УДК 004.946

МЕТОДЫ И СПОСОБЫ ОРГАНИЗАЦИИ ХРАНИЛИЩ ДЛЯ РАСПРЕДЕЛЕННЫХ ИНТЕРАКТИВНЫХ СИСТЕМ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ

Черняев А.А., Башков Е.А.
Донецкий национальный технический университет
кафедра прикладной математики и информатики
E-mail: chernyaev_a_a@mail.ru

Аннотация

Черняев А.А., Башков Е.А. Методы и способы организации хранилищ для распределенных интерактивных систем виртуальной реальности. *Рассмотрены задачи и области применения виртуальных миров, разбор организации хранилищ на основе систем виртуальной реальности, сделаны выводы об актуальности исследований способов организации хранилищ для распределенных интерактивных систем.*

Введение

Виртуальный мир – это интерактивное смоделированное окружение, к которому множество пользователей имеют доступ через онлайн интерфейс [3]. Виртуальные миры иногда называют «цифровыми мирами», «искусственными мирами» и «МОИ» (мультипользовательские он-лайн игры, MMOG). На сегодня существует множество разных виртуальных миров, все из которых имеют шесть признаков, которые присущи всем мирам:

1.Совместное пространство: участвовать в жизни мира могут одновременно много пользователей.

2.Графический пользовательский интерфейс: пространство в мире отражено виртуально, и варьируется по стилю от 2D «мультипликационного» изображения до более впечатляющих 3D изображений.

3.Оперативность: общение происходит в режиме реального времени.

4.Интерактивность: мир позволяет участникам изменять, развивать, строить или принимать содержание, подобранное специально для него.

5.Постоянство: существование мира продолжается независимо от того, находятся ли отдельные пользователи в системе.

6.Общение / общество: мир дает возможность и содействует формированию социальных групп внутри мира, таких как команды, гильдии, клубы, клики, соседства, комьюнити и так далее.

Виртуальные миры делятся на несколько типов по своему назначению:

- коммерческие
- коммуникация / построение онлайн сообществ
- образование
- военное обучение

С дальнейшим развитием подобных технологий в течение следующих лет, виртуальные миры смогут использовать для самых разных целей, поскольку все больше людей проводят в них время.

Анализ структур данных

Системы виртуальной реальности должны содержать всю информацию об устройстве внутреннего мира в некоторых хранилищах, базах данных. Примером такой информации могут служить данные о персонажах, текстуры, данные о взаимодействии объектов, типы

объектов и т.д. Существует большое количество способов хранения информации в базах данных, но оптимальным вариантом является логическое разделение между таблицами. Например, для пользователей создать таблицу users, в которой будут присутствовать все поля, необходимые для описания данных о пользователе, для структур создать таблицу textures, в которой будет храниться информация о текстуре и т.д. В таблице 1 приведены примеры простых таблиц для описания информации.

Таблица 1. Пример простых таблиц для хранения информации объектов

Название таблицы	Название поля	Назначение поля
Users		содержит данные о пользователях
	Id_user	уникальный идентификатор пользователя
	Email	электронный адрес
	Pass	пароль пользователя
	Name	имя пользователя, которое будет отображаться в виртуальном мире
	Id_avatar	идентификационный номер аватара
Avatars		содержит данные об аватарах
	Id_avatar	идентификационный номер аватар
	Id_sex	тип пола аватара
	Id_hair	тип волос
	Id_face	тип лица
	Id_body	тип строения тела
	Id_clothes	тип одежды
Textures		содержит данные о текстурах
	Id_texture	идентификационный номер текстуры
	Path	путь к изображению текстуры
	Id_type	тип текстуры
	Height	высота
	Width	ширина

Кроме того, у каждого аватара может быть свой инвентарь, список друзей и людей, с которыми не хотелось бы общаться, список сообщений. Информацию обо всех этих данных тоже необходимо хранить в базе данных. Все выше перечисленные таблицы можно отнести к общеобязательным, которые должны быть реализованы во всех многопользовательских играх (мирах). Кроме того, существуют специфические особенности, которые могут быть реализованы не во всех виртуальных мирах (например, начисление опыта, убийство персонажей и т.д.). Следовательно, какие данные хранить в базе данные, каким способом распределять данные между таблицами зависит от того, какое будет назначение виртуального мира.

В реальных системах количество полей и их назначение может отличаться от тех, что приведены в табл.1. Примером могут служить базы данных платформ Opensim [2] и UEF [1]. Платформа Opensim является достаточно популярной среди других подобных систем (предназначена для общения между людьми, проведение виртуальных конференций, выставок, галерей). Проект UEF является будущей многопользовательской игрой, главной целью игры – прокачивание персонажа.

Opensim представляет собой расширяемую платформу, которая может моделировать трехмерные виртуальные миры. В этих виртуальных мирах существует возможность динамического создания, изменения, удаления примитивов в пространстве. В качестве демонстрации силы такой платформы, она конфигурируется по умолчания так, что была совместима с приложением Second Life, выпущенная компанией Linden Lab.

В SL существует возможность создания своего собственного региона. На основе базовой конфигурации от Linden Lab, каждый регион содержит в себе пять серверов, которые необходимы ему для работы. Это такие сервера, как: User, Grid, Asset, Inventory, Messaging.

UserServer - отвечает за идентификацию пользователя, создает уникальный идентификатор сессии для клиента, который может использоваться для идентификации на других серверах в той же сети.

GridServer - отвечает за идентификацию регионов, которые находятся в данной сети.

AssetServer - база данных, в которой хранятся все звуки, текстуры, скрипты, изображения и т.д. Но необходимо учитывать, что если добавить какой-то элемент в эту базу, то его параметры не возможно изменить. Если, например, понадобится изменить какую-то текстуру, то ее придется изменить в графическом редакторе, а после загрузить на сервер. Причем старая версия так и останется, ее можно будет лишь удалить.

InventoryServer – сервер, отвечающий за инвентарь. Если пользователь переключается между различными серверами, необходимо, чтобы инвентарь оставался одинаковым.

MessagingServer – помогает общаться с различными людьми внутри симулятора.

На рис. 1 показана схема взаимодействия перечисленных серверов.

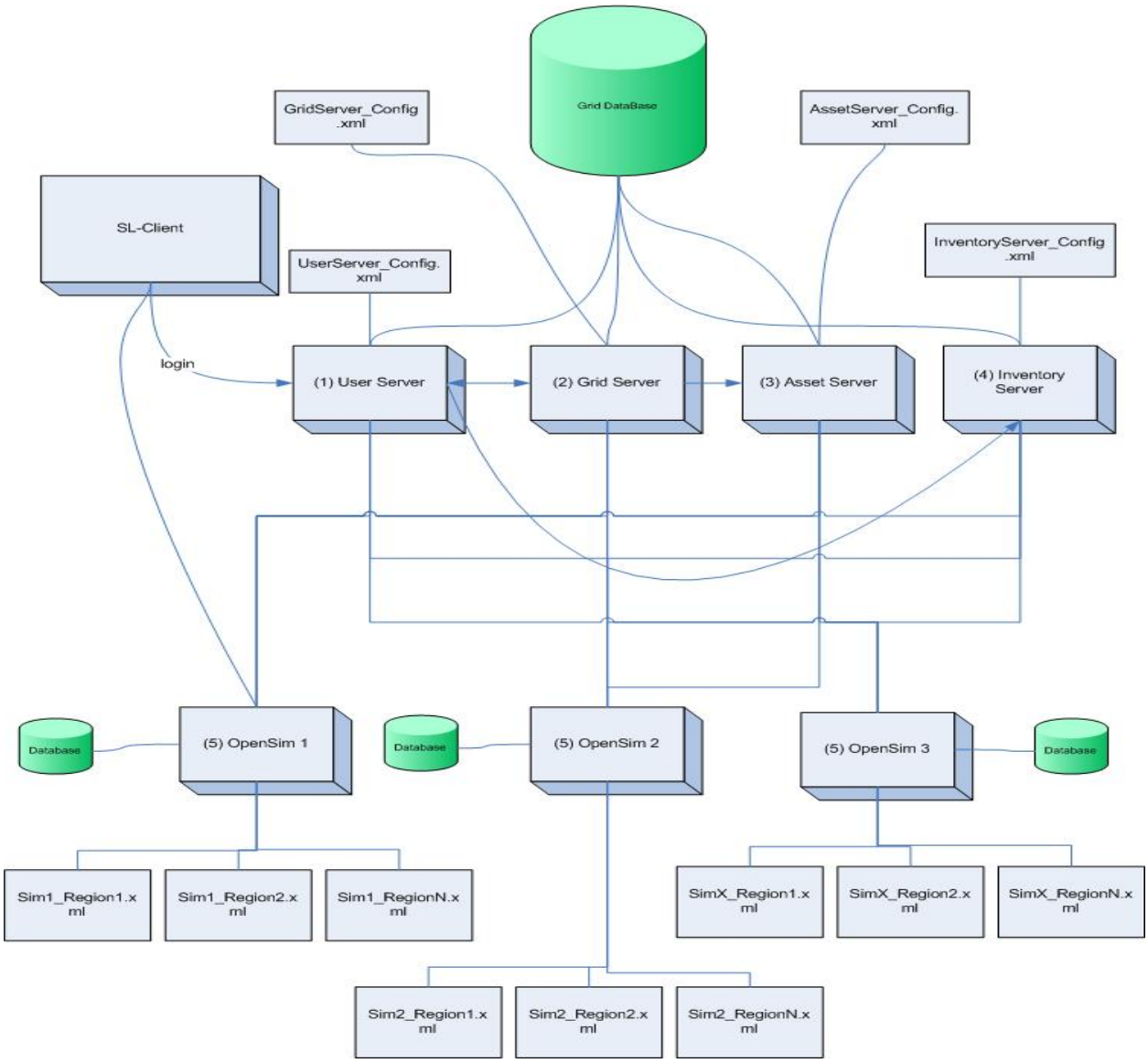

Рисунок 1 – Схема взаимодействия серверов в Opensim

Вся информация о пользователях и об окружающем их мире хранится в различных базах данных, разделенных логически на три типа: Common (Общий), Services (Сервисы), Simulator (Симулятор). В Common содержится всего одна таблица migrations, предназначенная для обновления платформы. Services – содержит настройки аватара, его инвентаря, авторизация пользователя. Simulator – настройки регионов и местности.

На рис. 2 изображена последовательность действий, выполняемых при подключении клиента

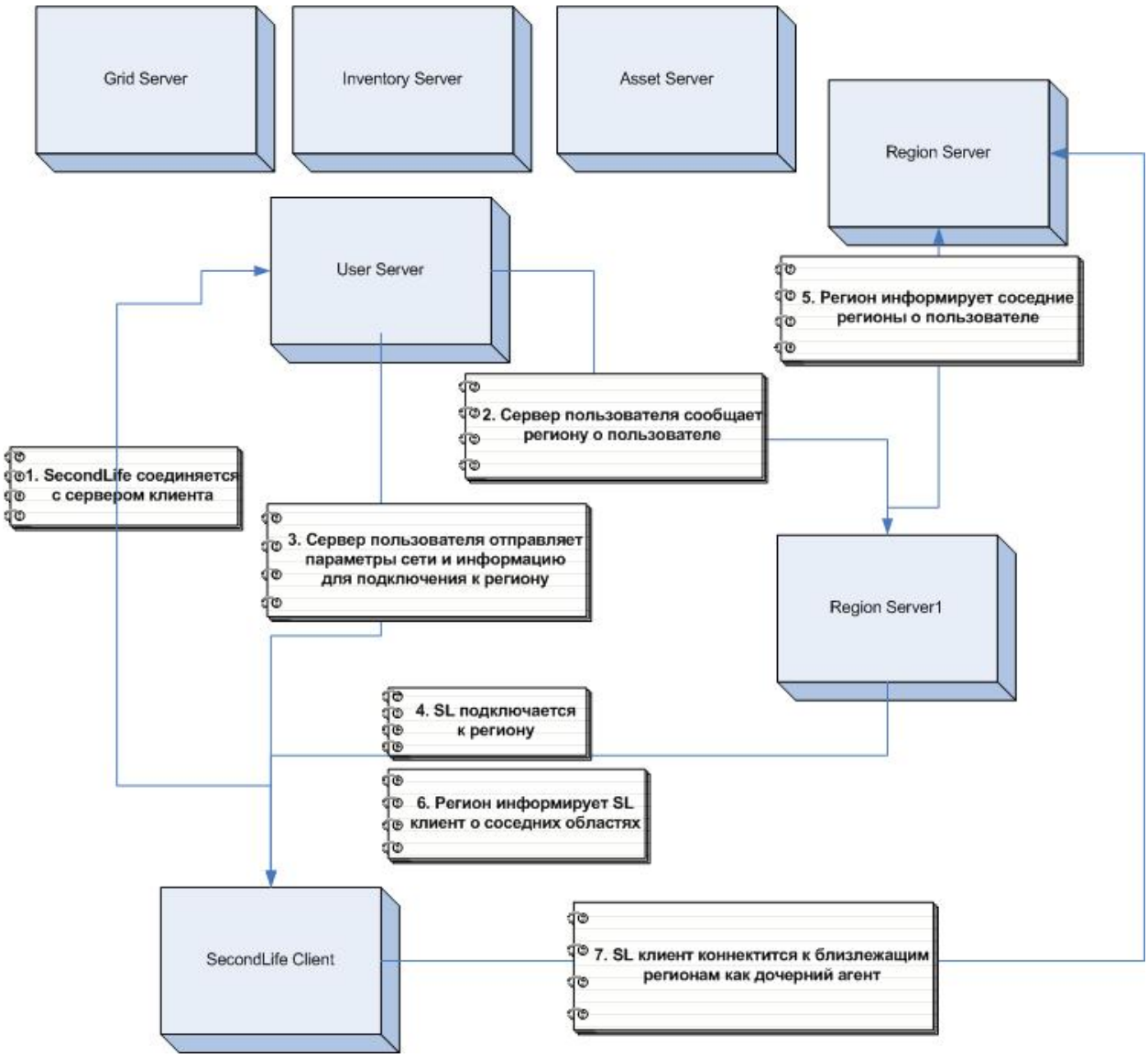

Рисунок 2 – подключение SL клиента к серверу

Как видно из рис. 2 SL клиент подключается к серверу пользователя, который в свою очередь сообщает серверу региона о новом клиенте и отправляет параметры подключения клиенту. Сервер региона собирает информацию о близлежащих регионах и пользователях в них, передает эту информацию SL клиенту. SL подключается к региону и коннектится к близлежащим регионам как дочерний агент.

Для разбора того, какие поля могут находиться в таблицах, были выдраны три таблицы: assets, inventoryfolders, inventoryitems.

- assets - сетка активов, включающая такие вещи как вещи, текстуры и звуки:

1) name – название актива в инвертаре
2) description – описание актива, его предназначение
3) assetType – целое число, которое означает тип актива
4) local – для будущего использования
5) temporary – для будущего использования
6) data – это поле содержит фактические данные актива: двоичное изображение, скрипт или другой тип данных
7) id – идентификатор, который однозначно характеризует актив в системе
8) create_time – дата и время, когда актив был создан
9) access_time - дата и время, когда было последнее изменение
10) asset_flag – флаг
11) CreatorID – идентификатор аватара

- inventoryfolders – папка для инвентаризации связанных деталей:
  1) folderName – название папки
  2) type – целое число, обозначающее тип папки
  3) version – это поле увеличивается, когда в папку добавляется новый элемент
  4) folderId – уникальный идентификатор папки
  5) agented – идентификатор агента, который создал эту папку
  6) parentFolderId – идентификатор папки-родителя.
- inventoryitems – все итемы, находящиеся в инвентаре
  1) assetsID – идентификационный номер актива (итема)
  2) assetsType – тип актива (скрипт, текстура или шаблон)
  3) inventoryName – текст, который будет высвечиваться в названии итема в папке инвентаря у клиента
  4) inventoryDescription – описания назначения итема
  5) inventoryNextPermissions – разрешение, что можно делать с итемом, если его перенести на другой аватар
  6) inventoryCurrentPermissions – разрешение, что можно делать с итемом текущему обладателю
  7) invType –тип итема
  8) CreatorID – идентификатор создателя
  9) salePrice – цена продажи объекта
  10) creationDate – дата создания
  11) avatarID – идентификатор владельца
  12) parentFolderId – идентификатор папки, в которой хранится итем

Второй проект – UEF, представляет многопользовательскую игру, главной целью которой является развитие персонажа. Вся информация об исходном мире хранится в базе данных. Учитывая, что эта игра будет направлена на развитие персонажей, а не общение как в Opensim, то должна быть еще некая ветка развития. Например, персонажи могут поднимать свой уровень, открывая перед собой новые способности и возможности. Данную информацию тоже необходимо хранить в базе данных.

Масштабируемость архитектуры закладывается изначально [2]. Все процессы станут крутиться вокруг баз данных, где будут храниться динамически изменяемые свойства объектов. Причём группы объектов могут выноситься в отдельные базы, которые можно будет переносить на отдельный сервер (или даже кластер) вместе с сопутствующими обработчиками событий. Вплоть до выделения особо нагруженных звёздных систем или высокоразвитых планет, а возможно и крупных (галактического масштаба) объединений, в отдельную базу данных вместе с программами управляющими событиями именно в данной группе объектов.

В главной базе будут храниться общие понятия о мире: информация о пользователе, его инвентарь, достижения. Для создания крупных объектов целесообразней будет выделить отдельную базу для хранения данных о нем, чтобы разгрузить нагрузку на главный сервер. В итоге создание планеты будет сопровождаться созданием комплекта из базы данных и модуля управления природой: освещённость и температура, облака и осадки, наполнение рек, приливы в морях и океанах, циклы активности животных и растений, возникновение и развитие поселений.

Кроме того, в главной базе хранятся все данные об окружающем мире: константы, формулы, простые алгоритмы, статичные и динамичные данные. В системе существует децентрализованная система менеджеров обработки событий. Она запускает экземпляры обработчиков событий в соответствие с потребностью системы (возникновение новых событий) прекращающая их деятельность при простое, либо зависании (бесконечной рекурсии), а также осуществляющая сравнение версий обработчиков событий на устойчивость и эффективность работы. Основная задача менеджеров, распознать возникновение нового события и запуск подходящего обработчика

Выводы

Исследования показали, что существует множество различных конфигураций баз данных для распределенных систем. Не существует конкретного способа создания конфигураций, все зависит от того, с какой целью создается проект [4].

Можно выделить несколько основных типов данных, которые будут храниться в базе:

- данные о пользователе
- данные о текстурах
- данные об отношениях объектов
- данные об объектах

В некоторых системах для достижения хорошей производительности пренебрегают качеством текстур, продуманной логикой и хорошей физикой. Следовательно, базы данных для таких систем могут быть проще в реализации, а нагрузка на систему может снизиться в несколько раз.

Создание новых способов организации баз данных для интерактивных систем являются привлекательными для разработчиков. Каждый новый способ организации может принести новые идеи и технологии, способные упростить конфигурацию баз данных.

Список литературы

1. Виртуальные миры [Электронный ресурс]. http://uef.me/
2. Developer Documentation – OpenSim [Электронный ресурс]. http://opensimulator.org/wiki/Development
3. Виртуальные миры. SecondLife. [Электронный ресурс] http://world2.ru/
4. Graham Morgan. Scalable Massively Online Games. Режим доступа http://www.cs.ncl.ac.uk/publications/trs/papers/888.pdf свободный. – Загл. с экрана. – Яз. англ.