# Chapter 9
# Case-Based Reasoning in a Travel Medicine Application

Kerstin Bach, Meike Reichle, and Klaus-Dieter Althoff

Intelligent Information Systems Lab
University of Hildesheim
Marienburger Platz 22, 31141 Hildesheim, Germany
lastname@iis.uni-hildesheim.de
http://www.iis.uni-hildesheim.de

**Abstract.** This chapter focuses on knowledge management for complex application domains using Collaborative Multi-Expert-Systems. We explain how different knowledge sources can be described and organized in order to be used in collaborative knowledge-based systems. We present the docQuery system and the application domain travel medicine to exemplify the knowledge modularization and how the distributed knowledge sources can be dynamically accessed and finally reassembled. Further on we present a set of properties for the classification of knowledge sources and in which way these properties can be assessed.

## 1 Introduction

This chapter will give an introduction how Case-Based Reasoning (CBR) can be used, among other methodologies from the field of Artificial Intelligence (AI), to build a travel medical application. There are a high variety of AI methods and we focus on how these methods can be combined, coordinated and further developed to meet the requirements of an intelligent information system. We will use Information Extraction techniques to analyze text, we have multi-agent technologies to coordinate the different methods that are executing the retrieval and in the following to combine the result sets. Further on we have to deal with rules and constraints that insure correct results and since we are accessing different kinds of knowledge sources we use XML and RDF as description languages. Within our application CBR will be the main underlying methodology. We will explain how travel medical case bases can be structured, how the required knowledge can be acquired, formalized and provided, as well as how that knowledge can be maintained.

The chapter will begin by describing Aamodt's and Plaza's 4R cycle [1] from the travel medical point of view. Then it will explain how the CBR methodology fits in the travel medical application domain. For this purpose we will present an intelligent information system on travel medicine, called docQuery, which is based on CBR and will serve as a running example throughout the whole chapter.

I. Bichindaritz et al. (Eds.): Computational Intelligence in Healthcare 4, SCI 309, pp. 191–210.
springerlink.com 

Following the definition and motivation of travel medicine as an application domain, we will introduce a novel approach to CBR using a number of heterogeneous case bases of which each one will cover one individual field within the general application domain. Each case base provides information that is required to compute a travel medical information leaflet and we will describe how we manage the case bases and use them to compose such information leaflets with regard to given constraints.

Moreover we will point out how we keep our case bases up-to-date using a web-based community as source of information. We will further describe the technologies we use to extract information, knowledge and experiences from the community, formalize them and use a Case Factory for its maintenance. Additionally we describe how and in which way techniques from Machine Learning and Information Extraction can be applied to extend a case base. The chapter will close up with a discussion of related topics followed by a short summary and future developments in this area.

## 2 Requirements of Travel Medicine as an Application Domain

Today the World Wide Web is a widely accepted platform for communication and the exchange of information. It does not matter where people live, to which culture they belong or of which background they are - web communities can be used from anywhere and by anyone. Especially in discussion forums a lot of topics are reviewed and experiences are shared. Unfortunately, much information gets lost in discussion boards or web pages caused by the quantity and variety of different web communities. Hence it is hard to find detailed information, since the topic of a discussion is often not clear and a wide range of expressions are used. Furthermore the users do not know enough about the authors and their background to ensure a high quality of information.

### 2.1 Motivation

Nowadays it is easier than ever to travel to different places, experience new cultures and get to know new people. In preparation for a healthy journey it is important to get a high quality and reliable answer on travel medicine issues. Both layman and experts should get information they need and, in particular, they understand. For that reason the idea of docQuery - a medical information system for travelers - has been developed.

docQuery provides information for travelers and physicians (those who are no experts in the field of travel medicine) by travel medicine experts and also gives an opportunity to share information and ensures a high quality because it is maintained by experts. Furthermore it will rise to the challenge of advancing the community alongside their users. User can obtain detailed information for their journey by providing the key data on their journey (like travel period, destination, age(s) of traveler(s), activities, etc.) and the docQuery system will prepare an information leaflet the traveler can take to his general practitioner to discuss the planned journey. The leaflet will contain all the information needed to be prepared and provide detailed information if they are required. In the event that docQuery cannot answer the traveler's question, the request will be sent to experts who will answer it. Further on, those information will not only be provided to the user, it will also be included in the docQuery case base so it will be available for future requests.

The information contained in docQuery will be processed using several methods from the field of artificial intelligence, especially CBR. Both existing knowledge about countries, diseases, prevention, etc. and experiences of travelers and physicians will be integrated and aid in further advancing docQuery's knowledge base. docQuery will provide information for travel medicine prevention work and can be used by:

- Physicians, who are advising their patients
- Physicians, who provide their knowledge
- Travelers who plan a journey and look for information about their destination

Because of docQuery's individual query processing and information leaflet assembling, the system is able to adapt to different target audiences.

### 2.2 Travel Medicine

Travel medicine is an interdisciplinary medical field that covers many medical areas and combines them with further information about the destination, the activities planned and additional conditions which also have to be considered when giving medical advise to a traveler. Travel medicine starts when a person moves from one place to another by any mode of transportation and stops after returning home without diseases or infections. In case a traveler gets sick after a journey a travel medicine consultation might also be required. A typical travel medical application could be a German family who wants to spend their Easter holidays diving in Alor to dive and afterward they will travel around Bali by car. In case a traveler gets sick after a journey a travel medicine consultation might also be required. First of all we will focus on prevention work, followed by information provision during a journey and information for diseased returnees. Since there are currently no sources on medical information on the World Wide Web that are authorized by physicians and/or experts, we aim at filling this gap by providing trustworthy travel medical information for everybody.

mediScon[1] is a team of certified doctors of medicine from European countries with a strong background in tourism related medicine, e.g. tropical medicine, and will support docQuery by providing travel medical information and assisting the modeling of the information. It is self-supporting and independent, and all information is scientifically proven and free of advertising. docQuery will provide all the information existing on mediScon and its sub domains. Hence the community can be used to provide new information and give feedback on given

[1] http://www.mediscon.com/

advices to ensure a high quality of information. Any information in docQuery is maintained by experts so users can trust the system.

docQuery will aim at providing high quality travel medicine information on demand. The system will not provide a huge amount of data that the traveler has to go through - instead, it focuses on the information the traveler already has and extend it with the required information required to travel healthily. Furthermore we will integrate the users of docQuery in its development. On the one hand, experts will take part in the community by exchanging and discussing topics with colleagues, and on the other hand, the travelers will share their experiences.

The research project is a collaborative multi-expert system (CoMES) using subsymbolic learning algorithms and offering travel medicine prevention work for any traveler. Each request will be processed individually, although the system will not substitute consulting a general practitioner. The leaflet should inform travelers and enable them to ask the right questions. Furthermore, the information given should help them to travel healthily and enjoy their stay. In developing docQuery the following requirements set our goal:

- Providing reliable, scientifically proven, up-to-date and understandable Information
- Giving independent information (no affinity to any pharmaceutical company)
- Informing any travelers without charging them
- Intuitional usability of the Front-End (accessible with a common web browser via WWW)
- Universally available
- Offering a communication platform for experts and travelers
- Enabling a multilingual and multicultural communication
- Applying new technologies and focusing on social problems to further their solution

docQuery is support travelers giving trustworthy information based on key data like destination, travel period, previous knowledge, planned activities and language. The information on the leaflets will cover the following issues:

- Medical travel prevention: vaccination, clarification of threats, information about medicaments
- Each information is tailored especially for the travelers and their needs - especially country-specific information as well as outbreaks or natural disasters (e.g. hurricanes, tsunamis, earthquakes)
- Information about local hospitals at the destination - especially hospitals where the native language of the traveler is spoken
- Outbreaks of diseases and regional epidemics
- Governmental travel advice
- General information and guidelines like "What to Do if..." in case of earthquakes, volcanic eruptions, flooding, etc.

docQuery is the core application and supports establishing a community to exchange experiences. Furthermore the users are involved in advancing the knowledge provided by docQuery and influence which issues are raised by sending requests, giving feedback and sharing experiences. docQuery is supposed to be a non-profit project and will provide travel medical information, prevention and preparation free of charge.

## 3 4R Cycle from the Travel Medical Point of View

CBR is a methodology based on Schank's theory [2] on the transfer of the function of human behavior onto computational intelligence. The main idea describes how people's experiences (or parts of the experiences) are remembered and later reconsidered when facing new and similar problems, reusing or adapting the previous solution in order to solve new problems. In CBR a case is described as a problem and its according solution.

In comparison to other methodologies like logical programming, CBR can deal with incomplete information and the domain does not have to be completely covered by a knowledge model before a system can be built. The integrated learning process allows a CBR system to learn while it is used. Based on Schank's ideas Aamodt and Plaza [1] introduced the Retrieve-Reuse-Revise-Retain (4R) process cycle that is until today the reference model for CBR applications.

Today there are three major types of CBR systems: Textual CBR systems [3] that basically deals with textual cases and combine Natural Language Processing Techniques with the CBR approach. Conversational CBR applications with are characterized by subsequent retrieval steps that narrow down the possible solution by iteratively setting attributes and the most often applied approach of Structural CBR which features a strict case representation and various retrieval techniques. Today many applications combine those approaches according to the given system requirements. A more detailed description of the CBR approaches and their application domains is discussed by Bergmann [4].

Although not all applications implement each process, most CBR systems are based on this model. To describe the 4R cycle, we will again use the travel medicine example presented in section 2.2 and illustrate the 4R example in Figure 1.

The current situation is a family plans that to spend their Easter holidays in Alor and Bali. This is the problem description that has to be transfered in the problem representation to initiate a *retrieval* request. Within this example we are only looking for vaccination suggestions. To enable an efficient retrieval different case base indexing structures have been developed and each of them addresses special features of a CBR type. Before a retrieval can be executed the problem has to be analyzed so an similarity-based retrieval within the case base can be executed.

The case base or knowledge base contains previous cases as well as background knowledge, this can be *rules* to either complete requests (enriched with tags) or modify solutions, or *similarity measures* to compute the similarity between two

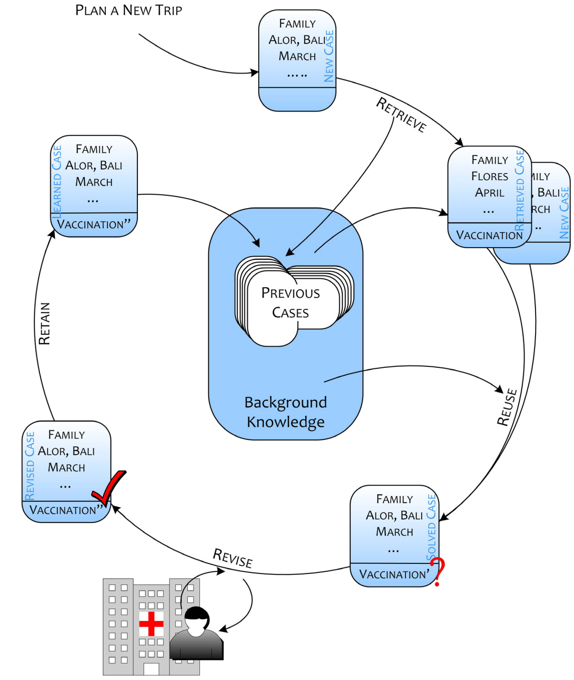


**Fig. 1.** 4R Cycle from the Travel Medical Point of View

cases or *vocabulary* to recognize keywords. The so called knowledge containers are described in more detail by Richter [5].

The background knowledge is required to find similar cases during the retrieval process and if necessary adapt solutions. After the retrieval process has been executed a CBR system has an ordered set of possible solutions, which usually do not match perfectly. Hence they need some modification and the *reuse* process is initiated. Within this process the system uses background knowledge, mostly adaptation rules, to change the solution in order to exactly (or as close as possible) fit to the problem description. In our example we can exchange the month April through March because the background knowledge, i.e. a rule, says that both months are in the rainy season and though can be handled equally. Also we can substitute Flores through Alor and Bali because this are all Indonesian islands with very similar properties (geographical location, climate, etc.).

Figure 2 exemplifies one type of knowledge representation for geographic regions. Based on continents, regions and countries we have developed a taxonomy that can be used to find similar countries. Knowledge models like taxonomies or light ontologies provide different types of knowledge in terms of knowledge containers: the names of the nodes and leafs are representing *vocabulary knowledge*. Since those terms are ordered in a taxonomy, *similarities* between countries, here regarding their geographic position. Also *adaptation knowledge* can be acquired

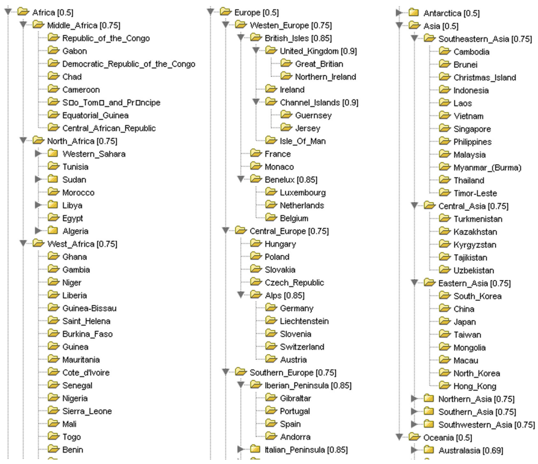


**Fig. 2.** Knowledge model representing the geographical location of countries

from taxonomies, because countries that share a node also might have features in common that can be applied completing incomplete case as described in Bach et. al. [6]. Along with taxonomies, similarity measures for symbolic representations can also be realized using tables, ontologies or individually defined data strucutres.

After the adaptation process has been executed the new case has to be revised. The *revision* can either be realized using again background knowledge or external feedback. In our example we send our solution to an expert who revises the case manually and gives feedback. Afterwards we have a new revised problem-solution pair (case) that can be included in the case base. In this way the case base and thus the whole CBR system is able to learn and to adapt to different circumstances.

## 4 Underlying Architecture

docQuery will be an intelligent information system based on experts which are distributed all over the world and use the platform giving information to travelers and colleagues. The implementation will pursue an approach mainly based on software agents and CBR. Both software agent and CBR have already been used to implement experience based systems [7,4,8]. docQuery will use different knowledge sources (diseases, medications, outbreaks, guidelines, etc.) which are created in cooperation with experts, provided in databases and maintained by the users of docQuery. However, medicine cannot deal with vague information how they might occur in extractions of community knowledge. Therefore we also integrated data bases as knowledge sources in case exact matches are required.

Collaborative Multi-Expert-Systems (CoMES) are a new approach presented of Althoff et. al.[9] which presents a continuation of combining established techniques and the application of the product line concept (known from software engineering) creating knowledge lines. Furthermore this concept describes the collaboration of distributed knowledge sources which makes this approach adequate for an application scenario like docQuery. The system will follow the CoMES-architecture, called SEASALT (Sharing Experience using an Agent-based System Architecture LayouT), as it can be seen in Figure 3 and is explained in detail in Reichle et. al.[10].

The SEASALT architecture provides an application-independent architecture that features knowledge acquisition from a web-community, knowledge modularization, and agent-based knowledge maintenance. It consists of five main components which will be presented in the remaining of this section.

The SEASALT architecture is especially suited for the acquisition, handling and provision of experiential knowledge as it is provided by communities of practice and represented within Web 2.0 platforms [11]. The *Knowledge Provision* in SEASALT is based on the *Knowledge Formalization* that has been extracted from WWW *Knowledge Sources*. Knowledge Sources can be wikis, blogs or web forums in which users, in case of docQuery travel medicine experts, provide different kinds of information. They can for instance discuss topics in web forums which are broadly established WWW communication medium and provide a low entry barrier even to only occasional WWW users. Enabling an analysis of the discussed topics, we enhanced the forum with agents for different purposes. Additionally its contents can be easily accessed using the underlying data base. The forum itself might serve as a communication and collaboration platform for the travel medicine community, which consists of professionals such as scientists and physicians who specialize in travel medicine and local experts from the health sector and private persons such as frequent travelers and globetrotters. The community uses the platform for sharing experiences, asking questions and general networking. The forum is enhanced with agents that offer content-based services such as the identification of experts, similar discussion topics, etc. and communicate by posting relevant links directly into the respective threads [12].

The community platform is monitored by a second type of agents, the so called Collector Agents. These agents are individually assigned to a specific Topic Agent, their task is to collect all contributions that are relevant with regard to their assigned Topic Agent's topic. The Collector Agents pass these contributions on to the Knowledge Engineer and can in return receive feedback on the delivered contribution's relevance. Our Collector Agents use information extraction tools, like GATE [13] or TextMarker [14] to judge the relevance of a contribution. The Knowledge Engineer reviews each Collector Agent's collected contributions and realizes his or her feedback by directly adjusting the agents' rule base.

The SEASALT architecture is also able to include external knowledge sources by equipping individual Collector agents with data base or web service protocols or HTML crawling capabilities. This allows us to include additional knowledge sources such as the web pages of the Department of Foreign Affairs or the WHO.

In order for the collected knowledge to be easily usable within the Knowledge Line the collected contributions have to be formalized from their textual

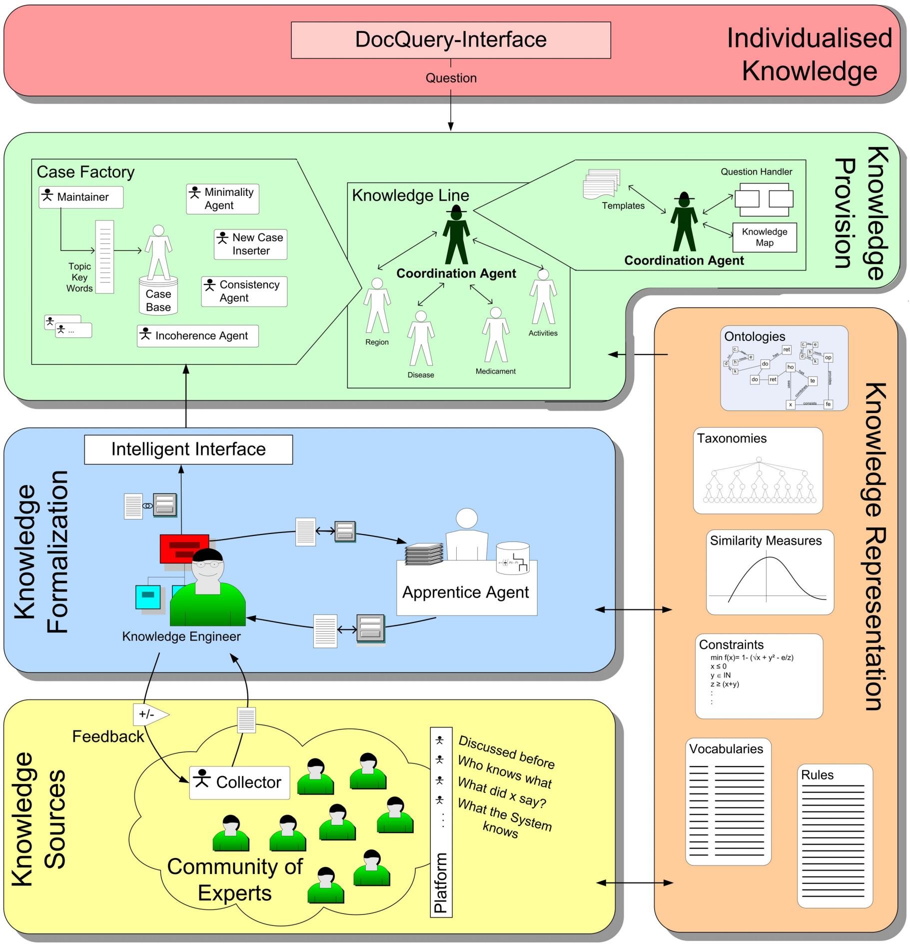


**Fig. 3.** SEASALT Architecture

representation into a more modular, structured representation. This task is mainly carried out by the Knowledge Engineer. In the docQuery project the role of the Knowledge Engineer is carried out by several human experts, who execute the Knowledge Engineer's tasks together. The Knowledge Engineer is the link between the community and the Topic Agents. He or she receives posts from the Collectors that are relevant with regard to one of the fields, represented by the Topic Agents, and formalizes them for insertion in the Topic Agents' knowledge bases using the Intelligent Interface. In the future the Knowledge Engineer will be additionally supported by the Apprentice Agent. The Intelligent Interface serves as the Knowledge Engineer's case authoring work bench for formalizing textual knowledge into structured CBR cases. It has been developed analogous to [15] and offers a graphical user interface that presents options for searching, browsing and editing cases and a controlled vocabulary.

The Apprentice Agent is meant to support the Knowledge Engineer in formalizing relevant posts for insertion in the Topic Agents' knowledge bases. It is trained by the Knowledge Engineer with community posts and their formalizations. The apprentice agent is currently being developed using GATE [13] and RapidMiner [16]. We use a combined classification/extraction approach that first classifies the contributions with regard to the knowledge available within the individual contributions using term-doc-matrix representations of the contributions and RapidMiner then attempts to extract the included entities and their exact relations using GATE. Considering docQuery's sensitive medical application domain we only use the Apprentice Agent for preprocessing. All its formalizations will have to be reviewed by the Knowledge Engineer, but we still expect a significantly reduced workload for the Knowledge engineer(s).

Although CoMES is a very new approach, the used techniques, like the Experience Factory[7], Case-Based Reasoning or Software Agents are well known. docQuery will integrate those techniques in a web community and creating an intelligent information system which is based on the knowledge of experts, experiences discussed on discussion boards and novelties presented by travel medicines that are a part of the community. Sharing knowledge at this level furthers the web 2.0 approach and allows us to develop new techniques.

## 5 Combination of Heterogeneous Knowledge Sources

When dealing with complex application domains it is easier to maintain a number of heterogeneous knowledge sources than one monolithic knowledge source. The knowledge modularization within SEASALT is organized in the Knowledge Line that is based on the principle of product lines as it is known from software engineering [17] and we apply it to the knowledge in knowledge-based systems, thus splitting rather complex knowledge in smaller, reusable units (knowledge sources). Moreover, the knowledge sources contain different kinds of information as well as there can also be multiple knowledge sources for the same purpose. Therefore each source has to be described in order to be integrated in a retrieval process which uses a various number of knowledge sources (see the third layer (Knowledge Line) in Figure 3).

The approach presented in this work does not aim at distributing knowledge for performance reasons, instead we are planning to specifically extract information for the respective knowledge sources from WWW communities or to have experts maintaining one knowledge base. Hence, we are creating knowledge sources, especially CBR systems, that are accessed dynamically according to the utility and accessibility to answer a given question. Each retrieval result of a query is a part of the combined information as it is described in the CoMES approach [18].

For each specific issue a case or data base will be created to ensure a high quality of knowledge. The data structure of each issue is different and so is the case format and domain model. Creating high quality "local knowledge bases" will guarantee the high quality of the systems knowledge.

### 5.1 Knowledge Sources

Considering knowledge sources, different characteristics, and aspects on which to assess knowledge source properties come to mind. The possible properties can refer to content (e.g. quality or topicality) as well as meta-information (e.g. answer speed or access limits). These properties do not only describe the individual sources but are also used for optimizing the query path. When working with distributed and – most importantly – external sources it is of high importance to be able to assess, store and utilize their characteristics in order to achieve optimal retrieval results. In detail we have identified the meta and content properties for knowledge sources (see Table 1, a more detailed description can be found in Reichle et. al. [19]).

**Table 1.** Knowledge source properties

| Meta Property | Content Property |
|---|---|
| Access Limits | Content |
| Answer Speed | Expiry |
| Economic Cost | Up-to-dateness |
| Syntax | Coverage |
| Format | Completeness |
| Structure | |
| Cardinality | |
| Trust or Provenance | |

Not all of the properties identified are fully unrelated. Properties like syntax, format, structure and cardinality for instance are partially related which allows for some basic sanity checks of their assigned values; also some of the properties such as answer speed, language or structure can be automatically assessed. Apart from these possibilities for automation the knowledge source properties currently have to be assessed and maintained manually by a Knowledge Engineer who assigns values to the properties and keeps them up to date. Adapting the properties' values based on feedback is only partially possible since feedback is mostly given on the final, combined result and it is thus difficult to propagate back to the respective knowledge sources. Also the more differentiated feedback is needed (in order to be mapped to the respective properties) the less feedback is given, so a good balance has to be found in this regard. Despite these difficulties the inclusion of feedback should not be ruled out completely. Even if good knowledge sources are affected by bad general feedback and the other way around the mean feedback should still provide a basic assessment of a knowledge source's content and can for instance be included in a combined quality measure. Depending on the respective properties we have defined possible values. Although there are not all properties usable for routing optimization, there are some properties like format, syntax, structure or content that cannot be used in the routing process since no valency can be assigned to them, that is one possible value cannot be judged as better or worse as the other. The computation of the routes with regard to defined properties is carried as described in Reichle et. al. [19].

docQuery will initially consist of the several heterogeneous knowledge sources and each type of knowledge source will cover on specific topic. Each knowledge source is accessible by the application and will be used to process the requests given by the user. Furthermore the knowledge sources will be able to be extended by more knowledge bases in future as well as maintenance processes can be defined for each knowledge base.

**Region:** For any country specific information consisting of "Before the journey", "During the journey" and "After the journey" will be provided. The country information includes required vaccinations and guides for a healthy journey. Further on this case base contains information on how to behave in various situations, which are explained to the users if necessary.

**Disease:** This knowledge base holds more than 100 diseases considered in travel medicine. They are described in detail and linked to medicaments, region, etc. It focuses on diseases that might affect a traveler on a journey, for instance Malaria, Avian Influenza, or Dengue. A disease in this case base is characterized 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.

**Medicament:** Details about medicaments and its area of application (diseases, vaccinations, age, etc.) used in the system are contained in this knowledge base. Basically it contains information about active pharmaceutical ingredients, effectiveness, therapeutic field, contraindication and interdependences.

**Dates/Seasons:** For each country we will cover dates and seasons. This information is used to assign the season to a request and subsequently only retrieve information that is necessary.

**Vaccinations:** If there are vaccinations recommended this database contains vaccination periods and types of vaccines. Further on it lists contraindications of each vaccination and experiences of users with the vaccinations in similar situations.

**Activity:** This case base will contain safety advice for intended activities when planning a journey. For travelers, 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 travelers will usually describe their activities which we can use to provide better guidance. Examples of such activities are diving, hill-climbing or even swimming.

**Health Risk:** This knowledge base contains information about health risks that might occur at a certain place under certain previously defined circumstances including medical details on prevention, symptoms and consequences. Further on it contains safety advice and the type of person who might be affected.

**Description:** Any information given in the system can be described in different ways. This knowledge base contains different descriptions which can be given to the user: there will be a specific and detailed description (e.g. for physicians), detailed descriptions for travelers (who are no physicians) and brief information (for experienced travelers as reminders, etc.).

**Guidelines:** This knowledge source will contain the "How to"-descriptions to help travelers to put the given information in practice. This case base is especially for travelers.

**Experience:** According to the motivation we will integrate the users experience to the system. This knowledge base will contain experiences and feedback given by travelers.

**Template:** This database contains templates to display the result created during the processing of the request. The templates will be used to ensure a structured and printable output.

**Profile:** This database contains user profiles of experts who edit data, administers or regular users who want to create a profile to get faster access to their required information.

### 5.2 Combination of Information Retrieved from Knowledge Sources

The flexible knowledge provision based on distributed, heterogeneous knowledge sources can be accessed in different ways. We combine retrieval results of several CBR systems embedded in a multi-agent system. The novelty of our approach is the use of heterogeneous case bases for representing a modularized complex knowledge domain. There have been other approaches using partitioned and/or distributed case bases, but still differ from our approach. In SEASALT the knowledge provision task is carried out by a so called Knowledge Line that contains a Coordination Agent and a number of Topic Agents that each covers one homogeneous area of expertise. In terms of SEASALT we use the modularization aspect to combine knowledge based on numerous different and homogeneous knowledge sources implemented as CBR software agents. The Coordination Agent is the center of the Knowledge Line and orchestrates the Topic Agents to enable the combination of the retrieval results. The implementation of the Coordination Agent followed a set of requirements that were derived from the SEASALT architecture description itself and from the implementation and testing of the Topic Agents.

During the design phase of the Coordination Agent the following requirements were identified:

- The case representations of the Topic Agents differ from each other as well as the agents' respective location might vary. This requires flexible access methods that are able to deal with distributed locations, different kinds of result sets and possibly also different access protocols.

- Some Topic Agents require another Topic Agent's output as their input and thus need to be queried successively, others can be queried at any time. In order for the Coordination Agent to be able to obey these dependencies they need to be indicated in the Knowledge Map in an easily comprehensible way.
- Based on the dependencies denoted in the Knowledge Map the agent needs to be able to develop a request strategy on demand. This request strategy should also be improvable with regard to different criteria such as the Topic Agents' response speed, the quality of their information, the possible economic cost of a request to a commercial information source and also possible access limits.
- In order to guarantee the quality of the final result of the incremental retrieval process there needs to be a possibility to control what portion of the result set is passed on to the subsequent Topic Agent. This portion should be describable based on different criteria such as the number of cases or their similarity.
- In order to allow for higher flexibility and a seamless inclusion in the SEASALT architecture the functionalities need to be implemented in an agent framework.

Firstly, in order for the Coordination Agent to be able to navigate the different knowledge sources a format for the Knowledge Map had to be designed and implemented. Since the dependencies between Topic Agents can take any form, we decided to implement the Knowledge Map as a graph where each Topic Agent is represented by a node and directed edges denote the dependencies. The case attributes that serve as the next Topic Agent's input are associated with the respective edges. The optimization criteria are indicated by a number between 0 (worst) and 100 (best) and are represented as node weights. In order to be able to limit the portion of the result that is passed on to the next node we implemented four possible thresholds, namely the total number of cases to be passed on, the relative percentage of cases to be passed on, the minimum similarity of cases to be passed on, the placement with regard to similarity of the cases to be passed on (For instance the best and second best cases). An example graph from the docQuery application can be seen in Fig. 4.

According to our example introduced in the beginning of this chapter the region agent would return a case including the information that Alor and Bali are Indonesian islands. Based on this information (i.e. Country = Indonesia) queries for general safety information about this country, diseases that can be contracted in the country, and certified (international standard) hospitals at the destination are initiated. In this example there are two agents offering that information: a free one[2] with information of lesser quality and a commercial one[3] with information of higher quality. The retrieved diseases (Malaria, Yellow Fever, Diphtheria, Tetanus, Hepatitis A, Typhoid Fever, etc.) are then subsequently used to query the medicaments agent for recommendable vaccinations

[2] The cost 100 denotes a minimal price, that is 0,-.

[3] The price is medium high, thus the cost value is 50, an agent with a higher cost would have an even lower cost value.

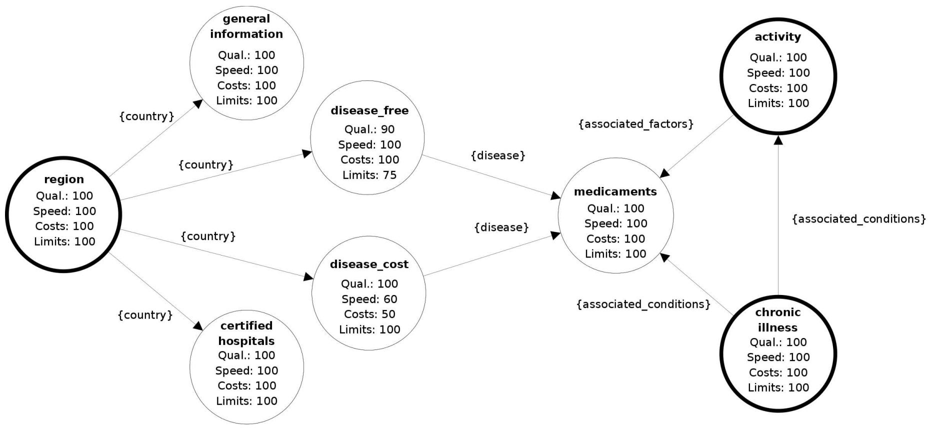


**Fig. 4.** Example graph based on the docQuery application

and medicaments that can be taken at the location. This query returns an initial list of recommendable medicament candidates. Further on, the information given by the user (Activities = "diving" and "road trip") is used to request information from the activity agent defining constraints for medicament recommendations (e.g. Activity = "Diving" ⇒ Associated_factors = "high sun exposure") which are then again used to query the medicaments agent. In this example a query for Counter_Indication = "high sun exposure" would return, among others, the Malaria prophylaxis Doxycyclin Monohydrat, which would then be removed from the initial list of recommended medicaments. Also, if specified, the influences of chronic illnesses on recommended medicaments and planned activities are queried. The combined information from all Topic Agents is compiled into an information leaflet using ready prepared templates. (*"When traveling to Indonesia, please consider the following general information: ... Certified hospitals can be found in the following places: ... A journey to Indonesia carries the following risks: ... We recommend the following medicaments: ... These medicaments are not recommended because of the following reasons ..."*) The Knowledge Map itself is stored as an XML document. We use RDF as the wrapper format and describe the individual nodes with a name space of our own. Based on the knowledge map we then use a modified Dijkstra algorithm [20] to determine an optimal route over the graph. The algorithm is modified in such a way that it optimizes its route by trying to maximize the arithmetic mean of all queried nodes. In the case of a tie between two possible routes the one with the lesser variance is chosen.

### 5.3 Maintenance of Knowledge Sources

docQuery deals with different kinds of data and each kind has to be maintained differently. We will define maintaining processes for each source focusing on exact, up-to-date and reliable data. Furthermore each source will have its own maintainer in case old or erroneous data has to be removed or corrected. To follow this goal the maintenance processes has to be created along with the data models regarding the interfaces and the applications built upon them.

To ensure up-to-date data the system has to be checked by experts regularly, and by integrating a web community new topics will have to be identified and new cases will have to be entered in the knowledge sources. For that purpose processes for updating (inserting, maintaining, deleting, extending, etc.) have to be implemented and established. For instance, we assume that a group of experts takes care of new entries in docQuery: In this case we are assigning topics with the expert's field of expertise to each of them and if there is a new discussion in the respective area detected, this is e-mailed to the expert so he oder she can follow this discussion. Further on, when the system extracted and processed information the complete set which should be inserted is sent to the expert and has to be approved before it can be inserted in the according case base. This proceeding is not for any application domain necessary, but since we deal with medical information we have to make sure that correct information are provided, although we are only giving information that do not substitute a medical consultation.

Even if we have different kinds of Topic Agents and their according Case Factories, the behavior of some Case Factory agents (like the new case inserter) can be reused in other Case Factories of the same Knowledge Line. We differentiate between agents that handle general aspects and are contained in any Case Factory and agents that are topic-specific and have to be implemented individually. General Case Factory agents usually focus on the performance or regular tasks like insertion, deletion, merging of cases. Topic specific Case Factory agents are for example agents that transfer knowledge between the knowledge containers [5] or define certain constraints and usually they have to be implemented for an individual topic considering its specifications or fulfilling domain dependent tasks. The Knowledge Line retrieves its information, which is formalized by a Knowledge Engineer and/or machine learning algorithms, from knowledge sources like databases, web services, RSS-feeds, or other kinds of community services and provides the information as a web service, in an information portal, or as a part of a business work flow. The flexible structure of the knowledge line allows designing applications incrementally by starting out with one or two Topic Agents and enlarging the knowledge line, for example with more detailed or additional topics, as soon as they are available or accessible.

## 6 Related Work

The approach of distributed sources has been a research topic in Information Retrieval since the mid-nineties. An example is the Carrot II project [21], which also uses a multi-agent-system to co-ordinate the document sources. However, most of our knowledge sources are CBR-systems, which is the reason why we concentrate on CBR-approaches. The issue of differentiating case bases in order to be more suitable for its application domain has been discussed before. Weber et al. [22] introduce the horizontal case representation, a two case base approach in which one contains the problem and the other one the solutions. They motivate splitting up the case bases for a more precise case representation, vocabulary and a simplified knowledge acquisition.

Retrieval strategies have been discussed in the context of Multi-Case-Base Reasoning in [23]. Leake and Sooriamurthi explain how distributed cases can be retrieved, ranked and adapted. Although they are dealing with the same type of case representations of the distributed case bases, both approaches have to determine whether a solution or part of solution is selected or not. The strategy of Multi-Case-Base Reasoning is to either dispatch cases if a case-base cannot provide a suitable solution or to use cases of more than one case base and initiate an adaptation process in order to create one solution.

Collaborating case bases have been introduced by Ontañón and Plaza [24] who use a multi-agent system to provide a reliable solution. The multi-agent system focuses on learning which case base provides the best results, but they do not combine or adapt solutions of different case bases. Instead their approach focuses on the automatic detection of the best knowledge source for a certain question.

Combining parts of cases in order to adapt given solutions to a new problem has been introduced by Redmond in [25] in which he describes how snippets of different cases can be retrieved and merged into other cases, but in comparison to our approach, Redmond uses similar case representations from which he extracts parts of cases in order to combine them. His approach and the knowledge provision in SEASALT have in common that both deal with information snippets and put them together in order to have a valid solution. Further on, Redmond mostly concentrates on adaptation while we combine information based on a retrieval and routing strategy.

Our notion of knowledge source properties is comparable to and thus benefits from advances in the respective field in CBR like the recent work of Briggs and Smyth [26], who also assign properties, but to individual cases. On the other hand the graph-like representation of the knowledge sources and its use in the composition of the final results do not have a direct equivalent in CBR. It depends on the cases' separation by topic and a clear dependency structure of the topics (e.g. *the country determines the possible diseases, the diseases determine the respective vaccinations and precautions, etc.*) which is not necessarily given in traditional CBR.

## 7 Conclusion and Final Remarks

The SEASALT architecture offers several features, namely knowledge acquisition from web 2.0 communities, modularized knowledge storage and processing and agent-based knowledge maintenance. SEASALT's first application within the docQuery project yielded very satisfactory results, however, in order to further develop the architecture we are planning to improve it in several areas. One of these are the Collector Agents working on the community platform, which we plan to advance from a rule-based approach to a classification method that is able to learn from feedback, so more workload is taken off the Knowledge Engineer.

docQuery is the first instantiation of SEASALT and has a strong focus on the knowledge modularization and reassembly with the goal to provide an information leaflet for a traveler. Moreover, docQuery shows how various AI methodologies can be used to realize an intelligent information system that provides complete and reliable information for individual journeys considering all aspects a travel medicine physician would do. We also introduced how various heterogeneous knowledge sources can be queried as well as we provided a web-based maintenance strategy that enables an intelligent system to use Web 2.0 platforms like web forums to extend its case base.

Travel medicine is for sure a specific application domain that cannot compared to any other application because the information we deal with are health related and we have to make sure that only correct and understandable are produced. We are confident that the techniques along with the SEASALT architecture can be used within different application domains that cover a combination of topics.

## References


1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications 1(7) (March 1994)
2. Schank, R.C.: Dynamic Memory: A Theory of Reminding and Learning in Computers and People. Cambridge University Press, New York (1983)
3. Wilson, D.C., Bradshaw, S.: Cbr textuality. In: Brüninghaus, S. (ed.) Proceedings of the Fourth UK Case-Based Reasoning Workshop, University of Salford, pp. 67–80 (1999)
4. Bergmann, R., Althoff, K.D., Breen, S., Göker, M., Manago, M., Traphöner, R., Wess, S.: Developing industrial case-based reasoning applications: The INRECA methodology. In: Bergmann, R., Althoff, K.-D., Breen, S., Göker, M.H., Manago, M., Traphöner, R., Wess, S. (eds.) Developing Industrial Case-Based Reasoning Applications, 2nd edn. LNCS (LNAI), vol. 1612. Springer, Heidelberg (2003)
5. Richter, M.M.: Introduction. In: Lenz, M., Bartsch-Spörl, B., Burkhard, H.D., Wess, S. (eds.) Case-Based Reasoning Technology. LNCS (LNAI), vol. 1400, p. 1. Springer, Heidelberg (1998)
6. Bach, K., Reichle, M., Althoff, K.D.: A value supplementation method for case bases with incomplete information. In: McGinty, L., Wilson, D.C. (eds.) Case-Based Reasoning Research and Development. LNCS (LNAI), vol. 5650, pp. 389–402. Springer, Heidelberg (2009)
7. Althoff, K.D., Pfahl, D.: Making software engineering competence development sustained through systematic experience management. Managing Software Engineering Knowledge (2003)
8. Minor, M.: Erfahrungsmanagement mit fallbasierten Assistenzsystemen. PhD thesis, Humboldt-Universität zu Berlin (Mai 2006)

9. 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 knowlegde-product-lines with case factories and distributed learning systems. In: Baumeister, J., Seipel, D. (eds.) Workshop Proceedings on the 3rd Workshop on Knowledge Engineering and Software Engineering (KESE 2007), Osnabrück (September 2007)
10. Reichle, M., Bach, K., Althoff, K.D.: The seasalt architecture and its realization within the docquery project. In: Mertsching, B., Hund, M., Aziz, Z. (eds.) KI 2009. LNCS, vol. 5803, pp. 556–563. Springer, Heidelberg (2009)
11. Plaza, E.: Semantics and experience in the future web. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 44–58. Springer, Heidelberg (2008)
12. Feng, D., Shaw, E., Kim, J., Hovy, E.: An intelligent discussion-bot for answering student queries in threaded discussions. In: IUI 2006: Proc. of the 11th Intl. Conference on Intelligent user interfaces, pp. 171–177. ACM Press, New York (2006)
13. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: Gate: A framework and graphical development environment for robust nlp tools and applications. In: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics, ACL 2002 (2002)
14. Klügl, P., Atzmüller, M., Puppe, F.: Test-driven development of complex information extraction systems using textmarker. In: Nalepa, G.J., Baumeister, J. (eds.) Algebraic Logic and Universal Algebra in Computer Science. CEUR Workshop Proceedings, vol. 425 (2008), CEUR-WS.org
15. Bach, K.: Domänenmodellierung im textuellen fallbasierten schließen. Master's thesis, Institute of Computer Science, University of Hildesheim (2007)
16. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale: Rapid prototyping for complex data mining tasks. In: Ungar, L., Craven, M., Gunopulos, D., Eliassi-Rad, T. (eds.) KDD 2006: Proc. of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, August 2006, pp. 935–940. ACM, New York (2006)
17. van der Linden, F., Schmid, K., Rommes, E.: Software Product Lines in Action - The Best Industrial Practice in Product Line Engineering. Springer, Heidelberg (2007)
18. Althoff, K.-D., Reichle, M., Bach, K., Hanft, A., Newo, R.: Agent based maintenance for modularised case bases in collaborative multi-expert systems. In: Proceedings of AI 2007, 12th UK Workshop on Case-Based Reasoning, December 2007, pp. 7–18 (2007)
19. Reichle, M., Bach, K., Reichle-Schmehl, A., Althoff, K.D.: Management of distributed knowledge sources for complex application domains. In: Hinkelmann, K., Wache, H. (eds.) Proceedings of the 5th Conference on Professional Knowledge Manegement - Experiences and Visions (WM 2009), March 2009. Lecture Notes in Informatics, pp. 128–138 (2009)
20. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271 (1959)
21. Cost, R.S., Kallurkar, S., Majithia, H., Nicholas, C., Shi, Y.: Integrating distributed information sources with carrot ii. In: Klusch, M., Ossowski, S., Shehory, O. (eds.) CIA 2002. LNCS (LNAI), vol. 2446, p. 194. Springer, Heidelberg (2002)
22. Weber, R., Gunawardena, S., MacDonald, C.: Horizontal case representation. In: Althoff, K.D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 548–561. Springer, Heidelberg (2008)

23. Leake, D.B., Sooriamurthi, R.: Automatically selecting strategies for multi-case-base reasoning. In: Craw, S., Preece, A.D. (eds.) ECCBR 2002. LNCS (LNAI), vol. 2416, pp. 204–233. Springer, Heidelberg (2002)
24. Ontañón, S., Plaza, E.: Learning when to collaborate among learning agents. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 394–405. Springer, Heidelberg (2001)
25. Redmond, M.: Distributed cases for case-based reasoning: Facilitating use of multiple cases. In: AAAI, pp. 304–309 (1990)
26. Briggs, P., Smyth, B.: Provenance, trust, and sharing in peer-to-peer case-based web search. In: Althoff, K.D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 89–103. Springer, Heidelberg (2008)