ИНФРАСТРУКТУРА РАСПРЕДЕЛЁННОЙ СИСТЕМЫ С СОХРАНЕНИЕМ ГЛОБАЛЬНОГО СОСТОЯНИЯ

В.Н. Брагилевский, С.Г. Найденов
Южный федеральный университет
Россия, 344006, г.Ростов-на-Дону, ул.Большая Садовая, 105/42
E-mail: bravit@sfedu.ru, red-2lip@ya.ru

В настоящей работе описывается архитектура и основные детали реализации распределенной системы с возможностью сохранения и загрузки глобального состояния. Отличительной чертой предлагаемой системы является поддержка произвольной функциональной нагрузки, организуемая посредством реализации открытых интерфейсов. В качестве примера использования разработанной архитектуры приводится система рассылки сообщений.

DISTRIBUTED SYSTEM ARCHITECTURE WITH GLOBAL STATE SERIALIZATION / V.N. Bragilevsky, S.G. Naidenov (Southern Federal University, 105/42 Bolshaya Sadovaya st., Rostov-on-Don, 344006, Russia). The paper is devoted to architecture and implementation details of distributed system with global state serialization. The described system supports any functional payload via implementing specified open interfaces. The sample system for message transfer is also considered as an example of concrete system implementation.

Введение

Широкое распространение компьютерных сетей и необходимость интеграции бизнес-процессов различных организаций свидетельствует об актуальности исследований в области технологий построения распределённых систем в целом и отдельных их компонентов в частности.

Распределенная система — это набор независимых компьютеров, представляющийся их пользователям единой объединенной системой. Основная задача распределенных систем — облегчение доступа к удаленным ресурсам и обеспечение их совместного использования. Глобальное состояние распределенной системы включает в себя локальные состояния каждого процесса вместе с находящимися в пути сообщениями (то есть посланными, но еще не доставленными). Сохранение глобального состояние позволяет получать полную информацию о деятельности системы, которую в дальнейшем можно использовать для анализа и подстройки параметров с целью обнаружения тупиков, балансировки нагрузки или повышения уровня отказоустойчивости [1]. Сохранённое глобальное состояние может использоваться для восстановления системы после сбоев или отказов, при этом система способна продолжать функционировать так, что для её пользователей факт отказа будет совершенно прозрачным. При простом способе записи глобального состояния распределенной системы, вводится понятие распределенного снимка состояния, отражающего состояние, в котором находилась распределенная система. Важным его свойством является то, что он отражает непротиворечивое глобальное состояние. Возможность восстановления состояния системы с некоторого ранее сохраненного состояния данной системы позволяет сократить риск потери результатов работы в случае некорректного завершения.

Задачи сохранения и восстановления глобального состояния тесно примыкают к задаче синхронизации в распределенных системах. Синхронизация позволяет поддерживать в системе единое время, что дает возможность процессам организовать совместное логически корректное взаимодействие. Такое время может быть как физическим (абсолютным), так и логическим (обеспечивающим относительную упорядоченность событий). По этой причине становится возможным сохранение и восстановление такого взаимодействия.

В настоящей работе предпринята попытка разработки обобщённой архитектуры распределённой системы, в которой существует возможность сохранения и восстановления глобального состояния системы.

Целью работы является не разработка конкретной распределённой системы, выполняющей поставленные перед ней задачи, а построение инфраструктуры, которая могла бы использоваться для решения произвольных задач. Поэтому основным требованием при разработке архитектуры, не связанным, вообще говоря, с глобальным состоянием, является требование её максимальной гибкости и, соответственно, минимальных ограничений на конкретную функциональность.

Сохранённое глобальное состояние должно восстанавливаться как во время работы системы, так и при её запуске. Первая возможность обеспечивает эффект приостановки системы с целью подстройки: состояние сохраняется, анализируется, изменяется в соответствии с задачами системы и, наконец, восстанавливается. Вторая возможность гарантирует сохранение данных системы в случае сбоя и продолжение корректного функционирования.

Компоненты системы

Система является централизованной, она управляется *менеджером распределённой системы*. На каждом из компьютеров, составляющих распределённую систему, размещается т. н. *коммуникационный узел*. Наконец, полезная нагрузка системы (*payload*) обеспечивается *функциональными узлами*.

В обязанности менеджера распределенной системы, входят:

- управление коммуникационными узлами;
- поддержание общего времени в системе;
- распространение актуальных функциональных узлов среди коммуникационных узлов системы;
- взаимодействие с пользователем.

Менеджер распределенной системы является системообразующим элементом и состоит из четырех основных компонентов:

1. таблица, содержащая информацию обо всех узлах распределенной системы, постоянно поддерживаемая в актуальном состоянии;
2. сервер синхронизации времени, функционирующий по принципу алгоритма Беркли [2];
3. сервер функциональной нагрузки, при помощи которого происходит хранение и распространение актуальных версий функциональных узлов среди коммуникационных;
4. сервер глобального состояния, ответственный за сбор и хранение всех локальных состояний функциональных узлов, а так же всех сообщений находящихся в пути.

Коммуникационный узел – это объект, распространенный среди пользователей распределенной системы, в обязанности которого входит выполнение команд менеджера распределенной системы и управление одним или несколькими функциональными узлами. Примерами команд менеджера могут являться следующие команды:

- загрузка и запуск функционального узла;
- передача команд менеджера напрямую функциональному узлу;
- получение и передача менеджеру состояние функционального узла;
- остановка выполнения функционального узла;

- перенос выполнения функционального узла на другой коммуникационный узел;
- синхронизация времени функционального узла и менеджера распределенной системы.

Функциональный узел – это программный код, реализующий заданный интерфейс, удовлетворяющий некоторым структурным требованиям языка программирования и выполняющий поставленную задачу. Реализуемый интерфейс позволяет функциональному узлу выполнять ряд основных команд:

- запуск функционального узла без остановки коммуникационного узла;
- запуск функционального узла с некоторого сохранного ранее состояния;
- получение локального состояния узла;
- выполнение прямых команд менеджера;
- команды необходимые для синхронизации времени;
- остановка выполнения функционального узла.

В качестве примера функциональной нагрузки разработана система распределенной передачи сообщений. В обязанности такого функционального узла входит пересылка сообщений и массовая рассылка сообщений. Распределение потока сообщений на различные узлы системы позволило добиться наилучшей эффективности при передаче большого количества сообщений и их сохранности.

Глобальное состояние распределенной системы включает в себя локальные состояния каждого процесса вместе с находящимися в пути сообщениями. Что именно считать локальным состоянием процесса, зависит от того, какую именно задачу выполняет функциональный узел системы [1]. В качестве локального состояния будем понимать срез, то есть снимок некоторого этапа выполнения алгоритма функционального узла. Так, например, в случае распределенной базы данных под состоянием будем понимать только лишь временные записи, используемые для вычислений. В случае распределенной системы со сменным функционалом разработчику предоставляется возможность самому определить, что именно должно являться локальным состоянием узла функциональной нагрузки, и какую задачу должен выполнять такой узел. Распределенная система предоставляет пользователю средство управления состояниями локальных узлов. Распределенная система способна сформировать глобальное состояние системы, являющееся совокупностью таких локальных состояний и сообщений, находящихся в пути. Под сообщениями, находящимися в пути, понимаются сообщения, предназначенные для взаимодействия элементов распределенной системы. Все локальные сообщения в системе хранятся и передаются в виде последовательности байтов. Локальные состояния не зависят от элементов распределенной системы. Передаваемое состояние подвергается процессу сериализации при помощи секретного ключа, без владения которым сообщение не может быть десериализовано.

Детали реализации

Менеджер распределенной системы включает в себя таблицу, содержащую информацию обо всех узлах системы. Запись такой таблицы содержит информацию об одном коммуникационном узле, одном или более функциональных узлах, выполняющихся на данном коммуникационном узле, информацию о текущем состоянии коммуникационного узла (активен или неактивен), информацию о наличии локальных состояний функциональных узлов. Данные в таблице хешированы по имени коммуникационного узла. При первом подключении коммуникационного узла к менеджеру системы в таблицу заносится новая запись. Предусмотрена возможность отслеживания коммуникационных узлов, не являющихся активными в течение длительного времени, с последующим их удалением. Специальный менеджер таблицы с заданной периодичностью производит опрос активности всех коммуникационных узлов и при необходимости переводит флаг состояния узла в активное (неактивное) состояние.

Сервер функциональных узлов является образующим элементом распределенной си-

стемы со сменной функциональной нагрузкой. Данный сервер хранит один или более актуальных функциональных узлов в виде *.jar файлов. Коммуникационный узел в любой момент времени способен по команде менеджера загрузить с сервера функциональных узлов новый функциональный узел и запустить его на выполнение. Такая команда должна содержать адрес и порт функционального узла, а также код команды «загрузить функциональный узел». После загрузки и запуска функционального узла коммуникационный узел отправит уведомление о запуске нового функционального узла менеджеру распределенной системы. Приход такого уведомления влечет за собой добавление в запись соответствующего коммуникационного узла в таблице коммуникационных узлов менеджера новой информации о запущенном функциональном узле. Также функциональный узел может быть запущен с некоторого ранее сохраненного состояния, полученного от менеджера системы. Сервер функциональных узлов реализован по принципу метода опроса каналов и способен одновременно обслуживать большое количество коммуникационных узлов. В ответ на запрос коммуникационного узла, сервер возвращает последовательность байтов, содержащуюся в *. jar файле.

Сервер состояний выполняет сбор и хранение состояний функциональных узлов. Все сообщения, передаваемые в системе, хранятся в специальном буферизированном хранилище до момента получения отчета о доставке. Такое хранилище содержится в сервере состояний, а сообщения, хранящиеся в нем, считаются сообщениями, находящимися в пути. Наличие такого хранилища сообщений позволяет характеризовать данную систему передачи сообщений как нерезидентную [1]. Сервер состояний способен возвращать состояния отдельных функциональных узлов менеджеру системы. По команде менеджера системы в любой момент времени сервер состояний способен создать глобальное состояние распределенной системы, при этом будет произведен сбор локальных состояний всех функциональных узлов. Глобальное состояние сохраняется в виде архивного файла.

Коммуникационный узел – это объект, который распространен среди пользователей распределенной системы. Коммуникационный узел состоит из нескольких компонентов:

- таблица, содержащая записи о функциональных узлах запущенных на данном коммуникационном узле;
- клиент передачи сообщений, функционирующий на основе сокетного соединения;
- сервер, реализующий взаимодействие по методу опроса каналов.

В обязанности коммуникационного узла входит выполнение команд менеджера и управление функциональными узлами, передача локального состояния менеджеру и запуск функциональных узлов.

Функциональный узел — программный код, реализующий заданный интерфейс, удовлетворяющий некоторым структурным требованиям языка, и выполняющий поставленную задачу. К структурным требованиям языка относиться требование реализации именно фиксированного интерфейса функционального узла:

```
public interface FunctionNode {
   void CreateNode();
   void CreateNode(int port);
   void CreateNode(int port,State s);
   int GetPort();
   String GetAddress();
   void SetTime(long t);
   long GetTime();
   State GetState();
   void ExecuteCommand(Command cmd);
   void Stop();
}
```

Система передачи сообщений

Разработанная распределенная система с сохранение глобального состояния реализует функциональный узел для системы передачи сообщений. Она предназначена для пересылки и массовой рассылки сообщений. Соответствующий функциональный узел состоит из нескольких основных элементов:

- сервер, функционирующий по принципу метода опроса каналов;
- очередь сообщений, предназначенных для отправки;
- внутренние часы;
- клиент, способный передавать одновременно большое количество сообщений, функционирующий на основе пула потоков;
- очередь отправленных сообщений, ожидающих отчета о доставке;
- структуру данных, накапливающую информацию об уже переданных сообщениях.

Состоянием такого функционального узла будет являться совокупность очереди сообщений, очереди сообщений находящихся в пути и история отправленных сообщений. В случае запуска данного состояния на другом коммуникационном узле будет продолжена пересылка сообщений, находящихся в очереди, накопление истории переданных сообщений и ход внутреннего времени.

Заключение

Предлагаемая инфраструктура реализована средствами платформы Java, что позволяет использовать отдельные компоненты на компьютерах с разными операционными системами. Сетевое взаимодействие организовано с использованием метода опроса каналов (пакет java.nio), который является наиболее эффективным способом организации передачи данных на платформе Java; передаваемые данные сериализуются в бинарную форму стандартными средствами платформы Java [3]. Всё это позволяет использовать систему в задачах с высокой нагрузкой. Для запуска функциональных узлов используется динамическая загрузка классов из jar-архивов [3], поэтому функциональные узлы могут обновляться «на лету», во время работы системы.

В работе предложена и реализована инфраструктура глобальной распределённой системы с сохранением и восстановлением глобального состояния, синхронизацией времени, динамической загрузкой и изменением конкретной функциональности. Новой является предлагаемая архитектура распределённой системы на основе менеджера и наборов коммуникационных и функциональных узлов. Инфраструктура реализует известные алгоритмы сохранения глобального состояния и синхронизации времени [1, 2], комбинируя их способом, ранее в литературе не описанным. Использование функциональных узлов, ограниченных фиксированным интерфейсом, и их динамическая загрузка также является стандартным приёмом объектно-ориентированного проектирования [4]. Минимальные ограничения на функциональные узлы позволяют использовать разработанную инфраструктуру в самых разнообразных задачах, таких как сложные распределённые вычисления или базы данных.

Литература

1. Таненбаум Э., ван Стеен М. Распределённые системы. Принципы и парадигмы. — СПб.: Питер, 2003. — 877 с.
2. Тель Ж. Введение в распределенные алгоритмы. — М.: МЦНМО, 2009. — 616 с.
3. Хорстманн К.С., Корнелл Г. Java 2. Библиотека профессионала. В 2 т. Т. 2. Тонкости программирования. — 8-е изд. — М.: Вильямс, 2008. — 992 с.
4. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования — СПб: Питер, 2007. — С. 366.