Scalable Computing: Practice and Experience
Volume 11, Number 2, pp. 161–172. http://www.scpe.org
ISSN 1895-1767

VINE TOOLKIT—TOWARDS PORTAL BASED PRODUCTION SOLUTIONS FOR SCIENTIFIC AND ENGINEERING COMMUNITIES WITH GRID-ENABLED RESOURCES SUPPORT

DAWID SZEJNFELD, PIOTR DZIUBECKI, PIOTR KOPTA, MICHAL KRYSINSKI, TOMASZ KUCZYNSKI, KRZYSZTOF KUROWSKI, BOGDAN LUDWICZAK, TOMASZ PIONTEK, DOMINIK TARNAWCZYK, MAŁGORZATA WOLNIEWICZ*, PIOTR DOMAGALSKI, JAROSLAW NABRZYSKI, AND KRZYSZTOF WITKOWSKI†

Abstract. In large scale production and scientific, academic environments, the information sets to perform computations on come from various sources. In particular, some computations may require the information obtained as a result of previous computations. Workflow description offers an attractive approach to formally deal with such complex processes. Vine Toolkit [1] solution addresses some major challenges here such as the synchronization of distributed workflows, establishing a community driven grid environment for the seamless results sharing and collaboration. In order to accomplish these goals Vine Toolkit offers integration on different layers starting from rich user interface web components embedded in portal frameworks like GridSphere [28] or Liferay [26], integration with workflow engine and grid security and ending up with a built-in meta-scheduling mechanisms, that allow IT administrators to perform load balancing automatically among computing clusters and data centers to meet peak demands. As a result of the wow2green [5] project a complete solution has been developed and delivered and still is being adopted in the pending projects like PL-Grid [4].

Key words: grid portals, web portal, workflows, grid computing, grid technology, grid, Vine toolkit, Vine, GRMS, flowify, wow2green, PL-Grid, GRIA, middleware, GridSphere, liferay

1. Introduction. Vine Toolkit (in short Vine) is a modular, extensible Java library that offers developers an easy-to-use, high-level Application Programming Interface (API) for Grid-enabling applications and more as it will be described in this article. It was developed within EU-funded projects like OMII_Europe [2] or BEinGRID [3] and polish infrastructural project PL-Grid [4]. It was used to build advanced portal solution integrating many different grid technologies within BEinGRID business experiment called wow2green (Workflows On Web2.0 for Grid Enabled infrastructures in complex Enterprises) [5]. The solution will be described to illustrate the great scope of Vine usage and its advanced features. Integration with web frameworks like Liferay [26] allows developers to build production quality web collaboration solutions. The article is an extended version of the PPAM 2009 article [27]—individual paragraphs were extended and some new added.

1.1. State of the art. Currently there are several grid portals on the market. While some of them allow only to submit and manage jobs in the dedicated system, other are application development platforms designed for building and running simulations in multiple middleware solutions. There are also environments which put focus on the workflow or collaboration capabilities. Possibility of working with multiple middleware at the same time combined with workflow support, collaboration capabilities as well as possibility of integration with multiple portal frameworks are features which distinguish Vine Toolkit amongst competitive open-source grid portal solutions.

Some of the grid portals which are related to the Vine Toolkit are described in the following subsections.

1.1.1. P-GRADE. [6]—first highly integrated parallel application development system for Grid and clusters. It uses Globus, Condor-G and MPICH-G2 as grid-aware middleware to conduct computations. To provide Grid programming interface with support for workflow and orchestration P-GRADE uses GRAPNEL language which allows a programmer to set a workflow of objects/library calls. Special tool may be used to generate MPI code from a GRAPNEL program. Supports legacy applications written in: C, C++, Fortran, MPI, PVM.

P-GRADE Grid Portal—is a web based, service rich environment for the development, execution and monitoring of workflows and workflow based parameter studies on various grid platforms. P-GRADE Portal hides low-level grid access mechanisms by high-level graphical interfaces, making even non grid expert users capable of defining and executing distributed applications on multi-institutional computing infrastructures. Workflows and workflow based parameter studies defined in the P-GRADE Portal are portable between grid platforms without learning new systems or re-engineering program code. Technology neutral interfaces and concepts of the P-GRADE Portal help users cope with the large variety of grid solutions. More than that, any

*Poznan Supercomputer and Networking Center, ({dejw, piotr.dziubecki, pkopta, mich, tomasz.kuczynski, krzysztof.kurowski, bogdanl, piontek, dominikt, gosiaw}@man.poznan.pl).
†FedStage Systems, ({piotr.domagalski, krzysztof.witkowski, jarek.nabrzyski}@fedstage.com).

P-GRADE Portal installation can access multiple grids simultaneously, which enables the easy distribution of complex applications on several platforms. Interoperability with Globus Toolkit 2, Globus Toolkit 4, LCG and gLite grid middlewares. Secured grid access mechanisms using X.509 certificates and proxy credentials. Built-in graphical editor to design and define grid workflows and grid parameter studies. Integrated workflow manager that orchestrates the fault tolerant execution of grid applications. On-line workflow and job monitoring and visualization facilities. MPI execution in Globus and gLite grid environments. Graphical MDS and BDII grid information system browser. Support to include local and remote storage files into grid applications. User level storage quota management to protect against server overloading. Legacy application publish and reuse capabilities by the GEMLCA mechanism. Workflow import-export-archive service.

License: GPL

1.1.2. myExperiment.org. [8]—is a collaborative environment where scientists can safely publish their workflows and experiment plans, share them with groups and find those of others. Workflows, other digital objects and bundles (called Packs) can now be swapped, sorted and searched like photos and videos on the Web. Unlike Facebook or MySpace, myExperiment fully understands the needs of the researcher and makes it really easy for the next generation of scientists to contribute to a pool of scientific methods, build communities and form relationships—reducing time-to-experiment, sharing expertise and avoiding reinvention. Launched in November 2007, contains the largest public collection of workflows across multiple workflow systems including Taverna (over 800 workflows) and Trident and is used by thousands of users ranging from life sciences and chemistry to social statistics and music information retrieval. Project source code is accessible and written in Ruby on Rails.

License: BSD

1.1.3. G-POD. [7]—European Space Agency Earth Observation G-POD (Grid Processing on Demand), it is a generic GRID-based operational environment where specific data handling applications can be seamlessly plugged into system. Coupled with high-performance and sizeable computing resources managed by GRID technologies, it provides the necessary flexibility for building an application virtual environment with quick accessibility to data, computing resources and results. The G-POD web portal is a flexible, secure, generic and distributed platform where the user can easily manage all its tasks. From the creation of a new task to the result publication, passing by the data selection and the job monitoring, the user goes trough a friendly and intuitive interface accessible from everywhere. The portal offers access to, and support for, science-oriented Earth Observation GRID services and applications, including access to a number of global geophysical ENVISAT products. Globus and gLite is used as main target middleware.

1.1.4. EnginFrame. [9] (EF) is a web-based front-end for job submission, tracking and integrated data management for HPC applications and other services. EnginFrame can be easily plugged on several different scheduler or grid middleware like: Platform LSF, Sun Grid Engine, PBS, gLite. Every service defined in the EF portal respects the JSR168 standard and can be exposed in any portlet container. EF is certified to be compliant with IBM WebSphere Enterprise Portal. Every service defined in the portal is automatically exposed trough a WSDL interface so it can be accessed like a web-service application and comes with a J2EE API so you can plug your java application to it. Can be linked with any authentication system (NIS, LDAP, Active Directory, MyProxy). Authorization system decides which services the user is able to see and/or use. Portal administrators can restrict the access to some applications to a defined class of users. Supports remote desktop tools like: Citrix metaFrame, IBM DCV, NoMachine's NX, VNC.

License: EnginFrame uses a proprietary license.

1.2. Problem Description. The general problem solved within wow2green business experiment is presented in this paragraph. This use case is just an illustration of how and where Vine Toolkit can be used. This article is focused mainly on Vine itself and only on the business logic layer without going into end user interface and functionality description.

Big companies consist of several units generating a variety of information sets and performing advanced computations. Workflow description offers an attractive approach to formally deal with complex processes. Unfortunately, existing workflow solutions are mostly legacy systems which are difficult to integrate in dynamically changing business and IT environments. Grid computing offers flexible mechanisms to provide resources in the on-demand fashion. Wow2green system—called officially Flowify [10] which is based on the Vine Toolkit—is a

Grid solution managing computationally intensive workflows used in advanced simulations. Each department within big enterprise or a small company in a supply chain:

- performs unsynchronized computations among departments,
- exchanges computing results independently with other project participants, so input data is re-entered in different formats, output data is lost or overwritten,
- overloads computing resources during deadlines.

Flowify helps users to synchronize the data processing for execution and collection of results in an automated fashion. Easy-to-use Flowify workflow interfaces are provided as intuitive Web applications or integrated with commonly used tools. Thus many users can simultaneously form dynamic workspaces, share not only data for their computing experiments but also manage the whole process of advanced computations. Built-in GRMS meta-scheduling mechanisms provided by Flowify allows IT administrators to perform load balancing automatically among computing clusters and data centers to meet peak demands.

Similar scenarios are present also in scientific, academic environments where people use scientific applications (often open-source) to conduct computations and analyze big sets of data. Often the relations between persons involved are less formal and there is no such restrictive bi-directional relations between members of the team working on the given problem. Vine with a set of web applications is an excellent solution to prepare web portal environment for advanced scientific and engineering applications with grid-enabled resources in the backend. Moreover, the heterogeneity of resources brings about an opportunity to integrate different sets of geographically-scattered grid resources. Integration scenario could be tightened by applying meta-scheduler for different sets of grid resources and middleware systems and make it the main entry to the grid. Component-based architecture of the GRMS meta-scheduler allows administrators to apply scheduling policies and brokering functionality across organization boundaries.

1.3. Main Requirements and Demands. The following main requirements have been identified within wow2green experiment:

- *enable workflow executions via the wow2green portal* (portal based solution should provide a suitable, user-friendly interface for workflow executions using Web 2.0 style easy-to-use web interfaces. Users should have a fully transparent access to underlying computing resources so they can take advantage of high level workflow management)
- *provide detailed information and feedback for users about their computing workflow simulations* (every workflow is a kind of template for creating a computing simulation which could be later executed on different computing resources. External workflow engine was selected—Kepler [11] which supports provenance mechanisms required to search and discover templates in created workflows. These features are also available via the wow2green portal)
- *enable more efficient platform usage through applying brokering of jobs defined in workflow nodes* (GRMS [12] broker is used as a meta-scheduling and brokering component for load balancing among computing resources. Thus, end users will not have to select computing resources manually and the overall utilization of computing resources will be improved), end user is freed from resource selection decision every time something is computed what is troublesome in large infrastructures
- *easy-to-use stage in and stage out simulation data* (users are able to upload and load simulation data and workflows to and from our portal in an intuitive way), this requirement is quite generic and appears in many other scenarios—users wants easier data management what could be especially hard to master in a large and heterogeneous infrastructures—portal solution makes it easier, users may overcome many troubles like firewall issues for example
- *easy-to-use, Kepler-like graphical user interfaces for workflow management and control* (to provide user friendly web interfaces we selected a new web technology called Adobe Flex. It will enable us to implement various web applications for easy management and control of workflows via portal)
- *support for new workflow definitions* (additional web tools for workflows will be developed to enable users to create, share and modify workflows. Users (workflow creators) will be able to share created workflows with other users interested only in performing simulations of existing workflows).
- *provide a repository of workflow definitions*

2. Flowify Architecture. The overall Flowify architecture is presented below, all components are described to show its function and reveal the complexity of the prepared environment (Fig.2.1).

Identified components:

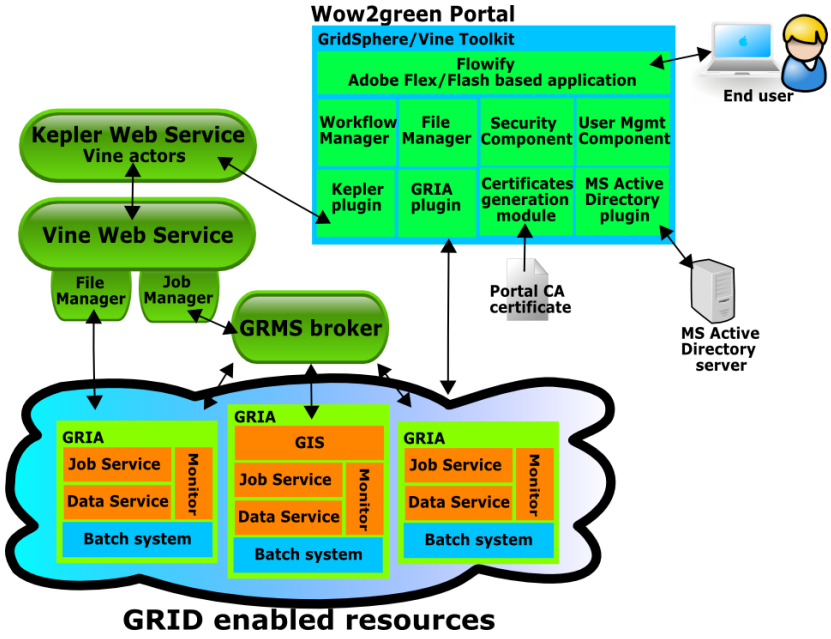

FIG. 2.1. *Wow2greem system (Flowify) overall architecture.*

- *Gridsphere portal framework*—portal framework that provides an open-source portlet based Web portal. GridSphere enables developers to quickly develop and package third-party portlet web applications that can be run and administered within the GridSphere portlet container. The wow2green portal is based on this framework, production-ready portal for a larger community could be deployed on Liferay
- *Vine Toolkit*—used as business model layer within the portal, applications created for the scenario uses the Vine mechanisms to interact with grid services and workflow engine, integrated Adobe Flex with BlazeDS technology allows creating advanced web applications. Vine Toolkit components used in portal:
  - *Workflow Manager*—general component used to interact with different workflow management solutions through common API; in case of the wow2green solution kepler plugin was introduced
  - *Role Manager*—role management component which allows administrators to manage role-based access to GUI elements and accessible actions and also to define role-based access policies regarding data managed through logical file manager component
  - *File Manager*—File Manager component which allows for interacting with different data management systems by means of common API
  - *Portal Security Component*—used for portal user certificates and proxies management, together with the Certificates generation module allows for automatic certificate generation for registered portal users
  - *Portal User Management*—used for portal users management, user registration in the portal; together with the MS Active Directory plugin it enables using existing users in the given organisation to make the integration easier with the existing IT environment (it is also possible to use some existing LDAP service it desired)
- *Vine web service*—web service which exposes a subset of functionality of the Vine Toolkit component to external systems/services—regarding Job and File Manager, the web service is used by Kepler actors to interact with grid enabled resources
- *Kepler workflow engine*—core part of the Kepler/Ptolemy standalone application/framework for workflow management, it is used in the wow2green solution as primary workflow management engine, kepler workflow description called MOML is used within the system
- *Kepler engine web service*—the web service exposes the workflow engine functionality to external systems/services, portal workflow manager component uses it as the workflow management service
- *Set of kepler's vine actors*—base components of the workflows' definitions to submit jobs and manage data through the Vine Toolkit service layer

- *GRMS broker with web service interface*—job broker component with advanced scheduling technology, scheduling policies could be easily exchanged dependently on the requirements, for the wow2green experiment version 2.0 was deployed
- *GRIA middleware [13]*—grid middleware—service-oriented infrastructure (SOI) designed to support B2B collaborations through service provision across organizational boundaries. Following services are used in the wow2green system: Job Service job management, Data Service data management, Membership Service to manage groups of users relevant to the roles defined in the portal
- *Griddle Information Service*—web service which exposes the information about available GRIA resources and batch queues statistics

In case of new scenarios for scientific communities, GRMS 3.0 is used, bringing about new possibilities like support for cross-cluster MPI application support. Such functionality was successfully applied during the QosCosGrid [22] project and could be useful with some scientific applications applied on many clusters. The new version of broker also introduces built-in support for workflows, which makes the whole architecture simpler and more robust rendering the workflow management much consistent and powerful. More advanced scheduling policies are possible and job-related data management is much more consistent and better operated. The workflow definition is also simplified which makes it more suitable for a subset of scenarios and enables faster implementation for the new scientific applications.

3. Vine Toolkit portal based solution. Vine Toolkit (Fig. 3.1) is a modular, extensible Java library that offers developers an easy-to-use, high-level Application Programming Interface (API) for Grid-enabling applications. Vine can be deployed for use in desktop, Java Web Start, Java Servlet 2.3 and Java Portlet 1.0 environments with ease. Additionally, Vine Toolkit supports a wide array of middleware and third-party services. Using Vine Toolkit, one composes applications as collections of resources and services for utilizing those resources (basic idea in Vine—generic resource model—any service and data source can me modeled as an abstract entity called resource and can be integrated with web applications using high-level APIs).

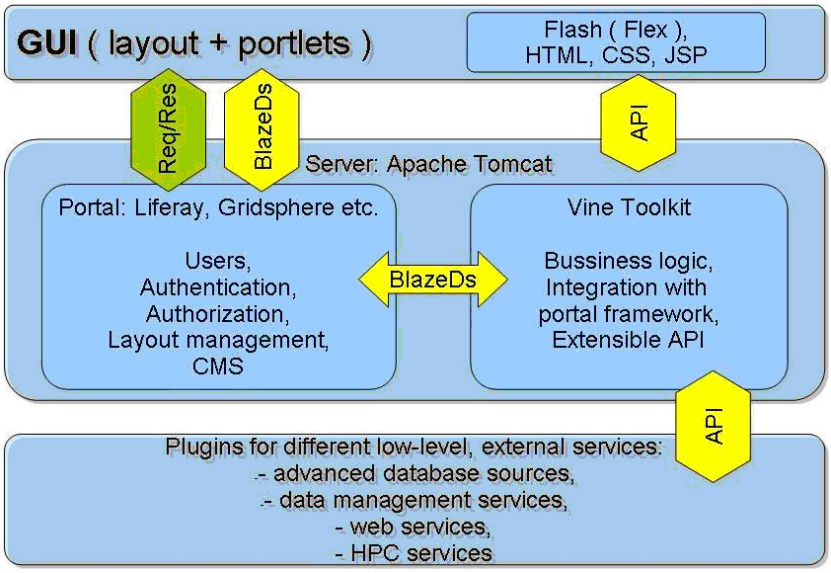

FIG. 3.1. *Vine Toolkit high level architecture in portal deploy scenario.*

The Vine Toolkit makes it possible to organize resources into a hierarchy of domains to represent one or more virtual organizations (VOs). Vine offers security mechanisms for authenticating end-users and authorizing their use of resources within a given domain. Other core features include an extensible model for executing tasks (every action is persisted as Task) and transparent support for persisting information about resources and tasks with in-memory or external relational databases. Adobe Flex [14] and BlazeDS [15] technology allows for creating advanced and sophisticated web applications similar to many stand-alone GUIs. Other GUI technologies like GWT, HTML, CSS and Javascript with Ajax could be used instead where Vine business logic layer could be used below to interact with different resources. Vine comes with many co-bundled components.

There is a set of bundled components. User / Role / Application managers—administrative tools for user management, their roles, application related right management for accessible methods of web applications, user login and user registration—it is module-based so the process could be extended and span any external service. like ldap or ms active directory and portal incorporates authenticated users automatically. Resource manager where the configuration of resources is accessible, could be displayed and changed by advanced users if desired. File browser component allows users to manage data on different data management servers and also portal file system (PFS)—Vine provides such components to allow storing of users' data in the portal. Job Manager component based on JSDL specification allows users to prepare JSDL based job description and submits jobs. Jobs are monitored and outputs could be retrieved later. Basic information about the job is also displayed. Different job managers are used here according to the installed and configured plugins. Credential manager allows for getting proxy credentials from MyProxy server configured for the portal. User is able also to add some mapping to his accounts for the accessible DN—so later it is possible to log in using MyProxy user account—single sign-on is possible then, proxy certificates are loaded automatically and user can use grid services directly. Resource browser is able to display information about grid resources—globus MDS services are used here—it is also possible to show dynamic values such as free memory.

3.1. Workflow Engine Support. Vine Toolkit API has been enriched with the generic Workflow Engine API. It was created as a separate subproject which could be added as a feature during the portal installation. API reflects identified requirements regarding functionality used within end user web applications to manage users' workflows through the independent API. The following main functionality was included in the workflow submission interface: *startWorkflow*, *stopWorkflow*, *pauseWorkflow*, *resumeWorkflow*, *terminateWorkflow*, *getWorkflowOutputs*, *getWorkflowStatusById*, *createWorkflowSpec* and beside this a set of support methods like *getWorkflowByTaskId* to retrieve workflow instance from Vine using its task id. It allows quite simply add support for different workflow engines if desired or required by some applications constraints. For the wow2green experiment Kepler engine was used as the primary workflow engine. The base API allows define so called *WorkflowDefinition* which represents the workflow definition prepared as a XML string. In the case of Kepler engine MoML [16] based workflow description is used. Through the use of generic *WorkflowDefinitionParameter* a set of input parameters could be passed along with the workflow definition to the target workflow engine plugin. *WorkflowManager* component plays role of the generic workflow manager while the concrete instance is created during workflow submission to the service deployed on the target host. Resources defined in the Vine domain description (it is a resource registry in the given domain). Vine contains Kepler plugin project which encapsulates the Kepler engine client to implement the generic Vine Workflow API. The figure 2.1 clearly shows the connection from the portal to the *Kepler Web Service* it is another component which exposes the functionality of the Kepler workflow engine to the external clients acting as a workflow management service. The key point here are so called the *Vine actors*—small java classes which are in fact *Vine Web Service* clients. These "bricks" are used by the workflow designer to prepare workflow description to execute applications using grid resources and to acquire results of the computations. User is not aware where the computations take place—*Vine Web Service* by using grid services like GRMS which meta-scheduler and job broker—dispatches the jobs on the accessible grid environment. Vine is used here in different separate role in servlet mode to be a backed of the *Vine Web Service* and provide ease and unified access to different grid services like in this case GRMS and GRIA. Such construction allows easily switch to different grid technology and reconfiguration of the whole environment to different middleware stack like Unicore 6 [17]. In case of new application scenarios and new GRMS 3.0 it is possible to use the broker directly regarding application workflows. New GRMS broker supports DAG-based workflows, which are enough for most of the potential applications and possible dependencies between them. New project added to the Vine allows send XML-based GRMS workflows to the broker service. It gives a great opportunity for many scientists to connect their computing clusters on-demand to share computing resources and run large-scale simulations reducing the execution time or dealing with much bigger problem instances. Two parallel programming environments like OpenMPI and ProActive cover a wide range of programming languages, including Fortran, C/C++, Python (Open MPI) or Java (ProActive), thus can be easily integrated with legacy code or naturally used as a communication layer among parallel processes and threads. So by using GRMS workflow description with massive parallel applications could bring new possible scenarios and even reduce potential costs of extending of the existing infrastructure. Resources could be shared to run big parallel applications between different organizations.

3.2. Uniform interface to HPC infrastructures. In order to provide end-users with transparent access to resources, we developed a mechanism responsible for the management of uniform interfaces to diverse Grid middleware. Advanced classloader management mechanism enables dynamic loading of components that provide access to the functionality of specific Grid middleware through a uniform interface. These components are called plug-ins in this context (and are realized as Vine subprojects with separate set of client jars). These uniform interfaces are provided for such functionality like job submission, control, monitoring, data management, role management, user registration and authentication. They are based on standards where possible (e.g. JSDL for job submission) and take into account both the gathered requirements and the functionality provided by Grid middleware deployed in HPC centers. The provided interface could be extended if needed of course. Such flexibility is required as we use different workflow or single job xml descriptions regarding different workflow management services like Kepler or GRMS. By extending *WorkflowDefinition* or *JobSpec* interfaces, developer is able to add support for other workflow or job descriptions. Thus many workflow or job management services could work at the same time hiding complexity of the grid infrastructure by web application unified GUI interfaces.

3.2.1. Job Submission, Control and Monitoring (JSMC) Interface. One of the interfaces which makes use of the mechanism described above is the uniform interface for job submission functionality. It was based, firstly, on an analysis of the functionality of middleware systems and, secondly, on the Job Specification Description Language (JSDL) [18] created by the JSDL working group at the Open Grid Forum [19]. JSDL specifies different information required for successful application execution. One of the sub elements is application element which allows the user to specify the application to be executed on the Grid and the various application parameters and arguments, including input and output files. Requirements element allows the user to specify different requirements needed for the execution of the job, such as amount of memory, number of CPUs, etc. Data element allows the specification of input and output staging, in order for the user to optionally define which files should be copied to and / or from the working directory on the host where the job will be executed before job submission and / or after job completion respectively. The following main functionality was included in the job submission interface: *startJob*, *stopJob*, *getJobOutputs*, *getJobStatusById*, *createJobSpec* and beside this a bunch of support methods like *getJobByTaskId* to retrieve job instance from Vine using its task id. Currently Vine supports such middleware like gLite 3.x [20] WmProxy and CREAM, Unicore 6 UnicoreX job submission, Globus GT4 [21] WsGram, GRIA Job Service and recently QosCosGrid SMOA Computing and GRMS. Vine's plugin management allows for simultaneous usage of different middleware technologies what enables uniform access to the different HPC resources. In case of wow2grenn use case GRIA and GRMS Plugins were used to access grid services. Another motivation beside seamless usage of different grid technologies was ease of configuration and possibility to switch on to different mieddleware stack without modifications to the developed web applications (Vine plays role of the abstract access layer with well defined interface). JSMC service persists the information about all executed jobs along with all required details and the submitted specification. Monitoring data consists of job state history and basic information like start, finish time for example.

3.2.2. Data Management Interface. Another interface is the uniform interface for data management functionality. It was based, first, on an analysis of the functionality of storage services of middleware systems trying to find common API. The following main functionality was included in the data management interface: *getHomeDirectory*, *createInputStream*, *createOutputStream*, *info*, *exists*, *list*, *goUpDirectory*, *changeDirectory*, *makeDirectory*, *copy*, *rename*, *move*, *delte*, *upload*, *download*, *createNativeURLString* and beside this a bunch of support methods like *getDefaultFileManager* to retrieve default data manager instance from Vine. Currently Vine supports such middleware like gLite 3.x SRM and LFC, Globus GridFTP, Unicore 6 Storage Management System (SMS), WebDAV, Storage Resource Broker (SRB), GRIA Data Service, HTTP. Vine's plugin management allows to simultaneous usage of different storage technologies and copying data between them. In case of wow2green use case GRIA Data Service and HTTP Data Service plugins were used to access grid services. By using Vine abstract layer the web applications could be written without taking into account different technological nuances. Vine also allows use different storage servers as uniform services making data management quite easy for the application developer. Vine is also shipped with the built-in Portal File System (PFS) which allows store portal end users files directly in the disk storage managed by the portal container. It is very convenient for developers to expose data by using disposable links and store users configuration files and others close to the portal instance. PFS is also used while downloading files from remote data storage systems. In conjunction with security related mechanisms and components Vine simplifies usage of different data management systems

not only by unifying the API but also by covering the whole complexity related to the security issues. Vine also defines logical data management API which is similar to the physical one described earlier but has also support for meta data attributes, permissions related to files, multi-criteria file search related to meta data attributes, tags support. The logical data management API is used to organize users data and describe them with meta data attributes and tags. During the wow2green experiment all experiments results were stored and tagged in the logical file system to facilitate searching for finished experiments and concrete results.

3.3. Security Issues. Vine can solve different security issues while accessing different grid services. A significant interoperability problem results from the differences in the security models of systems such as UNICORE or GRIA (where authentication is based on standard X.509 certificates) and Globus-based Grids (authentication based on GSI [23]). The main difference between these systems is that basic versions of UNICORE and GRIA do not support dynamic delegation while GSI-compliant systems do. Currently, to overcome this problem, we allow a portal to authenticate on behalf of end-users. To do this, the portal signs a job with its own certificate and adds the distinguished name of the end-user before submitting the job to, for example, a GRIA site. In case of Unicore Vine supports GSI-enabled extension for Unicore gateway what enables to authenticate users by using proxy certificates. In case of gLite3 middleware voms proxy certificate are created and used—Vine can be configured to support different voms servers and allow users interactions from different VO without any problem. Vine support MyProxy [24] server to provide true single sign-on—during the user logging in to the portal proxy certificate is retrieved and used while accessing different middleware services. In case of the wow2gren solution the GRIA middleware was used. So as mentioned before portal common certificate was used to submit jobs (GSI is not supported). To distinguish users so called GRIA policy rules are used to set job and data file owners by using users' public certificates and their distinguished names. One of the requirements of the experiment was to simplify the security management issues. So user certificates are maintained by the Vine-based portal integrated with Certificate Authority—when user is registered for the first time, their certificate is generated and hold in the safe place on the server. Later proxy certificates are used to interact with the grid middleware. Common portal certificate is used to submit jobs to enable GRMS broker to manage them on behalf of users without the need to pass the private user's key certificate. Vine supports all aspects of security-related issues and processes. Account creation allows automated, user-friendly user registration at the portal and to underlying Grid middleware and third-party services used through the portal (registration modules are used to register new user accounts in external services). User authentication allows the user to login into the portal and be authenticated, login portlet is shipped together with Vine and could be embedded in different portal frameworks. Single-Sign-On (SSO) management allows automated SSO user authentication in various Grid middleware and services used through the portal. The user must have been previously registered in these external services. Authorization allows the administrator to access third-party authorization systems to change user permissions.

3.4. User Management. Common problem to solve while using different middleware infrastructures is user management and account provision. Vine is able to register users in different middleware infrastructures like gLite3, Unicore 6 or Globus GT4. Vine contains *registration modules* which allow register users on different resources. Currently the following are provided: *GridsphereRegistrationResource*—allows adding new users to Gridsphere portal, *LiferayRegistrationResource*—allows adding new users to Liferay portal, *GssCertificateRegistrationResource*—creates x509 certificate and key pairs for new users, *Gt4RegistrationResource*—creates user accounts on Globus Toolkit 4 host machine, *GriaRegistrationResource*—creates user accounts in Gria 5.3 Trade Account Service, *Unicore6RegistrationResource*—creates user accounts on Unicore 6 XUUDB, *DmsRegistrationResource*—registers users on Data Management Service, *VomsRegistrationResource*—registers users to Virtual Organization Membership Service. Modular architecture and plugin management system allows easily plugin other registration modules if desired and extend Vine with new functionality. In case of wow2green use case *GssCertificateRegistrationResource*, *GridsphereRegistrationResource* and *DmsRegistrationResource* were used to generate users' certificates, create portal accounts and logical file system DMS [25]. Vine also comes with bundled administrative web applications regarding user managamnet. One of them is *UserRegistrationApp*, which allows create requests for new accounts in Vine. Another is *UserManagementApp* where administrator has got option for deactivate user account and edit user's profile information.

3.5. Role Management. Vine also introduces role management mechanism to authorize users while taking different actions in the web applications. Roles are used in two scenarios—as roles at the portal level

and middleware stack level. Roles at the portal level allow administrators to set permissions to access different functionality in the web applications based on Vine. Administrator is able to set permissions for roles to different part of application and distinguish users regarding their possible scope of actions. Roles at the middleware stack are used to set access permissions regarding jobs and data files at the storage services. For the wow2green use case *GriaRoleResource*, *DmsRoleResource* were implemented. Thus the roles created in the portal are propagated to the middleware stack—in this case of GRIA. The end user interface allows share data for the selected roles. Such action is propagated to the middleware resources and could be used later while accessing remote storage by using given role name. Such approach makes permission management much simpler and does not multiply remote policies for every user of the system—it is enough to assign policy for some role and assign an user to the given role at the target system. Of course it is applicable only on middleware system which supports roles management like GRIA or DMS logical file system.

4. Towards production quality portals. Vine integrates with different portal frameworks and proper one should be used to provide production quality of the final solution.

FIG. 4.1. *Vine Toolkit integrated with Liferay portal framework.*

4.1. Liferay as a primary solution for the production quality portal. Vine was initially designed to work with Gridsphere portlet container. The reason for that was Gridsphere's open source nature and its target community—mainly scientists dealing with High Performance Computing problems. It was good starting point but after a few years it became obvious that concurrent solutions need to be taken into account as well. After analysis and evaluation of possible options a decision has been made to create a Liferay integration thread. Liferay is a modern approach to create complex web portals. It supports many standards for content presentation and management. It is integrated with various application containers, database systems and has modular structure with well designed API allowing creation of custom extensions and integration with third party software.

4.2. Vine Toolkit integration with Liferay. Vine is designed with its modular structure in mind. It has several places—hooks providing easy adaptation to the external environment. Integration with the portal acting as a portlet container is conducted on several levels. The most important are:

4.2.1. Build system.

- Flex sources compilation for the Liferay context path. Every flex component needs to establish communication with BlazeDS backend server. As for that a context path must be compiled into the specific component. Vine's build system takes care of that task and simply prepares a version of component specific to the destination portal environment.
- New deployment routines including modification of portal configuration. Vine Toolkit deployed in the application container acts as complex web application. Mainly due to its advanced class loaders preventing from the conflict of jars it has to be deployed in a specific way. Vine's build system is responsible for the proper jar distribution, replacing specific tokens in the deployment descriptors, altering portal layouts etc.
- New installers designed for Liferay portal. To shorten the learning curve for new users, new installers have been introduced. They are general examples of proper configuration both Vine Toolkit along with its basic services and Liferay environment. Thanks to that user is able to startup and test new portal without any advanced configuration actions.

4.2.2. Integration with Liferay services.

- Authentication/password policies. Integration with the parent authentication and password policies is a crucial task to provide synchronization of applications' state between portal and Vine. Appropriate plugins have been developed specifically for Liferay which provide seamless authentication of user against Liferay and Vine.
- User management. Vine has its own user registry to enable Vine's deployment to various destination environments. In the portal integration scenario Vine has to integrate and synchronize with the existing portal users. To accomplish that *RegistrationModule* plugins are used. *LiferayRegistrationModule* plugin is provided for adding/editing/deleting portal users.
- Applications management. Vine components are accessible as portlet applications from the Liferay perspective. They are fully manageable with no difference to the ordinary portlets. On top of that it is possible to put more control over Vine applications via Vine Toolkit administration interfaces. They provide advanced management actions for all Vine components, i. e. enabling/disabling certain features in specific applications, granting/revoking special permissions to them etc.
- JSR 168 compliant renderer for Vine components within Liferay portlets. Vine uses portlet component to render Flex components. In that way it preserves internal components integrity and make Vine applications independent from the external portal. As an output a html wrapper code is generated with a reference to the Vine Flex object inside.

4.3. Future plans for enhancing user experience using portal.

- Introducing web messaging based on BlazeDs/Flex technologies to greatly improve applications functionality.

Currently the primary way of communication in Vine GUI is remoting based on BlazeDs/Flex implementation. It provides efficient and responsive link between the client and server side. In Vine Toolkit the Model View Controller design pattern has been implemented enabling reuse of business logic with different presentation layers and leaving a place for future GUI mechanisms evolution. The next enhancement will be adding web messaging as an alternate way of communication between the server and client side. It enables server to trigger notification of state's change directly to the registered client interfaces. It will enable developers to design new components, for instance, a status monitoring applications with great ease. Until now they had to implement a querying mechanism on the client side to ask for the updated model constantly. With the web messaging a business logic on the server side will trigger notifications only if its state has changed. That will result in more efficient and sophisticated applications delivered to end users.

- Integration with Liferay Collaboration Suite package.

Liferay Collaboration Suite is a set of web applications, collaboration tools that come boundled with Liferay portal framework. The set of applications consists of blogs, message boards, instant messenger, shared calendar, address book, mail client, RSS client, Wikis, meta-tagging support. The main interest here is to idea of sharing experiments results form executed jobs and workflows by the Vine management engine and files stored in a portal file system with other portal users. Integration will be achieved by development of dedicated plugins which will communicate with Liferay API. This approach will enable all applications using File Manager with

seamless integration with Liferay Collaboration Suite with respect to security policies expressed by Liferay. First stage of integration, includes sharing data with wikis and message boards.

5. Next step—nanotechnology support portal. Currently our main focus is on the web application level scenarios and domain specific scientific and engineering applications. The idea is to prepare portal-based solution for the nanotechnology community to take advantage from nanotechnology related scientific application ran in a large Polish grid infrastructure under the Pl-Grid project umbrella. Initial plan is to start with the support for open source nanotechnology applications like Abinit and Fireball. The basic functionality foreseen is to simplify application every day usage by applying graphical interface, identify possible scientific use cases and propose different GUI layout. Another important thing is to prepare workflow definitions where the raw results will be transformed and prepared for further analysis—automation of many steps usually done manually will speed up work and allow to focus on the given problem. Advanced problems also require parameter sweep parameters what is especially useful in case of preparing big set of potential experiments varying only by some parameter value. The intermediate results from the given experiment and also from parameter swep steps will be used to prepare and show graphical plot of different physical values almost in a real time. End user will be able to control current execution and make possible decision to break the process for example. GMRS v3.0 meta-scheduler and broker plays here significant role as it supports workflows, parameter sweep and cross cluster job execution. Advanced data management features make possible to extract intermediate data during job execution. Extracted data will be used not only to prepare 2D plot display but also to prepare 3D visualization of the molecules structures. 3D visualization thread was started to prepare web based with latest web standards like HTML5 and WebGL. Portal solution will bring up the desktop application and strong collaboration environment for the whole domain specific community allowing to interact between geographically separated scientists.

6. Conclusions. In this paper we presented Vine Toolkit as universal framework to build complete web solutions to interact with different services and technologies (among others different grid middleware stacks) by means of uniform, easy API. Integration with production quality portal frameworks like Liferay opens new opportunities for creating production-ready portal environments directed for large communities. The generic architecture of the Vine Toolkit was presented. We also included some requirements and initial ideas for the wow2green business experiment as illustration of Vine application in a real, practical use case. Vine allows submit jobs to different middleware stacks through a uniform user interface. In similar way uniform API for workflow management, data management, user authentication, role and user management are provided. Resource based model allows straightforward service support and configuration by uniform way regardless of the target technology. Security issues which always occur while accessing different middleware stacks are solved by using Vine registration, authorization modules and proxy certificates support. Role management mechanism allows administrators and end users to easily manage permissions in the applications and easy grant permissions to data files on the storage server if applicable. Beside simple jobs, whole workflows could be managed by the uniform workflow interface. Currently only the Kepler engine is supported but it could be easily extended by adding another workflow engine plugin. Vine Toolkit could be used as framework for building advanced web applications for example with use of Adobe Flex technology as well as business logic provider for advanced web service solutions. Prospective application scenario related with nanotechnology science will be beneficial for the scientists who want to focus on the concrete problems and large grid infrastructure exploitation. The on-going project will enrich our experience regarding generic scientific application support.

Acknowledgments. We are pleased to acknowledge support from the EU funded OMII-Europe and BEinGRID projects and polish infrastructural project PL-Grid.

REFERENCES

[1] Vine Toolkit web site `http://vinetoolkit.org/`
[2] OMII-Europe project `http://www.omii-europe.com/`
[3] BEinGRID project `http://www.beingrid.eu/`
[4] PL-Grid project `http://www.plgrid.pl/en`
[5] Wow2green business experiment `http://www.beingrid.eu/wow2green.html`, `http://www.it-tude.com/wow2green_sol.html`
[6] P-GRADE `http://www.p-grade.hu/`
[7] G-POD `http://gpod.eo.esa.int/index.asp`
[8] myExperiment.org `http://www.myexperiment.org/`, `http://wiki.myexperiment.org/index.php/Main_Page`

[9] EnginFrame http://www.enginframe.com/
[10] Flowify http://www.flowify.org/
[11] Kepler https://kepler-project.org/
[12] GRMS http://www.gridge.org/content/view/18/99/
[13] GRIA http://www.gria.org/about-gria/overview
[14] Adobe Flex http://www.adobe.com/products/flex/
[15] Adobe BlazeDS http://opensource.adobe.com/wiki/display/blazeds/BlazeDS/
[16] MoML language specification http://ptolemy.eecs.berkeley.edu/publications/papers/00/moml/moml_erl_memo.pdf
[17] Unicore 6 http://www.unicore.eu/
[18] JSDL http://www.ogf.org/documents/GFD.56.pdf
[19] Open Grid Forum (OGF) http://www.gridforum.org/
[20] gLite 3.x http://glite.web.cern.ch/glite/default.asp
[21] Globus GT4 http://www.globus.org/toolkit/
[22] QosCosGrid http://node2.qoscosgrid.man.poznan.pl/gridsphere/gridsphere
[23] Grid Security Infrastructure (GSI) http://www.globus.org/security/overview.html
[24] MyProxy http://grid.ncsa.illinois.edu/myproxy/
[25] Data Management System (DMS) http://dms.progress.psnc.pl/
[26] Liferay http://www.liferay.com/
[27] Dawid Szejnfeld, Piotr Domagalski, Piotr Dziubecki, Piotr Kopta, Michal Krysinski, Tomasz Kuczynski, Krzysztof Kurowski, Bogdan Ludwiczak, Jaroslaw Nabrzyski, Tomasz Piontek, Dominik Tarnawczyk, Krzysztof Witkowski, Malgorzata Wolniewicz, *Vine Toolkit—Grid-enabled Portal Solution for Community Driven Computing Workflows with Meta-scheduling Capabilities*, PPAM 2009.
[28] GridSphere http://www.gridsphere.org/

Edited by: Marcin Paprzycki
Received: April 10, 2010
Accepted: June 21, 2010