A Contingency Response Multi-agent System for Oil Spills

Aitor Mata, Dante I. Tapia, Angélica González, and Belén Pérez

Departamento Informática y Automática
Universidad de Salamanca
Plaza de la Merced s/n, 37008, Salamanca, Spain
University of Salamanca, Spain
{aitor,dantetapia,angelica,lancho}@usal.es

Abstract. This paper presents CROS, a contingency response multi-agent system for oil spills situations. The system makes use of a Case-Based Reasoning system which generates predictions to determine the probability of finding oil slicks in certain areas of the ocean. CBR uses past information to generate new solutions to the current problem. The system employs a distributed multi-agent architecture so that the main components of the system can be accessed remotely. Therefore, all functionalities can communicate in a distributed way, even from mobile devices. The core of the system is a group of deliberative agents acting as controllers and administrators for all functionalities. The system has been used to predict real oil spill situations. Results have demonstrated that the system can accurately predict the presence of oil slicks in determined zones. It has been demonstrated that using a distributed architecture can enhance the overall performance of the system.

Keywords: Oil Spill, Multi-Agent Systems, Case-Based Reasoning, Distributed Architectures.

1 Introduction

The response to minimize the environmental impact when an oil spill is produced must be precise, fast and coordinated. The use of contingency response systems can facilitate the planning and tasks assignation when organizing resources, especially when multiple people are involved.

This paper presents CROS, a contingency response multi-agent system for helping manage these situations. This system deploys a prediction model which makes use of intelligent agents and Case-Based Reasoning systems to determine the possibility of finding oil slicks in a certain area of the ocean. It also applies a distributed multi-agent architecture based on Service Oriented Architectures (SOA), modeling most of the system's functionalities as independent applications and services. These functionalities are invoked by deliberative agents acting as coordinators.

Agents and multi-agent systems have been successfully applied to several scenarios, such as education, culture, entertainment, medicine, robotics, etc. [1]. Agents have a set of characteristics, such as autonomy, reasoning, reactivity, social abilities, pro-activity, mobility, organization, etc. which allow them to cover several needs for developing contingency response systems [2].

Y. Demazeau et al. (Eds.): 7th International Conference on PAAMS'09, AISC 55, pp. 274–283.
springerlink.com

Predicting the behavior of oceanic elements is a quite difficult task. In this case, the prediction is related with external elements (oil slicks) and this makes the prediction even more difficult. The open ocean is a highly complex system that may be modeled by measuring different variables and structuring them together. Some of these variables are essential to predict the behavior of oil slicks. It is necessary to know the previous positions of oil slicks in order to predict the future presence in a specific area. That knowledge is provided by the analysis of satellite images which reveal the precise position of the slicks.

The system presented in this paper generates as a solution a probability of finding oil slicks for different geographical areas after an oil spill. Predictions are created using a Case-Based Reasoning system. The cases used by the CBR system contain information about the oil slicks (size and number) and atmospheric data (wind, ocean currents, salinity, temperature, height and pressure). CROS combines artificial intelligence techniques in order to improve the efficiency of the CBR system, thus generating better results. CROS has been trained using historical data acquired during the Prestige oil spill at the Galician west coast in Spain, from November 2002 to April 2003. Most of the data used by CROS has been acquired from the ECCO (Estimating the Circulation and Climate of the Ocean) consortium [3]. Position and size of the slicks has been obtained by treating SAR (Synthetic Aperture Radar) satellite images [4]. The development of agents is an essential piece in the analysis of data from distributed sensors and gives those sensors the ability to work together and analyze complex situations.

Next, the oil spill problem is presented showing the difficulties and the possibilities of finding solutions to this problem. Afterwards, the main components of the system, including its architecture are described. Finally, the results and conclusions are presented.

2 Different Approaches to the Oil Spill Problem

It is very important to determine if an area will be contaminated or not after an oil spill. To do so, it is necessary to know how the slicks generated by the spill behave for concluding about the presence of contamination in a specific area. First, position, shape and size of the oil slicks must be identified. One of the most precise ways to acquire that information is by using satellite images. SAR (*Synthetic Aperture Radar*) images are the most commonly used to automatically detect this kind of slicks [5]. Satellite images show certain areas where it seems to be nothing (e.g. zones with no waves) as oil slicks. Figure 1 shows a SAR image on the left side, which displays a portion of the Galician west coast with black areas corresponding to oil slicks. On the right side of Figure 1 an interpretation of the SAR image after treating the data is shown. SAR images make it possible to distinguish between normal sea variability and oil slicks. It is also important to make a distinction between oil slicks and look-alikes. Oil slicks are quite similar to quiet sea areas, so it is not always easy to discriminate between them. This can lead to mistakes when trying to differentiate between a normal situation and an oil slick. This is a crucial aspect in this problem that can be automatically managed by computational tools [6]. Once the slicks are correctly identified, it is also crucial to know the atmospheric and maritime situation

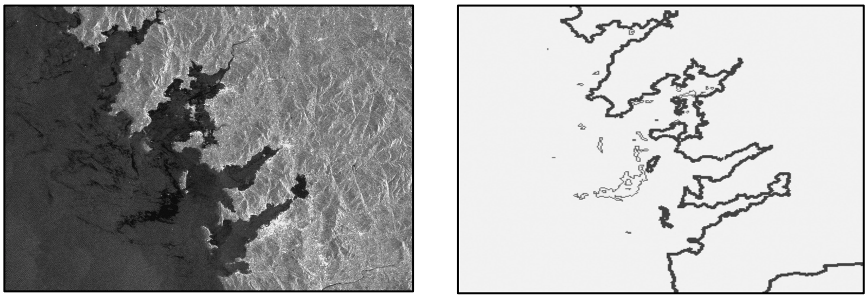

Fig. 1. Satellite image of an oil spill near the Galician west coast in Spain (left) and its interpretation by the CROS system (right)

that is affecting the zone at the moment that is being analyzed. Information collected from satellites is used to obtain the atmospheric data needed. That is how different variables such as temperature, sea height and salinity are measured in order to obtain a global model that can explain how slicks evolve.

There are different ways to analyze, evaluate and predict situations after an oil spill. One approach is simulation, where a model of a certain area is created introducing specific parameters (weather, currents and wind) and working along with a forecasting system. Using simulations it is easy to obtain a good solution for a certain area, but it is quite difficult to generalize in order to solve the same problem in related areas or new zones. Nevertheless arriving at these kinds of solutions requires a great data mining effort. Different techniques have been used to achieve this objective, from fuzzy logic to negotiation with multi-agent systems. One of these techniques is Case-Based Reasoning which is described in the next section.

3 CROS: A Contingency Response Multi-agent System for Oil Spill

CBR has already been used to solve maritime problems in which different oceanic variables were involved [7]. In CROS, the data collected from different satellites is processed and structured as cases. Table 1 shows the main variables that defines a case. Cases are the key to obtain solutions to future problems through a CBR system. The functionalities of CROS can be accessed using different interfaces executed on PCs or PDAs (Personal Digital Assistant). Users can interact with the system by introducing data, requesting a prediction or revising a solution generated (i.e. prediction). The interface agents communicate with the services through the agents' platform and vice versa.

The interface agents perform all the different functionalities which users can make use for interacting with CROS. The different phases of the CBR system have been modeled as services, so each phase can be requested independently. For example, one user may only introduce information in the system (e.g. a new case), while another user could request a new prediction.

Table 1. Variables that define a case

Variable	Definition	Unit
Longitude	Geographical longitude	Degree
Latitude	Geographical latitude	Degree
Date	Day, month and year of the analysis	dd/mm/yyyy
Sea Height	Height of the waves in open sea	m
Bottom pressure	Atmospheric pressure in the open sea	Newton/m^2
Salinity	Sea salinity	ppt (parts per thousand)
Temperature	Celsius temperature in the area	ºC
Area of the slicks	Surface covered by the slicks present in the analyzed area	Km^2
Meridional Wind	Meridional direction of the wind	m/s
Zonal Wind	Zonal direction of the wind	m/s
Wind Strength	Wind strength	m/s
Meridional Current	Meridional direction of the ocean current	m/s
Zonal Current	Zonal direction of the ocean current	m/s
Current Strength	Ocean current strength	m/s

All information is stored in the case base and CROS is ready to predict future situations. A problem situation must be introduced in the system for generating a prediction. Then, the most similar cases to the current situation are retrieved from the case base. Once a collection of cases are chosen from the case base, they must be used for generating a new solution to the current problem. Growing Radial Basis Functions Networks [8] are used in CROS for combining the chosen cases in order to obtain the new solution.

CROS determines the probability of finding oil slicks in a certain area. CROS divides the area to be analyzed in squares of approximately half a degree side for generating a new prediction. Then, the system determines the amount of slicks in each square. The squares are colored with different gradation depending on the quantity of oil slicks calculated.

Figure 2 shows the structure of CROS. There are four basic blocks in CROS: Applications, Services, Agent Platform and Communication Protocol. These blocks provide all the system functionalities:

Applications. These represent all the programs that users can use to exploit the system functionalities. Applications are dynamic, reacting differently according to the particular situations and the services invoked. They can be executed locally or remotely, even on mobile devices with limited processing capabilities, because computing tasks are largely delegated to the agents and services.

Services. These represent the activities that the architecture offers. They are the bulk of the functionalities of the system at the processing, delivery and information acquisition levels. Services are designed to be invoked locally or remotely. Services can be organized as local services, web services, GRID services, or even as individual stand alone services. CROS has a flexible and scalable directory of services, so they can be invoked, modified, added, or eliminated dynamically and on demand. It is absolutely necessary that all services follow a communication protocol to interact with the rest of the components.

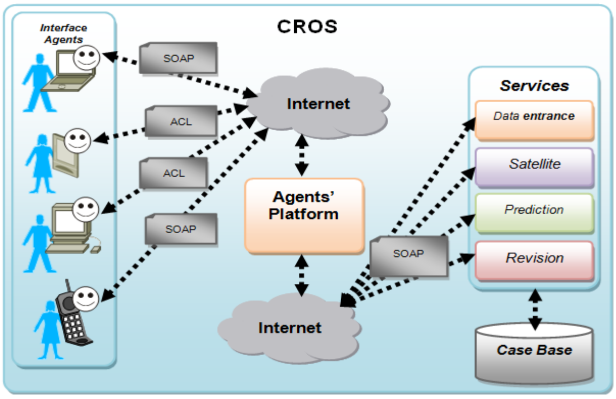

Fig. 2. CROS structure

***Agent Platform*.** This is the core of the system, integrating a set of agents, each one with special characteristics and behavior. An important feature in this architecture is that the agents act as controllers and administrators for all applications and services, managing the adequate functioning of the system, from services, applications, communication and performance to reasoning and decision-making. In CROS, services are managed and coordinated by deliberative BDI agents. The agents modify their behavior according to the users' preferences, the knowledge acquired from previous interactions, as well as the choices available to respond to a given situation.

***Communication Protocol*.** This allows applications and services to communicate directly with the Agents Platform. This protocol is based on SOAP specification to capture all messages between the platform and the services and applications [9]. Services and applications communicate with the *Agents Platform* via SOAP messages. A response is sent back to the specific service or application that made the request. All external communications follow the same protocol, while the communication among agents in the platform follows the FIPA Agent Communication Language (ACL) specification.

Agents, applications and services in CROS can communicate in a distributed way, even from mobile devices. This makes it possible to use resources no matter its location. It also allows the starting or stopping of agents, applications, services or devices separately, without affecting the rest of resources, so the system has an elevated adaptability and capacity for error recovery. Users can access to CROS functionalities through distributed applications which run on different types of devices and interfaces (e.g. computers, PDA).

Interface Agents are a special kind of agents in CROS designed to be embedded in users' applications. These agents are simple enough to allow them to be executed on mobile devices, such as cell phones or PDAs because all high demand processes are delegated to services. CROS defines three different *Interface Agents*:

CROS also defines three different services which perform all tasks that the users may demand from the system. All requests and responses are handled by the agents. The requests are analyzed and the specified services are invoked either locally or remotely. Services process the requests and execute the specified tasks. Then, services send back a response with the result of the specific task. In this way, the agents act as interpreters between applications and services in CROS. Next, CBR system used in CROS is explained.

3.1 Data Input Service

When data about an oil slick is introduced, CROS must complete the information about the area including atmospheric and oceanic information: temperature, salinity, bottom pressure, sea height. CROS uses Fast Iterative Kernel PCA (FIKPCA) which is an evolution of PCA [10]. This technique reduces the number of variables in a set by eliminating those that are linearly dependent, and it is quite faster than the traditional PCA. To improve the convergence of the Kernel Hebbian Algorithm used by Kernel PCA, FIK-PCA set η_t proportional to the reciprocal of the estimated values. Let $\lambda_t \in \mathfrak{R}^r{}_+$ denote the vector of values associated with the current estimate of the first *r* eigenvectors. The new KHA algorithm sets de i^{th} component of η_t to the files.

$$[\eta_t]_i = \frac{1}{[\lambda_t]_i}\frac{\tau}{t+\tau}\eta_0 \ , \tag{1}$$

When introducing the data into the case base, Growing Cell Structures (GCS) [11] are used. GCS can create a model from a situation organizing the different cases by their similarity. If a 2D representation is chosen to explain this technique, the most similar cells (i.e. cases) are near one of the other. If there is a relationship between the cells, they are grouped together, and this grouping characteristic helps the CBR system to recover the similar cases in the next phase. When a new cell is introduced in the structure, the closest cells move towards the new one, changing the overall structure of the system. The weights of the winning cell ω_c, and its neighbours ω_n, are changed. The terms ε_c and ε_n represent the learning rates for the winner and its neighbors, respectively. x represents the value of the input vector.

$$\omega_c(t+1) = \omega_c(t) + \varepsilon_c(x - \omega_c) \tag{2}$$

$$\omega_n(t+1) = \omega_n(t) + \varepsilon_n(x - \omega_n) \tag{3}$$

Once the case base has stored the historical data, and the GCS has learned from the original distribution of the variables, the system is ready to receive a new problem. When a new problem comes to the system, GCS are used once again. The stored GCS behaves as if the new problem would be stored in the structure and finds the most similar cells (cases in the CBR system) to the problem introduced in the system. In this case, the GCS does not change its structure because it has being used to obtain the most similar cases to the introduced problem. Only in the retain phase the GCS changes again, introducing the proposed solution if it is correct.

3.2 Prediction Generation Service

When a prediction is requested by a user, the system starts recovering from the case base the most similar cases to the problem proposed. Then, it creates a prediction using artificial neural networks. Once the most similar cases are recovered from the case base, they are used to generate the solution. Growing RBF networks [12] are used to obtain the predicted future values corresponding to the proposed problem. This adaptation of the RBF networks allows the system to grow during training gradually increasing the number of elements (prototypes) which play the role of the centers of the radial basis functions. The creation of the Growing RBF must be made automatically which implies an adaptation of the original GRBF system. The error for every pattern is defined by (4).

$$e_i = {}^{l}/_{p^*} \sum_{k=1}^{p} \left|\left|t_{ik} - y_{ik}\right|\right| , \tag{4}$$

Where t_{ik} is the desired value of the k_{th} output unit of the i_{th} training pattern, y_{ik} the actual values of the k_{th} output unit of the i_{th} training pattern.

Once the GRBF network is created, it is used to generate the solution to the proposed problem. The solution proposed is the output of the GRBF network created with the retrieved cases. The input to the GRBF network, in order to generate the solution, is the data related with the problem to be solved, the values of the variables stored in the case base.

3.3 Revision Service

After generating a prediction, the system needs to validate its correction. CROS can also query an expert user to confirm the automatic revision previously done. The system also provides an automatic method of revision that must be also checked by an expert user which confirms the automatic revision.

Explanations are a recent revision methodology used to check the correction of the solutions proposed by CBR systems [13]. Explanations are a kind of justification of the solution generated by the system. To obtain a justification to the given solution, the cases selected from the case base are used again. As explained before, a relationship between a case and its future situation can be established. If it is considered the two situations defined by a case and the future situation of that case as two vectors, a distance between them can be defined, calculating the evolution of the situation in the considered conditions. That distance is calculated for all the cases retrieved from the case base as similar to the problem to be solved. If the distance between the proposed problem and the solution given is not greater than the average distances obtained from the selected cases, then the solution is a good one, according to the structure of the case base. If the proposed prediction is accepted, it is considered as a good solution to the problem and can be stored in the case base in order to solve new problems. It will have the same category as the historical data previously stored in the system.

4 Preliminary Results

CROS uses different artificial intelligence techniques to cover and solve all the phases of the CBR cycle. Fast Iterative Kernel Principal Component Analysis is used to reduce the number of variables stored in the system, getting about a 60% of reduction in the size of the case base. This adaptation of the PCA also implies a faster recovery of cases from the case base (more than 7% faster than storing the original variables).

The predicted situation was contrasted with the actual future situation. The future situation was known, as long as historical data was used to develop the system and also to test the correction of it. The proposed solution was, in most of the variables, close to 90% of accuracy. For every problem defined by an area and its variables, the system offers 9 solutions (i.e. the same area with its proposed variables and the eight closest neighbors). This way of prediction is used in order to clearly observe the direction of the slicks which can be useful in order to determine the coastal areas that will be affected by the slicks generated after an oil spill.

Table 2. Percentage of good predictions obtained with different techniques

Number of cases	RBF	CBR	RBF + CBR	CROS
100	45 %	39 %	42 %	43 %
500	48 %	43 %	46 %	46 %
1000	51 %	47 %	58 %	64 %
2000	56 %	55 %	65 %	72 %
3000	59 %	58 %	68 %	81 %
4000	60 %	63 %	69 %	84 %
5000	63 %	64 %	72 %	87 %

Table 2 shows a summary of the results obtained after comparing different techniques with the results obtained using CROS. The table shows the evolution of the results along with the increase of the number of cases stored in the case base. All the techniques analyzed improve its results while increasing the number of cases stored. Having more cases in the case base, makes easier to find similar cases to the proposed problem and then, the solution can be more accurate. The "RBF" column represents a simple Radial Basis Function Network that is trained with all the data available. The network gives an output that is considered a solution to the problem. The "CBR" column represents a pure CBR system, with no other techniques included; the cases are stored in the case base and recovered considering the Euclidean distance. The most similar cases are selected and after applying a weighted mean depending on the similarity of the selected cases with the inserted problem, a solution s proposed. The "RBF + CBR" column corresponds to the possibility of using a RBF system combined with CBR. The recovery from the CBR is done by the Manhattan distance and the RBF network works in the reuse phase, adapting the selected cases to obtain the new solution. The results of the "RBF+CBR" column are, normally, better than those of the "CBR", mainly because of the elimination of useless data to generate the solution. Finally, the "CROS" column shows the results obtained by CROS, obtaining better results that the three previous analyzed solutions.

5 Conclusions and Future Work

CROS is a new solution for predicting the presence of oil slicks in oceanic areas after an oil spill. This system presents a distributed multi-agent architecture which allows the interaction of multiple users at the same time. Distributing resources also allows users to interact with the system in different ways depending on their specific needs for each situation (e.g. introducing data or requesting a prediction). This architecture becomes an improvement with previous tools where the information must be centralized and where local interfaces where used. With the vision introduced by CROS, all the different people that may interact with a contingency response system can collaborate in a distributed way, being physically located in different places but interchanging information in a collaborative mode.

CROS makes use of a Case-Based Reasoning system for creating new solutions and predictions using past solutions given to past problems. It has been demonstrated that the CBR system generates consistent results. The structure of the CBR system has been divided into services in order to optimize the overall performance of CROS.

Generalization must be done in order to improve the system. Applying the methodology explained before to diverse geographical areas will make the results even better, being able to generate good solutions in more different situations. The current system has been mainly developed using data from the accident of the Prestige in the north-west coast of Spain. With that information, CROS has been able to generate solutions to new situations, based on the available cases. If the amount and variety of cases stored in the case base is increased, the quality of the results will also be boosted.

Although the performed tests have provided us very useful data, it is necessary to continue developing and enhancing CROS. The number of possible interfaces can be augmented, including independent sensors that may send information to the system in real-time. The data received by the system must be analyzed in order to detect new spills and to generate fast and accurate solutions to existing problems without the direct intervention of the users. Then, the system will not only be a contingency response but also a kind of supervising system especially in dangerous geographical areas.

References

[1] Corchado, J.M., Bajo, J., De Paz, Y., Tapia, D.I.: Intelligent Environment for Monitoring Alzheimer Patients, Agent Technology for Health Care, Decision Support Systems (in press, 2008)

[2] Yang, J., Luo, Z.: Coalition formation mechanism in multi-agent systems based on genetic algorithms. Applied Soft Computing Journal 7(2), 561–568 (2007)

[3] Menemenlis, D., Hill, C., Adcroft, A., Campin, J.M., et al.: NASA Supercomputer Improves Prospects for Ocean Climate Research. EOS Transactions 86(9), 89–95 (2005)

[4] Palenzuela, J.M.T., Vilas, L.G., Cuadrado, M.S.: Use of ASAR images to study the evolution of the Prestige oil spill off the Galician coast. International Journal of Remote Sensing 27(10), 1931–1950 (2006)

[5] Solberg, A.H.S., Storvik, G., Solberg, R., Volden, E.: Automatic detection of oil spills in ERS SAR images. IEEE Transactions on Geoscience and Remote Sensing 37(4), 1916–1924 (1999)

[6] Ross, B.J., Gualtieri, A.G., Fueten, F., Budkewitsch, P., et al.: Hyperspectral image analysis using genetic programming. Applied Soft Computing 5(2), 147–156 (2005)

[7] Corchado, J.M., Fdez-Riverola, F.: FSfRT: Forecasting System for Red Tides. Applied Intelligence 21, 251–264 (2004)

[8] Karayiannis, N.B., Mi, G.W.: Growing radial basis neural networks: merging supervised andunsupervised learning with network growth techniques. IEEE Transactions on Neural Networks 8(6), 1492–1506 (1997)

[9] Cerami, E.: Web Services Essentials Distributed Applications with XML-RPC, SOAP, UDDI & WSDL. O'Reilly & Associates, Inc., Sebastopol (2002)

[10] Gunter, S., Schraudolph, N.N., Vishwanathan, S.V.N.: Fast Iterative Kernel Principal Component Analysis. Journal of Machine Learning Research 8, 1893–1918 (2007)

[11] Fritzke, B.: Growing cell structures—a self-organizing network for unsupervised and supervised learning. Neural Networks 7(9), 1441–1460 (1994)

[12] Ros, F., Pintore, M., Chrétien, J.R.: Automatic design of growing radial basis function neural networks based on neighboorhood concepts. Chemometrics and Intelligent Laboratory Systems 87(2), 231–240 (2007)

[13] Plaza, E., Armengol, E., Ontañón, S.: The Explanatory Power of Symbolic Similarity in Case-Based Reasoning. Artificial Intelligence Review 24(2), 145–161 (2005)