A Scheduling Web Service based on XML-RPC

Maria Leonilde R. Varela*, Joaquim N. Aparício†, and Sílvio do Carmo Silva+

*Dept. of Production and Systems, University of Minho, Campus Azurém, 4800-058 Guimarães, Portugal, Phone: +351 253 510341, Fax: +351 253 510343, Email: leonilde@dps.uminho.pt

†Dept. of Computer Science, New University of Lisbon, Quinta da Torre, 2829-516 Monte de Caparica, Portugal, Phone: +351 21 2948536, Fax: +351 21 2948541, Email: jna@di.fct.unl.pt

+Dept. of Production and Systems, University of Minho, Campus Gualtar, 4710-057 Braga, Portugal, Phone: +351 253 604745, Fax: +351 253 604741, Email: scarmo@dps.uminho.pt

Abstract

In this paper we make an XML-based modeling and communication contribution to production scheduling activity. Scheduling problems and resolution methods are modeled in XML[1]. This information modeling is the basis of a web service oriented to scheduling problem solving, which is under development. The resolution of scheduling problems is carried out with the aid of a web system that uses the XML-RPC protocol for the execution of methods, local or remotely available through the Internet.

Keywords: Production scheduling and the new information technology: XML Modeling and Web Service with XML-RPC.

Introduction

The scheduling activity in a production environment seeks to optimize the use of available production means or resources, ensuring short time to complete jobs and to satisfy other organizational optimizing criteria.
Production Scheduling may be defined as the activity, of allocating tasks or jobs to production resources, or vice versa, over time, for achieving good operating performance. The resulting schedule or scheduling plan can be more or less detailed, in accordance with the intended objectives and the planning time horizon. Thus, there are cases where we are only interested in obtaining the sequence in which the jobs should be processed in certain machines of a production system, and other cases where we are interested in knowing the planned start and finishing times of each job operation on each machine.

The effective and efficient resolution of scheduling problems begins with the identification of suitable scheduling methods to solve them. Sometimes we may encounter methods, which find optimum solutions. Frequently, however, for real world problems, this is not the case, due to the complexity of the problems. So we might have to draw upon available methods, which are likely to find good solutions but not necessarily optimum ones.

[1] eXtended Markup Language.

When there are alternative methods to solve a problem we can obtain alternative solutions, which should be evaluated against specified criteria or objectives to be reached. Thus, we are able to solve a problem, through the execution of one or more scheduling methods and, subsequently, select the best solution provided by them. These methods can either be local or remotely accessible through the Internet.

In our work we seek to improve the resolution process for scheduling problems by means of a web system. This system requires the specification of each problem to be solved and the access to resolution methods, which are available for solving them. For problems specification we propose a problem classification framework. Based on this, the XML language is used as a specification language for scheduling data modeling and processing on the Internet. This kind of data modeling allows, for instance, identifying scheduling problems, and methods for its resolution, as well as the communication necessary for the execution of 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 proposed. References to some well known scheduling problem classes and corresponding resolution methods are shown. The section also presents a brief summary of the main search techniques used by scheduling methods. Next, the XML-based specification for scheduling concepts modeling is presented. It is illustrated with some examples of Document Type Definitions (DTDs) and corresponding XML documents. Subsequently, the XML-RPC, an XML-based Remote Procedure Call protocol, is briefly presented and its use is exemplified through an example of a scheduling method execution for a certain problem instance. Finally, some concluding remarks are presented and planned future work is briefly referred.

Scheduling Problems and Resolution Methods

Scheduling Problems

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 an NP-hard problem 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 heuristic methods. These include both, those, which we know how near their solutions may be from optimum ones, known as approximation methods, and also a variety of heuristic methods, which are likely to achieve good solutions (French 1982).

In order to execute the scheduling process it is necessary to clearly specify the problem to be solved. Scheduling problems have a set of characteristics that must be clearly and unequivocally defined.
Due to the existence of a great variety of scheduling problems, there is a need for a formal and systematic manner of problem classification and representation. A framework for achieving this was developed by Varela (1999), summarized in Table 1, 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 put forward with this work.

The referred framework for problem representation includes three classes of notation parameters for each corresponding class of problem characteristics, Table 1.

Class	Factor	Description	Value
α	$\alpha 1$	Manufacturing system type	O, *P*, *Q*, *R*, *X*, *O*, *J*, *F*, PMPM, ...
	$\alpha 2$	Number of machines	O, *k*
β	$\beta 1$	Job/operation preemption	O, pmtn
	$\beta 2$	Precedence constraints	prec, chain, tree,sp-graph, …
	$\beta 3$	Ready times	O, r_j
	$\beta 4$	Restrictions on processing times	p_j=1, p_{ji}=1, p_j=p, $p_{inf} \leq p_j \leq p_{sup}$, …
	$\beta 5$	Due dates (deadlines)	O, d_j
	$\beta 6$	Batches/families processing	O, batch
	$\beta 7$	Number of jobs or of tasks in a job (job shop case)	O, n_j
	$\beta 8$	Job/task priorities	O, w_j
	$\beta 9$	Dynamic machine availability	O, avail
	$\beta 10$	Additional/auxiliary resources	O, aux
	$\beta 11$	Buffers	O, no-wait
	$\beta 12$	Setup (changeover)	O, setup*
γ	γ	Performance measure	C_{max}, F_{max}, $\sum C_j$, $\sum w_j C_j$, L_{max}, $\sum T_j$, …

Table 1 – Scheduling problems 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 and the number of machines that exist in the system. Another class allows specifying the interrelated characteristics and constraints of jobs and production resources, which is done by the class β (β_1 … β_{12}) of parameters. 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. For the optimization criterion, the third and last class of the framework, we use the parameter γ. This allows specifying the schedule evaluation or performance measure, which can be either a single criterion measure or a multi-criteria one.

An example of the use of this notation is "F,3|n|Fmax" which reads as: "Scheduling of non-preemptable and independent tasks of arbitrary processing time lengths, arriving to the system at time 0, on a pure flow shop, with 3 machines, to minimize the maximum flow time.

Good schedules strongly contribute to business success. This may mean meeting due dates, achieving short delivery times for accepted orders, low flow times, few ongoing jobs in the system, low inventory levels, high resource utilization and, certainly, low production costs. All these objectives can be better satisfied through the execution of the most suitable scheduling methods available for the resolution of each particular problem.

Resolution Methods

It is rather obvious that the time we often can devote for solving particular scheduling problems is usually short. Therefore, only low order polynomial time approaches are likely to be acceptable to solve real world problems, usually complex. Thus, the examination of the complexity of these problems should be the basis for further analysis to problem solving. Fortunately, not all NP-hard problems are equally hard from a practical perspective. Some NP-hard problems can be solved pseudo-polynomially using approximation methods that provide feasible solutions, which, although normally sub-optimum, are of good quality. Examples of this kind of methods are based on dynamic programming or branch and bound techniques. Other approaches to obtain good or at least satisfactory solutions, in acceptable time, are based on the nowadays widely used meta-heuristics, based on local or neighborhood search techniques, such as Genetic Algorithms (GA), Simulated Annealing (SA) and Tabu Search (TS). These are also known as extended neighborhood search techniques. We can still mention promising scheduling approaches based on bottleneck resources, neural networks, Petri-nets and computer simulation. Heuristic methods tend to provide good results in the available time to make decisions, reason why it is important to incorporate them in the web scheduling system here presented, as we are doing.

Table 2 shows a small sample of makespan optimization flow shop scheduling problems, for which methods are referenced, collected from Brucker (1995), and which may appear in real world production systems.

Information like this is used and available for retrieval through the web scheduling system. The system is able to quickly suggest methods for solving problems that occur in real world manufacturing environments and solve them through the execution of an appropriate method implementation, local or remotely available through the Internet. This draws upon the web system knowledge base for scheduling problems and methods.

Problem class	Method reference	Observations
F2 | | Cmax	Johnson (1954)	Maximal polynomially solvable Without preemption
F2 | rj | Cmax	Lenstra et al (1977)	Minimal NP-hard Without preemption
F2 | rj; no-wait | Cmax	Roeck (1984)	Maximal polynomially solvable With no wait
F3 | pmtn | Cmax	Gonzalez & Sahni (1978) Cho & Sahni (1981)	Maximal polynomially solvable With preemption
F3 | | Cmax	Garey et al (1976)	Minimal NP-hard Without preemption
F | pji=1; prec | Cmax	Leung et al (1984), Timkovsky (1998)	Minimal NP-hard Without preemption
FMPT | n=3 | Cmax	Kraemer (1995)	Minimal NP-hard With multiprocessor task
FMPT, m | rj; pji=1|Cmax	Brucker & Kraemer (1996)	Maximal polynomially solvable With multiprocessor task
FMPM, m | rj;pji=1|Cmax	Brucker et al (1997)	Maximal polynomially solvable With multipurpose machines
FMPM | prec; pji=1| Cmax	Ullman (1975)	Minimal NP-hard With multipurpose machines

Table 2 – Scheduling methods assigned to problems.

Scheduling Concepts Modeling using XML

Since 1995 great happenings have changed the world of information technology, especially the emergence of new Internet technologies. The eXtensible Markup Language (XML) is one of those new technologies that has been having a wide acceptance and is causing a great impact on Internet real world applications, since its release by the World Wide Web Consortium (W3C) in 1998 (Pardi 1999). XML enables to describe structures and meanings of data, with a simple syntax, and is an ideal candidate format for exchanging and processing data through the Internet. Other advantages of XML based representation are its openness, simplicity and scalability (Abiteboul et al 2000). These were important reasons for choosing XML to develop our web application. For details about XML and related technologies (DTD, XSL, XML Schemas, Namespaces, etc.) see, for example, Ceponkus and Hoodbhoy (1999) or Harper (2001).

The web applications can use XML for data storage and processing, for showing multiple views of the data and for representing complex data structures. Therefore, XML may guarantee the future utilization of data formats and the exchange of data structures, so that the web documents and the platforms become more robust for systems integration (Pardi 1999).

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).

There are many other web-based technologies available for data storage and transferring, but we think that it is more adequate and easier to develop a new system using these new techniques rather than using conventional techniques such as EDI (Electronic Data Interchange). XML based data exchange is becoming very popular in global manufacturing, and this will cause connectivity becoming more and more convenient and necessary.

Problems Specification

Following the lines already presented in (Varela 1999, Varela 2002a, and Varela 2002b), problems are classified and modeled by a DTD - Document Type Definition (c.f. Listing 1). Elements introduced in the referred DTD are expected to become part of a common namespace. Elements on the problem DTD file precisely characterize a scheduling problem, meaning that in order to interact with the system a problem must be described according to that grammar.

```
<!-- Elements and attributes declaration -->
<!ELEMENT problems (problem+)>
<!ELEMENT problem (alpha?, beta?, gamma?)>
<!ATTLIST problem
   problem_class CDATA #REQUIRED
   preferred (true | false) "false">
<!-- Alpha elements -->
<!ELEMENT alpha (alpha1?,alpha2?)>
<!ELEMENT alpha1 EMPTY>
<!ATTLIST alpha1 system_type (0 | P | Q | R | F |…) "0">
<!ELEMENT alpha2 EMPTY>
<!ATTLIST alpha2 value (0 | m) "0"
   machines_quantity CDATA #REQUIRED>
<!-- Beta elements -->
<!ELEMENT beta (beta1?, beta2?, ... , beta12?)>…
<!ELEMENT beta4 (job_times*)>
<!ATTLIST beta4 value (0 | pj) "0">
<!ELEMENT job_times EMPTY>
<!ATTLIST job_times
   name CDATA #REQUIRED machine CDATA
    #REQUIRED
   time CDATA #REQUIRED >…
<!ELEMENT beta7 EMPTY>
<!ATTLIST beta7 value (0 | nj) "0"
   job_quantity CDATA #REQUIRED>…
<!-- Gamma element -->
<!ELEMENT gamma EMPTY>
<!ATTLIST gamma measure (Cmax | SumCj | Cj_mean |
SumWjCj | Lmax | SumLj | Lj_mean | SumWjLj | Tmax |
SumTj | Tj_mean | SumWjTj | Emax | SumEj | Fmax … )
"Cmax">
```

Listing 1 – DTD for problem example (problems.dtd).

From Listing 1, one can read that, in order to define a problem we must optionally define the alpha, beta and gamma factors, because we always have a default value in the problem classification model assigned to each factor in the nomenclature.
The problem we are going to exemplify is known to belong to class F,3|n|Fmax and can be defined by an XML file as follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE problems SYSTEM "problems.dtd">
<problems>
   <problem problem_class="F,3|n|Fmax">
      <alpha>
         <alpha1 system_type ="F"/>
         <alpha2 machines_quantity="3"/>
      </alpha>
      <beta>…
          <beta4 value="0"><job_times name="J1"
             machine=”M1”  time="3"/>…</beta4>…
         <beta7 value="0" job_quantity="4"/>…
      </beta>
      <gamma measure="Fmax"/>
   </problem>
</problems>
```

Listing 2 – XML for problem example (problems.xml).

Methods Specification

In order to match problem instances to resolution methods we must be able to identify those problems and retrieve appropriate methods for its resolution. The scheduling methods and its implementations are described by a given DTD (c.f. methods.dtd and implementations.dtd below). Many scheduling methods may be more or less adequate to solve a given class of problems. In the methods knowledge base the system records the scheduling method(s) that can be used for solving a certain problem class.

```
<!ELEMENT methods (method*)>
<!-- Element method -->
<!ELEMENT method (information, implementation,
description?, input?, output?)>
<!ELEMENT information EMPTY>
<!ATTLIST information
   id CDATA #REQUIRED
   name CDATA #REQUIRED
   problem_class CDATA #REQUIRED
   method_class CDATA #REQUIRED>
<!ELEMENT implementation EMPTY>
<!ATTLIST implementation
   href CDATA #REQUIRED
   preferred (true | false) "false"
   description  CDATA #REQUIRED>
<!ELEMENT description (#PCDATA)>
<!ELEMENT input (#PCDATA)>
<!ELEMENT output (#PCDATA)>
```

Listing 3 – DTD for a B&B method (methods.dtd).

```
<?xml version="1.0"?>
<!DOCTYPE methods SYSTEM "methods.dtd">
<methods>
   <method>
      <information id="001" name="ExactBranchBound"
problem_class="F,m|n|Fmax" method_class="Exact B&B"/>
      <implementation href="http://localhost:5001/"
description="This implementation ..."/>
      <description>This method can be used …</description>
      <input>The method input is …</input>
      <output> The method output is …</output>
   </method>
</methods>
```

Listing 4 – XML for a B&B method (methods.xml).

Methods Implementations Specification

In the Internet many implementations may exist for a given method. 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. Unfortunately not all implementations work in the same way and in order for the system to use those implementations in a programmatic way, implementations 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, in turn, includes the definition of the parameters that are necessary for its execution and its output format. All this information is described in the implementation.dtd file (Listing 5) and an instance of an XML document for our example can be seen in Listing 6.

```
<!ELEMENT implementations (implementation*)>
<!ELEMENT implementation (information, signature)>
<!ELEMENT information EMPTY>
<!ATTLIST information
   id CDATA #REQUIRED
   name CDATA #REQUIRED
   href CDATA #REQUIRED
   preferred (true | false) "false"
   description CDATA #REQUIRED>
<!ELEMENT signature (input, output)>
<!-- Element input -->
<!ELEMENT input (input_information, jobs_information)>
<!ELEMENT input_information EMPTY>
<!ATTLIST input_information
   input_information CDATA #REQUIRED
   results_file CDATA #REQUIRED
   jobs_quantity CDATA #REQUIRED
   machines_quantity CDATA #REQUIRED>
<!ELEMENT jobs_information (job_input*)>
<!ELEMENT job_input EMPTY>
<!ATTLIST job_input
   name CDATA #REQUIRED
    machine CDATA #REQUIRED
   time CDATA #REQUIRED>
<!-- Element output -->
<!ELEMENT output (output_information, sequence?,
measures,
jobs_results?)>
<!ELEMENT output_information EMPTY>
<!ATTLIST output_information
   output_information CDATA #REQUIRED>
<!ELEMENT sequence (#PCDATA)>
<!ELEMENT measures (measure*)>
<!ATTLIST measures
   information CDATA #REQUIRED>
<!ELEMENT measure EMPTY>
<!ATTLIST measure
   name CDATA #REQUIRED
   value CDATA #REQUIRED>
```

```
<!ELEMENT jobs_results (job_output*)>
<!ELEMENT job_output EMPTY>
<!ATTLIST job_output
    name CDATA #REQUIRED
    machine CDATA #REQUIRED
   start CDATA #REQUIRED
   conclusion CDATA #REQUIRED>
```

Listing 5 – DTD for the implementation of a B&B method (implementations.dtd).

```
<?xml version="1.0"?>
<!DOCTYPE implementations SYSTEM
"implementations.dtd">
   <implementations>
      <implementation>
         <information id="1001"
            name="getExactBranchBound"
href="http://localhost:5001" description="The Ignall-Schrage
B&B method, for problem class F,m|n|Fmax belongs to …" />
       <signature>
        <input>
          <input_information input_information="The input
required by…" results_file="results.xml" jobs_quantity="4"
machines_quantity =”3”/>
             <jobs_information>
              <job_input name="J1" machine="M1"
                 time="3"/>…
             </jobs_information>
         </input>
            <output>
         <output_information output_information="The
            output..."/>
         <sequence>J1,J3,J4,J2</sequence>
         <measures information="The measures of ...">
            <measure name="Fmax" value="39"/>
         </measures>
         <jobs_results>
           <job_output name="J1" machine="M1" start="3"
              conclusion="3"/>
         </jobs_results>
        </output>
       </signature>
     </implementation>
   </implementations>
```

Listing 6 – XML code for the implementation of a B&B method (implementations.xml).

The system here described has been designed and implemented as a web service (http://www.w3.org) using the XML-RPC (eXtended Markup Language – Remote Procedure Call) protocol (Laurent et al. 2001).
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 more expressive output, like a Gantt chart or even other data representation.

Methods Invocation through XML-RPC

As mentioned above the main purpose of this work is to provide a framework to improve the resolution of scheduling problems based on XML modeling and related technologies. Figure 1 illustrates a general outline of the system architecture.

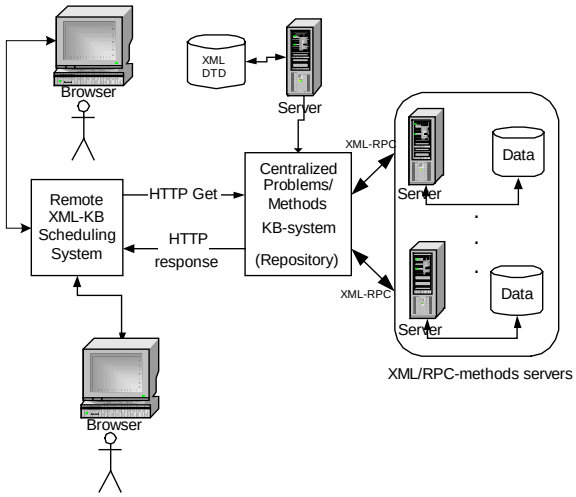

Figure 1 – Web-system architecture.

The main element of the web system structure is an interface located in the Centralized Problems/ Methods Knowledge Base (CPM-KB) unit, for introduction, validation, and transformation of manufacturing scheduling data. This interface is mainly controlled by DTD and XSL (eXtended Stylesheet Language) documents stored in a database. The scheduling information is also stored in XML documents and these documents are verified using DTDs, before being placed in the XML database. The XML and related documents may either be located on the server or on the client side. In this work the documents are stored on the server (e.g. XML, DTD, XSL and other documents) in order to achieve easy and efficient data transferring.

The system is being implemented as a web service and allows the execution of either local or remote scheduling methods through the XML-RPC protocol.
The term Web Services has emerged as a general category for loosely coupled, dynamically connected web-based services and are a set of tools that let us build distributed applications on top of existing web infrastructures.
These 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.

These Web Services are most commonly used to invoke remote application services or methods using a Remote Procedure Call (RPC) interaction implemented using only XML messages (Carlson 2001).
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). Nevertheless, XML-RPC is among the simplest and most foolproof web service approaches, and makes it easy for computers to call procedures on other computers (Laurent et al. 2001). The XML provides a vocabulary for describing remote procedure calls, which are then transmitted between computers using the Hyper Text Transfer Protocol (HTTP).

XML-RPC clients make procedure requests of 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. Listing 7 shows an example that joins the headers and XML payload to form a complete XML-RPC request for our example.

```
POST /rpchandler HTTP/1.0
User-Agent: AcmeXMLRPC/1.0
Host:localhost:5001
Content-Type: text/xml
Content-Length: 832
<?xml version="1.0"?>
<methodCall>
<methodName>getExactBranchBound</methodName>
<params>
  <param><value><int>4</int></value></param>
  <param><value><int>3</int></value></param>
  <param><value>
    <array><data>
       <value><string>J1</string></value>
       <value><string>M1</string></value>
       <value><double>3</double></value>…
     </data></array>
</value></param></params></methodCall>
```

Listing 7 – A complete 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 8 shows a complete response from an XML-RPC server, including both the HTTP headers and the XML payload.

```
HTTP/1.0 927 OK
Date: Fri, 25 Oct 2002 07:38:05 GMT
Server: MyCustomXMLRPCserver
Connection: close
Content-Type: text/xml
Content-Length: 868
<?xml version="1.0"?>
<methodResponse>
<params>
  <param><value><string>J1,J3,J4,J2</string></value>
  </param>
  <param><value><double>39</double></value></param>
  <param><value>
    <array><data>
       <value><string>J1</string></value>
       <value><string>M1</string></value>
       <value><double>0</double></value>
       <value><double>3</double></value>…
     </data></array>
</value></param></params></methodResponse>
```

Listing 8 – A complete XML-RPC response.

The response is provided to a call to the method getExactBranchBound, which returns a solution to the F,3|n|Fmax problem.

Figure 2 schematizes our web service framework, based on this protocol.

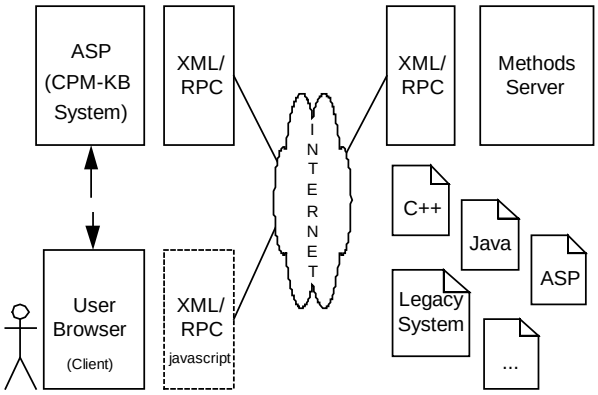

Figure 2 – Web-service based on XML-RPC.

By using the XML-RPC protocol we are able to execute scheduling methods implemented in different programming languages. Moreover, these methods, local or remotely available, may be running on different platforms.

As shown previously in Figure 1, our web scheduling system includes a Centralized Problem/ Methods Knowledge Base (CPM-KB) unit that encompasses all the knowledge and components necessary for remote methods invocation and for performing all other system functionalities. This unit is controlled by ASP (Active Server Pages) and corresponding server-side XML-RPC components.
On the other hand, there are also correspondent XML-RPC components for each of the methods servers waiting for RPC requests.

This environment is heterogeneous as servers can use their own technology, i.e. use different implementation languages or/ and different operating systems.
More detailed information about the XML-RPC protocol can be obtained from http://www.xmlrpc.com.

System Functionalities

This web system encompasses several 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 scheduling problem definition. The selection of one or more specific methods is made by the system through a searching process on the knowledge base about scheduling methods (CPM-KB). The system also enables problem results presentation and storage.

The fundamental system functionalities are those related to information modeling, which can be summarized as follows:
- The easy classification and identification of scheduling problems, by using the notation for the classification of those problems, presented in a previous section.
- The easy classification and identification of scheduling methods.
- The automatic association of scheduling methods to problems for its resolution and, finally,
- The possibility of solving scheduling problems, through the selection of one or several methods implemented, allowing results comparison and the selection of the most suitable one, for each particular case.

For a better illustration of system functionalities we will explain the resolution of an instance of the previously described F,3|n|Fmax problem class.

In order to find out which method's implementations are available for solving this problem, first of all, a set of problem factors has to be specified, according to the $\alpha|\beta|\gamma$ characterization model previously summarized in Table 1.

After having defined our problem the system returns the problem class to which it belongs. In case of having committed some kind of mistake we can always restart the previous problem characterization process, eventually using the system on-line help information, until we reach the correct one.

Next, when the problem is correctly classified the system provides resolution methods information as well as information related to the available methods' implementations. This information includes the links for executing methods' implementations and for other information, such as method's class and method's author. The system also provides detailed information about the method and its implementations, so that an easy selection of adequate scheduling methods can be achieved for solving the problem. Presently the system only has one method implementation available for solving F,3|n|Fmax problems that is a C++ implementation of the Ignall and Schrage method (Ignall and Schrage 1965, Conway 1967). This is an exact mathematical programming method, with exponential time complexity, based on the B&B technique.

Once knowing which method implementation to use for solving the problem, we only need to feed the system with problem data and run the method.

Lets consider a problem with 4 jobs, which have to be processed in a flow shop with 3 machines. The objective, already known, consists of minimizing the maximum flowtime (Fmax). Table 3 shows the time required for processing each job j on each machine i (pji).

i \ j	J1	J2	J3	J4
M1	3	11	7	10
M2	4	1	9	12
M3	10	5	13	2

Table 3 – Scheduling problem data.

Applying this B&B method the optimal solution can be reached for the problem. Figure 3 shows the Gantt chart obtained by the system for this problem, which has a minimal flow time of 39 time units.

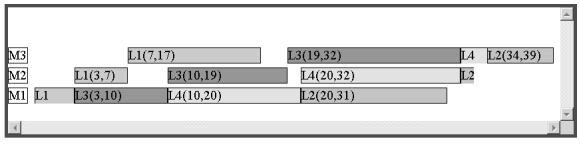

Figure 3 – Problem results.

Gantt charts are also automatically generated by the system. This is easily achieved because the problem output data is expressed in XML documents that enable an easy way of outputs conversion to different desired output forms, namely into Gantt charts. This facilitates comparing solutions obtained from the execution of several method implementations. Other alternatives for displaying the same data are available.

Conclusions

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.

This communication proposes an XML-based specification framework for production scheduling concepts modeling, together with a web-based production scheduling system. The XML-based data modeling is used in order to make possible flexible communication among different scheduling applications. Some of the important system's functions include the ability to represent scheduling problems and the identification of appropriate methods to solve them.

The XML-based scheduling data specification contributes to the improvement of the scheduling processes by allowing an easy selection of several alternative methods available for problem solving, as well as an easy maintenance of the knowledge base itself. This primarily includes both scheduling problems and solution methods, which are available through the Internet. The framework enables the registration of new problem classes, resolution methods and implementations, as well as post classification and matching among them.
The specification format is adequate for the exchange of scheduling data, since it enables to handle with loosely coupled systems and with complex hierarchical data.
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 production scheduling problems, which may occur in different types of manufacturing environments.

Furthermore, the web scheduling system under development facilitates the resolution of scheduling problems, through the execution of local or remote scheduling methods, available on different computers through the Internet, in order to greatly contribute to assist de scheduling decision-making process, by allowing different solutions comparison, obtained by the execution of different methods for a same problem and to choose the solution, which shows more suitable to solve each particular problem that occurs in the identified manufacturing environment under consideration.

Although the main goal is the service for problems resolution, the system can be used for teaching purposes, and from that point of view some additional functionalities are being implemented to be available, in an interactive (non programmatic) way, by a user browsing the system. Some of these functionalities include historical and referencing information about each problem class, methods and its implementation(s).

References

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

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

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

Brucker, P. Eds. 1995. Scheduling Algorithms, Germany: Springer-Verlag.

Carlson, D. eds. 2001. Modeling XML Applications with UML – Practical e-Business Applications, USA: Addison-Wesley.

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

Chrétienne, P., et al. eds. 1995. Scheduling Theory and its Applications, England: John Wiley & Sons Inc.

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

French, S. eds. 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. eds. 1979. Optimization and Approximation in Deterministic Sequencing and Scheduling: A survey, Annals of Discrete mathematics.

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

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

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

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

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

Pardi, W. eds. 1999. XML - Enabling Next-generation Web Applications, USA: Microsoft Press.

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

Varela, L., Aparício, J., Silva, S. 2002, An XML Knowledge Base System for Scheduling Problems. In Proceedings of the Innovative Internet Computing System Conference, 61-70. Kuhlungsborn, Germany: Springer-Verlag in the Lecture Notes in Computer Science series.

Varela, L., Aparício, J., Silva, S. 2002, Scheduling Problems Modeling with XML, In Proceedings of the 4th International Meeting for Research in Logistics, 897-909. Lisbon, Portugal: International Meeting for Research in Logistics, Inc.

Varela, M. L. 1999. Automatic Scheduling Algorithms Selection, Msc. Diss., Dept. of Production and Systems, University of Minho, Portugal.