Е.В. Симонова, Д.А. Новиков

СИНХРОНИЗАЦИЯ WEB-ПРИЛОЖЕНИЯ И БАЗЫ ДАННЫХ В МУЛЬТИАГЕНТНОЙ СИСТЕМЕ ЦЕЛЕВОГО ПЛАНИРОВАНИЯ КОСМИЧЕСКИХ АППАРАТОВ ДИСТАНЦИОННОГО ЗОНДИРОВАНИЯ ЗЕМЛИ

(Самарский национальный исследовательский университет
имени академика С.П. Королёва)

Введение

В связи с ростом числа использования веб-приложений для отображения данных и постепенным отказом от старых способов отображения информации, например, форм в C# и Swing в Java, проблема синхронизации современных веб-приложений и информации, хранящейся в базах данных (БД), становится все более актуальной.

Постановка задачи синхронизации web-приложения и БД в акторной системе

Имеется база данных, браузер клиента, клиент использует некоторое web-приложение. Необходимо показать, каким образом в окно web-приложения, открытое во вкладке браузера клиента, передаются сообщения об удалении, добавлении, изменении информации в БД. Взаимодействие между web-приложением и БД организовано через web-сервер. Архитектура системы представлена на рисунке 1.

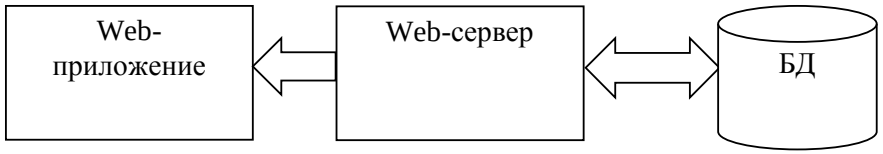

Рисунок 7 – Архитектура системы

Связь между web-приложением и web-сервером осуществляется посредством технологии WebSocket [1], которая идеально подходит для пересылки небольших сообщений. Веб-сервер написан на языке Java и основан на фреймворке Akka [2], в основе которого лежит модель акторной системы.

Синхронизацию web-приложения и базы данных должен осуществлять web-сервер. Следовательно, все изменения в акторной системе, т.е. добавление или удаление объектов (акторов) и изменение внутренних состояний (свойств) акторов, должны сохраняться в БД и одновременно отсылаться web-приложению. Для базы данных определены следующие типы акторов: спутники, объекты наблюдения, пункты приема информации. Типы акторов описываются с помощью таблиц, строки которых соответствуют акторам, состояния акторов определяются значениями строк таблиц.

Описание акторов в акторной системе web-приложения

В процессе инициализации web-сервера загружаются следующие данные из БД:

1) список спутников;
2) список объектов наблюдения;
3) список пунктов приема информации.

Для каждого объекта из каждого списка создается уникальный актор с параметрами каждого объекта.

Так как акторная система web-приложения в целом достаточно сложна и объемна, в рамках решаемой задачи имеет смысл рассмотреть только следующие акторы:

1) DBActor – актор взаимодействия с БД, осуществляющий операции запроса, добавления, изменения и удаления данных непосредственно из БД;
2) WebActor – актор взаимодействия с web-приложением, осуществляющий отправку событий на web-сокет (WebSocket).

Все акторы имеют доступ к единой шине событий (EventBus) и могут осуществлять ее прослушивание путем создания слушателя шины событий (EventBusListener), который регистрируется у шины событий посредством вызова метода Subscribe и передачи себя в качестве параметра. Акторы могут также генерировать произвольные события (Event) и отправлять их по шине событий.

Принцип взаимодействия акторов на стороне web-сервера

При создании нового объекта (актора) генерируется событие создания (CreateEvent) и отправляется по шине событий. Другие акторы, подписавшиеся на прослушивание шины событий, получают это событие. Акторы, получившие событие, должны определить, отвечают ли они за его обработку, путем сравнения типа события с типом, явно прописанным в них программистом, и при необходимости произвести действия, соответствующие каждому событию.

При изменении внутренних свойств актора происходит генерация события изменения (ChangeEvent) и отправка его на шину событий. Заинтересованные в этом событии акторы произведут соответствующие действия.

Принцип взаимодействия окна web-приложения и web-сервера

Каждое окно web-приложения, открытое в отдельной вкладке web-браузера, полностью независимо от других окон, открытых в соседних вкладках. Синхронизация осуществляется не между web-приложением в целом и web-сервером, а между окном web-приложения и web-сервером.

Каждое открытое окно создает своего слушателя и регистрирует его на прослушивание web-сокета, и подобно акторам в акторной системе, окна получают события и производят соответствующие им действия.

Решение задачи синхронизации

Например, был создан новый объект Спутник (SatteliteActor). При создании он сгенерировал событие (CreateSatteliteEvent) и отправил его на шину событий.

На прослушивание шины событий были подписаны DBActor и WebActor. Соответственно, каждому из них пришло событие добавления нового объекта. Актор DBActor по типу события произвел добавление новой строки в таблицу Спутники. В тот же момент, асинхронно с ним, актор WebActor сгенерировал новое событие и отправил его на web-сокет. Окна web-приложения, которые подписались на прослушивание web-сокета, получили данное событие и произвели соответствующие действия.

В случае изменения свойств Спутника будет сгенерировано событие (ChangeSatteliteEvent), агенты DBActor и WebActor выполнят соответствующие действия – DBActor изменит записи в таблицах, а WebActor уведомит окна, подписавшиеся на это событие, об изменении свойств Спутника.

Заключение

Была решена проблема синхронизации данных между web-приложением и базой данных для мультиагентной системы целевого планирования космических аппаратов дистанционного зондирования Земли с помощью взаимодействия акторов.

Литература

1) WebSocket. – Режим доступа: https://ru.wikipedia.org/wiki/WebSocket
2) Akka. – Режим доступа: http://akka.io/

Е.В. Симонова, Д.А. Проценко

МУЛЬТИАГЕНТНЫЙ ПОДХОД К ПЛАНИРОВАНИЮ ПОЛЁТНЫХ ОПЕРАЦИЙ

(Самарский национальный исследовательский университет
имени академика С.П. Королёва)

Введение

Задача планирования является сложно организуемой и нелинейно возрастающей с увеличением числа участников, возможных ограничений и критериев эффективности. В космической промышленности также существенным фактором является надёжность и требование быстроты реакции на те или иные события. Грамотное планирование необходимо для эффективного выполнения задач, а в условиях космической экспедиции требуется также соблюдать размещение определённых цепочек операций и их длительность.

В настоящее время разработана и введена в эксплуатацию автоматизированная система планирования программы пролета и грузопотока российского сегмента МКС (АСП РС МКС) [1, 2]. Эта система имеет модульную архитектуру, использует единую базу данных полётных операций и планов. Она позволяет создавать планы на разных уровнях иерархии (номинальный план полёта, общий план сопровождения, детальный план полёта), а также анализировать программу выполнения полёта.

В частности, для генерации детального плана полёта реализован модуль, использующий генетический алгоритм для оптимизации плана по фиксированным критериям эффективности.

Тем не менее, появилась потребность в его сравнительном анализе с другим методом, использующим мультиагентные технологии для планирования ресурсов.

Постановка задачи

Требуется разработать модуль для создания детального плана полёта, использующий мультиагентные технологии и реализующий планирование полётных операций с учётом следующих критериев эффективности:

1. Максимизация загрузки члена экспедиции (ЧЭ), т.е. данный ресурс должен иметь минимальные промежутки незанятости.
2. Максимальная эффективность распределения полётных операций по компетенциям, т.е. полётные операции должны быть запланированы на ресурсы с компетенцией не ниже требуемой.
3. Более приоритетные полётные операции вытесняют менее приоритетные, т.е. в случае невозможности запланировать все исходные полётные операции, необходимо отдавать предпочтение тем, у кого приоритет выше.

Методы решения

Для планирования предлагается использовать мультиагентный подход, заключающийся в представлении всех участников процесса в виде агентов – автономных программных объектов, имеющих определённую цель.

Каждый из сущностных агентов, представляющих интересы какой-либо сущности в предметной области, является либо агентом потребности (агент полётной операции), либо агентом возможности (агент средства связи S-band). Таким образом происходит формирование ПВ-сети – сети потребностей-возможностей, в которой будет происходить формирование плана удовлетворения всех потребностей путём переговоров агентов друг с другом.

Полётная операция (ПО) – структура, описывающая выполняемые ЧЭ действия с использованием фиксированной группы принадлежности и, опционально, средств связи (СС).

Группа принадлежности – термин, объединяющий в себе назначение ПО. Может являться бортовой системой или космическим экспериментом.

Средство связи – ресурс, используемый во время исполнения ПО.

Компетенция ЧЭ – вид ограничения, при котором для выполнения ПО требуется не фиксированный ЧЭ, а имеющий некий уровень компетенции, которая привязывается к определённой группе принадлежности.

Каждая ПО сдержит:

- название (полное и краткое);
- длительность (в минутах);
- приоритет – числовое значение от 1 до 10, где 1 – высший;
- флаг обязательности, доступный только для ПО с приоритетом 10, означающий возможность исключения из плана в случае конфликтов;