Deep Science Publishing
https://doi.org/10.70593/978-93-49910-80-5_3

DeepScience
Open Access Books

Chapter 3: Architecting cloud-based infrastructures for medical device data

3.1. Introduction

Advances in technology and access to devices and sensors that can be connected to networks have provided enormous amounts of momentum for the development of the Internet of Things (IoT). The healthcare domain is positioned to gain from IoT and device connectivity; more solutions are seen being brought to the market for data-driven technology improvements using connected devices. Connected Medical Devices (CMDs) form a special class of devices in the Internet of Things, which include patient monitoring devices such as Computer Tomography SCANNER, Magnetic Resonance Device, Digital X-ray Machine, Echocardiography Device, Renal Dialysis Device, Pulmonary Function Analyzer, Pathological Lab Analyzers, and various sensory-based devices, as well as CMDs that rely on wireless remote connection using Bluetooth or Wi-Fi technology. CMDs are capable of capturing and reporting sizable volumes of valuable patient data which, if handled with the necessary security and confidentiality protocols, can provide immense value-add service opportunities in related clinical areas. However, due to device heterogeneity, standards misalignment, and privacy and security challenges, the connected device ecosystem faces data management, integration, and interoperability difficulties. Exploiting the value of CMD-driven data becomes complex due to the isolated data residing in silos that need to be brought together to provide a broader view over CMD utilization (Griggs et al., 2018; Karthick et al., 2019; Bajaj & Ansari, 2023).

Pioneered by early success with mHealth paradigms that relied on mobile personal devices, the concept of IoT in healthcare has expanded over the recent years to cover various use cases in the enterprise space where the matter is not only integration and management of CMD-related contextual data, but also their integration with enterprise patient data for a more holistic view of enterprise healthcare outcomes. In view of these developments, this chapter lays down a Proof of Concept solution architecture to enable low-overhead CMD secure data management, integration, and interoperability. The solution supports MOM and event-driven patterns for CMD data with provisioning for both synchronous request and callback-driven responses, and asynchronous task-driven commands (Mamun et al., 2023; Rodríguez et al., 2023).

3.1.1. Background and Significance

The Internet of Things (IoT)-enabled medical devices substantially improve how healthcare is delivered, enhancing care in patients' homes while decreasing costs and freeing up hospital resources. Considering that healthcare systems worldwide are under immense pressure to improve quality and decrease costs, the clinical and economic utility of these novel, point-of-care medical devices is undeniable. However, realizing this potential requires addressing significant hurdles within the current healthcare technology landscape. Specifically, the uses of these devices must first be streamlined so as to create validated, optimal diagnostic algorithms that improve patient outcomes. Furthermore, investigators and providers need robust, accessible development and testing infrastructures to rapidly develop and implement new clinical uses of these devices. Current cloud infrastructures offer distinct advantages for these efforts, enabling near real-time data integration, storage, and algorithm development and testing to transform existing IoT-enabled medical devices into mobile, point-of-care diagnostic platforms.

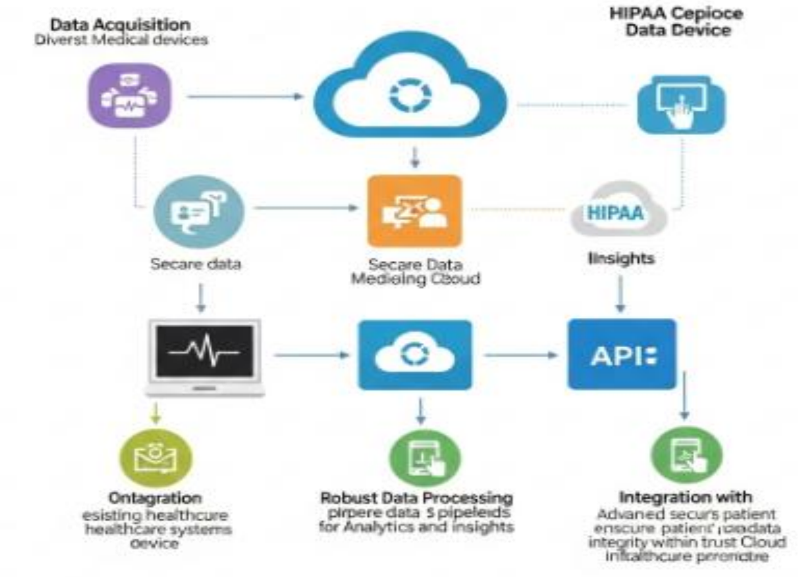

Fig 3.1 : Cloud-Based Infrastructures for Medical Device Data Management and Integration.

We present an overview of the unique challenges and opportunities in architecting IoT-based infrastructures for interoperable data management and algorithm development for mobile medical devices. Designing these testing infrastructures requires detailed consideration of numerous issues that span diverse, multidisciplinary technological and societal domains. Several key areas need to be addressed, distinguishing these data management and analytics models from those available previously – such as traditional research development testing infrastructure models for clinical laboratory analysis and general research computing verification and validation models for radiology imaging algorithms. Here, we focus on the most notable additional requirements for cloud-based development/testing infrastructures for algorithms based on IoT-enabled medical devices, specifically those associated with the specialized data characteristics of these devices compared to research medical devices or laboratory analyzers.

3.2. Overview of Medical Device Data Management

Traditionally, medical devices were standalone entities requiring little or no management. Physicians could use a medical device during a patient encounter, then pack it away until the next episode of care. Data were recorded during a patient encounter, the "three Rs" of biomedical recording, reporting, and research replay focused on emulsifiable arrays of hardcopy prints. During the past 40-50 years, the evolution of digital technology has made cellular, visual, audio, and intellectual property light, malleable information types. The management of coded digital data is easy; however, the volume of production is immense, although the primary challenge in managing medical device data is not data processing. Rather, the problem lies in intelligent management of digital data, that requires that devices be treated as malleable individual entities capable of acting autonomously as well as in concert as functional members of a team composed of other entities within the company of a cloud-based context.

Modern medical devices, like cameras, personal computers, video devices, telephones, and about any 'connected-circuitry' products are digital. While cameras in autonomous flying vehicles are at the limit of creating demand for cloud-based data management, a strategy that leverages the power of a cloud-based approach to digital device management should also scale up from small-scale coordinated device cloud groups in single locations or that move during episodic connections. In short, any technology-mediated path to information about a person or event embedded in a story that logically and effectively links device codes on cloud servers to secondary cloud data services presents a large potentially useful requirement for a growing body of information that multitasking medical sensors and devices already possess, or will inquire about and possess in the future.

3.2.1. Research Design

This Chapter highlights our design study research as developed in intervention research tradition from the fields of design science and architecture. We followed a multiphase approach, including observation, analysis, requirement specification, co-design, and evaluation phases. All study phases investigated design research cycles, defining, developing, and interfacing designs were central across all study phases. Our search for designs and design direction was motivated primarily by a specific use context, user information needs, user interaction, and usability concerns. Our qualitative study consisted of design information and requirement elicitation from focus groups and user interviews. The result was a requirement definition containing functional and non-functional requirements. Then we made design choices initiated by the requirements set in the previous part and guided by what had been explored. The design choices were the conceptual design of the MDDMS and the dialogues and other artifacts produced. An interactive system prototype was produced next, which modeled several design choices. Principles and strategies for the grasp for design choices were hierarchical vision, collaborative design, and iterating, designing, evaluating.

The peculiarities of such a software reside in the complexity, integration, adaptability, and heterogeneity, thus requiring the intervention of specific enlargement, molding, modification, and improvement actions. These traits and the critiques led us to envision a cloud-based MDDMIS, which is set on the architecture for MDDMS– clouds with functionalities, on the agent-based implementation – requesters and Microagents, and on the software-as-a-service distribution model.

3.3. Importance of Cloud Computing in Healthcare

Cloud computing has become an integral part of every industry, and the healthcare sector is no exception. The shift from paper-based records to electronic patient records is essentially paving the way for the widespread adoption of cloud computing solutions in healthcare. Healthcare organizations are increasingly becoming dependent on these solutions for improving efficiencies and cutting costs and also for addressing the ever-increasing regulatory compliance requirements. As this transition continues, health systems are forming strategic partnerships with technology companies, sharing the financial burden of infrastructure upgrades, and augmenting internal knowledge with external expertise. This marriage of the technology industry with healthcare is presenting an unprecedented opportunity for long-term investment returns in the field of cloud computing.

Moreover, as increasing numbers of healthcare organizations migrate to cloud-based infrastructure, cloud adoption by smaller healthcare organizations is also on the rise. The cost-effectiveness and convenience of cloud computing are also appealing to small practices that would need to allocate significant resources toward maintaining an in-house data center or dedicated IT staff. Additionally, as security measures like off-site backup, encryption, and redundancy provide cloud services with enhanced reliability and protection against data loss or breaches, smaller healthcare organizations are feeling comfortable storing their sensitive information in the cloud. As a result, the global healthcare cloud computing market is projected to surpass $13 billion with a CAGR of 14.8% by 2021.

3.3.1. Data Collection

The collection of data is one of the crucial actions of healthcare operations. Medical devices, hospital information systems, and workflow management tools are linked to the database to track patient-specific metrics, imaging information, and textual reports of the patient. Each medical center participates in collecting datasets, which are then collated and categorized. The purpose of these collected datasets is to improve individual healing dwell times and aid operations practically by availing timely patient data to all care service providers and reducing hospital expenses and associated costs. These datasets are also of interest for research institutions, large healthcare service organizations, or companies that revolve around developing healthcare business intelligence. These entities collate and categorize vast amounts of patient data that help insurance companies analyze and determine usage timelines and design their own plans as per necessary actions of the healthcare service providers.

As mentioned, there are multiple stakeholders in the management of patient-related data in a healthcare system. Medical devices installed in patient rooms (or carried by them) sense different metrics related to their health, like heartbeat, temperature, blood pressure, and many others and continuously relay this information to the central monitoring stations. The HISP is another major contributor to the collection of patient-related data. It consists of all the textual reports of the patients admitted to a hospital. Hospitals have been moving to an HIS deployment in recent years. These textual reports are of patients' history, surgery, X-rays, pathology reports, and whatever is available to correlate with the health of the patients. The data collated from HIS are of use for clinical research. By searching and filtering the datasets, the research scholars try to find correlations between certain factors, which cannot be precisely determined without large group datasets in restrictions. And the collected datasets also help mitigate chaos during aggressive patients’ workloads. These datasets have been utilized to optimize the care at healthcare service providers.

3.4. Architectural Patterns for Cloud-Based Solutions

Cloud-based Infrastructure solutions can be realized in a variety of architectural models. In this Section, we discuss three of these architectural models, namely, Microservices Architecture, Serverless Architecture, and Event-Driven Architecture.

3.4.1. Microservices Architecture Microservices Architecture is one of the most popular architectural patterns for Cloud-based solutions. This approach favors the decomposition of complex applications into a set of loosely-coupled services that are easy to develop, deploy, and scale independently. Each microservice is focused on a specific business capability and can be developed using different programming languages and frameworks. Additionally, microservices communicate with each other utilizing lightweight file formats and protocols, such as JSON over HTTP. This allows services created with different technology stacks and deployed in different hosting environments to interact seamlessly with each other.

The microservices pattern is well supported by Container Environments and, more specifically, by Container-Orchestration Engines. These tools allow easy deployment and large-scale orchestration of microservices while ensuring that containers conforming to the service definitions are always running. The Service-Orchestration Engine can also manage the load balancing and service discovery among running service instances. However, testing, deploying, and managing a microservices-based application is not trivial. Developers and operators of such systems need to deal with additional complexities in monitoring, logging, and ensuring security in communication.

3.4.2. Serverless Architecture Serverless Computing is an emerging Cloud Computing model in which the Cloud Provider runs the server and dynamically manages the allocation of machine resources. Serverless Computing frees developers from the need to provision and manage infrastructure, thus allowing them to focus exclusively on business logic. Serverless model allows developers to build applications hosted on the Cloud Provider’s infrastructure as small, stateless business logic functions that are executed in response to events.

A serverless application consists of functions, along with a set of triggers that execute these functions asynchronously in response to events. The Cloud Provider creates as many instances of a function as necessary to meet the incoming request rate, and the requester is charged only for the execution time. Serverless Architecture allows for the easy implementation of highly-scalable applications at a low cost, as the users pay only for execution time and the costs of idle time are reduced. Serverless Architecture is suitable for designing applications that embody complex event-driven workflows that can function independently of one another. For instance, an organization that needs to transfer and transform data from a third-party system into another could decide to implement these transfers on a business-function basis, with the business function invoked on a set schedule, triggered by events, or that are scheduled by an external event.

3.4.1. Microservices Architecture

Microservices architecture decomposes an application into small services independently executable, with a clear business objective and capability, using specialized protocols for communication between microservices over a network. Microservices architecture has gained momentum in recent years due, in part, to the large-scale growth of the internet and the need for high scalability solutions that support the explosive growth of users, devices, and the internet of things. Each microservice can be deployed, upgraded, and scaled independently, allowing for rapid innovation and responsiveness to fast-changing business goals. A microservices architecture also is generally more secure because if an attacker breaches one particular service, the damage is limited to that service and other services with different secure models can be employed to limit possible damage. Microservices architecture is normally implemented with an application orchestrator in front of the services that recognizes requests from clients and routes them to the proper microservice. In this way, a business application can perform multiple different functions or provide services that demand different locations, skills, tools, data systems, and processing characteristics. For any specific service, some businesses may need a real-time low-latency response to their user requests, while other noncritical but data-heavy transactions can be offloaded to a long batch-time processing application, freeing the high-demand microservices for more urgent requests. However, microservices are not panaceas and the development team must assess the size, goals, and vision of the business solve for designing a microservices-based approach. In general, microservices architecture is best for businesses and applications that want to grow and can afford development and orchestration of a microservices structure.

3.4.2. Serverless Architecture

Serverless computing offers cloud services that allow cloud-based infrastructures to execute business logic with minimal upfront effort to configure and provision cloud resources. Serverless architecture abstracts away the underlying cloud infrastructure through the use of functions as a service, while managing the execution of every request to the defined functions. Each individual function is a short-lived piece of code that is executed in response to specific events. This pattern has gained traction with many organizations using basic services. Functions are deployed in an ephemeral way, with only the specific execution packages needed for every function stored within the cloud provider's infrastructure. Some cloud providers also offer the ability to access event flows and resources without having to code and deploy every individual function.

One of the main benefits of serverless architecture is that it offers one of the simplest methods for achieving near instantaneous scaling of specific processes that are engaged to fulfill event requests. Because the cloud provider manages the underlying server infrastructure, organizations using serverless functions can drastically reduce the resources, costs, and effort involved in configuring and provisioning an infrastructure. Serverless architecture is ideal for environments with highly variable traffic that would otherwise require cumbersome and expensive container or virtual machine orchestration processes.

3.4.3. Event-Driven Architecture

Event-Driven Architectures (EDAs) are data-centric abstractions for decoupled, loosely-coupled communication between cloud components. EDAs are often used to facilitate interprocess communication by building a lightweight mechanism to decouple sender and receiver of events. Further, these events are emitted and listened to by different cloud components across the architecture, acting as triggers for workflows deployed and managed within a serverless function service. EDAs allow you to build highly-scalable system architectures and implementation patterns that can ingest streaming data from sources like website clickstream, telemetry device feeds, security information event management data, and many direct application APIs.

Events can be directly emitted using APIs or published to a messaging service, which then forwards and decorrelates matters to event handlers. Services allow you to push webhooks to callable HTTP endpoints when messages are sent or received. Solutions create Service Management events, establishing change jobs triggered as templates to be moved from a dev environment to a production system. Ingested public records for Postal Services stored in can be polled through a cron job that remove obsolete records and publish new messages to trigger any ETL or ML processes requiring geolocation functionality.

Event-Based architectures are well suited to ingest and transform public records stored in social media APIs known to reach high record sizes due to sudden burst of activity.

3.5. Data Integration Strategies

A common use case for storage, integrating data from different sources into a defined structure or model, is data integration. There are various strategies to include data integration into the Cloud-based infrastructure. One of the goals of the Cloud-based infrastructure is to find a balance between centralized and distributed use of technologies. Finding a strategy that is able to define a centralized structure of data while supporting distributed and local sources of information is an important task in the design of the infrastructure.

APIs are one of the simple tasks to perform decentralized integrations of data sources into a pharmacy chain system. A pharmacy chain system can be separated into a centralized core process and local or decentralized local systems. The local systems are mainly Point-of-Sale solutions, and the core processes are aggregation and storing. They needed to integrate information from the local systems into the centralized pharmacy chain. As a strategy, they supplied some APIs of the core solution. Meanwhile, the pharmacy local partner developed Point-of-Sale applications. The APIs were developed with a focus on user authentication, data integrity, and minimizing data redundancy.

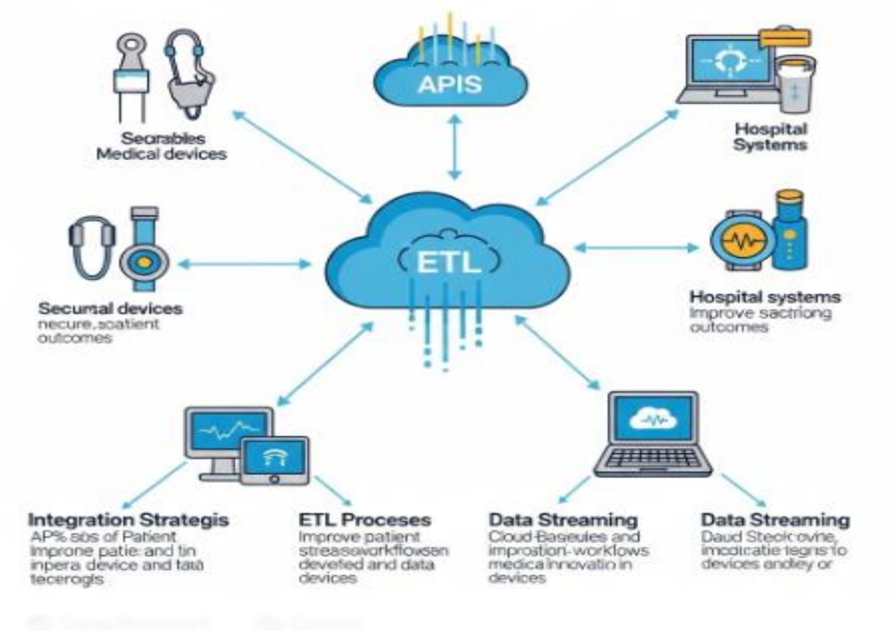

Fig 3.2 : Data Integration Strategies.

Data lakes are becoming a popular solution among organizations that are looking for a place to store all the data they generate. The adoption of the data lake has bloomed among various sectors, health care is no exception. Large health organizations looking for a new way to manage their growing data store have turned to this new technology in order to more effectively facilitate the sharing of data within their organization. While they expect data lakes to be a hub for all data from various departments, they understand that data lakes are just one of the components to an effective data management strategy and that there is a need to create data warehouses feeds from the data stored in the data lakes.

3.5.1. API Management

Data integration is a crucial element of the ecosystem for medical device systems as it provides the technical foundation for data subjected to secondary processing. While a cloud-based system can be available by opening designated network ports in a traditional manner, a more modern approach establishes a software developer interface that makes designated cloud-based capabilities accessible over standard Internet protocols. APIs encapsulate designated functionality, enforce usage initiatives, and expose standardized data representations. APIs can be extensive and apply to many different types of functionality, especially if the primary goal of a cloud-based system is to provide web services.

APIs can also be narrow in scope, designed primarily for serving specific types of access or user interactions. This is common with API gateways that primarily forward API requests to numerous downstream services, each of which provides additional, often more specialized APIs. And API gateways are also interfaces for accessing non-HTTP-based services, such as remote procedure calls, and often provide security, traffic management, availability, and request logging and monitoring capabilities. Gateways provide a management layer that is separate from both the upstream service provider and from external applications and users. The value of APIs makes them singularly popular among both service providers and users, making APIs into meal tickets for some companies, and leading to API fatigue among numerous technical developers who must adhere to potentially hundreds of different APIs from dozens of different services.

3.5.2. Data Lakes and Warehouses

The emergence of cloud computing has enabled the advent of new data mass storage technologies, distinct from usual CRUD operations enabled by relational databases or NoSQL. Big Data explosion has made traditional database technologies unable to keep up with current needs from applications, as well as business requirements of cost reduction due to the legends of the thousands of dollars per terabyte of storage in top billing for relational databases, not counting the costs of licenses, hardware, and engineers trained in such systems. Cloud data mass storage has manifested at two levels mainly: data lakes, to provide the right level of storage for unstructured and semistructured data at lower cost; and data warehouse platforms, that use cloud computing to enable enterprises to perform demands for quick queries on massive amounts of structured data, focusing on low response times and high throughput for analytical tasks.

Data Lakes are typically used for keeping low-value data, that is, the primary usage of data lakes are for companies to keep for inexpensive access in the future, data that no one is sure if it has any value. For instance, social media data, to keep for possible future inquiries about worldwide topics in the past. Data lakes can accept virtually any format and structure, with the primary condition of a low cost per terabyte. Then, if the data lake earns the right of becoming a data warehouse with high costs for storage and access, it can also hold structured data, with many indexing or other performance optimization strategies deployed over the enterprise data components for queries at urgent timelines.

3.5.3. ETL Processes

Extract, Transform and Load (ETL) processes are commonly deployed by cloud-disable organizations as their primary means of migrating data from on-premise applications or databases to a cloud. These processes are usually employed to move data between two physical stores, running in the same or different environments, and consist of three fundamental operations. The first step, the extraction, identifies and exports records from the source. Usually, a batch strategy is implemented to extract blocks of records from the source database, where in each run, thousands or millions of records are extracted. The second step transforms the extracted dataset. Common data preparation operations are applied in this stage, including format conversions, data type changes, data cleansing, or other functions. The last step in the process is loading the transformed dataset into the final target, usually a cloud-based data lake or warehouse.

ETL processes are widely considered a heavyweight strategy, due to their backend-centric nature. They introduce an inevitable migration delay during the initial bulk transfer from the source to the destination and require temporarily storing records in intermediate stores or infrastructure. Many ETL tools struggle with data consistency in case of certain types of updates, such as deletes at the source during the process execution. Additionally, traditional ETL tools and integrated environments are inflexible and heavily depend on the destination stores implementations. Enterprises nowadays want both on-premise databases and cloud-based resources to be integrated into a coherent system. For many organizations this means being able to move data back and forth with minimal effort. Results in both areas, consistency implementation for both low latency access and data process execution and storage for low volume data movement can be applied in the realization of a bi-directional ETL infrastructure.

3.6. Security and Compliance Considerations

Ensuring security is one of the foremost considerations for the cloud-based structured medical device data infrastructures. There are several reasons for this: Risks of data breaches, service interdependencies, and poor access control are a few potential pitfalls. These could have detrimental effects as the data housed in the structured cloud-based infrastructures could be for a group of patients undergoing trials to monitor the efficacy of computerized brain stimulation or for patients using a novel computerized noninvasive respiratory airway assist medical device in their homes. These diverse patients are at risk due to chronic diseases or suffering from neurodegenerative conditions. Potential consequences of any exposure or breach may include loss of privacy, distressed harm, or public embarrassment due to the exposure of vulnerable patients.

3.6.1. HIPAA Regulations

Health Insurance Portability and Accountability Act of 1996 Privacy Rule states that it sets national standards for the protection of individuals' medical records and other personal health information by requiring appropriate safeguards to protect the privacy of health information. The Privacy Rule applies to health records in electronic and coded formats. HIPAA sets rules defining the circumstances under which an individual's personal health information may be disclosed. In addition to privacy, the Act sets forth Security Standards to protect the confidentiality, integrity, and availability of electronic protected health information. It has three types of safeguards: Administrative, Physical, and Technical. Each type of safeguard has specific requirements. We will not be able to cover every aspect of the Act but focus on aspects related to the cloud-based infrastructure.

3.6.2. Data Encryption Techniques

Network security is key for ensuring remote communication security and ensuring data transfer over untrusted networks. The most common encryption techniques used today are Advanced Encryption Standard, Rivest-Shamir-Adelman, Elliptic Curve Cryptography, and hashing with Secure Hash Algorithms. AES allows 128, 192, and 256-bit keys for encrypting 128-bit blocks while RSA primarily uses 1024 or 2048-bit keys. ECC uses much smaller keys while ensuring similar-level security relative to sizes of RSA.

3.6.1. HIPAA Regulations

The Health Insurance Portability and Accountability Act is a U.S. law that provides guidelines for the protection of individually identifiable health information. The HIPAA

Privacy Rule establishes a national set of standards for the protection of certain health information which implements appropriate safeguards to protect the privacy of personal health information by relating to a patient's past, present or future physical or mental health, and prevents unauthorized use or disclosure of that information. The protections imposed by the HIPAA Privacy Rule are in addition to any other restrictions imposed by law. While the HIPAA law was enacted in 1996, the regulations were released in 2002 and were first significantly expected to be enforced in April 2003. These initial regulations were expanded in 2012, with major changes addressing enforcement and accounting for disclosures specifically affecting cloud-based solutions for HIPAA regulated data.

The HIPAA regulations apply to what are called "covered entities," which, with respect to medical device data management and integration, are typically healthcare providers. Covered entities may also be health plans and healthcare clearinghouses, provided that such entities transmit any health information in electronic form in connection with a HIPAA transaction. The HIPAA regulations also contain provisions that apply to business associates of covered entities, which are persons performing functions on behalf of a covered entity involving the use or disclosure of protected health information.

3.6.2. Data Encryption Techniques

Encryption can be accomplished using hardware, software, or a combination of both. Typical uses of encryption involve protecting data at rest, data in motion, and data in use. User data must be encrypted before it is transmitted to minimize theft. Data at rest on mobile devices, as well as servers, must store user data in an encrypted format. Computer processors assist or enable data encryption and decryption procedures. Special-purpose encryption chips are extensively used in payment systems to safeguard sensitive financial transactions. Tamper-proof libraries are often used on servers to assist in the authentication processes of various transactions. Software protocols are often used to facilitate secure data transmission. However, recent vulnerabilities in protocols have driven industry standards towards the use of approximately 150 cipher standards to implement robust encryption technologies.

An important consideration in encrypting data is the management of cryptographic keys. Sophisticated key management facilities can lead to a compromised encryption technology, as was the case with the large-scale compromise of an encryption method. This puts additional pressure on a cloud service provider's staff who are responsible for managing both service availability and key security. It becomes critical, therefore, for you to develop a strategy for using and managing encryption technologies that balances your legal requirements with usability and performance. Encryption has often been perceived as impacting the performance, reliability, and responsiveness of mobile, as well as cloud-hosted, applications. Advances in hardware-assisted encryption approaches, as well as the growing security threats imposed by hackers specializing in sensitive data, have lowered concerns related to using encryption.

3.6.3. Access Control Mechanisms

Access control concerns the organization of users' communication among the various architectural entities based on users' identification and authentication processes, followed by access control mechanism testing. The concept of "how" the operation will organize is captured in the concept of an access control mechanism; these are classified like this: password-based, token-based, biometric-based, and access policy-based. Authentication is the verification of who is the actual user based on the comparison of the identification with electronic characteristics that are generated from its data using hashing functions. Authorization is the process of controlling the access to a resource based on the enforcement of access control policies against the actual user.

Password-based authentication is by far the most common aspect of access control and consists of storing a hash and salt of the user password. The hashing function must be amended with salting to make its use safe against dictionary attacks. An additional authentication layer could be considered and consists of the use of one-time passwords sent to the users via email or short message system on their mobile devices. However, this technique has been compromised a lot, and its use is losing steam. Token-based authentication mechanisms rely on an external device that generates codes that must be inserted manually to authenticate users. This method is similar in concept to the additional authentication layer. An additional mechanism that is growing in popularity is the usage of biometric characteristics like fingerprints, iris patterns, face recognition, or voice recognition. Although things are moving slowly in this direction, password-based authentication is a technique that remained stable over many years and showed good usability; thus, it will probably continue to be the preferred choice for a long time.

3.7. Scalability and Performance Optimization

In the era of big data, it is rare, in fact, impossible to build data management and analysis infrastructure that is large enough to be able to handle the anticipated growth in data size and activity. Architecting systems, therefore, that can scalably handle performance demands is one of the most critical components, if not the most critical component, of the design. A scalable architecture is one that is designed for easy, adaptive growth, based on the analysis of future trends regarding data size and activity. Simple, unfancy concepts such as load balancing and caching turn into extremely important concepts when it comes to scalability. Together, they can often provide solutions to data handling scalability problems, especially when they are applied to devices that deal with a metaphorical double life of acting as both clients of the main data handling infrastructure and servers for local clients. Local clients generally demand access to devices for a greater portion of the day than people demand access to online sites. Thus, architecting specialized devices to cache the data for hours during which local demand is at a peak can effectively reduce the demand on the handling infrastructure during peak usage hours and speed up access time for nearby users at the same time.

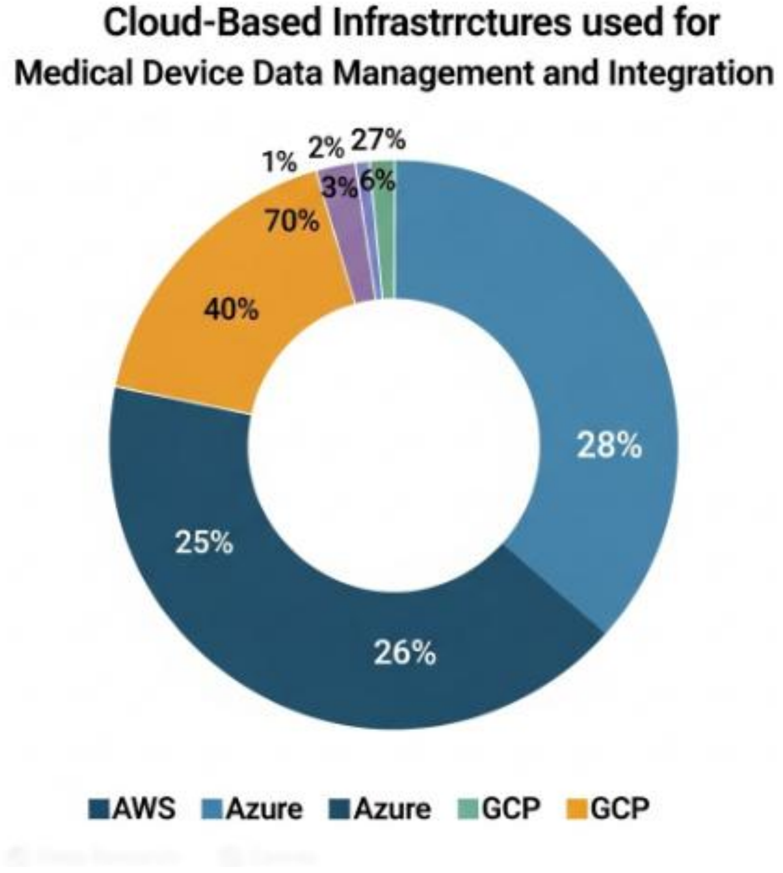

Fig : Cloud-Based Infrastructures for Medical Device Data Management and Integration.

Load balancing is one of the simplest and yet often most effective mechanisms for scaling up performance in a cloud-based data store infrastructure. By distributing the workload for a service among multiple data store clusters, we are able to provide users with lower access latency and higher throughput than can be provided by a single cluster, as well as picking up the slack when one of the clusters happens to be down due to hardware or software failure. Furthermore, distributing data to be processed among multiple storage devices and processing elements allows larger than can be handled in a reasonable time by a single device or element. New data can also be added to existing data stores. Similar parallelization in the local area through the use of local cache clients increases response speed for local users and helps to limit response time during peak online usage. By distributing the work across hundreds or thousands of geographically distributed servers, cloud processing utilities help users meet data-access performance goals.

3.7.1. Load Balancing Techniques

In distributed systems, entities can interact with the system in different fashions and through multiple avenues and approaches. Such as through a mobile app hitting an API, a partner accessing an API through a web app, or a device sending data to the system via a data ingesting API. Inherent to these avenues, approaches, and interactions, become multiple load events that can reach the system and if those events become excessive, they can affect the behavior of the APIs or the performance of the distributed system as a whole. This is why it is important to create distributed systems that are capable of spreading the load or are elastic, containing the algorithms or processes that do that, called load balancing.

The act of load balancing distributes the tasks on multiple instances of the system or on other infrastructures as a technique used to decrease the load or to optimize the performance. Load balancing may occur across many service instances of the same service type, across data partitions of the application, across microservices of an application, and in some cases it may balance the load across the entire geographically distributed clusters. Load balancing may also occur at different levels of the application. An application may do client load balancing using load balancer connectors, which are libraries that provide different services for that application. Or by using other load balancer technologies to provide that auxiliary service to the application. Load balancing decisions are made on the quality of latency, throttling, connection draining, and user-defined distribution algorithms. Factors for the initial load balancing decisions are availability zones, the service pool, user-defined tags, or weights. Factors for changing the load balancing decisions, balance between cost and proximity, and all decisions are adaptive based on load.

3.7.2. Caching Strategies

The volume of data collected by medical devices is large and increasing, particularly in the field of patient monitoring. Although relevant, not every single piece of generated data needs to be stored in the system, and frequent queries to the system to read mostly numeric historical data can become overwhelming and create performance bottlenecks.

It is essential to remove from the main database as soon as possible the data that is not needed and to make queries return only relevant data. Hence, caching mechanisms are fundamental to avoid swamp operations that fetch large amounts of data that are not necessary for the driving decisions made based on that cached data.

Using the concept of presentation layers, the uppermost layers consume data from user actions and responses and are thus a good place to leverage caches. Information is used and consumed for a given period, and data just prior or after that moment in time can be cached in these layers. These layers enable implementing various caching strategies like zone, device, and event caching to optimize memory use and improve performance. For a given zone, device type, or patient, the cache can retain the data temporarily, allowing quick retrieval and response to the user. In event caching, a user can pinpoint device or zone events, and the data from those periods can be identified and kept in memory until its utility has lapsed.

Both local and distributed caches are possible. Standalone devices and systems have memory limitations and renewal policies for local caches. However, they can be more effective than distributed caches or can be combined. Distributed caches using remote memory can eliminate the need to frequently access the back-end data models. These caches can also work based on cloud middleware indexes, reducing the number of database reads significantly.

3.8. Conclusion

A Cloud computing paradigm has introduced computer infrastructure as a service, without exposing the service outsourcing organization to the operational management of the shared infrastructure, investing in hardware or storage, or even on providing availability of the service with recovery at an affordable cost. They rely on the service provider for the security, availability, and recovery of the data in the shared data space. The risks associated with integrating this service with the enterprise environment, where the application hosted in the Cloud could have access to sensitive information or business-critical transactional data being extracted, transformed, and loaded using the Cloud-based service, has often prevented many organizations from considering outsourcing of business processes or functions such as Disaster Recovery, Data Warehousing, or Data Backup. While historically, such concerns around data ownership, security, privacy, and business continuity were confined to enterprises from regulated industries, the need to compete in a global marketplace by adopting strategies to improve while optimizing operating costs has now made Cloud an attractive option for enterprises from all industries.

However, certain targeted and planned moves toward outsourcing relevant functions to Cloud vendors while ensuring compliance to the regulatory environment, good practices for security, privacy, availability, performance, etc., have or can facilitate the transition. The relative ease of accessing IT capabilities through Cloud-based services is especially strong for enterprises whose strategic initiatives such as Internet-based innovations, or business model restructuring, combined with business pressures, require compelling business cases for reducing or deferring IT investments. It is also true for enterprises with limited IT management capabilities. In addition, certain technology-driven market sectors such as Internet-based content or service companies, with changing market focus, may be attracted to using Cloud Computing mainly for flexibility.

3.8.1. Future Trends

Healthcare continues to experience explosive growth in the adoption of technology and the nature of this technology continues to change rapidly. Advancements in wireless communications, implantable devices, wearables, mobile health systems, and the Internet of Things enable a new set of services to become possible and viable. Cloud-based architectures for medical data management and integration are beginning to integrate and enable these new services. When used in conjunction with the latest healthcare regulations, these architectures can enable a healthcare information technology commons, a barrier to entry for the development of new tools and services in healthcare.

For new services to develop and flourish, it is essential that healthcare continue to adopt a data liquidity approach. This is only possible if, over time, the major EMR vendors continue to open Application Programming Interfaces to ensure update and provide location completion of specific data types.

References

Mamun, A., Alam, M., Hasan, Z., et al. (2023). IoT-Based smart health monitoring system: Design, development, and implementation. In The Fourth Industrial Revolution and Beyond (Vol. 980, pp. 129–158). Springer. SpringerLink

Bajaj, V., & Ansari, I. (2023). Internet of Things in biomedical sciences, challenges and applications. Indian Institute of Information Technology, Design and Manufacturing. SpringerLink

Rodríguez, E., Otero, B., & Canal, R. (2023). A survey of machine and deep learning methods for privacy protection in the Internet of Things. Sensors, 23(1252). SpringerLink+1SpringerOpen+1

Griggs, K., Ossipova, O., Kohlios, C., et al. (2018). Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. Journal of Medical Systems, 42(7), 130. SpringerLink

Karthick, R., Prabaharan, A., & Selvaprasanth, P. (2019). Internet of things based high security border surveillance strategy. Asian Journal of Applied Sciences and Technology, 3, 94–100. SpringerLink