EGEE/SA3 – интеграция, тестирование и сертификация

В.В. Галактионов

Лаборатория информационных технологий, ОИЯИ, Дубна

В первой фазе проекта **EGEE -** интеграция, тестирование и сертификация компонент нового middleware **gLite** в проекте LCG выполнялись в рамках направления **JRA1**. Во второй фазе для них создано отдельное сервисное направление: **SA3**. Задача SA3 – руководство разработкой выпусков промежуточного программного обеспечения, готовых к развертыванию и сопровожденных документацией. Координация работ по этому направлению выполняется в группе SA3 (руководитель Keeble Oliver) отдела GD (Grid Deploiment) подразделения IT (Information Technology) в ЦЕРНЕ [1, 2, 3].

Представленные работы выполнялись в рамках трехстороннего соглашения между ОИЯИ, RDIG(Россия) и ЦЕРНом.

В группе SA3 создан технический и программный сертификационный комплекс-полигон (**CTB**, certification testbed) для исследования и сертификации новых версий компонент gLite [4]. По известным причинам доступ к ним извне ЦЕРНа ограничен, поэтому значительная часть работ выполнялась в ЦЕРНе во время командировок. Полигон содержит практически все необходимые как программные, так и технические компоненты LCG для выполнения тестирования (CE, SE, BDII, VOMS и др). Территориально отдельные компоненты полигона, такие как DCACHE, MPI располагаются вне ЦЕРНа.

Службы CTB

Установлены определенные правила для разработки тестов, их выполнения, документирования и хранения, для выполнения которых существуют специальные службы:

Мониторинговая служба. Подготовленные тесты должны включаться в мониторинговую систему для периодического их выполнения. В соответствии с этим и должны строиться тестирующие программы. До последнего времени в качестве такой системы использовался **SAM** (Service Availability Monitoring) [16]; в настоящее время ведутся работы по использованию для этой цели системы NAGIOS [5, 6]. Рассматривается также вопрос об использовании системы ETICS [7]. Комплекс программных тестов для отдельной тестируемой компоненты gLite называется **сенсором.** Пример сенсоров: **VOMS**, **LFC**, **UI**. В настоящее время подготовка тестов в SAM-формате прекращена. Для выполнения уже разработанных в этом формате сертификационных тестов в группе используется разработанный автором SAM-имитатор, который ранее предназначался лишь для автономной отладки сенсорных программ.

Служба документирования. Установлены несколько служб документирования: в формате WIKI [8, 9], а также служба сопровождения заданий и отчетов в портале SAVANNAH [10, 11, 12].

Служба архивирования. Для хранения разработанных программ и данным к ним применяется служба **CVS** (Concurrent Versions System) [13, 14], в дальнейшем предполагается миграция репозиториев CVS к **CVN** [15].

Пул виртуальных машин. Для выполнения специфических сертификационных работ применяется механизм виртуальных машин [17], на которых может устанавливаться новое или модифицированное обеспечение для gLite, которое отсутствует в стандартном обеспечении CTB. Каждая из виртуалных машин после актуализации может получить соответствующие хост-сертификаты.

Пул виртуальных пользователей (fake users). Этот пул используется для некоторых задач тестирования, требующих выполнение операций с пользователями ГРИДа, таких как например, операций с виртуальными организациями. Каждый из виртуальных пользователей, а их установлено около 200, имеет действительные грид-сертификаты, и они закреплены за определенными виртуальными организациями типа **DTEAM, TEST**.

Требования к тестирующим программам

Единых типовых правил для разработки тестовых программ – нет. Сценарии выполнения тестов каждый раз составляются индивидуально для каждого сенсора. Существуют лишь общие рекомендации, такие как: форматы передачи параметров в мониторинговой системе (SAM или Nagios), проверка функциональности тестов (правильность выполнения заданной операции или выдачи адекватной диагностики в случае невозможности ее выполнения). Были попытки создания предварительных проектов сценариев для сенсоров (описаний **test-case**), но это достаточно трудоемкий процесс, сравнимый с написанием самих тестов, и, как оказалось на практике, не всегда достаточно полным.

Автором выполнен ряд работ по подготовке сертификационных тестов для различных ком-

понентов в СТВ:

- Security access - система безопасности,
- Data Management – управление данными,
- Job Management – управление задачами.
- Test monitoring – мониторирование тестирования.

Система безопасности

Обеспечение безопасности для распределенных вычислительных систем – одна из важнейших задач. В Гриде она решается применением персональных сертификатов и закреплением каждого пользователя в определенной виртуальной организации (в одной или нескольких). Для этого используются серверы **VOMS** (Virtual Organization Membership Services). Число VOMS-серверов неограниченно. Для тестирования были выбраны основные операции с VOMS-сервером:

- **voms-admin** – для администрирования виртуальных организаций
- **Checking a Certificate** – пользовательские операции для ведения CA-сертификатов и proxy-сертификатов

voms-admin выполняет операции типа **creaqe-delete-list** для основных параметров виртуальных организаций (**users** (пользователей), **group** (групп пользователей), **Role** (ролей), **Attribute** (атрибутов), **ACL**-доступа). Полномочия для выполнения этой операции принадлежат администратору данной виртуальной организации.

Checking a Certificate – группа операций (**grid-proxy-init, grid-proxy-info, grid-proxy-destroy, grid-cert-info**) выполняемых самим грид-пользователем для различного типа манипуляций с персональным CA-сертификатом. С введением новейших версий программного обеспечения для работы с виртуальными организациями появились и разновидности команд (**voms-proxy-init, glite-voms-proxy-init, voms-proxy-info, glite-voms-proxy-info** и т.д). Каждая из перечисленных выше операций имеет большое количество параметров, определяющих (уточняющих) конкретику операции, для каждой из них готовится отдельный тест. Общее же количество подготовленных автором программных единиц – около 100. Программы написаны на языке BASH в формате для мониторинговой системы SAM. Документация к ним и исходные тексты размещены в [18, 19].

Управление данными

Чрезвычайно большие потоки данных в распределенной грид-системе требуют решения упорядочения этих данных. В настоящее время это решается созданием каталогов файлов типа **LFC** (LCG File Catalogue). Как правило, каталог этого типа устанавливается единым для виртуальной организации на соответствующем **LFC-сервере**, имя машины которого задается в глобальной переменной **LFC_HOST**. Тип каталога задается в переменной **LFC_CATALOG_TYPE**. В настоящее время тип каталога LFC в указанной переменной установлен по умолчанию. Определить имя машин с LFC-каталогом для вашей виртуальной организации можно командой, например:

lcg-infosites —vo dteam lfc

LFC-каталог представляет информацию о грид-файлах в формате, близком для операционной системы UNIX – логические имена файлов, тип доступа к файлам и др. Имеется группа команд для манипуляций с каталогом и файлами: **lfc-***. Например, коианды**: lfc-chmod, lfc-chown, lfc-ls, lfc-mkdir, lfc-rename** и др. Было подготовлено несколько вариантов тестовых программ. Во-первых, для различных мониторинговаых систем (SAM, Nagios, standalone mode). Во-вторых, для трех групп команд: сервисов для, **Python API,** непосредственно реализующих операции с каталогом, операций, выполняемых в режиме **CLI** (Command language Interface), а также набора тестов, собранных от разных исполнителей. Тесты для сервисов Python API (27 программ) были написаны сотрудником группы Robert Harakaly. Автором проведена адаптация этих программ для мониторинговых систем SAM и Nagios.

Всего было подготовлено свыше 50 программ. Документация к ним и исходные тексты размещены в [20, 21].

Управление задачами

Для сертификации представлен новый тип задач в LCG – задачи для выполнения параллельных вычислений в стандарте **MPI** (Message Passing Interface). Стандарты для включения задач этого типа разработаны рабочей группой EGEE (MPI working group) [24, 25]. Концепция MPI заключается в запуске на различных процессорах единой копии задачи. При этом каждый из процессоров “знает” свой идентификационный номер и в зависимости от этого выполняет свою часть параллельных вычислений. Процессоры, используя MPI-библиотеки, могут обмениваться информацией в процессе вычислений, в частности для сборки результатов вычислений. Для применения задач этого типа устанавливается в грид-сайте специальный **CE** (Computing Element), компонента **glite/MPI** и стандартные MPI-компоненты – библиотеки и трансляторы для выбранных языков программирования. Как правило, это языки C, C++, Fortran. В настоя-

щее время в glite/MPI реально существуют три реализации (implementation) MPI-2 – OpenMPI, MPICH и MPICH2. С точки зрения пользователя основу glite/MPI составляют три BASH script-программы MPI: **mpi-start** (для задания параметров) и 2 программы выходов mpi-hooks (названия условные), которые запускаются до и после компиляции исходной MPI-программы. Для системы управления задачами в **gLite WMS** (Workload Management System) задачи этого типа являются обычными грид-задачами, для них используется стандартный набор WMS gLite операций: **glite-wms-job-submit, glite-wms-job-status,** и **glite-wms-job-output.** Тип и параметры MPI-задачи задаются в стандартном **JDL-**файле. Пример такого файла:

```
JobType = "MPICH";
CPUNumber = 16;
Executable = "mpi-start-wrapper.sh";
Arguments = "mpi-test OPENMPI";
StdOutput = "mpi-test.out";
StdError = "mpi-test.err";
InputSandbox = {"mpi-start-wrapper.sh
"mpi-hooks.sh "mpi-test.c"};
OutputSandbox = {"mpi-test.err
"mpi-test.out"};
Requirements =
Member("MPI-START other.
GlueHostApplicationSoftwareRunTimeEnvironment)
&& Member("OPENMPI other.
GlueHostApplicationSoftwareRunTimeEnvironment);
```

В примере указаны три вышеупомянутых пользовательских файла: mpi-start-wrapper.sh (стандартный скрипт mpi-start), mpi-hooks.sh (программы выходов) и MPI-программа mpi-test. с на языке программирования C. В России организовано CE с glite/MPI и проведено исследование возможностей этой компоненты. Испытательный CE в Троицке включает 8 процессоров для параллельных MPI вычислений и при тестировании показал устойчивую работу. Автором были также разработаны инструкции для пользователей о применении gLite/MPI, подготовлены тесты для возможной сертификации интегрирования задач типа MPI в gLite. Тест включает проверку всех реализаций MPI-2 (OpenMPI, MPICH, MPICH2) для трех языков программирования C, C++, Fortran. Тест выполняет двухуровневую (по сложности) проверку: *первичную* (типа HelloWorld для каждого из 4 процессоров), и *развернутую* – вычисление числа Пи на 8 процессорах. Результаты исследований представлены в отчетах ЦЕРНа [11, 23].

Мониторирование тестирования

Как уже выше отмечалось, в группе SA3 ведутся работы по исследованию возможности использования стандартного программного обеспечения **Nagios** для автоматизации запусков сертификационных тестов, слежение за их результатами через WEB-интерфейс в CTB. Эта задача сводится к совместной работы Nagios-сервера и gLite UI. Такие исследования были проведены автором, полученные результаты обнадеживают, и, возможно эти исследования ускорят запуск новой мониторинговой системы (взамен уже не функционирующего SAM-монитора) для сертификационных тестов. По крайней мере, группа сертификационных тестов для каталога LFC была успешно опробована в режиме Nagios. Надо отметить, что пришлось выполнить ряд достаточно серьезных работ, таких как инсталляционные работы Nagis и gLite UI в двух режимах (one box и two boxes), проблему включения сертификационных тестов в виде **Nagios-сервисов**, проблему длительного тестирования (проблема продления proxy-сертификатов). Отчеты о проделанной работе и инструкции по совместному использованию Nagios и gLite размещены в [12,22].

Список литературы

[1] https://twiki.cern.ch/twiki/bin/view/EGEE/SA3
[2] https://twiki.cern.ch/twiki/bin/view/EGEE/EGEECertification
[3] http://it-div.web.cern.ch/it-div/
[4] http://lxbra2302.cern.ch/nodestatus/
[5] http://www.nagios.org/
[6] https://twiki.cern.ch/twiki/bin/view/EGEE/CTBNagios
[7] htthttp://www.nagios.org/ps://twiki.cern.ch/twiki/bin/view/EGEE/ETICS
[8] https://twiki.cern.ch/twiki/bin/view/
[9] https://twiki.cern.ch/twiki/bin/view/EGEE/EGEECertification
[10] https://savannah.cern.ch/
[11] https://savannah.cern.ch/task/?9680
[12] https://savannah.cern.ch/task/?8430
[13] http://cvs.web.cern.ch/cvs/howto.php#admgrant
[14] http://glite.cvs.cern.ch/cgi-bin/glite.cgi/org.glite.testsuites.ctb/
[15] http://svn.web.cern.ch/svn/cvs2svn.php
[16] https://twiki.cern.ch/twiki/pub/LCG/WhiteAreas/SAM-WhiteAreas-2008-03-07.ppt
[17] https://lxbrfa2706.cern.ch/vnodes-2.0/
[18] https://twiki.cern.ch/twiki/bin/view/LCG/AvailableTests#Virtual_Organization_Membership
[19] http://glite.cvs.cern.ch/cgi-bin/glite.cgi/org.glite.testsuites.ctb/VOMS/
[20] http://glite.cvs.cern.ch/cgi-bin/glite.cgi/org.glite.testsuites.ctb/LFC/
[21] https://twiki.cern.ch/twiki/bin/view/LCG/AvailableTests#LCG_File_Catalog_LFC
[22] https://savannah.cern.ch/file/Plugins%20for%20Nagios.doc?file_id=7909
[23] https://savannah.cern.ch/file/MPIUserGuide.doc?file_id=9067
[24] http://egee-uig.web.cern.ch/egee-uig/production_pages/MPIJobs.html
[25] http://egee-intranet.web.cern.ch/egee-intranet/NA1/TCG/wgs/mpi.htm