TECHNICAL REPORT

CTR/35/04 **October 2004**

The evolution of web-based optimisation from ASP to e-service.

Patrick Valente and Gautam Mitra

Brunel UNIVERSITY WEST LONDON

Department of Mathematical Sciences
Brunel University, Uxbridge, Middlesex, UB8 3PH
www.carisma.brunel.ac.uk

CARISMA

The evolution of web-based optimisation: from ASP to e-Services

Abstract

In many application domains, optimisation, decision analysis and other analytic models are often embedded as a decision engine within the respective business processes. In order to respond effectively to global as well as regional competition, successful companies have started to formalise and improve their Business Process Management (BPM) procedures. In this paper, we study recent trends in the provision of optimisation tools and optimisation based decision support systems as web-enabled distributed applications. We also analyse the evolution from the Application Service Provision (ASP) model to the e-Services model, and we illustrate the importance of distributed optimisation components in the effective deployment of embedded "business analytics". As implementation examples of these paradigms, we provide an overview of the OSP and the WEBOPT projects, which exploit web service technology to deliver optimisation based applications and optimisation components in a distributed environment.

Table of contents

1 Introduction and background

Practical optimisation is playing an increasingly important role in the disciplines of engineering, computer science, and their many applications in industry. Optimisation modelling and solution techniques have seen sustained developments from the early sixties until now. The main achievements have come from the improvements in model conceptualisations, solution algorithms and software techniques. More recently, during the nineties, the important connections between analytic data (data marts), optimisation (decision) models and powerful solver engines have emerged as a growing field of research and development [31],[41]. Advances in this area, combined with the availability of ever more powerful PCs, has proven instrumental in the acceptance of optimisation as an inference engine in decision support system in a variety of industrial sectors, including finance, manufacturing and supply chain management, energy and utilities and environmental planning. In these fields, modern optimisation tools have not only been successfully incorporated into existing organisational information systems: they have also become integral part of organisational business processes. As a result, decision technology plays a role of growing importance in business process management.

Concurrently with this integration trend, the modern setting of the digital economy is creating abundant opportunities for applying operational research (OR) models which harness the growing presence and power of the Internet [22]. Many experts have identified that a key issue in the acceptance of optimisation is the progressive adoption of web-driven technologies. The World Wide Web already offers information, advice and remote access to software for solving optimisation problems. Well established companies such as IBM, SAP and ILOG offer products which are specially developed and evolved to enhance existing optimisation software tools with the support of Internet technology devices.

Since the work in [3], where the authors analyse the opportunities and requirements in terms of electronic markets for decision technologies, new concepts have emerged and have become established: the Application Service Provision (ASP) model, which enables the delivery of software through the Internet, as well as the more general e-Service paradigm, are considered in this paper as viable business models for the effective deployment of optimisation techniques and optimisation based decision support systems in an electronic market. The evolution of the optimisation tools, which are now available in component form, combined with the recent developments in information technology (with emphasis on the architecture of distributed applications), have lead to novel approaches in the provision of decision support systems. This paradigm is analysed in this paper, together with the requirements which such architecture presents in terms of business model.

The paper is organised as follows: in section 2, we provide an overview of the evolution of the software tools for mathematical programming and highlight their growing acceptance as decision making tools. We also show how traditional "desktop" applications based on stand-alone optimisation tools can now be replaced with distributed systems thanks to the availability of new software components and the adoption of Internet technology. In section 3, we illustrate the concepts behind the ASP model and introduce the more recent e-Services paradigm, outlining a business strategy which should be taken into account by the providers of optimisation e-Services. In section 4, we describe the technology requirements for the effective implementation of distributed systems for optimisation, and illustrate the architecture of web services and the grid computing environment. In section 5, we present the results of the OSP (Optimisation Service Provision) project, and introduce WEBOPT, which extends OSP by adopting the e-Services model for the provision of optimisation tools, optimisation components and vertical applications over the Internet. We also provide some explanation of the distributed component technologies and implementation details which underpin the WEBOPT project. In section 6 we set out our conclusions.

2 Optimisation tools and decision support systems

2.1 The evolution of the software for optimisation

The software tools for modelling, solving, data management and deployment of optimisation applications have coevolved with generations of software systems; Figure 1 captures this evolution in a historical perspective.

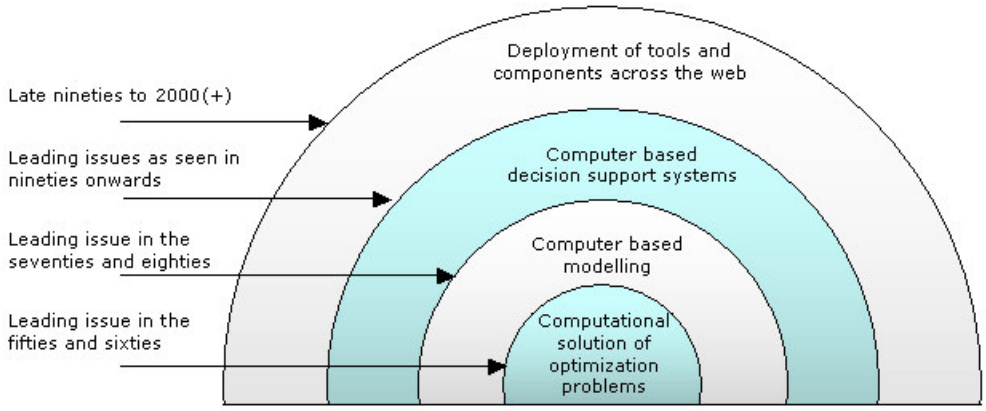

Figure 1. The evolution of optimisation tools.

This is an update of our perspective set out in our eighties paper [40]. Leading issues in the fifties and sixties were computational solution of optimisation problems; in the seventies and eighties the focus moved to computer based modelling of mathematical (optimisation) problems, while in the nineties researcher worked on integrating optimisation within decision support systems. Since then

connectivity with analytic data has gained in importance. From late nineties to the present time the use of the web technology for the deployment of these tools has emerged as the leading issue. The importance of the web enabled use of these otherwise well established tools merit further discussion. Over the last five to seven years with growing use of the pervasive Internet platform there has been a steady move towards net-based use of these software systems. In general, the natural trend is to fuse these together to create prototype as well as full-scale applications. In the rest of this section we provide a summary analysis and review of these tools and their natural evolution to the present day usage. In this review we highlight how as a consequence of this evolutionary process, optimisation based decision making has progressed from exploratory software systems to prototype applications and finally to fully-fledged DSS with embedded optimisation.

Modelling systems

In the early applications of mathematical programming, models were generated by ad-hoc computer programs written using procedural languages such as FORTRAN. These programs were used to generate the matrix associated with a given LP/IP model; an appropriate solution algorithm was used to process the matrix and report the optimal solutions. However, this approach lacked flexibility: even minor changes in the model's data or structure could require major adjustment of the matrix generation code. To aid practitioners in the creation of mathematical programming models and in the interpretation of the results, special purpose programmes, the combined *matrix generators/report writers*, (MGRW) appeared during the 1970s. These systems enabled the generation of an LP/IP model's matrix in a well defined readable format, which provided a standard interface between modelling and solving systems. This format, first adopted by the *Mathematical Programming System* (MPS), was introduced by IBM [25] and is still in use to date as the de facto standard of matrix input format. MGRWs, however, were affected by a number of limitations; in particular, the algebraic model was data dependent, thus a model could not be easily re-instantiated using different data sets.

In the 1980s, a new class of software systems for mathematical programming emerged. These modelling systems, based on Algebraic Modelling Languages (AMLs) enable the definition of models via symbolic algebraic expressions. Algebraic modelling systems interpret the algebraic model and use a given set of data to create model instances in MPS format or equivalent. Algebraic modelling languages are traditionally declarative languages. A characteristic of all declarative languages is that they describe *what* a given problem is, without actually specifying *how* the problem must be solved. The algebraic notation used in the formulation of MP models and supported by the AMLs plays an important role in the comprehension and maintenance of the models [33]. Indeed, the algebraic formulation implies the *abstraction* of the model from the specific *instance* of the problem, thus enabling the separation between data modelling and modelling of the problem's structure. MPL [38],

LINGO [35], CAMPS [37] are some representative algebraic modelling languages which are almost entirely declarative. Some AMLs also include procedural features, such as IF-THEN-ELSE statements and looping constructs: AMPL [19], GAMS [6], LPL [23], UIMP [13], OPL [27] and AIMMS [4] belong to this family of mixed declarative/procedural languages. Procedural constructs enable a closer coupling of modelling systems and solvers, which can be exploited for techniques such as column generation and the implementation of decomposition algorithms. Modern algebraic modelling systems also provide direct connections to database management systems, which can therefore be used to store both the analytic data used by the model, as well as the decision data obtained by solving the model. A comprehensive review of modelling systems and their features can be found in [11].

Solvers

Similarly, algorithmic developments for the solution of mathematical programming problems have seen continuous improvements from the 1950s until now. In particular, the introduction of the MPS format for the representation of LP and MIP model instances enabled researchers to separate the development of solution algorithms from the creation of model instances, thus simplifying the testing and benchmarking of the algorithms. Linear Programming (LP) solvers are typically based on the simplex method and interior point method, which are well established from both the theoretical and the implementation point of view. However, substantial reductions in computing time continue to be reported, particularly for the largest LPs. Advances in integer programming and the use of clever heuristic within the branch and bound, branch and cut and branch and price algorithms, have also lead to very fast mixed integer programming solvers. More recently, quadratic programming and quadratic mixed integer programming, as well as stochastic programming problems can be solved very rapidly using commercial packages such as CPLEX [26], FortMP [12], XPRESS [9], OSL [24]. For a survey on the current state of the art in modelling systems and solvers the reader is referred to [11, 18].

Data-Model interaction

As discussed in [31], the effective deployment of optimisation as a decision support tool relies in the integration of optimisation models with organisational data. Data modelling involves the definition of relationships between data items, and allow the decision maker to extract classify, and store the information associated with the parameters which are relevant to the decision problems. This ultimately leads to a relational data model, which provides the sets and indices of the algebraic model [17],[43]. It was only in the 1990s, however, that modelling systems started to support direct connections to relational databases for the retrieval of analytic data. Modern modelling systems are able to connect to database systems and spreadsheets, typically (but not exclusively) using the Open

Database Connectivity protocol (ODBC) and SQL queries. More recently, database systems have evolved and are now capable to store data using the XML format (eXtensible Mark-up Language), and some modelling systems such as AIMMS and MPL have been extended to capture data in this format. The importance of XML, particularly in association with Internet technology and distributed components, is discussed in section 4.

From stand-alone tools to components

The solvers and modelling systems developed during the 1990s evolved both in terms of capabilities of the languages and solution algorithms, as well as in terms of interconnectivity and technology platform. For the purpose of this paper, we focus on the technology platform, which we identify as the IT model associated with a software program. This last generation of optimisation tools were created as stand-alone programs (that is, an executable program which could be called from a command line environment such as MS DOS or UNIX), users started to express the need for a deeper interaction with these tools at a programmatic level. Indeed, in the late 1990s, algebraic modelling systems and solvers started to be used as inference engines within customised applications, leading to optimisation-based Decision Support Systems. However, it was only when the optimisation tools became available in the form of callable libraries, that the developers of these systems could finally integrate the underlying optimisation tools effectively.

Nowadays, most modelling systems and solvers are available as either object libraries based on technologies such as the Common Object Model (COM) and the Common Object Request Broker Architecture (CORBA), or as callable libraries (dynamic and static). In particular, the MPL algebraic modelling system was the first to be made available as a library of COM objects called OptiMax 2000, which enables rapid deployment of the algebraic modelling features into vertical applications. The AIMMS modelling system is also available as an object library, while some development is under way to create a similar component based version for the AMPL system [46]. These object libraries are extremely flexible usually contains very extensive sets of collections, objects, methods and properties, covering all aspects of implementing optimisation models in end-user applications. This includes reading and maintaining models, writing solution files, importing and exporting data, solving models, and option handling. Developers have programmatic access to the internal structures used in the modelling process, such as sets, data and decision vectors, as well as the matrix associated with the model. In essence, the users are able to integrate the declarative features of these modelling system in a procedural environment such as C++, Visual Basic and other programming languages, greatly extending the scope of optimisation based decision support systems. For instance, repeated optimisation within a loop can be easily implemented and used to evaluate solution and to simulate the behaviour of the model using different data sets. For instance, in section 5.1 of this paper, we

illustrate a DSS which uses the OptiMax 2000 library and computes a set of points in an efficient frontier by repeated optimisation. This DSS application is created using a library of data access, modelling and solver objects.

2.2 Optimisation as a decision making tool

The traditional view of a mathematical programming based modelling and solving environment is shown in Figure 2.

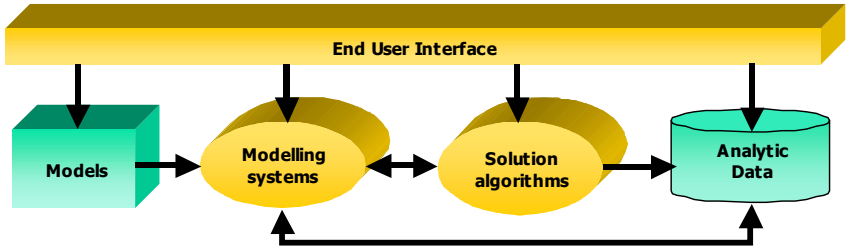

Figure 2. Traditional view of an optimisation-based application

The architecture is centred on the (algebraic) modelling systems, which process one or more algebraic models and the analytical data in order to generate a model instance. The instance is then passed onto the solvers (which implement one or more of solution algorithms). These optimise the model instance and return the values of the optimum decisions and objective (decision data) to the modelling system, which in turn stores them into the database. The components are typically controlled by a Graphical User Interface (GUI).

The architecture described above can be applied to implement decision support system based on mathematical programming. Indeed, optimisation models are considered as critical components of an organisation's analytical Information Technology (IT) systems, as they are used to analyse and control critical business measures such as cost, profit, quality, and time. As discussed in [30], quantitative models constitute an important component within the information value chain (IVC) (see Figure 3). Typically, transactional (operational) data which is available in a firm is analysed and synthesised to create analytic data or information, which is the first step in the information value chain. The analytic data is stored in data marts, which are subsets of the overall collection of data within a firm called data warehouse.

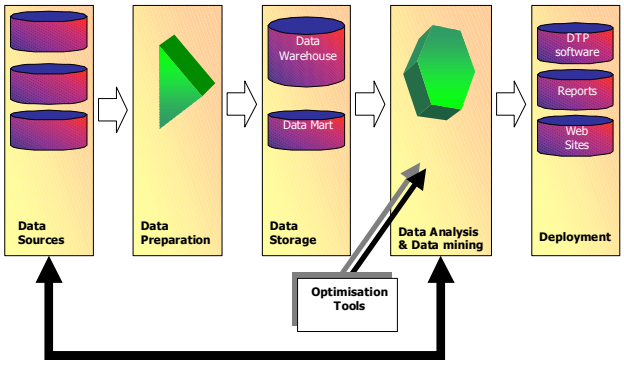

Figure 3. The information value chain

In this context, optimisation models and the tools which enable their formulation and solution are considered act as inference engines for knowledge discovery. They operate on the analytic data stored in the data marts to provide decision makers with decision data which is inferred from the models. Optimisation models can also be used within the more general framework of simulation, enabling decision makers and analysts to take advantage of descriptive models as well as prescriptive models. Another important aspect of the information value chain is that the decision data which is created by the optimisation models is itself analytic data, and can therefore be fed into the chain again. An example of this is the use of prescriptive (optimisation) models to obtain a set of optimum decisions, which are then used as input for a descriptive (simulation) model for risk analysis under alternative scenarios.

The importance of optimisation within the information value chain is supported by the ever growing number of industrial strength applications which are based on OR methods. Besides well established domains such as supply chain management and transportation, other fields traditionally permeated by scepticism about the utility of OR are becoming more and more populated with optimisation based applications. This is due to two factors: the advances in optimisation techniques and tools and the evolution of the IT and information system infrastructure, which greatly facilitate the integration of the OR tools with the firm's data sources. This seamless integration is essential for an effective use of optimisation as a Decision Support System within a firm.

Another observation is that the amount of data (and hence information) which firms can track and maintain has grown exponentially due to the developments in Internet and database technology. The Internet not only provides a heterogeneous communication media, but also an enormous data and information repository. From the IT and computing point of view, the Internet represents the perfect platform for distributed computing. The recent advances in distributed computing based on Internet technology and the introduction of object libraries and software components which

encapsulate the functionality of modelling systems and solvers, have led to the possibility of redesigning the architecture of optimisation-based decision support systems as shown in Figure 4.

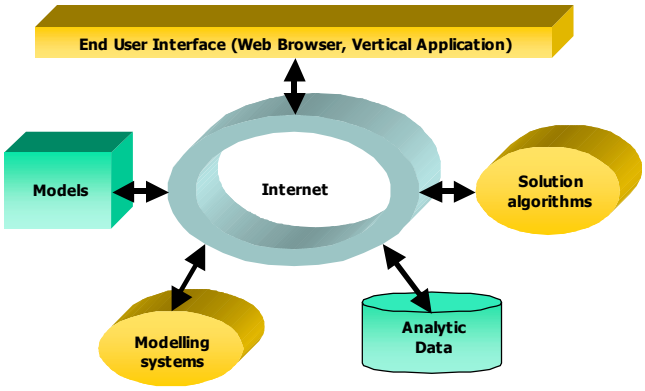

Figure 4. View of a distributed optimisation-based application

Using the Internet as the underlying channel of communication, the traditional mathematical programming system is split and the components can be relocated across the net, leading to the possibility of creating entirely distributed environments for optimisation applications.

2.3 Web based environments for optimisation

The early developments of distributed environments for optimisation took place in the late 1990s, when a number of optimisation server were conceived and implemented. These were based on a client-server architecture and allowed users to submit problems and retrieve solutions using protocols such as *ftp* (file transfer protocol), *smtp* (the email protocol) and even directly on the lower level protocol *TCP/IP*, which is the backbone of the World Wide Web. These first generation of servers rapidly evolved to take full advantage of the power of expression and higher level of communication offered by the World Wide Web and its http protocol. The first innovation was the creation of web pages which served as front end to the remote optimisation servers. For a comprehensive review of these systems, the reader is referred to [20].

Initially, optimisation servers were dedicated to a specific tool; they could be therefore classified as:

- Solver servers
- Modelling servers

Solver servers offered access to commercial or custom implementation of solution algorithms for linear and non linear programming. The users could solve mathematical programming problems by submitting model instances using either standard formats such as MPS or other formats specific to that solver. Modelling servers enabled the users to instantiate mathematical programming models

defined using an algebraic modelling language by submitting to the server a file containing the model definition and the data required by the model. The results were then sent back to the users either via email or could be downloaded from the server using the ftp or the http protocol.

Entire decision support systems based on modelling systems and solvers are also available as server applications: in [7], the authors illustrate a few examples of web-based decision support systems for use in Supply Chain Management, Inventory Control and an e-procurement application for the optimisation of a portfolio of suppliers.

In the remaining part of this section, we illustrate two representative systems for optimisation which are based on Internet technology, but adopt different underlying IT paradigms.

NEOS

The NEOS server for optimisation [8] is probably the best known existing optimisation server. NEOS enables modellers and practitioners to submit their problems using web forms, e-mail or a TCP/IP based client submission tool, and to choose between a large number of solvers. The solvers available on NEOS span the areas of Linear and Integer Programming, Quadratic Programming, Non Linear Programming, Stochastic Programming, and many others. NEOS also supports the three modelling languages AMPL, GAMS and MP-MODEL; these enable the user to submit optimisation problems and data as and let the server instantiate the problems and pass them onto the solvers.

The general architecture of the NEOS server is shown in Figure 5.

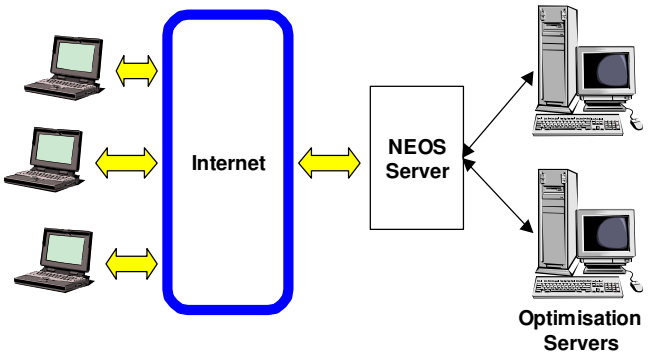

Figure 5. NEOS server architecture

The architecture of NEOS is such that it enables the server to accept and deal with several hundred submissions each week. A server farm (itself distributed across multiple networks) sits behind the NEOS server which receives and deals with the job submissions using a queuing system. In the server farm, multiple server machines run the same solver or modelling system, while the NEOS server takes care of dispatching the jobs to the first available server machine on a first come –first serve basis.

NEOS also enables limited programmatic access to the optimisation tools by supporting the CORBA distributed architecture (www.corba.org). Client program such as *kestrel* [10] take advantage of the CORBA middleware to create a connection between a locally installed AMPL system and a solver which runs remotely on NEOS.

AURORA

The AURORA Financial Management system [44] is a comprehensive project which combines research in the fields of financial planning (in particular using stochastic programming techniques) as well as computer science.

The aim of the AURORA project is to provide a Decision Support System for financial decision making based on optimisation models for portfolio planning, asset liability management as well as other related problems such as the pricing of financial instruments. The optimisation relies on decomposition algorithms for the solution of large dynamic stochastic optimisation problems (linear and convex). The system is based on the multiple server distributed computing paradigm which is better known as *GRID computing* [16], and adopt web services as major building blocks to create DSS.

3 The ASP model and the e-Services paradigm

3.1 Migration from Desktop to Web Based Applications

The desktop applications of early nineties brought computing in general and DSS in particular closer to the "hands on" end user and decision maker. The pervasive use of the Internet has steadily encouraged a migration of many critical applications to the web. This trend is supported by the business model of 'pay per use' or the 'pay as you go' paradigm. Additionally, the attraction of remote servers and distributed computing is easily appreciated as this simplifies the complexity of acquiring and maintaining many versions of software updates. The web-based approach is also able to support business models for online buying and selling of (decision) technologies as services. More generally *work flow systems* and *business rules* are a new genre of management systems which support business process (re)organisation. The distributed computing approach makes it easy to integrate embedded decision support applications such as human resource scheduling or scheduling of assets (vehicles, ships etc) within these work flow systems or make it part of business rules. These web-enabled DSS can be made available over the Internet or deployed within organisational Intranet.

3.2 Application Service Provision (ASP)

The terms ASP and *Application Service Provider* are used to refer to companies providing services via the Internet. The ASP Industry Consortium defines an ASP as an organization that "manages and

delivers application capabilities to multiple entities from a data centre across a wide area network (WAN)." This means a company can rent access to software over the Internet or a WAN for a monthly or pay-as-you-go fee: in essence, software delivered as a service. An ASP employs the personnel needed to install and maintain the application and the servers. Subsequently, ASPs make the application available to customers anywhere in the world via the Internet, either in a browser or through thin client technology. The idea behind ASP is therefore that of *outsourcing*.

In the past few years, a number of products have emerged which enable existing applications to run on remote servers. These include for instance Microsoft's Windows Terminal Server, Citrix's MetaFrame and New Moon's LiftOff. These allow centralized servers to run complex software systems, including full graphical user interfaces, while local client machine can access the applications as if the software had been installed locally. In essence, they provide a sort of "Virtual Desktop". Many of these systems are based on proprietary protocols and on standard web technology for the communication between clients and servers.

The advantage of using systems such as Terminal Server or Citrix Metaframe is obvious if one considers that through these systems almost any existing (desktop) application can instantly published and accessed remotely. However, there are issues related to security and data confidentiality which are not explicitly taken into account by the technology itself, but need to be addressed by the providers of the applications.

3.3 From ASP to e-Services

The original ASP model can be revised and extended to what is known as the xSP concept, that is, the provision of "any" service [30]. For instance, an NSP is a Network Service Provider, whereas an VSP is an Vertical Solution Provider. Most of these business models concentrate on providing some sort of IT service to the customers, and a customer may decide to adopt more than one xSP solution. However, the continuous expansion of the Internet in both terms of number of users and online services has created new marketing opportunities for the service sector, and hence a refined business model. Indeed, Traditional e-commerce is giving way to the new e-Service paradigm. The philosophy of e-Service is the focus on users and customers (meeting their needs precisely and providing them control over the service). While ASP concentrates in the provision of a solution or a tool to the user, an e-Service also provides the entire support and management infrastructure together with the tool. In this respect, e-Services can be seen as a combination of xSP solutions.

Optimisation may fit well the definition of e-Service: one can think of modelling tools and solvers tools as the components offered by the provider, with added infrastructure consisting of secure data storage and data communication, technical support on the usage of the tools, management and

consultancy for the development of user specific models and applications, and some measure of the quality of the optimisation services which are provided.

A particularly serious criticism of ASP is that the service is set up by a third party; hence different constituents (users) have limited capability in terms of customising domain-specific applications. For instance, within the ASP architecture it is not possible fur users to create a typical, say, portfolio planning application. This is because the tools provided are stand alone and cannot be easily brought together and composed in a meaningful way. In section 5 we highlight how using distributed components this obstacle is overcome. Composing an application using web services provided by more than one vendor is viable within e-Services since different vendors can be rewarded through the business model described in the following section.

3.4 A business model for optimisation e-Services

In [45], Rust and Kannan illustrate how traditional e-commerce is giving way to the more general paradigm represented by the e-Services. The use of network technology as a marketing tool, has forced organisations to explore alternative approaches in order to survive the increasing pressure by competing organisations. Customers are now offered (and have instant access to) a range of products and solutions which was simply unthinkable before the establishment of the electronic markets. The immediate outcome is that businesses are now obliged to adopt marketing strategies which are customer-centric, as opposed to goods-centric. Indeed, the recent trend in the global economy is for organisation to promote brands which can be recognised and trusted by customers, as opposed to promoting the actual product which these organisation manufacture [29].

As the attention of businesses moves from product to customer service, so the providers of software applications progress from the ASP model (which is product-focused) to the more general e-Service model. In the specific domain of optimisation software, vendors and providers have started, rather belatedly, to offer web-enabled versions of their systems (see for instance the OSP platform in section 5.1). However, as far as we are aware, there seem to be no full-blown optimisation e-Service yet available. Our work on WEBOPT (section 5.2) aims at embracing the e-Service model for the delivery of modelling tools, solvers and customised/customisable optimisation-based decision support systems.

In [45], the authors analyse the requirements for generic e-Services in terms of marketing strategy and business model. We follow a similar approach and identify a number of issues which need to be addressed in the definition of a sustainable business strategy for optimisation e-Services.

Customer Equity

This concept is based on a bi-directional communication and mutual interests between customers (individuals or organisations) and providers. In the case of decision technology, potential customers should be first persuaded of the increased value arising by the use of optimisation software services remotely with respect to the traditional approach, which involves buying and maintaining local copies of the software. User support can be strengthen to add more value to the customers. On the other hand, the use of optimisation services by the customers in their own applications may generate new markets for the suppliers, particularly in a business to business (B2B) relationship.

Personalisation and customisation.

In an electronic environment, suppliers have the opportunity to gather "transactional" information about the customers. Within an optimisation e-Service, this information can be used in several ways, including the identification of billing models which take into account the usage of the services by the customers. For instance, institutions who use a solver service to optimise the allocation of land in an environmental application (see for instance [5]) may need to solve a large model only very seldom, while students who want to solve their assignment may need to do so several times a day. Customisation helps the provider to establish a fair system of charges which makes the e-Service attractive for different categories of users.

If the e-Service not only offers optimisation tools, but also considers optimisation based decision support systems, personalisation and customisation should enable the users to adapt the underlying decision making model to their own specific instance of the problem. For instance, the OSP supply chain system which is briefly described in section 5.1, allows the users to switch on and of f constraints on the availability of external capacity or inventory, or the possibility of having shortfalls and to carry over unsatisfied product demand to subsequent time periods.

Self-service strategies

One of the most important aspects related to e-Services in general, but also to optimisation e-Services, is the customers wish to gain control of the service. For instance, considering again a solver e-Service, once the user has requested his or her model to be optimised by a given solver, he may want to monitor the status of the solution process, at any point in time. An optimisation e-Service should therefore be built using technology which supports the maximum interaction between the user and the service.

Privacy, and security risk management

Security and privacy are highly relevant in the context of e-Services for decision making. As we have illustrated in section 2.2, optimisation-based decision support systems operate on corporate data

marts which may contain sensitive information. Individuals or a businesses who use a remote decision making e-Service, in one way or another need to send their analytic data across the Internet, and at least temporarily, this data is physically stored on the servers of the service provider; similarly, the decision data generated by the service needs to be sent back to the users. Optimisation e-Services need to address these issues of confidentiality and security, both in terms of technology (encryption, etc.) and in terms of legal matters.

e-Service measurement

e-Services are customer-centric, it is therefore expected that the success of an e-Service depends on customer satisfaction. In the case of optimisation e-Services, the level and quality of the service may depend upon:

- Quality and reliability of the optimisation tools provided
- Quality of the decision support services
- Range of choice amongst different tools offered to the customer
- Reliability and usability of the e-Service as a whole (including server up-time, user support, documentation, international support)

Besides a strong development and Internet support team which can ensure the quality of the technology, the supplier of optimisation e-Services will have to consider the deployment of resources into customer care personnel, which can provide appropriate user support as well as monitoring the level of customer satisfaction by means of surveys and audits.

4 Technology for distributed optimisation

Taking into account the e-Service model, a web-enabled system for optimisation should aim at:

- Provide each single optimisation tool as an independent service
- Enable remote interaction between the optimisation components to provide a complete problem solving environmentCombine the provision of the tools with the provision of user support and securityThis can be achieved by adopting an IT architecture based on *web services*.

4.1 Web services

A web service is defined as a software system running on a server and providing a service by exposing a set of functions and methods. The description of the service is formally and explicitly given so that other software systems (which in this case are referred to as consumers) are able to interact with it. Once the consumer has discovered and understood the service description, the

system can then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols. A Web service accepts a request, performs its function based on the request, and returns a response. The request and the response can be part of the same operation, or they can occur separately, in which case the consumer does not need to wait for a response. Both the request and the response usually take the form of XML, a portable data-interchange format, and are delivered over a wire protocol, such as HTTP. The architecture of a web service is shown in Figure 6.

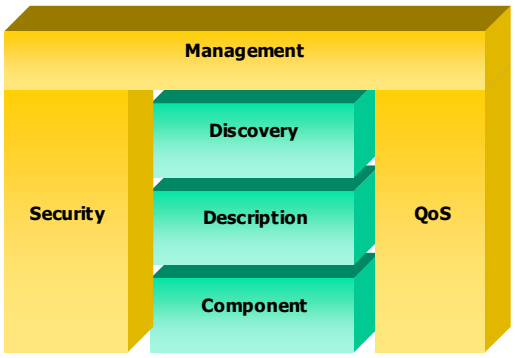

Figure 6. Architecture of a web service

A Web service provides access to an underlying IT component (which in the case of optimisation would be a solver or a modelling system or a data source) and also provides means for the automatic discovery of the service and understanding of the service interface. An overarching layer which includes service management, quality and security completes the architecture. It is easily seen that the IT architecture closely matches the definition of an e-Service: in essence, web services can be considered as the enabling technology for an e-Service. The following languages and protocols are used in conjunction with web services (see :

- XML (eXtensible Mark-up Language) is used as the underlying representation format for the data sent to and from a web service
- SOAP (Simple Object Access Protocol) is based on XML and is used to invoke methods provided by the web services
- WSDL (Web Service Definition Language) is also an XML based language which is used to formally describe the public interface of the service
- UDDI (Universal Description, Discovery and Integration) provides a worldwide registry for advertisement, discovery and integration of web services.

4.2 Data exchange for optimisation web services

In designing distributed optimisation environment based on web services, one needs to consider the issue of data exchange format between the components of the system and between the user and the components. Considering the diagram of Figure 4, the user may wish to access only a modelling system, or only a solver, or both. At least four different items are communicated across the distributed system:

- Algebraic model
- The analytic data required by the model
- The model instance (coefficient of the matrix)
- The results of the optimisation (decision data).

Web services use XML to exchange parameters and request services. It makes therefore sense to create of an XML representation for each one of the four data sets mentioned above. Since year 2000, practitioners and researchers have taken this issue as an opportunity to design a new format for the exchange of mathematical models which can overcome some of the limitations imposed by the current standards. In particular, extensive work has been done to create an XML based format for the representation of model instances. OptML [32], SNOML [36] and FML [21] are alternative approaches in the use of XML based format to replace the MPS format for linear programming and the SMPS format for stochastic programming. Unfortunately, there is no common acceptance of a unique standard.

Some work has also been reported in the use of XML to represent algebraic models prior to the instantiation. A possibility would be the use of MathML [2] developed by the W3C consortium to formulate the expressions which form a mathematical programming model. In [14], the authors present a comprehensive system called Open Optimisation Framework, and propose an XML based format for solution reports called ORML and a format called AML specifically designed to represent algebraic optimisation models. One issue related to the representation of algebraic models is the fact that a new standard should be able to represent at least all models which can currently be formulated by existing algebraic modelling languages. The representational power of languages such as AMPL and the like is not easily matched without major effort in the design of these new formats.

The lack of standard formats for the communication between optimisation tools does not preclude the possibility to implement optimisation web services. In fact, the implementation could simply be based on any of the existing proposition, and be subsequently adapted should a new standard arise.

4.3 Definition of interfaces

The availability of commercial solvers and modelling systems in component form greatly simplifies the implementation of web services which "wrap" these components. However, the interface specifications of the products offered by the various vendors are usually very different from one another. So for instance, the calling interface of CPLEX is very different from that of XPRESS or FortMP, even though the algorithms which are embedded in these solvers (the sparse simplex, interior point method, branch and bound and others) are very similar. Ideally, a solver web service should expose a unique common interface which enables the users to transparently swap between a CPLEX web service and a FortMP web service.

The COIN-OR (Common Infrastructure for Operations Research, www.coin-or.org) is an initiative which seems to address this issue. COIN-OR promotes the development and use of interoperable, open-source software for operations research. The software repository of COIN-OR [47] includes source code of various solution algorithms and libraries of reusable functions for the customisation of these algorithms. In particular, the OSI (Open Solver Interface) is a library of classes of particular interest, in that enables different solvers to be "wrapped" by a common software layer, hence enabling different solution algorithms to be invoked in a unified way. A single WSDL description of solver web services is then easily implemented for alternative solver web services.

4.4 Grid computing

Grid computing is a novel form of distributed computing that involves coordinating and sharing computing, application, data, storage, or network resources across dynamic and geographically dispersed organizations. The innovation of the Grid with respect to "traditional" distributed lies in the orientation towards large-scale application and high performance computing. Indeed, the Grid architecture assumes that every single machine which is connected to the Internet and could be used to build a massive multi-node system. For this reason, the concept of meta-computing is often associated to the Grid. However, the aims of the Grid community go beyond the simple use of massive resources. The architecture of the Grid, as described in [16], is defined at a much higher level of abstraction: the main entity in a grid environment is a Virtual Organisation, which represents a set of individuals or institutions who share resources (computers, software, data etc), in a consumer/provider environment. Virtual Organisations are then "mapped" onto the physical grid of computers, in which the execution of software applications on the nodes and the data exchange are determined by a predefined workflow.

According to Foster, Kesselman et al. [15], Grid services extend the concept of web services by providing a set of well-defined interfaces and that follows specific conventions. These conventions are encapsulated into *service semantics*, and an *ontology* is used to enable services interaction and to

provide a mechanism for service discovery. The open grid service architecture (OGSA) defines the requirements for the integration of the heterogeneous and distributed services which form a Virtual Organisation.

Given that the aim of these concepts is to abstract a collaborative problem solving strategy from the physical computing environment, the approach seems to fit well in the case of optimisation based decision making. It is possible to envisage Virtual Organisations which provide domain specific decision support solutions such as financial planning or supply chain management, as well as direct access to optimisation tools. A possible architecture for such Virtual Organisation is illustrated in Figure 7.

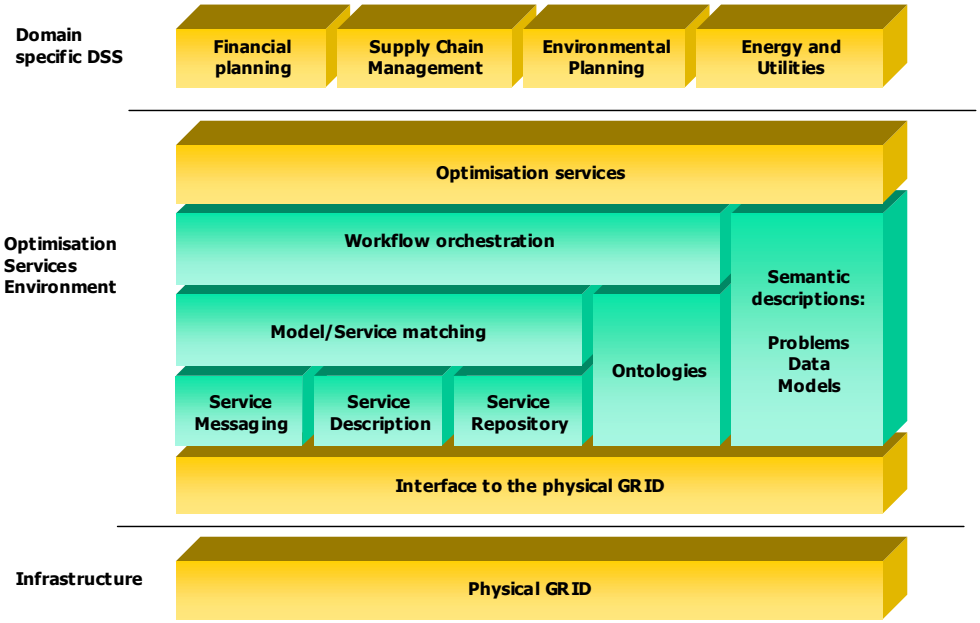

Figure 7. Virtual organisation for optimisation services

The underlying nodes in the physical grid can be used either as a meta-computer, for instance to run parallel decomposition algorithms for stochastic programming, or as a set of software services which are interact in a distributed problem solving environment. Applications of meta-computing to optimisation include amongst others the MetaNEOS project (http://www-unix.mcs.anl.gov/metaneos/), which uses geographically distributed computers (and other resources such as visualization and storage devices) to run massively parallel solution algorithms for large scale optimisation [34],[1]. For other examples of parallel optimisation, see [39].

5 Optimisation Service Provider (OSP) and WEBPOT

5.1 OSP

OSP is acronym for Optimisation Service Provision, a project sponsored by the European Commission [42]. The OSP system adopts the ASP model to provide end users with access to a set of optimisation tools on a subscription basis. These include the modelling systems AMPL and MPL, the commercial solvers CPLEX, OSL, FortMP and FortSP. Besides these tools, OSP also implements two prototype decision support systems:

- A Supply Chain Management system which deploys a stochastic programming model for strategic and tactical planning,
- A Portfolio Optimisation system which makes use quadratic and quadratic mixed integer programming to generate efficient frontiers based on mean variance (Markowitz) models.

The OSP web site (www.osp-craft.com) currently offers free access to the optimisation tools and DSS. Figure 8 shows an example of efficient frontier generated by the OSP Portfolio optimisation. The discontinuity is due to the use of cardinality constraints on the number of assets to be included in the portfolio (3 in this case). The pie chart shows the composition of one of the portfolios on the efficient frontier.

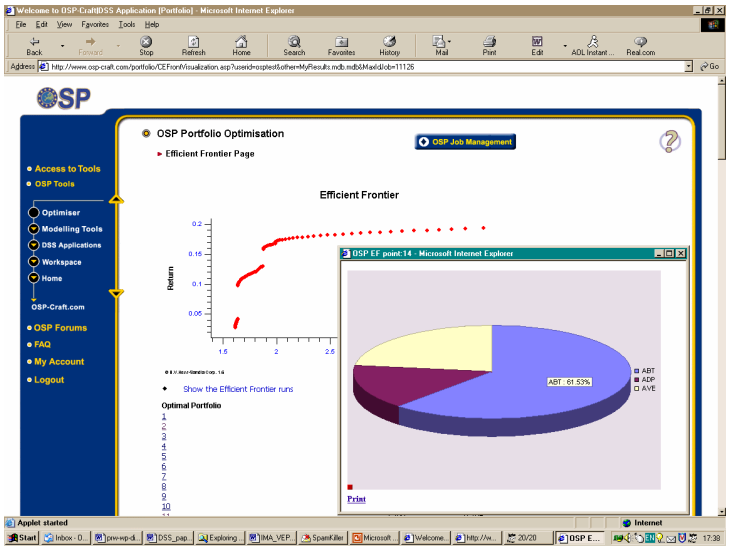

Figure 8. OSP Portfolio optimisation

Figure 9 shows the scenario-dependent profits which the user may expect by implementing the optimal decisions obtained by the supply chain management DSS. The supply chain data and the (optimum) decision data are displayed using OLAP (On Line Analytical Processing), whereby the user can roll-up and drill-down the data in order to obtain different levels of aggregated views.

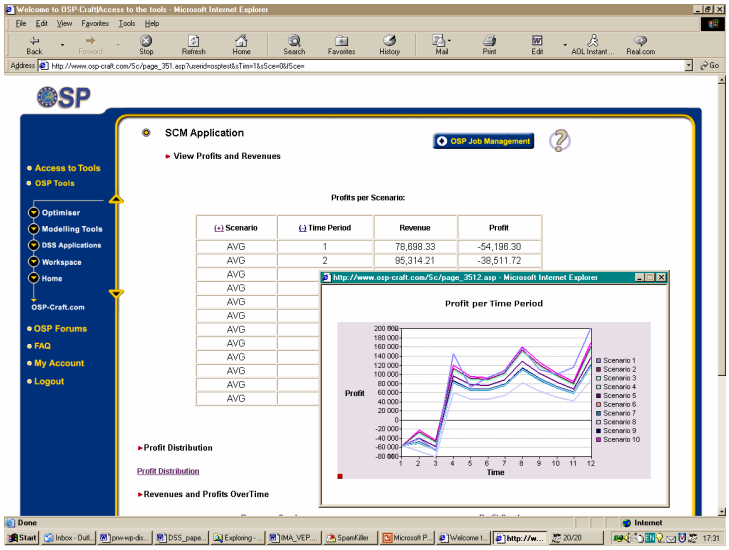

Figure 9. OSP Supply Chain Management

These decision support systems have been developed using a combination of optimisation tools, which include an SQL Server database, the OptiMax 2000 modelling component library, and the FortMP solver.

The architecture of OSP is based on a queuing system built around a database which holds the job requests (the *job queuing, monitoring and notification system*). The web interfaces (clients) send XML records to the database with specific job instructions. The server applications (optimisation tools and decision support systems) are notified of new requests, and produce results in the user's workspace. The web pages are then updated to display the results to the users. Every submission is tracked by a billing system, which implements a flexible model for charging the usage of the applications. Several variables are used in the computation of these charges, including computation time, memory usage, size of models, software royalties and other application specific items such as number of stocks in the portfolio optimisation DSS and number of scenarios in the supply chain management DSS. The diagram of Figure 10 shows the overall architecture of the OSP system.

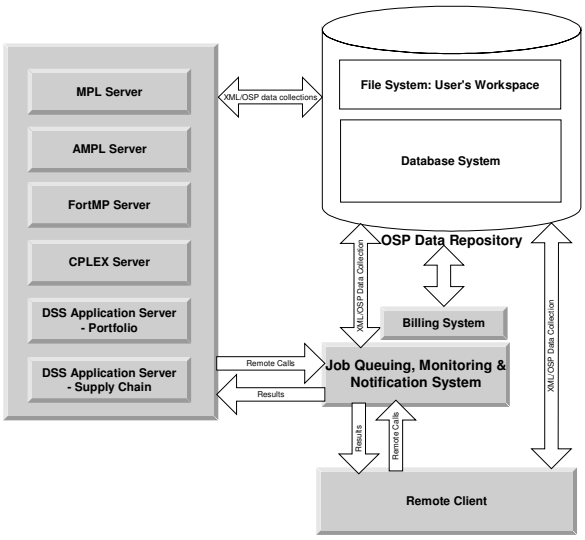

Figure 10: OSP architecture.

Besides this functionality, the OSP platform provides an information infrastructure which is designed to maximise the effectiveness of its use:

- *Support Services.* Due to the complexity and sophistication of optimisation applications and the resulting DSS, it is common for a user to require the help of an expert in order to understand, use, or customise an optimisation-based DSS to his requirement. Support services are available in the form of on-line help, on-line tutorials, or physical interaction (discussion) with an expert.

- *Technical Support.* Issues strictly relating to the system's functions, rather than the use of the DSS, are resolved through a technical support help-desk.

- *Trainings.* OSP also provides training material such as an introduction to mathematical programming and an introduction to stochastic programming.

The availability of these tutorials and user support documents, which are translated into five European languages, also makes OSP valuable as an educational tool. As a pilot scheme, the OSP platform has been deployed in academic institutions such as the CARISMA Centre of Brunel University, UK to create OptiLab: an optimisation laboratory where the OSP tools and applications are used for teaching purposes (www.carisma.brunel.ac.uk). The OSP service has also been used for teaching purposes in the University of Milano Bicocca, Italy, in Jadavpur University in Calcutta, India, IIT Bombay, India, and Vienna University, Austria.

5.2 WEBOPT

The EU-sponsored WEBOPT project (www.webopt.org) sets out to create a network of research scientists and develop an integrated optimisation software environment to be used as web enabled

products and services. The target groups are made up of academic researchers in leading technical and management institutes, faculty of regional technical colleges, SME suppliers of software tools, practitioners within the IT and management teams of public and private sector industries. The aim is to acquire research results from European, American as well as Indian projects and use optimisation as an analytical engine in decision support systems in a number of industrial sectors, with applications in portfolio planning, supply chain management, agricultural and energy planning.

WEBOPT is a superset of OSP, since it provides direct access to the existing OSP applications via an improved web interface. However, WEBOPT moves from the basic ASP model towards an e-Service model by deploying web services. These can be seamlessly integrated to create optimisation-based distributed systems and to outsource computational resources from the user's desk to a remote server.

The provision and sharing of domain knowledge via trainings and user documentation is envisaged as the overarching support infrastructure of WEBOPT.

Figure 11. The WEBOPT portal.

Figure 11 shows the main page of the WEBOPT portal. Registered users may use this web interface to access the optimisation applications and the web services.

WEBOPT architecture

The applications in WEBOPT can be accessed via a presentation layer which implements a predefined set of web interfaces. This mode of use is based on the basic ASP model supported by OSP. The innovation in WEBOPT consists in the provision of web services which can be accessed using the SOAP/ XML protocol. The overall architecture of the WEBOPT platform is shown in Figure 12.

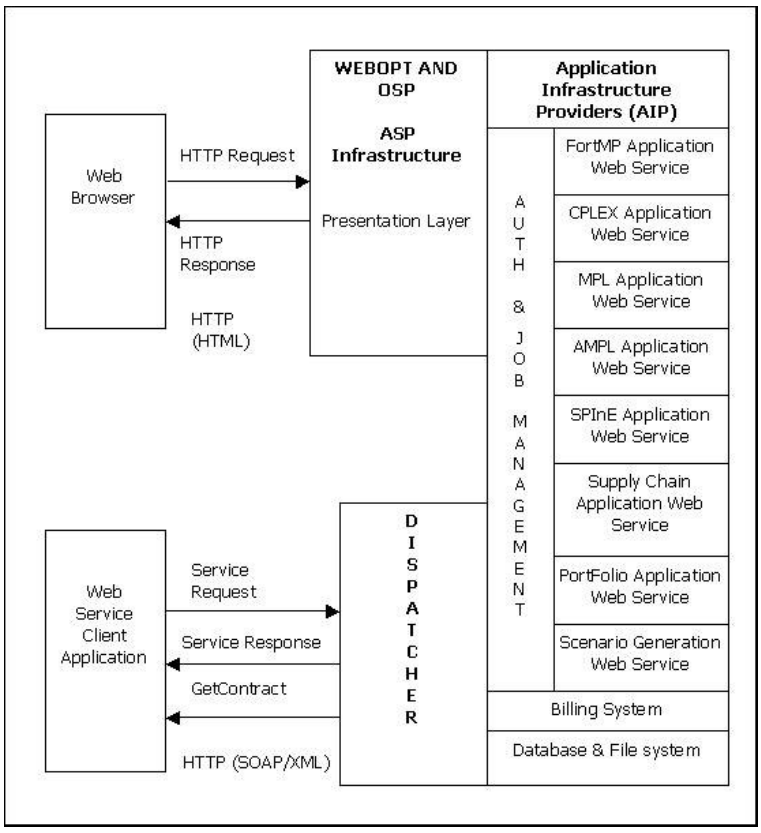

Figure 12. WEBOPT architecture

Requests to the web services are orchestrated by the Authorisation and Job Management service (AJM), which is in turn a web service. The AJM takes care of authenticating the users who send requests to the web services, and instantiates a new job record, which is used to track all the subsequent requests by the same user. These transactions are used to monitor the usage of resources, and as indicators of the quality of service (for instance, by measuring the time elapsed between a user's request and the fulfilment of the request). A commercial exploitation of the WEBOPT technology could also use the transaction data to devise appropriate charges for the usage of the web services.

The web services provided by WEBOPT are grouped into:

- Scenario Generators (SG)
- Modelling Systems (MS)
- Solvers (SOL)

The web services are implemented using Microsoft .NET technology. Each group exposes a set of methods, whose WSDL definition is automatically generated by the .NET framework. The methods exposed by the SOL service are shown in Table 1, together with a section of the WSDL description.

Table 1. Interface for the SOL web service.

A client application invokes the web services using the following mechanism (we use the solver web service SOL as an illustrative example):

1) Obtain a unique key from the AJM by providing a valid user name and password. The key is then used in all subsequent method calls.

2) Invoke the *init* method of the SOL service using the key. SOL will in turn call the AJM to authenticate the key and obtain a job id which will be used to record all transactions.

3) Invoke the *setParameters* and *setModel* methods to set solver controls and specify the model instance which needs to be solved.

4) Invoke the *solve* method. This starts the actual solution algorithm in either synchronous or asynchronous mode. In the first case, the web service call only returns when the solution process is completed. In the second case the client needs to regularly call the *getStatus* method to check whether the solution process is completed.

5) Invoke the *getResults* method to obtain the optimal solution vectors from the web service.

6) Clear the web service cache by calling the *clear* method.

Composition of web services

The infrastructure provided by WEBOPT enables the creation of distributed applications which may take advantage of multiple specialist knowledge. For instance, a financial planning application may be constructed by joining expertise of:

- Financial data analysts

- Stochastic programming modelling experts
- Solution algorithm experts.
- Managers/Decision makers

These four parties can develop highly specialised sub-systems using relevant optimisation web services which can then be combined into a distributed decision support solution. As an example, consider a financial planning application for asset and liability management.

1) The financial data analysts use live data feed to retrieve historical data regarding some securities of interest. The data is then used with the SG web service to generate a number of scenarios for the price of the securities. The scenarios are in turn processed by the analyst who fine tunes the scenario generator web service to produce a new set of scenarios with well defined mathematical properties and which also incorporate some expert knowledge.

2) The modelling expert implements a stochastic programming model for asset and liability management using an algebraic modelling language. The modeller uses increasingly large sets of virtual (fictitious) or real data to test the scalability of the model and calls the MS web service to instantiate the model. The modeller may also access the SOL web service to test whether the model instances provide the correct or realistic optimal solutions.

3) A solver expert uses the model instances to fine-tune the solution algorithm provided by the SOL service. For instance, basis saving and restart in algorithms such as Benders decomposition may be beneficial and improve the performance of the solution algorithm if the structure of the sub-problems does not change.

4) The decision maker assesses and investigates the model's solutions via an integrated user interface which hides the complexity of the DSS application which is constructed in a distributed components framework.

This is a typical example in which the e-Service business model, which is described in section 3, proves more appropriate than the ASP model, since the different "customers" make a very different use of the tools, yet they need their solutions to be able to communicate in order to achieve the common goal of creating a financial planning application.

6 Conclusions

In this paper, we have analysed the features of the optimisation tools which are typically deployed within decision support systems, and have shown how these tools have evolved from "desktop" programs to software components. These components greatly facilitate the development of vertical

applications and decision support systems. We have also illustrated that, by combining web technology with these software components, optimisation tools and optimisation based applications can be transformed into distributed applications, which can be delivered to users as remote services.

An effective and successful optimisation based decision support system requires not only to be fully integrated with the information system of an organisation, but also to be embedded within the organisation's business process management. As companies move on fully to web-enabled information systems, the new generation of DSS must embrace this distributed architecture, and must therefore be built using web-enabled components (web services).

We have investigated optimisation services covering both the technology requirements, and the business models which can be adopted to support their provision across the Internet. In particular, web services and grid computing environments are considered viable technological platforms. From the business model point of view, we have discussed how the ASP model is extended to the more general e-Service paradigm, whereby the focus is moved from the provision of goods (in this case optimisation software tools and decision support systems) to a more customer-centric view. A major benefit of the formalisation of service descriptions is that e-Services lead to B2B relationships of higher propensity, and we believe that the adoption of this business model will create many examples of effective integration of decision technology within organisational processes. As practical examples of the ASP model and the e-Services model in the context of optimisation, we have introduced the features of the OSP and the WEBOPT portals respectively. These platforms enable users to access either individual optimisation tools such as solver and modelling systems, or entire decision support systems. While OSP does not support important requirements such as service discovery, service management and quality measurements, we aim to introduce these in WEBOPT to create a true e-Service for optimisation.

Finally, our experience with the above two projects have been reinforced by the findings in [28] that adopting an open architecture, addressing the clients security and privacy issues and maintaining the investment of successful legacy applications are important aspects of an acceptable e-Service.

7 Acknowledgments

The authors would like to thank the EU Framework 5 programme for funding the OSP project and EuropeAid for funding WEBOPT, which is an ASIA IT&C project. In particular we would like to thank Nikitas Koutsoukis and Triphonas Kyriakis for their leading role in OSP and Chandra Poojari for his role and support in the WEBOPT project. Thanks are also due to Uddipta Das for his contribution to the development of the user interface and client services within OSP as well as

WEBOPT. We also acknowledge Sudipta Das for his contribution in the architecture design and system implementation of the OSP and WEBOPT projects.

8 References

[1] Anstreicher, K., N. Brixius, J.P. Goux, and J. Linderoth, Solving large quadratic assignment problems on computational grids, Mathematical Programming 91, No. 3 (2002) 563-588.

[2] Ausbrooks, R., S. Buswell, D. Carlisle, S. Dalmas, S. Devitt, A. Diaz, M. Froumentin, R. Hunter, P. Ion, M. Kohlhase, R. Miner, N. Poppelier, B. Smith, R. Soiffer, N. Sutor, and S. Watt, Mathematical Markup Language (MathML) Version 2.0 (Second Edition), W3C, http://www.w3.org/TR/MathML2/.

[3] Bhargava, H.K., R. Krishnan, and R. Muller, Decision support on demand: Emerging electronic markets for decision technologies, Decision Support Systems 19, No. 3 (1997) 193-214.

[4] Bisschop, J.J. and M. Roelofs, AIMMS: The Language Reference, (Paragon Decision Technology BV, Haarlem, The Netherlands, 1999).

[5] Bosetti, V., E. Messina, and P. Valente, Optimization technologies and environmental applications, Ima Journal of Management Mathematics 13, No. 3 (2002) 167-186.

[6] Brooke, A., D.A. Kendrick, A. Meeraus, and R. Raman, GAMS : a user's guide, Release 2.50, (1998).

[7] Cohen, M.-D., C.B. Kelly, and A.L. Medaglia, Decision support with web-enabled software, Interfaces 31, No. 2 (2001) 109-129.

[8] Czyzyk, J., M.P. Mesnier, and J.J. Moré, The NEOS Server, IEEE Journal on Computational Science and Engineering 5, No. 3 (1998) 68-75.

[9] Dash Associates, XPRESS-MP, User Guide, (1999).

[10] Dolan, E.D., R. Fourer, J.-P. Goux, and T.S. Munson, Kestrel: An Interface from Modeling Systems to the NEOS Server, Optimization Online, www.optimization-online.org, (2002).

[11] Dominguez-Ballesteros, B., G. Mitra, C. Lucas, and N.-S. Koutsoukis, Modelling and Solving Environments for Mathematical Programming (MP): A status review and new directions, Journal of the Operational Research Society 53, No. 10 (2002) 1072-1092.

[12] Ellison, E.F.D., M. Hajian, R. Levkovitz, I. Maros, G. Mitra, and D. Sayers, FortMP Manual, (OptiRisk Systems and Brunel University, 1999).

[13] Ellison, E.F.D. and G. Mitra, UIMP: User Interface for Mathematical Programming, ACM Transactions on Mathematical Software 8, No. 3 (1982) 229-255.

[14] Ezechukwu, O.C. and I. Maros, OOF: Open Optimisation Framework, Imperial College, Departmental Technical Report 2003/7, (April 2003).

[15] Foster, I., C. Kesselman, J.M. Nick, and S. Tuecke, Grid Services for Distributed System Integration, Computer 35, No. 6 (2002) 37-47.

[16] Foster, I., C. Kesselman, and S. Tuecke, The anatomy of the grid: Enabling scalable virtual organizations, International Journal of Supercomputer Applications 15, No. 3 (2001).

[17] Fourer, R., Database structures for mathematical programming models, Decision Support Systems 20, No. 4 (1997) 317-344.

[18] Fourer, R., Linear Programming Survey Solver or modeling: OR tool can take different approaches to reach common goal, Or Ms Today 28, No. 4 (2001) 58-59.

[19] Fourer, R., D.M. Gay, and B.W. Kernighan, AMPL: a modeling language for mathematical programming. Scientific Press, series, (Danvers Mass Boyd Fraser Pub. Co, 1993).

[20] Fourer, R. and J.-P. Goux, Optimization as an Internet Resource, Interfaces 31, No. 2 (2001) 130-150.

[21] Fourer, R., L. Lopes, and K. Martin, A W3C XML schema for linear programming, Northwestern University and The University of Chicago.

[22] Geoffrion, A.M. and R. Krishnan, Prospects for Operations Research in the E-Business Era, Interfaces 31, No. 2 (2001) 6-36.

[23] Hürlimann, T., LPL: A Modelling Language., Modelling Tools for Decision Support, Series in Computer Science, University of Fribourg, 1993).

[24] IBM, IBM Optimization Library Guide and Reference, (2001).

[25] IBM World Trade Corporation, IBM Mathematical Programming System Extended/370 (MSPX/370), IBM Program Reference No. SH19-1095-1.

[26] ILOG Inc., ILOG CPLEX 8.0 User Manual, (2002).

[27] ILOG Inc., ILOG OPL Studio User's Manual, (1999).

[28] Kleijnen, S. and S. Raju, An Open Web Services Architecture, ACM Queue 1, No. 1 (2003).

[29] Klein, N., No logo,(Flamingo, London, UK, 2001).

[30] Koutsoukis, N.-S. and G. Mitra, Decision modelling and information systems: The information value chain, (Kluwer Academic Publishers, 2003).

[31] Koutsoukis, N.-S., G. Mitra, and C. Lucas, Adapting on-line analytical processing for decision modelling: the interaction of information and decision technologies, Decision support systems 26, No. 1 (1999) 1-30.

[32] Kristjansson, B., How New Technologies such as XML and SOAP allow OR to provide Web-based Services, in: INFORMS Roundtable Savannah, Georgia, (2001).

[33] Kuip, C.A.C., Algebraic Languages for Mathematical Programming, European Journal of Operational Research 67, No. 1 (1993) 25-51.

[34] Linderoth, J. and S. Wright, Decomposition Algorithms for Stochastic Programming on a Computational Grid, Computational Optimization and Applications 24, No. 2 (2003) 207-250.

[35] Lindo Systems Inc., LINGO Modelling System Version 4.0, (1998).

[36] Lopes, L. and R. Fourer, An XML-Based Format for Sharing Mathematical Programs, in: INFORMS international conference on Miami, (2001).

[37] Lucas, C. and G. Mitra, Computer-Assisted Mathematical-Programming (Modelling) System - CAMPS, The Computer Journal 31, No. 4 (1988) 364-375.

[38] Maximal Software, MPL Modelling System, Release 4.2, (USA, 2002).

[39] MirHassani, S.A., C. Lucas, G. Mitra, E. Messina, and C.A. Poojari, Computational solution of capacity planning models under uncertainty, Parallel Computing 26, No. 5 (2000) 511-538.

[40] Mitra, G., Tools for modelling support and construction of optimisation applications, in Impacts of Recent Computer Advances in Operations Research, Editors Ramesh Sharda, Bruce Golden, Edward Wasil, Osman Balci, William Stewart, (1989), 484-496, North Holland.

[41] Mitra, G., H.J. Greenberg, F.A. Lootsma, M.j. Ricckaert, and H.J. Zimmerman, Eds. Mathematical Models for Decision Support, NATO ASI Series F: Computer and Systems Sciences, Vol. 48, (Springer-Verlag, 1988).

[42] Mitra, G., T. Kyriakis, N. -S. Koutsoukis, and P. Valente, Optimization Service Provision: A European Union Project, in: INFORMS. Miami, USA, (2001).

[43] Mitra, G., C. Lucas, S. Moody, and B. Kristjansson, Sets and Indices in Linear Programming Modelling and their Integration with Relational Data Models, Computational Optimization and Applications 4, No. 3 (1995) 263.

[44] Pflug, G.C., A. Swietanowski, E. Dockner, and H. Moritsch, The AURORA Financial Management System: Model and Parallel Implementation Design, Annals of Operations Research 99, No. 1 (2000) 189-206.

[45] Rust, R.T. and P.K. Kannan, E-Service: A New Paradigm for Business in the Electronic Environment, Communications- Acm 47, No. 6 (2003) 36-42.

[46] Sadki, M., P. Valente, G. Mitra, and R. Fourer, Strategies for interacting with algebraic modelling languages, in: EURO/INFORMS. Istanbul, Turkey, (2003).

[47] Saltzman, M.J., COIN-OR: An Open-Source Library for Optimization, in: S.S. Nielsen, Ed., Programming Languages and Systems in Computational Economics and Finance. (Boston MA; Kluwer Academic Publishers, 2002) 3-32.