Оглавление

[Введение 2](#_Toc406738008)

[ГЛАВА 1. ОБЩИЕ ПРИНЦИПЫ И ТЕХНОЛОГИИ РАЗРАБОТКИ ОБЛАЧНЫХ СЕРВИСОВ 5](#_Toc406738009)

[1.1. Задача построения облачного сервиса на базе программных комплексов «Пирамида» и СУППЗ 5](#_Toc406738010)

[1.2. Стандарты и руководства по использованию облачных вычислений 6](#_Toc406738011)

[1.3. Требования к облачному сервису на базе программных комплексов «Пирамида» и СУППЗ 10](#_Toc406738012)

[1.4. Существующие открытые решения, реализующие управление облачными вычислениями 12](#_Toc406738013)

[1.5. Стенд разработчика ПК «Пирамида» под управлением СУППЗ 17](#_Toc406738014)

[1.5.1. Механизм совмещения ПК «Пирамида» с СУППЗ 17](#_Toc406738015)

[1.5.2. Стенд разработчика ПК «Пирамида» под управлением СУППЗ 19](#_Toc406738016)

[1.5.3. Опытная эксплуатация ПК «Пирамида» в составе СУППЗ 20](#_Toc406738017)

[Выводы по главе 1 22](#_Toc406738018)

# Введение

В последние годы значительно увеличилось число пользователей, решающих прикладные задачи на параллельных вычислительных системах. При этом далеко не все пользователи обладают достаточными навыками программирования для эффективной организации параллельных вычислений.

Во многих случаях последовательные задачи таких пользователей можно с удовлетворительной эффективностью в автоматическом режиме привести к виду параллельных задач. При этом не потребуется помощь специалиста по параллельному программированию.

Сотрудниками кафедры разработан программный комплекс «Пирамида», представляющий собой систему, позволяющую организовать параллельное выполнение последовательной программы с распараллеливанием по данным посредством запуска множества ее экземпляров на иерархически организованной вычислительной системе. «Пирамида» сокращает время планирования вычислений и распределения ресурсов, а также повышает отказоустойчивость вычислений. Таким образом, комплекс позволяет ускорить процесс разработки параллельной программы пользователем, не обладающим достаточными навыками в области программирования с организацией параллельных вычислений.

Использование ПК «Пирамида» на реальных вычислителях вызывает проблему организации многопользовательского доступа к комплексу. Решением такой проблемы является сопряжение ПК «Пирамида» с системой управления прохождением параллельных задач (СУППЗ). СУППЗ – совместная разработка Института прикладной математики им. М.В. Келдыша РАН и МСЦ РАН, которая обеспечивает выделение и освобождение ресурсов вычислительной установки, а также ведение очередей на запуск задач.

Однако, остается не решенной задача совместной работы нескольких комплексов СУППЗ и организации многопользовательского доступа к ПК «Пирамида» в их составе.

Еще одной 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.

Целью настоящей дипломной работы является разработка макета облачного сервиса вида SaaS на базе программного комплекса «Пирамида» и СУППЗ.

В рамках дипломной работы подлежат выполнению следующие задачи:

 изучение и практическое освоение ПК «Пирамида» и СУППЗ, создание стенда разработчика, организация работы ПК «Пирамида» под управлением СУППЗ на созданном стенде;

 разработка web-интерфейса ПК «Пирамида», позволяющего в динамике визуализировать процесс вычислений;

 разработка требований к облачному сервису, аналитический обзор существующих решений для построения облачных сервисов;

 проектирование облачного сервиса, в том числе разработка web-интерфейса;

 реализация проекта в виде макета облачного сервиса, установленного на стенде разработчика;

 разработка программной документации на созданный макет.

Настоящая дипломная работа изложена в пояснительной записке, состоящей из трех глав.

Первая глава пояснительной записки посвящена изучению и практическому освоению работы ПК «Пирамида» под управлением СУППЗ. Изучены существующие открытые решения и средства для построения облачных сервисов. Сформированы требования к облачному сервису.

Вторая глава отражает процесс проектирования и разработки облачного сервиса.

В третьей главе пояснительной записки описан процесс проведения опытной эксплуатации облачного сервиса, рекомендации администратору и пользователю сервиса.

# ГЛАВА 1. ОБЩИЕ ПРИНЦИПЫ И ТЕХНОЛОГИИ РАЗРАБОТКИ ОБЛАЧНЫХ СЕРВИСОВ

## 1.1. Задача построения облачного сервиса на базе программных комплексов «Пирамида» и СУППЗ

В настоящее время наблюдается бурное развитие технологий облачных вычислений. Пользователям предоставляются различные услуги на удаленных серверах их организации, либо стороннего поставщика облачных услуг. К таким услугам можно отнести предоставление программного обеспечения, вычислительных мощностей, инфраструктур и др. Доступ осуществляется, как правило, посредством веб-интерфейса, адаптированном под конкретные задачи.

Предоставление ПК «Пирамида» в виде облачного сервиса с веб-интерфейсом позволит снизить порог вхождения пользователей. В настоящее время пользователи ПК «Пирамида» 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. Веб-интерфейс облачного сервиса позволит в упрощенном виде производить составление паспорта задания и конфигурирование системы.

С точки зрения системной организации облачного сервиса необходимо решить следующие задачи.

В распоряжении организации предоставляющей ПК «Пирамида» как сервис может быть несколько вычислительных установок. При этом вычислительные установки 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 (на рис. 1.1. Контроллер вычислений).

Общая структура облачного сервиса приведена на рис 1.1. Подробные требования пользовательского и системного уровней приведены в п. 1.2.



Рис. 1.1. Общая структура облачного сервиса

## 1.2. Стандарты и руководства по использованию облачных вычислений

Для широкого и эффективного внедрения технологий нужны методические и нормативные документы, разъясняющие, правовые рамки применения этих технологий, имеющиеся проблемы и риски и способы их минимизации. Облачные технологии - не исключение. Однако, многие стандарты, которые сегодня применяются к облачным вычислениям, были разработаны для «дооблачных» технологий, таких как веб-сервисы и Интернет. Поэтому сейчас идет активная разработка стандартов и руководств, предназначенных именно для облачных вычислений.

В 4-ом квартале 2014 года опубликованы первые стандарты ISO:

* ISO/IEC 17788 «Информационные технологии - Облачные вычисления - Общие положения и словарь» (Information technology - Cloud Computing - Overview and Vocabulary);
* ISO/IEC 17789 «Информационные технологии - Облачные вычисления - Эталонная архитектура» (Information technology - Cloud Computing - Reference Architecture).

Стандарты ISO еще не успели войти в мировую практику. В настоящее время используются рекомендации Национального Института стандартов и технологий (NIST). Ниже приведены определения, характеристики и виды облачных вычислений. Дальнейшая разработка облачного сервиса будет опираться на эту терминологию.

Облачные вычисления (cloud computing) - это модель предоставления повсеместного, удобного сетевого доступа «по-требованию» к разделяемому пулу конфигурируемых вычислительных ресурсов (например, сети, серверы, память, приложения и сервисы), которые могут быть предоставлены и освобождены в короткие сроки с минимальными усилиями в управлении или с минимальным взаимодействием с поставщиком услуги. Такая облачная модель включает в себя пять основных характеристик, три модели сервиса и четыре модели развёртывания.

Основные характеристики:

1. Самообслуживание «по-требованию». Потребители могут в одностороннем порядке получить вычислительные ресурсы (такие как серверное время или сетевое хранилище) когда это требуется автоматически без необходимости взаимодействия с каждым сетевым провайдером.
2. Общий сетевой доступ. Вычислительные ресурсы доступны повсюду в сети, доступ осуществляется при помощи стандартных механизмов, которые содействуют использованию разнородных тонких и толстых клиентских платформ (например, мобильные телефоны, планшеты, настольные компьютеры и рабочие станции).
3. Объединение ресурсов. Вычислительные ресурсы провайдера объединяются для обслуживания множества потребителей, используя многопользовательскую модель с различными физическими и виртуальными ресурсами, назначаемыми и переназначаемыми динамически в зависимости от запросов потребителя. Складывается ощущение физической независимости, в котором потребители, главным образом, не контролируют или не знают о настоящем местоположении предоставляемых ресурсов, но местоположение может быть указано на высоком уровне абстракции (например, страна, штат или центр обработки данных). Примеры таких ресурсов включают хранилища данных, процессорное время, память и пропускную способность сети.
4. Быстрая эластичность. Вычислительные ресурсы могут быть эластично переданы и освобождены автоматически на некотором этапе для быстрого увеличения и уменьшения, соизмеримо с требованиями потребителя. Для потребителя вычислительные возможности, пригодные для резервирования, часто кажутся неограниченными и могут быть выделены в любом количестве в любое время.
5. Существование сервиса мониторинга. Облачные системы автоматически контролируют и оптимизируют ресурсы, используя измерение вычислительных ресурсов на некотором уровне абстракции в зависимости от типа обслуживания (например, хранилище данных, время на обработку, пропускная способность и активные аккаунты пользователей). Информация о использующихся ресурсах может собираться, контролироваться, формироваться в отчёт, предоставляя прозрачность как для провайдера, так и для потребителя использованного сервиса.

Модели сервиса:

1. Программное обеспечение как сервис (Software as a Service, SaaS). Вычислительные ресурсы, предоставляемые потребителям, используют приложения провайдеров, которые запущены в облачной инфраструктуре. Приложения доступны с различных устройств клиентов через тонкий и толстый интерфейс пользователя такой, как веб-браузер (например, электронная почта) или программный интерфейс. Потребители не могут управлять и контролировать лежащую в основе облака инфраструктуру, включая сеть, серверы, операционные системы, хранилища данных или возможности конкретного приложения с ограничением специфических установок конфигурации приложения.
2. Платформа как сервис (Platform as a Service, PaaS). Вычислительные ресурсы, предоставляемые потребителям, базируются на облачной инфраструктуре, созданной потребителем, или полученных приложениях, созданных с использованием языков программирования, библиотек, инструментов, поддерживаемых провайдером. Потребители не управляют и/или не контролируют лежащую в основе облака инфраструктуру, включая сеть, сервера, операционные системы или хранилища данных, но они контролируют другие разворачиваемые приложения и возможные настройки конфигурации для среды размещённых приложений.
3. Инфраструктура как сервис (Infrastructure as a Service, IaaS). Вычислительные ресурсы, предоставляемые потребителям, - это время обработки, хранилища данных, сети и другие фундаментальные компьютерные ресурсы, с помощью которых потребители могут развёртывать и запускать произвольное программное обеспечение, которое может включать операционные системы и приложения. Потребители не могут управлять или контролировать лежащую в основе облака инфраструктуру, но имеют контроль над операционными системами, хранилищами данных и развёртываемыми приложениями.

Модели развертывания:

1. Частное облако (Private cloud): Облачная инфраструктура функционирует целиком в целях обслуживания одной организации. Инфраструктура может управляться самой организацией или третьей стороной и может существовать как на стороне потребителя (on premise) так и у внешнего провайдера (off premise).
2. Облако сообщества или общее облако (Community cloud): Облачная инфраструктура используется совместно несколькими организациями и поддерживает ограниченное сообщество, разделяющими общие принципы (например, требования к безопасности, политики, требования к соответствию регламентам и руководящим документам). Такая облачная инфраструктура может управляться самими организациями или третьей стороной и может существовать как на стороне потребителя (on premise) так и у внешнего провайдера (off premise).
3. Публичное облако (Public cloud): Облачная инфраструктура создана в качестве общедоступной или доступной для большой группы потребителей не связанной общими интересами, но, например, принадлежащих к одной области деятельности. Такая инфраструктура находится во владении организации, продающей соответствующие облачные услуги/ предоставляющей облачные сервисы.
4. Гибридное облако (Hybrid cloud): Облачная инфраструктура является композицией (сочетанием) двух и более облаков (частных, общих или публичных), остающихся уникальными сущностями, но объединенными вместе стандартизированными или проприетарными технологиями, обеспечивающими портируемость данных и приложений между такими облаками (например, такими технологиями, как пакетная передача данных для баланса загрузки между облаками).

## 1.3. Требования к облачному сервису на базе программных комплексов «Пирамида» и СУППЗ

Общей задачей облачного сервиса является организация многопользовательского доступа к ПК «Пирамида» посредством совместно работающих комплексов СУППЗ. Выполнение этой задачи облачным сервисом (далее CS – Cloud Services) является функциональным требованием бизнес уровня.

Разобьем общую задачу на составляющие подзадачи, определяющие требования к CS в соответствии с классификацией, приведенной на рис. 1.2.



Рис. 1.2. Классификация требований к облачному сервису

Функциональные требования пользовательского уровня для роли «пользователь»:

* CS должен предоставлять интерфейсы регистрации и авторизации пользователя;
* CS должен предоставлять интерфейс формирования задач пользователя;
* CS должен предоставлять информацию о доступных вычислительных ресурсах;
* CS должен предоставлять интерфейс отслеживания состояния и управления задачами пользователя;
* CS должен предоставлять интерфейс биллинговой подсистемы пользователя;
* CS должен предоставлять интерфейс доступа к личной папке пользователя;
* CS должен предоставлять интерфейс управления учетной записью пользователя.

Функциональные требования пользовательского уровня для роли «администратор»:

* CS должен предоставлять интерфейс управления учетными записями пользователей;
* CS должен предоставлять интерфейсы мониторинга состояний вычислителей и задач;
* CS должен предоставлять интерфейсы управления вычислителями и задачами.

Нефункциональные требования пользовательского уровня:

* доступ к CS возможен только для авторизованных пользователей;
* доступ к CS осуществляется посредством web-интерфейса;
* пользователи должны быть изолированы друг от друга – пользователь должен иметь доступ только к своей личной папке, только к своей учетной записи и только к своим задачам в очереди, либо на выполнении.

Требования системного уровня:

* CS должен отслеживать загруженность комплексов СУППЗ и производить балансировку нагрузки;
* Взаимодействие CS с вычислительными установками должно осуществляться через ssh соединение.

## 1.4. Существующие открытые решения, реализующие управление облачными вычислениями

В соответствии с эталонной архитектурой облачных вычислений, предложенной NIST, сервис вида SaaS может быть построен на основе более гибких PaaS и IaaS решений. Изучение автором работы открытых решений облачных сервисов привело к выводу, что все ведущие SaaS решения построены именно таким образом.

На рис. 1.3 и 1.4 приведены передовые открытые решения облачных сервисов вида PaaS и IaaS соответственно. Все они имеют схожую архитектуру, соответствующую руководствам NIST по построению платформ облачных вычислений.

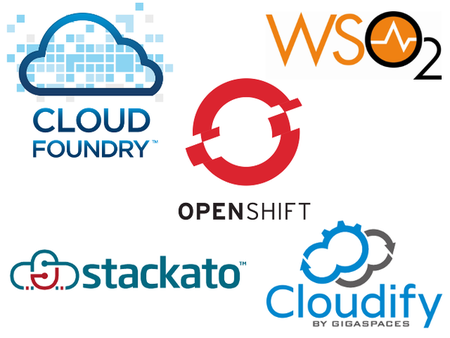


Рис. 1.3. Открытые решения PaaS

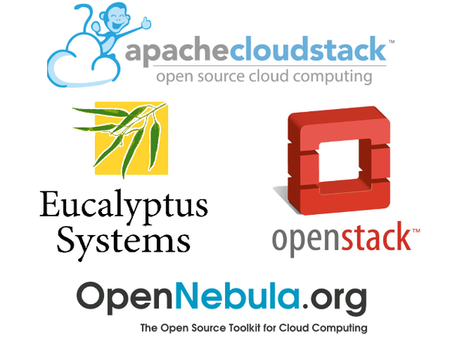


Рис. 1.4. Открытые решения IaaS

В качестве показательного примера построения облачного сервиса рассмотрим решение OpenStack, как наиболее развитое и гибкое.

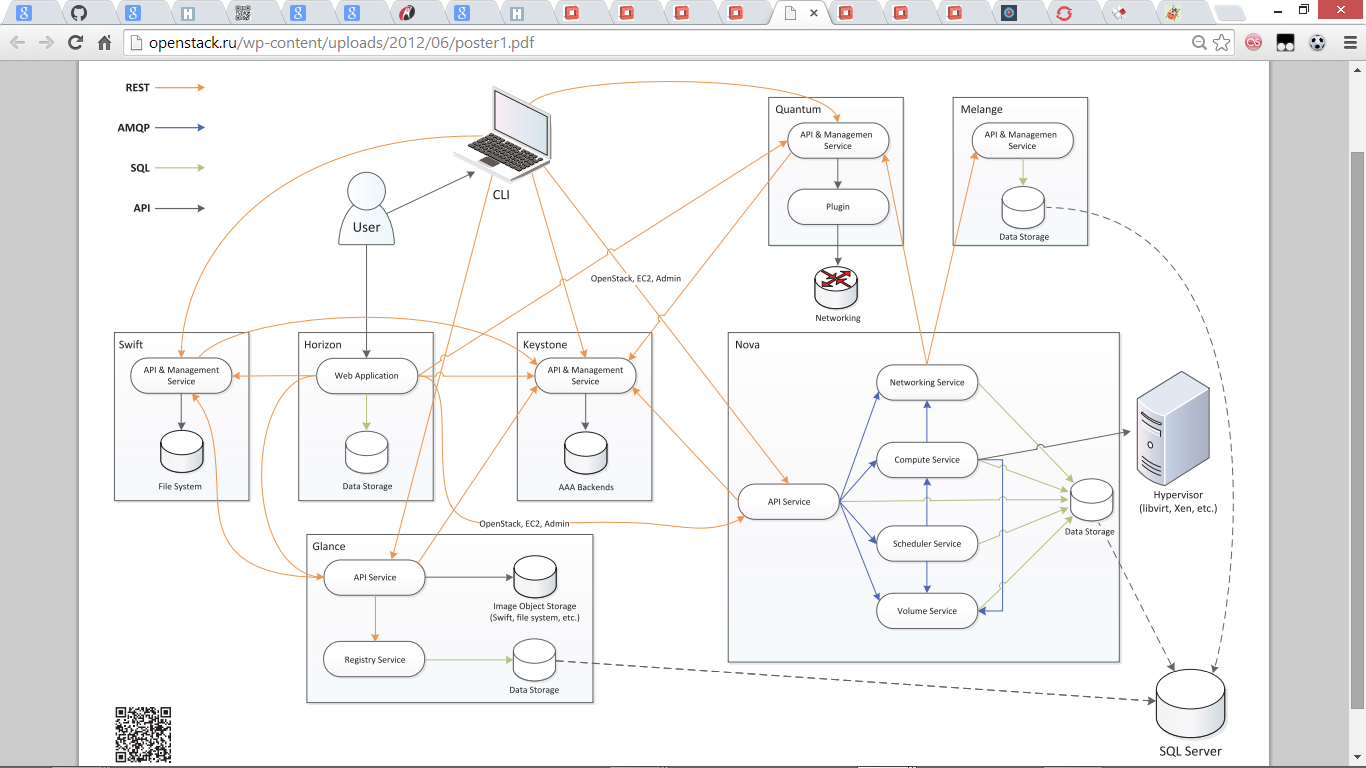
Основные структурные элементы сервиса (рис. 1.5):

* Nova (Compute) - контроллер, управляющий работой виртуальных машин;
* Swift (OpenStack Object Storage) - распределенное хранилище;
* Horizon - инструмент, через который можно осуществлять взаимодействие с различными сервисами OpenStack;
* Glance - библиотека образов виртуальных машин;
* Neutron – компонент для подключения к сети;
* Cinder – сервис блочного хранения данных;
* KeyStone – компонент, который предоставляет услуги идентификации;

Основной компонент OpenStack — это Nova (Compute), контроллер, управляющий работой виртуальных машин. Nova сосредоточена на таких функциях, как обработка запросов на создание виртуальных машин, соединение их с внешним миром, контроль за работоспособностью и распределением нагрузки на физические машины и каналы связи, реакция на сбои и т.д. В основе Nova лежит код системы NASA Nebula, язык программирования Python и протокол обмена сообщениями AMQP.

В системе существует восемь обособленных компонентов:

* контроллер Облака (Cloud Controller) следит за состоянием системы и является связующим звеном всех остальных компонентов;
* сервер API (API Server) реализует web-интерфейс, позволяющий управлять контроллером облака;
* контроллер вычислений (Compute Controller) отвечает за запуск виртуальных машин и их связь со всей остальной инфраструктурой;
* хранилище (Object Store) предоставляет сервис хранения данных, совместимый с Amazon S3;
* менеджер аутентификации (Auth Manager) предоставляет сервисы аутентификации и авторизации;
* контроллер томов (Volume Controller) дает возможностьподключать виртуальные устройства хранения к виртуальным машинам;
* сетевой контроллер (Network Controller) создает виртуальные сети, позволяя виртуальным машинам взаимодействовать друг с другом и с внешней сетью;
* планировщик (Scheduler) ответственен за выбор подходящего контроллера вычислений для запуска новой виртуальной машины.



SQL server

Рис. 1.5. Архитектура OpenStack

Пользователь при помощи клиентских инструментов инициирует запрос на создание виртуальной машины к серверу API. Далее происходит аутентификация и авторизация пользователя, после чего сервер API разбирает запрос и посылает его контроллеру облака. Последний инициирует три новых запроса: к сетевому контроллеру, к хранилищу и к планировщику. Сетевой контроллер выделяет IP-адрес для новой виртуальной машины и возвращает его контроллеру облака. В сетевом хранилище происходит поиск подходящего образа жесткого диска для будущей виртуальной машины, адрес которого возвращается контроллеру облака. Имея все необходимое для создания новой виртуальной машины, контроллер облака посылает запрос планировщику, который выбирает наиболее подходящий контроллер вычислений и отдает ему запрос на создание ВМ. После того, как новая ВМ будет запущена, контроллер облака завершает свою работу и сообщает серверу API об успехе всей операции. Теперь пользователь может подключиться к ВМ, а всю деятельность по поддержанию ее в работоспособном состоянии, выделении дискового пространства, маршрутизации сетевых пакетов и прочую работу система берет на себя.

## 1.5. Стенд разработчика ПК «Пирамида» под управлением СУППЗ

Перед началом разработки облачного сервиса необходимо разработать испытательный стенд ПК «Пирамида» под управлением СУППЗ, который будет выступать в качестве вычислительной установки для проведения отладки сервиса.

### 1.5.1. Механизм совмещения ПК «Пирамида» с СУППЗ

Совмещение ПК «Пирамида» с СУППЗ позволяет организовать коллективный доступ пользователей к вычислительной установке с возможностью использования на ней ПК «Пирамида» при организации вычислений.

Механизм сопряжения СУППЗ с прикладными программными системами используется следующим образом:

* пользователь запускает командный файл epkrun, на вход которому подает паспорт задания для ПК «Пирамида». Команда epkrun анализирует свои параметры запуска, составляет паспорт задания для СУППЗ и выполняет постановку паспорта в очередь;
* в процессе выполнения файла runmvs.bat при постановке задачи на счет, на основе выделенных вычислительных модулей генерируется конфигурационный файл ПК «Пирамида», в котором выделенные модули выстраиваются в виде иерархической древовидной структуры;
* после генерации конфигурационного файла, на выделенных модулях происходит последовательный запуск исполняемых файлов ПК «Пирамида» в соответствие с составленной в конфигурационном файле иерархической структурой.

Полная схема представлена на рис. 1.1.

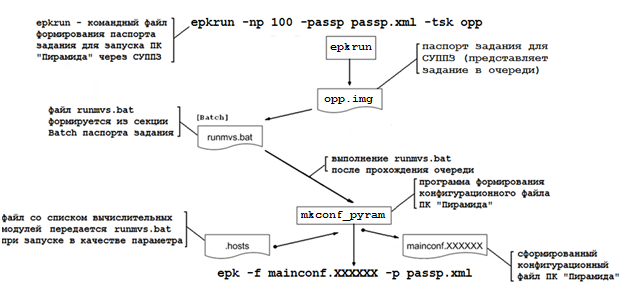


Рис. 1.6. Схема сопряжения ПК «Пирамида» и СУППЗ

При вызове командного файла epkrun происходит формирование паспорта задачи и постановка его в очередь. Из секции Batch паспорта генерируется файл runmvs.bat, который на момент запуска получает файл со списком выделенных вычислительных модулей. В процессе запуска задачи файл runmvs.bat, на основе полученного списка формирует конфигурационный файл запуска ПК «Пирамида» и производит запуск командного файла центрального менеджера epk.

В общем случае, для каждой конкретной задачи, на вычислителе разворачивается собственный экземпляр ПК «Пирамида», на котором производится вся вычислительная работа.

### 1.5.2. Стенд разработчика ПК «Пирамида» под управлением СУППЗ

Стенд представляет из себя набор виртуальных машин под управлением VM Ware. Состав стенда:

* центральный сервер “head” (ubuntu-14.10-server-i386);
* 3 узла: “node1”, “node2”, “node3” (ubuntu-14.10-server-i386).

Сетевая инфраструктура настроена в соответствии с правилами, описанными в документации по СУППЗ и «Пирамида». На рис. 1.6 отображены адреса машин в сети.

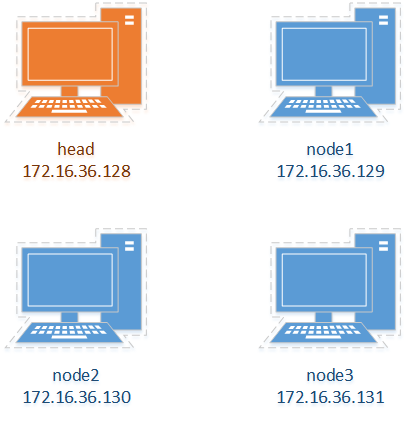


Рис. 1.7. IP – адреса сети испытательного стенда

Все файлы СУППЗ и «Пирамида» хранятся в сетевой папке /common центрального сервера. Также в ней находится ПО организующее работу «Приамида» в составе СУППЗ. К папке имеют доступ все узлы. Задания хранятся в домашнем каталоге пользователя user на каждой машине.

### 1.5.3. Опытная эксплуатация ПК «Пирамида» в составе СУППЗ

Запуск задачи на счет производится пользователем с логином user из каталога test, домашней директории, в которой находится исполняемый файл последовательной программы opp и предназначенный для нее паспорт задания example.ini ПК «Пирамида» (Рис. 1.8). Пользователь при этом находится на управляющей машине supz с адресом 172.16.32.128.

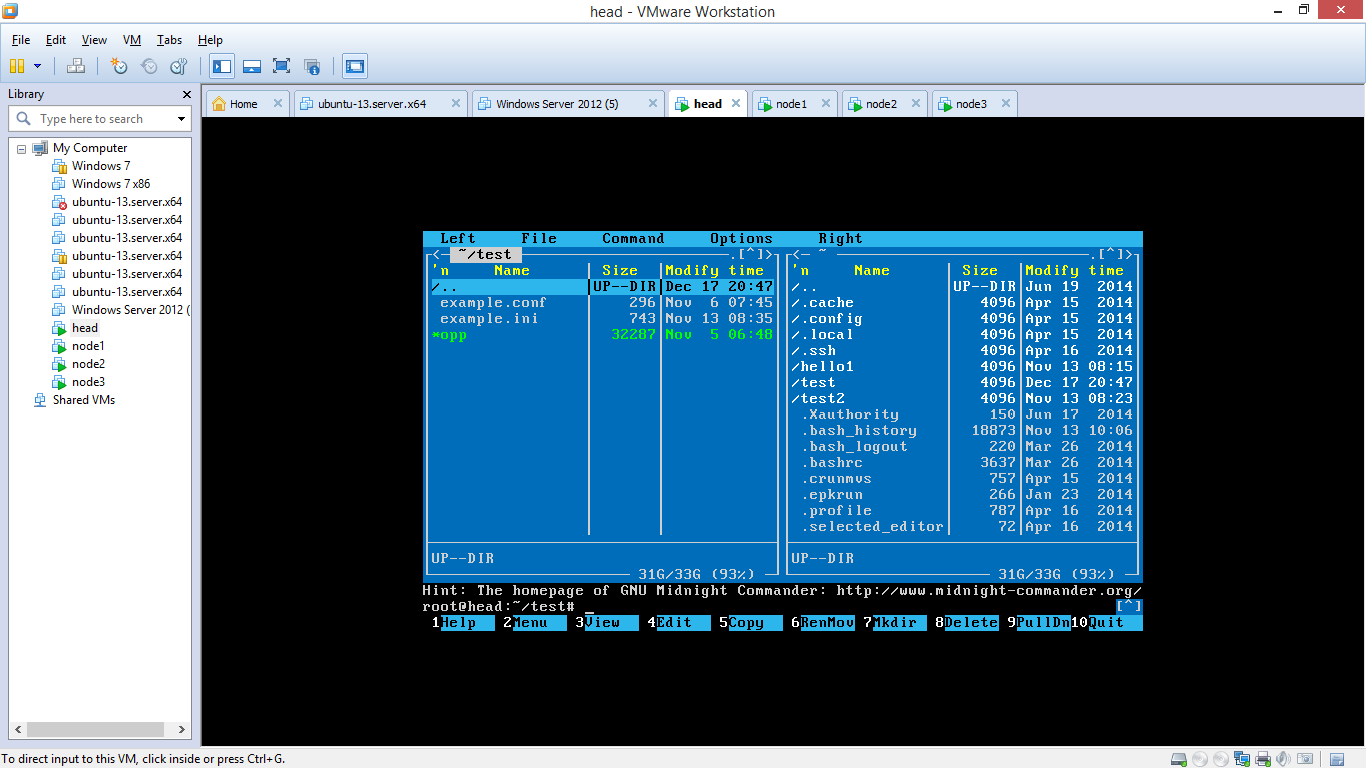


Рис. 1.8. Каталог задания

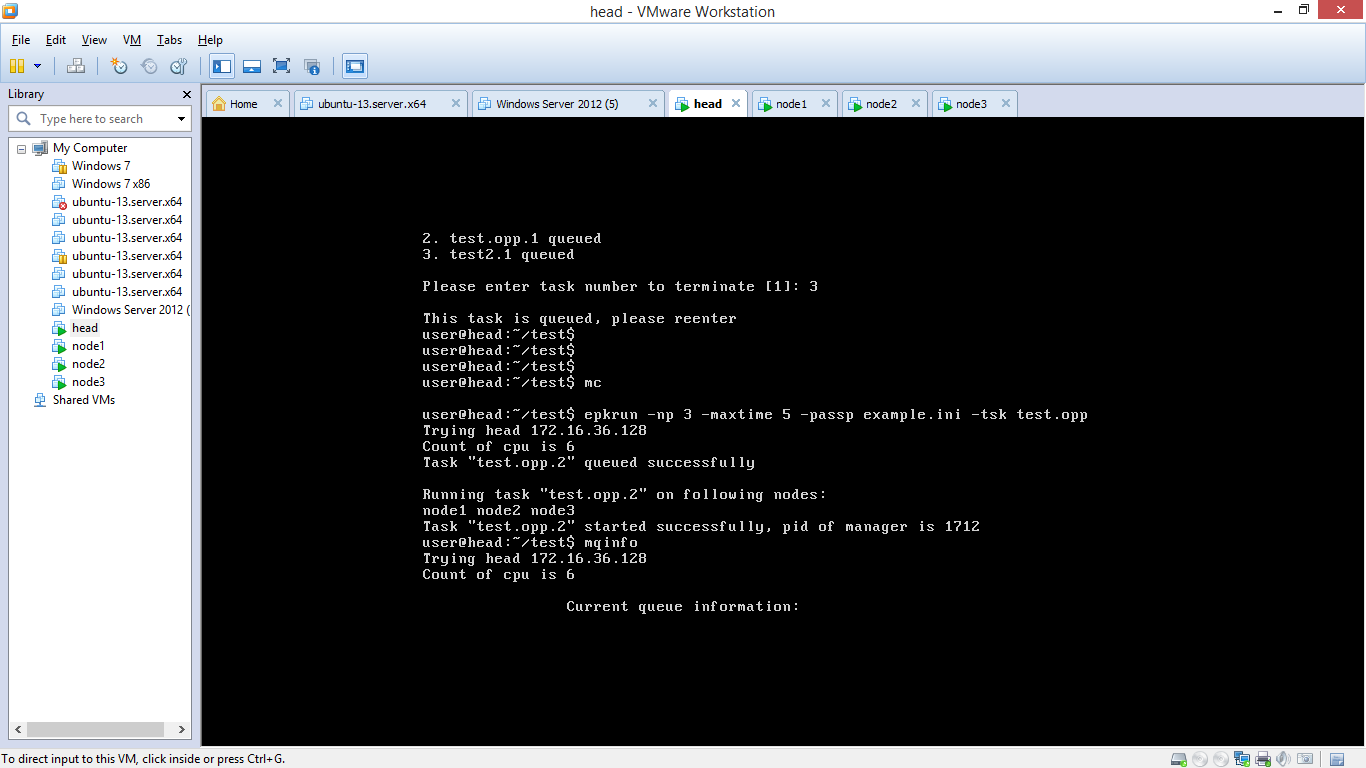


Рис. 1.9. Запуск ПК «Пирамида» через СУППЗ

Состояния очереди после запуска задачи представлено на рис. 1.10. Проверка состояния очереди осуществляется путем введения команды mqinfo.

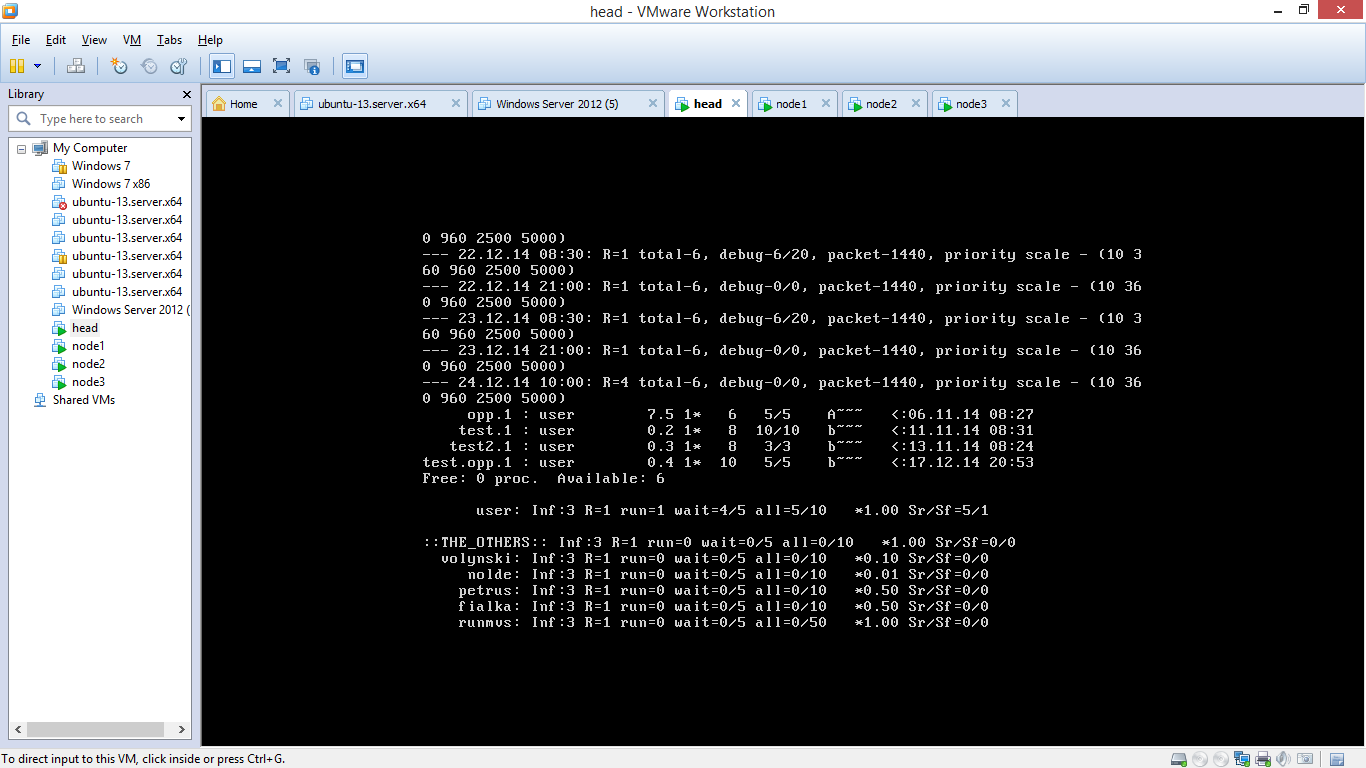


Рис. 1.10. Состояние очереди после запуска задачи

К моменту запуска ПК «Пирамида» был сформирован конфигурационный файл запуска с именем mainconf.UvLvAd (рис. 1.11).

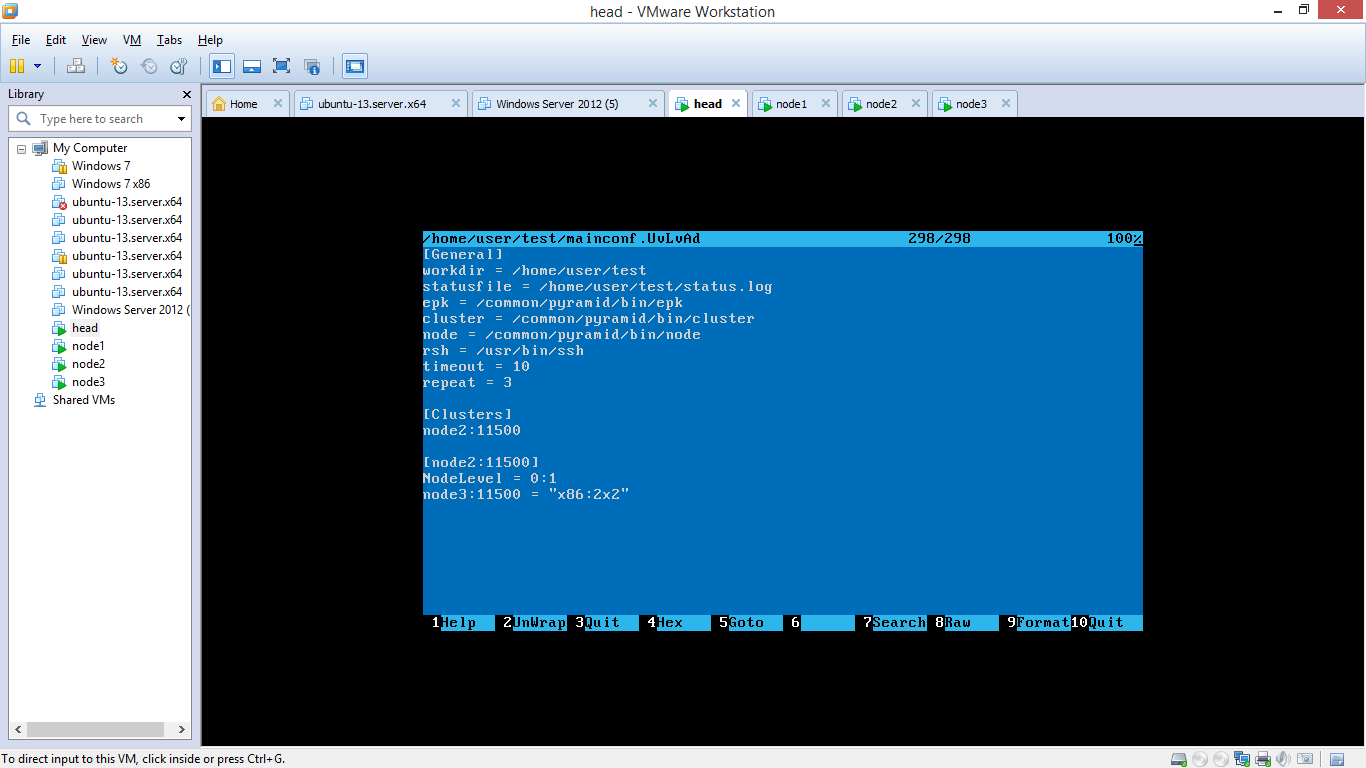


Рис. 1.11. Конфигурационный файл запуска ПК «Пирамида»

В секции Clusters показано, что Пирамида в процессе счета будет иметь один кластер, с менеджером node2, и один вычислительный модуль в кластере node3. Рассмотренный конфигурационный файл существует до тех пор, пока не завершит работу менеджер центрального сервера epk. После того как epk завершает работу, файл удаляется.

В результате работы ПК «Пирамида», в каталог из которого происходил запуск, записываются файлы-журналы работы программного комплекса, файлы с результатами работы, а также отчеты о выявленных ошибках.

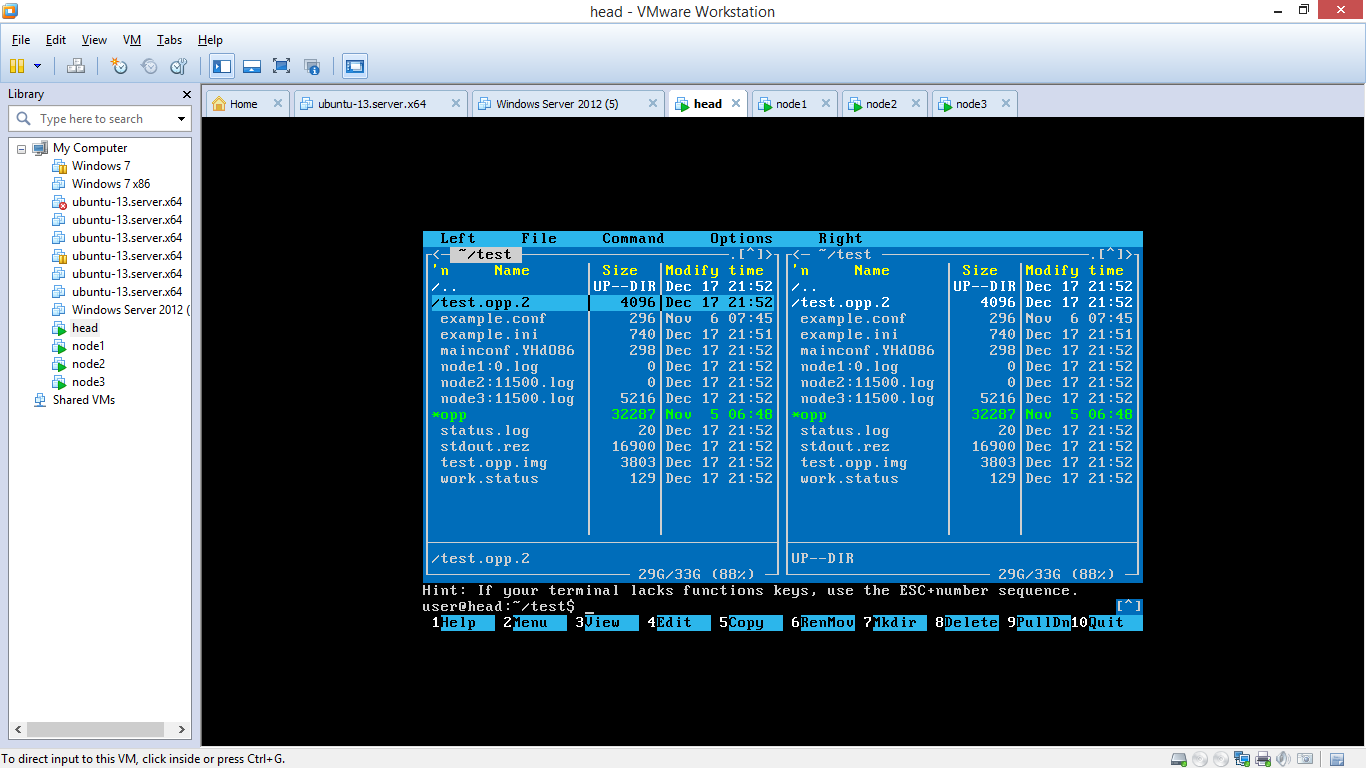


Рис. 1.12. Структура каталога после завершения работы ПК «Пирамида»

В составе СУППЗ, ПК «Пирамида» отображает ход своей работы файл errors, который находится в каталоге имя\_задачи.номер\_задачи, в данном случае в каталоге test.opp.2.

Результаты счета записываются в итоговом файле результатов stdout.rez, при этом отображается каждый вызов одной последовательной программы. Более подробно формат отображения результатов рассмотрен в документе «Программный комплекс «Пирамида». Руководство пользователя».

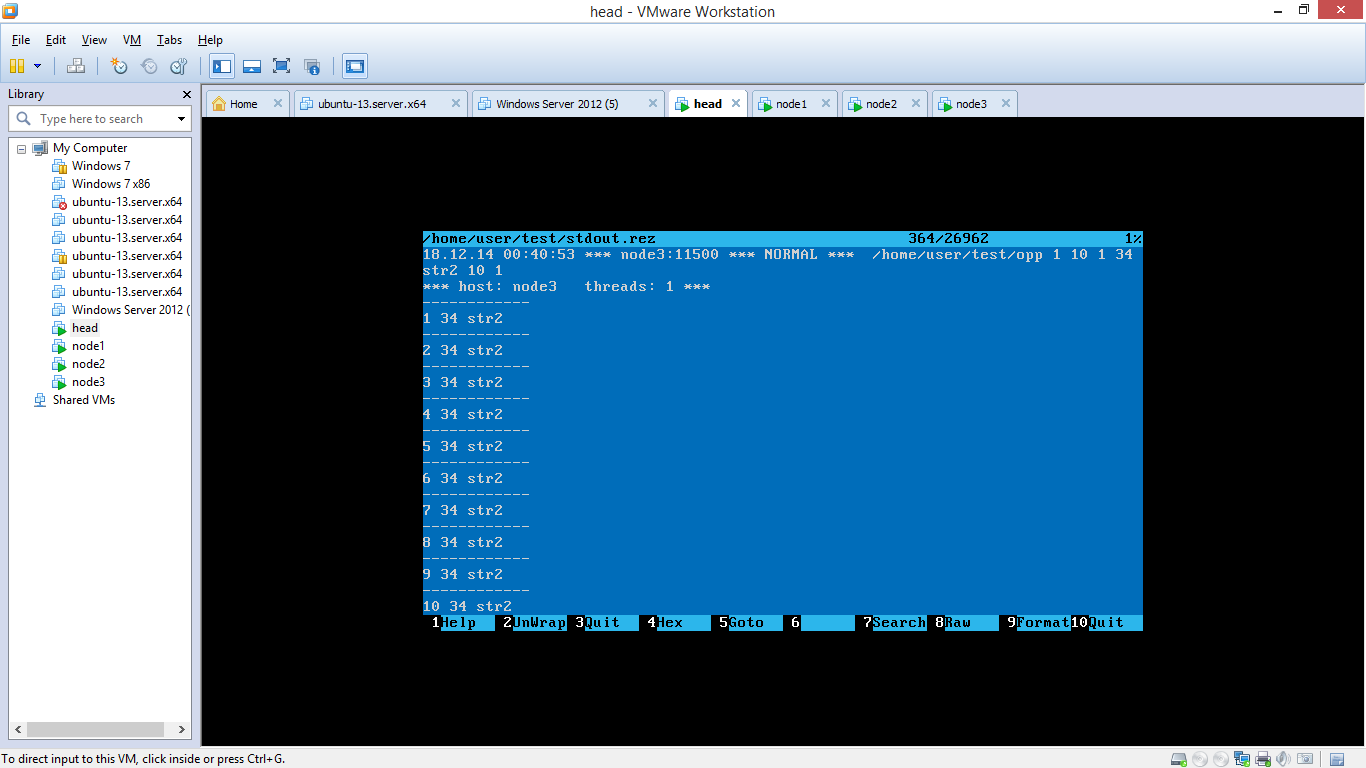


Рис. 1.13. Результаты работы ПК «Пирамида» в составе СУППЗ

## Выводы по главе 1

В первой главе автором работы cформированы требования к облачному сервису на базе программных комплексов «Пирамида» и СУППЗ. В соответствии с этими требованиями проведен поиск открытых решений облачных сервисов. Среди найденных решений выделено OpenStack. Автором принято решение использовать в разработке основные идеи и некоторые функциональные модули готового решения OpenStack.

Разработка облачного сервиса будет вестись в рамках модели IaaS. На основе разработанного IaaS сервиса будет организован доступ к ПК «Пирамида», как к программному обеспечению (SaaS).