**СУППЗ ПИРАМИДА**

**Слайд 2.** Существует широкий класс вычислительных задач, при решении которых применяется подход распараллеливания по данным. Очень часто вычисления при этом организуются путем запуска множества копий одной и той же последовательной программы с различными вариантами входных параметров.

Для организации подобных вычислений на кафедре был разработан программный комплекс «Пирамида». Пирамида позволяет организовать выполнение множества экземпляров последовательной программы на иерархически организованной вычислительной системе.

В такой системе выделяется центральный сервер управления, которому подчиняются серверы управления кластерами. Каждый кластер управляет множеством ВМ. На ВМ запускаются экземпляры ОПП.

Все множество исходных входных данных Пирамида автоматически разбивает на порции для кластеров. Порции кластеров разбиваются на порции для ВМ. Порции для ВМ разбиваются на порции для экземпляров ОПП.

Размеры порций могут быть произвольными и задаются программистом. Таким образом обеспечивается балансировка нагрузки кластеров и ВМ, даже если они имеют разную производительность.

«Пирамида» повышает отказоустойчивость вычислений. В случае выхода из строя одного или нескольких вычислительных модулей, а также одного или нескольких кластеров, работа перераспределяется на рабочие кластеры и ВМ. При этом расчеты не останавливаются.

**Слайд 3.** Коллективный доступ пользователей к вычислительной установке с возможностью использования на ней «Пирамиды» позволяет организовать система управления прохождением параллельных заданий (СУППЗ).

СУППЗ выполняет прием входного потока пользовательских заданий, осуществляет ведение очереди заданий. Обеспечивает распределение и постановку заданий на выполнение. Выделение и освобождение ресурсов вычислительной установки для заданий.

**Слайд 4.** Задача совмещения Пирамиды и СУППЗ решена ранее, в ходе дипломной работы выпускника 732 кафедры Кондратьева.

Для каждого задания СУППЗ динамически выделяет вычислительные модули. В случае заданий для Пирамиды автоматически конфигурируется иерархическая вычислительная система.

**Слайд 5.** Актуальной задачей является разработка прозрачного веб-интерфейса к ПК «Пирамида». Интерфейс должен быть построен таким образом, чтобы избавить пользователя от необходимости изучения работы ПК «Пирамида» и СУППЗ при постановке заданий на обработку.

Кроме этого, возможности «Пирамиды» по организации вычислений с распараллеливанием по данным делают целесообразным построение на его базе специализированного облачного сервиса вида SaaS.

Разработка подобного облачного сервиса и является целью настоящей дипломной работы.

Предоставление «Пирамиды» в виде облачного сервиса с веб-интерфейсом позволит снизить порог вхождения пользователей В настоящее время, пользователи «Пирамиды» вынуждены вручную выполнять подготовительные действия по составлению паспорта заданий и конфигурации системы. При выполнении задания на нескольких комплексах «Пирамида» подготовительные действия фактически дублируются для каждого комплекса.

В распоряжении организации предоставляющей ПК «Пирамида» как сервис может быть несколько вычислительных установок. При этом они могут быть распределены территориально. Облачный сервис в этом случает должен выполнять роль дополнительного уровня абстракции, позволяющего объединить вычислительные установки и организовать к ним единый интерфейс управления

**ТРЕБОВАНИЯ**

**Слайд 6.** Разобьем общую задачу сервиса на составляющие подзадачи.

Общей задачей облачного сервиса является организация многопользовательского доступа к ПК «Пирамида» посредством совместно работающих СУППЗ. Выполнение этой задачи облачным сервисом является функциональным требованием бизнес уровня.

**Слайд 7.** Для роли «пользователь» облачный сервис должен предоставлять следующие интерфейсы:

* регистрации и авторизации пользователя;
* формирования шаблонов заданий;
* отслеживания состояния и управления заданиями.

Для роли «администратор» должен быть предоставлен интерфейс управления учетными записями пользователей.

**Слайд 8.**

* Доступ к сервису осуществляется посредством web-интерфейса и возможен только для авторизованных пользователей;
* пользователи должны быть изолированы друг от друга, т.е. работа пользователя не влияет на работу других пользователей.

**Слайд 9.**

* Сервис должен отслеживать состояние соединений с СУППЗ и производить запуск заданий по доступным соединениям;
* Взаимодействие сервиса с вычислительными установками должно осуществляться через ssh-соединение.

**АНАЛИЗ ГОТОВЫХ РЕШЕНИЙ**

**Слайд 10.** Мной был проведен анализ готовых решений построения облачных сервисов. Как правило, сервисы вида SaaS строятся на основе PaaS или IaaS решений.

Проанализированные готовые решения направлены на формирование сервисов общего назначения. Они во многом избыточны, но ни одно не содержит готовых модулей, позволяющих организовать взаимодействие облачного сервиса с установленной на вычислительных установках СУППЗ по протоколу SSH. Такие модули необходимо было разрабатывать самостоятельно.

Поэтому было принято решение разрабатывать облачный сервис без использования готовых решений.

**ИНСТРУМЕНТЫ**

**Слайд 11.** При выборе инструментальных средств разработки облачного сервиса я руководствовался следующими критериями:

* новизна и перспективность технологий;
* практика использования в известных, крупных проектах;
* полнота документации и объем сообщества разработчиков.

ОС состоит из разработанных мной клиентской и серверной частей приложения.

Клиентская часть функционирует в браузере пользователя. Пользовательский интерфейс разработан с использованием технологий: HTML5 и CSS3. Логика клиентского приложения написана на языке программирования JavaScript в рамках шаблона разработки Angularjs.

Модуль Angular-device средствами одноименной JavaScript библиотеки организует клиентские функции механизма аутентификации.

Серверная часть функционирует на сервере организации, предоставляющей Пирамиду, как сервис.

Логика серверной части разработана на языке программирования Ruby с использованием шаблона разработки веб-приложений Ruby on Rail. Ruby библиотека Passenger организует взаимодействие RoR-приложения c веб-сервером Apache.

В качестве СУБД для хранения и обработки пользовательских данных выбрана MySQL.

Работа с реляционной БД ведется через объектное представление ActiveRecord.

Взаимодействие с вычислительными установками по протоколу SSH организовано при помощи Ruby-библиотеке Net-ssh-shell.

Механизм аутентификации основан на библиотеке Devise.

**СТРУКТУРА**

**Слайд 12.** Взаимодействие клиентской и серверной частей строится в соответствии с архитектурой RESTful. Данные, которыми обмениваются клиент и сервер организованы в единый формат JSON.

*Сервисы REST API* клиентской части сервиса преобразуют запросы контроллеров в формат JSON и производят обмен данными с сервером.

*Контроллеры REST API* серверной части сервиса в ответ на запрос клиента производят необходимые действия с БД или СУППЗ, посредством контроллера ssh-соединений, и отправляют ответ клиенту в формат JSON.

**Демонстрация**

Перед вами интерфейс администратора ОС.

Страница управления ssh-соединениями.

Администратор может создавать новые соединения. Проверять работоспособность созданных ранее. И удалять соединения.

На странице управления соединениями пользователей администратор может назначить пользователям доступные соединения. Каждое соединение дает возможность выполнения заданий на отдельной СУППЗ.

Перейдем к интерфейсу пользователя.

На странице шаблоны пользователь может создавать новые шаблоны заданий, установив параметры. Это могут быть параметры команды запуска Пирамиды. Либо параметры паспорта задания Пирамиды.

Пользователь может редактировать и удалять шаблоны заданий. А так же запускать задание по шаблону.

Запустим 3 задания по разным шаблонам.

Одно задание встало на выполнение, 2 находятся в очереди.

Посмотрим детали задания на выполнении.

Это наименование задания. Статус. Остаток заказанного времени выполнения. Текущее время выполнения. Прогнозируемый остаток времени выполнения. Часть выполненной работы в процентах. Структура иерархической вычислительной системы с порциями данных. Состояние задания. И результаты.

С этой страницы пользователь может остановить выполнение задания.

Результаты сохраняются на сервисе.

Для задания в очереди пользователь может посмотреть время до запуска. Или удалить из очереди.

На странице задания пользователь также может останавливать задания, удалять их из очереди и удалять результаты завершенных заданий.

Наблюдаем визуализацию процесса вычислений в динамике.

Визуализируется выполнение одного задания в ускоренном виде.

В результате дипломной работы разработан ОС на базе ПК Пирамида и СУППЗ, позволяющий в динамике визуализировать процесс вычислений и управлять пользовательскими заданиями.