Добрый день! Вашему вниманию представляется работа на тему «проектирование и разработка прототипа интегрированного решения по консолидации ресурсов СХД серии EMC VNX на основе существующих компонентов»

Для лучшего понимания содержания работы несколько определений из области систем хранения.

Storage Pool – единое хранилище физических дисков, которые автоматические конфигурируются в однотипные RAID-массивы. Примеры приведены на слайде.

LUN – выделенное на физическом пуле пространство блокового доступа фиксированного размера

Virtual Pool – уровень абстракции над физическими пулами с консолидацией по различным параметрам, например, как приведено на слайде, по уровням репликации, резервного копирования или протоколам доступа. Также позволяет выделять LUN, абстрагируясь от физических пулов

Целью данной работы является частичная реализация концепции выделения ресурсов блокового доступа для рассматриваемых систем на основе требований к динамическим параметрам производительности (времени отклика), который активно используется в современных СХД более высокого уровня

* Автоматизируемым процессом является создание виртуального пула на основе нескольких физических пулов, которые удовлетворяют требованиям приложения к времени отклика
* На данный момент единое решение для автоматизации этого процесса для СХД с минимальной конфигурацией отсутствует
* Ручное выполнение процесса очень времязатратно, особенно в больших датацентрах

На слайде представлена последовательность действий, выполняемых администратором вручную, а именно:

* Получение информации о СХД, физических пулах из продукта ViPR SRM с подходящими типами дисков и уровнями RAID
* Ввод полученной информации вместе с требуемыми характеристиками приложения, в утилиту Sizer. После вычислений она выдаст характеристики пулов под заданной нагрузкой
* Далее, ручной отбор пулов с подходящим временем отклика и создание виртуального пула через ViPR Controller

Немного об интегрируемых продуктах:

* Первый продукт – ViPR Controller, имеет Open-source версию – CoprHD, представляет собой платформу для централизованного управления огромным количеством разнообразных СХД
* Он привносит уровень абстракции – виртуальный пул
* Помимо пользовательского интерфейса имеет REST API для выполнения различных операций, в том числе получения статической информации о СХД и физических пулах, а также создание виртуальных пулов
* К сожалению, не поддерживает консолидацию по динамическим параметрам производительности
* Второй продукт – ViPR SRM, платформа для мониторинга и анализа использования ресурсов СХД в реальном времени. Позволяет получать более детальную информацию, необходимую для автоматизации рассматриваемого процесса. Помимо пользовательского интерфейса имеет SOAP API
* Третий продукт – утилита VNX Sizer, которая позволяет рассчитать характеристики СХД и входящих в нее физических пулов под заданной нагрузкой приложений
* Не имеет сетевого интерфейса, только пользовательский
* Можно использовать как консольную утилиту – принимает и выдает данные в формате JSON
* Все три продукта вместе предоставляют достаточный объем функциональности и информации об СХД для решения поставленной задачи
* Компонент-интегратор представляет собой самостоятельное веб-приложение с пользовательским интерфейсом, который будет взаимодействовать с остальными компонентами по сетевому протоколу.
* Для VNX Sizer будет необходимо реализовать оболочку для сетевого взаимодействия
* Архитектура спроектирована с учетом необходимости последующей интеграции в продукт ViPR Controller

На слайде представлена последовательность действий при автоматизированном выполнении рассматриваемого процесса.

- Как можно заметить, пользователю больше не нужно осуществлять какие-либо действия по подбору вручную, требуется только ввод исходных данных, и выбор физических пулов из предложенных к консолидации.

- Для проектируемых компонентов можно выделить следующие высокоуровневые варианты использования.

- Более детальное их описание с развернутыми диаграммами деятельности приведено в п.2.1.

На слайде изображена системная архитектура, разрабатываемые компоненты выделены зеленым цветом

* Выбор виртуальной инфраструктуры обусловлен ограничениями интегрируемых продуктов, также для быстрого развертывания и резервного копирования как отдельных компонент, так и всей системы в целом.
* Для реализуемых компонентов реализованы собственные сетевые протоколы взаимодействия
* Выбор способа взаимодействие с остальными компонентами обусловлен наличием у них соответствующих протоколов

Немного о программной архитектуре компонента-оболочки для утилиты Sizer.

Использование фреймворка Spring и организация взаимодействия по сети подразумевает наличие контроллера (выделен зеленым) и сервисов (выделены желтым).

* Контроллер занимается валидацией, обработкой запросов и делегированием задач сервису.
* На слайде также изображена часть классов, отображающих структуру входных и выходных JSON файлов, с которыми работает утилита Sizer.

На данном слайде можно увидеть диаграмму классов главного компонента системы. Если рассматривать ее более детально, то

* Так как основной компонент системы также основан на Spring, в нем тоже присутствует контроллер и сервисы.
* В отличие от ранее рассмотренного, этот компонент поддерживает HTTP сессию для хранения информации, полученной от интегрируемых продуктов.
* Все запросы делегируются главному сервису, который с помощью других узкоспециализированных сервисов решает поставленные задачи.
* Вся архитектура построена на принципе слабой связности компонентов, что облегчает тестирование и замену компонентов в реальном времени. Для интегрируемых компонентов реализованы компоненты-заглушки, которые позволят продемонстрировать работу прототипа даже при выходе интегрируемых продуктов из строя.

Пара слов о хранимых данных.

* Добавление параметра консолидации лишь немного расширяет информационную структуру Virtual Pool, поэтому то достаточно хранить лишь его идентификатор в ViPR Controller, заданное время отклика и требуемые характеристики приложения.

На данном слайде представлено описание сущностей, которыми обмениваются компоненты через интерфейс для утилиты Sizer.

* Он имеет лишь один метод взаимодействия, принимающий информацию о СХД, конфигурации пула и нагрузке, и отдает характеристики этого пула под заданной нагрузкой.
* Здесь можно увидеть более глубокое описание структуры запроса. Часть вводимой информации носит перечислимый характер, поэтому оформлена в виде перечислений для упрощения валидации.
* Выходными параметрами являются утилизация и время отклика пула под заданной нагрузкой
* Главный компонент системы предоставляет интерфейс с 6 методами
* непосредственно открытие веб-страницы
* получение списка SP с краткой информацией об их производительности
* получение детальной информации о конкретном SP
* метод для расчета характеристик пулов под заданной нагрузкой
* Он имеет аналогичную ранее описанному интерфейсу структуру запроса
* Метод для создания VP
* и получения списка созданных VP. Он не является необходимым для рассматриваемого варианта использования, но наглядно иллюстрирует то, как будут реализовываться смежные варианты использования, например, отображение созданных пулов

В завершении пара слов об особенностях реализации и тестирования.

* В качестве языка программирования и фреймворка для разработки серверной части выбраны Java и Spring. Данный выбор обоснован необходимостью дальнейшей интеграции в продукт ViPR, который разработан на основе этих технологий.
* Дополнительно используется модуль Spring Boot, который позволяет запускать и разворачивать приложение без использования стороннего сервера приложений, что упрощает процесс разработки и развертывания. В качестве базы данных используется MongoDB. Это обосновано структурой хранимых данных, а также тем, что она является основной в продукте ViPR Controller. Для разработки веб интерфейса использовано jQuery.
* Для разработанных компонентов были составлены наборы тестов, покрывающие основную бизнес-логику компонентов.
* В результате выполнения ВКР были разработаны приведенные на слайде компоненты и интерфейсы, которые в дальнейшем планируется интегрировать в продукт ViPR Controller.