GSiB Visual Environment for Web Service Composition and Enactment

Yan Huang and Qifeng Huang
School of Computer Science, Cardiff University,
5 The Parade, Roath Cardiff CF24 3AA, United Kingdom
{yan.huang, q.huang}@cs.cardiff.ac.uk

Abstract
This paper presents VSCE, the Visual Service Composition Environment, which is one of the main visual components provided by the GSiB (Grid-Service-in-a-Box) project. VSCE uses SWFL as its main workflow language and employs SWFL workflow engines to provide an enactment environment for the SWFL jobs created in VSCE. The employer and employee relationship between a VSCE client and a workflow engine is discussed. Other features of VSCE, including support for static parallelism and multiple workflow language, are also discussed.

Key words: Workflow, Web services, composition, parallelism, XML language.

1. Introduction to GSiB

GSiB provides an easy-to-use graphical user interface for both service providers and clients. It consists of two GUI packages: the Service Providers GUI (SP-GUI) package and the Service Client GUI (SC-GUI) package. The SP-GUI package contains a variety of graphical interfaces for service providers to easily and quickly create services from existing legacy software and libraries, to compose complex services from existing services, and to update, monitor and manage services. The SC-GUI package presents interfaces for a service client to browse and query services, create applications composed of web services, submit jobs, monitor the execution of jobs, and retrieve and display results. [1]

GSiB is designed to function in conjunction with a WS-compliant Grid system and similar web service oriented systems. The service we refer to generally in GSiB is any service that is consistent with Web Services standards. The Web Service composition language that GSiB mainly supports is SWFL. SWFL is a service workflow language which is used to specify a web service composite application by describing the graphical structure of its workflow model [3]. The corresponding SWFL workflow engine provides an execution environment for web services composite applications described in SWFL. It also works as a host for composite services (i.e., services composed of other services). Figure 1 illustrates the main functionalities provided by GSiB.

2. Visual Service Composition Environment

The visual service composition environment (VSCE) of GSiB allows users to build a web service application by simply drawing the graph of the workflow model. The SWFL language is a graph-oriented language in which each flow model is viewed as a directed graph. The nodes of the graph are services and the edges of the graph represent the data and control flows...
The development of VSCE is simplified by taking good advantage of the consistency between the graphical approach used in SWFL and the graphical structure of a workflow model.

Java Swing, Java2D and JAXP and JAXB are used for XML processing and binding.

A graphical flow model created on a VSCE canvas can be directly converted and stored as a SWFL document, and a workflow graph can be repainted onto a VSCE canvas directly from a SWFL document. This also simplifies the task of supporting the parallel processing of SWFL jobs. Because an SWFL job is specified by describing the graphical structure of its flow model, according to the data dependencies specified by data links, a workflow graph can be partitioned into sub-graphs and each sub-graph can work as an individual workflow job to be dispatched to different job processing servers [3] to be executed in parallel. To run an SWFL application, a user normally can use a default workflow engine or specify a particular workflow engine. The job is then dispatched to the engine to be executed and the result is either sent back to the VSCE to be displayed or is stored to a previously specified location to be collected later. Figure 2 displaces the graphical window of VSCE.

Figure 2. GSIB Visual Composition Environment

3. VSCE and SWFL Engines

VSCE is not limited to use only one SWFL engine. In order to support parallel job processing, multiple SWFL engines are allowed to work for a particular VSCE at the same time. There is no master and slave relationship between a VSCE and an SWFL engine but rather an employer and employee relationship. Both VSCE and SWFL are autonomic and self-organizing. A VSCE always works as an employer who employs SWFL engines to work for the SWFL jobs created by using VSCE tools. A SWFL engine can work for multiple VSCEs as well as independent clients.

An SWFL engine is presented as a Web Service, so that it normally is discovered from a UDDI registry server. To employ an SWFL engine to work for a particular VSCE, a contract needs to be agreed by both the employer and employee. The content of a contract includes the length of the employment and communication-related agreements.

Since a VSCE can employ multiple SWFL engines at the same time, there is an issue of which workflow engine an SWFL job is supposed to be submitted to. It should be up to the owner of the job to choose an engine to process his job, but normally VSCE allows the owner to simply submit his job to a default engine which is dynamically chosen among the available engines according to some performance criteria such as the workload of an engine and the communication bandwidth between the engine and the VSCE client.

4. Static Parallelism Support

SWFL and its workflow engine are designed mainly for scientific applications composed from web services, so that efforts are made to support parallelism. The support for parallelism in a workflow engine is provided at three levels: service-level parallelism, flow-level parallelism and message-passing level parallelism. Service-level parallelism happens in a single workflow instance. Instead of being executed in a sequential order, based on the data dependency, the task nodes of a workflow, which are normally invocations of individual services, may be executed in parallel to achieve a shorter execution time. Flow-level parallelism usually involves multiple job processing servers. A workflow application is partitioned into multiple sub-flows according to the data dependency in the workflow graph, and each sub-flow is dispatched to a different engine to be executed in parallel. Static and dynamic flow-level parallelism can be identified. Dynamic flow-level parallelism is supported dynamically in an SWFL engine. After receiving an SWFL job which requires parallel processing, the workflow engine partitions the job dynamically into multiple sub-jobs based on
the available job processing servers, and then send the sub-jobs to multiple job processor servers to be executed in parallel. Static workflow-level parallelism is supported by VSCE. VSCE provides a job partition mechanism for the user to partition his job either manually or automatically using the tool provided by the visual environment. After a job partition is done, the user is allowed to modify the created sub-jobs, allocate workflow engines for the sub-jobs and initiate the execution. Message-passing level parallelism support involves MPFL—a newly emerged XML language for specifying advanced scientific parallel applications [4]. The work on MPFL is still at its early stage, detailed discussion will be in a separated paper.

5. Multiple Language Support

As one of its many objectives, GSiB is to support multiple workflow languages. IBM's Business Process Execution Language BPEL [1] is one of the third-party languages GSiB is supports. GSiB uses SWFL2BPEL and BPEL2SWFL to convert between the languages. Because of the graphical approach used in SWFL, SWFL can act effectively as an intermediate language for the conversion of third party languages. The advantages of the support for multiple language includes: third party applications can be built, displayed and executed in the GSiB environment, the third party applications can be run in parallel via the GSiB interface and by using SWFL workflow engines, and the SWFL applications can be run in a third party workflow engine.

6. Conclusions

As one of the main components provided by GSiB, VSCE, provides an easy-to-use graphical interface for users to build and execute their web services composite applications. By using autonomic, self-organizing SWFL workflow engines, VSCE supports a decentralized, loosely-coupled workflow execution environment which is consistent with the Web Service framework. The support for static parallelism allows users to design and control the parallel processing of their web service composite jobs. Third party language support in GSiB allows third party users to take full advantage of GSiB’s service composition tools and its parallel execution environment.


