ПРОТОКОЛ ВЗАИМОДЕЙСТВИЯ С БРОКЕРОМ РЕСУРСОВ В СИСТЕМЕ CAEBEANS

А.В. Шамакина

Технология CAEBeans ориентирована на инкапсуляцию и предоставление ресурсов инженерных (CAE) пакетов в среде грид [1]. В основе данного подхода лежит процесс построения дерева проблемно-ориентированных оболочек CAEBeans над CAE пакетом на основе классификации прикладных задач.

Система CAEBeans представляет собой совокупность программных средств, данных и аппаратных ресурсов, ориентированных на поддержку технологии CAEBeans. Программные средства, входящие в систему CAEBeans, призваны обеспечить поддержку процессов создания, модификации и использования оболочек CAEBeans. Выделяются следующие основные программные продукты, входящие в состав системы CAEBeans:

1. CAEBeans Portal – web-портал, предназначенный для постановки пользователями задачи в системе CAEBeans;
2. CAEBeans Server – web-сервис XML, который обеспечивает выполнение задачи пользователя, отвечает за хранение и интерпретацию CAE-проектов;
3. CAEBeans Broker – автоматизированная система регистрации, анализа и предоставления ресурсов распределенной вычислительной среды.

Брокер ресурсов выступает в качестве промежуточного звена между CAEBeans Server и грид-средой, обеспечивающего поиск и предоставление ресурсов, оптимальных для решения данного задания [2]. Протокол необходим для обеспечения взаимодействия между сервером и брокером в процессе поиска и выделения вычислительных ресурсов целевой системы для решения определенного задания.

Все ресурсы, которыми оперирует брокер, делятся на две группы: лицензии и целевые системы. Каждая целевая система представляет собой один физический ресурс. Брокер формирует на базе физического ресурса несколько виртуальных ресурсов разного размера. В каждый момент времени на виртуальном ресурсе выполняется только одно задание. После передачи сервером требований к ресурсам для конкретного задания производится его постановка в очередь к виртуальным ресурсам и лицензиям.

Обозначим $\Im=\{J_i \mid 1 \le i \le n\}$ – конечный набор заданий обрабатываемых брокером; $\vartheta=\{V_j \mid 1 \le j \le m\}$ – набор виртуальных ресурсов; $R_{Ji,Vj}$ – рейтинг задания J_i в очереди к ресурсу V_j; *timeout* – максимальное время ожидания ответа от сервера, отправившего запрос на выполнение задания J_i.

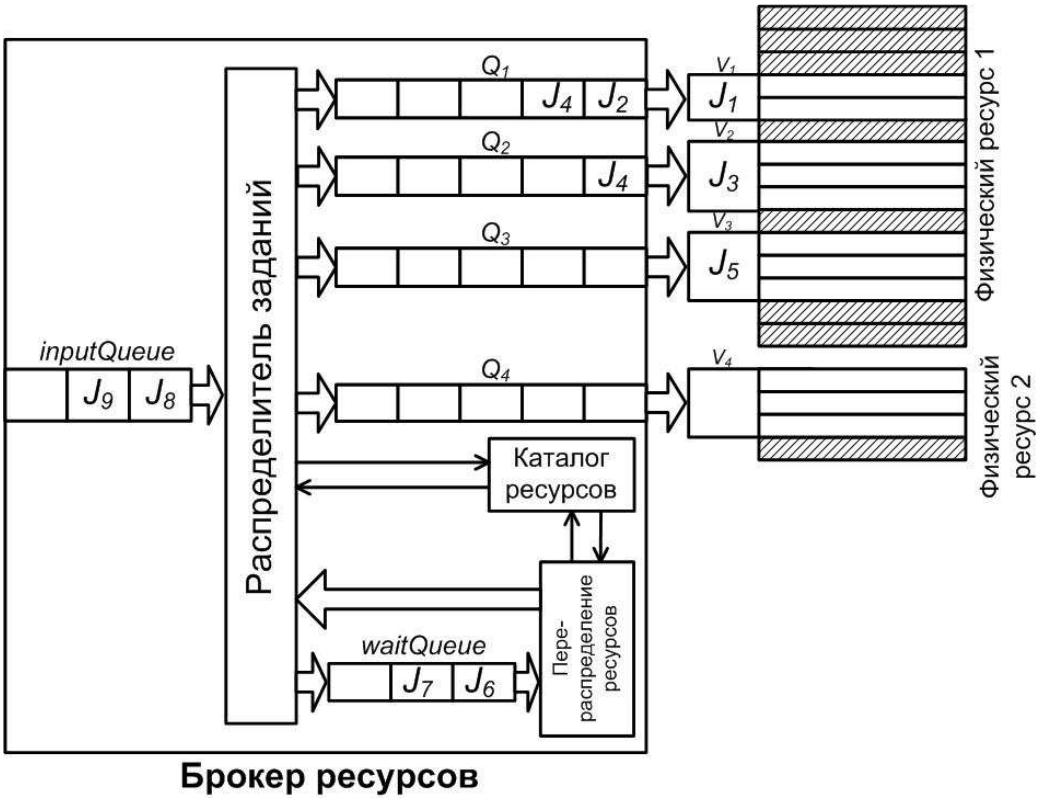

Рис. 1. Очереди к вычислительным ресурсам

Протокол захвата ресурсов основывается на очередях к вычислительным ресурсам. Множеством $Q=\{Q_j \mid 1 \le j \le m\}$ назовем множество очередей ожидания к виртуальным ресурсам. Отдельно определим *waitQueue* – очередь, в которую помещаются задания, которые не могут быть решены посредством имеющихся виртуальных ресурсов, но объединение вычислительных возможностей нескольких виртуальных ресурсов, находящихся на базе одного физического ресурса, позволит обеспечить решение данного задания. На Рис. 1 приведены очереди к вычислительным ресурсам.

Протокол взаимодействия сервера и брокера ресурсов имеет три фазы. Во время выполнения фазы I сервер передает заявку на необходимые для решения задания ресурсы. Брокер возвращает серверу статус выполнения функции: «*Задание ожидает в очереди*» или «*Нет ресурсов, соответствующих запросу*». Если брокер поставил задание в очередь, то сервер может в любой момент удалить ее из очереди. Фаза I может длиться сколь угодно долго. При завершении работы задания на ресурсе брокер проводит аукцион, отсылает извещение победителю и начинается исполнение фазы II: сервер производит захват ресурса и удаляет текущее задание из очередей на альтернативные ресурсы. Фаза II выполняется мгновенно. Задания, не получившие доступа к требуемому ресурсу и не перешедшие в Фазу II, остаются в очереди на данный ресурс и ожидают его дальнейшего освобождения. Фаза III необходима для освобождения ресурса после выполнения на нем задания.

Во время выполнения **фазы I** сервер делает заявку на ресурсы, необходимые для выполнения задания J_i.

1. Сервер вызывает функцию *queryResource(Query query)* брокера ресурсов. В качестве аргументов данной функции передаются спецификации к вычислительному ресурсу, которые должны выполняться для задания J_i, а также предельное время выполнения задания J_i и время, необходимое для отклика сервера.
2. Брокер производит поиск соответствующих ресурсов. Возможны следующие ситуации.
   1. Если брокер в результате поиска нашел виртуальные ресурсы, удовлетворяющие спецификации, то он выполняет следующие действия:

- добавляет задание J_i в конец очередей всех ресурсов, которые удовлетворяют спецификациям;
- формирует рейтинг R для задания J_i на основе позиции P, занимаемой заданием в каждой из очередей, с учетом времени отклика сервера T. Формула для расчета рейтинга имеет вид:

$$R = \frac{1}{P+T}$$

- возвращает *queryResource()* серверу статус: «*Задание J_i ожидает в очереди*».

   2. Если для задания J_i подходящих виртуальных ресурсов на данный момент нет, то брокер ресурсов выполняет следующие действия:

- добавляет J_i в очередь брокера для заданий, требующих перераспределения виртуальных ресурсов;
- возвращает *queryResource()* серверу статус: « *Задание J_i ожидает в очереди*»;
- производит поиск виртуальных ресурсов, объединение которых может удовлетворить требованиям запроса задания J_i;
- отмечает маркером виртуальные ресурсы необходимые для формирования нового ресурса;
- производит дальнейшую постановку заданий в очереди к виртуальным ресурсам, которые не маркированы;
- дожидается выполнения всех заданий на маркированных ресурсах;
- выполняет перераспределение виртуальных ресурсов, учитываются все задания, стоящие в очереди для перераспределения;
- ставит J_i на первое место в очереди к новому виртуальному ресурсу.

   3. Если в распоряжении брокера ресурсов для выполнения задания J_i нет физических ресурсов, то брокер возвращает серверу статус: «*Нет ресурсов, соответствующих запросу*». В этом случае сервер прерывает исполнение текущего задания.
3. Если брокер вернул статус «*Задание J_i ожидает в очереди*», то сервер ожидает оповещения от брокера об освобождении одного из виртульных ресурсов, к которым задание стоит в очереди, чтобы перейти в фазу II.
4. До того, как сервер совершит переход в фазу II, в любой момент он может удалить задание Ji из очереди к ресурсам. Для этого сервер вызывает функцию брокера ресурсов *delJob(identifier JobId)*. Если сервер вызвал функцию *delJob()* брокера ресурсов, то брокер удаляет задание Ji из всех очередей к виртуальным ресурсам.

Фаза II начинается, когда задание, выполняющееся на виртуальном ресурсе, завершает свою работу.

1. Заданию, стоящему в очереди к виртуальному ресурсу на первом месте, брокер посылает извещение об освобождении ресурса. Вместе с оповещением брокер отсылает вектор виртуальных ресурсов, в очереди у которых стоит задание. Вектор ресурсов содержит характеристики виртуальных ресурсов и позиции в

очереди. Сервер должен поддерживать оповещения об изменении статуса ресурса и быть подписан на эти оповещения (в контексте технологии WSRF оповещение сервера производится по стандарту WS-Notification).

2. Для захвата освободившегося ресурса сервер, в контексте задания J_i, вызывает функцию res *blockResource(URL resourceURL)* брокера ресурсов.

3. Если была вызвана функция *blockResource()*, то брокер выполняет следующие действия:

- выделяет виртуальный ресурс для задания Ji;
- удаляет задание J_i из всех очередей виртуальных ресурсов.

4. Если после извещения об освобождении ресурса сервером не был инициирован захват ресурса в течение времени отклика, то брокер удаляет задание из очереди и отправляет извещение следующему в очереди заданию.

Фаза III предназначена для освобождения ресурсов, занимаемых заданием.

1. Сервер вызывает функцию *unblockResource()* брокера ресурсов.
2. Если была вызвана функция *unblockResource()*, то брокер освобождает виртуальный ресурс и переходит во вторую фазу.

Во время работы брокера ресурсов автоматически производятся следующие действия:

- ревизия и опрос ресурсов для определения динамических характеристик ресурсов и проверки работоспособности ресурсов;
- проверка времени жизни выполняющихся на ресурсах заданий;
- перераспределение виртуальных ресурсов.

В настоящее время реализован прототип системы CAEBeans Broker как грид-сервиса, поддерживающего стандарты WSRF. CAEBeans Broker введен в опытную эксплуатацию на базе вычислительных ресурсов Суперкомпьютерного центра Южно-Уральского государственного университета.

ЛИТЕРАТУРА:

1. Г.И. Радченко, Л.Б. Соколинский Технология построения виртуальных испытательных стендов в распределенных вычислительных средах // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики. -№ 54. -2008. -С. 134-139.
2. Yu J., Buyya R., Ramamohanarao K. Workflow Scheduling Algorithms for Grid Computing // Metaheuristics for Scheduling in Distributed Computing Environments. -2008. -Vol. 146. -P. 173-214.