Smart Problem Solving Environment for Medical Decision Support

Andrei Petrovski
The Robert Gordon University
St. Andrew Street
ABERDEEN, AB25 1HG, Scotland
PHONE: +44 1224 262718
ap@comp.rgu.ac.uk

John McCall
The Robert Gordon University
St. Andrew Street
ABERDEEN, AB25 1HG, Scotland
PHONE: +44 1224 262780
jm@comp.rgu.ac.uk

ABSTRACT

This paper presents a medical problem solving environment (PSE) designed for modelling, simulation, and optimisation of clinical cancer chemotherapy. In order to find optimal chemotherapeutic treatments, two population-based evolutionary algorithms – Genetic Algorithms and Particle Swarm Optimisation – have been applied, which can use web services and grid computing to evaluate potential solutions in a distributed and customizable manner. The versatility and robustness of these algorithms make the suggested problem solving environment scalable and adaptable to other problem domains.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimisation – *constrained optimisation.*

I.2.8. [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *heuristic methods.*

I.6.7. [**Simulation and Modelling**]: Simulation Support Systems – *environments.*

J.3. [**Computer Applications**]: Life and Medical Sciences – *medical information systems.*

General Terms

Algorithms, design, experimentation.

Keywords

Problem solving environments, evolutionary algorithms, medical decision support, web services, cancer chemotherapy.

1. INTRODUCTION

Medical conditions often require practitioners to make complex, and often life-critical, decisions about selecting the correct treatment. Typically patient information is incomplete and noisy, the range of treatment options is subject to complex constraints and the aims of treatment are multi-objective. For these reasons, medical decision support is a rich application area for evolutionary algorithms and related techniques. However, the process of developing smart medical decision support tools is fraught with difficulty. In particular, solution encoding and evaluation requires a level of understanding of the application domain that is difficult and time-consuming to achieve. Also, medical practitioners need to understand how to apply the tools to problem scenarios. This is easiest when the interface to the problem presented by the tools is a natural one for the practitioners.

The aim of this paper is to explore how recent research on Problem Solving Environments (PSE) provides a framework for building smart systems that use computational intelligence approaches to assist medical problem solvers. The next section describes the architecture and organization of a generic PSE. In Section 3 we will describe our own system, which uses evolutionary algorithms to assist cancer chemotherapy design, as an instance of a PSE. In Section 4, we abstract key features of a generalised framework for smart PSEs for medical decision support and discuss the technologies that exist to implement them. Finally, in Section 5 we will draw some conclusions and outline possible directions for future research.

2. PROBLEM SOLVING ENVIRONMENTS

The most frequently cited definition of a problem solving environment (PSE) is given in [5], which states that a PSE is a computational system that provides a complete and convenient set of high level tools for solving problems from a specific domain. The PSE allows users to formulate and modify problems from this domain, choose solution strategies or algorithms, execute simulation or optimisation tasks, interact with and manage appropriate hardware and software resources, and record and coordinate problem solving sessions.

A user communicates with a PSE in the language of the problem, not in the language of a particular operating system, programming language, or network protocol [7]. PSEs also support collaboration among people separated in space or time, and provide access to a diverse set of information and computing recourses.

The main reasons for the development of PSEs are to simplify the usage of existing software modules and distributed computing resources, to assist in problem specification within the ballpark of the chosen application domain, to effectively find a solution to the specified problem, and to present this solution in a convenient and customisable form.

Modern PSEs are event-driven, i.e. the process of solving the problem is steered by the user, incorporate collaboration capabilities, and easily interface with existing software products. The realisation all these features in PSEs requires a very flexible data management architecture, which is described in the following subsections.

2.1 Common PSE Properties

A problem solving environment should allow users to concentrate on dealing with the problem without having to become experts in networks, parallel computing, and the WWW. All these computing tools and paradigms need to be adapted and customised in PSEs to the problem domain in question.

Many problems, faced by the users of PSEs, and their solution strategies are extremely heterogeneous: in model, in programming codes used, in software applications dealing with the problems, and in hardware platforms they are running on. PSEs are designed to manage this heterogeneity in an integrated way, so that the user sees a predictable and consistent interface.

Most science and engineering projects are performed in collaborative mode with physically distributed participants. A typical PSE provides the ability to foster collaborative solution strategies, such as data sharing, workflow control, common information services, and general-purpose conferencing tools [7]. Besides the need to support collaboration, effective solutions to many complex problems require seamless access to large distributed data resources.

Developing a solution to a complex problem may also necessitate significant time: therefore, PSE need to be able to provide certain persistency that allows the user to resume the solution process at a later time at a potentially different location. The persistence of a PSE can be enhanced with preferences that are either set by the user or are detected automatically.

Finally, in order to be useful, a PSE needs to be open and adaptable – the environment should be capable of incorporating new functionality with its existing base when necessary or to tailor its tools to a novel problem. The usability of the PSE can also be enhanced with the help of graphics and visual aids.

2.2 PSE Organisation

The organisation of a typical problem solving environment is based on the set of conceptual models that coherently and comprehensibly describe the problem domain. This set includes models of various types, where each model type defines a viewpoint from which the problem domain of the PSE is considered, concentrating on some aspects and hiding irrelevant ones in order to reduce the complexity [4].

The PSE organisation usually has three layers [3], depicted in Figure 1.

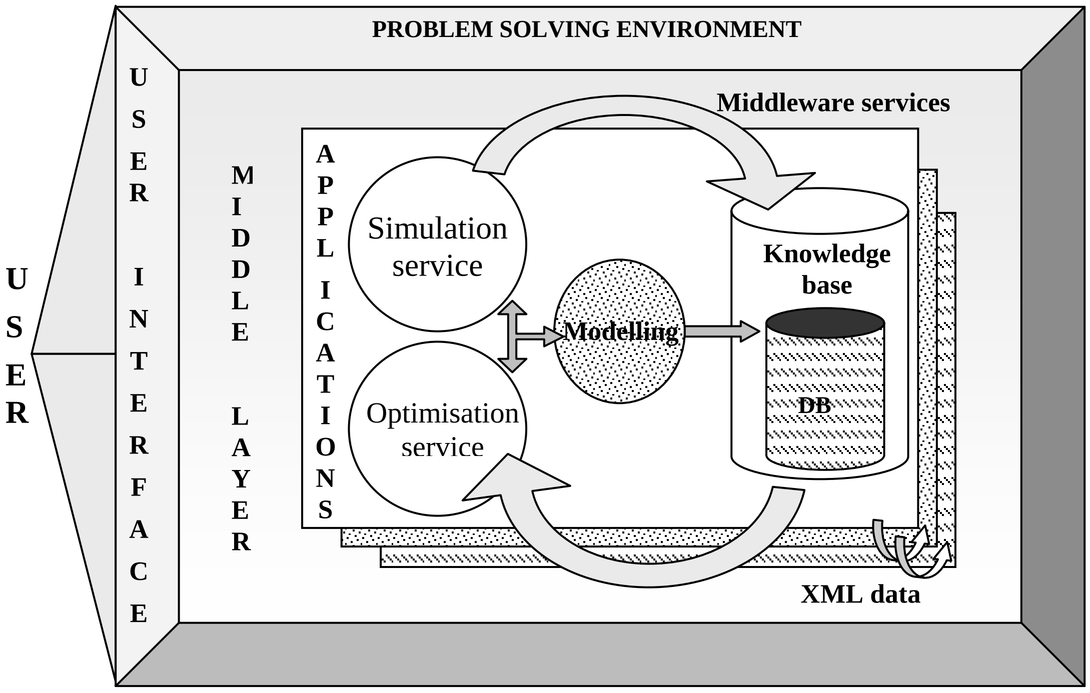


Figure 1. Layered organisation of a typical PSE

The top layer in the figure above is an intelligent GUI for application development and use. With the help of the GUI the users of PSEs create and modify, if necessary, various problem solving applications. For this purpose a set of computational templates can be used that assist in rapid prototyping and composing new application modules. The intelligence of the GUI layer emerges from its ability to pre-select and adapt the application templates on the basis of user requirements and the problem specification.

The middle layer of the PSE consists of coordinating and monitoring "middleware" that is based on two components. The first component is the software architecture for designing and building PSE services. Therefore, it is a service-oriented architecture (SOA) that provides the user with an illusion that the environment operates on a single 'virtual' machine with its own operating system. This operating system is responsible for providing secure access to PSE services, for ensuring fault-tolerance, and for improving responsiveness and efficiency of the environment as a whole.

Another role of SOA is to configure entities, such as services, registries, contacts, and proxies, to maximise loose coupling and reuse of PSE components [9]. In Section 4 of this paper we will outline the implementation details of this architecture, where web services and their usage within PSEs will be discussed.

The second component of the middle layer of the PSE is a documentation system that allows the PSE itself and its various applications and modules to be described, automatically discovered, and used. The eXtensible Markup Language (XML) addresses the data representation issue in PSEs by providing a standard way to define the document structure that is suitable for automatic processing. Again, more details on the documentation component of the PSE are to be given in Section 4.

The lowest (application) layer of the PSE is composed of software libraries, modules, knowledge- and data-bases necessary to execute the functions of the problem solving environment. When a new application-specific PSE is created, only this layer and the GUI are affected; the middle layer will remain unchanged, promoting versatility of the PSE concept.

Having described the general features and organisational structure of problem solving environments, we will particularise them in the next section on the example of the bespoke software system for designing chemotherapeutic schedules of cancer treatment.

3. CANCER CHEMOTHERAPY PSE

Cancer progression in a patient undergoing chemotherapy treatment is an extremely complex phenomenon that affects many levels of human body organisation: the gene level, the cell level, the level of organs and organ formation (e.g. angiogenesis) [1]. The processes happening at each level are very complicated, and the overall system (i.e. the cancer patient) is even less tractable due to the multiple interactions between these processes.

This generates a fast growing number of different possible protocols of cancer chemotherapy. Given the limited human and financial resources for clinical trails, optimal protocols cannot be determined empirically. Instead, a formal and systematic methodology is necessary for suggesting promising drug schedules that achieve certain objectives set by the clinician and satisfy the constraints imposed on systemic treatments.

To date, an extensive effort has been invested in both the theoretical investigation and in practical realisation of cancer chemotherapy control methods. In our previous work [8], we presented simulation modelling and computational optimisation as decision support techniques that can be useful for understanding problems related to cancer treatment using chemotherapy. In the next subsection we briefly describe the suggested methodology of decision support.

3.1 Services of Oncology Workbench

In [8], we gave a detailed description of the existing architecture of the Oncology Workbench (OWCH). The main components of this system, shown in Figure 2, are the GUI, which combines the Treatment Editor (TE) and Results Viewer (RV), Simulation and Optimisation Engines (SE and OE), and Information Repository (IR).

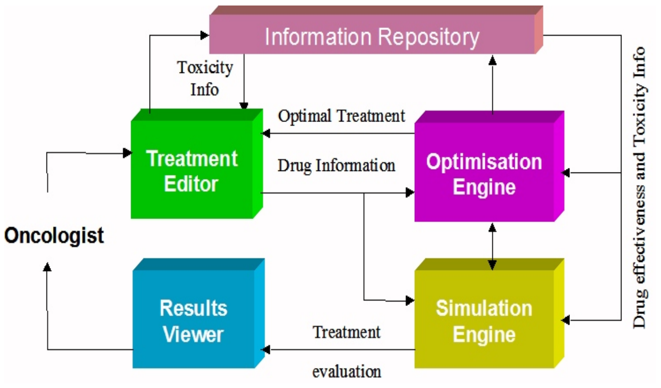


Figure 2. Oncology Workbench

The TE enables the user to compose chemotherapy schedules by selecting various anti-cancer drugs, their dosages and timing. While involved in this activity, the user is able to monitor the toxic effects of the treatment on various organs; the RV displays the results of simulation (Figure 3).

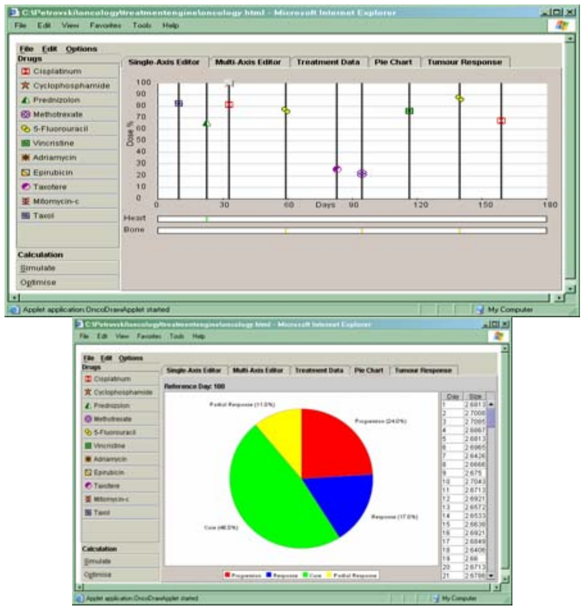


Figure 3. Oncology Workbench GUI

Behind its graphical interface, OWCH runs two software engines, SE and OE, that provide tumour simulation and optimisation services. The real power of simulation modelling lies in its ability to support decision making by allowing clinical oncologists to consistently evaluate the effects of known treatment strategies. SE is therefore used to predict the patients' response to the composed chemotherapy schedules.

The optimisation service, on the other hand, is aimed at suggesting novel treatment strategies so that a comprehensive analysis of a wide range of alternative treatments can be done. The operation of both services is dependent on the reliability of data that describes the patients' characteristics (e.g. toxicity limits, tumour growth rates, etc.), drug details (e.g. efficacy, level of drug-drug interaction) and various types of cancer. All these data are stored in the Information Repository, which can be updated on a regular basis.

OE uses the simulation service, provided by SE, for evaluating novel treatment schedules, and relies on intelligent search techniques based on evolutionary algorithms, which are described in the next subsection.

3.2 Evolutionary Algorithms in Chemotherapy Optimisation

Many decision-making activities involve searching through a large space of possible solutions. In the chemotherapy problem we have studied [10, 13], the size of the solution space increases exponentially with the number of decision variables (i.e. drug doses administered over the period of chemotherapeutic treatment), the values of which need to satisfy certain feasibility criteria.

The requirements imposed on decision variables often make the structure of a solution space quite intricate - regions of feasible solutions are scattered irregularly throughout the solution space, and only one of these regions contains the optimal solution. To find the optimal solution in such situations becomes a difficult task for conventional optimisation methods (gradient-based or simple heuristics). Similarly, the methods of mathematical programming cannot easily deal with multiplicity of feasible regions in the solution space.

Evolutionary algorithms, on the other hand, show a good and robust performance on a class of non-linear, multi-constrained problems. In particular, the population-based algorithms, such as Genetic Algorithms (GAs) and Particle Swarm Optimisation (PSO), are specifically suitable to the optimisation problem of cancer chemotherapy, the characteristic features of which are multimodality and a disjoint nature of feasible regions in the solution space [11].

3.2.1 Chemotherapy Treatment as GA Chromosomes

For multi-drug treatments the solutions to the problem of chemotherapy optimisation may be expressed as decision vectors $\mathbf{c} = (C_{ij}), i \in \overline{1,n},\ j \in \overline{1,d}$ of n discrete doses for each of the d anti-cancer drugs used. Using the Genetic Algorithms' terminology, the representation space $\mathbf{I}$ (a discretized version of the search space Ω) can then be expressed as a Cartesian product:

$$\mathbf{I} = A_1^1 \times \ldots \times A_1^d \times \ldots \times A_n^1 \ldots \times A_n^d \qquad (1)$$

of allele sets A_i^j. Each allele set uses a 4-bit representation scheme

$$A_i^j = \left\{ a_1 a_2 a_3 a_4 : a_k \in \{0,1\} \forall k \in \overline{1,4} \right\} \qquad (2)$$

so that each drug dose C_{ij} takes an integer value in the range of 0 to 15 concentration units. In general, with n treatment intervals and up to 2^p concentration levels for d drugs, there are up to 2^{npd} individual elements. Henceforth we assume that $n = 10$ and that the number of available drugs is also restricted to ten.

The values $n = 10$ and $d = 10$ result in the representation (search) space of power $\left|\mathbf{I}\right| = 2^{400}$ individuals, referred to as chromosomes.

Thus, a chromosome $x \in \mathbf{I}$ can be expressed as

$$x = \left\{ a_1 a_2 a_3 \ldots a_{4nd} : a_k \in \{0,1\}\ \forall k \in \overline{1,4nd} \right\} \qquad (3)$$

and the mapping function $m : \mathbf{I} \to \mathbf{C}$ between the individual $\mathbf{I}$ and the decision vector $\mathbf{C}$ spaces can be defined as

$$C_{ij} = \Delta C_j \sum_{k=1}^{4} 2^{4-k} a_{4d(i-1)+4(j-1)+k},\ \forall i \in \overline{1,n},\ j \in \overline{1,d} \qquad (4)$$

where ΔC_j represents the concentration unit for drug j. This function symbolizes the decoding algorithm to derive the decision vector $\mathbf{c} = m(x)$ from a chromosome x.

If this vector violates any of the constraints imposed on cancer chemotherapy treatment [8], penalties are applied to the fitness function based on the following optimisation objective:

$$\underset{\mathbf{c}}{\text{maximise}} \quad J(\mathbf{c}) = \int_{t_1}^{t_n} \ln\left(\frac{\Theta}{N(\tau)}\right) d\tau \qquad (5)$$

which corresponds to minimising the overall tumour burden during treatment [10].

The initial GA population of 50 individuals is chosen at random in accordance with the representation scheme (2)-(4). The selection procedure is based in the roulette-wheel selection, augmented by a linear fitness normalization technique [8] and an elitist strategy that reserves two best chromosomes in the population. Recombination is implemented as a two-point crossover followed by a uniform mutation.

3.2.2 Particle Swarm Optimisation

The PSO algorithm is initialised with a population of random candidate solutions, conceptualised as particles. These particles are flown through the hyperspace Ω of solutions to the chemotherapy optimisation problem described in the previous section. The position of each particle $\overline{c}_i^{k+1}$ at iteration $k+1$ corresponds to a treatment regimen of anti-cancer drugs and is determined by the following formula:

$$\overline{c}_i^{k+1} = \overline{c}_i^k + \overline{v}_i^k \quad (6)$$

where is $\overline{v}_i^k$ a randomised velocity vector assigned to each particle in a swarm. The velocity vector drives the optimisation process and reflects the 'socially exchanged' information. Therefore, each particle in the swarm is attracted towards the locations representing best chemotherapeutic treatments found by the particle itself, its neighbours, and/or the entire population.

Similar to the GA population, initial positions of 50 PSO particles are generated at random. Each particle in the swarm is assigned a random velocity value from the range $[0,2]$; this value changes at each iteration of the algorithm according to:

$$\overline{v}_i^k = w \cdot \overline{v}_i^{k-1} + b_1 \cdot r_1 \cdot \left(\overline{c}_i^* - \overline{c}_i^{k-1}\right) + b_2 \cdot r_2 \cdot \left(\overline{c}_i^{**} - \overline{c}_i^{k-1}\right) \quad (7)$$

where:

- w is the inertia coefficient assigned a randomly generated value from the range $[0.5,1]$;
- $b_1 = b_2 = 4$ are empirical coefficients used to improve PSO performance;
- r_1 and r_2 are random numbers is the range $[0,1]$;
- $\overline{c}_i^*$ and $\overline{c}_i^{**}$ are the best locations in Ω found by the particle i and the entire population respectively;
- $\overline{v}_i^{k-1}$ is the value of particle i velocity at previous iteration of the algorithm; the values $\overline{v}_i^0$ are initialised at random.
- the velocity bound values $|v_{\max}|$ are set to 1 to help keep the swarm under control.

The programs implementing both GA and PSO algorithms are written in Java; these programs run until a predefined termination criterion is satisfied. The termination criterion was chosen to be 25,000 fitness function evaluations, because it has been empirically found that this number of evaluations guarantees finding a feasible solution for all trial runs of at least one algorithm [11].

The methodology of using evolutionary algorithms opens many more perspectives in optimising cancer chemotherapy if more biological and medical features are incorporated into the decision support activity. In order to make this activity more useful in clinical practice, the following aspects of cancer treatment need to be considered.

3.3 Biological and Medical Aspects of Cancer Chemotherapy

In cancer chemotherapy, it is important to design treatment strategies that ensure a desired effect on tumour cells without overdosing the host [14]. Mathematical modelling and techniques from optimal control are usually used to achieve the desirable outcome subject to satisfying treatment constraints.

Mathematical models have been developed to aid in describing the mechanisms of cytotoxic drug delivery to the tumour site (pharmaco-dynamics/kinetics modelling), the action of the drug on tumour cell populations, and its toxic side effects. Optimal control theory and heuristic optimisation use the models developed to find optimal chemotherapy strategies [15].

However, these models need to be modified when more accurate insights into biological processes are obtained and new drugs are developed with a different mode of action or toxic effects. For instance, hamotopoietic growth factors (HGF) used in conjunction with cancer chemotherapy have led to safer delivery of standard- and high-dose chemotherapy regimens. Also, in order to realistically evaluate the response of the tumour to chemotherapy, one needs to incorporate the effects of drug resistance into the tumour growth model [2].

When metastatic tumours with different growth and drug response characteristics are present, multiobjective optimisation that both maintains the normal cell population and tries to eliminate all existing tumours needs to be applied. Multiobjective optimisation does not have to rely only on mathematical models of tumour behaviour - one alternative is to model metastatic tumours using cellular automata [1], naturally allowing computational methods to start playing a more important role in this problem domain.

In summary, the considerable variety of biological and medical processes involved in cancer chemotherapy makes it practically impossible to take them all into account at once. A better approach would be to segregate these processes into relevant groups and to provide a specific procedure (referred to as *service*) of dealing with all processes within each group. The combination of all available services creates an integrated framework, or a PSE, with the help of which the problem of cancer chemotherapy can be addressed in the most comprehensive and effective manner.

4. DEVELOPING A PSE ON THE BASIS OF ONCOLOGY WORKBENCH

The rapid development of computer and information system technology enables creating virtual problem solving environments for complex practical problems. As can be seen from the arguments we presented in Section 3.2, treating cancer using chemotherapy is one of such problems that necessitates sophisticated modelling, simulation and optimisation of medical processes within an integrated environment.

One of the most important underpinnings of PSEs is the management of diverse types of information and of heterogeneous software across a variety of network platforms. The distributed information technologies collectively known as web services recently have been shown to offer powerful capabilities in implementing scalable software systems. A particular appeal of the web services approach is the rapid integration cycle it provides, which enables automatic composability of PSEs by using a discovery service on web service registries [12]. For this reason, we are going to adopt web services as a standardised way to expose the functionality of the PSE and to make it available through established web technologies.

4.1 Web Services

A web service is a software component with a unique Uniform Resource Identifier (URI), whose public interfaces and bindings are defined and described using XML [9]. The web services framework provides a set of operations, modular and independent applications that can be published, discovered and invoked by using industry standard protocols, such as the XML, Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL), Unified Description, Discovery and Integration (UDDI).

The vision of the web service technology is that services will work together seamlessly because they are developed to the same standards of self-description, publication, location, communication, and data exchange [12]. As a result, applications that use web services can dynamically locate and invoke necessary functionality, whether available locally or from across the Internet. From the point of view of designing a PSE for optimising cancer chemotherapy treatment, such capability is extremely important given the vast amount of clinical data stored in databases all over the world and the heterogeneity of medical procedures and regulations of using these data.

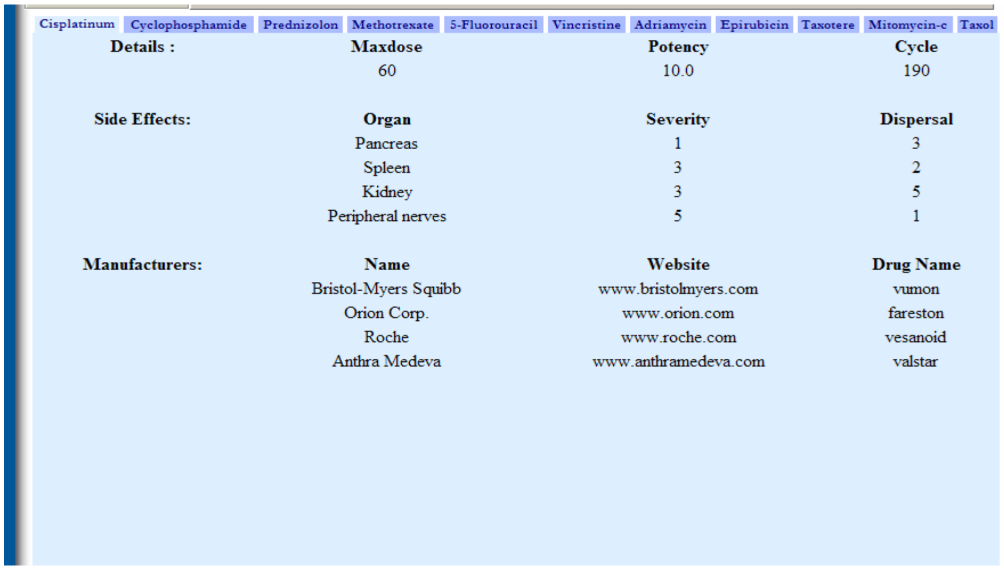
However, in order to implement a cancer chemotherapy PSE, we need to define a shared ontology for data exchange between the web services involved. The common vocabulary for these web services takes the form of an XML tagset, described in the next subsection.

4.2 XML Format for Medical Data

The main advantage of XML is that it provides platform-independent data exchange and a framework for the specification of data structures. In our decision support methodology, XML is used for the design of user interfaces and as a configuration language for describing the interconnection of tasks.

The main data component of the Oncology Workbench is an anti-cancer drug used in chemotherapy treatment. The drug characteristics, internally represented as XML tags, are shown in Figure 4 and include the name, toxicity details, efficacy, typical delivery mode, and manufacturer. These characteristics are required by the PSE services.

The list can be easily extended if more data become available or relevant. Similarly, more data structures (e.g. the population of patients) in the same format can be introduced in the PSE, should a need arise.



Details :	**Maxdose**	**Potency**	**Cycle**
	60	10.0	190
Side Effects:	**Organ**	**Severity**	**Dispersal**
	Pancreas	1	3
	Spleen	3	2
	Kidney	3	5
	Peripheral nerves	5	1
Manufacturers:	**Name**	**Website**	**Drug Name**
	Bristol-Myers Squibb	www.bristolmyers.com	vumon
	Orion Corp.	www.orion.com	fareston
	Roche	www.roche.com	vesanoid
	Anthra Medeva	www.anthramedeva.com	valstar

Figure 4. Representation of an anti-cancer drug

4.3 Deployment of Web Services

Depending on the way how the user's data is handled by the business logic programs or libraries, three programming paradigms have been identified for network-based systems and environments: proxy computing, code shipping, and remote computing. PSEs usually use the remote computing paradigm, whereby the user's data is sent to a remote server, is operated upon, and the result is then sent back to the user's machine [16].

The most common way of deploying web services consists of a SOAP messaging on top of HTTP that is used as a transport mechanism. However, the web service technology is far more flexible and conducive to applying it with the realms of grid computing.

Grid computing can be seen as a natural evolution of distributed computing technologies such as RMI and CORBA [18]; it refers to a parallel distributed system composed of heterogeneous resources, located in different places, that are connected over a network using open standards and protocols.

Both grid computing and web services are related to distributed computing, but they address it in largely 'orthogonal' ways [12]. The definition of web services focuses on the use of XML to describe both service interfaces and the communication data format. Grid computing, on the other hand, focuses on the system architecture, leaving the particulars of protocols and message formats unspecified. With such complementary goals, it is beneficial to merge both technologies together in order to create a powerful platform for deploying the cancer chemotherapy PSE.

5. DISCUSSIONS

In this paper we have presented a problem solving environment for medical decision support aimed at finding the best chemotherapeutic treatment of cancer. It is possible to abstract three distinct elements in the suggested methodology of decision support. Firstly, the exploration of a solution space of possible treatments is done by the optimisation services that in our system are based on evolutionary algorithms. Two alternatives – Genetic Algorithms and Particle Swarm Optimisation – have been successfully applied to the specified problem domain.

Secondly, the optimisation algorithms use simulation services to assess the quality of potential solutions. These simulation services, in turn, may use toxicity modeling and drug-drug interaction services to more precisely evaluate potential treatments. Thirdly, the problem-solving process is configured and directed by middleware control services that enable data exchange between the first two elements of the PSE, as well as with the user [17]. The latter services are likely to be distributed throughout the system.

In conclusion we would like to say that the concepts, methodology and techniques described in this paper by no means are confined to cancer chemotherapy optimisation alone. By adapting the existing services within the PSE or by introducing the new ones [6], other smart PSEs can be built that provide medical decision support in other problem domains (e.g. a diabetes management system, telemedicine and e-health applications, etc.).

6. REFERENCES

[1] Agur, Z., Hassin, R., and Levy, S.: Optimizing chemotherapy scheduling using local search heuristics. *Journal of Operations Research* (**In press**).

[2] Barbolosi, D., Iliadis, A.: Optimizing drug regimens in cancer chemotherapy: a simulation study using a PK-PD model. *Computers in Biology and Medicine*, *31*, (2001), 157-172.

[3] Caskey, K.R.: A manufacturing problem solving environment combining evaluation, search, and generalization methods. *Computers in Industry*, *44*, (2001), 175-187.

[4] Delen, D., Benjamin, P.C.: Towards a truly integrated enterprise modeling and analysis environment. *Computers in Industry*, *51*, (2003), 257-268.

[5] Gallopoulos, E., Houstis, E., Rice J.R.: Problem Solving Environments for Computational Science. *IEEE Computational Science and Engineering*, *1*, (1994), 11-23.

[6] Hey, A. J. G., Papay, J., Keane, A.J., Cox, S.J.: Component Based Problem Solving Environment. *Lecture Notes in Computer Science*, *2400*, (2002), 105-112.

[7] Laszewski, G., *et al*: Designing Grid-based Problem Solving Environments and Portals. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, (Maui, Hawaii, January 2001).

[8] McCall, J., Petrovski, A.: A Decision Support System for Cancer Chemotherapy Using Genetic Algorithms. In *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation*, *1*, (Vienna, Austria, 1999), IOS Press, 65-70.

[9] McGovern, J., et al: *Java Web Services Architecture*. Morgan Kaufmann, Elsevier, 2003.

[10] Petrovski, A., McCall, J.: Multi-objective optimisation of cancer chemotherapy using evolutionary algorithms. In *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimisation*, (Zurich, Switzerland, 2001).

[11] Petrovski, A., Sudha, B., McCall, J.: Optimising Cancer Chemotherapy Using Particle Swarm Optimisation and Genetic Algorithms. In *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature*, (Birmingham, U.K. September 2004), Lecture Notes in Computer Science **3224**, 633-641.

[12] Pullen, J. M., *et al*: Using Web services to integrate heterogeneous simulations in a grid environment. *Future Generation Computer Systems*, *21*, (2005), 97-106.

[13] Tan, K., *et al* : Automating the drug scheduling of cancer chemotherapy via evolutionary computation. *Artificial Intelligence in Medicine*, *25*, (2002), 169-185,

[14] Verweij, J., De Jonge, M. J. A. Achievements and future of chemotherapy. *Review Article. European Journal of Cancer*, *36*, 12 (2000), 1479-1487.

[15] Villasana, M., Ochoa, G.: Heuristic Design of Cancer Chemotherapies. *IEEE Transactions on Evolutionary Computation*, *8*, 6 (2004), 513-521.

[16] Youn, C., Pierce, M., Fox, G.: Building Problem-Solving Environments with Application Web Service toolkits. *Future Generation Computer Systems* (**In Press**).

[17] Wang, Y.D., Shen, W., Ghenniwa, H.: WebBlow: a Web/agent-based multidisciplinary design optimization environment. *Computers in Industry*, *52*, (2003), 17-28.

[18] Wikipedia, Grid Computing: http://en2.wikipedia.org/wiki/Grid_computing