LEAD-VGrADS Integration

Introduction to LEAD SOA

LEAD infrastructure is being built on the foundation of the concept WOORDS -- Workflow Orchestration for On-Demand, Real-Time, Dynamically-Adaptive Systems. WOORDS provides analysis tools, forecast models, dynamically adaptive, on-demand, grid-enabled systems that can a) change configuration rapidly and automatically in response to weather; b) continually be steered by new data; c) respond to decision-driven inputs from users; d) initiate other processes automatically; and e) steer remote observing technologies to optimize data collection for the problem at hand.

To achieve the LEAD vision of enabling users and technologies to dynamically interact, existing weather tools are being wrapped into web services, in order to construct a Service Oriented Architecture (SOA), in which service components can be dynamically connected and reconfigured. A Grid portal is being developed to serve in the top tier of this SOA as a client to the services exposed in the LEAD system.

The LEAD SOA (Figure 1) is realized as five distinct yet highly interconnected layers. The bottom layer represents resources consisting of computation as well as application and data resources distributed throughout the LEAD Grid and elsewhere. At the next level up are web services that provide access to capabilities including those found in Globus (e.g., the Grid resource allocation manager (GRAM) and the Grid file transfer service, GridFTP), as well as services for accessing weather data like LDM, OPeNDAP and data access services such as the Replica Locater Service (RLS) and Open Grid Service Architecture Data Access and Integration service (OGSA-DAI).

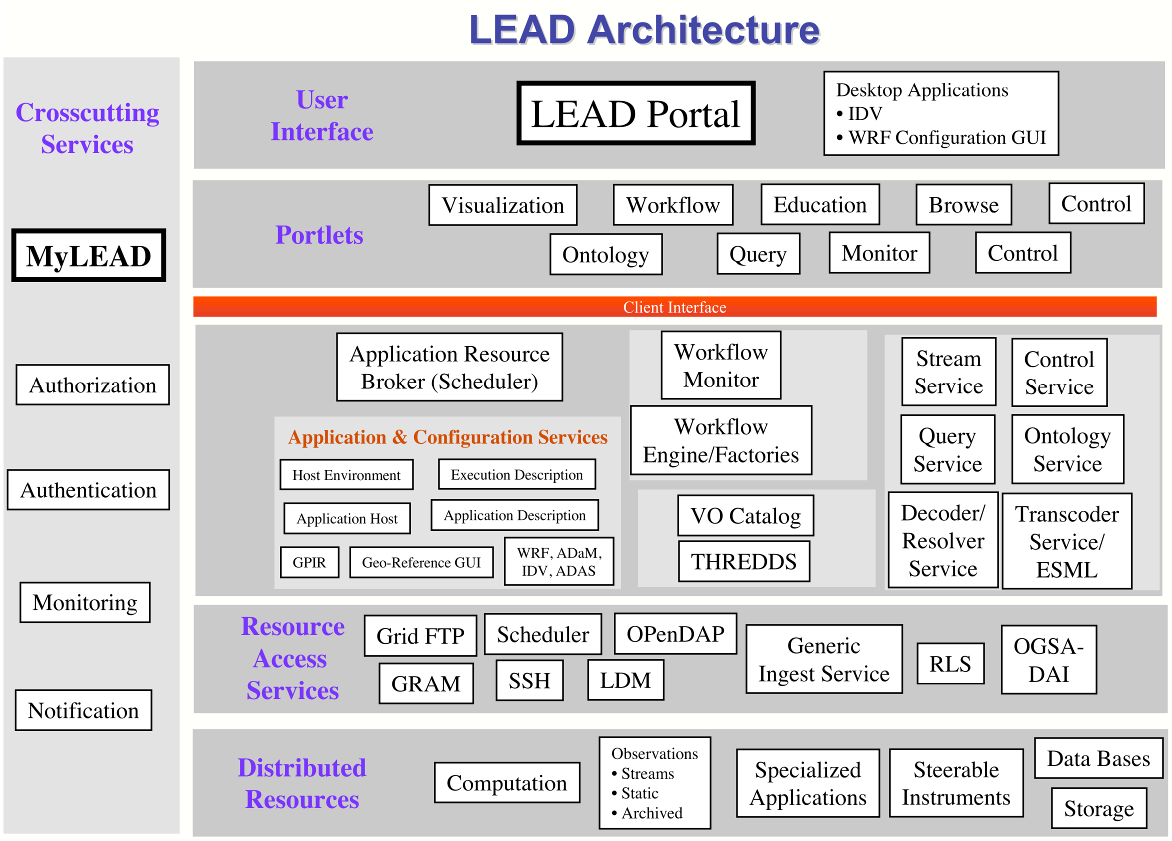

Figure 1: LEAD Architecture

Wide varieties of configuration and execution services compose the next layer and represent services invoked by LEAD workflows. They are divided into four principal groups, the first being the application and configuration service that manages the deployment and execution of fundamental user applications such as the Weather Research and Forecast (WRF) model – which is a next-generation limited-area atmospheric prediction and simulation model that runs on single or multiple processors at grid spacings ranging from meters to hundreds of kilometers, the ARPS Data Assimilation System (ADAS) – which is a sophisticated tool for data quality control and assimilation including preparation of model initial conditions, the Algorithm Development and Mining (ADAM) -- which is a powerful suite of tools for mining observational data, assimilated data sets and model output. For each of these, additional services are needed to track deployment and execution environment requirements to enable dynamic staging and execution on any of the available host systems. Catalog services control the manner in which a user discovers data for use in experiments via a Resource Catalog (RC). They do so by indexing the contents of THREDDS catalogs, which store pointers to a wide variety of data.

Finally a host of data services are used to search for and apply transformations to data products. An ontology service resolves higher-level atmospheric concepts to specific naming schemes used in the various data services, and decoder and interchange services, such as the Earth System Markup Language (ESML), transform data from one form to another. Stream services manage live data streams such as those generated by the NEXRAD Doppler radar network.

Several services are used within all layers of the LEAD SOA and are referred to as crosscutting services. One such service is the notification service, which lies at the heart of dynamic workflow orchestration. Each service is able to publish notifications and any service or client can subscribe to receive them. This strategy is based upon the WS-Eventing standard, where notifications signal the completion of tasks, the failure of a job or an immediate command from a user. Another critical component is the monitoring service, which provides, among other things, mechanisms to ensure that desired tasks are completed by the specified deadline – an especially important issue in weather research.

User's metadata catalog called myLEAD – which is a flexible personalized data management tool to store metadata associated with data products generated and used in the course of scientific investigations and education activities, ties multiple components of SOA together. As an experiment runs, it generates data that are stored on the LEAD Grid and cataloged to the user's myLEAD catalog. Notification messages generated during the course of workflow execution also are written to metadata and stored on behalf of a user. A user accesses metadata about the products used during or generated by an investigation through a set of metadata catalog-specific user interfaces built into the LEAD Portal. Through these interfaces the user can browse holdings, search for products based on rich meteorological search criteria, publish products to broader groups or to the public, snapshot an experiment for archiving, or upload text or notes to augment the experiment holdings. Authentication and authorization are handled by specialized services based upon grid standards.

Finally, at the top level of the architecture is the user interface, which consists of the LEAD web portal and a collection of "service-aware" desktop tools. The portal is a container for user interfaces, called portlets that provide access to individual services. When a user logs into the portal, his or her grid authentication and authorization credentials are loaded automatically. Each portlet can use these certificates to access individual services on behalf of the user, thus allowing users to command the portal to serve as his or her proxy for composing and executing workflows on back-end resources.

LEAD Users

Initial LEAD users have been divided into four categories: Category I – Modelers/Application Scientists who research on improving model capabilities and these users intend to change code very often. Category II users run models with different set of input conditions and mostly are researches trying to improve weather predictions and operational scientists performing periodic weather forecasts. Category III users are Educational users who will perform simulations to understand and learn atmospheric phenomenon. Category IV users are casual browsers who will visualize pre-run workflows.

LEAD Worklfows

LEAD workflow tools allow users to construct and schedule execution task graphs with data sources drawn from archived as well as real-time sensor streams and which adapts to the changing weather is shown in the figure. The workflow is scheduled by monitoring and performance estimations, which helps in estimating resource behavior to ensure timely completion of tasks. LEAD workflows respond to the dynamic behavior of the computational and grid resources in order to meet the requirement of "faster than real time" prediction. As the workflow progresses it must allocate new resources for ensemble runs of the WRF simulation system. This will require an adaptive scheduler and the ability to monitor the execution of each component of the workflow. The monitoring infrastructure will analyze monitoring output from the application, workflow and resources. Performance and reliability metrics will be used to establish a simple performance contract for a workflow that will be dynamically monitored and based on changing conditions. The execution of on-demand workflow will be on set of processors provided by the **Resource Provisioning Service** and will begin by a specified wall clock time and allow the computational task to finish within a specified time interval.

LEAD workflows have the unique characteristics of large data transfer, real-time data streams, huge computational demands, strict deadlines for workflow completion, need to steer external radars to collect new data, respond to weather phenomena. So a timely co-ordination of different types of resources to meet soft real-time guarantees is required. Co-ordination across the layers to allocate, monitor and adapt in real-time while meeting strict performance and reliability guarantees and co-allocation of real-time data streams and computational resources is required. Multi-layered monitoring and adaptation to support the dynamic nature of workflows is also required.

Use-Case Workflow for LEAD-VGrADS Demo at SC-06

The use-case workflow analyzes terrain and surface data, interpolates the current weather conditions, and launches a forecast model to predict future weather. Visualization services are launched using which user can manually see the 2D and 3D animations of workflow output. Description of services in the example workflow shown in Figure 2:

Service 1: Terrain Preprocessor - Performs analysis of terrain data and generates a terrain file by interpolating the data to the ARPS forecasting grid.

Service 2: WRF Static Data Preprocessor - Prepares the surface characteristic data set for use in ARPS and generates surface characteristic files with soil types, vegetation types, leaf area index and surface roughness.

Service 3,4: 3D Model Data Interpolator -- Extracts and interpolates pertinent fields from a National Weather Service model forecast dataset to an ARPS forecasting grid to provide an ADAS analysis background or initial conditions and lateral boundary conditions for an WRF forecast.

Service 5: 88D Radar Remapper - Converts raw NEXRAD Level II radar data in polar coordinates to Cartesian coordinates and remaps the data to the ARPS forecasting grid.

Service 6: NIDS Radar Remapper - Converts WSR-88D Level-III raw velocity and reflectivity data and remaps it onto a sigma-Z Cartesian ARPS forecasting grid.

Service 7: Satellite Data Remapper - Remaps observed McIDAS GVAR AREA satellite data from the satellite-observed pixels to the ARPS forecasting grid.

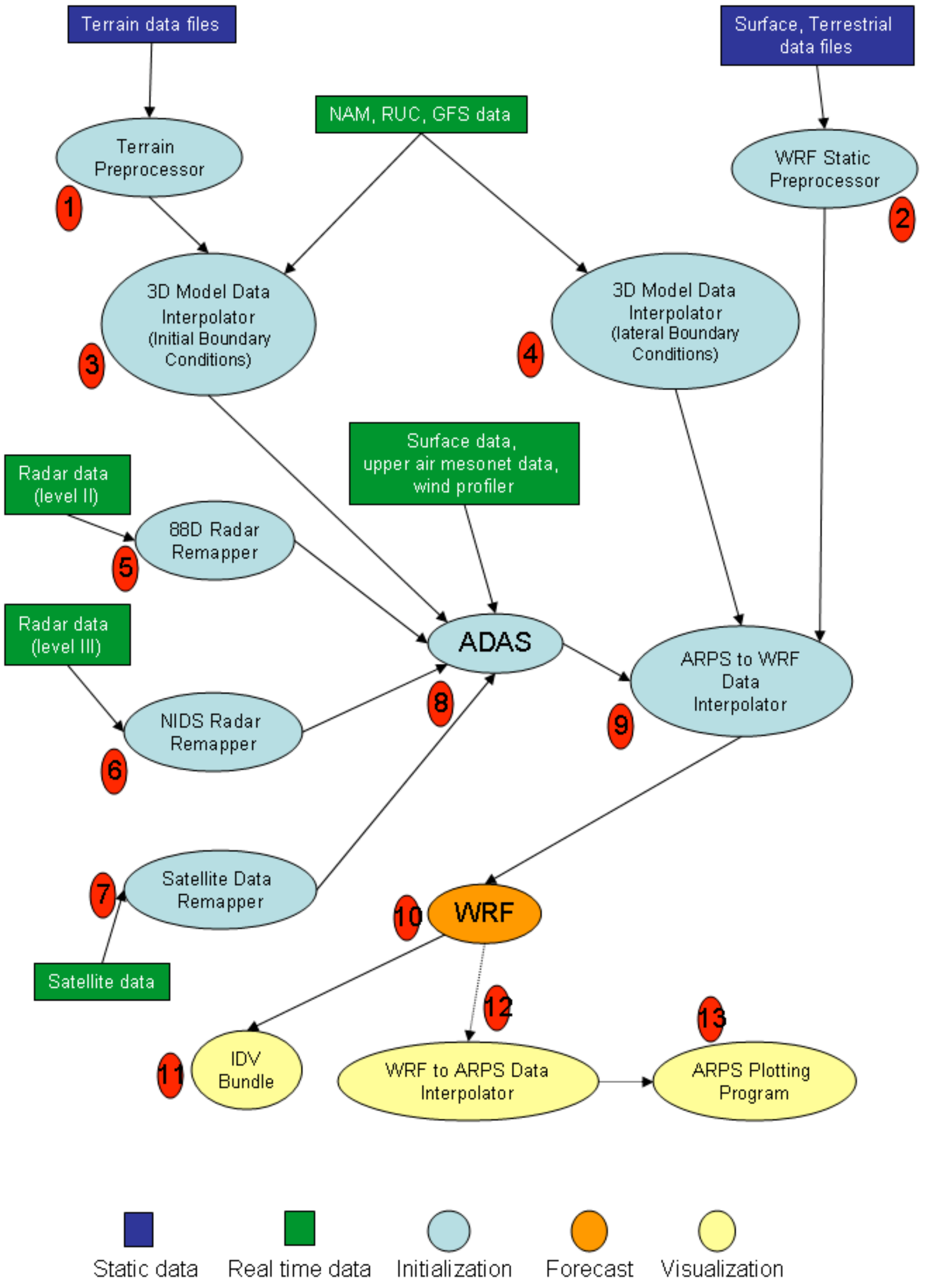

Figure 2: LEAD-VGrADS Demo Workflow

Service 8: ADAS-ARPS Data Analysis System - Generates 3D gridded analysis of the current atmosphere by combining the observed information from NEXRAD radars, wind profilers, satellites, surface observation networks and aircrafts with a background field created by external model data interpolator.

Service 9: ARPS-to-WRF Data Interpolator -- Ingests data files in ARPS history format and generates WRF input and lateral boundary files.

Service 10: WRF -- Performs storm, mesoscale and synoptic weather prediction by a non-hydrostatic, limited area model to study convection, baro-clinic waves, boundary layer turbulence, real-time weather phenomena.

Service 11: Integrated Data Viewer (IDV) is a widely used desktop application for visualizing, in an integrated manner, a broad array of multi-dimensional geophysical data.

Service 12: WRF-to-ARPS Data Converter -- Converts WRF output to ARPS format to feed into ARPS post processing components.

Service 13: ARPS Plotting Program -- Generates contour and vector plots of 2D cross sections and vertical profiles. The graphical output is in Postscript format.

Most of the large meteorological applications have extremely complex and large parameter sets which are encoded as FORTRAN name lists. Some services (e.g., ADAS, WRF) may have several hundred parameters, but most users may wish to modify only one or two dozen. Depending on the level of the user, a different subset of parameters may need to be modified. To turn this process into a service, the input to the service should be a document which describes changes wanted by the user relative to default values.

The FORTRAN programs use namelist files to specify run-time configurations to the application. Namelists provide a way of selecting different options without the need of recompiling the application code. Majority of LEAD users and atmospheric community are category II users who will not changing the source code of applications but will be running the same executables in different modes by changing the configuration parameters in FORTRAN namelist input files. These changes force the changing in input observational data and resource requirements before run time.

Architecture of LEAD Workflow Composition, Configuration, Scheduling and Execution

Task A: Applications are wrapped as web services

LEAD applications are first “wrapped” as web services. “Wrapping” an application as a web service refers to the process of creating an additional web service layer on top of the application. The web service layer is an interface to the underlying application and is often referred to as an application service. All clients and end-users interact with the application through its application service. When an application service is invoked with a given set of input parameters, it invokes the underlying application with those input parameters and returns the output results.

A Generic Service Toolkit is used to wrap applications as application services. To do so, first they describe the application service using an XML document called the ServiceMap document. The ServiceMap document is not a WSDL. It is a higher level language than WSDL for describing the input and output messages of an application service (which are internally and automatically mapped to input and output parameters of its application), the security policies and the soft-state lifetime management policies of the application service. Scientists then use the Generic Service Toolkit to register the ServiceMap document with a well known Registry service so that it can be retrieved later to automatically create application service instances. The registration process also automatically creates and registers with the registry, another XML document that we call an abstract WSDL (AWSDL). An abstract WSDL is a WSDL document that specifies only the interface of a web service without actually binding it to a specific instance of that web service. In other words, while an AWSDL describes the interface of a web service, its WSDL describes a specific instance of that web service.

Task B : Composing a workflow

User browses though the application services registered to the resource catalog and composes a workflow using a graphical workflow composer tool.

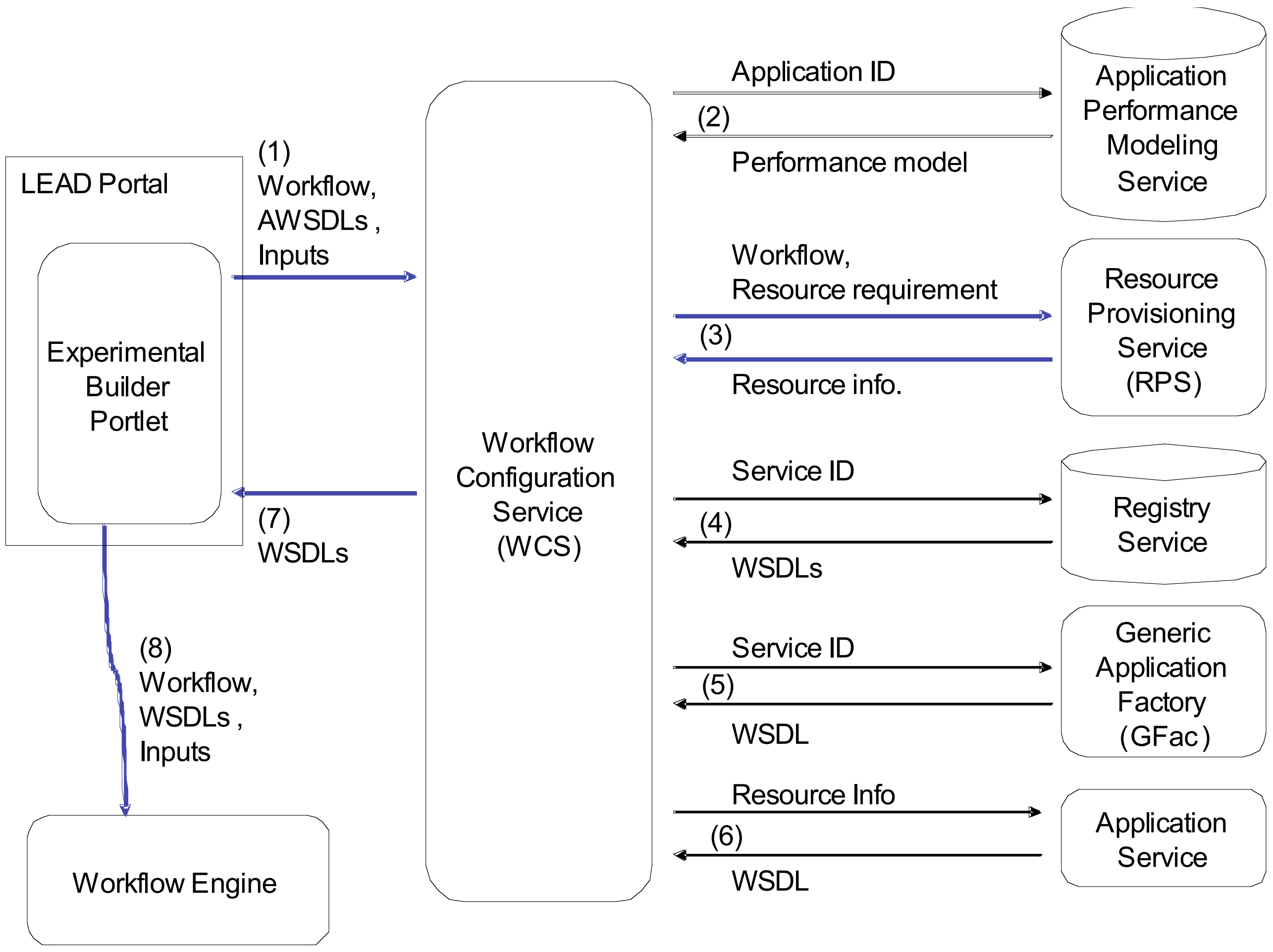
LEAD Portal
Experimental Builder Portlet
(1) Workflow, AWSDLs , Inputs
(7) WSDLs
(8) Workflow, WSDLs , Inputs
Workflow Engine
Workflow Configuration Service (WCS)
Application ID
(2)
Performance model
Application Performance Modeling Service
Workflow, Resource requirement
(3)
Resource info.
Resource Provisioning Service (RPS)
Service ID
(4)
WSDLs
Registry Service
Service ID
(5)
WSDL
Generic Application Factory (GFac)
Resource Info
(6)
WSDL
Application Service

Figure 3 : Workflow Configuration and Execution Architecture

TASK C: Configuring the Workflow

After composing a workflow, scientists need to provide the required input parameters and data to the workflow before executing it. Most large meteorological workflows have a large number of input parameters; weather forecasting workflows involving ADAS and WRF have several hundred input parameters. However, most users specify not more than a few dozen of these. Also, different categories of users may wish to specify different sets of input parameters. The LEAD portal provides a rich graphical user interface that allows different categories of users to easily specify different sets of input parameters and data to the workflow.

After the input parameters and data to a workflow are specified, Resources Requirements are to be estimated and the workflow needs to be scheduled on the available resources. These steps are executed in a way that is completely transparent to the users and are described below.

TASK D: Determining Resource Requirements

Determining the resource requirements for an application is a complex task and depends on several factors including but not limited to the input parameters and data to the application. The workflow configuration service will contact the Performance Modeling Service with the workflow description and user inputs to obtain the resource requirements. This performance model does not exist today.

TASK E: Resource Provisioning

Once the resource requirements for all the applications in the workflow are determined, the resources for running each application need to be provisioned. Information about the provisioned resources for running an application needs to be conveyed to its application service instance.

In Figure 3 the Experiment Builder portlet in the LEAD portal provides the Workflow Configuration service (WCS) with the workflow along with its input parameters and data sets. In step 2, the WCS first contacts an “Application Performance Modeling” service to obtain a performance model for each application in the workflow. The WCS then determines the resource requirements for each application based on the input parameters and data sets provided by the user. Once the resource requirements for each application in the workflow has been established, the WCS requests the “Resource Provisioning” service (RPS) in step 3 to get the required resources to run the workflow. Based on the availability of resources and the resource requirements provided by the WCS, the RPS provisions and reserves resources for each application in the workflow. The WCS can then “configure” the application service instances in the workflow if they are running. To find out if an instance of an application service is running or not, in step 4 the WCS tries to find its WSDL in the Registry service. If a WSDL is found in the Registry, WCS uses it to invoke the application service instance and “configures” it. If no WSDL is found, WCS requests the “Generic Application Factory” (GFac) to create an instance of the application service in step 5. GFac then returns the WSDL for the newly created application service instance to the WCS, which then invokes the application service instance and “configures” it in step 6. After all the application service instances in the workflow have been created and configured, in step 7 the WCS returns their WSDLs to the “Experiment Builder” portlet, which then requests the Workflow Engine to execute the workflow in step 8.