Workflow with Dynamic Measurement Scenarios in the Virtual Laboratory

Marcin Lawenda, Norbert Meyer, Tomasz Rajtar, Marcin Okoń, Dominik Stokłosa, Damian Kaliszan, Mirosław Kupczyk, and Maciej Stroiński

Poznań Supercomputing and Networking Center,
ul. Z.Noskowskiego 10, 61-704 Poznań, Poland,
Corresponding author: Marcin.Lawenda@man.poznan.pl,
WWW home page: http://vlab.psnc.pl/

Abstract. In the paper some considerations about dynamic measurement scenarios aspects are discussed. The Dynamic Measurement Scenarios are used for creating and managing job workflow in the Virtual Laboratory system. It facilitate defining and monitoring measurement process which quite often consists of many tasks connected between themselves.

1 INTRODUCTION

Virtual Laboratory (VL) is a heterogeneous, distributed environment, which allows scientists all over the world to work on a common group of projects [1]. This environment should allow conducting experiments with the usage of physical devices, doing simulation using the computational application, and communication between users working on the same topic. Similarly to typical laboratory tools and research techniques depending on the specific field of science, virtual laboratories can benefit from some collaboration techniques as tele-immersion, but they are not mandatory.

In order to meet these goals, the virtual laboratory should specify some requirements. The most important requirements follow:

- the possibility of performing remote experiments (real and computational),
- load balancing - allows to run the experiment on the less loaded sites,
- reservation of time for experiments execution; especially for experiments that require supervision or can be observed by more people on-line,
- communication with the physical laboratory staff, which can be necessary to help run some specific experiments, e.g. setting up and operating devices to experiments,
- sharing experiments results - the result should be stored in a specialized storage system and available for other scientists (with the possibility to restrict access rights),
- library of electronic papers, reports and books grouped by a specific topic,
- the possibility of comparing the achieved results in real experiment and simulation, which can help to properly adjust the simulation parameters,

– communication with other people working on similar problems using chat, audio and video conferences etc.

To assure such complex functionality, the Virtual Laboratory system consists of numerous modules and uses many outside services. Most important of them are depicted in Fig. 1.

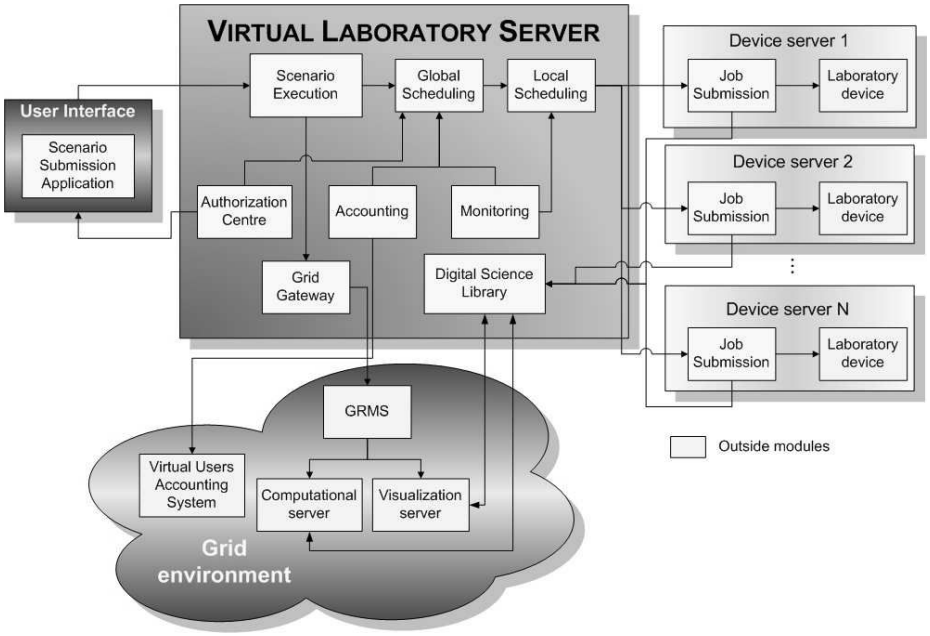

Fig. 1. Connection between major modules in the virtual laboratory system

Research work of the authors is focused on virtual laboratory aspects [4] that have significant influence on creating a general system which will allow taking control on many different devices. Another issue (which we want to discuss in this paper) is managing the jobs workflow to assure appropriate comfort to the user and spare his/her time.

The Virtual Laboratory system is developed in Poznań Supercomputing and Networking Center in collaboration with the Institute of Bioorganic Chemistry and Radioastronomy Department of Nicolaus Copernicus University.

2 MOTIVATIONS

Experiments executed in the science laboratories are complex and consist of many stages [5]. Usually it looks as follows: a scientist prepares a sample and/or input data (e.g. parameters) which will be measured/computed. Next she/he

uses laboratory devices to achieve data which are the processed by a specialized software. Processing can include the visualization stage if it is needed to assess the results. In case of undesirable results some measurement stages should be repeated. At each stage the scientist decides which way the research should go next. As we can see, the experiment process execution may consist of very similar stages in many scientific disciplines (laboratories). The given experimental process is often executed recurrently by some parameters modification. Obviously, the presented scenario is typical but we realize that scholars can have more sophisticated needs [6].

Thus, we can define a graph which describes the execution path specified by a user. Nodes in this graph correspond to experimental or computational tasks. Edges (links) correspond to the path the measurement execution is following. In nodes we have defined a type of application and its parameters. In links the passing conditions which substitute decisions made by the user are defined. These conditions are connected with applications and let us determine if the application is finished with the desired results (if not, the condition is not met). This execution graph with specified nodes and links (with conditions) is called the Dynamic Measurement Scenario (DMS). The term "dynamic" is used, because the real execution path is determined by the results achieved on each measurement stage. All of these parameters can be written down using a special language: the Dynamic Measurement Scenario Language (DMSL).

Let us resume, the most important advantages of MDS:

- connection of different types of jobs (experimental and computational),
- speed up of tasks sequence execution,
- simplifying scenario monitoring,
- possibility of multiple use of a given scenario,
- legible way of the workflow control,
- possibility of defining many measurement execution ways.

The concept of the Dynamic Measurement Scenarios allows defining the process of an experiment in any way, from pre-processing, through executing the experiment, to the post-processing and visualization tasks. Users are also allowed to add their own module as a part of the scenario. Defining the measurement scenario allows to spare a lot of time during computation. The user does not have to wait for the end of a given process stage to submit another one. It is made automatically.

Initially, we divide the experiment execution process into four steps: preprocessing (data preparation to use as an input source), experiment (executed on real device), postprocessing (output data processing) and visualization.

This simple task execution chain is called the Static Measurement Scenario (SMS). It allows connecting particular stages only between themselves. Here, the execution path is specific and the user can not manipulate it.

To increase the possibility of the jobs scenario the dynamic measurement scenario (DMS) was defined. In the DMS model, besides the definition of the tasks execution sequence, we can also define some extra connections (e.g. loops, parallel connections), conditions on the connections and different lengths of the

execution paths. In fact, the DMS allows to define many SMS models in one scenario. Thanks to the possibility of the conditions defining on the connections paths, the real path is determined during execution and can depend on computational results.

To properly define the Dynamic Measurement Scenario the system has to have knowledge about available applications and connections which are enabled to create. In case of the laboratory type, where the processing software is well known, it seems to be easy. The issue will be more complex when we want to define the DMS model for a wider range of virtual laboratories.

To solve this problem we have defined a special language (DMSL) which determines the following conditions: names of the connected applications, a condition (or conditions) connected with a given path, an additional condition which has to be met to pass a given path (e.g. when special conversion between application is needed) and, finally, a list of input or output files for applications.

An expertise from a particular discipline is necessary to define rules for DMS. It can be done by a laboratory administrator together with a domain expert.

3 DESIGNING

The DMS schema must be designed by the VLab administrator before it is available for users. Creating a new Dynamic Measurement Scenario needs a special attitude. We assume that some stages will be performed by the computer science engineer and scientist from a scientific discipline DMS concerns, and some by application.

The designing of the DMS consists of the following stages:

- application analyzing,
- connection diagram preparing,
- describing additional dependencies in the connection diagram,
- applications and links description generating,
- measurement scenario description generating.

All stages are tightly connected between themselves and should be executed in the presented sequence. The first three phases should be performed, as we mentioned before, by the VLab administrator and engaged scientist (see 3.1, 3.2, 3.3). The last two can be done by a special application which take into consideration user's rights and information introduced by the user (see 3.4, 3.5). Next, we will analyse each stage of DMS designing. Because of the paper limitations we had to focus only on the major aspects of these issues.

3.1 APPLICATION ANALYZING

Each application available in the dynamic scenario must be first analyzed from the functional point of view. Input and output parameters have to be taken into consideration. Also, input and output format files must be described. An exemplary option to describe in these files follows: file type (binary, text), filename

format (if exists): filename mask, filename extension. If the file format type is text, the analyst can analyze information within. A specified pattern connected with parameters important for the final user (calculated by application) can be used to specify link conditions.

For all parameters and files we have to take into consideration the security aspect. It is very important to give users access only to these options which are secure from the administrator's point of view.

3.2 CONNECTION DIAGRAM PREPARATION

After the application analysis the VLab administrator can make the connection diagram. In this phase applications are assigned to execution stages. Exemplary stages for the laboratory of NMR spectroscopy follow:

- acquisition,
- processing,
- visualization,
- structure determination,
- structure visualization.

The following stages do not have to occur one after another in the measurement process but connection dependencies have to be met. The available measurement sequence is determined in the next step. Each application from each stage is connected to another one taking into consideration data achieved in the previous analysing step (see 3.1).

At this stage the necessity of cooperation between a computer science engineer and a scientist connected with a given science discipline is strictly required.

In Fig. 2 we present an exemplary diagram for the Virtual Laboratory of NMR Spectroscopy.

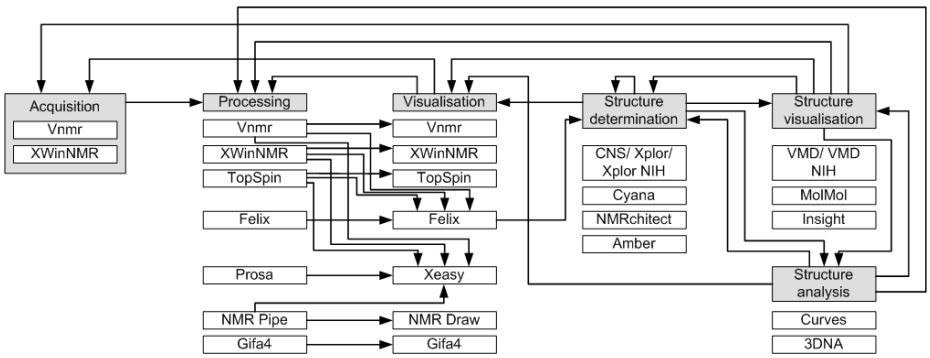

Fig. 2. An exemplary connection diagram for the Virtual Laboratory of NMR Spectroscopy

3.3 DESCRIBING ADDITIONAL DEPENDENCIES IN THE CONNECTION DIAGRAM

At this stage we present conditions on the connections between applications which were created at the previous stage (see 3.1). We should focus our attention on:

- connection conditions,
- conversion issues,
- files types related to links.

Conditions defined on the connections can influence the measurement execution path. After the end of each application execution they are verified and in this way the following execution path is determined. For more information please go to chapter 4.

Conversion is performed when two connected applications have a different input-output file format. It should be described for each connection separately. Depending on the connected applications conversion can be realized in a different way. More about the conversion issues can be found in chapter 5.

It is necessary to determine which type of file can be used as an input file to the target application. Therefore, a set of output files of a given application must be related to each link. Note that a different set of files can be related to each link. This means that some output files of a source application can be used for one target application but not for another one and vise versa.

3.4 APPLICATIONS AND LINKS DESCRIPTION GENERATION

In the following two subsections the generation of application specification, links description and the final measurement scenario file are characterized.

The next steps can be performed using the Scenario Submission Application (SSA) based on the information from the previous stages.

DMS is encoded in the Dynamic Measurement Scenario Language (DMSL). DMSL base on the XML and XSD standard.

The general DMS consists of a description of all possible applications with all parameters available for users.

XSD SCHEMA. XML Schemas express shared vocabularies and allow machines to carry out rules made by people. They provide means for defining the structure, content and semantics of XML documents. An XML schema is an XML based alternative to DTD (Document Type Definition) and describes the structure, content and semantics of an XML document (a set of simple data types which can be associated with XML element types and attributes).

The purpose of an XML Schema is to define the legal building blocks of an XML document, just like a DTD. An XML Schema:

- defines elements that can appear in a document,

– defines attributes that can appear in a document,
– defines which elements are child elements,
– defines the order of child elements,
– defines the number of child elements,
– defines whether an element is empty or can include text,
– defines data types for elements and attributes,
– defines default and fixed values for elements and attributes.

The XML Schema language is also referred to as the XML Schema Definition (XSD).

The next paragraphs describe the XSD schemas which are used by the VLab System. The XSD schemas have been created for the Dynamic Measurement Scenarios definition.

COMPONENTS DESCRIPTION SCHEMA. The Components Description Schema (CDS) defines the Java swing components (JComponent) which are used for displaying the resource properties (See RDS description). The schema structure is presented in a diagram in Fig. 3.

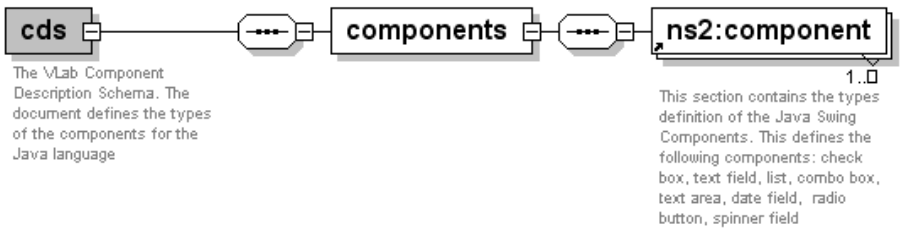

Fig. 3. Component description schema

Also, a set of component attributes is available. Some of the attributes are required by the component, some are optional. They are all used to define the component behaviour i.e. whether the component should be visible, editable, etc. A list of all available attributes is presented in the table below.

No	Name	Description
1	id	the component identifier
2	name	component name
3	class	java class name which implements the given component
4	modelDataAttached	specifies whether the component requires the model data (i.e. JList requires the list of items. From the set of items the user chooses the item)
5	model	the java class which implement the component data model
6	document	the java class, which implements the component document
7	editor	the java class, which implements the component editor
8	enabled	Determines whether this component is enabled. An enabled component can respond to user input and generate events
9	visible	Determines whether this component should be visible when its parent is visible
10	editable	tells whether the component is read only
11	columns	the number of columns (i.e. in the JTextarea)
12	rows	the number of rows (i.e. in the JTextarea)
13	min	the minimum value of the component (i.e. JSpinner)
14	max	the maximum value of the component (i.e. JSpinner)
15	step	the step value of the component (i.e. JSpinner)
16	selectedIndex	the index of the component item which should be selected by default (i.e. JComboBox)
17	dateFormat	specifies the date format of the component
18	icon	the component icon (i.e. JCheckBox)
19	selectedIcon	the component selected icon (i.e. JCheckBox)

Using the schema described above the following Java components have been defined:

- check box (JCheckBox),
- text field (JTextField),
- list (JList),
- combo box (JComboBox),
- text area (JTextArea),
- date field,
- radio button (JRadioButton),
- spinner field (JSpinner).

The combo box component is presented below as an example of an XML file which conforms to the CDS schema.

```
<!-- Combo Box  -->
      <component id="4" name="JComboBox"
      class="pl.psnc.vlab.scenario.gui.resource.components.ComboBoxElement"
      enabled="true" visible="true" editable="false" selectedIndex="0"
      model="pl.psnc.vlab.scenario.gui.resource.model.ModelDataComboBoxModel"
      modelDataAttached="true"/>
```

LINKS DESCRIPTION SCHEMA (LDS). The LDS schema describes the available connections between resources. It also specifies the conditions which have to be taken into consideration while connecting resources. The schema structure is presented below (Fig. 4).

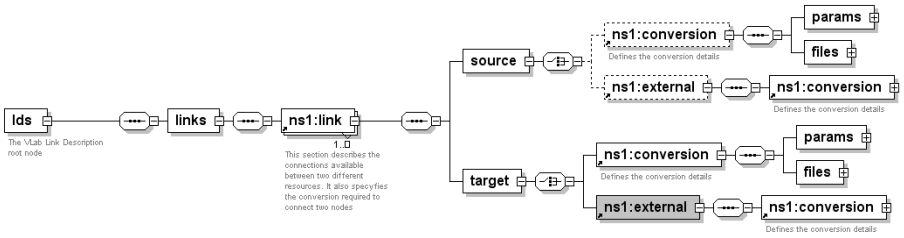

Fig. 4. Links description schema

THE LINK ELEMENT. The link element, as it is shown in the figure below contains the connection definition between two nodes (Fig. 5).

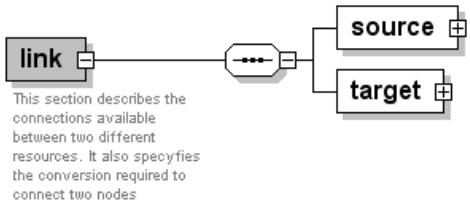

Fig. 5. The link element

The connection attributes are presented in the table below.

No	Name	Description
1	id	the link identifier
2	resourceId	component name
3	externalConversion	necessity of external conversion

RESOURCE DESCRIPTION SCHEMA. The Resource Description Schema (RDS) has been created for the VLab resource description. The general resource description structure is presented in the figure below (Fig. 6).

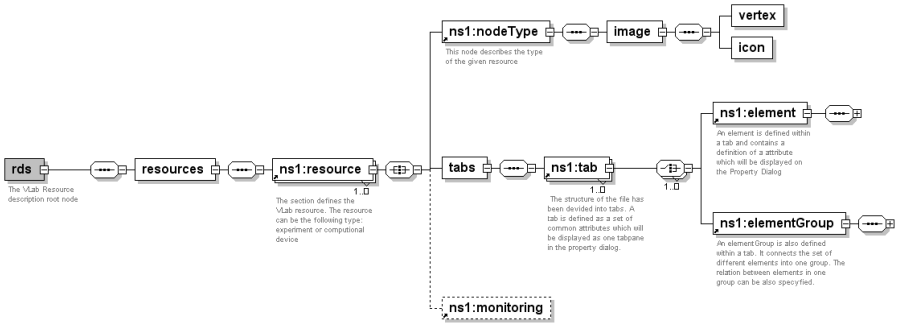

Fig. 6. Resource description schema

The schema defines a list of the resources which are all attached to the resources node. Every resource element contains the following sections:

- **nodeType** - defines the type of the given resource, i.e. NMR experiment of the spectrometer; it also defines the type properties which are used in the Scenario Submission application like type icon, resource icon, etc.,
- **tabs** - the resource properties are grouped in tabbed panes which are displayed in the JTabbedPanes; each tab node contains a set of elements; each element represents a resource property; the elements can also be put together in a group; the relationship between elements in the group can also be defined; an example of the spectrometer properties is shown in the figure below,
- **monitoring** - this optional section contains information about the given resource in the VLab system.

The Varian Spectrometer resource description snippet is shown below.

```
   <resource id="1" name="Varian 300
Mhz" description="Varian 300 MHzICHB">
<!-- ============================ -->
<!-- ======== Node Type ======== -->
<!-- ============================ -->
<nodeType id="1" name="Experiment" subtypeId="1" subTypeName="NMR">
<image>
<vertex path="images/NMR_enabledB.gif"/>
<icon path="images/NMR_enabledS.gif"/>
</image>
</nodeType>
<tabs>

   <!-- ================================ -->
<!-- == tab-Experiment Description == -->
<!-- ================================ -->
```

```
<tab id="1" name="expDesc" displayName="Experiment details">

   <!-- The list of available devices -->
<!-- An experiment description -->
<element id="1" name="experimentDesc" displayName="Experiment description"
visible="true"> <component id="2" modelDataAttached="false" editable="false">
</component>
<out/>
</element>

   <!-- An experiment type -->
<element id="2" name="experimentType" displayName="Experiment type"
toolTipText="Enter an experiment type." visible="true"> <component id="4"
selectedIndex="1" modelDataAttached="true">
<!-- The element model data -->
<modelData>
<item id="0">1D</item>
<item id="1">2D</item>
<item id="2">3D</item>
</modelData>
</component>
<!-- The data selected or inserted by the user -->
<out/>
</element>
```

The resource definition can be visualized by the Scenario Submission Application. The result is presented in Fig. 7.

3.5 MEASUREMENT SCENARIO DESCRIPTION GENERATION

The user is welcome to create the measurement diagram using the Scenario Submission Application (SSA).

The applications and links definition specified in the previous stage are imported to the SSA. A List of imported applications take into account the user's rights. Thus, it consists of only these applications descriptions (and possible connections) which are available for the user.

After finishing the measurement scenario defined by the user, a new DMS description is generated. It is defined on the basis of the diagram and information added by user. Only the chosen application descriptions with specified parameters are placed there.

An exemplary diagram for NMR measurement is presented in Fig. 8.

4 CONNECTION

The user can define conditions on the connections which can influence the measurement workflow. These conditions concern the computational or experimental

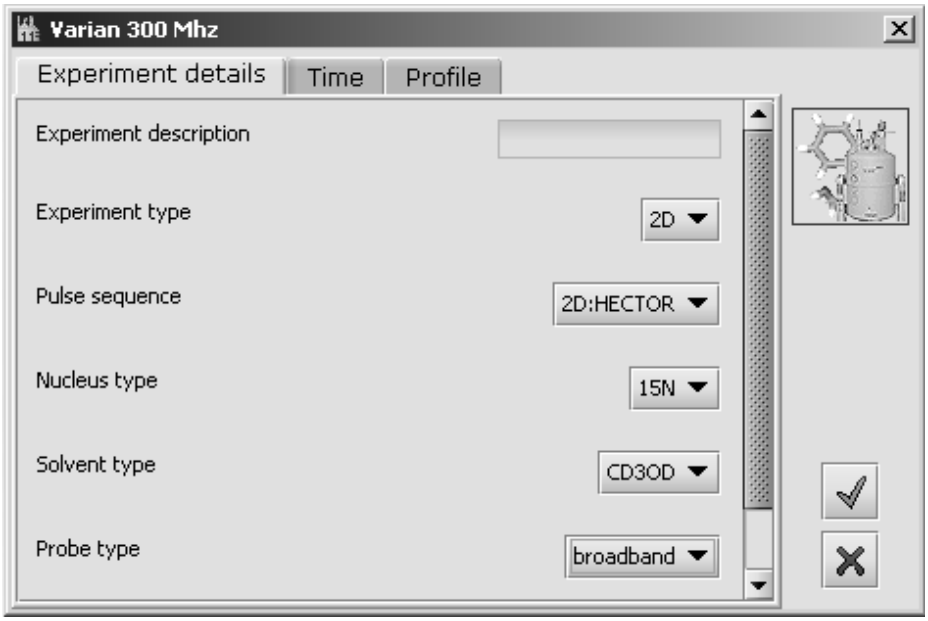

Fig. 7. Visualization of RDS file snippet

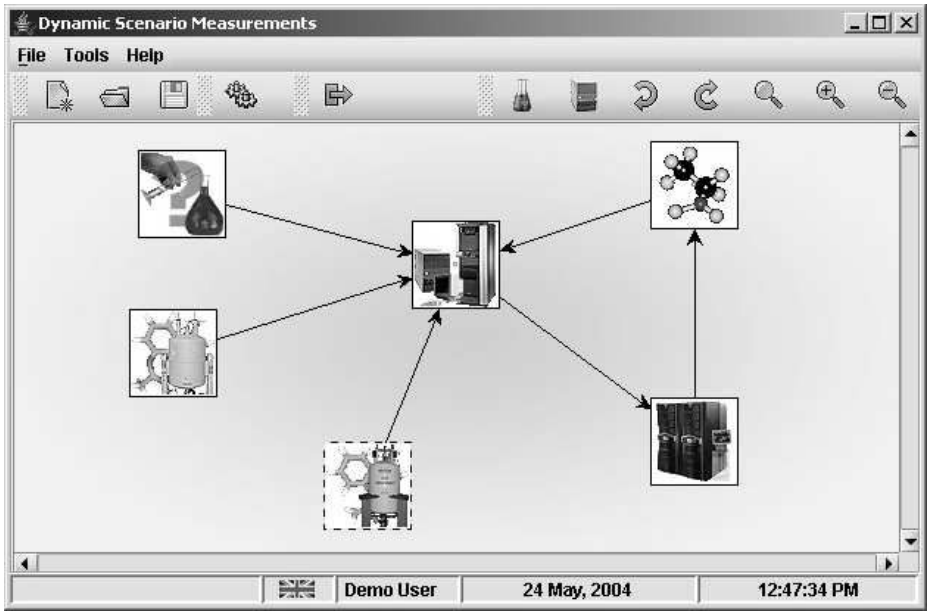

Fig. 8. The Scenario Submission Application with an exemplary diagram

results. A set of available conditions to be specified by the user (while a new measurement is defined) is determined on the "describing additional dependencies" stage of the DMS creation process (see 3.3). To enable a given connection all conditions defined there have to be met.

The user is also welcome to define logical conditions between links. There are two types of conditions: *OR* (default) and *AND*. Logical conditions can be defined in the beginning and end of connections. Their meaning depends on their localization.

BEGINNING CONDITIONS. The Beginning conditions are connected with the results achieved at the computational stage. Each link is marked by logical ***1*** if all conditions are met or logical ***0*** otherwise. The default logical operation defined between links is *OR*, which means that each execution path can be done separately. In case of the *AND* operation simultaneously execution of two or more paths is possible on condition that achieving results in the source application are met (value ***1*** on each link). This situation is illustrated in Fig. 9.

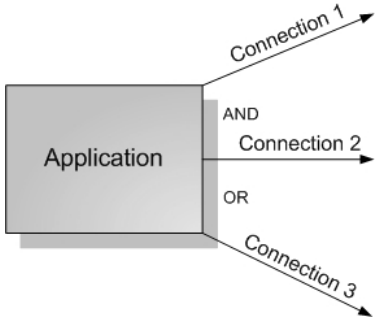

Fig. 9. Defining out-connections conditions

ENDING CONDITIONS. The ending conditions are defined to assure that all necessary data is available before the start of the next processing task.

Similarly as before, for the *OR* condition the link is independent from the other ones. This means that if conditions on a given link are met, the measurement process can go on without waiting for other processes.

The *AND* condition is used when results computation from a few sources is needed. In this way we can force waiting for finishing a number of processes and only then run the next task.

The discussed situation is presented in Fig. 10.

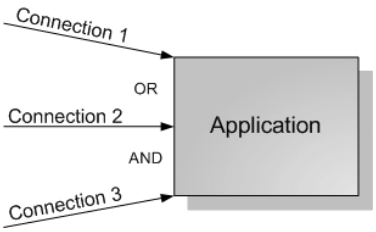

Fig. 10. Defining in-connections conditions

5 CONVERSION

In this section we want to focus on the conversion issues. As we mentioned before, conversion is made when input and output file application formats are different. We assume that conversion may be done in four different ways:

- setting up the switch in the source application to export data in an appropriate format (1),
- using a program on the source server to convert the output file (mini post-processing) (2),
- using a program on the source server to convert the output file (mini pre-processing) (3),
- setting up the switch in the target application to import data in an appropriate format (4).

The discussed situation is illustrated on figure below (Fig. 11).

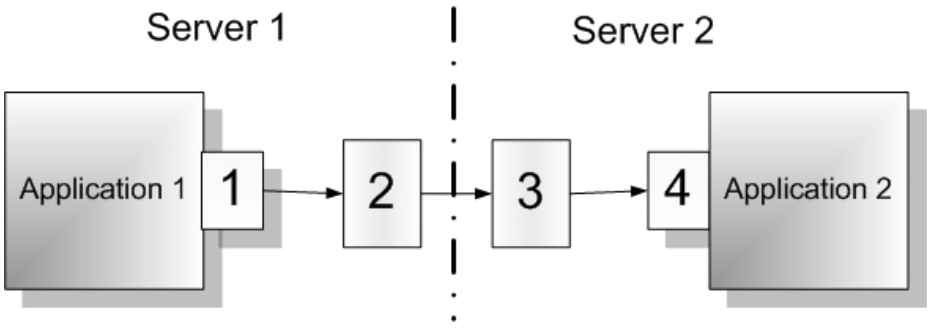

Fig. 11. Four ways of conversion

The conversion way has to be matched with a given connection. The preferred way is using the application switches because it usually does not cause additional CPU power consuming. Please note that not every conversion way will be possible in each situation.

6 Performing

This chapter presents how DMS is performed in the Virtual Laboratory system.

After preparing and sending the measurement scenario to the system, it is received by the Scenario Execution (SE) module (see Fig. 1). We can say that SE is the "heart" of the whole system. Briefly put, the SE receives scenario, decomposes it, and send each task to the system one after another.

We can describe the SE algorithm as follows:

1. Start by receiving client scenario data (XML structure describing graph connections and node details).
2. Create a directed graph and detect initial node(s).
3. Call the Monitoring module in a synchronous mode to create a new scenario - wait to receive new scenario identifier.
4. Create the first task description for a new scenario and call the Monitoring module to create an identifier for it.
5. Create a datagram with task description and send it to the Global Scheduler module.
6. Check if other initial tasks exist. If so, go to step 4.
7. Wait for response from Monitoring (as a separate thread) saying that the task has finished. Check if other tasks exist in the current scenario. If not, finish the algorithm, otherwise go to step 8.
8. Analyze graph connections:
   - Check if all preceding tasks have finished,
   - Check the incoming and outgoing links from the current node (task),
   - Check passage conditions between the current task and the next ones,
   - Contact the Data Management System module if necessary; download files and analyze them; check if they contain data which meet links conditions.
9. Create new datagram(s) for new task(s) and send it(them) to the Global Scheduler module.
10. Go to step 7.

XML files with a scenario description (received from the SSA) and current tasks status are locally stored on the SE machine to restart module in case of any accident.

7 PUTTING INTO PRACTICE

The Virtual Laboratory system is developed in Pozna Supercomputing and Networking System [7] in collaboration with the Institute of Bioorganic Chemistry [8] and Radioastronomy Department of Nicolaus Copernicus University [9].

The Dynamic Measurement Scenarios will apply in two of our exemplary implementation of virtual laboratories: the Virtual Laboratory of NMR Spectroscopy (devices: Bruker Avance 600, Varian Unity 300) and the Virtual Laboratory of Radiotelescopy (device: radiotelescope - diameter 32m) available to the Grid users [10].

Basing on the DMS we would like also entirely automating the process of defining and executing the VLBI experiment in the Virtual Laboratory of Radiotelescopy. Virtual laboratory user only needs specify necessary observation parameters. Parameters distribution to radio telescopes, recalculation them to appropriate radio telescope values, launching observation on time, data transporting and correlating and post-processing execution can be performed automatically.

8 CONCLUSIONS

Virtual Laboratories are undoubtedly the technology of the future. They are perceived as a remedy for problems connected with access or buying costs of expensive and rare devices. Current technologies connected with VL are in their first stadium of development. New created services allow to run simple experiments, which can be managed by changes in a limited parameter pool.

The Dynamic Measurement Scenario is a flexible and very convenient way to define workflow in many types of virtual laboratories. It allows to spare a lot of user's time and speed up of the project realization. Thanks to the Submission Scenario Application the user can easily define, submit and monitor the progress of the DMS realization. In a case of an interactive task realization, it is possible to reserve a time slot on a laboratory device or computational server and notify the user about that. Therefore, the user avoids waiting for free resources. Of course, the scheduling and reservation processes are made by other modules, but on the SE (which executes the DMS) request.

Despite the necessary time to prepare the DMS (application analysing, connection defining, etc.) the Dynamic Measurement Scenario idea seems to be worth being introduced in many systems where the defining complex workflow is needed to preform the science job.

To prove our assumptions, and as an exemplary implementation, it is planned to make the NMR Spectrometer (Bruker Avance 600) and radiotelescope (diameter 32m) available to the Grid users [10].

References

1. Afsarmanesh, H., Benabdelkader, A., Kaletas, E.C., Garita, C., and Hertzberger, L.O.: Towards a Multi-layer Architecture for Scientific Virtual Laboratories. In: Bubak, M., Afsarmanesh, H., Williams, R., Hertberger, B., (Eds.), Proc. Int. Conf. High Performance Computing and Networking, Amsterdam, May 8-10, 2000, Lecture Notes in Computer Science 1823, 162-176, Springer, 2000
2. Afsarmanesh, H., Belleman, R.G., Belloum, A.S.Z., Benabdelkader, A., van den Brand, J.F.J., Eijkel, G.B., Frenkel, A., Garita, C., Groep, D.L., Heeren, R.M.A., Hendrikse, Z.W., Hertzberger, L.O., Kaandorp, J.A., Kaletas, E.C., Korkhov, V., de Laat, C.T.A.M., Sloot, P.M.A., Vasunin, D., Visser, A., Yakali, H.H.: VLAM-G: A Grid-based virtual laboratory. Scientific Programming, (Special issue on Grid Computing) vol. 10, nr 2 pp. 173-181. (R.H. Perrott and B.K. Szymanski, editors), IOS Press, 2002. ISSN 1058-9244.

3. Nicholas. P., Fushman. D., Ruchinsky. V., Cowburn. D.: The Virtual NMR Spectrometer: a computer program for efficient simulation of NMR experiments involving pulsed field gradients. (2000)
4. Lawenda. M., Meyer. N., Rajtar. T.: General framework for Virtual Laboratory. The 2nd Cracow Grid Workshop, Cracow, Poland, December 11 - 14, 2002
5. Lawenda, M., Meyer, N., Rajtar, T., Okoń, M., Stokłosa, D., Stroiński, M., Popenda, Ł., Gdaniec, Z., Adamiak, R.W.: General Conception of the Virtual Laboratory. International Conference on Computational Science 2004, LNCS 3038, pp. 1013-1016, Cracow, Poland, June 6-9, 2004
6. Lawenda, M., Meyer, N., Rajtar, T., Okoń, M., Stokłosa, D., Stroiński, M.: Job workflow in the Virtual Laboratory. GLOBAL GRID FORUM 10, Grid Workflow Workshop, Berlin, Germany, March 9, 2004
7. Poznan Supercomputing and Networking Center
`http://www.psnc.pl/`
8. Institute of Bioorganic Chemistry Polish Academy of Sciences
`http://www.ibch.poznan.pl/`
9. Radioastronomy Department of Nicolaus Copernicus University
`http://www.astro.uni.torun.pl/`
10. Virtual Laboratory project `http://vlab.psnc.pl/`