Do CHANGE platform

International Journal of Medical Informatics 117 (2018) 103–111

Contents lists available at ScienceDirect

International Journal of Medical Informatics

journal homepage: www.elsevier.com/locate/ijmedinf

Do CHANGE platform: A service-based architecture for secure aggregation and distribution of health and wellbeing data

Idowu Ayoola[a,b,*], Mart Wetzels[a], Peter Peters[a], Sander van Berlo[b], Loe Feijs[a]

[a] Designed Intelligence, Department of Industrial Design, University of Technology, Laplace 32, 5612 AZ Eindhoven, The Netherlands
[b] Onmi B.V., Torenallee 30-10, 5617 BD Eindhoven, The Netherlands

ARTICLE INFO

Keywords:
IoT
Personal data
Health services
Privacy
Security
Data aggregation
Data distribution

ABSTRACT

Over the last decade, the adoption of open API standards offers new services meaningful in the domain of health informatics and behavior change. We present our privacy-oriented solution to support personal data collection, distribution, and usage. Given the new General Data Protection Regulations in Europe, the proposed platform is designed with requirements in mind to position citizens as the controllers of their data. The proposed result uses NodeJS servers, OAuth protocol for Authentication and Authorization, a publish-subscribe semantic for real-time data notification and Cron for APIs without a notification strategy. It uses Distributed Data Protocol to control and securely provision data to distributed frameworks utilizing the data and those distributed applications are exemplified. The platform design is transparent and modularized for research projects and small businesses to set-up and manage, and to allow them to focus on the application layer utilizing personal information. This solution can easily be configured to support custom or new data sources with open API and can scale. In our use cases, maintaining the separate ecosystem services was trivial. The adopted distributed protocol was the most challenging to manage due to its high RAM usage. And implementing a fine-grained privacy control by end-users was challenging in an existing clinical enterprise system.

1. Introduction

The field of the quantified-self integrates technology for data acquisition in a person's daily life [1]. Targeted parameters include dietary-intake [2], home-environment air-quality [3], emotional state [4], and physical or mental performance [5]. Quantified-self has become an inevitable trend in the domain of personalized healthcare and the cornerstone of numerous health services and platforms. Existing mashup services that aggregate data from various sources are limited in their integration [6] and the increased privacy regulation—like the European General Data Protection Regulation [7]—introduces new rules in securely obtaining, storing, and sharing data. Although proposed standards for electronic health care transactions such as HL7 FHIR [8] and HIPAA [9] introduce a common approach for presenting data, real-life services often maintain their own implementation [10].

In the recent years, authorization of data sharing has been standardized using the OAuth2.0 authorization framework [11] and has become an industry standard for authorization. Many enterprises have implemented this protocol to offer their customers the ability to grant third-party access to their data. This feature enables secure access control to personal data on which a large number of new services depend on (e.g., MyFitnessPal can securely interchange personal data with a Fitbit data source, and see all activity and nutritional meal summaries in one place). Popular trackers such as Fitbit, Beddit, Moves, Strava, Runkeeper, Peloton, MapMyRun, MyFitnessPal, Waterlogged, etc. use OAuth2 for granting third-party access.

Aggregation of different types of personal data, such as fitness- and sleep-trackers, require multiple devices and mobile applications to be used by end-users. Using multiple apps and having a scattered overview of personal data is a problem, which can be addressed by providing an aggregation service and a single interface that makes data actionable and manageable. An example of such an aggregation service is Human API. The Human API Data Platform provides normalized data for researchers or companies to utilize in their services. Similar, well-known, services are Apple HealthKit,[1] GoogleFit (see footnote[1]), and Microsoft HealthVault (see footnote[1]). These services provide a blended overview of health data that is available and provide access to these data for new health and fitness solutions. Some of these platforms have received considerable attention for improving wellbeing; however, they offer services in different ways. There are still challenges to unify the data

* Corresponding author.
E-mail addresses: i.b.i.ayoola@tue.nl, idowu@onmi.design (I. Ayoola), m.h.wetzels@tue.nl (M. Wetzels), p.j.f.peters@tue.nl (P. Peters), sander@onmi.design (S. van Berlo), l.m.g.feijs@tue.nl (L. Feijs).
[1] See Table 1 for reference.

https://doi.org/10.1016/j.ijmedinf.2018.06.004
Received 16 January 2018; Received in revised form 1 May 2018; Accepted 6 June 2018

Table 1
List of websites and GitHub repositories.

Name	Author/Organization	URL
Apache Kafka	Apache Software Foundation	http://kafka.apache.org
Apache Storm	Apache Software Foundation	http://storm.apache.org
Apple Health	Apple Inc	http://apple.com/ios/health
Auth0	Auth0 Inc.	http://auth0.com
Do CHANGE	European DoCHANGE Consortium	http://do-change.eu
Docobo	Docobo Ltd	http://www.docobo.co.uk
GraphQL	Facebook Inc	http://graphql.org
Human API	Human API	http://humanapi.co
MeteorJS	Meteor Dev. Group Inc.	http://meteor.com
Microsoft Health-Vault	Microsoft	http://microsoft.com
MongoDB	MongoDB Inc	http://docs.mongodb.com
Moves	Oy, ProtoGeo	http://moves-app.com
PM2	Keymetrics	http://pm2.keymetrics.io
React Native	Facebook Inc.	http://facebook.github.io/react-native

derived from these services [12]. [13] proposed a query method named the Aggregation Query Language (AQL) that uses a proxy system for users to easily invoke multiple Web APIs by SQL-like statements. They later extended AQL in [14] to compensate for its lack of OAuth support and lack of data organization support. Despite the wide adoption of the OAuth standard, many implementations of this standard remain inconsistent or vulnerable [15] to well-known attacks such as the Cross-Site Request Forgery (CSRF) attacks. It is easy for developers (especially those in small R&D teams) to overlook critical implementations that are not enforced by the mashup services. For us, the proper utilization of the OAuth scheme and a security-sensitive organization and re-use of user data and access tokens are fundamental to the design of the Do CHANGE platform. Instead of the just-in-time query approach in retrieving user data from the authorized sources, we also follow a notification approach (which some of the mashup services currently supports) to retrieve, organize and distribute the user dataset reactively.

Privacy is a prevalent issue that must not be overlooked [16] since the services hold personal information from disparate sources and enable richer knowledge of the end-user's behavior. The adoption of these services and personalized devices raise many privacy concerns (see [7]); to mention a few: (1) data ownership often remains at the vendor who reserves the right to use, commercialize or share personal data, (2) company's non-conformance to government regulations on the storage location and usage of personal data, (3) third-party services also holds the right to use and store personal data as they wish after granting them access, but without the means for the owner to revoke access and delete the shared dataset.

Despite the privacy concerns mentioned, the existing platforms are limited to a number of features. But often essential features are lacking, to mention a few, they do not offer (1) persistent control (i.e., personal data retrieved by third-party services cannot be redrawn) and data management, (2) partial data distribution to third-party services as controlled by the end-user, (3) the possibility for on-premise deployment of the data aggregation and distribution services that allow consumer teams (i.e., research teams) to take full responsibility for the personal data in their possession. Our approach deals with how to design the platform using micro-services to improve maintainability, security, scalability and privacy control. The proposed platform has two main parts—aggregation and distribution. In the following sections, we will share the design of the Do CHANGE platform, present working use-cases, and highlight the shortcomings of the current implementation.

2. Information security and privacy considerations

New regulations concerning personal data protection in the European Union are in [7]. It describes citizens to have a right to (1) receive clear and understandable information about who is processing their data; (2) consent organization's rights to access their data; (3) ask organizations to delete their data, etc. The design of the Do CHANGE platform addresses some of these aspects.

2.1. Securing personal data by organization

The organization holding personal data must comply to the various types of laws and regulations that impose security and confidentiality obligations on the business and potentially impact the service initiatives involving data location, records management, privacy and security controls. The location of personal information is often outside the reach of the service providers utilizing the information, and so the proposed design aims to be relatively easy to configure to enable the organization to receive the data and store them in the location of the appropriate authority's jurisdiction. The environment hosting data and the data services should be secured and can implement IP address whitelisting to ensure that interactions between sub systems and processes do not compromise the security of data and its applications. Additionally, all access points must be secured using valid authentication and authorization rights.

2.2. Identity and access management

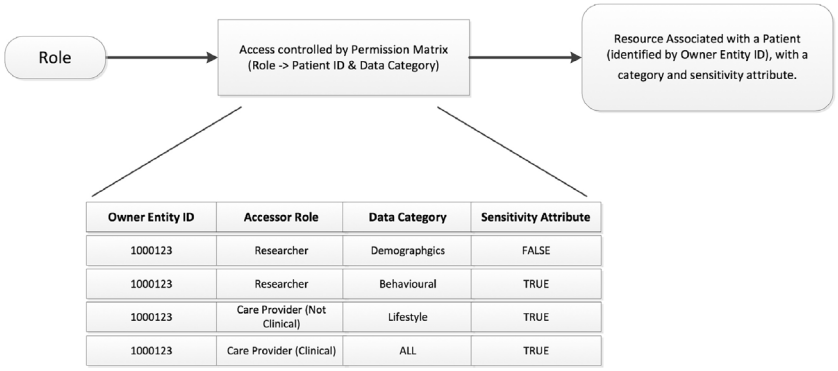
Users can initiate access to third-party data providers through OAuth protocol and grant access using their login credentials. The proposed system utilizes Auth0[2] as a Federated Identity Provider. In practice the Identity Provider can be of any kind, Auth0 uses OAuth standards which is widely adopted. Auth0 allows applications to choose the location to store user information and comply with appropriate regulations. Auth0 can also support JSON Web Token (JWT) [17] management, the setup of APIs, Delegation Access and applications with access control support. In practice, the platform aggregation and distribution services should be unaware of users' linkable identity to safeguard their identity if access is compromised. All profile related information that contains the username, age, demographics, etc., are thereby stripped off the records and discarded (maintaining pseudonymity [18]); however managed directly by the Federated Identity Provider. Only the pseudonymized identifiers as provided by Auth0 can be present within the data aggregation and distribution services.

At the time of writing, JWTs security tokens issued by Auth0 has been used solely for identity verification between the proposed ecosystem services and its extended services. To simplify integration testing, the enforcement of access preferences per application has not yet been made. Potentially this will be achieved using Delegation, an advanced access control feature offered by Auth0. For example, if application “A” which has a valid access token for a user, it can then call application “B's” API while controlling access according to the user's preferences for application “A”. Finer grain access control to the level of individuals, groups of individuals, individuals with particular roles or organizations, is then implemented locally by each service/sub-system.

2.3. Distributed resource management and protection

Data protection concerns the ability to control access, to secure data while at rest, in transit, and in use. In production settings, standard data protection practices should be in place such as using Secure Hypertext Transfer Protocol (HTTPS), firewall, removing logging that may compromise security, etc. For the Do CHANGE system, we are also

[2] See Table 1 for reference.



Owner Entity ID	Accessor Role	Data Category	Sensitivity Attribute
1000123	Researcher	Demographgics	FALSE
1000123	Researcher	Behavioural	TRUE
1000123	Care Provider (Not Clinical)	Lifestyle	TRUE
1000123	Care Provider (Clinical)	ALL	TRUE

Fig. 1. Role based permission matrix.

concerned with the management of data even after the data has been distributed to various organizations (i.e., consumer sites). Most aggregation systems leave this control up to the data holder once they have been granted access. We aim to put the user in control by allowing them to set individual's resource distribution or access rights, which can result in adding, updating, or deleting data on site.

To provide coordination between each service within the Do CHANGE ecosystem, conceptionally,[3] we propose a common trust and permissions model. In this context, the user shall be able to exhibit different levels of granularity regarding controlling access to their information. At the highest level, they may wish only to control access by role. To facilitate such access control, each service shall effectively maintain for each end-user, details of the access permitted by role for different categories of information and should further enforce these rights where possible; this is summarised in Fig. 1. In this example, a Researcher may access non-sensitive demographics (e.g., age, gender, social group, ethnicity, general location) and all behavioral data, a non-clinical carer may only access lifestyle data, while a clinical care provider may access all data.

To facilitate finer-grained access control, conceptionally (see footnote[3]), each service may optionally and by its scope support an entity based permissions matrix as indicated in Fig. 2, where an entity may be a person, an organization or a service. Hence, this controls the access that specific entities in the organization are permitted to have access to a user's data.

3. Overall system architecture

The ecosystem platform aims to aggregate and distribute personal data. It is built as a multi-service application platform, which comprises of micro-services that each implements its own API that runs independently. The nature of the system architecture allows for distributed deployment and scaling up the number of application processes in expanding the system. There are three parts to the entire ecosystem as shown in Fig. 3—the aggregation part (for collecting user data), the distribution part (for sharing user data) and other organisation specific services that utilize the data. Data from third-party providers are managed using Consume [19] & Collect, data from the custom devices are provisioned using Channel, the aggregated dataset are distributed using Connect, and the Connector, which is placed at the organization site help's to securely obtain the data.

[3] The way that a service supports the common trust and permissions model is an implementation decision. At one extreme, a service dealing with a wide range of data and stakeholder types may implement it fully. In another extreme, a service dealing with a narrow range of data types and stakeholders/relationships between stakeholders may effectively implement it because of these constraints. The key thing is that they effectively implement the model and support communication of its associated data categories, roles and data access types on external interfaces.

The platform is predominately developed in NodeJS [20] due to its asynchronous nature and the broad applicability of Javascript [21]. Via the Business-to-Consumer (B2C) interface (see Fig. 3), the aggregation service collects data from external sources asynchronously. The collection services ensure that the user records are up to date while the distribution services reactively distribute the data to service providers that are permitted to access the data. The service providers (i.e., organizations) can, in turn, derive new information for the user and provision it via the Business-to-Business (B2B) interface.

4. Aggregation services

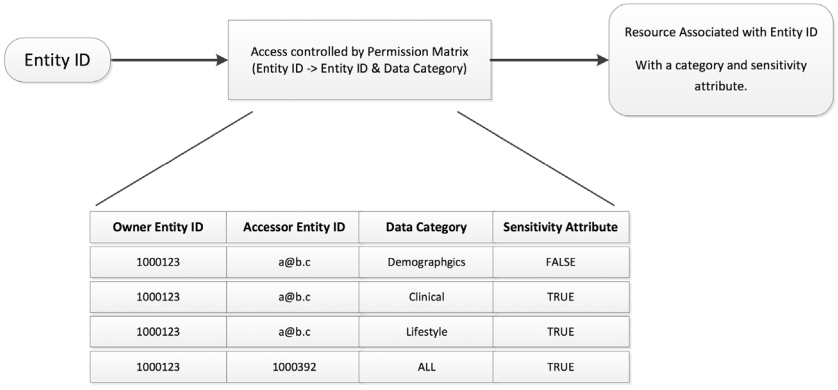
The aggregation section is responsible for acquiring user data. It can obtain data from external sources or the ecosystem private service providers. The collection services comprise of Consume (the authentication and authorization service), Collect (the request management service), and Channel (the private aggregation service).

4.1. Consume

Consume, [19], is part of the aggregation service that consumes third-party APIs. It is the B2C interface that links to the external data sources for the ecosystem. Consume facilitates the authorization and authentication for external API access. It uses OAuth [11] to grant third-party access in exchange for access tokens. It subscribes to third-party applications with a subscription API to receive notifications when new data is available. Consume itself does not store any data except for the access tokens. It provides end-points to enable pass-through-requests to the external APIs securely. Data can be fetched on demand or aggregated to a dedicated database. Considering the amount of contextual data needed to identify health-related behaviors, and provide personalized healthcare, Consume provides the flexibility to grow with current health technologies by removing the complexity of authentication, authorization, and token management.

4.2. Collect

Collect is a service that helps to extract user data from external APIs through Consume using HTTPS. It works with notification and Cron (a time-based job scheduler) to ensure all user data are up to date. The notifications received in Consume are passed through to Collect before performing the actual data request. Fig. 4 shows the sequence used by Collect in making requests based on notification services. All requests are secured by JWT, and the provider's request template is extracted for making external requests via Consume. Collect does not store any access token or user profile information. It stores the actual user data received on demand. All storage is pseudonymous and requires user lookup from the identity provider before data can be linked to a real



Owner Entity ID	Accessor Entity ID	Data Category	Sensitivity Attribute
1000123	a@b.c	Demographgics	FALSE
1000123	a@b.c	Clinical	TRUE
1000123	a@b.c	Lifestyle	TRUE
1000123	1000392	ALL	TRUE

Fig. 2. Entity based permission matrix.

Fig. 3. Visualisation of Do CHANGE ecosystem services, highlighting the aggregation and distribution sub-services.

user. Collect is developed in NodeJS, and Mongo is used as its private database.

4.3. Channel

Channel implements the interface for provisioning data within the ecosystem, and ready for distribution to data subscribers. The Channel service is used by internal service providers as supposed to Consume, that interfaces with external data providers. The internal data providers are authenticated with a valid JWT containing claims for the audience, issuer, user, date and a B2B data identifier. The received data is handled according to the claims and stored in the Collect database.

5. Distribution services

Data made available by the aggregation services are ready for immediate distribution via the distribution services (Connect & Connector). The design layout is as follows:

1. The Collect service stores users' data in its secured database, which is accessible by Connect. Connect is the Data Controller for connecting the data to other services/applications through its publication mechanism.
2. With the Connector application, the organizatioon subscribes to its required publications provided in 1 above.
3. The access rights of the organization are checked by Connect through a Federated Resource Manager.
4. A subset of the information subscribed to is sent to the Connector based on its access rights.
5. The Connector maintains a copy of the dataset received in 4 above. All later changes to the dataset are propagated to the Connector.
6. The Connector enforces the changes remotely to its local database.

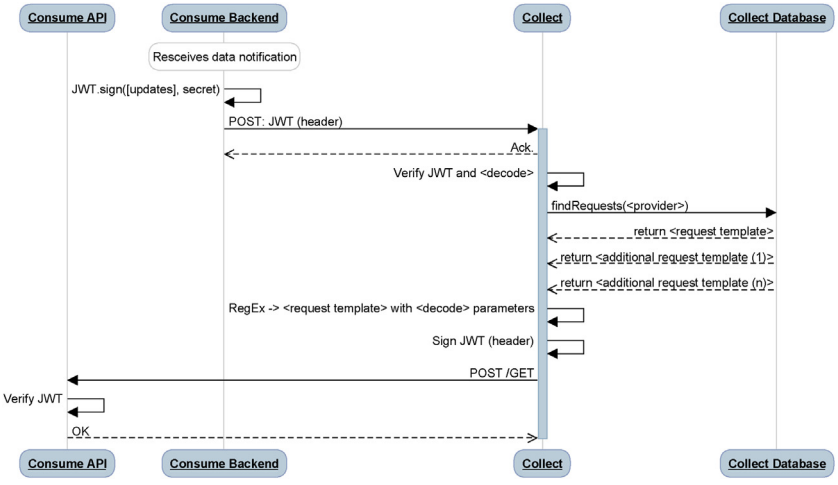


Fig. 4. Simplified sequence diagram for requesting data based on subscription notifications through Consume.

5.1. Connect

Connect is a server-side application responsible for publishing user data to the various distribution, or organization sites. The Connect service seats next to the aggregated user database, making the data securely accessible. The current version of Connect uses a server implementation of Meteor Distributed Data Protocol (DDP)[4] to serve data to connected clients. DDP is based on JavaScript Object Notation (JSON) and WebSockets. The protocol has a publish and subscribe regime for its server and client implementations respectively.

By using a DDP server, various publications are created and protected on a client basis. For example, a publication can provide all user data while another gives limited access to user data. The application publishes only documents updated within 14 days. Once subscribed, the publication will return a cursor to the user database collection. The Connect application also implements two DDP methods[5]—one for requesting the unsynchronized user data identifiers and the second for requesting the actual data by their identifiers. These methods are useful for synchronizing the remote database of the Connector at startup.

The DDP protocol provides three types of notification to the client application according to the changes in the publications:

1. **added** – this applies when a new item is added to the local data set;
2. **changed** – this applies when an item in the local set is updated;
3. **removed** – this applies when an item is removed from the local set or publication.

The current version of Connect does not implement a common trust and permissions model, and therefore all the access rights are hard-coded.

5.2. Connector

The Connector is the client application that subscribes to user data remotely. The key requirement for the Connector is to maintain the distributed dataset reactively. It takes care of the ecosystem data access events for adding, updating and deleting remote dataset. Connector utilizes the client implementation of the DDP protocol for receiving related user data. Once initialized, the Connector login to Connect. It then uses a DDP method call to update its remote database. After updating, it subscribes to the data publications. Subsequent changes to the dataset- or changes to the access rights, are received and implemented by the Connector remotely. Fig. 5 is a function-sequence diagram that shows the interaction between the Connector and Connect.

Aside from the DDP client, the Connector also implements Apache Kafka[6] client. The purpose of using Kafka is to provision the synchronized user data to the rest of the remote system; still keeping the responsiveness and order of records. Apache Kafka conforms to the publish-subscribe regime and makes it possible for multi-applications to consume the same data concurrently.

6. Use cases

6.1. Mobile application (Vire and Synergy)

With new health mashups, making sense of multiple streams of well-being and contextual data for presentation on mobile devices has become more important than ever [22]. Vire is our application that does that. It transforms users information and visualizes them to be legible and meaningful. Vire aims to help cardiovascular disease patients with their rehabilitation, by stimulating a healthier lifestyle. Using various data sources, Vire provides three essential factors of daily living—Activity, Variety, and Social Opportunity.

Activity, Variety, and Social Opportunity are presented as scores that update throughout the day, Fig. 6. By viewing the scores regularly, users can learn about their behavior and exert continuous adjustments. Behavioral nudges, called Do's are consequently sent to the user to help them adjust their behavior one step at a time. Do's are personalized, also in response to the related variables. The Do's are based on the Do Something Different behavioral methodology [23] that helps to expand one's behavioral flexibility. Also, Vire helps users to capture images of their daily foods, which can be reviewed by a nutritionist for advice.

This use-case applies the Connector to obtain permitted user records. Synergy (the server-side application for Vire) automatically detects the information changes, which then updates Vire (the mobile applications) via DDP. Only running Vire with a logged-in user can receive data scoped to that user. The use of the platform makes Vire highly responsive to changes in the user information to be immediately visible. Besides presenting information, Vire also produces food images as a service. For that, Channel B2B interface is used to collect the photos to send to other service providers utilizing the data through the distribution services. Synergy and Vire are built using Meteor Framework,[7] and React-Native (see footnote[1]) respectively.

6.2. Data analytics platform (Nimbus)

Nimbus is the application that processes user data online. It aids the computation of critical variables abstracting user behavior useful for the user's or expert's interpretations. These variables include Activity, Social Opportunity, and Variability scores. Through Channel, the analysis results are sent to other ecosystem service providers that require the variables (i.e., Vire in Section 6.1). Nimbus uses a queuing mechanism that guarantees to process all messages that arrive through the Connector. The records are queued in Kafka and consumed sequentially by the application processing topology built in Apache Storm (see footnote[1]).

The Nimbus applications outlined in Fig. 7 are summarized as follows:

1. *Connector* The ecosystem Connector is responsible for syncing user records with the remote database.
2. *Kafka Message Broker* Apache Kaka (see footnote[1]) is a distributed streaming platform. Nimbus uses Kafka as a message broker between its many micro-applications. The message brokers are set-up to seat across the internal applications and deliver notifications to registered consumers when new data arrives.
3. *Processing Topology* Nimbus uses Apache Storm[8] for its distributed computing. A Storm cluster with multi-worker nodes is set-up for high availability. The processing topologies used for computing are built in various languages and submitted to the cluster using Storm multi-lang protocol. The entry point for the topologies is a Kafka Spout that consumes data from the separate Kafka topics and tuples the output through the different analytic stages. The output of the analytic processes are the user variable scores and measurements. The topology produces the output to a designated Kafka topic for further consumption, and a service database stores all the processed results.
4. *Notify Service* A B2B Interface is implemented in Javascript. The service subscribes to the output topic in Kafka and receives notification when a new processed result is available. The B2B Interface extracts the relevant variables and posts them to the ecosystem Channel endpoint. After that, the ecosystem distribution services

[4] See Table 1 for reference.
[5] A DDP method is like a regular function in Javascript, which is located on the server side and can be called remotely via the DDP connection.
[6] See Table 1 for reference.
[7] See Table 1 for reference.
[8] See Table 1 for reference.

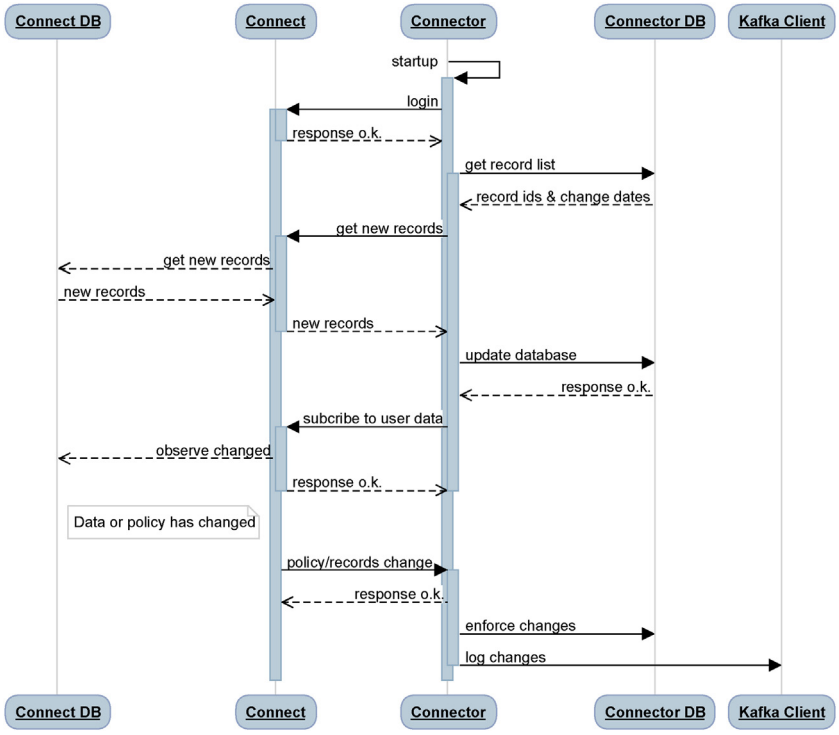


Fig. 5. The Connector sequence diagram showing the interaction with Connect, databases and Kafka client.

Fig. 6. Vire smartphone application (left: history view, middle: day summary, right: settings).

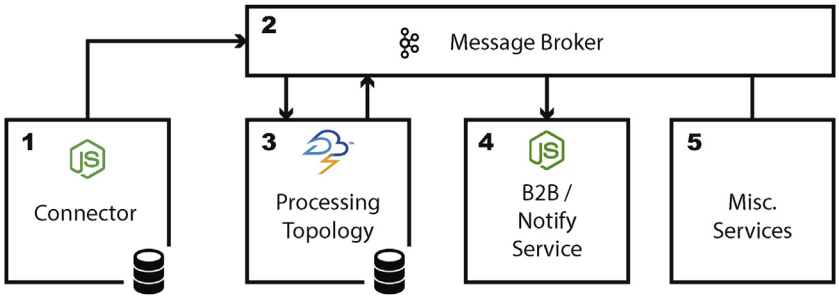


Fig. 7. Nimbus application service and subsystems.

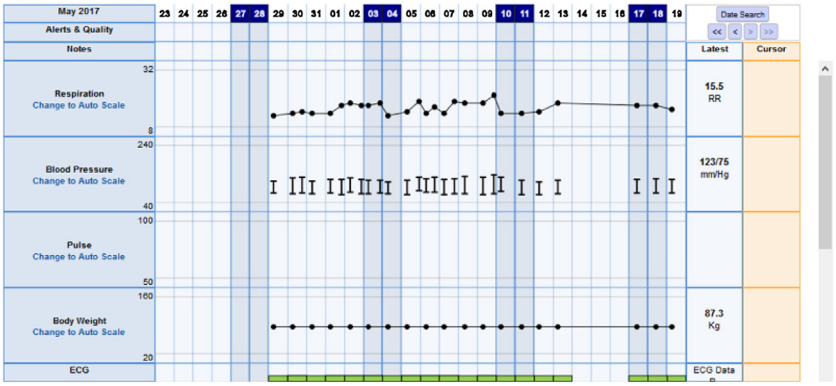


Fig. 8. Data view of the Docobo Clinician's Portal.

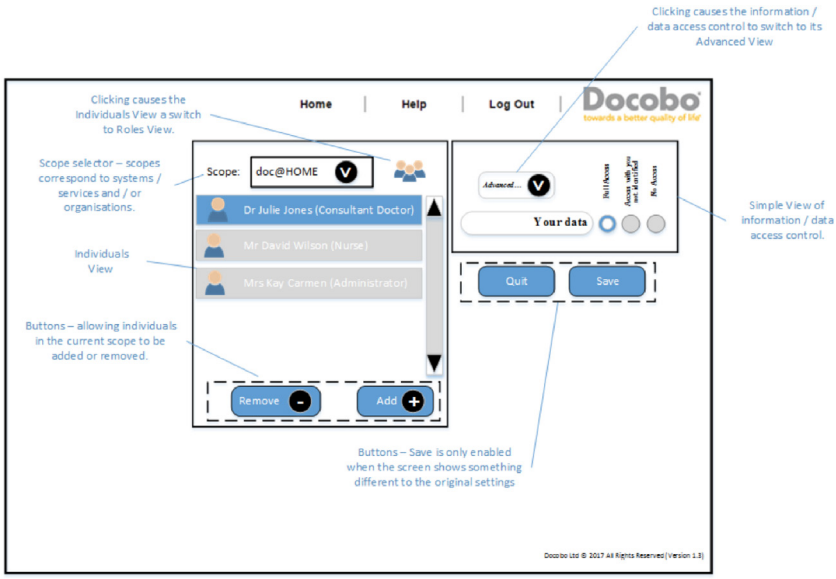


Fig. 9. Simplified access control widget.

immediately sends the new data to the organizations that subscribed.

6.3. Clinical application (Docobo Clinician's Portal)

This application provides clinicians with their patients' information in a comprehensible form (see Fig. 8). The Clinician's Portal is part of Docobo Digital Health and Telemedical Solutions,[9] a medically certified platform. Medical information on weight, ECG, blood pressure, respiration, symptoms, etc. is obtained using the Docobo CAREPORTAL®. Integrating the ecosystem Connector with the portal was trivial. A dedicated virtual machine (VM) was set-up to run the Connector. The VM was protected from external access, and only the database port is accessible within the secure local network of the clinical platform. The Connector populated the database accordingly. The clinician's web-view then sources the patient's information directly from the synchronized database.

The implementation of access control by the patient has provided two challenges.

1. *Mapping the detailed permissions into the existing provision for permissions.* This was particularly challenging in a clinical enterprise system, which may have thousands of patients in a single instance. In this, summary status information for large number of patients had to go through additional levels of filtering to provide the additional access controls. This was found to unacceptably impact the responsiveness of the care portal. To address this, existing permissions and data types were mapped such that the existing setup provided the required filtering and therefore had minimal impact upon performance in the clinical user interface. Then, where patients requested access controls that were different to the ones provided by default, additional filtering information has been added to the permissions mapping in the clinical system's database such that this can be accessed efficiently for filtering when for example listing patients.
2. *Complexity of the access control widget.* With a heuristic evaluation, we considered the original design of the access control user interface widget to be complicated for users who lack experience or confidence with Information Technology. The reviewed version is shown in Fig. 9, aimed to give a simpler view. The left pane shows the option to choose the person, group or organization to apply the privacy settings to (e.g., consultant doctor, researcher, etc.). The access rights can be set on the right pane. The dimension of the data type was removed on the right pane, and all data types are presented as a rolled-up category "Your data". The grant type can either be "full access", "anonymous access", or "no access". The option to switch to an advanced view allowed the user to choose the detailed

[9] See Table 1 for reference.

data types and categories. This was considered to be more acceptable, although we still find it essential to conduct an extensive user evaluation.

7. Results and discussion

New developments in IoT systems benefits personalized healthcare services. For example, it offers the possibility to extend regular spot-check of vital signs to continuous monitoring at home and at the same time provide the data for near-real-time analysis and intervention. User behavior can also be measured continuously. The service opportunities sometimes require various types of user's—personal, behavioral, clinical, or contextual—information to be present. Therefore, there is a need for data aggregation and distribution platform that can service these personalized health applications. However, the presence of such platforms must be privacy-preservative and scalable. Although the proposed Do CHANGE platform was developed for research with cardiac patients to help them in managing their health conditions, it can be used by different research teams working with personal data from mashup services or a custom data source.

Do CHANGE data management platform is a multi-service platform designed for secure data collection and distribution, at the same time, for privacy reasons, it gives users control of their data. The platform has two parts; the data aggregation section gives the users the ability to grant the ecosystem access to their data using OAuth protocol. The secure access tokens derived from the grant process and the collected user data are then securely managed by the aggregation services. The platform consolidates user data within a dedicated environment to reduce security risks. The platform aims to distribute the user data according to a common trust and permissions model, conceptually. This is in view of the EU General Data Protection Regulation [7] to empower citizens with full ownership of their personal data.

Also, the platform helps to remove the complexity of setting up authentication and authorization for multiple APIs; increases the flexibility for supporting new vendors or experimenting with new devices or services registered within the ecosystem. Our experience in building the Do CHANGE platform (adopting a multi-service approach) and its use-cases have revealed some benefits and challenges:

- The independent development and testing of the individual services with little interactions between them was beneficial for managing the workflow and the application sizes.
- The ability to monitor and scale-up the services according to their resource usage (i.e., by increasing the number of nodes) was trivial. This was made easy using the NodeJs process manager (PM2)[10] in cluster mode.
- The advantage that error events and software updates for a given service can only suspend its associated services. Once back online, the disrupted service resumes its normal operations making it easy to manage the entire ecosystem. This combined with a cluster implementation of each service can reduce the downtime to zero.
- From a security perspective, the vulnerability of one service does not entirely compromise the remaining system. This was considered by design to help improve the platform's security tolerance.
- The DDP technology used for distributing data to remote sites was a bottleneck for scalability. Reaching 200+ users with connected Moves, Fitbit, Beddit, and custom devices, the Connect service peaked in memory, running to over 1GB in RAM usage while the other services used less than 50MBs. This was due to the in-memory method used by DDP for tracking changes to the distributed dataset. As a temporary solution, we scaled-up the number of nodes used by the Connect process.
- Sending data to the remote sites securely for extended services was trivial to implement and manage. The Connector took care of adding, updating and deleting remote records. Although, due to the high RAM usage of the DDP, the underlying WebSocket connection occasionally dropped-out.
- Mapping the detailed permissions into the existing provision for permissions was particularly challenging in a clinical enterprise system. The large numbers of patients using the Docobo Clinical Portal had to go through additional levels of filtering to provide the additional access controls.
- Presenting a detailed user resource management interface can be cumbersome. To better understand the design requirements for this, additional user-study should be conducted.

8. Options for future work

The following enumerates the opportunities for near future research:

- *Development of a Federated Resource Manager* The resource manager will give users the ability to control access to their data that resides within the ecosystem. An appropriate user study can be undertaken in understanding the requirements for the system.
- *Changing the MEM-hungry DDP mechanism* DDP has shown to require high RAM usage. GraphQL[11] may be explored as an alternative solution to improve the scalability of the distribution services.
- *Use of a distributed-personal repository* The platform stores user data centrally in a secure way. In improving data security and trust, a personal data store is desired in which personal information is persisted in secure physical memory. The storage can be centrally managed or kept in possession of the user.

9. Conclusion

A system for data aggregation and distribution is developed. The system has shown to be operational and expandable with substantial considerations for data security and privacy. Several use cases were presented for use in clinical, analytics, and mobile applications. An essential feature for the end-users to control access to their data was conceptualized and implemented on a clinical enterprise platform. However, a federated user resource manager based on the proposed concept is yet to be fully developed and tested.

Author's contribution

I.A. and M.W. conceived and designed the data aggregation and distribution services presented. M.W. with the support of I.A. performed the coding for Consume, Collect, Connect and Channel services presented. I.A. with the support of M.W. performed the coding for the Connector service. Nimbus was partly implemented and managed by I.A. and Synergy-Vire was partly implemented and managed by M.W. Docobo Clinician's Portal was developed and managed by Docobo Ltd and I.A. helped in setting up the Connector for the Portal.

I.A. drafted the manuscript and M.W. provided Fig. 4.

I.A., M.W., P.P., S.B., L.F. revised the work and reviewed subsequent versions of the manuscript.

Conflict of interest

The authors reports grants from European Union's Horizon 2020 research and innovation program under grant agreement no 643735 and the Taiwanese Ministry of Economics, DOIT, during the conduct of the study.

I.A. holds company shares at Onmi B.V., who is a consortium

[10] See Table 1 for reference.

[11] See Table 1 for reference.

member of the Horizon 2020 European Do CHANGE project.

No further conflict of interest known.

Acknowledgements

This work is supported by Do Cardiac Health: Advanced New Generation Ecosystem (Do CHANGE) project,[12] funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no 643735 and the Taiwanese Ministry of Economics, DOIT.

References

[1] M. Swan, The quantified self: fundamental disruption in big data science and biological discovery, Big Data 1 (2) (2013) 85–99, http://dx.doi.org/10.1089/big.2012.0002.

[2] D.F. Tate, R.R. Wing, R.A. Winett, Using internet technology to deliver a behavioral weight loss program, JAMA 285 (9) (2001) 1172–1177, http://dx.doi.org/10.1001/jama.285.9.1172.

[3] A.P. Jones, Indoor air quality and health, Atmos. Environ. 33 (28) (1999) 4535–4564, http://dx.doi.org/10.1016/S1352-2310(99)00272-1.

[4] M.E. Morris, Q. Kathawala, T.K. Leen, E.E. Gorenstein, F. Guilak, M. Labhard, W. Deleeuw, Mobile therapy: case study evaluations of a cell phone application for emotional self-awareness, J. Med. Internet Res. 12 (2) (2010), http://dx.doi.org/10.2196/jmir.1371.

[5] P.C. Hallal, L.B. Andersen, F.C. Bull, R. Guthold, W. Haskell, U. Ekelund, L.P.A.S.W. Group, et al., Global physical activity levels: surveillance progress, pitfalls, and prospects, Lancet 380 (9838) (2012) 247–257, http://dx.doi.org/10.1016/S0140-6736(12)60646-1.

[6] R.H. Weber, Internet of things-new security and privacy challenges, Comput. Law Secur. Rev. 26 (1) (2010) 23–30, http://dx.doi.org/10.1016/j.clsr.2009.11.008.

[7] Regulation (EU) 2016/679 of the European parliament and of the council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation), Off. J. Eur. Union (L119) (2016) 1–88 http://data.europa.eu/eli/reg/2016/679/oj.

[8] D. Bender, K. Sartipi, Hl7 FHIR: an agile and restful approach to healthcare information exchange, 2013 IEEE 26th International Symposium on IEEE Computer-Based Medical Systems (CBMS) (2013) 326–331, http://dx.doi.org/10.1109/CBMS.2013.6627810.

[9] A.S. Miller, B.R. Payne, Health it Security: An Examination of Modern Challenges in Maintaining Hipaa and Hitech Compliance. http://digitalcommons.kennesaw.edu/ccerp/2016/Academic/8.

[10] J.Q. Chen, A. Benusa, Hipaa security compliance challenges: the case for small healthcare providers, Int. J. Healthc. Manage. 10 (2) (2017) 135–146, http://dx.doi.org/10.1080/20479700.2016.1270875.

[11] D. Hardt, The OAuth 2.0 Authorization Framework. https://tools.ietf.org/html/rfc6749.

[12] M. Saaranen, J. Parak, H. Honko, T. Aaltonen, I. Korhonen, W2e – wellness warehouse engine for semantic interoperability of consumer health data, IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI) (2014) 350–354, http://dx.doi.org/10.1109/BHI.2014.6864375.

[13] C.-C. Huang, J.-L. Huang, C.-L. Tsai, G.-Z. Wu, C.-M. Chen, W.-C. Lee, Energy-efficient and cost-effective web API invocations with transfer size reduction for mobile mashup applications, Wireless Netw. 20 (3) (2014) 361–378, http://dx.doi.org/10.1007/s11276-013-0608-7.

[14] H.Y. Lin, J.L. Huang, A web API aggregation service for mobile mashup applications, 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (2014) 73–76, http://dx.doi.org/10.1109/IIH-MSP.2014.25.

[15] E. Shernan, H. Carter, D. Tian, P. Traynor, K. Butler, More guidelines than rules: CSRF vulnerabilities from noncompliant OAuth 2.0 implementations, in: M. Almgren, V. Gulisano, F. Maggi (Eds.), Detection of Intrusions and Malware, and Vulnerability Assessment, Springer International Publishing, Cham, 2015, pp. 239–260.

[16] W. Jansen, T. Grance, Sp 800-144. guidelines on security and privacy in public cloud computing, Tech. rep., Gaithersburg, MD, United States, (2011) https://dl.acm.org/citation.cfm?id=2206222.

[17] M. Jones, J. Bradley, N. Sakimura, Json web token (jwt), Tech. rep. (2015) https://www.rfc-editor.org/rfc/rfc7519.txt.

[18] A. Pfitzmann, M. Hansen, A Terminology for Talking About Privacy by Data Minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management. http://www.maroki.de/pub/dphistory/2010_Anon_Terminology_v0.34.pdf.

[19] M. Wetzels, I. Ayoola, S. Bogers, P. Peters, W. Chen, L. Feijs, Consume: a privacy-preserving authorization and authentication service for connecting with health and wellbeing APIS, Pervasive Mobile Comput. 43 (2017) 20–26, http://dx.doi.org/10.1016/j.pmcj.2017.11.002.

[20] S. Tilkov, S. Vinoski, Node. js: using javascript to build high-performance network programs, IEEE Internet Comput. 14 (6) (2010) 80–83, http://dx.doi.org/10.1109/MIC.2010.145.

[21] D. Flanagan, JavaScript: The Definitive Guide, O'Reilly Media, Inc., 2006.

[22] K. Tollmar, F. Bentley, C. Viedma, Mobile health mashups: making sense of multiple streams of wellbeing and contextual data for presentation on a mobile device, 2012 6th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth), IEEE (2012) 65–72, http://dx.doi.org/10.4108/icst.pervasivehealth.2012.248698.

[23] B. Fletcher, K.J. Pine, Flex: Do Something Different, University of Hertfordshire Press, 2011.

[12] See Table 1 for reference.