Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Факультет прикладной математики и механики

Кафедра «Вычислительная математика, механика и биомеханика»

Направление подготовки: 09.04.02 «Информационные технологии и системная инженерия»

Отчет

по дисциплине: «Программирование с использованием облачных платформ Google-Apps Amazon-Azure»

Выполнил:

Студент группы ИТСИ-18-1м

Давыдова Ирина Николаевна

Проверил:

Ассистент кафедры ВММБ

Истомин Денис Андреевич

Пермь 2019 г

Оглавление

[Лабораторные работы по дисциплине 3](#_Toc26648330)

[Лабораторная работа №1 3](#_Toc26648331)

[Лабораторная работа №2 6](#_Toc26648332)

[Лабораторная работа №3 7](#_Toc26648333)

[Доклад по дисциплине 11](#_Toc26648334)

[Azure Stream Analytics – это… 11](#_Toc26648335)

[Как работает Stream Analytics? 12](#_Toc26648336)

[Основные возможности и преимущества 13](#_Toc26648337)

[Простое начало работы 13](#_Toc26648338)

[Создание Stream Analytics задания для работы с панелями мониторинга в режиме реального времени 13](#_Toc26648339)

[Внедрение в реальном времени аналитических сведений о приложениях с помощью сообщений о событиях 14](#_Toc26648340)

[Динамические приложения и веб-сайты 14](#_Toc26648341)

[Внедрение аналитических сведений в приложение в режиме реального времени с помощью хранилищ данных 15](#_Toc26648342)

[Вывод 16](#_Toc26648343)

[Список литературы 17](#_Toc26648344)

# Лабораторные работы по дисциплине

## Лабораторная работа №1

**Цель:** Получить теоретические знания и практические навыки работы с docker и docker- compose.

**Задача:** Создать простое веб-приложение, запустить в docker, создать yaml файл для docker-compose.

**Теория:**

Docker это отличный инструмент, но чтобы воспользоваться его преимуществами по полной, каждый компонент вашего приложения должен работать в своем собственном контейнере. Если у вас сложное приложение со множеством компонентов, то вам будет сложно делать так, чтобы все контейнеры запускались и закрывались вместе (не говоря уже о сообщении между ними).

Сообщество Docker придумало популярный инструмент под названием Fig, который позволил использовать один файл YAML для управления всеми контейнерами и конфигурациями Docker. Инструмент стал насколько популярен, что команда Docker наконец решила создать свою собственную версию, основанную на Fig. Они назвали свою новую разработку Docker Compose. Если вкратце, то он упрощает процессы управления Docker-контейнерами (например, запуск, закрытие и настройка связей и объемов внутри контейнеров).

Использование Docker Compose требует объединения нескольких разных элементов Docker в одно. Поэтому перед тем, как начать, давайте рассмотрим основные элементы.

Каждый Docker-контейнер это локальный экземпляр Docker-образа. Docker-образ можно рассматривать как полную установку Linux. Обычно минимальная установка включает установку минимального набора пакетов, необходимых для управления образом. Эти образы используют ядро операционной системы, но поскольку они функционируют внутри Docker-контейнера и видят только свою собственную файловую систему, то вполне можно выполнить распределение, например CentOS на системе Ubuntu (или наоборот).

Большинство Docker-образов распространяются через Docker Hub, поддерживаемый командой Docker. Наиболее популярные общедоступные проекты имеют соответствующий образ, загруженный в Docker Registry, который вы можете использовать для развертывания программного обеспечения. При возможности, лучше брать «официальные» образы, поскольку команда Docker гарантирует, что они соответствуют передовым стандартам Docker.

Docker-контейнеры по умолчанию изолированы от хост-компьютера. Это значит, что по умолчанию хост-компьютер не имеет доступа к файловой системе внутри Docker-контейнера и не имеет никаких средств связи с ней через сеть. Понятно, что это усложняет конфигурацию образа и работу с образом внутри Docker-контейнера.

Docker предлагает три основных способа решения этой проблемы. Первый и наиболее распространенный – сделать так, чтобы Docker указал переменные среды, которые будут заданы внутри Docker-контейнера. Код, выполняемый внутри Docker-контейнера, затем при запуске проверит значения этих переменных среды и использует их для собственной конфигурации.

Еще один распространенный способ – объем данных Docker. Docker-хранилище данных бывают внутренними и общими. Docker-контейнеры по умолчанию работают таким образом, что каждый раз при закрытии или запуске контейнера он не сохраняет свои данные. Он возвращается в состояние, в котором он находился в момент запуска. Это хорошо для тестирования и разработки, потому что вы гарантированно будете работать с одной и той же средой. Но это плохо в той ситуации, если вы надеетесь, что посты вашего блога, которые вы напечатали в своем WordPress, сохранятся до следующей загрузки Docker.

Если вы задаете внутреннее хранилище, то в папке, которую вы указали для определенного контейнера Docker, данные между запусками будут сохраняться. Например, если вы хотите убедиться, что файлы вашего журнала сохраняются между запусками, вы можете задать внутреннее хранилище /var/log.

Общее хранилище отображает папку внутри Docker-контейнера на папке на хост-компьютере. Это позволяет легко передавать файлы между Docker-контейнером и хост-компьютером.

Третий способ связи с Docker-контейнером - через сеть. В Docker сообщение между разными Docker –контейнерами осуществляется через links (связи). Также через links осуществляется переадресация портов, что позволяет вам переадресовывать порты из Docker-контейнера в порты на хост-сервере. Например, вы можете создать связь, которая позволит вашим Docker-контейнерам WordPress и MariaDB переговариваться друг с другом, а переадресации портов – предоставить WordPress для внешнего пользования, чтобы пользователи могли подключиться к нему.

## Лабораторная работа №2

**Цель:** Получить теоретические знания и практические навыки работы с S3 и облачными хранилищами.

**Задача:** Создать веб-страницу на java, с которой можно читать и загружать объекты по протоколу S3

**Теория:**

Услуга S3 объектного хранилища, предлагаемая облачными провайдерами, становится популярной среди различных компаний. Представляя собой высоконадежное решение, S3 storage может хранить миллиарды-триллионы файлов, масштабируясь до сотен петабайт. За счет своих характеристик в виде совместимости, безопасности, надежности, отказоустойчивости, услуга продолжает быть востребованной и популярной не только в бизнесе, но и в повседневном жизни.

S3 хранилище – это сервис хранения объектов, предлагаемый поставщиками облачных услуг. Основное преимущество решения – возможность хранить файлы любого типа, любого объема, с высоким уровнем надежности и доступности.

Принцип работы с S3 хранилищем сводится к созданию контейнеров и добавлению туда необходимых файлов, которые представляются в виде объектов. Таким образом все, что попадает в контейнер, можно просматривать, перемещать или удалять. Сами контейнеры, в случае необходимости, тоже могут быть удалены.

Важный момент: в object-хранилищах помимо самих объектов хранятся метаданные, определяющие свойства объекта и глобальный уникальный идентификатор в виде присвоенного адреса. Эти атрибуты хранятся в плоском адресном пространстве, позволяя избавиться от проблем, которые встречаются при работе с иерархической файловой системой на основе сложных путей к файлам. Примечательно, что один объект может содержать разнородные метаданные, которые наиболее подробным образом его характеризуют. Например, это может быть аудиофайл с заданными метаданными в виде исполнителя, названии песни,альбома и прочей информации. В дальнейшем метаданные файлов индексируются, что в значительной степени облегчает и в разы ускоряет поиск нужных объектов по заданным признакам.

## Лабораторная работа №3

**Цель:** Получить теоретические знания и практические навыки работы с облачными когнитивными сервисами, а именно с компьютерным зрением.

**Задача:** Создать веб-страницу на java, с которой можно загружать изображения и использовать функции компьютерного зрения (например, распознавания).

**Теория:**

Компьютерное зрение — это такая дисциплина (раздел искусственного интеллекта), которая занимается извлечением информации из изображений, причем изображения могут быть разного типа. Это могут быть фотографии, могут быть видео, могут быть наборы фотографий или медицинский снимок из магнитно-резонансного томографа. В последнее время стали очень популярны алгоритмы, которые работают с изображениями и сочетают цветовую информацию о точках и их пространственное положение, потому что сенсоры, получающие такую информацию, стали гораздо более доступными, а понимать мир с помощью таких сенсоров роботам или компьютерам стало гораздо проще.

Информация, которую разные алгоритмы извлекают из изображений в компьютерном зрении, тоже может иметь разную природу, разный тип. Некоторые алгоритмы просто разбивают изображение на части, которые соответствуют отдельным объектам или разным частям объектов. Например, врач может взять медицинский снимок или часть медицинского снимка, разбить его на части и взять ту, которая соответствует интересующему его объекту, например органу или опухоли, а дальше померить объем или измерить диаметр этого объекта. В таком случае не только экономится время врача, но и уменьшается вариативность, связанная с его личностью, что особенно полезно в случаях, когда измерения для одного и того же пациента проводятся несколькими врачами или когда измерения отделяют несколько месяцев.

Еще один важный раздел компьютерного зрения — это построение трехмерных моделей из изображений, например из набора фотографий или из видео, причем эти фотографии могут быть сняты одним фотографом или скачаны из интернета. Скажем, тысячи фотографий, которые сняты разными фотографами и на которых изображена Красная площадь. Или эти данные могут быть получены роботом, путешествующим по какому-то незнакомому помещению или в какой-то незнакомой местности и пытающимся во время путешествия построить трехмерную модель. И чтобы построить такую трехмерную модель, компьютеру нужно сопоставить части сцены, то есть понять, что на разных изображениях разные фрагменты соответствуют одним и тем же объектам в трехмерном мире.

Это на самом деле сделать очень тяжело. Например, если у вас есть здание, то левый нижний уголок окна будет выглядеть одинаково для разных окон этого здания, и если просто посмотреть на те несколько пикселей, которые окружают этот уголок окна, то ни человек, ни компьютер не поймут, разные это окна или одни и те же. Поэтому, чтобы построить такую модель, чтобы сопоставить такие части изображений, компьютеру надо постепенно строить модель на основе более уникальных фрагментов сцены, а потом, сопоставляя более уникальные фрагменты, распространять это знание на менее уникальные. Компьютер для этого использует алгоритмы оптимизации, а что для этого используют люди или животные, когда строят в мозгу трехмерные карты, мы до конца не понимаем.

В результате такого моделирования компьютер получает фактически трехмерный слепок сцены, и, что еще очень важно, он понимает положение камеры в момент, когда происходила фото- или видеосъемка. И дальше эта информация может быть использована разными способами. Например, можно измерять размеры в сцене — это так называемая фотограмметрия. Или полученную трехмерную модель и положение сцены можно использовать в приложении компьютерной графики. Например, такие системы используются во всех современных фильмах, которые используют спецэффекты, сочетающие реальные и виртуальные объекты. Кроме того, трехмерная модель будет в ближайшее время все более активно использоваться в компьютерных играх, приложениях реальной и дополненной реальности.

Однако главный прогресс происходит в областях компьютерного зрения, связанных с пониманием изображений, то, что называется высокоуровневое компьютерное зрение. Оно занимается задачами, которые очень легко формулируются. Например, вот фото, выпиши список объектов, которые на этом фото, их класс (человек, автомобиль, собака, кусок лужайки) и то, где они на изображении находятся. Или вот видео, определи, что делают люди на этом видео: идут они спокойно по делам или начинается что-то необычное, например драка, и нужно срочно привлечь внимание оператора к этому видео. Потому что недостаточно просто завесить весь город камерами — нужно, чтобы видеопотоки с этих камер как-то обрабатывались.

В середине 2000-х алгоритмы высокоуровневого зрения работали плохо. Компьютер не мог отличить фотографию кошки от фотографии собаки, то есть самые лучшие алгоритмы работали почти как подбрасывание монетки. Десять лет спустя студент, прошедший курс компьютерного зрения и глубинного обучения, сможет за несколько дней создать систему, натренировать ее, и эта система будет отличать снимок кошки от снимка собаки с точностью до 99%. Это происходит за счет того, что теперь у нас есть программы, называемые сверточными нейронными сетями, которым можно подать на вход изображение, и эта программа с очень хорошей точностью скажет, изображение это кошки или изображение собаки и даже, если нужно, какая порода кошки или собаки на этом изображении.

Эти программы работают так хорошо во многом за счет того, что у них есть огромное количество настраиваемых параметров — миллионы, десятки миллионов. Настраивает их не программист вручную (это было бы невозможно), а сам компьютер в процессе машинного обучения, глядя на большое количество обучающих фотографий. И когда такая сверточная нейронная сеть обучена, причем необязательно для распознавания фотографий кошки и собаки, она может быть научена и для распознавания других классов. После этого мы можем взять эту предобученную нейронную сеть и доучить ее для задач, которые нас интересуют. Более того, обученная нейросеть будет отображать изображения, содержащие похожие объекты или один и тот же объект, в многомерные вектора, которые находятся рядом в многомерном пространстве. Это очень важное и удобное свойство для всех систем обработки информации. И, что интересно, это будет происходить, даже если объекты на этих изображениях сеть во время обучения не видела. Как говорят специалисты по машинному обучению, происходит обобщение, перенос знаний на новые классы.

# Доклад по дисциплине

## Azure Stream Analytics – это…

Azure Stream Analytics — это служба аналитики, работающая в режиме реального времени, которая представляет собой сложный механизм обработки событий. Она позволяет анализировать и обрабатывать большие объемы быстро передающихся потоковых данных из многочисленных источников одновременно. Она обеспечивает определение закономерностей и связей в данных, извлеченных из нескольких источников, таких как устройства, датчики, данные посещений сайта, каналы социальных сетей и приложения. На основе этих закономерностей можно активировать действия и инициировать рабочие процессы, например создавать оповещения, отправлять данные в средства составления отчетов или сохранять преобразованные данные для дальнейшего использования. Кроме того, Stream Analytics доступна в среде выполнения Azure IoT Edge и поддерживает такие же язык или синтаксис, что и в облаке.

Ниже описаны примеры сценариев, в которых удобно использовать Azure Stream Analytics:

* анализ потоковой передачи данных телеметрии в реальном времени с устройств Интернета вещей;
* анализ веб-журналов и сведений о посещении сайта;
* геопространственная аналитика для управления транспортной системой и автономными транспортными средствами;
* удаленный мониторинг и прогнозное обслуживание высокоценных ресурсов;
* аналитика данных точки продаж в режиме реального времени для контроля запасов и обнаружения аномалий.

## Как работает Stream Analytics?

Задание Azure Stream Analytics состоит из входных данных, запроса и выходных данных. Stream Analytics принимает данные из Центров событий Azure, центра Интернета вещей Azure или хранилища BLOB-объектов Azure. Запрос на языке SQL-запросов можно использовать для фильтрации, сортировки, агрегирования и объединения данных потоковой передачи за определенный промежуток времени. Этот язык SQL-запросов можно расширить с помощью определяемых пользователем функций JavaScript и C#. Вы можете легко настроить параметры упорядочивания и продолжительность временных окон при осуществлении операций агрегирования с помощью простых языковых конструкций и (или) конфигураций.

На данном слайде показано, как данные отправляются в Stream Analytics, анализируются и направляются для выполнения других действий, например хранения или отображения.

Каждое задание поддерживает вывод преобразованных данных. Вы можете также настроить действия по результатам анализа информации. Например, вы можете просматривать:

* отправлять данные в службы, такие как Функции Azure, разделы служебной шины или Очередь, чтобы активировать связи или нисходящие пользовательские рабочие процессы;
* отправлять данные на информационную панель Power BI для мониторинга в режиме реального времени;
* сохранять данные в других службах хранилища Azure, чтобы обучать модель машинного обучения на основе данных журнала или выполнять пакетную аналитику.

## Основные возможности и преимущества

Служба Azure Stream Analytics должна быть простой в использовании, гибкой, надежной и масштабируемой до любого размера задания. Она доступна в нескольких регионах Azure. На следующем рисунке показаны ключевые возможности Azure Stream Analytics.

## Простое начало работы

Приступить к работе с Azure Stream Analytics несложно. Подключиться к нескольким источникам и приемникам и создать конвейер можно в несколько щелчков. Служба Stream Analytics может подключаться к Центрам событий Azure и Центру Интернета вещей Azure для приема потоковых данных, а также к хранилищу BLOB-объектов Azure для приема исторических данных. Входные данные для задания также могут содержать статические или редко меняющиеся эталонные данные из хранилища BLOB-объектов Azure или Базы данных SQL, которые можно подключить к потоковым данным для операций поиска.

Stream Analytics может направлять выходные данные задания в несколько систем хранения, таких как хранилище BLOB-объектов Azure, База данных SQL Azure, Azure Data Lake Store и Azure Cosmos DB. Вы можете выполнить пакетную аналитику с использованием сохраненных данных с помощью Azure HDInsight или направить выходные данные в другую службу, например в Центры событий для обработки или в Power BI для визуализации в режиме реального времени.

## Создание Stream Analytics задания для работы с панелями мониторинга в режиме реального времени

С помощью Azure Stream Analytics можно быстро разделять панели мониторинга и оповещения в режиме реального времени. Простое решение принимает события из концентраторов событий или центра Интернета вещей и передает Power BI панель мониторинга с набором данных потоковой передачи.

Это решение можно создавать всего за несколько минут от портал Azure. Не существует обширного кода, и для выражения бизнес-логики используется язык SQL.

Этот шаблон решения обеспечивает наименьшую задержку от источника событий к панели мониторинга Power BI в браузере. Azure Stream Analytics является единственной службой Azure с этой встроенной возможностью.

## Внедрение в реальном времени аналитических сведений о приложениях с помощью сообщений о событиях

Вторым наиболее популярным использованием Stream Analytics является создание оповещений в режиме реального времени. В этом шаблоне решения бизнес-логика в Stream Analytics можно использовать для обнаружения временных и пространственных шаблонов или аномалий, а затем создавать сигналы предупреждения. Однако в отличие от решения для панели мониторинга, где Stream Analytics использует Power BI в качестве предпочтительной конечной точки, можно использовать ряд промежуточных приемников данных. Эти приемники включают концентраторы событий, служебную шину и функции Azure. Вы, как построитель приложений, должны решить, какой приемник данных лучше подходит для вашего сценария.

## Динамические приложения и веб-сайты

Вы можете создавать пользовательские визуализации в режиме реального времени, например панель мониторинга или визуализацию карт, с помощью Azure Stream Analytics и службы Azure SignalR. С помощью SignalR можно обновлять веб-клиенты и отображать динамическое содержимое в режиме реального времени.

## Внедрение аналитических сведений в приложение в режиме реального времени с помощью хранилищ данных

Сегодня большинство веб-служб и веб-приложений используют шаблон "запрос — ответ" для обслуживания уровня представления. Шаблон "запрос — ответ" прост в создании и может легко масштабироваться с малым временем отклика с помощью фонового и масштабируемого хранилищ без отслеживания состояния, например Cosmos DB.

Большой объем данных часто создает узкие места производительности в системе на базе CRUD. Шаблон решения "источники событий" используется для устранения узких мест производительности. Временные шаблоны и аналитические сведения также сложно и неэффективно извлекать из традиционного хранилища данных. Современные приложения, управляемые большими объемами данных, часто используют основанную на потоках архитектуру. Azure Stream Analytics как движок вычислений для данных в движении - это стержень в этой архитектуре. В этом шаблоне решения события обрабатываются и объединяются в хранилища данных с помощью Azure Stream Analytics. Прикладной уровень взаимодействует с хранилищами данных, используя традиционный шаблон запроса / ответа. Благодаря способности Stream Analytics обрабатывать большое количество событий в режиме реального времени, приложение легко масштабируется без необходимости наращивать уровень хранилища данных.

# Вывод

После прохождения курса «Программирование с использованием облачных платформ Google-Apps Amazon-Azure» были получены теоретические знания и практические навыки по работе контейнерами docker, с протоколом S3 для работы с файлами в облачных хранилищах и когнитивными сервисами в облачных сервисах на примере компьютерного зрения.

# Список литературы

1. Amazon Simple Storage Service Documentation. – Режим доступа: – <https://docs.aws.amazon.com/s3/index.html> - (Дата обращения: 12.12.2019).
2. Парминдер Сингх Кочер. Микросервисы и контейнеры Docker. М.: ДМК Пресс, 2019. — 240с.
3. Паттерсон Джош, Гибсон Адам. Глубокое обучение с точки зрения практика. М.: ДМК Пресс, 2018. — 418.