Even if ‘only’ the specific context-information for a given focus could be determined as a proceduralised context this information is still valuable for the mining for a solution to the focus, in our case sub problem, given by the feature implementation from the available experience on the Experience Web. Figure 3 maps the application of Brézillon and Pomerol’s view on context on the present scenario of implementing an application following the rapid prototyping approach.

The problem now is to mine for former established/generated proceduralised contexts, i.e., chunks of highly specific knowledge that can be used as solutions to a specific problem given by the implementation of a sub problem, i.e., a single feature in the meta context of the implementation of the overall application itself. By basing the mining process upon the context knowledge associated with the problem we can exploit its problem description capabilities to mine for more accurate solutions

Due to the highly specific nature of the sub problems at hand there is no option to look them up in any present ‘How to’ web portal, as they only provide experiences of a certain level of abstraction, to abstract to fit a highly specific problem. So far there are only two ways to derive highly specific solutions from the available experience. The first one is to rely on the available general knowledge e.g., API or Library documentations and involves the necessary abstraction and respective specialisation of the problem at hand to the abstraction level of the available documentation and back to the specific problem at hand. The second one is to search for a similar problem description of the problem at hand manually and to try and find a matching solution which requires minimal effort in adaptation to the problem at hand. The latter is a very common approach as the following quote illustrates[4]:

> ‘Last week I finished my first Android application. All through the development stage I had to google a lot for examples which some were really hard to find (even though you can find reference for everything in the SDK, for me, it’s easier to understand from a code sample).’

[4]`http://www.morethantechnical.com/2009/07/30/first-steps-in-android-programming`

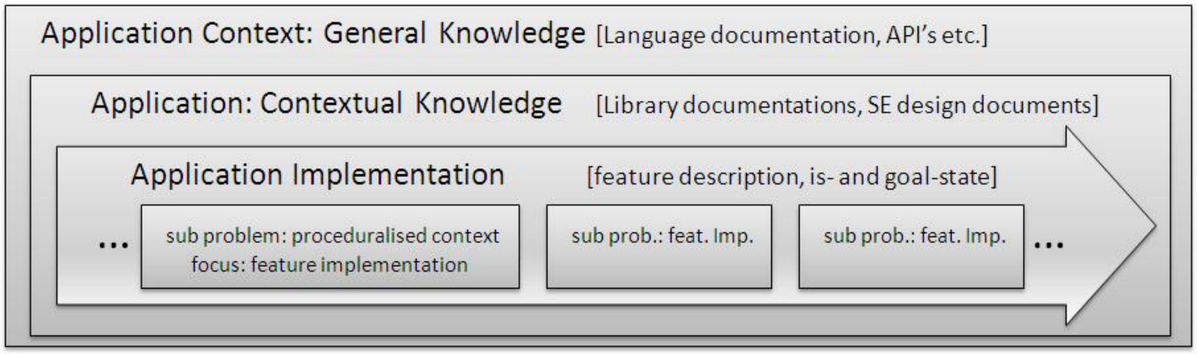
Application Context: General Knowledge [Language documentation, API's etc.]
Application: Contextual Knowledge [Library documentations, SE design documents]
Application Implementation [feature description, is- and goal-state]
...
sub problem: proceduralised context focus: feature implementation
sub prob.: feat. Imp.
sub prob.: feat. Imp.
...

Figure 3: The three forms of context mapped to the guiding example

This approach is typically followed while learning new programming languages and or beginning to develop in new SDKs or fields of applications where no prior specific contextual knowledge is available to the developer. Alternative approaches, besides the herein proposed mining for solutions from the Experience Web, are already being developed. For example, code repositories can be of help but are also bound to certain agreed upon levels of abstractions for the solutions they provide. Another approach to solution providing is using knowledge-based systems interacting with the development environment, as for example described in [22].

After defining the scenario and mapping to it the three kinds of context proposed by Brézillon and Pomerol we now deploy our experiences made during the actual development of the use case into a guiding example. The guiding example aims at highlighting the key technical as well as content related aspects of mining for a solution of a contextualised highly specific problem and give insight on which of these aspects need to be addressed to establish an automated mining process for said solutions.

4. PROCESS AND COMMON ASPECTS OF SOLUTION MINING

We base the process of mining for a solution to a contextualised highly specific problem on the notions regarding experience formulated by Schank and Abelson [28]. This notion assumes that all knowledge and thus all accounts of experience are story-based. As Schank and Abelson put it, 'where people's experiences in life are exactly as expected, there is no cognitive utility in storing these experiences away in memory' [28]. However, there is a cognitive utility to generate a story if a problem, particular a highly specific one was solved, especially if it was solved for the first time by the storyteller or after a long period of searching for/trying to generate a solution to the problem which is often the case for highly specific problems.

So for the inspection of the task of mining for solutions to highly specific problems we define such solutions as a problem solving experiences, composed of a part describing the preconditions P of the problem, the goal conditions G after the solution is applied, and the missing part M of the experience given by the working solution to the problem described within the experience. Thus the problems solution can be defined as the sequence of actions needed for the transfer of a situation from P to G.

The question for the transfer from P to G is formulated in dependency to the desired characteristics of P, G, and most importantly, the aspects of M, which are constituted by and describe the context the problem is situated in. So basically we are looking to fill the gap in the experience based upon the description of its known beginning and it's supposed but maybe not yet known or even unexpected ending (illustrated by Figure 4). By describing the context of the problem and thus defining the characteristics of the missing part we are looking for to fill in the gap in our experience. However we are aiming at mining an existing formalised experience of solving a problem and are not aiming at deep analysis of the problems nature and/or apply complex Natural Language Processing (NLP) to derive or determine a solution from the problem description itself.

As we have already established in section 3 we differentiate between three kinds of context, general knowledge, contextual knowledge and proceduralised context for goal oriented tasks. Thus we are aiming to mine a experience representing a proceduralised context formerly applied successfully. This experience is searched for by providing as much context information to describe the expected, needed, proceduralised context, i.e., the gap in our experience and by providing as much contextual information about the known parts of the experience, given by the preconditions P and goal conditions G as well as describing the sector of general knowledge we aim for to search within by providing as much information about the meta problem at hand in which the specific problem we are looking at is embedded, which is the overall Java application in our guiding example.

So we have to take into account three levels of contextualising the available data. These levels are high level which consists of general knowledge in the problems domain. So the high level is used to establish the context of the meta problem at hand.The second next finer level of contextualising is the structure of the meta problem we are trying to solve. the problem structure itself can be used for contextualising by describing the sequence of sub problems down to P and G of one single sub problem. The final and most fine granular level of contextualisiation is aiming at placing the gap in the experience we try to fill into context. This most defined level of contextualisiation mainly uses the pre- and postcondition of the gap in the subtask to determine and describe the context of the subproblem. This gap has to be contextualised by as much information regarding the expected characteristics of the chunk of knowledge, i.e., the

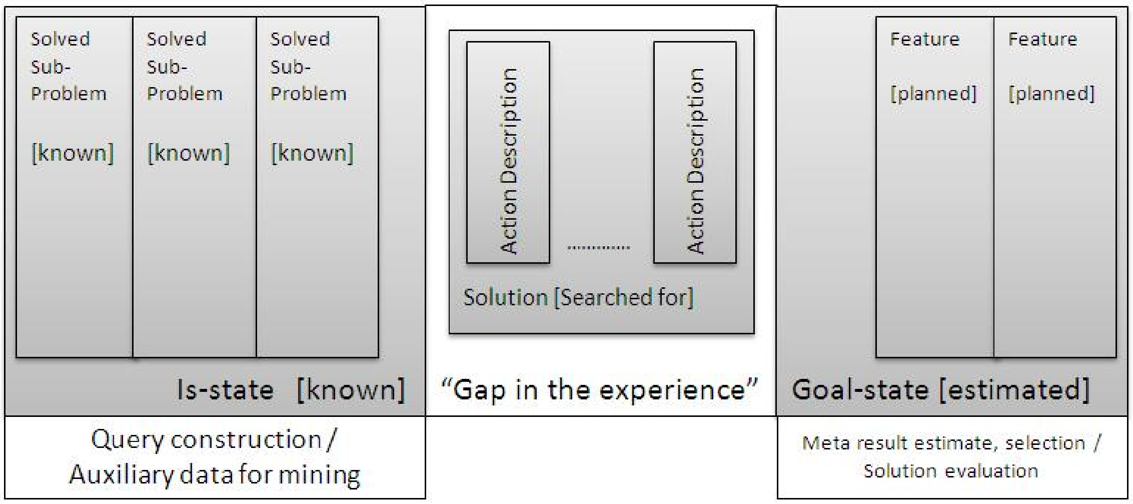
Solved Sub-Problem [known]
Solved Sub-Problem [known]
Solved Sub-Problem [known]
Is-state [known]
Action Description
Action Description
Solution [Searched for]
"Gap in the experience"
Feature [planned]
Feature [planned]
Goal-state [estimated]
Query construction / Auxiliary data for mining
Meta result estimate, selection / Solution evaluation

Figure 4: The missing experience chunk

experience or part of an experience we seek to mine to fill the gap thus complete the experience. If we complete a number of our highly specific sub problems one after the other we form a chain of them to finally get a chain that solves the given meta problem at hand. The sub problems are being embedded into a chain to form the solution to a meta problem. It can be split in four tasks: decomposing the problem and characterising its sub problems, result evaluation and classification of mining targets, mining solutions, and integrating solution in context.

4.1 Meta problem decomposition and characterising sub problems

The first task at hand is to decompose meta problem into specific sub problems. This task normally, at least for our guiding example, is covered by software engineering steps involved in the generation of the application, such as feature planning. We are not aiming at automatic problem decomposition. However it could be necessary if the solution mining is not aiming at software engineering or other beforehand decomposed problems or if only a broad problem is presented to the mining process which could benefit from a breakdown into sub problems with regards to providing a detailed and in depth solution.

Basic to problem decomposition is the problem cognition 'What is the problem?'. This problem cognition test aims at establishing in which domain of general knowledge we will have to mine and which are the top level (or most abstract) characteristics of the problem at hand being integrated in the contextualising of the top level description of the problem to be used in the mining process. After completing this first characterisation of the meta problem we have to decompose the problem into sub problems. This is often done by, e.g., temporal ordering in a series of sub problems or, as in our guiding example, by prioritising feature implementation. Further we define or determine for all sub problems the P and G, preconditions and goal conditions, as far as we know them already or as we desire them to be, with regard to the G's of our sub problem stories. Thus we should end up with a set of sub problems given now by a P and G descriptions forming the contextualising information to describe the contextual knowledge we will look at, the second most abstract level of knowledge involved. And each sub problem experience should now have a gap between its P and G waiting to be filled by a chunk of knowledge or proceduralised context we aim to mine for.

If the mining takes place in a learning context, where the next sub problem to reach a meta-problem solution might not yet be known a complete decomposition of the meta problem into sub problems cannot be accomplished from the start on. In such a case the approach will be 'growing naturally', in coordination with task 4 'Integrating a solution, proceduralised context, in the contextual- and external knowledge' and using this growing number of classified/controlled proceduralised contexts to identify further sub problems in a short range, in the manner of the 'next step in the right direction', i.e., the next solution to a sub problem useful to reach the goal of solving the meta problem at hand or just growing the knowledge graph in the problems domain.

For each sub problem identified we than have to establish the main characteristics of the sub problem again. This could for example be, in a text mining approach, to identify key concepts involved in the sub problem. In our guiding example this could be for an example a certain Java class and the exact use of a method the class provides to rotate geometry around a specific axis. Thus we would end up with the name of the class, the name of the method, the concepts of geometry, the verb rotate and the name of the axis as the key ingredients of the sub problem at hand thus forming the main components of any question or answer regarding the solution to this sub problem at hand.

For text mining one could further determine, based upon empirical data or statistical approaches, common grammar structures used often for the formulation of similar questions and for the answers to them. These grammar structures could then be incorporated into either a direct text mining on sources of experiences from the web or are incorporated in the query formation to be used on available web tools, such as search engines. At this point we are actually going to access the experience available on the Internet which progresses us to the second task.

4.2 Evaluation and selection of meta result, queueing and classifying mining targets

The second task consists of the identification and evaluation of the experience sources available to us and our means to access them. By means of access we refer to the device we are using to access the web, either mobile or desktop for example, and the tools that we are about to employ in the mining for solutions. Tools here are a variety of means to realise the raw knowledge acquisition phase of the mining process. This raw knowledge acquisition aims at focusing the actual mining for solutions on areas of the experience web likely to contain the correct solutions. These areas can be, for example, expert forums in which a certain topic or domain, in our guiding example programming for Java3D, is covered. Tools to identify such areas of interest can be a variety of search engines, crawlers or different means of accessing the sources of experience available on the Internet.

In our guiding example we relied on a number of expert forums dealing with Java3D coding. For any search engine based approach it is important to compose an effective query. This step is equivalent to designing effective auxiliary data to steer the behaviour of automated knowledge gathering tools such as crawlers. The data from the problem characterisation within the first task is incorporated at this time in to the query composition or gathering component programming. Besides the problem description aiding the query composition obviously when searching for a transfer from P to G, representing the solution to the problem, P and G themselves have to be well described, i.e., formalised to be incorporated in a query to be used as auxiliary data guiding the raw data acquisition tools e.g. crawlers.

The next step is processing the query with the tools chosen (do a search in the guiding example) or run the gathering tools a first time. The results, which we call meta results for now, are either a list of possible sources (websites) or the first volume of raw knowledge gathered by the tools, e.g. raw text from forums. These meta results are then subjected to coarse-grained analysis with regard to the density of key problem concepts mentions and/or matching to established common grammars used in questions/answers related to the problem at hand. If this analysis returns a considerable lack of such concepts and or mismatch of the grammars within the meta results, the query in use/the auxiliary data guiding the gathering-tools has to be refined. Contrariwise such a poor result of a first retrieval of knowledge sources could be a stern clue on how to reformulate the query/adjust the auxiliary data, by deriving common concepts or grammar structures from the meta results of the first retrieval. This is similar to the behaviour a human shows, when she discovers the correct way to formulate a question on a problem new to her and thus a particular complex step to automatise.

As the next task aims for the detailed evaluation of the concrete results represented by the now present meta results (such as website snippets forming a search result list or the volume of raw text from a crawl) it is desirable to schedule the meta results. This scheduling can be achieved by queueing the detailed analysis of each concrete result represented by its meta result reference with regard to the expected quality, usefulness of the concrete result derived from the characteristics of its meta result reference. This could be done for example by temporal ordering with regard to the already established temporal ordering of sub problems that is expected for the meta problem. Another approach is the ordering of the meta results by sheer quantification of mentions of the key concepts to the problem at hand. Further parameters that can be employed to refine the sequencing of the meta results are: estimate of usefulness, based upon concept-mentioning, grammar analysis, syntactical impression, ranging from coherent, well ordered to chaotic, the provenance of the meta result, ranging for example from sponsored meta results to well established expert forums, which of course demands the necessary meta data to establish the quality of the source of experience. A criterion not sequencing the meta results but classifying them and thus distribute them to tailored mining approaches, is the knowledge formalisation approach a concrete source of experience instruments. Knowing the target sources knowledge representation is important to employ the adequate techniques for the solution mining task at hand. Furthermore one needs to adapt said techniques further with regard to make them scalable.

4.3 Mining solutions from concrete results, evaluating, and using/adapting them

The third task is the deep evaluation of a determined artifact or concrete result from the shallow analysis of the meta results. In this task the semantics and not only its syntactic or statistical properties like grammar structure or term frequencies, of the contents of the concrete results are taken into account for a first time. The need for analysing, at least roughly, the semantic of the experience contained in the concrete source at hand is given by the need to determine if we are dealing with a chunk of knowledge describing a solution or are we, maybe, only looking at a paraphrased version of our own question, which of course would match most of the criteria listed so far to also identify a chunk of knowledge, or experience or bit of an experience, describing the solution we are searching for. One further could employ pattern matching on the information surrounding the concrete source considered to be a solution. This could be done by, for example, analysing the position of a text in a discussion thread, where a text at the very beginning is most likely a question for a solution and a text at the end is most likely a text describing or rating a solution. This descriptions of usefulness, correctness of solutions provided is also an aspect we can exploit while mining for solutions to our problem. We could employ various well established techniques used for opinion mining on candidates of bits of knowledge to evaluate the solutions or even identify them as solutions. This could be achieved by identifying positive, negative sentimental keywords, grammar structures and the like, think of identifying sentences like 'THX that did the trick :)' and mine on responses to the chunk of knowledge we are examining to determine if they were positive or negative. Further options for the evaluation of solutions are provided by community based annotations, such as ratings of solutions and highlighting problem descriptions as 'solved'.

If we found a solution text relevant enough to our specific problem at hand, which includes that we might also want to roughly determine the abstraction grade of the text in the concrete result at hand, we must decide if we either want to: Directly use the solution or adapt it which most often means a specialisation of the still too abstract, but best find, solution. If we use abstraction and/or specialisation to match the best found present solution to our specific problem it means we are moving the solution between our three contexts. Such movement can range from the proceduralised context, as the most specific, up the levels to find an abstracted version of the solution or problem, and down again to specialise the solution down to our specific sub problem at hand. This possibility of the necessity of moving between different levels of abstraction also raises the question for the knowledge formalisation most likely to being minimal and most exact thus not needing to being abstracted and/or specialised to adapt the solution mined.

While much effort is invested in the mining for the best possible and most specific solution to our specific sub problem, it also can't hurt to memorise a number of known failed solutions to the problem. These negative examples can help refine the query and/or auxiliary data guiding the knowledge acquisition tools, processes by providing filter criteria.

4.4 Integrating solution, a proceduralised context, in contextual and external knowledge

The last task in mining and using a solution for a specific problem is the integration of the solution in the contextual knowledge or even the external knowledge. This means that a mined, applied and possibly adapted solution, made up of a chunk of knowledge, i.e., a proceduralised context, has to be integrated in a surrounding context to be easily retrieved, reused if the need to solve a problem similar to the just solved problem should occur, which represents a form of learning in the problem domain at hand. Therefor we need to classify the solutions or proceduralised contexts to annotate them with the problem they are applicable to.

Depending on the evaluation of the solution's application we can either integrate the solution as a valid one in our knowledge in the way described above and move on to mine for the solution to the next sub problem in the meta problem queue or we need to mine for a new solution. For each successfully applied solution we have to document its provenance (in our guiding example for example place a bookmark) to enable fast access for mining similar solutions problems and thus learning to how to effectively access the experience sources available. This can be fostered by grouping the identified sources of experiences, providing successful solutions to sub problems, into groups of similar sub problems or into groups describing common forms of meta problems. This structuring of experience resources could, for example, follow a graph based approach to provide a growing graph of proceduralised contexts, representing successful problem solutions, with graph regions themselves describing certain meta problem chains, representing contextual knowledge for high level problems and the whole graph itself describing the general knowledge in the specific family of meta problems solutions were mined for. Another possible way to rearrange the mined solutions is to build up an ontology describing the possible solutions in a given domain of meta problems. This approach would be most likely very similar to the approach to derive large scale situational ontologies from how-to instructions from the web, described by Jung [12].

5. CHALLENGES OF AUTOMATING SOLUTION MINING WITHIN SEASALT^{EXP}

An architecture aiming at integrating experience extracted from the web into its own knowledge repositories is the SEASALT^{exp} architecture [21]. SEASALT^{exp} provides an application independent architecture that features knowledge acquisition from a web community, knowledge modularisation, and agent-based explanation capabilities and knowledge maintenance. The individual components are grouped into layers according to their function in knowledge management. Knowledge is drawn from different sources, formalised by a knowledge engineer with the help of an apprentice agent and stored as problem-solving and explanation knowledge. Knowledge provision is realised using the knowledge line approach [2]. The knowledge line's basic idea is to modularise knowledge analogous to modularising software in the product line approach within software engineering [30]. The answer procured by the knowledge provision layer then is further individualised and enhanced with explanatory information according to the needs of the enquirer.

Integrating solution mining into this architecture seems to us a valuable addition and a good background to identify some of the challenges still to be overcome to achieve automated solution mining. The module where automated solution mining would be realised in is the *knowledge formalisation*. This interface can already make use of knowledge automatically extracted from experience present in web communities and transformed into taxonomies [26].

The architecture also is suitable for automatised problem mining as suposed in this paper. Due to the architextures approach to knowledge formalisation, using the Knowledge factory approach and the use of distributed case bases, it is possible to disassemble complex problems into simpler ones. This could be used to store sequences of sub problems describing a workflow to solve a more complex meta problem. Dealing with sub parts of a complex problem as described in the guiding example, allows for the mining for the solutions of such sub problems and as the SEASALTexp architexture allows, due to its Knowledge factory technique, for such a problem decomposition which in turn allows to be used for a problem mining approach as described in this work. It further already allows for crawling Internet forums as a way of acquiring initial, or meta, results to mine concrete solutions from.

The first challenge to be dealt with is the actual design of the auxiliary data to help provide the meta results. The next task is to determine schedule for the concrete results to be analysed. This requires the already mentioned modelling of common grammars found in problem descriptions and their respective question- and solution-texts for example. It further implies the question of how to establish initial Gazetteers of problem specific keywords and how to evolve/optimise them. Another problem is the question of the knowledge formalisation approach. Currently the architecture is centred on analysing unstructured text. It is an interesting question on how it might benefit from better usage of possibilities provided by Semantic Web techniques. One such aspect, already supported by the architecture is the use of Linked Open Data. What is not currently supported by the architecture is the use of meta tools, such as Web search engines.

What is supported and in fact is one of the main features of the architecture is the use of a distributed knowledge base following the approach of the experience factory [1]. Adding a solution knowledge base and diversifying it to represent any given number of problem sub spaces or sub problems should be easily doable within the architecture. However the layout of this distributed problem and/or solution knowledge base must be designed before the actual KBS is designed

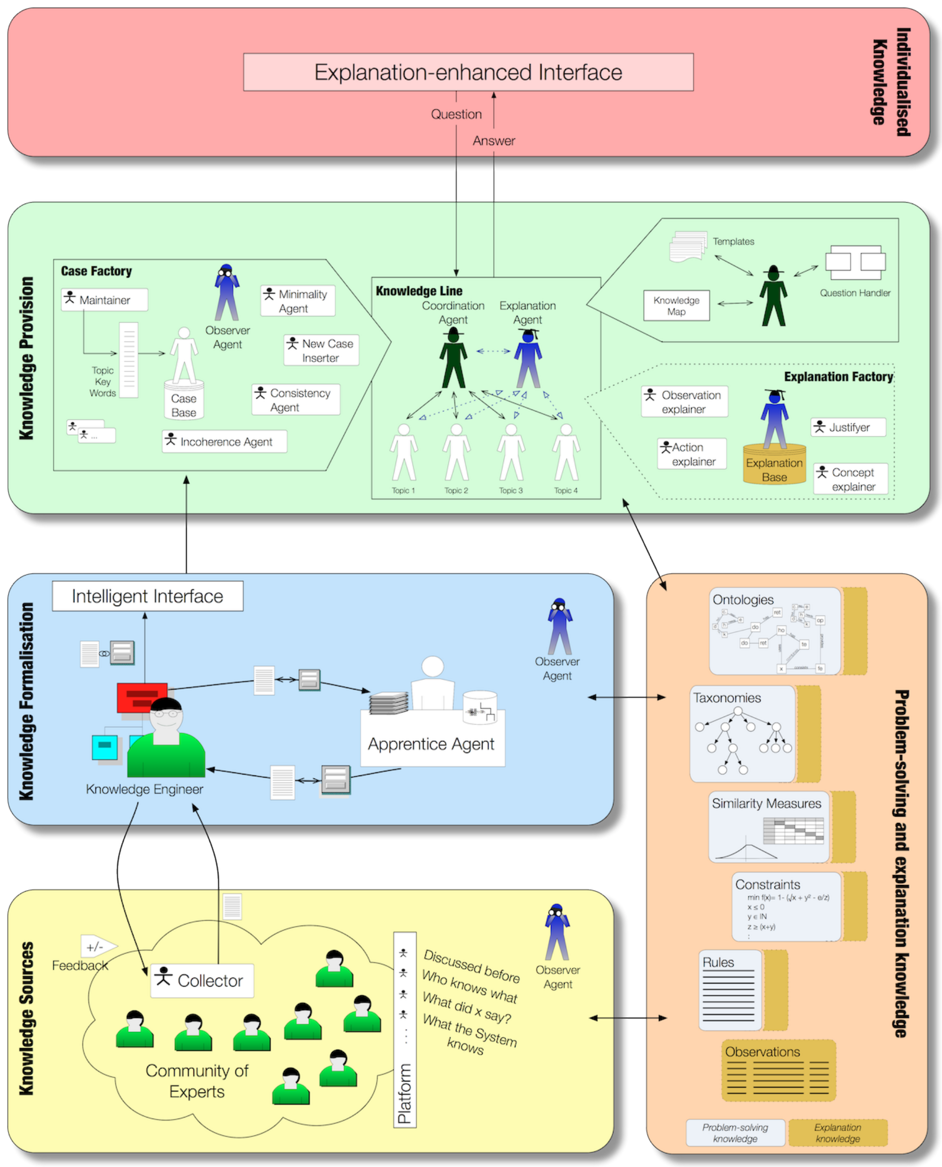

Figure 5: Explanation-aware knowledge acquisition from web communities with SEASALT^{exp} [21]

and thus it isn't possible yet within the architecture to create dynamic 'on the fly' decompositions of meta problems into chains of sub problems as of their respective knowledge bases would need to be designed prior such an problem composition could be done.

6. SUMMARY AND OUTLOOK

Within this paper we have described the task of automated mining for solutions to highly specific problems. We did so under the premise of mapping the split view on context introduced by Brézillon and Pomerol onto three different levels of abstraction of the problem domain. By this we integrate the notion of activity or focus and its influence on the context into the mining for a solution. We loosely followed a guiding example consisting of the development of a Java application to demonstrate a real application of solution hunting and problem decomposition. Based upon this example we introduced basic steps of the solution mining process and common aspects we deem interesting to be analysed closer in upcoming research on solution mining. The main aspects of the process of automated mining for solutions to highly specific problems were identified as meta problem decomposition and the characterising of sub problems chaining up to a meta problem. Further we identified the need to evaluate meta results and selecting, queueing and classifying them in an intelligent way to form an effort optimising schedule for mining of the concrete results represented by said meta results. We than established example approaches of mining from the concrete results and identified the need to again evaluate the mined solutions before use and decide if and how to use the solutions with regard to possible adaption mainly with regard to needed abstraction, specialisation of the mined problem solution to fit the abstraction level of the sub problem at hand. Finally we identified the need to integrate mined successful solutions as well as mined unsuccessful solutions to a certain degree, into an overall knowledge representation describing the problem space we are mining solutions for.

We proposed to use a graph structure representing the problem space in whole, with regions of the graph representing common meta problem families in the problem domain and single nodes of the graph representing on successful solution to one highly specific sub problem, representing on successful chunk of knowledge, bit of experience or proceduralised context. We then took the newly established outlines for automatic solution mining for highly specific problems and considered their integration in an currently developed architecture for the design of knowledge-based systems, namely SEASALT^{exp}. We thereby were able to give a first insight on possible problems occurring while trying to integrate automatic solution mining.

For the near future we are aiming at refining the main tasks described in this paper. We will particularly focus our efforts on a detailed analysis of the requirements existent for the syntactic analysis of meta results to establish their likelihood to hold adequate solutions to mine from. Therefor we are aiming to build a customised NLP application based on the well established GATE framework [8]. This application will then be put to use in a mining process for meta results from a manually established source of experience. We will than further analyse which characteristics of the meta results are most useful to act as indicators for the presence of solutions in the source the meta result stands for.

7. REFERENCES

[1] K.-D. Althoff, J. Mänz, and M. Nick. Integrating case-based reasoning and experience factory: Case studies and implications. In U. Furbach, editor, *Proceedings of the 28th German Conference on Artificial Intelligence Workshop on Knowledge Engineering and Software Engineering*, pages 1–12, Koblenz, Germany, 11. - 14. September 2005. Springer, LNAI 3698.

[2] K. Bach, M. Reichle, A. Reichle-Schmehl, and K.-D. Althoff. Implementing a coordination agent for modularised case bases. In M. Petridis and N. Wiratunga, editors, *Proceedings of the 13th UK Workshop on Case-Based Reasoning (at the Artificial Intelligence Conference, AI 2008)*, pages 1–12, dec 2008.

[3] K. Bach, C. S. Sauer, and K.-D. Althoff. Deriving case base vocabulary from web community data. In C. Marling, editor, *ICCBR-2010 Workshop Proceedings: Workshop on Reasoning From Experiences On The Web*, pages 111–120, 2010.

[4] R. Bergmann. *Experience Management: Foundations, Development Methodology, and Internet-Based Applications*, volume 2432 of *LNCS*. Springer, 2002.

[5] D. M. Boyd and N. B. Ellison. Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230, October 2007.

[6] P. Brézillon. A context approach of social networks. In *proceedings of the Workshop on Modeling and Retrieval of Context, Ulm, Germany*, volume 114, 2004.

[7] R. Brezillon and J. Pomerol. Contextual knowledge sharing and cooperation in intelligent assistant systems. *Travail Humain*, 62:223–246, 1999.

[8] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. Gate: A framework and graphical development environment for robust nlp tools and applications. In *Proc. of the 40th Anniv. Meeting of the Assoc. for Comp. Linguistics (ACL'02)*, 2002.

[9] A. Gordon. Mining commonsense knowledge from personal stories in internet weblogs. *Automated Knowledge Base Construction*, page 8, 2010.

[10] N. Ihle, A. Hanft, and K.-D. Althoff. Extraction of adaptation knowledge from internet communities. In S. J. Delany, editor, *ICCBR 2009 Workshop Proc., Workshop Reasoning from Experiences on the Web*, pages 269–278, July 2009.

[11] K. Inui, S. Abe, K. Hara, H. Morita, C. Sao, M. Eguchi, A. Sumida, K. Murakami, and S. Matsuyoshi. Experience mining: Building a large-scale database of personal experiences and opinions from web documents. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on*, volume 1, pages 314–321. IEEE, 2008.

[12] Y. Jung, J. Ryu, K. Kim, and S. Myaeng. Automatic construction of a large-scale situation ontology by mining how-to instructions from the web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 8(2):110–124, 2010.

[13] P. Mika. Flink: Semantic web technology for the extraction and analysis of social networks. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(2-3):211–223, 2005.

[14] P. Milne, N. Wiratunga, R. Lothian, and D. Song. Reuse of search experience for resource transformation. In *Workshop Proceedings of the 8th International Conference on Case-Based Reasoning*, pages 45–54, 2009.

[15] I. O'Murchu, J. G. Breslin, and S. Decker. Online social and business networking communities. In Y. Ding, D. Fensel, R. Lara, H. Lausen, M. Stollberg, and S.-K. Han, editors, *ECAI Workshop on Application of Semantic Web Technologies to Web Communities*, volume 107 of *CEUR Workshop Proceedings*, 2004.

[16] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[17] K. Park, Y. Jeong, and S. Myaeng. Detecting experiences from weblogs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1464–1472. Association for Computational Linguistics, 2010.

[18] E. Plaza and C. Baccigalupo. Principle and praxis in the experience web: A case study in social music. In S. J. Delany, editor, *ICCBR 2009 Workshop Proc., Workshop Reasoning from Experiences on the Web*, pages 55–63, July 2009.

[19] E. Plaza and C. Baccigalupo. Principle and praxis in the experience web: A case study in social music. In *The Eighth International Conference on Case-Based Reasoning*, pages 55–63, 2009.

[20] M. Reichle, K. Bach, and K.-D. Althoff. The seasalt architecture and its realization within the docquery project. In B. Mertsching, editor, *Proc. of the 32nd Conference on Artificial Intelligence (KI-2009)*, LNCS, pages 556–563, Sept. 2009.

[21] T. Roth-Berghofer, K.-D. Althoff, C. Sauer, K. Bach, and R. Newo. SEASALTexp — an explanation-aware architecture for extracting and case-based processing of experiences from internet communities. In K. Bach and C. Sauer, editors, *Workshop Proceedings FGWM-2011 Workshop on Knowledge and Experience Management*. University of Magdeburg, 2011.

[22] T. Roth-Berghofer and D. Bahls. Code tagging and similarity-based retrieval with mycbr. *Research and Development in Intelligent Systems XXV*, pages 19–32, 2009.

[23] J. Ryu, Y. Jung, K. Kim, and S. Myaeng. Automatic extraction of human activity knowledge from method-describing web articles. *Automated Knowledge Base Construction*, page 16, 2010.

[24] C. S. Sauer. Analyse von webcommunities und extraktion von wissen aus communitydaten für case-based reasoning systeme. Master's thesis, Institute of Computer Science, University of Hildesheim, 2010.

[25] C. S. Sauer, K. Bach, and K.-D. Althoff. Integration of linked open data in case-based reasoning systems. In M. Atzmüller, D. Benz, A. Hotho, and G. Stumme, editors, *Proceedings of LWA2010 - Workshop-Woche: Lernen, Wissen & Adaptivitaet*, Kassel, Germany, October 2010.

[26] C. S. Sauer and T. Roth-Berghofer. Web community knowledge extraction for myCBR 3. In M. Bramer, M. Petridis, and L. Nolle, editors, *Research and Development in Intelligent Systems XXVIII. Proceedings of AI-2011. The Thirty-first SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*. Springer, 2011.

[27] R. C. Schank. *Dynamic memory: A theory of reminding and learning in computers and people*, volume 240. Cambridge University Press Cambridge, 1982.

[28] R. C. Schank, R. Abelson, and R. S. Wyer. *Knowledge and memory: The real story*. Erlbaum, 1995.

[29] B. Smyth, P.-A. Champin, P. Briggs, and M. Coyle. The case-based experience web. In S. J. Delany, editor, *ICCBR 2009 Workshop Proc., Workshop Reasoning from Experiences on the Web*, pages 74–82, July 2009.

[30] F. van der Linden, K. Schmid, and E. Rommes. *Software Product Lines in Action - The Best Industrial Practice in Product Line Engineering*. Springer, Berlin, Heidelberg, Paris, 2007.