Phillip Yarbrough
9/9/2021
A590

TOPICS IN PROGRAMMING

The purpose of this research paper is to discuss various imperative topics throughout programming that are essential towards current tools, technologies and framework in modern Web-based data management and software development. Topics that will be covered include data-centric architecture, Daap (Data as a platform), Saap (Service as a product), API architecture, MVC design pattern in Django, the five most popular databases, five databases suitable for real-time query and processing of large-scaled data, when a user should choose Apache Spark as a data processing engine, and Docker and Kubernetes. The first part of this research paper will be discussing data-centric architecture, Daap, and Saap.

According to the academic article *DBOS: A Proposal for a Data-Centric Operating System* by Michael Cafarella, a huge problem in current operating systems is that they are hard to scale and secure due to the absence of a centralized data model of operating system state. [1] An example of this is the Linux Kernel, which contains numerous data structures that manage different parts of the operating system. Some of these different parts include the scheduler, page cache, namespaces, and many permissions. Each of these kernels' present interfaces for management, such as perf, netstat, etc. To improve the scalability, operations, and security of operating systems, a data-centric architecture is imperative. According to Vista Projects, "A data-centric outlook is a core concept in data-centric architecture, where data is viewed as a critical and perpetual asset used in support of applications to produce deliverables. Within data-centric architecture, the data model precedes the implementation of a given application and remains valid long after the application is gone." [2] Focusing on a data-centric mindset, data must steer the advances of projects, business judgements, and culture. Cloud computing has been a tremendous advancement for data-centric architecture due to being able to remote access and analyze large databases to make further decisions that could benefit all parties involved. Data-centric execution has two major advantages over prior documentation-driven methods. The first advantage of data-centric execution is that it creates a single source of truth (SSOT). This creates a single data model that will be utilized by all project employees and information systems. This will permanently eliminate several instances of data. The second advantage of data-centric execution is that it is current and up to date data. The next section will be discussing Daap (Data as a platform) and its significance towards business.

According to Kaiti Norton, "Data-as-a-Platform (DaaP) is a business model in which a company collects, stores, and manages a large volume of data then offers it to external partners and third-party systems." [3] This is something that is currently on-going throughout the world. We are seeing thousands of businesses buy, sell, and trade volumes of data for profit driven purposes, performing better business decisions, and finding out new ways to get consumers to buy more of their product. A good example of this is a big company collecting data about their audience's purchasing habits over an extended period. After all this data is collected, the company decides to sell all this data to a third-party company so that they can make better business decisions based on consumer habits. There is a purpose because advertisements are focusing now on consumers likes, dislikes, interests, and hobbies whenever an ad comes onto your device. These are targeted for your profile specifically. There are huge financial gains from this strategy for businesses throughout the world. Daap is substantial due to the philosophy being concentrated on collecting as much raw data as possible. The only way Daap is possible is for a company must collect, maintain, and gain large volumes of data. If a company has all this raw data, many other businesses will pay to have access to this data to drive decision making without having to collect, store, manage, and maintain the databases. This can save a company huge amount of money without needing to hire an internal data scientist and they can utilize someone else's data simply by remoting in. The possibilities truly are endless. The next section will discuss Saap (service as a product) and its significance towards IT and business.

According to David Moran on SaaP, "Software as a product requires a one-time license purchase at a higher fee. This allows users to download a product onto a server and host it themselves. In this sense, there is a physical element to SaaP as it is a product purchase and not a service rental. Once installed it is up to the business or user to ensure product maintenance. Upgrades of a product usually come at an additional cost and your IT team will be involved in ensuring things run smoothly. Internal servers are equipped to accommodate SaaP, and offline functionality is sometimes possible," [4] SaaP is an extremely popular software purchase for many companies throughout the world. I can guarantee everyone can think of a specific software they have purchased using the SaaP principle. Three of the most admired software as product areas are CRM (customer relationship management), a virus protector, and an ERP system (enterprise resource planning system). There are many advantages and disadvantages for a company to purchase SaaP. Some of the advantages is that most of the software has numerous customizable features, it brings enhanced data control, and it is single-user friendly. Some of the disadvantages is that this software can take up a lot of space for storage, the owner must pay for upgrades, and the owner must consider what to do with the old model of the software. SaaP provides internal data control and improves online security as well. When it comes to SaaP, the breakthrough is in ownership. The next section will discuss API architecture and its significance towards IT and business.

According to Shyam Purkayastha, "API architecture refers to a software building block that implements an API." [5] There are four primary obligations of an API. The first obligation is interfacing, and this is when the interface defines the specifications for accessing the API. When it comes to a Web API, the interface identifies the protocol specifications for accessing the API. The second obligation is interpretation, and this is when the data is sent as part of an API request. The third obligation is computing, and this is when the API executes business logic and its imperative to have a computing environment. The last obligation is data processing. Every single application relies on data storage that needs to transfer, dump, or fetch data. APIs rely on these tasks as well. API architecture is split into four building blocks that define its structure. The first block is the interface, and the primary purpose is to define the specifications for the API. An example of this is that the API interface defines additional HTTP headers. The second block is the controller, and its primary purpose is for the API request and response to be translated. The third block is the runtime, and its primary purpose is to take care of the computing responsibility of the API. The last building block is the data bridge, and its primary purpose is to be responsible for coordinating this data access. Everyone is aware of the multiple advantages that an API architecture serves, but many people do not realize there are also disadvantages for having this system in place. If a hacker was able to enter the single point of entry for an API, all other applications and systems become vulnerable. APIs can be accessed over the internet, and this leads to multiple disadvantages such as CSRF attacks, SQL injection, and DDos attacks. The next section will discuss MVC design pattern in Django and its significance towards IT and business.

According to the Djangoproduct.com, "Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source." [6] MVC is an acronym for Model View Controller, and it comprises of the three components model, view, and controller. The MVC design pattern in Django provides the working cycle of Django MVC architecture. The model is part of a web-app that acts as a mediator between the website interface and the database. The model primarily consists of business logic. An example of this is signing up on a website. The user is sending information to the controller, then it is transferred to the model, business logic is applied, and it stores into the database. The view component comprises of UI logic in the Django Architecture. An example of this is when a user clicks on any link, the new page is the specific views that stores and generates when we are cooperating with components. The controller is the main control component. This handles all user interaction and selects a view. There are many benefits when sing the MVC design pattern in Django such as rapid development, ease of modification, and you can save on a server rather than saving on a webpage. The next section will discuss the five most popular databases and its significance towards IT and business.

The most widely used and popular database in the world is MySQL. According to Md Kamaruzzaman, "Today, MySQL is one of the most popular and widely used SQL databases. It is also one of the most used databases in Web Applications. Some of the world's largest Web-Scale applications (e.g., Facebook, Uber) uses MySQL." [7] MySQL is still maintaining high traction and MySQL Community Edition is the most widely used free database in the industry. Its commercial version is used extensively in the industry. Some of the key features is that its open-source RDBMS with two licensing models, offers ACID transactional guarantee, offers multi-master ACID transactions, and it supports both SQL and JSON.

The second most popular database in the world is Oracle. According to Md Kamaruzzaman, "Currently, Oracle is the number one commercially supported database and one of the widely used RDBMS overall. Its latest release (21.c) has added many innovative features that will make it an attractive option in the coming years." [7] Some of the key features are proprietary RDBMS, offers ACID transactional guarantee, Advanced Multi-Model databases supporting Structured Data (SQL), Semi-Structured Data(JSON, XML), Spatial Data, and RDF Store, offers blockchain tables, and supports both OLTP and OLAP workload. Oracle has many competitors and competition from open-source SQL databases and NoSQL databases. With Oracles newest release, it has gained lots of new customers in the last couple of years.

The third most popular database in the world is PostgreSQL. According to Md Kamaruzzaman, "PostgreSQL was born as part of the POSTGRES project, which is a Relational Database Management System. Over the past 30 years, PostgreSQL leads the way in Modern database Development, contributing many innovations, and Michael Stonebraker got a Turing Award in 2014 primarily for his work in PostgreSQL." [7] PostgreSQL is one of the most used databases and is one of the most advanced open-source relational databases. Some of the key features are open-source RDMBS, it offers immediate consistency as a single server, Citrus Data, offers more advanced indexes such as partial index and bloom filters, and it's a feature-rich multi-model database supporting Structured Data (SQL), Semi-Structured Data (JSON, XML), Key-Value, Spatial Data. Due to its advanced features, community support, and improvements, PostgreSQL is trending. Its users are growing in the last five years.

The fourth most popular database in the world is Microsoft SQL Server. Microsoft SQL Server is a leading commercial database system with tremendous tooling support from Microsoft. Some of the key features are a proprietary RDBMS with diverse licenses, offers ACID transactional guarantee, supports server-side scripting via T-SQL, .NET languages, R, Python, and Java, multi-model database supporting SQL, JSON, and Spatial Data, and tremendous tooling support. MS SQL is the most popular commercial database in the Windows platform and the preferred SQL database in Azure Cloud. There has been a downward trend on MS SQL use since 2018.

The fifth most popular database in the world is MongoDB. According to Md Kamaruzzaman, "MongoDB went through major improvements. It has addressed many of its shortcomings (e.g., Security) and innovated and pioneered many features. Currently, it is the principal Document Database and the leading NoSQL Database." [7] Some of the key features are open-Core NoSQL Document Database (BSON) with various licenses, offers horizontal scaling, built-in replication, distributed multi-document ACID transactions with snapshot isolation, the query language is powerful using pipelines, and it offers a full-text search engine. MongoDB has been trending and rising in popularity dramatically in the recent years. Since 2016 there has been a dramatic rise in popularity, and it continues to soar. The next section will discuss five databases suitable for real-time query and processing of large-scaled data. Discussed will be its significance towards IT and business.

Using databases to process real-time queries for large amounts of data is an extremely powerful and useful tool for business logic, IT advancements, and makes you a huge competitor for analytic business practices. To process such a huge amount of data in real-time, this requires building a data processing pipeline. The biggest challenge building these pipelines is that there is a need to figure out strategy to minimize latency to process high-throughput data. To keep this real-time element, this requires the user to seek out the use of NoSQL database. The first database suitable for real-time query and processing large amounts of data would be Apache Hadoop. According to the Apache Hadoop website, "The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing. The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly available service on top of a cluster of computers, each of which may be prone to failures." [8] In order to understand how this can be used on a large scale with real-time querying, here are a few users of Apache Hadoop: Amazon, Adobe, Google, IBM, Spotify, Twitter, and many Universities.

The second database to process real-time queries for large amounts of data is BangDB. According to BangDBs' website, "BangDB natively integrates AI, Streaming, Graph, analytics within the DB itself to enable users to deal with complex data of different kinds, such as text, images, videos, objects etc. for real time data processing and analysis. Ingest or stream any data, process it, train models, do prediction, find patterns, take action and automate all these to enable use cases such as IOT monitoring, fraud or disruption prevention, log analysis, lead generation, 1-on-1 personalization and many more." [9] BangDB is a multi-model, embedded, distributed, high performance, analytical, time-series NoSQL database written in C/C++ and designed from scratch for solving contemporary and future problems in simple and easy manner which otherwise requires huge amount of time and resources. [9] One of the huge advantages for choosing BangDB is that it offers key-value stores, document storage, and graph storage. The user can build a plethora of SaaS applications to meet business, IT, and personal related needs.

The third database to process real-time queries for large amounts of data is Amazon DynamoDB. According to Amazons website, "Amazon DynamoDB is a key-value and document database that delivers single-digit millisecond performance at any scale. It's a fully managed, multi-region, multi-active, durable database with built-in security, backup and restore, and in-memory caching for internet-scale applications. DynamoDB can handle more than 10 trillion requests per day and can support peaks of more than 20 million requests per second. Many of the world's fastest growing businesses such as Lyft, Airbnb, and Redfin as well as enterprises such as Samsung, Toyota, and Capital One depend on the scale and performance of DynamoDB to support their mission-critical workloads. Hundreds of thousands of AWS customers have chosen DynamoDB as their key-value and document database for mobile, web, gaming, ad tech, IoT, and other applications that need low-latency data access at any scale. Create a new table for your application and let DynamoDB handle the rest." [10] The three major benefits for choosing this database are its performance at scale, no servers to manage, and it is enterprise ready. For performance at scale, DynamoDB supports some of the world's largest scale applications by providing consistent, single-digit millisecond response times. DynamoDB is serverless with no servers to provision, patch, or manage and no software to install, maintain, or operate. DynamoDB supports ACID transactions to enable you to build business-critical applications at scale. These are huge benefits for a database and Amazons cliental are huge corporations that use this database to its fullest potential. Some of Amazon DynamoDB customers are Disney, Samsung, The Pokémon Company, Nike, Netflix, and A+E Networks.

The fourth database to process real-time queries for large amounts of data is Apache Cassandra. According to the Apache Cassandra website, "Apache Cassandra is an open-source NoSQL distributed database trusted by thousands of companies for scalability and high availability without compromising performance. Linear scalability and proven fault-tolerance on commodity hardware or cloud infrastructure make it the perfect platform for mission-critical data." [11] One major advantage for Apache Cassandra is that its databases are distributed that yields both technical and business advantages. "Distributed" means that Cassandra can run on multiple machines while appearing to users as a unified whole. Cassandra can run on multiple nodes and these nodes can communicate with one another through a protocol galled gossip. Another advantage of Cassandra is that is has multiple datacenters. Cassandra has been gaining popularity in the last couple years and trending. Some of Cassandras top customers are Activision, coursera, and instana.

The fifth database to process real-time queries for large amounts of data is Apache Spark. According to Apache Sparks' website, "Many organizations run Spark on clusters of thousands of nodes. The largest cluster we know has 8000 of them. In terms of data size, Spark has been shown to work well up to petabytes. It has been used to sort 100 TB of data 3X faster than Hadoop MapReduce on 1/10th of the machines, winning the 2014 Daytona GraySort Benchmark, as well as to sort 1 PB. Several production workloads use Spark to do ETL and data analysis on PBs of data."[12] Apache Spark achieves high performance for both batch and streaming data, using a state-of-the-art DAG scheduler, a query optimizer, and a physical execution engine. Spark extends over 80 high-level operators that make it effortless to build parallel apps. You can use it interactively from the Scala, Python, R, and SQL shells. A major advantage of Spark is that you can run Spark using its standalone cluster mode, on EC2, on Hadoop YARN, on Mesos, or on Kubernetes. Access data in HDFS, Alluxio, Apache Cassandra, Apache HBase, Apache Hive, and hundreds of other data sources. The next section will discuss when you should choose Apache Spark as a data processing engine and its significance towards IT and business.

Apache Spark is a powerful tool that lets you process big data sets faster by dividing the work up into chunks and allocating those chunks across resources. Apache Spark can easily handle up to petabytes of data and can also handle thousands of physical and virtual machines at one time. A user or business should choose Apache Spark if they are already using one of its supported languages such as Java, Python, Scala, or R. A user should choose Apache Spark when they are working with distributed data or NoSQL databases. Finally, a user should choose Apache Spark when they are using functional programming. The most important advantage that sticks out to me is that Apache Spark should be used when performing machine learning tasks. There are always reasons when a user should choose Apache Spark, but it is also imperative to know when to not use Apache Spark. According to Allison Honold, "For multi-user systems, with shared memory, Hive may be a better choice[2]. For real time, low latency processing, you may prefer Apache Kafka[4]. With small data sets, it's not going to give you huge gains, so you're probably better off with the typical libraries and tools." [13] The next section will discuss what Docker and Kubernetes are and what are they used for. Detail on its significance towards IT and business will be discussed as well.

Docker is a containerization platform and Kubernetes is a container orchestrator for container platforms like Docker. Both tools used containers and these containers solved the imperative issue of portability allowing you to separate code from the infrastructure it is running on. Developers could gather everything up they worked on into a small container image. When it comes time for production these containers can be ran on any computer that has a containerization platform. According to Katie Lane, “Kubernetes and Docker are both comprehensive de-facto solutions to intelligently manage containerized applications and provide powerful capabilities, and from this some confusion has emerged. “Kubernetes” is now sometimes used as a shorthand for an entire container environment based on Kubernetes. They are not directly comparable, have different roots, and solve for different things. Docker is a platform and tool for building, distributing, and running Docker containers. It offers its own native clustering tool that can be used to orchestrate and schedule containers on machine clusters. Kubernetes is a container orchestration system for Docker containers that is more extensive than Docker Swarm and is meant to coordinate clusters of nodes at scale in production in an efficient manner. It works around the concept of pods, which are scheduling units (and can contain one or more containers) in the Kubernetes ecosystem, and they are distributed among nodes to provide high availability. One can easily run a Docker build on a Kubernetes cluster, but Kubernetes itself is not a complete solution and is meant to include custom plugins. Kubernetes and Docker are both fundamentally different technologies, but they work very well together, and both facilitate the management and deployment of containers in a distributed architecture.” [14]

REFERENCES

[1] Cafarella, M., DeWitt, D., Gadepally, V., Kepner, J., Kozyrakis, C., Kraska, T., Stonebraker, M., & Zaharia, M. (2020, July 21). *Dbos: A proposal for a data-centric operating system*. arXiv.org. Retrieved September 11, 2021, from https://arxiv.org/abs/2007.11112.

[2] *Data-centric architecture - a different way of thinking*. Vista Projects. (2021, May 28). Retrieved September 11, 2021, from https://www.vistaprojects.com/blog/data-centric-architecture/.

[3] *What is data as a platform (daap)?: Webopedia definition*. Webopedia. Retrieved September 11, 2021, from https://www.webopedia.com/definitions/data-as-a-platform/.

[4] *Saas versus saap: Meeting different business needs*. Canto. (2021, February 16). Retrieved September 11, 2021, from https://www.canto.com/blog/saas-vs-saap/.

[5] Purkayastha, S. (2021, June 4). *Api architecture: Components and best practices for building robust apis - rapidapi*. The Last Call - RapidAPI Blog. Retrieved September 11, 2021, from https://rapidapi.com/blog/api-architecture/.

[6] Django. (n.d.). Retrieved September 11, 2021, from https://www.djangoproject.com/.

[7] Kamaruzzaman, M. (2021, March 22). *Top 10 databases to use in 2021*. Medium. Retrieved September 11, 2021, from https://towardsdatascience.com/top-10-databases-to-use-in-2021-d7e6a85402ba.

[8] Apache Hadoop. (n.d.). Retrieved September 11, 2021, from https://hadoop.apache.org/.

[9] *BangDB - NoSQL database as a service provider*. BangDB = NoSQL + AI + Stream. (2021, September 1). Retrieved September 11, 2021, from https://bangdb.com/.

[10] Rangel, D. (2015). *DynamoDB: Everything you need to know about Amazon Web Service's NoSQL database*. Amazon. Retrieved September 11, 2021, from https://aws.amazon.com/dynamodb/.

[11] *Open source nosql database*. Apache Cassandra. (n.d.). Retrieved September 11, 2021, from https://cassandra.apache.org/_/index.html.

[12] *FAQ: Apache Spark*. FAQ | Apache Spark. (n.d.). Retrieved September 11, 2021, from https://spark.apache.org/faq.html.

[13] Honold, A. (2020, January 12). *The what, why, and when of Apache Spark*. Medium. Retrieved September 11, 2021, from https://towardsdatascience.com/the-what-why-and-when-of-apache-spark-6c27abc19527.

[14] *Kubernetes vs Docker*. Sumo Logic. (n.d.). Retrieved September 11, 2021, from https://www.sumologic.com/blog/kubernetes-vs-docker/.