Heras Barberá, SM.; Garcia Pardo Gimenez De Los Galanes, JA.; Ramos-Garijo Font De Mora, R.; Palomares Chust, A.; Botti Navarro, VJ.; Rebollo Pedruelo, M.; Julian Inglada, VJ. (2009). Multi-domain case-based module for customer support. Expert Systems with Applications. 36(3):6866-6873. doi:10.1016/j.eswa.2008.08.003.

Multi-Domain Case-Based Module for Customer Support ⋆

Stella Heras [a,*] Juan Ángel García-Pardo [a]

Rafael Ramos-Garijo [a] Alberto Palomares [b] Vicente Botti [a]

Miguel Rebollo [a] Vicente Julián [a]

[a] *Information Systems and Computing Department*

Universidad Politécnica de Valencia

Camino de Vera s/n. 46022 Valencia, Spain. (+34) 96 387 73 50 Ext: 83528

[b] *TISSAT S.A., Parque Tecnológico*

Av. Leonardo Da Vinci, 5, 46980 Paterna - Valencia, Spain. (+34) 96 393 99 00

Abstract

Technology Management Centres provide technological and customer support services for private or public organisations. Commonly, these centres offer support using a helpdesk software that facilitates the work of their operators. In this paper, a CBR module that acts as a solution recommender for customer support environments is presented. The CBR module is flexible and multi-domain, in order to be easily integrable with any existing helpdesk software in the company.

Key words: Helpdesk Systems, Customer Support, Decision Support Systems, Case-Based Reasoning

1 Introduction

Technology Management Centres (TMCs) are entities which control every process implicated in the provision of technological and customer support services to private or public organisations. Usually, TMCs are managed by a private company that communicates with its customers via a call centre. This kind of centres allow customers to obtain general information, purchase products or lodge a complaint. They can also efficiently communicate public administrations with citizens. In a call centre, there are a number of operators attending to a big amount of calls with different objectives –sales, marketing, customer service, technical support and any business or administration activity–. The call centre operators have computers provided with a helpdesk software and phone terminals connected to a telephone switchboard that manages and balances the calls among operators.

Nowadays to differentiate a company over other companies competing in the same market is very difficult. Products, prices and quality are very similar and companies try to obtain an advantage over their competitors by offering a careful attention to their customers. Most commercial activity is done via phone and it is necessary to avoid non-answered calls, busy lines, to ask the customer for repeating the query several times or to give incoherent answers. Moreover, a good customer support depends, in many cases, on the experience

⋆ This work was partially supported by CONSOLIDER-INGENIO 2010 under grant CSD2007-00022 and by the Spanish government and FEDER funds under CICYT TIN2005-03395 and TIN2006-14630-C0301 projects.

* Corresponding author
Email address: sheras@dsic.upv.es (Stella Heras).
URL: www.dsic.upv.es/users/ia/ia.html (Stella Heras).

and skills of its operators. A quick and accurate response to the customers problems ensures their satisfaction and a good reputation for the company and, therefore, it can increase its profits.

To store, and reuse later, the solution applied to each problem and the information about the problem-solving process could be a suitable way to improve the customer support offered by a company. Case-Based Reasoning (CBR) systems have been widely applied to perform this task. A CBR system tries to solve a problem (i.e. case) by means of reusing the solution of an old similar case [10]. This solution is previously stored in a memory of cases (i.e. case-base) and it can either be retrieved and applied directly to the current problem, or revised and adapted to fit the new problem. The suitability of CBR systems in helpdesk applications to manage call centres has been guaranteed for the success of some of these systems from the 90s to nowadays.

The Compaq *SMART* System [2], developed in collaboration with *Inference Corporation*, provided automated technical support directly to the users of its products. The system significantly increased the amount of satisfactorily solved calls and, therefore, the customer satisfaction. The software company *Broderbund*, also collaborating with *Inference Corporation*, incorporated the CBR methodology into its web-based customer support system *GizmoTapper* [20]. This system gave an on-line customer support, consistent with the information of databases employed by the operators of the company. This fact ensured the immediate availability of new information for problem solving. *Union Camp Corporation* developed *SmartUSA* [12], a self-improving helpdesk system intended for reduce or avoid, if possible, the amount of calls received by its operators. More recently, in the course of the *INRECA-II* [7] project, the internal CAD/CAM helpdesk system *Homer* [6][14] was developed. The

system helped the operators of the Information Technologies department at the *Daimler Chrysler* company to solve the problems that arise from software and hardware updates. In addition to these examples, there are a lot of CBR systems successfully applied in helpdesk and customer support domains [16][11][15][9][13]. Many companies also sell products that apply CBR to helpdesks –e.g. *eGain Service 7 Suite* from *eGain* [4] (formerly *Inference Corporation*), *Kaidara Advisor* from *Kaidara* [8] (formerly AcknoSoft) and *empolis Orenge* from *Empolis* [5] (formerly tec:inno GmbH)–.

However, most of the applications reported are complete case-based systems that were developed for research purposes or designed by any vendor for covering the needs of a private company with the help of a CBR tool [3][20]. In most cases, these systems are very adjusted to work well in a specific domain and to change this domain will involve modifying the entire system. Therefore, they are not suitable for multi-domain environments, where the work domains are very dynamic and undergo many changes.

The Spanish company TISSAT S.A. [18] is a TMC that develops and integrates Information Technology and communication products, services and solutions for advanced Internet environments. TISSAT runs a call centre where its operators answer the queries of its customers. The company has developed a new helpdesk application to manage the call centre and to improve the weaknesses of the former system. With the deployment of this application TISSAT tries to avoid the problems that arise from using a system copyrighted by a private vendor, which cannot be modified and updated without the authorisation of the vendor, and to improve some weaknesses. One of the most important functionalities that the company wants the new helpdesk to incorporate is the possibility to store in an appropriate format the information about past

problems and the solution that was applied. This information will provide the operators with useful hints to solve new problems and, thus, to shorten their solving time. To implement this feature, a new CBR module has been developed. The CBR module had to be easily integrated in the existing helpdesk. Moreover, TISSAT works with very dynamic domains and a complete case-based system adapted to a specific domain and, thus, difficult to be updated to new domains is not suitable. To some extent, the aim of this research was also to check the suitability of CBR methodology and its easiness of integration for multi-domain environments.

The rest of the paper is structured as follows. Section 2 shows the environment where the CBR module has been integrated and better explains the motivation to develop this module. Section 3 describes the module and its operation. Finally, the CBR module is evaluated in section 4 and the conclusions of this research are summarised in section 5.

2 Background and Motivation

In a TMC, there are a number of technicians whose role is to provide the customers with technical assistance –microcomputing, security and network management among other services–. This help is commonly offered via a call centre. The staff of the TISSAT's call centre is divided into three levels:

- First level operators, who receive customer queries and answer those ones from which they have background training or their solution is registered in the company manuals of action protocols.
- Second level operators, who are the technicians in charge of solving the

problems that the first level operators could not solve.

- Chief technicians and administrators, who are in charge of organising working groups, of assigning problems to specic technicians and of creating generic solutions, which will be registered and used later by the operators of lower levels.

To guarantee a high-quality service, the company subscribes to a Service Level Agreements (SLAs) with the customer, where the different characteristics of the services to provide are specified –service descriptive labels that identify each request as belonging to a certain type (category trees), service priority, attention and assistance maximum times and certain parameters that measure the fulfilment degree of these services–. In case of breach of the agreements, the company is economically penalised.

Once the SLAs have been established, the customer can request the supply of the services that have been agreed by means of several entry channels –phone call, website, e-mail, post and fax–. When the centre receives the request, the so-called incidence register or *ticket* is generated with the customer data and a description of the incidence. From the customer point of view, the tickets are fundamentally characterised by their state –assigned, in progress, solved, closed, pending, require external provider or require software development– and by the problem-solving time, which allows the customer to know the degree of the agreements fulfilment. For the centre, the tickets are also characterised by other parameters, such as the type of the incidence (category), to which operator or group the ticket has been assigned, work-notes about the incidence, provider data or affected equipment (inventory). The problem-solving process generates more information that helps to explain the solution that has been applied–solving-method, operator level, keywords, URL's, at-

tached documents or observations–.

To date of the beginning of this project, TISSAT had used a helpdesk application to manage the big amount of information that processes its call centre. The basic functions of this application were the following:

- To register the incidences information: customer data, entry channel and project (identifying the customer and the specific service that is being used).
- To track each ticket and to scale it from one operator level to a more specialised one.
- To warn when the maximum time to solve an incidence is about to expire.
- To search for past registered solutions given a set of words. It returns the solutions containing those words or any of them.

However, the former helpdesk had several weaknesses that posed important problems and hindered the improvement of the platform:

- The software was copyrighted by other company. Therefore, TISSAT had to pay licenses and could not make any change in the application.
- There were many integration and access difficulties with other software products and mobile devices.
- The information about past problems was not stored in a suitable format (using manuals or hand-written notes, most of them out-of-date). Therefore, the information transfer between operators was not fluent.
- There was not a centralised database of successfully applied solutions. The operators lost time in solving problems that had been previously solved by other operator. The company lost time and money in training new operators or re-training operators for solving new problems when a new service was offered. In addition, the knowledge of experienced operators should be

registered in some way to avoid missing it when the operators leave the company.

In order to cope with these problems, TISSAT has developed the new helpdesk tool I2TM (**I**ntelligent and **I**ntegrated **T**icketing **M**anager). This application improves the operation of the former helpdesk and, thus, the quality of the customer support. In addition, some research has been done with the aim of predicting the number of forthcoming service requests as well as the time when they will occur in order to optimise the resources used to solve these events [17]. Simultaneously, a system able to propose suitable solutions to help the operators has been developed. This system will shorten the problem-solving time. The CBR-TM system (**C**ase-**B**ased **R**easoning for **T**icketing **M**anagement) implemented with this aim stores and reuses in an efficient and intelligent way the information about past problems and their solutions. Therefore, it is mainly addressed to solve the last two weaknesses of the above list. CBR-TM was developed as flexible as possible in order to ease its adaption to work with the data of any new project that the company may manage in the future.

3 The CBR-TM module, Case-Based Reasoning for Ticketing Management

CBR-TM acts as a separate module of I2TM for solution-advising. Both systems communicate and synchronise their data by means of webservice calls. In this way, their architectures are independent and it is possible to make changes in one of them without affecting the other.

Moreover, CBR-TM implements each phase of the common CBR reasoning

cycle –Retrieve, Reuse, Revise and Retain [1]– using an independent *plugin-algorithm*. Therefore, the system is flexible and portable and it is possible to modify or even add an algorithm whenever it will be necessary. The specific algorithm that has to be used in each phase is configured in a XML configuration file. CBR-TM also includes a *plugin* to store algorithm variables among different executions of the system. In this way, some algorithm specific features, such as weights and constants, can be saved when the system changes the algorithm that implements a reasoning phase in the configuration file.

3.1 Data format

The first step to design the CBR-TM module was to decide the system data format. For it, the old call centre tickets database was analysed. Each register of the database contained information of an old problem. As a result of this process some weaknesses were encountered. On one hand, the database was very imbalanced and most registers represented actually the same problem and, on the other hand, in most cases there were not any information about the problem solution. In view of the above, to structure the cases of CBR-TM as the prototyped representation of a set of tickets was adopted as design decision. The module was developed using an object-oriented approach where the cases are serialisable objects. Figure 1 shows an overview of the data structure in I2TM and CBR-TM.

The I2TM application maintains a *Typification Tree* that organises hierarchically the taxonomy of problem types (i.e. categories) from less to more generic problems. These categories identify each ticket as belonging to a certain type of incidence, depending on the project to which the ticket is associated. There-

fore, the first level nodes of the tree represent the projects that the company manages, and the nodes below them represent the possible problems that the customers of each project can request for solving to the operators. When a change in the tree is done or a new project is supported by the company, CBR-TM is able to read again the tree and re-synchronise itself with the I2TM system data. In this way, CBR-TM is able to work in the multiple domains of the company projects.

A case in CBR-TM, thus, is an object that has an identifier pointing to the *category* that represets the type of problem to which the case belongs. The case also has a set of *attributes* depending on its category. These attributes are the answers of some questions that the operator asks to the customer when he performs a new request. The questions are created and maintained by domain experts and give more information about the customer request.

Finally, a case has associated one or more solution identifiers pointing to the solution database of TISSAT. A solution can also be associated to more than one case. The solution types are very diverse, from attached documents explaining the steps followed to solve the problem, to websites or manuals. Each solution of a case also has a suitability degree to solve that specific problem.

3.2 CBR-TM Reasoning Cycle

When a customer asks an operator for solving a request, the operator can either try to solve the problem by his own or to use CBR-TM for giving a piece of advice about the solution to apply. In that case, once the operator has created the new ticket using I2TM, the application transfers the ticket information to

CBR-TM. From that moment, the CBR-TM module starts its reasoning cycle and tries to give a solution to the operator. Figure 2 shows an overview of the dataflow between I2TM and CBR-TM during the problem-solving process.

Retrieval Phase. The first step when CBR-TM is asked for searching a solution for the current ticket is to retrieve a set of similar cases from its case-base. Three *plugin-algorithms* implement this retrieval stage. The *Indexer* algorithm starts the process organising hierarchically the case-base to ease the case retrieval. In the current implementation, the operators carry out this function by categorising by hand the ticket. In this way, the module profits by the operators expert knowledge to speed up the problem-solving process. After that, the *Mapper* algorithm (see algorithm 1) searches the *Typification Tree* for retrieving the upper categories that are in the same branch as the ticket category. Those categories represent more generic problems, but they may also be related with the current problem and thus, their solutions may also be suitable to solve it. Once this set of categories is selected, the *Mapper* algorithm retrieves from the case-base the cases that are categorised by them. Finally, the *Similarity* algorithm sorts the set of retrieved cases by their degree of similarity with the ticket.

The similarity computation in CBR-TM is hindered by the heterogeneity of its cases. Note that cases with different categorisations can share some attributes and also have different ones. In addition, there are several possible attribute types (numeric, string, enumerated and boolean) and the attributes can also have missing values. In order to work with heterogeneous cases some known similarity metrics an measures have been adapted and implemented. The similarity metrics, which are shown in equation (1), calculate local dis-

Algorithm 1 *MapperAlgorithm*

Require: Typification Tree, Ticket

1: $CategorySet = Ticket.Category$

2: $Cases = \emptyset$

3: $node = Ticket.Category$

4: **while** $node.parent \neq \emptyset$ **do**

5: $\quad CategorySet \leftarrow CategorySet \cup node.parent$

6: $\quad node \leftarrow node.parent$

7: $Cases = Retrieve(CategorySet)$ //Retrieve Cases from case-base where case.category ∈ CategorySet

tances between a pair of attributes i and j.

$$distance(i,j) = \begin{cases} if\ i,j\ \in\ \Re, & |i-j| \\ if\ i,j\ \in\ string, & Levenshtein(i,j) \\ if\ i,j\ \in\ enumerated, & dist(i,j) \\ otherwise, & 1 \end{cases} \quad (1)$$

Specifically, the absolute difference is used for computing the distance between numeric values and the *Levenshtein* distance for strings. The distance between enumerated values can be configurable to different values depending on the application domain ($dist(i,j)$). Since all distances are normalised in a 0 to 1 range, the local distance is set to 1 when there is a missing value in an attribute. Note that, therefore, the distance between two different boolean values are also set to 1.

Next, the similarity measures combine the local distances to calculate the

global distance between cases. Finally, the set of retrieved cases is sorted using a *k-nearest neighbour* algorithm. Several similarity measures based on the *Euclidean* distance (varying the importance assigned to the cases attributes) and one similarity measure based on the ratio model proposed by *Tversky* [19] have been adapted in the current implementation of CBR-TM as shown in equations (2) and (3)):

$$EuclideanSimilarity(a,b) = \frac{1}{1+\sqrt{\sum_{i=1}^{N} w_i^2 distance(a_i,b_i)^2}} \tag{2}$$

where a and b are two cases of the CBR-TM case-base and $w_i \in [0,1]$ is a weight assigned to each attribute i of the cases in order to indicate its importance.

$$TverskySimilarity(a,b) = \frac{\alpha(\#commonAt)}{\alpha(\#commonAt)+\beta(\#differentAt)} \tag{3}$$

where $\#commonAt$ and $\#differentAt$ represent the number of similar and different attributes of a pair of cases and α and β are the corresponding weights assigned to each group.

Reusage Phase. Once the set of cases that are similar to the current ticket is selected, the *SolutionSelection* plugin algorithm proposes a list of possible alternatives to solve the current problem. The algorithm proposes first the solutions of the most similar case to the ticket, sorted by their degree of suitability to solve the specific problem that represents that case. If desirable, the algorithm can also propose the solutions of the second most similar case and so on. The number of solutions to propose is configurable.

Revision and Retention Phases. The revision and retention stages in CBR-TM are very related. In fact, the retention stage can be viewed as a consequence of the revision stage. The *Rewarder* plugin algorithm implements both phases (see algorithm 2). Each time a problem that was requested to I2TM is solved, the customer must report to the system his degree of satisfaction with the solution proposed. Then, the solution and the ticket that was created to represent the incidence are appraised. The category that was initially assigned to the ticket is revised by domain experts and changed if necessary before ending the service and closing the ticket. The final category is sent to CBR-TM together with the ticket. The system always reports to the CBR-TM module the tickets that have been solved, even when the operator solved the problem directly and without asking to the module for a solution proposal. With this process CBR-TM improves its performance and corrects its mistakes and, thus, increments quickly its knowledge about the domain.

The revision stage starts again the reasoning cycle in CBR-TM, in order to discover if the module has a case in its case-base that coincides with the problem that the ticket represents. CBR-TM repeats the case retrieval stage for each ticket that is closed, instead of keeping in memory the case that was retrieved as the most similar case when the ticket was transferred to the module. Note that, on one hand, it may occur that the ticket had never been reported to CBR-TM, but solved directly by the operator. On the other hand, CBR-TM may have proposed an invalid solution without making any mistake, since this error may come from an erroneous manual categorisation. In addition, the CBR-TM module is a recommender system, but the operator can choose any different solution to apply to the problem based on his experience.

Algorithm 2 *RewarderAlgorithm*

Require: Solved Ticket T

1: $C = mapper(T)$ //Retrieve the set of cases from the case-base that could be related with similar problems

2: $similarity(C, T)$ //Calculate the similarity between each case $c \in C$ and T

3: **if** $\exists\ c \in C\ /\ similarity(c, T)\ >\ threshold$ **then**

4: **if** $\exists\ s \in c.solutionList\ /\ s = T.solution$ **then**

5: $s.suitability++$

6: **else**

7: $c.solutionList \leftarrow c.solutionList \cup s$

8: **else**

9: $CaseBase \leftarrow CaseBase \cup newCase(T)$

With this revision phase, the module is able to learn from the operator expert knowledge.

If CBR-TM does not find a similar enough case in its case-base for a ticket that has been solved, it creates a new case. The similarity threshold to decide when a new case must be created is configurable. If there is a similar case in the case-base and the solution applied to the ticket is already associated with that case, the suitability degree of the solution is increased. Otherwise, that solution is added to the list of solutions of the case. To modify the suitability degree of the solutions decreases the possibility of proposing obsolete solutions. To date, there is not any adaptation of solutions to fit the current problem, but the solutions are proposed to solve the problem without changes.

The CBR-TM module has been designed to be able to work with multiple domains, to support multiple requests and to be easily integrable with any existing helpdesk software of the company. Each call to the module opens a new execution thread. Moreover, the communication between I2TM and CBR-TM is synchronous and the system always waits for an answer of the module. When CBR-TM has not a similar case in its case-base, it replies a void answer. A *timeout* to interrupt the communication when the maximum time to give an answer is exceeded must also be specified. Some important design decisions are detailed as follows.

Webservice. Following the scalability and flexibility guidelines, the communication between CBR-TM and I2TM is done through webservice calls. The main webservice calls are the following.

- The request for a solution given a ticket (getSolutions webservice call): It is the call that starts the reasoning cycle of CBR-TM when an operator asks the module for possible solutions to the problem that has been requested. The call is used to transfer the information about the current ticket (features and category) from the I2TM helpdesk system to the CBR-TM module.
- The feedback produced when a ticket is solved (closeQuestion webservice call): This call is used by I2TM to notify the final solution of each ticket to the CBR-TM module. The parameters of the call are the ticket features, the category and the solution adopted (note that the ticket could have not been requested to CBR-TM before). With this information the CBR-TM module

is able to correct its wrong proposals and increment quickly its knowledge about the domain. In any case, the solutions that the module proposed for a ticket are not reviewed until the feedback call is completed.

- The case-base maintenance: The maintenance calls allow the I2TM system to stop CBR-TM for maintenance, pause it and restore the CBR-TM status. They are also used to notify to the CBR-TM module modifications of the I2TM databases –move a category to a different location inside the category tree, add a category or even activate or deactivate a whole project, which would change an entire branch in the category tree.

Synchronisation and Cache policies. Accessing the case-base of CBR-TM can be very expensive in terms of time. A cache system was developed to facilitate this task, loading in a quick access memory those cases that have been recently used or with a higher probability to be used. The cache system was designed as a plugin, and can be changed on-line. The system operates as following.

When a ticket is requested to the CBR-TM module, the *Mapper* algorithm searches in the *Typification Tree* the category node of the ticket and the set of upper categories that are in the same branch. Then, the algorithm tries to retrieve first the cases that belong to these categories from the cache memory. If such cases are not there, the cache system loads them from the CBR-TM case-base. Note that the system loads all cases that belong to a category and, each time a new case is created the module adds first the case to the cache memory. The cases are finally stored in the case-base depending on each cache policy. By default, all policies update the case-base when the CBR-TM operation is interrupted. A maximum time to load the cases in the case-base can also be specified.

In the current implementation of CBR-TM, the cache policy is based on the frequency of data access. When the cache memory exceeds a specified amount of memory used, CBR-TM starts to unload the memory by storing in the case-base those cases that belong to the categories with a lower rate of utilisation. Again, the plugin architecture of CBR-TM allows to implement and use any policy specifically helpful in a particular customer support scenario.

4 CBR-TM Evaluation

Two test software tools have been developed to check the performance of the CBR-TM module. Both tools use webservice calls to communicate with the CBR system.

The *Load Tool* tests the system strongness when dealing with the simultaneous requests of different number of customers. The tool use a thread to represent each customer. Therefore it is possible to simulate the simultaneous access of several customers to the CBR-TM module. The number of customers accessing to the system, the number of requests per customer and the statistical distribution –Random Normal distribution or Uniform Random distribution– which determines the time that the customer waits in between two requests are configurable. The client also shows, if desirable, the specific solutions proposed by CBR-TM, the average answer time to solve a request or to notify a new ticket that has been solved and their standard deviations.

Several executions of this tool varying its parameters have to be done in order to test the robustness of CBR-TM. The results of those executions are revised after to elaborate evaluation graphs.

The *Test Tool* checks the system correctness and assesses its performance. This tool can do several types of tests based on *cross-partition* and *leaving-one-out* techniques to appraise some aspects of the system, such as reliability, consistency and duplicity, among others.

Using these tools some verification tests have been performed. For this purpose, a synthetic database (extracted from the tickets database) with a total amount of 360 tickets in the domain of computer errors has been used. Each one of them represents a computer error problem which can be requested to be solved by the system and its features. Note than an entry in this database is not equivalent to a case, since, as it is pointed out before, a case is the prototyped representation of a set of tickets with the same features and the same solution applied satisfactorily to solve the problem that those tickets represent.

Several scripts to modify and evaluate the database are also implemented. By means of them, a specified percentage of noise or loss in the data of the original database has been introduced. In this way, the operation of the system when there are void attributes (not answered) or printing errors has been checked.

4.1 Test Results

CBR-TM has been checked from the following points of view [20]:

(1) Verification: the accuracy of the solutions offered by the system has to be tested. To do it it is necessary to verify the non-existence of:
   - Duplicate cases.
   - Inconsistencies: cases that lead to contradictory conclusions.

- Omissions: queries that do not match any case in the case-base.
- Isolations: cases that are never retrieved.

(2) Validation: the suitability of the system to solve the specified problem has also to be tested.

TISSAT was interested in checking the behaviour of some of the similarity measures implemented in a particular computer error domain, which is related with a project that is currently managed by the company. Therefore, the tests were repeated setting the system to work with a different similarity measure each time.

First, the **precision** of the system in retrieving the cases stored in the case-base when a new problem is presented was checked using the *Test Tool*. Since the *k-nearest neighbour* algorithm is used to select the most similar stored case, a case must fit exactly with itself (100% rate of similarity). To perform this test all the tickets of the database were requested to CBR-TM in order to create the corresponding cases and, once they were loaded in the case-base, each ticket was used again to carry out a new request to the system. As all the tickets have been processed already by the CBR system, the requests must retrieve exactly the case that was created the first time that each ticket was requested with a 100% rate of similarity.

Another characteristic that must be checked if the *k-nearest neighbour* algorithm is used is the **consistency** of the case-base. Therefore, two equal requests must retrieve the same cases from the case-base with the same similarity rates. The same process described above was followed to load the case-base of CBR-TM. In order to test this condition, each ticket of the database was used to perform two identical requests.

Finally, the existence of **duplicities** in the case-base that the system creates has to be ruled out. This fact does not harm the system operation, but it could cause some problems, such as to slow down the response time of the system. For this purpose, the *Test Tool* was used again to load the case-base with the tickets database in the same way as it was done in the other tests. After that, each ticket was used again as a new request to the CBR-TM module. If the same database has been used to create the case-base and to perform the requests and the system does not produce duplicated cases, it can not retrieve more than one case from the case-base with a similarity rate of 100%.

In all these tests, the system behaved as expected with all the similarity measures tested.

Once it was verified that the system implemented performs well and gives correct answers, its behaviour when some errors or omissions in the data are introduced was tested by using the *Test Tool*. In order to check the system ability managing **noisy or lost data in the requests**, the case-base was loaded with the original tickets database and, after that, it was disturbed for a rate of 10%, 20%, 40% and 60% of the total number of features stored. Therefore, the corresponding amount of features were changed by introducing a random value from the range of the possible values for each feature (noise) or deleting the value (loss). Finally, this disturbed database was used to perform the requests to the CBR system and to compute the classification errors. Note that it is considered an error when a disturbed request does not retrieve the same case than the original request. The results obtained setting the system to work with the different similarity measures implemented are shown in figures 3 and 4. As it can be appreciated, CBR-TM manages well corrupt data and the mean error in the answers that the module provides continues being low.

The behaviour of our CBR system when the **case-base** itself is **disturbed with noisy or lost data** was also tested. Thus, the tickets database was disturbed for a rate of 10%, 20%, 40% and 60% of the total number of features stored changing or deleting their values as it is explained above. In this case, the *Test Tool* was used to load the disturbed database and to test the module with requests coming from the original tickets database. The results obtained are shown in figures 5 and 6.

The figures show that CBR-TM performs well with noisy or lost data in the case-base, although, as it was expected, the corrupt data reduces its efficiency.

Finally, the *Load Tool* was used to test the system performance. This performance could be influenced by the number of tickets reported to CBR-TM or the number of customers performing requests simultaneously. As it can be appreciated in figure 7, as the amount of tickets that have been requested to CBR-TM increases, the mean error in the answers that the system provides decreases. This fact shows that the system knowledge goes up quickly as the amount of processed data increases.

Figure 8 analyses the CBR-TM response time when the load of customers making simultaneous requests increases. CBR-TM is able to give a quick answer even when the number of simultaneous requests is large.

Regarding to the behaviour of the similarity measures tested, the results of the figures show that they behave in a very similar way in the computer error domain, with a little advantage of the *Tversky-based* similarity measure when the data is disturbed.

5 Conclusions

We have developed a modular and flexible CBR system called CBR-TM. The system helps the problem solving process done by the operators of a helpdesk application in a customer support environment. CBR-TM has been implemented as a module for an existing helpdesk application. It has been developed as generic as possible. This fact allows quickly updating the CBR module to any software or technology change in the helpdesk system. The CBR-TM module has been integrated successfully into the core of the I2TM helpdesk application of the Spanish company TISSAT S.A.

The CBR-TM module behaves well when it attends to the requests of simultaneous customers. The accuracy of CBR-TM in offering solutions improves as the size of its case-base increases. Therefore the system learns properly the new solutions created by the helpdesk operators. The tests performed show also the robustness of the system in presence of corrupt data.

Although in this paper the CBR-TM module has been tested in a specific helpdesk working in the domain of computer errors in public administration organisms and private companies, it could be easily adapted to work with different domains. Moreover, this system is recently implanted and more intensive research to improve the algorithms and the techniques applied will be done. One of the main interests is to change the manual categorisation into an automatic one. This would prevent CBR-TM from the mistakes that the operators can make. Anyway, the approach that has been taken in this research was to adapt generic algorithms that could be suitable, in principle, to any customer support domain. The flexibility of the CBR-TM module architecture

allows to add any new algorithm that can fit better the purposes of a specific helpdesk.

References

[1] Aamodt, A., & Plaza, E. (1994). Case-based reasoning: foundational issues, methodological variations and system approaches. AI Communications, 7(1), 39-59.

[2] Acorn, T., & Walden, S. (1992). SMART: Support Management Automated Reasoning Technology for Compaq Customer Service. In: Proc. ITS'92, vol. 4, AAAI Press, (pp. 3-18), Berlin.

[3] Althoff, K-D., Auriol, E., Barletta, R., & Manago, M. (1995). A Review of Industrial Case-Based Reasoning Tools, AI Perspectives Report, AI Intelligence, Oxford OX2 7XL.

[4] eGain (2008). www.egain.com.

[5] Empolis Knowledge Management GmbH - Arvato AG (2008). http://www.empolis.com/.

[6] Goker, M., & Roth-Berghofer, T. (1999). The development and utilization of the case-based help-desk support system HOMER. Engineering Applications of Articial Intelligence, 12(6), 665-680.

[7] Bergmann, R., Althoff, K-D., Breen, S., Goker, M., Manago, M., Traphoner, R., & Wess, S. (2003). Developing Industrial Case-Based Reasoning Applications, The INRECA Methodology (2nd ed.). Lecture Notes in Articial Intelligence, vol. 1612, Springer-Verlag.

[8] Kaidara Software Corporation (2008). http://www.kaidara.com/.

[9] Kang, B.H., Yoshida, K., Motoda, H., & Compton, P. (1997). Help Desk System with Intelligent Interface. Applied Articial Intelligence, 11(7-8), 611-631.

[10] Kolodner, J. (1993). Case-based Reasoning. Morgan Kaufmann Publishers.

[11] Kriegsman, M., & Barletta, R. (1993). Building a Case-Based Help Desk Application. IEEE Expert: Intelligent Systems and Their Applications, 8(6), 18-26.

[12] Raman, R., Chang, K.H., Carlisle, W.H., & Cross, J.H. (1996). A self improving helpdesk service system using case-based reasoning techniques. Computers in Industry, 2(30), 113-125.

[13] Roth-Berghofer, T., & Iglezakis, I. (2000). Developing an Integrated Multilevel Help-Desk Support System. In: Proc. GWCBR'00, DaimlerChrysler, Research and Technology, FT3/KL, (pp. 145-155).

[14] Roth-Berghofer, T. (2004). Learning from HOMER a case-based helpdesk support system. Advances in Learning Software Organizations, Springer-Verlag, 88-97.

[15] Shimazu, H., Shibata, A., & Nihei, K. (1994). Case-Based Retrieval Interface Adapted to Customer-Initiated Dialogues in Help Desk Operations. In: Proc. AAAI'94, vol. 1, AAAI Press, (pp. 513-518).

[16] Simoudis, E. (1992). Using Case-Based Retrieval for Customer Technical Support. IEEE Intelligent Systems, 7(5), 10-12.

[17] Soria, E., Balaguer, E., Palomares, A., & Martn, J. (2006). Predicting Service Request in Support Centers based on Nonlinear dynamics, ARMA Modelling and Neural Networks. Expert Systems with Applications, vol. 34(1), 665-672.

[18] Tissat S.A (2008). www.tissat.es.

[19] Tversky, A. (1997). Features of similarity. Psychological Review, 84(4), 327-352.

[20] Watson, I. (1997). Applying Case-Based Reasoning, Techniques for Enterprise Systems. Morgan Kaufmann Publishers Inc.

Figure Captions

Figure 1: Overview of the data structure in I2TM and CBR-TM.

Figure 2: System Operation.

Figure 3: CBR-TM ability to manage noisy data in the requests features.

Figure 4: CBR-TM ability to manage lost data in the requests features.

Figure 5: CBR-TM ability to manage noisy data in the case-base.

Figure 6: CBR-TM ability to manage lost data in the case-base.

Figure 7: Influence of the processed data on the CBR-TM performance.

Figure 8: Influence of the amount of simultaneous customers on the CBR-TM performance.