Agent Based Maintenance for Modularised Case Bases in Collaborative Multi-Expert Systems

Klaus-Dieter Althoff, Meike Reichle, Kerstin Bach, Alexandre Hanft, Régis Newo

Intelligent Information Systems Lab
University of Hildesheim, Germany
althoff|reichle|bach|hanft|newo@iis.uni-hildesheim.de

Abstract. This paper features SEASALT, an intelligent information system architecture that follows the example of collaborating human experts. Modularised knowledge is provided in a number of case bases that offer their topic expertise which is used to combine knowledge for individual queries. We describe how collaborative multi-expert systems can be instantiated using CBR technology to cover the experts' roles and agent technology in order to enable their collaboration. We focus on the maintenance of cases in such multi-case-base systems regarding inter-case base dependencies. We evaluate our approach based on the real-life application of travel medicine and show how the requirements of an information system on travel medicine can be fulfilled using our hybrid, agent based CBR system architecture.

Keywords: CoMES; Collaborative Multi-Expert Systems; SEASALT; hybrid, agent based CBR system; case base maintenance; distributed case bases; inter-case base dependency; travel medicine

1 Introduction

Nowadays, in our mostly goal-oriented need for information, we rely on modern information systems, which we expect to be good enough to provide us (quickly) with all the information we need. One of the main requirements of those systems is that they should be sufficiently smart to support their users requiring as little effort as possible. Hence they should offer knowledge-intensive services that are able to adapt and expand their knowledge. In this paper we present the maintenance aspects of Collaborative Multi-Expert Systems (CoMES), an approach for developing intelligent information systems for delivering knowledge-intensive services and supporting knowledge workers. Here we especially focus on experience (case-specific knowledge, cases) which is known to be the success-critical knowledge in these kinds of applications [1].

As its name already suggests, CoMES is related to the expert system[1] approach [2]. While prominent problems of the latter were that they mostly ignored the task of

[1] In this article we do not differentiate between the notions of *expert system* and *knowledge-based system*.

embedding expert systems in real business processes and lacked a good solution for overcoming the so-called knowledge acquisition bottleneck, CoMES tries to overcome these problems by integrating various approaches from artificial intelligence and software engineering. For example, the software product-line approach [3] is applied to the knowledge within an expert system with the goal of decomposing the knowledge into different modules (Knowledge Line). We use a multi-agent system [4] to model and realise this. As case-specific knowledge is a prominent part of the knowledge to be captured and reused, these software agents are mostly implemented as case-based reasoning (CBR) systems. By this we also want to enable these agents to learn from their own and others' experience. Here we also benefit from the already achieved integration of CBR in Experience Factory, an approach for (organisationally) learning from experience by means of human-enacted (business) processes [5]. While in this integration CBR helped to model these processes more precisely, the Experience Factory provides business processes that are suited for evaluating and/or maintaining case bases. We now try to enact these processes using software agents; an approach which we call Case Factory [6].

As a first evaluation of our approach we present the real-life application docQuery [7], which we currently develop together with mediScon worldwide. It covers the travel medicine domain and is currently being realised using the SEASALT (Sharing Experience using an Agent-based System Architecture LayouT) architecture [8], our first instantiation of CoMES. We will implement the docQuery application which will provide travellers with travel medicine information tailored to suit their journey. Queries to the system contain data like travel period, destination, or age of the traveller. We will be able, by the means of docQuery, to create a community to exchange knowledge and offer a multi-expert system on travel medicine. Integrating a community challenges our system to organise the knowledge using both, software and human agents.

In chapter 2 we will focus on the maintenance of multiple case bases for realising Knowledge Lines. We then evaluate our approach using the docQuery application on travel medical data in chapter 3. Finally some conclusions and an outlook on potential future work are presented in chapter 4.

2 Using Multiple Case Bases to Realise Knowledge Lines

The provision and reuse of knowledge in a knowledge-based system (Knowledge Lines) is realised by splitting rather complex knowledge in smaller, reusable units that are represented by different intelligent Topic Agents, which are realised as individual CBR systems. Each of these agents' case bases is maintained by a case factory, an organisational unit that emulates the experience factory approach [5] from software engineering.

The first practical instantiation of the CoMES approach is the SEASALT architecture that picks up on Knowledge Lines and case factories. Within the SEASALT architecture the information system's knowledge domain – in the case of docQuery travel medicine – is modularised into different topics or sub-domains. Each Topic Agent is equipped with information on one particular aspect of travel medicine (for instance geographic information or travel related diseases) and a case factory taking care of its case base.

The information stored in the respective case bases is acquired from an online discussion forum for a community of domain experts that is equipped with collector agents. These agents collect experts' contributions that are relevant for the collector's respective Topic Agent and pass them to a knowledge engineer for formalisation, so they can be handled in a structured CBR system. The formalisation is done by one or more human domain experts in the beginning but shall be at least partially automated as more training data becomes available. While this human knowledge engineer will increase the system's overall expenses (be it financial expenses or volunteer work), we consider the direct connection to the domain experts community to be valuable enough with regard to the system's accuracy and timeliness as to warrant these expenses. A query to the overlying information system will be handled by a Coordination Agent that subsequently queries the different Topic Agents according to a request strategy, which indicates which Topic Agents' answers give constraints for another Topic Agent's query, and a Knowledge Map that indicates where/how the individual Topic Agents can be contacted. The Coordination Agent will then combine the information it received from the individual Topic Agents into an answer, using ready-prepared templates, that is finally passed back to the user. This approach is comparable to the negotiated case retrieval approach presented in [9], but relies on a more strict partition with regard to the case bases' content, which allows a fixed definition of combination rules (the request strategy, for example "Find out the region first, use the region to find relevant diseases, ask for chronic illnesses, use diseases and preconditions to retrieve medicaments") and does not require negotiation between the individual Topic Agents.

The approach of using several case bases within a case-based reasoning system (multi-case-base reasoning, MCBR) exists within the CBR community for several years already [10]. In the beginning case bases were partitioned mostly for performance reasons, in order to allow a faster case retrieval within systems with a very large case base. In other applications cases can not be centralised within one single case base for privacy or maintenance reasons, for instance in a situation where different parties each run their own individual case base and do not want to share their entire content or maintenance with other involved parties. Also, Leake und Sooriamurthi [11] present experimental evidence that, given data from different sources, treating these as individual case bases, being coordinated using a dispatcher, which selects appropriate knowledge sources/case bases and also combines returned cases, yields to better results than simply merging all available data into one centralised case base.

The modularisation in sub-domains instead of simply partitioning a case base into several smaller ones with the same domain/case format has several different advantages. Firstly the individual case bases are easier to maintain with regard to the correctness of their content, since they represent a more simple knowledge domain. Also breaking up the rather complex domain of travel medical advisories into more simple sub-domains that are then recombined on demand, gives the whole information system more flexibility. Provided the appropriate combination, the contents of the individual case bases can also be combined into cases that have not yet been presented to the system, as long as they adhere to the respective combination rules.

Further, not all knowledge domains that are included in an information system on travel medicine (geographic information, diseases, medicines and their active pharmaceutical ingredients, travel activities) require the same type of maintenance and are subject to the same amount of change over time. While it is for instance no problem to keep a rather simple domain such as countries and regions as minimal and consistent as possible, the domain of travel related diseases is better served by including as many cases as possible, even if some of them are very similar. By splitting the knowledge domain into these smaller sub-domains, they can all be maintained in a way that is best suited for the respective sub-domain.
In our approach the employment of multiple case bases is motivated by the individual case bases' different maintenance needs. We intend to implement the case bases' maintenance following the case factory approach, which means that it will be carried out by several agents, each performing a single and rather simple task. These different tasks will each ensure one individual case base property, as described for instance in [12]. Analogous to the case base properties presented there, we will employ individual agents to ensure the case bases' consistency, uniqueness/minimality, incoherence (avoiding cases that are too similar) while other agents will be used for adding new cases, following an intelligent case-addition policy such as [13] or creating generalisations of similar cases. Distributing these different maintenance tasks among individual agents allows for a better fine tuning of the individual tasks and also for better adjusting the balance between conflicting case base maintenance aims such as minimality (and thus better performance) and competence [14].

Since our modularisation happens with respect to different sub-domains, the individual case bases that are the result of this modularisation are not absolutely independent of each other. Hence they have a net of dependencies between them that indicates what other case bases are affected by changes in one individual case base. These dependencies also affect some of the aforementioned maintenance tasks and split them into two different kinds: those which have to take other case bases into consideration and those which do not. A case factory agent that performs the task of maintaining a case base's uniqueness, minimality, incoherence or consistency can do this without knowing about other case bases. An agent that inserts new cases or adds new data to existing cases is a different matter.

3 Maintenance on Travel Medical Data

Travel medical data contain information about countries, diseases, medications, vaccinations, guidelines, and experiences. Aiming at higher accuracy each case base will serve its own topic and the case format will exactly fit for the type of knowledge. Furthermore the case bases will contain similarity measures, adaptation knowledge and the vocabulary to represent and retrieve cases.
Moreover, in travel medicine it can be dangerous to use CBR for the whole set of information, because the combination of medications, vaccinations, side effects, contraindications, etc. regarding the traveller's health history should be correct, without any contradicting information. Instead of that we will apply CBR for each topic and do the combination of the responses afterwards using the constraints given in the

response sets. Each issue handled in a case base will be provided using CBR methods and the strength of CBR, finding similar information on a given topic, will ensure a higher quality of information provision.

In this section we will exemplify four docQuery case bases that are each representing one topic and explain the dependencies between them. The selected case bases are examples to illustrate our approach. For the implementation of docQuery there will be at least six more case bases.

The case base *country* will contain specific country information a traveller has to consider planning a journey. Furthermore this information will be separated into sections a traveller has to pay attention to before, during, and after the journey. The country information also includes the required vaccinations and additional information, for example guidelines for a healthy journey.

The case base *disease* holds more than 100 diseases considered in a travel medical consultation. It focusses on diseases that might affect a traveller on a journey, for instance Malaria, Avian Influenza, or Dengue. A disease in this case base is characterised by general information on the disease, how to avoid the disease, how to behave if one has had the disease before, and how to protect oneself.

The third case base we will introduce is *medicament* that holds details about medicaments and their areas of application (diseases, vaccinations, age, etc.). Basically it contains information about active pharmaceutical ingredients, effectiveness, therapeutic field, contraindication, interdependences, and the countries in which those medicaments are approved. In *diseases* we do not store information on chronic illnesses, because they will be modelled in their own case base.

The fourth case base will hold *activities* which are used within docQuery to provide safety advice for intended activities when planning a journey. For travellers, activities are the major part of their journey, but they may involve certain risks for which safety advice is needed and furthermore when asked for their plans travellers will usually describe their activities which we can use to provide better guidance. Examples of such activities are diving, hill-climbing or even swimming.

Complete travel medical information will contain knowledge of all four case bases enhanced with descriptions, guidelines, and previous experiences. The combination of the information retrieved from each case base will be done by a so called Coordination Agent. The Coordination Agent will request each agent and based on the agents' response and the given information by the traveller the next request containing all constraints will be created and send to another Topic Agent.

Storing new cases in one of the case bases possible dependencies on other cases bases have to be observed. For instance if a new case is inserted into the *diseases* case base, or an existing disease breaks out in a new region, the inserting agent has to check, whether every risk region indicated in the respective disease's *regions* attribute is actually included in the *country* case base's underlying taxonomy, otherwise this *diseases* case will never be retrieved. The same is true for new data being added to the *medicament* case base. Here the *area of application* has to yield at least one result in the *disease* case base; otherwise the new data have to be passed to a domain expert for a review. Although this approach is rather maintenance-intensive, our medical application scenario requires this very conservative case addition strategy in order to preserve the system's overall accuracy and the case factory approach allows us to realise this strategy and also adapt to new dependencies, should they arise.

Inter-case base dependencies arise from features that influence each other even if they are contained in different case bases. Fig. 1 shows the case formats for the four case bases and their inter-case base dependencies on which we will give some examples.

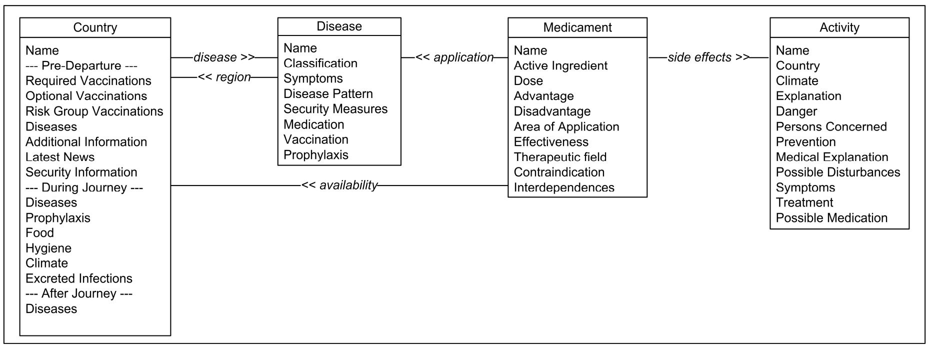

Fig. 1. Case formats for four case bases that provide knowledge for the Topic Agents: country, disease, activity and medicament and their inter-case base dependencies. This figure shows what kind of information is stored in the case bases and gives some examples on how dependencies can be defined.

An example for a travel medical request might be that a German family will travel in March to Alor to dive and afterwards they will travel around Bali by car. Using the given information all four case bases have to be requested to find appropriate information. The first information given by the travellers is the fact that they will go to Alor and Bali in March. Since Alor and Bali are not countries, first we have to figure out where those places are situated: both are Indonesian islands. Furthermore, the travelling period will be March, which will be in the rainy season. Now the inter-case base dependencies *disease* and *region* have to be applied, because travelling to Indonesia, especially in the region of Alor/Flores/Komodo/Sumbawa/Sumba will require a malaria prophylaxis. For that reason information on malaria has to be provided and therefore the home destination is required, because depending where travellers come from, certain medicaments can be recommended, or not. Also the dependency *availability* describes if a certain medicament is approved in that particular country. Medicaments can cause *side effects* which might influence the planned activities during a journey. An example for such a side effect is that Doxycyclin Monohydrat as malaria prophylaxis causes light toxicity in some cases. Light toxicity can occur using some kinds of medicaments and causes people's skin to react hypersensitive to sun. As a consequence, the traveller should try to avoid the sun and use a high factor sun blocker.

In this case this side effect has either to be mentioned to the traveller (together with information on how to cope with it) or another medicament has to be suggested. Of course the choice has to fit the disease, but it must also be considered whether the traveller can take it or not (*application*). If the travellers in our example take this medicament as malaria prophylaxis they have to be careful to get information how they should behave and which side effects might occur while diving, swimming, etc. We mentioned above one side effect and a possible safety measure, but usually there are many more and docQuery is generally aiming to give travellers the opportunity to follow their activities instead of forbidding them. Furthermore we are trying to give the travellers the opportunity to pursue their activities safely by following medical advice.

As the given example shows our application domain is very complex and we decided to follow the modularised knowledge structure to achieve high quality case bases and in a second step combine the results of each field of expertise. The dependencies between the case bases will help us to create valid combinations and maintain all relevant data. In our architecture the dependencies are modelled in the request strategies and support the Coordination Agent to send requests and constraints to each Topic Agent.

We expect *country* information to change very frequently, because these cases do not only contain geographic but also security information. The content of the case bases *disease* and *medicament* will change less often and the most static cases will be in *activity*, because its characteristics do not vary once they are covered. Also the information in our case bases can be inserted individually, for instance new medicaments can be inserted in their case base as they are discussed in the expert's web community without having to change the other case bases.

4 Conclusion and Outlook

In this paper we gave an introduction to the CoMES approach and its resulting architecture for intelligent information systems, SEASALT. Within SEASALT we focused on knowledge modularisation, which splits a complex domain into several sub-domains, each represented by an individual case base (Knowledge Line), and how such a modularised knowledge base can be maintained using intelligent agents (Case Factory).

We first gave an overview on already existing work in CBR maintenance and detailed our own position in respect to the individual approaches. Following this, we introduced the SEASALT architecture and its main components, the Knowledge Line and the Case Factories. Subsequently we discussed how a system of interdependent case bases can be maintained, adopting existing approaches, transferring them to a multi-agent setup, and adapting them to the additional requirements of inter-case base dependencies. Next we evaluated our methodology by applying it to the real-life application scenario of travel medicine. We will research, whether the SEASALT architecture will help to develop the travel medicine application within a shorter amount of time and/or with less effort and/or with a more complete knowledge base etc., than other approaches like, for instance, a "common CBR" approach.

Once the docQuery system has been fully implemented we will use the SEASALT architecture to realise other, related application scenarios such as FLOSSWALD [15], an information system based on free/open source software and SIMOCOSTS [16], a model for simulating cognitive processes in order to analyse how human beings act in critical situations by emphasising which coping strategies they use.

References

1. Plaza, E.: Semantics and Experience in the Future Web. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) Proc of 9th European Conference on Case-Based Reasoning, LNAI, Vol. 5239, pp. 44-58. Springer Verlag. (2008)
2. Althoff, K.-D., Bach, K., Deutsch, J.-O., Hanft, A., Mänz, J., Müller, T., Newo, R., Reichle, M., Schaaf, M., Weis, K.-H.: Collaborative Multi-Expert-Systems – Realizing Knowledge-Product-Lines with Case Factories and Distributed Learning Systems, In: Baumeister, J., Seipel, D. (eds) Proc. Workshop Knowledge Engineering & Software Engineering, (2007)
3. Van der Linden, F., Schmid, K., Rommes, E. (eds.) Software Product Lines in Action – The Best Industrial Practice in Product Line Engineering. Springer Verlag, Berlin (2007)
4. Weiss, G.: Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence. The MIT Press (1999).
5. Basili, V.R., Caldiera, G., Rombach, H. D.: Experience Factory. In: Marciniak, J.J (ed.) Encyclopedia of SE, Vol 1, pp. 469–476. John Wiley & Sons, New York (1994)
6. Althoff, K.-D., Hanft, A., Schaaf, M. Case Factory – Maintaining Experience to Learn. In: Göker, M., Roth-Berghofer, T. (eds.) In: Advances in Case-Based Reasoning – Proceedings of the 8th European Conference, Fethiye, Turkey, September 2006. LNAI, vol. 4106, pp. 429–442. Springer Verlag, Berlin Heidelberg (2006)
7. Bach, K.: docQuery – A Medical Information System for Travellers. Internal project report, September 2007, University of Hildesheim (2007)
8. Bach, K., Reichle, M., Althoff, K.-D.: A Domain Independent System Architecture for Sharing Experience, In Hinneburg, A.(ed): Proc. of LWA 2007, Workshop Wissens- und Erfahrungsmanagement, pp. 296–303, Martin-Luther-University Halle-Wittenberg, (2007)
9. Prasad, M. V. N., Lesser, V. R., Lander, S. E.: On retrieval and reasoning in distributed case bases. In: Proc. IEEE Intl. Conference on Systems, Man, and Cybernetics 1995 (1995)
10. Plaza, E., McGinty, L.: Distributed case-based reasoning, The Knowledge Engineering Review, Vol. 1–4. 2005, Cambridge University Press, Cambridge (2005)
11. Leake, D. B., Sooriamurthi, R.: Dispatching Cases versus Merging Case-Bases: When MCBR Matters. In: Proc. of the 16th Intl. Florida Artificial Intelligence Research Society Conference, FLAIRS-2003, pp. 129–133 (2003)
12. Reinartz, T., Iglezakis, .I, Roth-Berghofer, T.: On quality measures for case base maintenance. In: Proceedings of the 5th European Workshop on Case-Based Reasoning (EWCBR). LNAI, vol 1898, pp. 247–259. Springer-Verlag, Berlin Heidelberg (2000)
13. Zhu, J. Yang, Q.: Remembering to add: Competence–preserving case addition policies for case base maintenance. In: Proceedings of the International Joint Conference in Artificial Intelligence, Stockholm, pp. 234–241. Morgan Kaufmann, San Francisco (1999)
14. Leake, D. B., Wilson, D. C.: Guiding Case-Base Maintenance: Competence and Performance. In: Proceedings of the 14th European Conference on Artificial Intelligence (ECAI), Workshop on Flexible Strategies for Maintaining Knowledge Containers, pp. 47–54. (2000)
15. Hanft, A., and Reichle, M.: The FLOSSWALD Information System on Free and Open Source Software, In Gronau, N. (ed): Proc. of the 4th Conference on Professional Knowledge Management - Experiences and Visions, Gito Verlag, 135 - 142, (2007)
16. Newo, R., and Althoff, K.-D.: Simulating Coping Processes in Critical Situations - An Agent based Approach. In: Klügl, F., Timpf, S., and Schmid, U. (eds.) Proc. of the Workshop on Agent-Based Simulation: From Cognitive Modelling to Engineering Practice (KI 2008), Kaiserslautern, Germany, (2008)