INFORMATION SOCIETY TECHNOLOGIES
(IST)
PROGRAMME

OpenMolGRID

SPECIFICATION OF SOFTWARE MODULES FOR MOLECULAR STRUCTURE GENERATION AND THEIR GRID INTERFACE COMPONENTS

Contract Reference	**IST-2001-37238**
Document identifier:	**OpenMolGRID-3-D3.1-0101-1-3**
Date:	**07/10/2003**
Work package:	**WP3: Computational Molecular Engineering of New Compounds and Materials**
Partner:	**UT, UU, FZJ, CGX**
Lead Partner:	**UT**
Document status:	**APPROVED**
Classification:	**PUBLIC**
Deliverable identifier:	**D3.1**

Abstract: This document describes requirements and specification of software modules in the Molecular Engineering Environment.

Delivery Slip

	Name	Partner	Date
From	S Sild	UT	18/09/2003
Verified by	S Sild	UT	07/10/2003
Approved by	GHF Diercksen (TC) R Ferenczi (QE)	OMC CGX	21/10/2003 07/04/2004

Document Log

Issue	Date	Comment	Author
0-0	03/02/2003	First version	S Sild
0-1	07/02/2003	Second stable version	S Sild, A Papp
0-2	30/06/2003	Third stable version	S Sild
1-0	16/09/2003	Fourth version	S Sild
1-1	18/09/2003	Fifth revised version	S Sild
1-2	07/10/2003	Sixth version	S Sild
1-3	12/04/2004	Seventh version	S Sild

Document Change Log

Issue	Item	Reason for Change
0-1	Updates to requirements, use case from A Papp incorporated	Refers to Original Submission to EC
0-2	Comments from M Karelson and GHF Diercksen addressed.	Refers to version that was resubmission to EC before review meeting
1-0	New template adapted, information related to theoretical background moved to D3.6, document structure changed, recommendations from the project reviewers addressed.	Changes made to address reviewers comments
1-1	Reference style to deliverables changed; Corrections to grammar; Small improvements in Section 1 and 2.	Comments from B Schuller and D McCourt
1-2	Some typing and grammatical errors corrected.	Internal review process
1-3	Project information form on page 4; Updates to Section 4	Change of project information; XML schema updated.

Files

Files in this section relate to actual storage locations on the BSCW server located at https://hermes.chem.ut.ee/bscw/bscw.cgi. The URL below describes the location on BSCW from the root OpenMolGRID directory

Software Products	User files / URL
Word 2000/XP	OpenMolGRID/Workpackage 3/Deliverables/ OpenMolGRID-3-D3.1-0101-1-3-StructGen

Project information

Project acronym:	OpenMolGRID
Project full title:	Open Computing GRID for Molecular Science and Engineering
Proposal/Contract no.:	IST-2001-37238
European Commission:	
Project Officer:	Annalisa BOGLIOLO
Address:	European Commission - DG Information Society F2 - Grids for Complex Problem Solving B-1049 Brussels Belgium
Office	BU31 4/79
Phone:	+32 2 295 8131
Fax:	+32 2 299 1749
E-mail	annalisa.bogliolo@cec.eu.int
Project Coordinator:	Mathilde ROMBERG
Address:	Forschungszentrum Jülich GmbH ZAM D-52425 Jülich Germany
Phone:	+49 2461 61 3703
Fax:	+49 2461 61 6656
E-mail	m.romberg@fz-juelich.de

Contents

1 Introduction

1.1 Purpose and scope

One of the core applications of the OpenMolGRID system is molecular engineering. For this, several modules for the generation of chemical structures with predefined target properties have to be developed. The purpose of this document is to analyse requirements relevant to the Molecular Engineering Environment. The analysis covers the definition of user requirements to the system, specification of input/output formats, the specification of application specific metadata and the selection of most appropriate methods for the structure generation.

1.2 Document Overview

The primary objective of the Molecular Engineering Environment is to provide software prototype for the construction of molecular structures with given chemical properties or biological activities. The detailed description of the molecular engineering process is described in Deliverable D3.6 [1]. The design of molecular structures relies on the basic knowledge that the properties of molecular compounds are determined by the properties of the molecular fragments and their interaction. The Molecular Engineering Environment implements algorithms to combine molecular fragments into new compounds. For all generated candidates the target properties are estimated by the quantitative structure-property relationships (QSPR) or quantitative structure-activity relationships (QSAR) models and candidates that match the predefined target property are presented to the end user. The QSPR/QSAR predictions use software modules from the data mining environment, which are developed within the WP2 of the OpenMolGRID project.

The modules developed for the Molecular Engineering Environment will be integrated into the general architecture of the OpenMolGRID system, which is described in detail in Deliverable D4.5a [2].

1.3 Document Structure

The specification is divided into four sections, as follows:

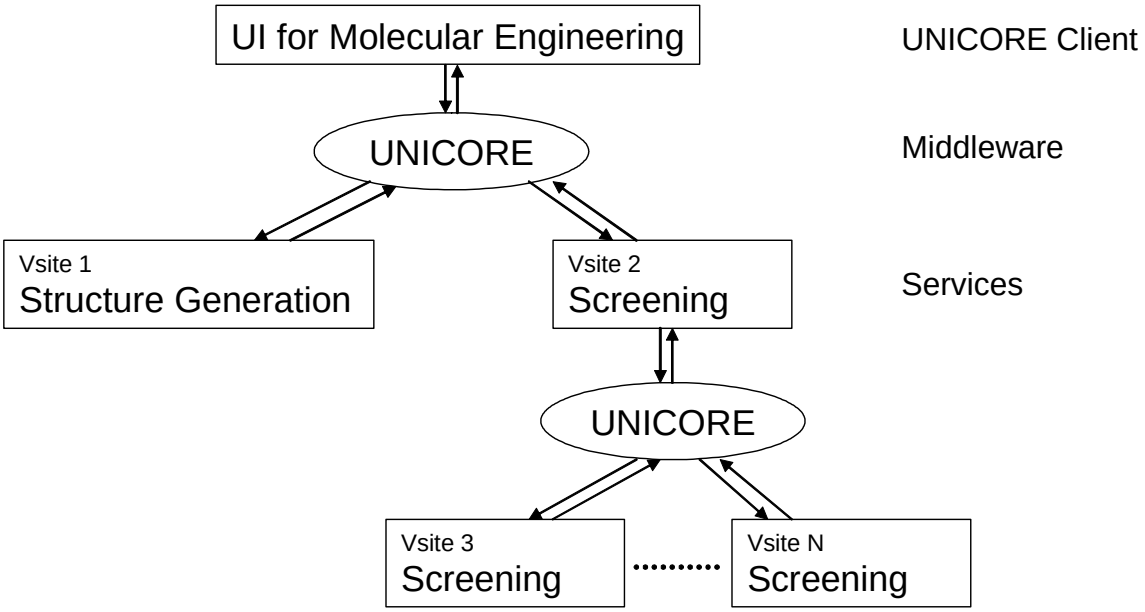
- Section 2 describes requirements and use cases for the Molecular Engineering Environment.
- Section 3 is describes UNICORE applications in the Molecular Engineering Environment.
- Section 4 contains the literature references.

2 Requirements to Molecular Engineering Environment

The primary task of the Molecular Engineering Environment is the construction of molecular structures with predefined chemical properties or biological activities. The analysis of the user requirements for the Molecular Engineering Environment identified three general use cases for the generation of molecular structures. The detailed descriptions for these use cases are given in Sections 2.1 (Evolutionary generation of molecular structures with predefined target properties), 2.2 (Enumeration of fragment library) and 2.3 (Enumeration of fragment library to find subset of structures with predefined target properties). These use cases use two methods for the structure generation.

The method to be selected for the generation of the molecular structures depends heavily on the size of the fragment library, which determines the number of structures that can be generated. For the fragment library with a modest size the full enumeration is a feasible option. In this case all the possible combinations are evaluated and tested. In the case of large fragment libraries it is not possible to evaluate and test all the possible combinations and evolutionary structure generation methods have to be used to reduce the amount of computations.

The structure generation process in the Molecular Engineering Environment involves several subtasks. The first step is the generation of molecular structures and the second step is the screening of generated structures with the help of QSPR/QSAR models [3, 4]. Therefore, facilities for realising these complex workflows are required. The necessary workflow elements are provided by the OpenMolGRID architecture and are used as a basis for the Molecular Engineering Environment. The architecture of the OpenMolGRID system is described in detail in Deliverable D4.5a [2]. The OpenMolGRID system is set up on top of the UNICORE infrastructure and adds support for the automated workflow processes. The support for automated workflows on top of the Grid adds a great deal of flexibility to the molecular engineering process. For example, depending on the resources offered at different sites, various subtasks can be executed there. In addition, computationally intensive tasks such as the screening of candidate structures can be split up and distributed to more than one system. For example, if the structure generation software is provided by the site 1 and software for the screening of candidate structures is provided by the site 2 and the screening process is the limiting factor, then the molecular engineering process on top of the Grid infrastructure can be described with the following scheme:



2.1 Use case: Evolutionary generation of molecular structures with predefined target properties

Scenario: The user defines a maximal number of candidate structures and target property values for structures to be generated. The user starts structure generation module to develop molecular structures with given target properties. The generation of structures is performed from large fragment libraries, using an evolutionary algorithm that enables to avoid the combinatorial explosion from all possible fragment combinations. The list of generated structures is saved to storage.

Preconditions:
1. Fragment library with fragment descriptors.
2. One or several QSPR/QSAR models.

Trigger: User starts evolutionary structure generation tool.

Description:
1. System displays a list of available QSPR/QSAR models.
2. User selects one or more QSPR/QSAR models and defines the targeted property values.
3. System extracts the relevant molecular descriptors for selected models. Based on the target property values, system proposes hypothesis about the optimal descriptor values.
4. User refines hypothesis (optional).
5. User refines the library of core and substituent fragments, e.g. excludes certain fragments from consideration (optional).
6. User starts the structure generation.
7. System uses evolutionary algorithm to generate set of molecular structures with given target properties.
8. User uses visualisation tool to view or edit list of generated structures and predicted property values.
9. User saves the list of generated structures to storage.

Extensions: N/A

Variations: N/A

Postconditions: List of structures saved to storage.

2.2 Use case: Enumeration of virtual library

Scenario: User makes a selection of core and substituent fragment structures from the fragment library. After that, user starts the structure enumeration process, where selected core structures with given substitution points are combined with a list of substituent fragments (R-groups) to build molecular structures. The resulting list of structures is saved to storage and can be further refined by screening with QSPR/QSAR models.

Preconditions: Fragment library.

Trigger: User starts structure enumeration tool.

Description:

1. System displays a list of core structures that are available for the structure generation in the fragment library.
2. User selects one or more core structures from the list.
3. For each core structure, user defines substituent fragment list for each substitution site.
4. System generates connectivity information for the virtual library.
5. User uses visualisation tool to check connectivity information (list of applicable fragments at each substitution site) and saves connectivity information.
6. User initiates the structure enumeration
7. System starts the structure enumeration and builds molecular structures
8. User saves list of generated structures to the storage

Extensions:

1a User uses sub-structure search to limit the list of available core structure fragments.

1a1 System displays core structures that match sub-structure search criteria.

8a User uses visualisation tool to check the generated molecular structures and then saves only selected structures to the storage.

Variations:

3a User uses sub-structure search to select a list of substituent fragments.

3b User specifies applicable substituent fragment types.

3c User specifies combination of 3a and 3b.

3d User selects all substituent fragments.

3f User selects predefined list of substituent fragments.

Postconditions: List of structures saved to storage.

2.3 Use case: Enumeration of virtual library to find subset of structures with predefined target properties

Scenario: User has a fragment library and one or more QSPR/QSAR models. This information can be used to construct the molecular structures with predefined target properties or activities. User defines the values of target properties and initiates the enumeration of virtual library (described in use case 2.2). The resulting molecular structures are then screened using pre-selected QSPR/QSAR models and a subset of candidate structures satisfying the user-defined criteria is picked out.

Preconditions:

1. Fragment library.
2. One or more QSPR/QSAR models.

Trigger: User starts structure prediction tool.

Description:

1. System displays a list of available QSPR/QSAR models.
2. User selects one or more QSPR/QSAR models and defines property values for the selection criteria
3. System displays tools for enumeration of virtual library
4. User generates list molecular structures by enumerating virtual library (Use case: Enumeration of virtual library)
5. System uses selected QSPR/QSAR models to predict property/activity values and select subset of structures that satisfy user-defined criteria.
6. User uses visualisation tool to view or edit list of generated structures and predicted property values
7. User saves list of generated structures to the storage

Extensions:

2a User defines additional criteria with target descriptor values (e.g. range of molecular weight values, number of hydrogen bonds, etc.).

Variations: N/A

Postconditions: List of structures with predicted property values saved to storage.

3 UNICORE Applications

3.1 Enumeration of Virtual Library

The methodology for generating molecular structures by enumerating the virtual library is given in Deliverable D3.6 [1]. Often this is a two-stage process, where initial step involves the enumeration of structures in the virtual library and produces a list of candidate structures. The second stage uses QSPR/QSAR predictions to select (or screen) best structures that have predicted property values closest to the target properties.

3.1.1 Input Formats

The structure enumeration task takes a list of molecular fragments as input. The XML schema for the fragment list is defined in Section 4.1 (FragmentList). The atomic coordinates of fragment structures are defined in Chemical Markup Language (CML) format. In addition, the MDL's MOL file format is supported.

If screening is performed, then additional input includes QSPR/QSAR models in XML format and target property/activity values. The respective XML schema is defined in Deliverable D2.1, Section 9.5 [4].

If fragment descriptors are used for screening, then additional input includes fragment descriptor values in XML format. The respective XML schema is defined in Section 4.2 (FragmentDescriptorFile).

3.1.2 Output Formats

This task generates list of molecular structures in XML format. The XML schema for the structure list is defined in Deliverable D2.1, Section 9.1 [4].

If enumeration process was followed with screening process, then calculated descriptor and predicted property/activity values are stored in separate XML files. The respective XML schemas are defined in Deliverable D2.1, Sections 9.3 (DescriptorFile) and Section 9.4 (PropertyFile).

3.1.3 Application Specific Metadata

A table with configurable parameters that control the enumeration of virtual library:

Parameter Name	type	Description
Max structures	Integer	The maximum number of structures that is allowed to be generated. Default is zero, which means that the number of structures is not limited.
Output Type	String	Representation of atomic coordinates in generated structures. Possible values are "2D" and "3D". Default is "3D"
Store Fragments	Boolean	Save a list of fragment ID numbers that were used to assemble the molecule. Default value is "false".

3.1.4 Requirements

The output options for the structure generation algorithm should include 2D and 3D representations. The visualisation of the generated molecular structures should include options to view structures in 2D or 3D representation. The output should include the following information together with generated molecular structures:

- List of fragment ID numbers that were used to assemble the molecule (optional)
- Name of the fragment library that was used as a source for molecular fragments

- Name and version of the construction algorithm
- Generation time

When preparing virtual library for structure enumeration, each substitution site at the core structure should be controlled by the following substitution options:

- Use substituents that have compatible types
- Use substituents from predefined substituent list
- Use substituents from user defined list of substituents. User should have an option to use substructure search to select matching fragments to the list.

Before the structure enumeration is started, there should be possibility to check the number of molecular structures that will be generated. The structure generation process should give feedback about the generation progress (e.g. percentage of generated structures).

3.2 Evolutionary Generation of Structures

The evolutionary approach for constructing molecular structures with predefined property values involves several steps, which are carried out in the following order: (1) selection of appropriate QSPR/QSAR models for target properties, (2) definition of target property values, (3) definition of optimal target descriptor values, (4) evolutionary generation of molecular structures on the basis of a fragment library, and (5) QSPR/QSAR validation of candidate molecules against property values.

3.2.1 Input Formats

The structure enumeration task takes a list of molecular fragments as input. The XML schema for the fragment list is defined in Section 4.1 (FragmentList).

QSPR/QSAR models in XML format and target property/activity values. The respective XML schema is defined in Deliverable D2.1, Section 9.5 [4].

Fragment descriptor values in XML format. The respective XML schema is defined in Section 4.2 (FragmentDescriptorFile).

3.2.2 Output Formats

The molecular structures are represented in XML format. The respective XML schema is defined in Deliverable D2.1, Section 9.1 [4].

Calculated descriptor and predicted property/activity values are stored in separate XML files. The respective XML schemas are defined in Deliverable D2.1, Sections 9.3 (DescriptorFile) and Section 9.4 (PropertyFile).

3.2.3 Application Specific Metadata

A table with configurable parameters that control the evolutionary generation of molecular structures:

Parameter Name	type	Description
Output Type	String	Representation of atomic coordinates. Possible values are “2D” and “3D”. Default is “3D”
Store Fragments	Boolean	Save list of fragment ID numbers that were used to assemble the molecule. Default value is “false”.
Iterations	Integer	Number of iterations.
Subset Size	Integer	Size of the subset to be selected for the next step.

Candidates Limit	Integer	Number of candidate structures to be selected.

3.2.4 Requirements

The output options for the evolutionary structure generation algorithm are the same as for the enumeration of the fragment library. In addition, the required substitution options for the substitution sites are the same.

The evolutionary generation of structures should have following options for pre-defined activity/property criteria:

- Property/activity is less than user defined threshold
- Property/activity is larger than user defined threshold
- Property/activity is equal to user defined threshold
- User defined range for property/activity values

In addition, the following user-defined constraints should be available to limit the number of generated structures:

- Criteria for user selected descriptor values (range, min/max value)
- Presence or absence of some functional group or fragment

The evolutionary structure generation algorithm selects at the end of the each iteration a subset of structures to the next step that are closest to user defined input criteria. The size of this subset should be configurable by user. At the end of iteration, the status of calculation should be updated to show the number of generated structures, the number of structures matching target descriptors, and the number of structures that match the target properties.

The visualisation of the generated molecular structures should include options to view structures in 2D or 3D representation. The descriptors and properties that were calculated during the structure should be displayed together with the structures (e.g. as a spreadsheet). In addition, it should include facilities to select and remove unsuitable candidates from the list of generated structures.

4 XML Schema Definitions

4.1 http://www.openmolgrid.org/namespaces/FragmentList

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd=http://www.w3.org/2001/XMLSchema
            xmlns=http://www.openmolgrid.org/namespaces/StructureList
           targetNamespace=http://www.openmolgrid.org/namespaces/StructureList
            elementFormDefault="qualified">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">
      XML schema for a list of fragments
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="fragmentList">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="source" type="sourceType" minOccurs="0"
                     maxOccurs="1"/>
        <xsd:element name="fragment" type="fragmentType" minOccurs="0"
                     maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:simpleType name="idType">
    <xsd:restriction base="xsd:long">
      <xsd:minInclusive value="1"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:complexType name="sourceType">
    <xsd:sequence>
      <xsd:element name="softwareName" type="xsd:string" minOccurs="0"
                   maxOccurs="1"/>
      <xsd:element name="version" type="xsd:string" minOccurs="0"
                   maxOccurs="1"/>
      <xsd:element name="method" type="xsd:string" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="architecture" type="xsd:string" minOccurs="0"
                   maxOccurs="1"/>
      <xsd:element name="operatingSystem" type="xsd:string" minOccurs="0"
                   maxOccurs="1"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="fragmentType">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="coordinates" type="coordinatesType" minOccurs="0"
           maxOccurs="1"/>

    </xsd:sequence>
    <xsd:attribute name="id" type="idType" use="required"/>
  </xsd:complexType>

  <xsd:simpleType name="formatType">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="CML"/>
      <xsd:enumeration value="MOL"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="categoryType">
```

```
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="2D"/>
      <xsd:enumeration value="3D"/>
      <xsd:enumeration value="3Dopt"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:complexType name="coordinatesType">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
       <xsd:attribute name="format" type="formatType" use="required"/>
       <xsd:attribute name="category" type="categoryType" use="required"/>
    </xsd:extension>
  </xsd:simpleContent>
 </xsd:complexType>

</xsd:schema>
```

4.2 http://www.openmolgrid.org/namespaces/FragmentDescriptorFile

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd=http://www.w3.org/2001/XMLSchema
            xmlns=http://www.openmolgrid.org/namespaces/DescriptorFile
            targetNamespace=http://www.openmolgrid.org/namespaces/DescriptorFile
            elementFormDefault="qualified">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">
      XML schema for fragment descriptors
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="fragmentList">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="source" type="sourceType" minOccurs="0"
                     maxOccurs="1"/>
        <xsd:element name="metadata" type="metadataType" minOccurs="0"
                     maxOccurs="unbounded"/>
        <xsd:element name="fragment" type="fragmentType" minOccurs="0"
                     maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:simpleType name="idType">
    <xsd:restriction base="xsd:long">
      <xsd:minInclusive value="1"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:complexType name="sourceType">
    <xsd:sequence>
      <xsd:element name="softwareName" type="xsd:string" minOccurs="0"
                   maxOccurs="1"/>
      <xsd:element name="version" type="xsd:string" minOccurs="0"
                   maxOccurs="1"/>
      <xsd:element name="architecture" type="xsd:string" minOccurs="0"
                   maxOccurs="1"/>
      <xsd:element name="operatingSystem" type="xsd:string" minOccurs="0"
                   maxOccurs="1"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="metadataType">
    <xsd:sequence>
      <xsd:element name="descriptor" type="descriptorMetaType" minOccurs="0"
```

```
                    maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="fragmentType">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="descriptorList" type="descriptorListType" minOccurs="0"
                    maxOccurs="1"/>
    </xsd:sequence>
    <xsd:attribute name="id" type="idType" use="required"/>
  </xsd:complexType>

  <xsd:complexType name="descriptorMetaType">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="type" type="xsd:string" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="subtype" type="xsd:string" minOccurs="0"
                    maxOccurs="1"/>
      <xsd:element name="category" type="xsd:string" minOccurs="0"
                    maxOccurs="1"/>
    </xsd:sequence>
    <xsd:attribute name="id" type="idType" use="required"/>
  </xsd:complexType>

  <xsd:complexType name="descriptorListType">
    <xsd:sequence>
      <xsd:element name="descriptor" type="descriptorType" minOccurs="0"
                    maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="descriptorType">
    <xsd:complexContent>
      <xsd:restriction base="xsd:anyType">
        <xsd:attribute name="id" type="idType"/>
        <xsd:attribute name="value" type="xsd:float"/>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

</xsd:schema>
```

5 References

1. S Sild, Description of the Molecular Engineering Procedure, OpenMolGRID-3-D3.6-0103-1-2, **2003**.
2. M Romberg, B Schuller, Description of the OpenMolGRID Grid Architecture, Security Architecture, and Infrastructure and the Deployment of the Testbed, OpenMolGRID-4-D4.5a-0110-1-1, **2003**.

3 A Lomaka, U Maran, Description of the Quantitative Structure Property/Activity Relation Model: Model Building and Application, OpenMolGRID-2-D2.4a-0102-1-0, **2003**.

4 S Sild, A Lomaka, Specification of Software Modules for Descriptor Calculation and Model Development and their Grid Interface Components, OpenMolGRID-2-D2.1-0101-1-3, **2003**.