SCHEDULING DECISION-MAKING USING WEB SERVICE

Maria Leonilde Rocha Varela, Sílvio do Carmo Silva
Department of Production and Systems, University of Minho, Azurém[*]/ Gualtar[+] Campus ,Guimarães[*]/ Braga[+], Portugal

Joaquim Nunes Aparício
Department of Computer Science, New University of Lisbon,Caparica, Portugal

Keywords: Production Scheduling, XML Modeling, Web Service and remote methods invocation.

Abstract: In this paper we make a contribution for scheduling problems solving through the web, by using web service. The XML-based modeling and communication is applied to the production scheduling activity. Therefore, scheduling concepts, like manufacturing problems and solving methods, including corresponding inputs and outputs are modeled using XML. This kind of data modeling is used for building our web-based scheduling decision support system, which works as a web service under the XML-RPC protocol. This protocol is used for invoking the implemented methods, for solving problems defined by the user, which are local or remotely available through the Internet. New methods can be continuously incorporated in the system's distributed repository in an easy and interactive way.

1 INTRODUCTION

The scheduling activity in an organization seeks to optimize the use of available production means or resources, ensuring short time to complete jobs and, in addition, to satisfy other important organization objectives. Thus, it can highly contribute to good service to customers and to high profitability of an organization.

Production Scheduling may be defined as the activity of allocating tasks to production resources, or vice versa, over time. The result of this is usually expressed in a production schedule.

With this work we make a contribution for the better resolution process of scheduling problems by means of a web-based decision support system. This system has been designed and implemented as a web service using the XML-RPC protocol and requires, first of all, the specification and identification of each problem to be solved, followed by the access to resolution methods, which are available for solving them. When there are different methods available we can obtain alternative solutions, which should be evaluated against specified criteria or objectives to be reached. Thus, we are able to properly solve a problem through the execution of one or more scheduling methods and, subsequently, select de best solution provided by them. These methods can either be local or remotely available and accessible through the Internet.

In this paper a classification framework for scheduling problems and related concepts is used, which is represented through XML (extensible markup language). This kind of data modeling allows, for instance, specifying scheduling problems and identifying methods for their resolution and to establish the necessary communication for the execution of the implemented scheduling methods through the web.

This paper is organized as follows. The next section describes the nature of scheduling problems and the classification model used. Section 3 starts with a general outline of the web system architecture then goes on dealing with the main system functionalities, which are related to methods searching, methods execution and new methods insertion. Section 4 briefly refers to some related work and finally, section 5 presents some conclusions.

Leonilde Rocha Varela M., do Carmo Silva S. and Nunes Aparício J. (2004).
SCHEDULING DECISION-MAKING USING WEB SERVICE.
In *Proceedings of the First International Conference on E-Business and Telecommunication Networks*, pages 169-176
DOI: 10.5220/0001399601690176

2 SCHEDULING PROBLEMS

2.1 Characterization Nomenclature

Due to the existence of many different scheduling problems there is a need for a formal and systematic manner of problem representation that can serve as a basis for their classification. A framework for achieving this was developed by Varela et al. (2002a and 2002b), based on published work by Conway (1967), Graham et al (1979), Brucker (1995), Blazewicz (1996), and Jordan (1996), as well as on other information presented by (Morton, 1993) and by other authors namely (Artiba, 1997), and (Pinedo, 1995). This framework allows identifying the underlying characteristics of each problem to be solved, and is used as a basis for the XML-based problem specification model used in this work.

The referred framework for problem representation includes three classes of notation parameters for each corresponding class of problem characteristics. The first class of characteristics, the α class, is related with the environment where the production is carried out. It specifies the production system type (α_1) and, eventually, the number of machines that exist in the system (α_2). The other classes allow specifying the interrelated characteristics and constraints of jobs and production resources, which is expressed by the class β (β_1 … β_{14}) of parameters, and also the performance criterion represented by class γ. Some important processing constraints are imposed by the need for auxiliary resources, like robots and transportation devices and/or the existence of buffers, among others factors. The evaluation criteria, the third class of parameters, may include any kind of performance measure, including multi-criteria measures. More information about this three-field problem classification nomenclature is referred, for example, in Varela et al. (2002a, 2002b and 2002c).

An example of use of this notation is “F2|n|Cmax” which reads as: “Scheduling of non-preemptable and independent tasks of arbitrary processing time lengths, arriving to the system at time zero, on a pure flow shop, with two machines, to minimize the maximum completion time or makespan (Cmax).

2.2 Class Model Description

Figure 1 below shows a general UML (Unified Modeling Language) class diagram about the scheduling problems model.

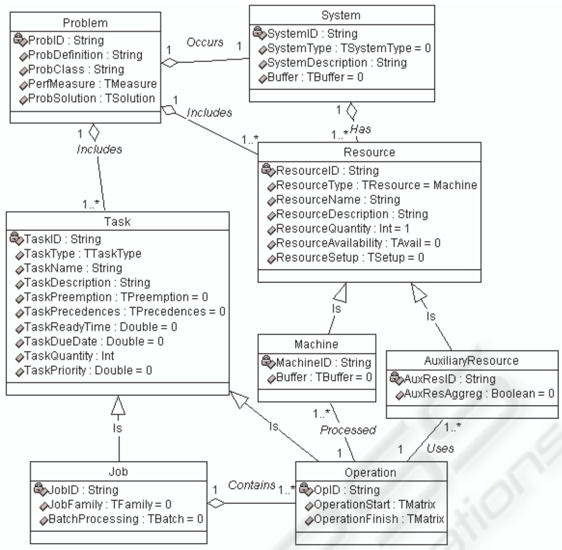

Figure 1: Class diagram of scheduling problems model.

There are eight main classes, related to the problem itself, as well as to the system, where the manufacturing process is carried out, and its resources. These include the processors or machines and other auxiliary resources. Two additional classes representing each task related to jobs and its operations are also shown. In this diagram, the default values are usually set to zero. Therefore, for example, for the SystemType attribute, in the System class, the default value zero corresponds to the single machine production system that is defined in TSystemType data type. This also includes attributes for other systems, such as parallel machines, flow shops, open shops, job shops and mixed shops, under the problem classification factor α_1, as previously described in section 2.1.

3 PROBLEMS SOLVING

3.1 System Architecture

Figure 2 below illustrates a general outline of the web system architecture. This web system integrates a distributed knowledge base for supporting the manufacturing scheduling decision-making process. Therefore, through a web interface each methods’ server owns his particular knowledge base component, which enables searching information about scheduling problems and corresponding solving methods. These methods can be local or remotely available and accessible through the Internet, by any of those available methods’ servers, which are organized in a P2P (peer-to-peer) network, forming a distributed knowledge repository (Papazoglou, 2003).

Moreover, the web system enables new knowledge insertion, including validation, and transformation of manufacturing scheduling data about problems and methods that can be put forward in a user-friendly way. This interface is mainly controlled by DTD (document type definition) and XSL (extensible stylesheet language) documents stored in the repository of each peer of the network. The scheduling information is also stored in XML documents and these documents are verified using DTDs, before being included in the corresponding XML repository.

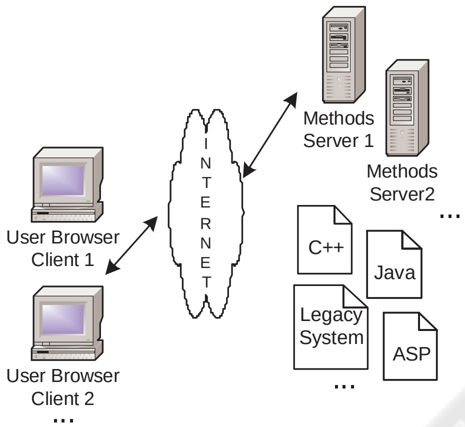

Figure 2: Web-system architecture.

Scheduling problems as well as related concepts, namely methods and its implementations, including the definition of its inputs and outputs are specified by a DTD. Elements on the problem DTD precisely characterize a scheduling problem, meaning that in order to interact with the system a problem must be described according to that grammar (2002a, 2002b and 2002c).

Some interesting XML applications, which are more or less related with this work, are PDML (Product Data Markup Language), RDF (Resource Description Format) and STEPml (Harper, 2001). Other XML specifications devoted to manufacturing processes are JDF (Job Definition Format), PSL (Process Specification Language), PIX-ML (Product Information Exchange), PIF (Process Interchange Format) and XML-based workflow (Abiteboul et al, 2000).

This web system encompasses several main functionalities, which include knowledge insertion, about scheduling problems and resolution methods, and correspondent information searching. Users can make requests for visualizing scheduling problem classes and methods information or even browse information about other concepts presented by the system. The data can be shown in different views, using existing XSL documents, adequate for each specific visualization request. Another important functionality is the execution of scheduling methods, given the manufacturing scheduling problem definition. The selection of one or more specific methods is made through a searching process on the distributed knowledge base of scheduling methods. The system also enables problem results presentation and storage.

3.2 Methods Updating

Many implementations of a given method may be accessible through the Internet. From the point of view of the web system two implementations of the same method may differ if, for example, they differ on its outputs. Moreover, not all implementations work in the same way. Therefore, for the system to be able to use such implementations in a programmatic way, they must also be described within the system. This description must include, among other things, the address to the running method or program and its signature, which includes the definition of the parameters that are necessary for its execution, i.e. the inputs, and its output format. All this information is described in a corresponding DTD file (Varela et al., 2002a and 2002b). Figure 3 illustrates the system interface for defining the method's signature for an implementation of the branch and bound method proposed by Ignall and Schrage in 1965 for a flow shop problem.

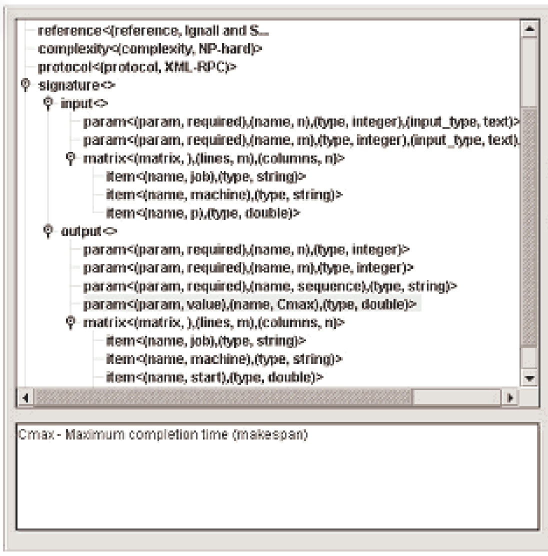

Figure 3: Method signature definition.

The inputs include the definition of a parameter n for number of jobs to be processed, a parameter m for number of machines in the production system, and a set of three items in a matrix structure, which represent the job name, the machine name and an additional parameter p that corresponds to job processing time. There is also the definition for the method's output following the same lines. This information, once validated using the corresponding DTD, is subsequently inserted in an XML document in order to enable repeated executions of methods, information retrieval and automatic generation of interfaces for the implemented methods' inputs and outputs.

3.3 Methods Searching

Scheduling problems belong to a much broader class of combinatorial optimization problems, which, in many cases, are hard to solve, i. e. are NP-hard problems (Ceponkus, 1999; Jordan, 1996; Blazewicz, 1996; Brucker, 1995). In presence of NP-hard problems we may try to relax some constraints imposed on the original problem and then solve the relaxed problem. The solution of the latter may be a good approximation to the solution of the original one. Many times we do not have a choice and have to draw upon what we may generally call approximation methods (French, 1982). These include both, those that we know how near their solutions may be from optimum ones and also a variety of heuristic methods, including those based on meta-heuristics, which are likely to achieve good solutions.

Examples of this kind of methods are approximate dynamic programming or branch and bound methods. Other approaches to obtain good or at least satisfactory solutions, in acceptable time, are based on the nowadays widely used local or neighborhood search techniques, such as Genetic Algorithms (GA), Simulated Annealing (SA), and Tabu Search (TS), which are also known as meta-heuristics and extended neighborhood search techniques (Osman, 1996; Arts, 1997). There are also other interesting types of scheduling approaches widely used like simulation-based approaches or bottleneck methods or even neural network or petrinet-based approaches, among others. Methods based on these heuristic or approximate approaches tend to provide good results in the available time to make decisions.

The system can be used for searching about suitable methods for solving particular scheduling problems, once classified through the previously described characterization nomenclature. In the distributed methods knowledge base the system records for which problem class a certain method may be used. If this information is not available the user is encouraged to relax some of the problem constraints in order to discover some other methods for solving it. Figure 4 shows a small sample of methods assigned to solve closely related scheduling problem classes and Figure 5 presents a list of available methods for solving some selected problem types, which are generally characterized by occurring in flow shop systems and by having as performance measure the maximum completion time or makespan.

<table>
<tr><th colspan="5">Problem types for the general problem characteristics: [system_type=F, jobs=n, measure=Cmax]</th></tr>
<tr><th>Class</th><th>Complexity</th><th>Context</th><th>Problem characteristics</th><th>Select</th></tr>
<tr><td>F2|n|Cmax</td><td>Maximal Polinomially Solvable</td><td>≥</td><td>[(system_type, F), (machines, 2), (jobs, n), (measure, Cmax)]</td><td>☑</td></tr>
<tr><td>F2|rj,n|Cmax</td><td>Minimal NP-hard</td><td>≥</td><td>[(system_type, F), (jobs, n), (arrivals, rj), (machines, 2), (measure, Cmax)]</td><td>☐</td></tr>
<tr><td>F2|rj,n,no-wai</td><td>Maximal Polinomially Solvable</td><td>≥</td><td>[(system_type, F), (machines, 2), (jobs, n), (arrivals, rj), (buffers, no-wait), (measure, Cmax)]</td><td>☐</td></tr>
<tr><td>F3|rj,n|Cmax</td><td>Minimal NP-hard</td><td>≥</td><td>[(system_type, F), (machines, 3), (jobs, n), (arrivals, rj), (measure, Cmax)]</td><td>☐</td></tr>
<tr><td>F3|n|Cmax</td><td>Minimal NP-hard</td><td>≥</td><td>[(system_type, F), (machines, 3), (jobs, n), (measure, Cmax)]</td><td>☑</td></tr>
<tr><td>Fm|n|Cmax</td><td>Minimal NP-hard</td><td>≥</td><td>[(system_type, F), (machines, m), (jobs, n), (measure, Cmax)]</td><td>☑</td></tr>
<tr><td>Fm|prec,pji=1,</td><td>Minimal NP-hard</td><td>≥</td><td>[(system_type, F), (machines, m), (jobs, n), (precedences, prec), (times, pji=1), (measure, Cmax)]</td><td>☐</td></tr>
</table>

Figure 4: Problem classes selection.

<table>
<tr><th colspan="8">Available methods for solving the problem classes: [class=F2|n|Cmax, class=F3|n|Cmax, class=Fm|n|Cmax]</th></tr>
<tr><th>ID</th><th>Prob. Class</th><th>Method</th><th>Problem characteristics</th><th>Reference</th><th>Complexity</th><th>Protocol</th><th>Select</th></tr>
<tr><td>1001</td><td>Fm|n|Cmax</td><td>BranchBoundIS</td><td>[(system_type, F), (machines, m), (jobs, n), (measure, Cmax)]</td><td>Ignall and Schrage, 1965</td><td>NP-hard</td><td>XML-RPC</td><td>◉</td></tr>
<tr><td>32</td><td>F2|n|Cmax</td><td>Johnson</td><td>[(system_type, F), (machines, 2), (jobs, n), (measure, Cmax)]</td><td>Johnson, 1954</td><td>Polynomial</td><td>XML-RPC</td><td>○</td></tr>
</table>

Figure 5: Methods' implementations for solving selected problem classes.

Many scheduling methods may be more or less adequate to solve a given class of problems and the system gives some detailed information in order to assist the scheduling decision-making process.

The system is able to quickly assign methods to problems that occur in real world manufacturing environments and solve them through the execution of one or more appropriate solving methods, whose implementations are local or remotely available and accessible through the Internet, by using a certain communication protocol for remote methods invocation.

3.4 Methods Execution

The system here described has been designed and implemented as a web service (http://www.w3.org) using the XML-RPC protocol (Laurent et al., 2001) for remote methods invocation (Varela et al., 2003a and 2003b).

The primary motivation for this work is to be able to provide a service that, for a given set of parameters describing an actual scheduling problem, i.e. a problem instance, returns a set of output values which may be accepted as a solution for the problem introduced, Figure 6.

To solve this task some intermediate steps need to be accomplished. In addition to information already illustrated and explained in the previous section we provide users with access to intermediate knowledge about methods and its implementations to better support the scheduling decision-making process.

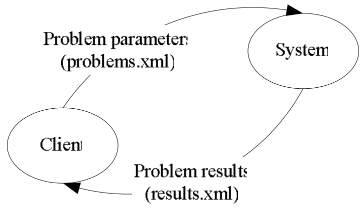

Figure 6: Web service.

A web service promises easy access to remote content and application functionality using standard mechanisms, without any dependency on the provider's platform, the location, the service implementation, or the data format. In a web service a certain method accepts as input a problem definition and returns a result in some particular form.

Different implementations may provide results in different formats, and the system must have a description of them in order to format them according to the problem output to be returned to the client as the very last step of the service. The result from running a method implementation on the given problem instance can then be delivered to the client as an XML file and/or can be transformed into some expressive output, like a Gantt chart.

Web services use XML to encode both the message wrapper and the content of the message body. As a result, the integration is completely independent of operating system, language or other middleware product used by each component participating in the service. The only fundamental requirement is that each component has the ability to process XML documents and that each node connected in a distributed system supports HTTP as a default transport layer.

The XML-RPC protocol is the sequence and structure of requests and responses required to invoke communications on a remote machine. Several other protocols that could also be used exist, namely SOAP (Simple Object Access Protocol), UDDI (Universal Description, Discovery, and Integration of business for the web), WSDL (Web Services Description Language), or other well known, like CORBA (Common Object Request Broker Architecture), RMI (Remote Method Invocation) or DCOM (Distributed Component Object Model).

The eXtensible Markup Language provides a vocabulary for describing remote procedure calls, which are transmitted between computers using the Hyper Text Transfer Protocol (HTTP). XML-RPC clients make procedure requests to XML-RPC servers, which return results to the XML-RPC clients. XML-RPC clients use the same HTTP facilities as web browser clients, and XML-RPC servers use the same HTTP facilities as web servers. XML-RPC requires a minimal number of HTTP headers to be sent along with the XML method request for solving a given problem instance.

For a better illustration of system's functionalities a problem instance belonging to the previously referred F2|n|Cmax problem class will be shown next.

Once the problem is correctly specified the system provides solving methods information as well as information related to the available methods' implementation(s). This information includes the link(s) for implemented methods' execution and other general information, for example, about each method class and author, as previously shown in section 3.3. The system also provides more detailed information about the method and its implementation(s), so that an easier selection of adequate scheduling methods can be achieved for solving the problem.

After having selected an implemented method available for solving the current problem, we only need to feed the system with the problem instance data and run it. Possible implemented methods are the Branch-and-Bound (B&B) method from Ignall and Schrage (Ignall and Schrage, 1965; Conway, 1967), and the Johnson method or rule, as previously shown in Figure 5. The first one belongs to the class of exact mathematical programming methods using the B&B technique and is an exponential time complexity method and the Johnson rule is a simple sequencing rule, with polynomial time complexity.

Therefore, let us consider the Johnson's rule for solving a problem instance with 4 jobs, which have to be processed in a flow shop with 2 machines. The objective, already known, consists on minimizing the maximum completion time (Cmax). Table 1 shows the time required for processing each job j on each machine i (p_{ji}).

Table 1: Scheduling problem data.

i / j	job1	job2	job3	job4	job5
m1	2	3	2	6	2
m2	4	1	5	3	4

Listing 1 shows an example that joins the headers and XML payload to form a complete XML-RPC request for solving this problem instance.

```
POST /rpchandler HTTP/1.0
User-Agent: AcmeXMLRPC/1.0
Host:localhost:6001
Content-Type: text/xml
Content-Length: 873
```

```
<?xml version="1.0"?>
<methodCall>
<methodName>ExactJohnson
</methodName>
<params>
<param>
<value><int>5</int></value>
</param>
<param><value><int>2</int></value>
</param>
 <param><value><array><data>
      <value><string>job1</string></value>
      <value><string>m1</string></value>
      <value><double>2</double></value>…
</data></array></value></param>
</params>
</methodCall>
```

Listing 1: An XML-RPC request.

Upon receiving an XML-RPC request, an XML-RPC server must deliver a response to the client. The response may take one of two forms: the result of processing the method or a fault report, indicating that something has gone wrong in handling the request from the client. As with an XML-RPC request, the response consists of HTTP headers and an XML payload.

Listing 2 shows a complete response from an XML-RPC server, for the considered problem instance, including both the HTTP headers and the XML payload.

```
HTTP/1.0 927 OK
Date: Tue, 8 Jun 2004 08:49:02 GMT
Server: MyCustomXMLRPCserver
Connection: close
Content-Type: text/xml
Content-Length: 893

<?xml version="1.0" encoding=" UTF-8"?>
<methodResponse>
<params>
<param><value>
<string>job5,job3,job1,job4,job2</string>
</value></param>
<param><value><double>19</double>
</value></param>
   <param><value><array><data>
      <value><string>job5</string></value>
      <value><string>m1</string></value>
      <value><double>0</double></value>
      <value><double>2</double></value>…
    </data></array></value>
</param>
</params>
</methodResponse>
```

Listing 2: An XML-RPC response.

Applying this Johnson method or rule (ExactJohnson) an optimal solution can be reached for the problem F2|5|Cmax. Figure 7 shows a Gantt chart for this problem, which has a minimal mekespan of 19 time units.

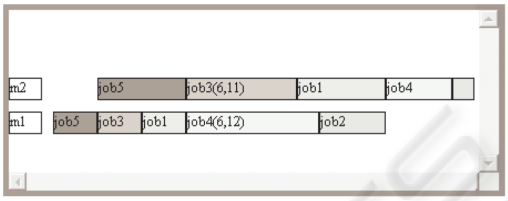

Figure 7: Gantt chart.

Gantt charts, like this one, are also automatically generated by the system, given the outputs provided by methods' executions. This is easily achieved because the methods' output data is expressed in XML documents, which enable an easy way of outputs conversion into different desired problem results presentation forms, including Gantt charts.
Listing 3 presents a DTD specification for modeling problem results as Gantt charts.

```
<!ELEMENT gantt (problem*)>
<!ELEMENT problem (job+)>
<!ATTLIST problem
   id CDATA #REQUIRED
   class CDATA #REQUIRED>
<!ELEMENT job (machine+)>
<!ATTLIST job
   id CDATA #REQUIRED
   name CDATA #REQUIRED>
<!ELEMENT machine (start+,finish+)>
<!ATTLIST machine
   id CDATA #REQUIRED
   name CDATA #REQUIRED>
<!ELEMENT start (#PCDATA)>
<!ELEMENT finish (#PCDATA)>
```

Listing 3: DTD specification for Gantt chart.

Listing 4 exemplifies the corresponding XML code for the Gantt chart, which represents the given problem results.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE gantt SYSTEM "gantt.dtd">
<gantt>
   <problem id="0010" class="F2|n|Cmax">
         <job id="j001" name="job1"/>
          <machine id="m001"
             name="m1">
                <start>0</start>
                <finish>3</finish>
```

```
        </machine>
       </job>…
   </problem>
</gantt>
```

Listing 4: XML code for Gantt chart.

These time-based charts continue to be largely used due to its expressive power, enabling a very easy way of comparing results obtained from the execution of several different implemented methods available. Other alternatives for displaying those outputs are available, including direct outputs presentation through XML documents and tables.

4 RELATED WORK

Recently we notice a tendency for an increase in scheduling systems accessible through the Internet. The web systems that we came across usually involve solvers or a community of solvers, addressing each one the resolution of a restricted range of scheduling problems, using specific techniques or tools, such as mathematical programming. Moreover, they are not usually designed to easily incorporate new methods implementations by users.

An example of a web system that can be used for scheduling is the NEOS Server, developed under the auspices of the Optimization Technology Center of Northwestern University and Argonne National Laboratory, for optimization problems solving. It makes nearly 50 solvers available through a broad variety of network interfaces. According to the authors, although having evolved along with the web and the Internet, it is limited to some degree by early design decisions (http://www-neos.mcs.anl.gov/).

We can also refer the BBN's Vishnu scheduling system, a web-based optimisation scheduling system (http://vishnu.bbn.com) and the FortMP, a Mathematical Programming Solver from Mitra's Group at Brunel University (http://www.brunel.ac.uk/depts/ma/research/com).

The e-OCEA, a portal for scheduling intends to help identifying scheduling problems, to help development of new algorithms and to conduct benchmarks through the Internet. However, one requirement of this system is to only consider elements (algorithms, data sets, schedules and modules) that are e-OCEA compatible (http://www.ocea.li.univ-tours.fr/eocea).

The LekiNET, a prototype Internet scheduling environment, by Benjamin P. C. et al. (Yen et al., 2004), which has migrated from LEKIN, a flexible job shop system, is a system that has some similarities to our system, focusing more on cost effective choice of scheduling agents for solving problems. The authors propose a migration scheme to transform existing standalone scheduling systems to Internet scheduling agents that can communicate with each other and solve problems beyond individual capabilities. They treat each system as an agent and build the relations between the systems. Therefore, wrappers need to be specifically designed for each system.

In our case, any method which is accessible through the Internet, provided its signature and location are specified within our web system, can be used, for solving problems put by users. No further requirements are necessary for being able to remotely use available methods. The association of scheduling problems to resolution methods is done using the information available in the DKB, about both problems and solving methods. We did not come across to systems with identical architecture and underlying approach, i.e. oriented for solving a large variety of manufacturing scheduling problems based on a continuously updatable distributed knowledge base, which allows a network of peers to provide the scheduling service to users and the dynamic enlargement of the number of methods that can be accessed.

5 CONCLUSION

In manufacturing enterprises, it is important nowadays, as a competitive strategy, to explore and use software applications, now becoming available through the Internet and Intranets, for solving scheduling problems, which can be achieved in an easily way by using web service technology.

This work is based on an XML-based specification framework for production scheduling concepts modeling, which is used as specification framework for production scheduling decision-making through a web service. Some of the important functions include the ability to represent scheduling problems and the identification of appropriate available methods for solving them.

The XML-based data modeling is used in order to make possible flexible communication among different scheduling applications. This modeling and specification contributes to the improvement of the scheduling process, by allowing an easy selection of several alternative methods available for problem solving, as well as an easy maintenance of the distributed knowledge base supporting the scheduling service. This is achieved by providing a user-friendly way of new knowledge insertion. This knowledge primarily includes scheduling problems and solving methods as well as its implementations, and corresponding inputs and outputs form. The implemented methods are available and accessible through the Internet. These may be implemented on different programming languages and running on different platforms. Such implementations are easily accessible through the web service here presented for solving scheduling problems, through the execution of local or remotely available scheduling methods. The system allows comparing different solutions obtained by the execution of different methods for a same scheduling problem, and to choose the best solution found to solve the problem, according to a given performance measure defined.

The XML based specification can be generated and visualized by computers in appropriate and different ways. An important issue is that the data representation model is general, accommodating a large variety of scheduling problems, which may occur in different types of manufacturing environments.

REFERENCES

Artiba, A., 1997. Elmaghraby, S., *The Planning and Scheduling of Production Systems*, Chapman & Hall. UK.

Abiteboul, S., et al., 2000. *Data on the Web - From Relations to Semistructured Data and XML*, Morgan Kaufmann Publishers. USA.

Arts, E., Lenstra, J. K., 1997. *Local Search in Combinatorial Optimization*, John Wiely & Sons.

Blazewicz, J., et al., 1996. *Scheduling Computer and Manufacturing Processes*, Springer-Verlag. Germany.

Brucker, P., 1995. *Scheduling Algorithms*, Springer-Verlag. Germany.

Ceponkus, A., Hoodbhoy, F., 1999. *Applied XML*, Wiley Computer Publishing. USA.

Conway, R. W., Maxwell, W. L., Miller, L. W., 1967. *Theory of Scheduling*, Addison-Wesley Publishing Company, Inc. England.

French, S., 1982. *Sequencing and Scheduling – An Introduction to Mathematics of the Job-Shop*, John Wiley and Sons, Inc.

Graham, R. L., Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., 1979. Optimization and Approximation in Deterministic Sequencing and Scheduling: A survey. In *Annals of Discrete mathematics*.

Harper, F., 2001. *XML Standards and Tools*, eXcelon Corporation, USA.

Ignall, E., Schrage L., 1965. Application of the Branch-and-Bound Technique to Some Flow-Shop Problems. *In Operations Research*, 13:3.

Jordan, C., 1996. *Batching and Scheduling*, Springer-Verlag. Germany.

Laurent, S., et al., 2001. *Programming Web Services with XML-RPC*. O'Reilly & Associates, Inc.

Morton, T., Pentico, D., 1993. *Heuristic Scheduling Systems*, John Wiley & Sons Inc. USA.

Osman, I. H., Kelly, J. P., 1996. *Meta-Heuristics: Theory and Aplications*, Kluwer Academic Publishers.

Papazoglou, M.P.; Krämer, B.J.; Yang, J.; 2003. Leveraging Web-Services and Peer-to-Peer Networks, In: *Proceedings of Advanced Information Systems Engineering, 15th International Conference, CaiSE,* Klagenfurt, Austria. June 16-18, pp. 485-501.

Pinedo, M., 1995. *Scheduling Theory, Algorithms and Systems*, Prentice-Hall Inc. USA.

Varela, L., Aparício, J., Silva, S, 2002a. An XML Knowledge Base System for Scheduling Problems. *In IICS'02 - Proceedings of the Innovative Internet Computing System, Kuhlungsborn. Germany*. Springer-Verlag in the Lecture Notes in Computer Science series.

Varela, M. L. R.; Aparício, J. N.; Silva, S. C., 2002b. Developing a Web Scheduling System Based on XML Modeling, In: *Knowledge and Technology Integration in Product and Services – Balancing Knowledge and Technology in Product and Service Life Cycle, BASYS'02*, Cancun, Mexico. Kluwer Academic Publishers, pp. 61-70.

Varela, L., Aparício, J., Silva, s., 2002c. Scheduling Problems Modeling with XML. *In The Proceedings of the 4th International Meeting for Research in Logistics*, Lisbon. Portugal.

Varela, M. L. R.; Aparício, J. N.; Silva, S. C., 2003a. A Scheduling Web Service based on XML-RPC, In: *Proceedings of the ICAPS-03 Workshop on Planning for Web Services*, Trento, Italy, pg. 43-50.

Varela, M. L. R.; Aparício, J. N.; Silva, S. C., 2003b A Scheduling Web Service, In: *Proceedings of the 1st Multidisciplinary International Conference on Scheduling: Theory and Applications, MISTA'03*, Nottingham, UK. ASAP, The University of Nottingham, pg. 540-551.