Ann. Telecommun.
DOI 10.1007/s12243-010-0195-x

UNICORE 6 — Recent and Future Advancements

Achim Streit · Piotr Bala · Alexander Beck-Ratzka · Krzysztof Benedyczak · Sandra Bergmann · Rebecca Breu · Jason Milad Daivandy · Bastian Demuth · Anastasia Eifer · André Giesler · Björn Hagemeier · Sonja Holl · Valentina Huber · Nadine Lamla · Daniel Mallmann · Ahmed Shiraz Memon · Mohammad Shahbaz Memon · Michael Rambadt · Morris Riedel · Mathilde Romberg · Bernd Schuller · Tobias Schlauch · Andreas Schreiber · Thomas Soddemann · Wolfgang Ziegler

Received: 11 September 2009 / Accepted: 17 August 2010

Abstract UNICORE is a European Grid Technology with more than 10 years of history. Originating from the Supercomputing domain, the latest version UNICORE 6 has turned into a general-purpose Grid technology that follows established standards and offers a rich set of features to its users. The paper starts with an architectural insight into UNICORE 6, highlighting the workflow features, standards and the different clients. Next, the current state of advancement is presented by describing recent developments. The paper closes with an outlook on future planned developments.

Keywords Grid · Supercomputing · UNICORE · Middleware

A. Streit (✉) · S. Bergmann · R. Breu · J. M. Daivandy · B. Demuth · A. Giesler · B. Hagemeier · S. Holl · V. Huber · N. Lamla · D. Mallmann · A. S. Memon · M. S. Memon · M. Rambadt · M. Riedel · M. Romberg · B. Schuller
Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich GmbH, Jülich, Germany
e-mail: a.streit@fz-juelich.de, achim.streit@kit.edu

P. Bala · K. Benedyczak
Interdisciplinary Center for Mathematical and Computational Modeling (ICM), Warsaw University, Warsaw, Poland

A. Beck-Ratzka
Max Planck Institute for Gravitational Physics, Potsdam, Germany

A. Eifer · T. Schlauch · A. Schreiber
Simulation and Software Technology, German Aerospace Center (DLR), Cologne, Germany

T. Soddemann · W. Ziegler
Fraunhofer Institute for Algorithms and Scientific Computing (SCAI), Sankt Augustin, Germany

1 Introduction

In the last 3 years, activities in Grid computing have changed; in particular, in Europe, the focus moved from pure research-oriented work on concepts, architectures, interfaces and protocols towards activities driven by the usage of Grid technologies in day-to-day operation of e-infrastructure and in application-driven use cases. This change is also reflected in the UNICORE activities [1]. The basic components and services have been established, and now, the focus is on enhancement with higher-level services, integration of standards, deployment in e-infrastructures, setup of interoperability use cases and integration of applications. The development of UNICORE started back more than 10 years ago, when, in 1996, users, supercomputer centres and vendors were discussing 'what prevents the efficient use of distributed supercomputers?' The result of this discussion was a consensus which still guides UNICORE today: seamless, secure and intuitive access to distributed resources. In 2004, the UNICORE software became open source, and since then, UNICORE has been developed within the open source developer community.

The structure of the paper is as follows: In Section 2, the architecture of UNICORE 6, including implemented standards, is described. Section 3 covers recent developments and advancements, while in

Section 4, a glimpse outlook on future planned developments is given. The paper closes with a conclusion. An extended version of this paper with all details is available in [2].

2 Architecture and standards

2.1 Client layer

On the top layer, a variety of clients is available to the users, ranging from graphical clients such as the Eclipse-based URC (UNICORE Rich Client), to a command-line interface named UCC (UNICORE Command-line Client), to a programming API named HiLA (High Level API). For a tight integration of various types of applications, the GridBean concept [3] was invented, which offers an API to easily implement graphical client extensions and connect them with UNICORE 6's core functionalities (cf. Section 3.6 for a few GridBean examples).

2.2 Service layer

The middle layer comprises all services and components of the UNICORE Service-Oriented Architecture. Figure 1 shows three sets of services, the left and right ones containing services at a single site while the middle shows the central services, which serve all sites and users in a UNICORE Grid. The Gateway component acts as the entry point to a UNICORE site and performs the authentication of all incoming requests. It is used for both the central services and the single site services. The XNJS component is the job management and execution engine of UNICORE 6. It performs the job incarnation, namely, the mapping of the abstract job description to the concrete job description for a specific resource according to the rules stored in the IDB (Incarnation Database). The functionality of the XNJS is accessible via two service interfaces in UNICORE 6's WS-RF hosting environment. UNICORE 6's proprietary interface is called UNICORE Atomic Services (UAS) [3] and offers the

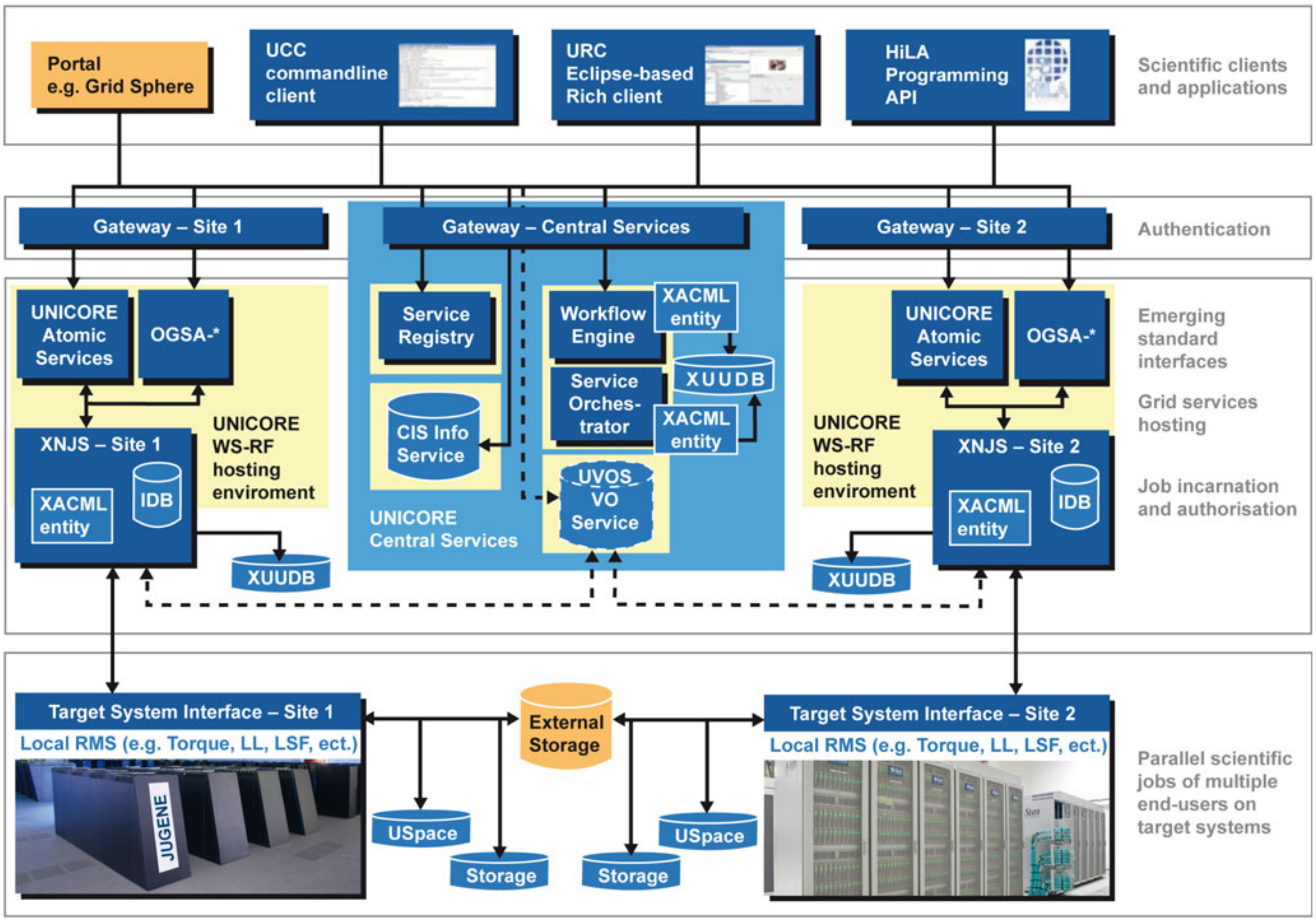

Fig. 1 UNICORE 6 architecture

full functionality to higher-level services, clients and users. In addition to the UAS, a standardised set of interfaces based on open, common standards is available (depicted as 'OGSA-*' in Fig. 1). For authorisation of users, the XNJS uses the XUUDB user database to perform the mapping from X.509 certificates to the actual users' logins. The XUUDB component is a Web service in itself, so that it can be used from multiple UNICORE installations, e.g. within the same computing centre. Like in many service-oriented environments, a central service registry is available, where the different services can register once they are started. The central service registry is necessary to build-up and operate a distributed UNICORE infrastructure. This service registry is contacted by the clients in order to 'connect to the Grid'. An infrastructure may have multiple central registries. From the beginning, workflow support has been of major importance for UNICORE. The two-layered design with a separation of the workflow engine and the service orchestrator was primarily done for better scalability, but also offers the possibility to plug-in domain-specific workflow languages and workflow engines. The workflow engine originates from the EU-project Chemomentum. Besides simple job-chains, while- and for-loops, workflow variables and conditional execution are supported. The service orchestrator deals with brokering the execution and monitoring of the workflow and its respective parts, as well as provides call-back mechanisms to the workflow engine. The resource brokering is performed via pluggable strategies. More details can be found in [4]. The workflow capabilities are offered to the users on the client layer via the URC. Furthermore, the definition, submission, monitoring and control of workflows is also possible from the UCC.

2.3 System layer

On the bottom layer, the Target System Interface (TSI) component is the interface between UNICORE and the individual resource management/batch system and operating system of a Grid resource. In the TSI component, the abstracted commands from the Grid layer are translated to system-specific commands. As the TSI component is performing the proper setting of users' UID and invocation of his/her environment, it needs to be executed with root privileges. The TSI is available for a variety of commonly used batch systems such as Torque, LoadLeveler, LSF, SLURM, OpenCCS, etc. The USpace is UNICORE's job directory. A separate directory exists for every job, where all input and output data are stored. The TSI also allows access to local file systems, such as the user's Home directory. These are called UNICORE Storages and can be used in stage-in and stage-out operations.

2.4 Standards

Common open standards are of major importance to realise interoperable world-wide Grid infrastructures in order to satisfy scientists who require resources in more than one Grid. In order to increase the interoperability with Grid infrastructures that deploy other middleware solutions, several standards from the Open Grid Forum and OASIS are used in UNICORE 6 in various domains. A full Web services stack based on WS-RF 1.2, SOAP, and WS-I is implemented to build UNICORE 6's service-oriented architecture. In security, full X.509 certificates are used as base line, while the access control is based on XACML policies. A support for SAML-based virtual organisation management service (VOMS) is available, as well as support for proxy certificates. The information service of UNICORE 6 called Common Information Service (CIS) [5] is based on the GLUE 2.0 information model and thus allows for a standardised way of exposing computational resources. For job management, OGSA-BES and several profiles (i.e. HPC-BP, HPC-FSP) are used for the creation, monitoring and control of jobs, including data-staging. Job definition is compliant with the JSDL standard and its profiles. In the area of data management and transfer, OGSA-ByteIO can be used for data transfer, both for site-to-site and client-to-site transfers. For a transfer of data from and to external storage, the GridFTP transfer protocol can be used as an alternative to the default HTTP-based mechanism.

3 Recent developments

3.1 Virtual organisation management

A key element of the UNICORE Virtual Organizations System (UVOS) [6] is the central server, which acts both as authentication service and attribute authority. It is used by two kinds of clients: consumers and management clients. Consumers do not modify the UVOS content but query it. The modification of the VO data can be performed using management clients. The UVOS system is highly portable, and all UVOS server operations are available via the web services interface. The UVOS consumers use open standard SAML 2.0 as a communication protocol and the server implements the core SAML specification together with additional profiles.

3.2 Shibboleth integration in URC

Bridging Shibboleth [7] and the UNICORE 6 security model enables Single Sign-On (SSO) among multiple UNICORE 6 Grid sites, as well as authentication and authorisation interoperability with all other middlewares supporting Shibboleth and GridShib-CA. As with Shibboleth, users authenticate and authorise while using a Web browser, this creates a mismatch with the UNICORE 6 security model, as in UNICORE 6, the client usually is a non-Web browser and uses X.509 certificates for user authentication and SOAP for message exchange. Therefore, a GridShib-Certificate Authority (CA) is needed to provide X.509-based Short-Lived Grid Credentials (SLC) with embedded SAML2 assertions. Nonetheless, the GridShib-CA does not solve the problem of using Shibboleth with UNICORE 6 as it still restricts the user to use a Web browser to fetch the user's SLC. Hence, a client API was developed to allow any non-Web browser client to communicate with the GridShib-CA; it is implemented as an Eclipse-based URC plug-in.

3.3 Interactive access in URC

To realise an interactive access in the URC, it is extended by Eclipse plug-ins, which provide terminal emulation and SSH communication functionality. The terminal plug-in is based on the terminal emulation software from the gEclipse project [8]. It offers a standard VT100 emulation, which allows the display of graphical output from programs using the curses library. The plug-ins have been designed in an extensible way, so that multiple communication providers can be plugged in. These communication providers are likewise implemented as Eclipse plug-ins, which encapsulate the connection and communication logic and extend the functionality of the terminal plug-in. Currently, three different communication providers have been implemented: X.509 enabled SSH, GSI-SSH and standard plain SSH with password. The latter plug-in could, in principle, also be used without UNICORE 6 in any other Eclipse framework.

3.4 DataFinder integration

The DataFinder [9] system provides sophisticated means for data organisation and is in productive use in various scenarios at the German Aerospace Centre (DLR). Integrating DataFinder in UNICORE 6 [10] delivers logical organisation of data objects and abstracts common data management concepts hiding the specifics of storage systems. On this basis, advanced means for data organisation through metadata management functionalities and support of custom data models are provided. Moreover, the integration with UNICORE 6 is achieved by the data management client API, which has been integrated in the URC. The developed system and its compatibility with DataFinder have been successfully validated in the project AeroGrid.

3.5 JavaGAT

The Grid Application Toolkit (GAT) [11], a high-level API for accessing Grid services, provides application developers with a unified, simple and middleware-independent interface to the Grid. GAT was first implemented in C and next in Java, named JavaGat. One of the major advantages of JavaGAT [12] compared to C-GAT is that JavaGAT enables Grid access without an additional installation of a Grid middleware. The UNICORE adaptor for JavaGAT is based on HiLA [13], as it supports the access to UNICORE 6 via an easy and unique API. Furthermore, it is not necessary to install components of UNICORE on the submitting host.

3.6 Support of scientific applications through GridBeans

Application support in the URC is realised by the GridBean concept [14]. The advantage of the GridBean technology is that GridBeans can be easily developed and independently distributed to the users. For the molecular dynamics (MD) simulation package AMBER [16] and the computational quantum chemistry program Gaussian [15], two GridBeans were recently developed. The AMBER GridBean was established within a new GridBean concept, the sequence GridBeans [17]. This concept allows the management of applications, which consist of a set of programs that need to be executed in a sequence, one program after another, on the same target system. The AMBER package makes use of this concept, as it consists of approx. 50 programs, and a subset of them is usually run in sequence for one calculation. The sequence GridBean concept provides a framework for automatically building the required GUIs for such a sequence. The Gaussian GridBean assists the users in the preparation of the required input parameters and allows to generate them automatically without requiring knowledge of the internal format of the input file. Furthermore, the user input and specific resource settings will be verified to check the correctness of a job setting before a job can be submitted to a remote system.

3.7 Monitoring of UNICORE 6 services in operation

The Simon 6 tool has been developed to check the health status of every UNICORE 6 component. The availability and functionality tests are submitted into the UNICORE infrastructure using the Gateway as the entry point by using the UCC. All UNICORE 6 components (registry, workflow engine, XNJS, XUUDB and TSI) are monitored one after the other, and afterwards, the results come back to SIMON 6 again. If any problem with a UNICORE 6 component occurs, the administrator will be informed by email or a short text message to his/her mobile phone.

3.8 Remote administration of UNICORE 6

The decentralised topology and distributed operation of a Grid make a strong case for dynamic service deployment and remote administration capabilities. Services can be dynamically deployed at runtime either via an administration Web service interface or on the file system level by copying (either locally or remotely) a Web service JAR into a specified directory triggering an automatic deployment task. The new AdminService provides means for, e.g. (un)deploying UNICORE 6 services, (de)activating dynamic and automatic deployment of UNICORE 6 services, etc. In addition, the AdminService logs all its actions and outcomes (success or failure and cause) in a retrievable manner. Since there might be scenarios in which remote administrators should not be allowed access to all UNICORE 6 configuration properties (e.g. keystore and database passwords), local administrators must explicitly enable remote access (either ‘read’ or ‘read–write’). Additionally, a mechanism has been added to allow local administrators to set UNICORE configuration properties at runtime without having to use an application programming interface, as well as to overrule remote administrators.

3.9 New workflow constructs

The UNICORE workflow system offers basic workflow constructs like *while-loops* for repeating certain tasks automatically and *if-statements* for altering the flow of execution based on the evaluation of conditions. These constructs can also be nested in order to address more complex tasks. In addition, the system supports the parallel execution of many similar jobs or sub-workflows by introducing the *for-each-loop*. This powerful construct helps to minimise the effort of creating workflows with many jobs and reduce the size of the resulting workflow descriptions. There are numerous usage scenarios where extensive parallel job processing is desired and for-each-loops can be applied. For covering these scenarios, the for-each-loop offers two different modes of operation: iteration over sets of differing parameter values and iteration over a set of files.

4 Future developments

Scientific applications require different execution environments on computing resource, e.g. a parallel MPI execution. These environments are mostly defined by the execution mode, as well as their specific parameters and variables. Since UNICORE 6 users want to work in different execution environments, the usage of different and configurable execution environments should be ensured in Grid infrastructures. To facilitate this, a new design concept is developed to support the execution of applications in different execution environments within UNICORE 6. This new development provides a powerful and easy-to-use configuration mechanism in the URC. In detail, a new section in the URC resource panel, to set the execution environment will be added. It will allow for modifying parameters, as well as variables, and provides an immediate validation of those. All available execution environments, their parameters, variables and validators of a specific system are described by the system administrator in the XNJS and its IDB. These descriptions are exported by the URC and are automatically graphically displayed in the new section of the resource panel, to be configured by the user. To sum up, this new execution environment concept will provide users with additional options for easier and more precise definition of their application execution on the Grid.

5 Summary

In this paper, we presented recent and future advancements of the UNICORE Grid technology. UNICORE has a three-layered architecture, with client, service and system layers. Several standards from the Web services and Grid domain are implemented in UNICORE 6 so that standards-based interfaces are offered. Recent functional additions to UNICORE 6 contain a support for VOs and Shibboleth in the security domain, an interactive access based on X.509 certificates, and the integration of the DataFinder for improved data management. The support for applications from users is improved through new application-specific GridBeans and the implementation of JavaGAT as a new client to UNICORE. In addition, new workflow constructs to allow for-each-loops and an iteration over file-sets was added. To improve the operational aspects of UNICORE 6, a new monitoring tool was developed, which allows to monitor both the availability and functionality of UNICORE 6 services. Furthermore, the administration capabilities of UNICORE 6 were enhanced with a dynamic service deployment technology. Future advancements of UNICORE 6 contain improved execution environments to, e.g. specify the parameters and variables of MPI jobs in a more intuitive way.

Acknowledgements The work summarised in this paper was done by many people. We gratefully thank them for their past, present, and future contributions in developing the UNICORE technology. Most of the work described here was supported and funded by the Helmholtz Programme 'Scientific Computing' and various projects from the European Commission and national research ministries under the respective contract numbers.

References

1. Streit A, Erwin D, Lippert Th, Mallmann D, Menday R, Rambadt M, Riedel M, Romberg M, Schuller B, Wieder Ph (2005) UNICORE—from project results to production grids. In: Grandinetti L (ed) Grid computing: the new frontiers of high performance processing, advances in parallel computing, vol 14. Elsevier, Amsterdam, pp 357–376
2. Streit A, Bala P, Beck-Ratzka A, Benedyczak K, Bergmann S, Breu R, Daivandy JM, Demuth B, Eifer A, Giesler A, Hagemeier B, Holl S, Huber V, Lamla N, Mallmann D, Memon AS, Memon MS, Rambadt M, Riedel M, Romberg M, Schuller B, Schlauch T, Schreiber A, Soddemann T, Ziegler W (2010) UNICORE 6—recent and future advancements, JUEL-4319. ISSN 0944-2952
3. Riedel M, Schuller B, Mallmann D, Menday R, Streit A, Tweddell B, Memon MS, Memon AS, Demuth B, Lippert Th, Snelling D, van den Berghe S, Li V, Drescher M, Geiger A, Ohme G, Benedyczak K, Bala P, Ratering R, Lukichev A (2007) Web services interfaces and open standards integration into the European UNICORE 6 grid middleware. In: Proceedings of 2007 middleware for web services (MWS 2007) workshop at 11th international IEEE EDOC conference "the enterprise computing conference". IEEE Computer Society, Annapolis, pp 57–60. ISBN 978-0-7695-3338-4
4. Schuller B, Demuth B, Mix H, Rasch K, Romberg M, Maran U, Sild S, Bala P, del Grosso E, Casalegno M, Piclin N, Pintore M, Sudholt W, Baldridge KK (2008) Chemomentum—UNICORE 6 based infrastructure for complex applications in science and technology. In: Euro-Par 2007 workshops: parallel processing. LNCS, vol 4854. Springer, Berlin Heidelberg New York, pp 82–93
5. Memon AS, Memon MS, Wieder Ph, Schuller B (2007) CIS: an information service based on the common information model. In: Proceedings of 3rd IEEE international conference on e-science and grid computing, Bangalore, India. IEEE Computer Society, pp 465–472. ISBN 0-7695-3064-8
6. Unicore VO System (2010) The UVOS project. http://uvos.chemomentum.org
7. Shibboleth. (2010) Shibboleth homepage. http://shibboleth.internet2.edu
8. Eclipse (2010) gEclipse Project. http://www.geclipse.eu/
9. Schlauch T, Schreiber A (2007) Datafinder - a scientific data management solution. In: Ensuring the long-term preservation and value adding to scientific and technical data, PV 2007, Oberpfaffenhofen, Germany
10. UNICORE DataFinder project. http://tor-2.scai.fraunhofer.de/gf/project/unicoredata/
11. Allen G, Davis K, Dramlitsch T, Goodale T, Kelley I, Lanfermann G, Novotny J, Radke T, Rasul K, Russell M, Seidel E, Wehrens O (2002) The GridLab grid application toolkit. In: Proceedings of the 11th IEEE international symposium on high performance distributed computing (HPDC). IEEE Computer Society, p 411
12. Beck-Ratzka A, Zangerl T (2009) GAT beginners guide. http://www.aei.mpg.de/ alibeck/documents/gat-usage.pdf
13. UNICORE (2009) HiLA 1.0. http://www.unicore.eu/community/development/hila-reference.pdf
14. Ratering R, Lukichev A, Riedel M, Mallmann D, Vanni A, Cacciari C, Lanzarini S, Benedyczak K, Borcz M, Kluszcynski R, Bala P, Ohme G (2006) GridBeans: supporting e-science and grid applications. In: Second IEEE international conference on e-science and grid computing: IEEE conference proceedings, p 45
15. Gaussian (2009) Gaussian homepage. http://www.gaussian.com/
16. Amber (2010) The Amber Molecular Dynamics Package. http://ambermd.org/
17. Holl S, Riedel M, Demuth B, Romberg M, Streit A, Kasam V (2009) Life science application support in an interoperable e-science environment. In: Proceedings of IEEE international symposium of computer-based medical systems (CBMS) (in press)