IST-2004-004475

DataMiningGrid

Data Mining Tools and Services for Grid Computing Environments

Specific Targeted Research or Innovation Project
2.3.2.8 Grid-based Systems for Complex Problems Solving

D11(3): Common Requirements Analysis, Specification and Evaluation of DataMiningGrid Interfaces and Services

Due date of deliverable: M24 (30.08.2006)

Actual submission date: 5 November 2006

Start date of project: 1 September 2004 Duration: 27 months

University of Ljubljana (LJU)

Revision: version 07

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	**Public**	**X**
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the Consortium (including the Commission Services)	
CO	Confidential, only for members of the Consortium (including the Commission Services)	

DATAMINING grid

Deliverable D11(3): Common Requirements Analysis, Specification and Evaluation of DataMiningGrid Interfaces and Services

DATA MINING TOOLS AND SERVICES FOR GRID COMPUTING ENVIRONMENTS

Deliverable D11(3): Common Requirements Analysis, Specification and Evaluation of DataMiningGrid Interfaces and Services

Responsible author(s): Vlado Stankovski
Co-author(s): Jernej Trnkoczy

Project funded by the European Community under the 'Information Society Technology' Programme Contract IST-2004-004475

Revision history

Deliverable administration and summary		
Project acronym: DataMiningGrid		ID: IST-2004-004475
Document identifier:	DataMiningGrid-del-D11(3)CommonRequirements-s	
Leading Partner: LJU in collaboration with all Partners		
Report version: 07		
Report preparation date: 05.11.2006		
Classification: Public		
Nature: Report		
Author(s) and contributors: Vlado Stankovski, Jernej Trnkoczy (LJU) in collaboration with all Partners		
Status:		Plan
		Draft
		Working
		Final
	X	Submitted
		Approved

The DataMiningGrid © Consortium has addressed all comments received, making changes as necessary. Changes to this document are detailed in the change log table below.

Date	Edited by	Status	Changes made
01.01.2006	Jernej Trnkoczy	Draft	Template for report.
20.10.2006	Vlado Stankovski	Working	D11 (2) was taken as basis for this document.
05.11.2006	Werner Dubitzky	Submitted	Final consistency checks and minor updates.

Note that other documents may supersede this document. A list of newest public DataMiningGrid deliverables can be found at the http://www.DataMiningGrid.org/deliverables.htm.

Copyright

Citation

Vlado Stankovski, Jernej Trnkoczy (2006) et al. Deliverable D11(3): Common Requirements, DataMiningGrid© Consortium, University of Ljubljana, www.DataMiningGrid.org

Acknowledgements

The work presented in this document has been conducted in the context of the project **IST 2004 004475 DataMiningGrid.** DataMiningGrid is a 27-month project that started on September 1st, 2004 and is funded by the European Commission as well as by the industrial Partners. Their support is appreciated.

The Partners in the project are University of Ulster (UU), Fraunhofer Institute for Autonomous Intelligent Systems (FHG), DaimlerChrysler (DC), Israel Institute of Technology (TECH) and University of Ljubljana (LJU). The content of this document is the result of extensive discussions within the DataMiningGrid© Consortium as a whole. This report owes to a collaborative effort of the above organizations.

More information

Public DataMiningGrid reports are available through DataMiningGrid public Web site (www.DataMiningGrid.org).

Executive Summary

This document specifies the technical requirements, the goals for the development of DataMiningGrid system and services. The Consortium successfully developed the system, and in this document we also describe the level of achievement of the requirements we specified in the beginning of the project.

The Partners identified and elaborated and evaluated the level of achievement of DataMiningGrid requirements in the following areas:

Set of requirements	Level of achievement
Identifying (locating) DataMiningGrid resources by using metadata	Partial (by using the Provenance unit)
Accessing and selecting subsets of data	Complete
Data transfer	Complete
Data (pre-) processing	Complete
Data mining tasks	Complete
Text mining and ontology learning	Complete
Workflow editing and submission	Complete
Data privacy, security and governance	Complete
Integration of domain knowledge	Complete
Grid infrastructure and middleware functionality	Complete
Usability, response times and user-friendliness	Complete

All individual tasks relating to requirements analysis and specification have contributed to this joint deliverable; therefore, D11(3) is the product of the following tasks:

- Task T12: Common requirements specification;
- Task T21: Requirements analysis for data access, transfer, and manipulation; and
- Tasks T61-T64, that is, demonstration tasks contributed by all Partners.

Table of Contents

1 Introduction

The need for data mining in grid computing environments was first illustrated by describing several representative use cases (in Deliverables D61(1) and D62). Some of these use cases are based on efforts currently underway in industry and academia, others demonstrate long-term developments in business and industry. In Deliverables D11(1) and D11(2) we presented the requirements, i.e. a set of features that should be present in the DataMiningGrid system and we provided a motivation for those features.

The detailed design of the DataMiningGrid system was made on the basis of the following considerations:

- The requirements that are contained in the Deliverable D11(2);
- End-user demonstrator scenarios as in Deliverables D61(1) and D62;
- Review comments on Deliverable D11(2) from public (end users) feedback, invited experts and working group members;
- Specifications of (or proposals for) grid data mining tools and services that meet many of these requirements; and
- DataMiningGrid requirements recognized by other projects from the concertation framework of the same strategic objective 'Grids for Complex Problem Solving', such as the SIMDAT, InteliGrid and others.

1.1 DataMiningGrid development approach

The underlying philosophy (approach) of the DataMiningGrid project is depicted in Figure 1. The Consortium decided to use the 'three-legged stool' methodology for project development process. This methodology basically consists of three main concepts:

- Use-case-driven process – all the stages in the process are driven by use cases;
- Architecture-centric – architecture has to be outlined at a very early stage of the project and it evolves over time; and
- Iterative process of improvements of the various models (use case model, requirements model, analysis model, and design model). In the DataMiningGrid project we improve these models at least in three iterations.

The requirements were chosen based on the aspects of the use cases that the Consortium considered most important. As such, one should not assume that the DataMiningGrid components directly support every aspect of each use case.

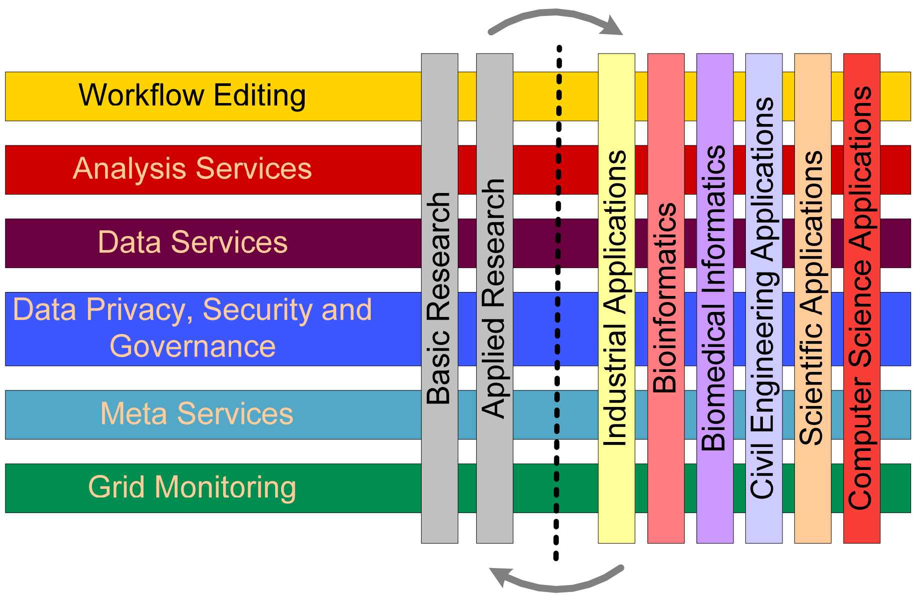

Figure 1: DataMiningGrid application areas and system components.

The use cases were extensively described in Deliverables D61(1) and D62. The final technology demonstrators including speed-up and scale-up measures of selected demonstrators are described in the final Deliverable D61(2).

1.2 Methodology

We developed our DataMiningGrid system by following standard software development process. The first step in this process is requirements management, which is a systematic approach to finding, documenting, and managing requirements. The goal of the requirements management is to describe what the system should do and allows the developers and the customer to agree on that description. This stage is extremely important since bad requirements management is one of the main factors contributing to the fact that many projects end up missing user needs, are late or over budget.

In requirements management stage we elicited, organized, and documented the required functionality and constraints; track and document tradeoffs and decisions. In this stage actors were identified, representing the users, and any other systems that may interact with the system being developed. Use cases were identified, representing the behavior of the system. Each use case (and the respective demonstrators) is described in detail in Deliverable D61(1) (and D61(2)). The notions of use case and scenarios proscribed in the process is an excellent way to capture functional requirements and to ensure that these drive the design, implementation and testing of software, making it more likely that the final DataMiningGrid interfaces, services and tools fulfill the end user needs. Managing use cases along with all other requirements is key to understanding the state of the project and better enables us to deliver the right technology.

The use-case description shows how the system interacts step by step with the actors and what the system does (this is part of Deliverable D61(1)). By providing a user's view of what the system should do, use cases provide functional requirements, which are subject to the Deliverable D11(2). Special requirements section of a use case, stores all non-functional requirements specifically related to a particular use case. Non-functional requirements about the DataMiningGrid framework as a whole are stored separately as supplementary requirements specification.

1.3 Evaluation

In the end of the project, once we had developed the DataMiningGrid system and services and run seven demonstrator applications in the DataMiningGrid test bed, it was relatively easy to evaluate the developed system and to see to which extent did we satisfy the functional and non-functional requirements.

Following the definition of each requirement we provide an evaluation statement 'Level of achievement'.

2 Requirements

The use cases motivate a number of technical requirements for the development of DataMiningGrid tools and services. The DataMiningGrid Consortium currently feels that the **technical requirements** described below are **essential** to the DataMiningGrid© technology. Each requirement includes a short description and is motivated by one or more use case(s) from the previous section.

Functional requirements lists the set of operations that the DataMiningGrid services and tools must be able to perform in order for the services and tools to be considered functional. **Non-functional requirements** are additional criteria that generally focus on the system properties, such as environmental and implementation constraints, performance, platform dependencies, maintainability, extensibility, reliability etc. of the services and tools. Non-functional requirements are further divided into *special* (non-functional) requirements and *supplementary* (non-functional) requirements. The special requirements are related to the specific use cases and need to be handled in subsequent system models, such as the analysis, design and implementation models. The supplementary requirements are generic requirements and cannot be connected to a particular use case or a particular real world problem. They should instead be managed separately in a list of supplementary requirements.

The main classes of functionality required by users were identified as:

- Identifying (locating) DataMiningGrid resources by using metadata;
- Accessing and selecting subsets of data;
- Data transfer,
- Data (pre-) processing;
- Data mining tasks;
- Text mining and ontology learning;
- Workflow editing and submission;
- Data privacy, security and governance;
- Integration of domain knowledge;
- Grid infrastructure and middleware functionality; and
- Usability, response times and user-friendliness.

2.1 Identifying (locating) DataMiningGrid resources by using metadata

Functional requirement 1: Provide mechanism for accessing metadata

Rationale: For the client to make decisions about how to handle the data provided by the data service, it is necessary for the client to access the metadata associated with the data. There are different forms of metadata. Application metadata describes properties of primary data that help the user to interpret it.

Technical metadata describes how the primary data is stored in physical resources. So in a biological example, the application metadata might describe biological variables like proteins, genes, metabolic networks, and so on, while the technical metadata would describe the hardware and software used to store this e.g. a Postgres relational database version 7.3 on a 2.2 GHz PC running windows XP.

Initial Assessment: Application metadata would be most useful if provided the early stages of workflow design. This would suggest that application metadata should be part of the information supplied by the data service registry.

Technical metadata would mainly be used internally by the data services to construct data access and mediation workflows.

Motivation: Text mining use cases especially the DC use case for Subtask T61.3: Finding related and similar documents in the intranet. Document repositories on the intranet contain metadata describing the kind, format, up-to-datedness and accessibility of their data.

Level of achievement: This was partially achieved by using the MDS4 service in connection to the Information Integrator unit. From end users perspective, the Provenance unit was developed, which addresses the ‘meta-data’ needs of all demonstrator applications.

Functional requirement 2: Provide mechanism for locating data services

Rationale: Data services are distributed all around the world. Locating all the required data services using ad-hoc search techniques is unlikely to be successful. By providing an automatic mechanism for locating data services, appropriate data services will be found quickly and easily.

Initial Assessment: This requires a data discovery service. Data discovery services identify data items by matching specified characteristics, attributes, or metadata such as:

- A description, summary, or overview of the data source,
- Data provenance: how and where the data was generated i.e. what scientific instruments were used to collect the data; or what processing has been applied to the data; who is the creator, the owner, or the last modifier of the data,
- Physical metadata such as the data size, access control policies, transfer rates, error handling, number of disks, and the number of heads,
- It is also important to identify those data sources that are also available as computational resources so that data can be analyzed at its source.

Motivation: text mining use cases especially the DC use case for Subtask T61.3: Finding related and similar documents in the intranet. Identification and location of document servers/repositories on the intranet which can or have to be searched.

Level of achievement: This was fully achieved by developing and using the Data Resources group of units. See Deliverable D32(2) for more details.

2.2 Requirements concerning data privacy, security and governance

Functional requirement 3: Authenticate request and provide appropriate feedback.

Rationale: The security of access to data and other resources may be critical for some applications. It will be necessary to authenticate requests and provide appropriate feedback. Not all services are public. For those that are not it is necessary to confirm client identity and to check client access rights. The actions that the client takes next will depend on knowing if the authentication was successful or not, so the client requires appropriate feedback.

Initial Assessment*:* Authentication can be provided via a simple username/password combination or more elaborate techniques can be employed. On the other hand, for public access sites, authentication may not be needed at all.

Many individual datasets may be protected with different authentication techniques. Any service that provides access to these datasets may not only have to deal with the heterogeneous nature of data itself but also with heterogeneous authentication and access procedures.

There may also be restrictions on who can have access to which datasets or datasets may need to be anonymised (substituting untraceable codes in place of personal details) in order to ensure that personal information is not revealed.

Motivation:

- These use cases are the text mining from DC and FHG, general data access cases from UU, and system diagnostics from TECH,
- DC text mining use cases: Data and document repositories are distributed over different business units and departments. This implies different access policies on the distributed data. Therefore authentication must not only take place for the provided grid services in general, but the grid service itself must do authentication on a per server/repository basis.

Level of achievement: This was fully achieved by using existing GT4 primitives and implementing them in the end users interface, e.g. the CredentialsGenerator Unit (see Deliverable D32(2) for more details).

Nonfunctional requirement 1 (supplementary): Useable from firewall-protected environments.

Rationale: To protect systems from unauthorized access, many networks make use of a firewall. This restricts external access to particular ports. Ideally the data services and tools will not be affected by the restrictions set by a typical firewall.

Initial Assessment: Most firewalls will allow the HTTP protocol on port 8080 This is sufficient to allow Web access but not allow unauthorized access beyond the confines of public directories. Thus any protocol that makes use of port 8080 is likely to be accepted by most firewalls.

Motivation:

- DC text mining use cases: Data and document repositories at DC are distributed over different business units and company departments. The DC intranet is not only protected against the internet by a firewall, but different units within the company intranet are also protected against each others by firewalls,
- Data mining distributed medical databases use case,
- Ecological use case.

Level of achievement: This was fully achieved by providing a mechanism to identify known IPs and include them in the firewall list of the grid servers. This mechanism is documented in Deliverable D63 (the part which relates to grid administrators).

Nonfunctional requirement 2 (supplementary): Use secure, encrypted channels.

Rationale: It may be necessary to pass sensitive information across the Internet. In such circumstances the information should be encrypted to ensure that sensitive information remains secure.

Initial Assessment: The inclusion of secure socket layer (SSL) software will encrypt data traffic to and from the server.

Motivation:

- Text mining use cases from DC,
- Data mining distributed medical databases use case.

Level of achievement: This was fully achieved by using existing GT4 primitives and implementing them in the end users interface, e.g. the CredentialsGenerator Unit (see Deliverable D32(2) for more details), and the use of WS-Security and WS-SecureConversation, standards implemented by GT4.

2.3 Accessing and selecting data

Functional requirement 4: Initiate data service in response to successful request

Rationale: Assuming that a data service will be connection-oriented a session is required that will maintain the current state of the data service.

Initial Assessment: Data services can also be connectionless, like Web servers. In this case only one instance of the data service is needed, but each request must be self-contained and include authentication details where necessary. This means that a single request has the potential to be very complicated. On the other hand, a connection-oriented data service will maintain a current state, like FTP. Each individual request is simpler because its context does not have to be restated every time it is passed to the data service.

Motivation: All use cases.

Level of achievement: This was fully achieved by developing and using DataMiningGrid units from the group of Data Resources (see Deliverable D32(2) for more details).

Functional requirement 5: Data service provides access and mediation services to one or more datasets

Rationale: A data service may provide access to one or more datasets. The client application requires a consistent approach to access and handle the data provided by the data service. A mediation service will provide an integrated view of distributed data to the client application.

Initial Assessment: Data sources need to have structure in order for them to be recognized or understood. This structure should be described by metadata. Data sources are usually file orientated or database orientated. Sources of datasets include:

- File systems e.g. NFS or AFS as a flat file or collection of flat files such as ASCII text files that have been formatted according to some clear convention or documents from which data can be extracted and then presented in a structured manner,
- Structured or semi-structured XML data,
- Database management systems: particularly relational (e.g. ORACLE or MySQL),
- Abstract or virtual data that has been derived from a subset of one or more data sets,
- Directories of grid services i.e. data that is used to describe available resources, to support the operation of the grid, and to describe its state and configuration.

Motivation: Most, if not all, use cases will require access to data for data mining.

Level of achievement: This was fully achieved by developing and using DataMiningGrid units from the group of Data Resources (see Deliverable D32(2) for more details).

Functional requirement 6: Data service processes basic queries from client

Rationale: The data service must accept queries that allow the client to select data and perform other basic operations.

Initial Assessment: Users want to access both data and metadata from the located source(s) and select a data subset for processing or mining. For the client to make decisions about how to handle the data provided by the data service, it is necessary for the client to access the metadata associated with the data. The way in which data is structured will affect the manner in which it can be accessed:

- Data access can involve both reading data from, and writing data to, a data source. These operations will be implemented in different ways for different resources,
- Data access may be needed to update data sources, and to ensure the consistency of all data replicas,
- Data sources have different access mechanisms:
  - File system based commands like open, close, read, write. Data subsets can be selected using file processing scripts,
  - Relational query mechanisms like SQL, XQUERY, XPATH,
  - Mechanisms for hierarchical storage systems.

Diverse sources will result in a diversity of access mechanisms. To enable robust generic access mechanisms we will want:

- Uniform methods to access a data source e.g. use GridFTP, although it is also good for multiple mechanisms to exist,
- To be able to transform data amongst the different data types that are stored in the different data sets,
- To be able to mediate between different data models and database schemas,
- Federated or virtual databases may be required in order to integrate various data sources into a transparent global schema. Individual data sources may still need to be available for direct access,
- Specific views of a data source may need to be provided.

It is often desirable to be able to select a subset of the data accessed from a data source before it is transferred or stored elsewhere. Data selection may be an interactive operation and require sophisticated tools to guide the user:

- At the simplest level a data subset can be selected using relational database query mechanisms,
- At the most complex level there are many different ways of filtering, combining, or processing data, and these methods can require sophisticated code to be executed.

Motivation: All use cases.

Level of achievement: This was fully achieved for the needed demonstrators (e.g. the medical demonstrator) by developing and using DataMiningGrid units from the group of Data Resources (see Deliverable D32(2) for more details).

2.4 Data transfer

Functional requirement 7: Data service must be able to transfer datasets from one server to another

Rationale: By its very nature, grids are geographically distributed. Therefore, in order to access the resources available at different locations, it must be possible to move datasets from one geographical location to another.

Initial Assessment: Moving data requires no knowledge of the data structures. The basic unit of data transfer and access is:

- Primitives: floats, integers, characters and arrays, images or objects,
- Files: these are uninterrupted sequences of bytes.

GridFTP is the fundamental data access and transport mechanism and it provides a uniform interface to different file storage systems. When moving data it is necessary to consider:

- Maximizing file sharing, minimize replication and to monitor shared file space i.e. perform garbage collection,
- Dealing with failure and restarting transfers,
- Synchronizing updates to all replicas,
- Allowing different types of file transfer:
  - o The simultaneous transfer of multiple files with each job individually monitored and managed,
  - o Reliable data transfer i.e. GridFTP with some enhanced features.

Motivation: All use cases.

Level of achievement: This was fully achieved by developing and using DataMiningGrid units from the group of Data Resources (see Deliverable D32(2) for more details) as well as using the existing services such as GridFTP. In order to transfer directories with data, the Consortium developed adapters than zip the whole directory structure and unzip everything at the destination.

Functional requirement 8: Data service must be able to transfer at least 20 000 files in a row without user interaction

Rationale: In the area of text mining raw data is typically stored in flat files. The number of files containing training data, test data or new data for classification may range from only one to tens of thousands (e.g. data from news agencies). These files usually reside inside a few, if not a single, directory in a file system. As it is unfeasible for users to specify each individual file, data services must be capable of transferring the whole contents of these user specified directories without any additional user interaction.

Initial Assessment: GridFTP is the fundamental data access and transport mechanism and it provides a uniform interface to different file storage systems.

Motivation: Text mining use cases.

Level of achievement: This was fully achieved by developing and using DataMiningGrid units from the group of Data Resources (see Deliverable D32(2) for more details) as well as using the existing services such as GridFTP. In order to transfer directories with data, the Consortium developed adapters than zip the whole directory structure and unzip everything at the destination.

Functional requirement 9: Data service must be able to transfer multiple files of at least 500MB in a row

Rationale: In the area of text mining raw data is typically stored in flat files. The number and sizes of files containing training data, test data or new data for classification may range from a great number of files of only a few kilobytes each to very few (i.e. <10) or only one file of hundreds of megabytes.

Initial Assessment: GridFTP is the fundamental data access and transport mechanism and it provides a uniform interface to different file storage systems.

Motivation: Text mining use cases.

Level of achievement: This was fully achieved by developing and using DataMiningGrid units from the group of Data Resources (see Deliverable D32(2) for more details) as well as using the existing services such as GridFTP. In order to transfer directories with data, the Consortium developed adapters than zip the whole directory structure and unzip everything at the destination.

2.5 Data pre-processing

Functional requirement 10: Include additional functions such as data cleaning operations and data transformation operations

Rationale: Operations on large amounts of data should be kept as close to the data as possible in order to reduce data transfer overheads. Ideally such operations would be integrated with the data services and tools.

Initial Assessment: Additional grid data service functions may execute specific algorithms on the data, such as imputing in missing values or transforming the data in different ways (discretization, normalizatoin, summarization, etc). This can be done more efficiently by the data service since fewer data transfers are required.

At the extreme level, all the data mining tasks could be performed by the grid data service. Programs (probably either scripts or Java programs) could be uploaded to the grid data service and used to perform tasks on the data.

Motivation:

- Genetic algorithms for gene regulatory reengineering use case,
- Molecular simulations of protein unfolding use case,
- Text mining use cases,
- Data mining distributed medical databases use case.

Level of achievement: This was fully achieved by grid-enabling existing applications that are used for pre-processing. For this purpose the Consortium developed the generic Data Mining Application Enabler applications. So far, there are more than 20 grid-enabled applications that are ready to execute in the grid environment.

Functional requirement 11: Data processing taking place near where data is located

Rationale: For large amounts of data, there may be high overheads of transferring the data from where it is stored to where it is processed. There might also be cases, where the right to move or distribute the data to other servers may be restricted due to data privacy or copyright reasons. Where possible, as much processing should take place as near to where the data is located as possible in order to keep data transfer times to a minimum and to avoid inconvenience to other users due to the increased traffic caused by data transfers. Other, non-technical (i.e., legal, ethical) reasons may constrain the transfer/copy of data from one physical location to another.

Initial Assessment: Processing data close to its source may be important in order to:

- Scale computation,
- Reduce the dataset's size before it is transferred,
- Realize virtual datasets i.e. a new dataset that has been derived in some way from one or more data sources. These results could then be accumulated in other data collections: this is similar to virtual data warehousing,
- Format the dataset (into XML for example) before it is transferred,
- Access and process data which can not be redistributed or transferred,
- Process highly dynamic data, where data changes faster, than transferring the data to another place would take.

Motivation:

- Text mining use cases: Text data should be converted in place from their original document format (different binary and proprietary file formats like MS Word, PowerPoint, PDF, …) to a data format suitable for performing text mining and retrieval (e.g. ASCII, XML, vector-representation, …). If the conversion takes place on the data repository, the converted data can be cached for future queries and analyses,
- Genetic algorithms for the gene-regulatory network re-engineering use case,
- Molecular simulation of protein unfolding use case.

Level of achievement: In the current implementation of the PrarameterControl Unit (from the end user Workflow Editor and Manager) it is possible to specify exactly on which machines the execution of the application should take place. In this way, the end user can completely control where the execution is taking place.

Functional requirement 12: Storing of intermediate results from preprocessing tasks for further usage by other algorithms

Rationale: In data mining typically many preprocessing tasks have to be performed before the actual data mining/analysis algorithm can be applied. These preprocessing tasks each modify the original data set and provide the intermediate results as input for the next task. While not all of these intermediate results need to be stored permanently, some of them are potentially reusable either at later stages of the process (e.g. during a classification phase) or when repeating the whole process with different parameters (e.g. of the data mining algorithm). The reuse of these results would then lead to a significant reduction of computing/processing time as the intermediate results to not have to be recomputed.

Initial Assessment: For this purpose these intermediate results need to stored somewhere in a grid environment and be made available users. Furthermore, to find the intermediate results resulting from pre-processing tasks performed to train an individual model, these results need to be mapped to this model in some way.

Motivation: Text mining use cases, ecological use case

Level of achievement: The DataMiningGrid system is extremely flexible and it allows the execution of simple as well as complex DataMiningGrid applications. This means that it is possible to feed the results from one execution to become an input for next execution. In fact some of the text mining demonstrators (see Deliverable D61(2) for more details) use complex workflows with 4 such synchronization steps.

2.6 Annotation of Data mining models with provenance information

Functional requirement 13: Provide mechanism for permanently recording provenance information about intermediate results in certain stages of a data mining process

Rationale: It is not enough to only store intermediate results on some node in a grid, since these results need to be found by potentially different users in later stages. Therefore, these intermediate results need to be annotated in a way that allows users to search for them and to receive their storage location. Minimal provenance information that needs to be recorded permanently in order to enable users to find these results even after a longer period of time includes the following:

- Date of creation,
- Name of user who created it,
- Name and version of algorithm/program that created it.

Initial Assessment: Annotation of all sorts of intermediate results in a generic way that covers all aspects (e.g. how they were generated, information about the original data used, etc.) is very difficult and cumbersome to implement due to the lack of a universal standard for this purpose. However, if only limited provenance information is stored (e.g. in a central repository) basic data services can be applied for this task.

Motivation: Text mining use case, ecological use case

Level of achievement: This was fully achieved by using the MDS4 service and the generic Data Mining Application Description schema in connection to the Information Integrator unit. From end users perspective, the Provenance unit was developed, which satisfies the needs of all demonstrator applications (both the ecological demonstrator and the text-mining demonstrators).

Functional requirement 14: Provide mechanism for locating intermediate results based on provenance information stored with the results

Rationale: In principal two different ways of using intermediate results exist. Intermediate results can serve as input to algorithms that are executed immediately after these results have been generated. In this case the location of these results can be passed to the algorithm and no search operation for them has to be performed. However, in some cases intermediate results will also often be stored somewhere in a grid and not re-used immediately. In these cases the user has to specify in the workflow which intermediate results serve as input for certain algorithms he intends to use. As the actual storage location of these results in a grid may be unknown, mechanism are needed that enable the user to receive the location of an individual set of intermediate results base on the provenance information stored together with these results as specified above.

Initial Assessment: Basic data services for querying information repositories such as database and file systems can be used to receive the location of particular intermediate results. However, the user searching for this location needs to know the restrictions on the parameters he can search for that describe these results.

Motivation: Text mining use case, ecological use case

Level of achievement: This was fully achieved by using the MDS4 service and the generic Data Mining Application Description schema in connection to the Information Integrator unit. From end users perspective, the Provenance unit was developed, which satisfies the needs of all demonstrator applications (both the ecological demonstrator and the text-mining demonstrators).

2.7 Data mining tasks

Functional Requirement 15: Decoupling WEKA components

Rationale: Decoupling Weka components (data mining services, etc.) for optimal extensibility and platform/language independence

Initial Assessment: Weka [Weka04] is a widely known and accepted open source data mining tool. It contains more than 100 algorithms for data preprocessing, data mining, validation, and visualization of the results. Weka's modular structure and uniform application programming interface facilitates the integration of many of its algorithms into grid systems.

Motivation: Weka's components can be used to build one integrated system encompassing demonstrators from several (or all) Partners. Furthermore, with Weka being able to execute on the grid users can choose between many more additional analysis services.

Level of achievement: This was fully achieved for the purpose of the demonstrator applications, and Weka was grid-enabled by using the generic Data Mining Application Enabler.

2.8 Text mining and ontology learning

All text-mining demonstrators were realized on decentralized document sources and the processing of the documents will also be realized decentralized in a grid environment. The requirements relating to the 'text mining' and 'ontology learning' use cases were already covered in sections 3.2 to 3.4 (accessing distributed data services, data preprocessing and transfer).

2.9 Workflow editing and submission

Functional requirement 16: Ability to prepare workflows by graphical editor and submit workflows for execution

Rationale: Data mining is a complex process consisting of many tasks for pre-processing, analysis, and visualization. In order to compile these tasks for automatic execution, a graphical workflow editor providing such capability is needed. Also such a visual tool will reduce error rates compared to traditional batch scripting and shield users from the complexity of the underlying system.

Furthermore, as the workflow editor will not execute the workflow, it has to be able to submit it to some machine where it is executed.

Initial Assessment: In order to contribute to the system's usability for end users, the workflow editor has to support various demonstrators while hiding as many grid specific aspects as possible from end users. This demands for the editor to provide a generic interface for adding new services during runtime by using the Data & Analysis Discovery/Location services as explained in functional requirements #1 and #2 to present available services to the user.

The workflow editor will also perform basic validation of the workflow chain ensuring syntactical soundness.

These functions require the computational and data services to provide extensive descriptions about themselves. It has to be carefully evaluated which information is needed in order to keep data exchange and overall complexity to a minimum while still ensuring full functionality.

Execution of the workflow is not part of the editor. Therefore, it is necessary to submit the workflow to some sort of manager, which starts the execution and monitors the progress.

Motivation: The basic idea is to provide an extensible workflow editor, which can be used in many demonstrators.

Level of achievement: This was fully achieved by employing Triana and providing a variety of units that can be used to deal with various aspects of workflow development (see D32(2) for more details).

2.10 Requirements for integrating domain knowledge

Functional requirement 17: Ability to store models in a knowledge base

Rationale: The data to be mined in complex problem solving scenarios will increasingly require the integration of existing domain knowledge into the mining process, particularly in knowledge-intensive domains. This knowledge is often dispersed across geographically distributed sites and organizations, either in digital form (ontologies, metadata, knowledge-based or simulation systems) or provided interactively by human experts. Grid-enabled data mining technology will need to provide services and tools to support knowledge-aided data mining.

Initial assessment: Storing formulae in a knowledge database could be highly beneficial for some machine learning algorithms, e.g. Lagramge.

Motivation: Ecological modelling use case.

Level of achievement: This was partially achieved as the 'knowledge base' is currently represented by files in distributed file systems.

2.11 Grid infrastructure and middleware requirements

Functional requirement 18: Middleware able to execute generated workflows including conditional control flows

Rationale: Data mining is complex process including many sub-tasks. As some may only be executed if specific conditions are fulfilled and others may be repeatedly executed a specific number of times, the workflow editor has to provide conditional control flows.

Initial Assessment: Conditional control flows are essential to ensure usefulness of the workflow editor, since the workflows of many use cases include branches and loops according to certain conditions. Without such control flows the workflow editor would only be of limited or no use for these use cases.

Motivation:

- Text mining use cases: Distributed document clustering and training of classifiers takes place as iterative process, being repeated until a certain quality criteria is met; and
- Ecological use case.

Level of achievement: This was fully achieved by developing the Resource Broker service that works in connection to the Execution Unit at the end users side.

Nonfunctional requirement 3 (supplementary): The DataMiningGrid services and tools should be compatible with existing grid infrastructure.

Rationale: Use of existing grid infrastructure will reduce development and evaluation complexity. However this will only be possible if the existing grid infrastructure is capable of supporting the required services and tools.

Initial Assessment: Where possible, existing protocols (grid interoperability profile WS-I+) and infrastructure should be used. This may mean compromising on some (functional) requirements of the project, but will enhance compatibility and reduce the risk of project failure.

Motivation: All use cases.

Level of achievement: This was fully achieved services may combine well with other test beds, e.g., InteliGrid test bed.

Nonfunctional requirement 4 (supplementary): Grid infrastructure, test-bed for the DataMiningGrid project.

Rationale: Grid infrastructure should be developed by the project Partners for technology testing purposes.

Initial Assessment: Where possible, existing protocols and infrastructure should be used. This may mean compromising on some aspects of the project but will enhance compatibility and reduce the risk of project failure.

Motivation: All use cases.

Level of achievement: This was fully achieved. The partners developed a test bed which is fully operational for almost a year. See Deliverable D63 for details of how to establish or join such a test bed.

Functional Requirement 19: Provide monitoring services.

Rationale: Constant system diagnosis is of critical importance in complex, distributed systems. In such systems it is extremely difficult to track and monitor job execution, and to understand what causes failures and poor performance. An effective ongoing analysis of the system's condition is necessary in order to enable corrective actions, and improve system performance.

Initial Assessment: Ongoing analysis of the system's condition enables early detection of the occurring problems, detects points of failure, and exposes weaknesses that potentially or actually harm the system's performance. The corrective actions and the performance improvement based on this analysis need to be automated to the maximal extent catering for the dynamic and often unpredictable modifications and alterations occurring in the system. The user needs to be notified regarding the status of her job submissions, and provided with the sufficient information to carry out corrective actions, if these are within her capacity as a user (for example, job resubmission with the complete and correct data). The system administrator needs to be updated regarding the reoccurring problems and potential weaknesses, and provided with the sufficient information to reconfigure the system if necessary.

Motivation:

- Grid monitoring use cases;
- Text mining use cases; and
- Ecological modeling use case.

Level of achievement: This was fully achieved by providing a GridMonitor unit as part of the end user GUI.

2.12 Usability, response times and user-friendliness

Nonfunctional requirement 5 (supplementary): Data services and tools response time is within a reasonable time period.

Rationale: The user will wish to know if an operation has succeeded or failed within a reasonable time period.

Initial Assessment: Response time when client-application connects to a service should be within seconds (10 seconds is realistic response time for job submission confirmation. However, resource allocating facilities of the current distributed systems operate in minutes, rather then seconds. For example, EDG Resource Broker takes on average 5 minutes to execute a simple echo job). Response time when client-application sends a query should also be within seconds in order to inform the user if the query can be processed or not (30 seconds is suggested as a reasonable time). For the DC Text-Mining Use case 'Find related documents' even shorter response time guarantees seem reasonable. Otherwise, system should give the user regular progress reports.

Time to transfer a large amount of data will depend on the amount of data to be transferred (no limit can be placed on this).

Motivation: All use cases.

Level of achievement: This was fully achieved, and for many demonstrator applications these response times were documented (in D61(2)).

3 Objectives – Research Goals Beyond the Scope of the Project

In addition to the set of features that addressed by the DataMiningGrid technology (as defined by requirements in the previous section), there are additional features that may be useful for many use cases. These requirements can be classified as 'nice-to-have' requirements. These useful, but non-essential, requirements were also addressed by the Consortium to various level of extent.

Functional requirement 20: Provide mechanism for locating computational services

Rationale: Similar to the data services, the analysis services are also distributed all over the grid. Since ad hoc search techniques are inadequate to locate such distributed services, an automatic mechanism for locating them is needed in order to find them in a quick, easy, and convenient way.

Initial Assessment: This requires an analysis discovery service. Similar to data discovery services, analysis discovery services identify analysis services by matching specified characteristics, attributes, or meta-data such as:

- A description, summary, or overview of the analysis service,
- A description of the input and output data,
- The version of the service,
- Who is the creator, the owner, or the last modifier.

Motivation: All the use cases which will need to locate appropriate computational service(-s) will benefit from automatic service discovery.

Nonfunctional requirement 6 (supplementary): Data services and tools must be scalable

Rationale: Scalable means that the processing larger datasets should be possible without causing the system to grind to a halt.

Initial Assessment: Where possible, data should be divided into subsets and sent to distributed processes or processors. The data can then be processed in parallel and, in theory, should not take significantly longer than the time it would take one data subset to be processed on by one process. In practice there will be additional overheads due to scheduling of processes and distribution of datasets. Furthermore there may be other bottlenecks, for example if all the data must pass through a single gateway.

Where data is already distributed, it may be possible to avoid bottlenecks by transferring the data directly to the distributed process without passing through the client or a specific service node. However, this raises some technical problems in how to clean the data and prevent repetitions in the data.

Motivation: All use cases.

Nonfunctional requirement 7 (supplementary): DataMiningGrid services and tools must be transparent.

Rationale: The end user should not need to know about or understand the low-level details of how the technology works. However, users should always be aware of the possible intricacies of distributed systems, like latency, availability and reliability (see [Spol00] and [Spol02]).

Initial Assessment: It is assumed that end users will be conversant with how to set up data mining workflows. However, the end user will not necessary be conversant with the structure of the grid technology that underlies the data services and tools. Data services should be presented to the user as if it were a file or any other source of data.

Motivation: All use cases.

Nonfunctional requirement 8 (supplementary): Extensible and 'future-proof'

Rationale: The design of the DataMiningGrid services and tools should be flexible enough to allow additional functionality to be included without inhibiting backward compatibility.

Initial Assessment: The key to making a system 'future-proof' is simplicity. For example, a simple set of functions and operations that can be combined to perform any task are better than a smaller set of functions that require more arguments and are more likely to require change in the future. If necessary, new functions can be introduced but these should follow the same principal.

Motivation: All use cases.

Nonfunctional requirement 9 (supplementary): Grid infrastructure allows interoperability between heterogeneous programming environments and different operating systems

Rationale: The existing data mining components of the project Partners run on different operating systems (Linux and Windows) and are written in different and incompatible programming language. To reduce development time it must be possible to adopt them to the grid with out having to port them to another programming environment or operating system.

Initial Assessment: Grid interoperability standards should guarantee this requirement. It may be necessary to write glue/bridge code in some cases.

Motivation: All use cases.

Nonfunctional requirement 10 (supplementary): Data service interactions should be kept simple.

Rationale**:** A complex protocol is more difficult to implement and more likely to fail. The additional functionality provided by a more complex protocol can often be achieved by using several simpler commands.

Initial Assessment: A simple text-based language that is both simple and extensible. Every command has a response in order to confirm the command has been successfully processed.

Motivation: All use cases.

4 Conclusions and Future Work

While investigating the generic data mining requirements of emerging grid-enabled problem solving environments based on a selected set of representative application sectors and problem domains, the DataMiningGrid© Consortium successfully developed the DataMiningGrid system facilitating grid-based data mining i.e. custom grid applications.

The Deliverable D11(3) was prepared to evaluate the level of achievement of the most important requirements for data mining in grid computing environments.

The DataMiningGrid system facilitates:

- a dynamic and secure way of accessing, retrieving, and manipulating (join, subset selection, filtering, etc) data sets from heterogeneous and distributed sources;
- a dynamic and secure association of these data sets to operations provided by data mining servers available in a distributed computing environment, i.e. the grid;
- a dynamic and secure execution of these operations on the data sets,
- a dynamic and secure pipelining of such operations and the resulting intermediate data sets;
- a dynamic and secure allocation and addition of new data mining services and operations (servers), new databases and data sets to the grid; and
- a highly interactive, intuitive, and secure way for users to define, execute, monitor, and manage complex data mining workflows in such a distributed data mining environment.

Along with these developments the DataMiningGrid© Consortium demonstrated the deployment of the developed Web-based, grid-enabled, data mining applications, modeling tools and services in a carefully selected sets of representative application sectors (see Figure 1). The selected applications and technologies include: re-engineering of gene regulatory networks via distributed genetic programming, analysis of biological databases for gene/protein annotation, data mining based monitoring of grid systems (including analysis of recurring failure and descriptive analysis), distributed text classification and ontology learning for customer relationship management and quality management, finding related and similar documents in the intranet, mining of digital (scientific) libraries, analysis of distributed medical databases for endemic goitre and iodine deficiency studies, distribution of algorithms for mining of data in the context of sanitary engineering, and mining of literature databases (including equation discovery, data check/filtering).

5 References

[Avaki04] AVAKI at http://www.avaki.com/

[Bax02] Baxter R, A Complete History of the Grid at www.nesc.ac.uk/talks/sdmiv/SDMIV-25Oct2002- Baxter.pdf

[Condor04] The Condor Project at http://www.cs.wisc.edu/condor/

[Condor-G04] http://www.cs.wisc.edu/condor/condorg/

[Czaj04] K. Czajkowski, D. Ferguson, I. Foster, J. Frey, S. Graham, T. Maguire, D. David Snelling and S. Tuecke, 'From Open Grid Services Infrastructure to WSResource Framework: Refactoring & Evolution,' http://www-fp.globus.org/wsrf/OGSI%20to%20WSRF%201.0.pdf , Tech. Rep. 1.0, 12/2/2004, 2004.

[DataGrid04] The DataGrid Project at http://eu-datagrid.Web.cern.ch/eu-datagrid/

[Dee90] Deerwester, S. S. & Dumais, T. K.Landauer, G.W. Furnas, and R.~A. Harshman (1990). Indexing by latent semantic analysis, in: Journal of the American Society of Information Science, 41(6):391--407, 1990.

[DQP04] www.ogsadai.org.uk/dqp/

[Dub01] Dubitzky, W., Krebs, O., and Eils, R. (2001), 'Minding, OLAPing, and Mining Biological Data: Towards a Data Warehousing Concept in Biology', Proc. Network Tools and Applications in Biology (NETTAB), CORBA and XML: Towards a Bioinformatics Integrated Network Environment, Genoa, Italy, pp78-82.

[ELDAS04] www.edikt.org/eldas

[Fer03] Ferreira, L., Jacob, B., Slevin, S., Brown, M., Sundararajan, S., Lepesant, J., Bank, J. (2003) Globus Toolkit 3.0 Quick Start http://www.redbooks.ibm.com/redpapers/pdfs/redp3697.pdf

[Fos02] I. Foster, C. Kesselman, J.M. Nick, S. Tuecke, The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, at www.gridforum.org/ogsi-wg/drafts/ogsa_draft 2.9_2002-06-22.pdf

[Fos04] I. Foster, J. Frey, S. Graham, S. Tuecke, K. Czajkowski, D. Ferguson, F. Leymann, M. Nally, T. Storey, W. Vambenepe and S. Weerawarana, 'Modeling Stateful Resources with Web Services,' http://www-106.ibm.com/developerworks/library/ws-resource/ws-modelingresources.pdf , Tech. Rep. 1.0, 01/20/2004, 2004.

[Fos99] I. Foster, C. Kesselman, The GRID, Morgan Kaufmann Publishers, Inc., San Francisco, 1999.

[GGF04] The Global Grid Forum Database Access and Integration Services Working Group at http://www.gridforum.org/6_DATA/dais.htm

[Globus04]The Globus Alliance at http://www.globus.org/

[Gra97] Ingo Graf and Ulrich Kreßel and Jürgen Franke, Polynomial Classifiers and Support Vector Machines, 1997, 00 -- 00, Inter. Conf. on Artificial Neural Networks, Zürich

[Gru93] Gruber, T. R. (1993): Toward principles for the design of ontologies used for knowledge sharing, in: Formal Analysis in Conceptual Analysis and Knowledge Representation, Kluwer 1993.

[Hof01] Hofmann, Thomas, Unsupervised learning by probabilistic latent semantic analysis, in: Machine Learning,42:177--196, 2001.

[Joa98] Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features, in: Proceedings of the Tenth European Conference on Machine Learning (ECML '98), Lecture Notes in Computer Science, Number 1398

[Lan97] Landauer, Thomas K. & Dumais, Susan T. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge Psychological Review 104 (2): 211--240, 1997.

[Leo02] Edda Leopold & Jörg Kindermann (2002): Text Categorization with Support Vector Machines. How to Represent Texts in Input Space?; in: Machine Learning 46, pp. 423 - 444.

[Man99] Manning, C. D. & Schütze, H. (1999): Foundations of Statistical Natural Language Processing, MIT Press: Cambridge MA, London.

[OGSA-DAI04] www.ogsadai.org.uk/docs/current/doc/DAIOverview.html

[Park03] Park, K., and Kanehisa, M. (2003), 'Prediction of protein subcellular locations by support vector machines using compositions of amino acids and amino acid pairs', Bioinformatics, Vol. 19, pp. 1656-1663.

[Rom02] M. Romberg, 'The UNICORE Grid Infrastructure', Scientific Programming Special Issue on Grid Computing, 2002, 10, pp. 149-158.

[Sal83] Salton, G.& McGill, M. J. (1983) Introduction to Modern Information Retrieval, McGraw Hill, New York.

[Spol00] Spolsky Joel; Three Wrong Ideas From Computer Science, August 22, 2000; www.joelonsoftware.com/articles/fog0000000041.html

[Spol02] Spolsky, Joel; The Law of Leaky Abstractions, November 11, 2002; http://www.joelonsoftware.com/articles/LeakyAbstractions.html

[Stan04] Stankovski V., May M., Franke J., Schuster A, McCourt D., Dubitzky W., A Service-Centric Perspective for Data Mining in Complex Problem Solving Environments, H.R. Arabnia and J. Ni (eds) Proc of Int'l Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'04), Vol II, pp. 780-787, 2004.

[Stork04] www.cs.wisc.edu/condor/stork/papers/stork-icdcs2004.pdf

[Stur03] Sturgeon, B., McCourt, D., Cowper, J., Palmer, F., McClean, S., and Dubitzky, W. (2003), 'Can the Grid Help to Solve the Data Integration Problems in Molecular Biology?', 3rd International Symposium on Cluster Computing and the Grid (CCGRID 2003) , Tokyo, Japan, pp. 594-600.

[Tom99] M. Tomassi. Parallel and distributed evolutionary algorithms: A review. In K. Miettinen, M. Makela, P. Neittaanmaki, and J. Periaux, editors, Evolutionary Algorithms in Engineering and Computer Science, pages 113-133. J. Wiley and Sons, Chichester, 1999.

[Weka04] – Weka Homepage; http://www.cs.waikato.ac.nz/ml/weka/

[Weka99] – Witten, Ian.H, Frank, Eibe; Data Mining Practical Machine Learning Tools with Java Implementations; Morgan Kaufmann; 1999

More relevant bibliographical material and abbreviations/acronyms can be found in the Project Manual.