Framework for Disease Outbreak Notification Systems with an Optimized Federation Layer

Farag Azzedin[1], Mustafa Ghaleb[2], Salahadin Adam Mohammed[3], Jaweed Yazdani[4]
Information and Computer Science Department,
King Fahd University of Petroleum and Minerals
Dhahran, Saudi Arabia

***Abstract*—Data that is needed to detect outbreaks of known and unknown diseases is often gathered from sources that are scattered in many geographical locations. Often these scattered data exist in a wide variety of formats, structures, and models. The collection, pre-processing, and analysis of these data to detect potential disease outbreaks is very challenging, time-consuming and error-prone. To fight disease outbreaks, healthcare practitioners, epidemiologists and researchers need to access the scattered data in a secure and timely manner. They also require a uniform and logical framework or methodology to access the relevant data. In this paper, authors propose a federated framework for Disease Outbreak Notification Systems (DONSFed). Using advanced design and an XML technique patented in the US in 2016 by our team, the framework was tested and validated as part of this work. The proposed approach enables healthcare professionals to quickly and uniformly access data that is required to detect potential disease outbreaks. This research focuses on implementing a cloud-based prototype as a proof-of-concept to demonstrate the functionalities and to verify the concept of the proposed framework.**

***Keywords*—Disease outbreak notification system; database federation; web services; service oriented architecture; health systems**

I. Introduction

The world population growth is causing disease outbreaks to occur frequently and the advancement in transportation technology is making them spread quicker and farther. As a result, fighting modern disease outbreaks demands minimum response time from relevant healthcare professionals. One way to minimize the response time of healthcare professionals is to build an efficient disease outbreak notification system (DONS). Building an efficient DONS has many challenges and has attracted many researchers [1], [2], [3], [4], [5]. Some of the main challenges are:

- DONS data often reside in data-sources located across many geographical, jurisdictional and organizational boundaries. Beside technical obstacles, collecting data from such diverse data-sources poses other defiances.
- DONS data can be huge [6]. Processing such volume of data on time can be challenging.
- DONS data often exist in a wide variety of formats, structures, data models, and data types. Pre-processing such variety of data can be time-consuming.
- Collecting data from heterogeneous data-sources is a complex operation. Some of these data-sources are databases while others can be as simple as web-pages. These heterogeneous data-sources often require multiple interfaces, languages, and protocols.
- Arrival of the required data on time from the data-sources may not be guaranteed.
- Integrating, processing, and presenting the collected data in a beneficial way to healthcare professionals is challenging. [7], [8].

To tackle the above-mentioned difficulties, researchers proposed the following two approaches. In the first approach, researchers proposed programs that enable each data-source to share and integrate data with other data-sources. This approach requires each pair of data-source to have a separate integration program, which makes adding a new data-source very costly. It can't simultaneously and seamlessly integrate data from multiple data-sources [9]. In the second approach, researchers proposed federated databases. However, this approach has a number of limitations [7], [8], [10], [11], [2], [1]. First, adding a new data-source to the federation is costly and modifying any of the services offered by the federated database is time-consuming. In addition, this approach is slow in identifying potential disease outbreaks and requires local to global schema translation to resolve the data model heterogeneity among various data-sources. Furthermore, this approach's data-sources are limited to relational databases and need to know the local schema of each data-source. Knowing the local schema of each data-source may not be provided by some data-sources for security reasons.

Motivated by the above-mentioned challenges and limitations, this article proposes a framework called *Federated System for Disease Outbreak Notification Systems* (DONSFed) which is based on federated databases and web services technology. DONSFed is a federation of many data-sources. It is robust and scalable, and it doesn't intervene with the local operation of any of its data-sources. It only asks the data-source for data specific to potential disease outbreaks. It offers its data-sources the required security and autonomy. Unlike the traditional federated databases, its data-sources are not limited to relational databases. It can include other types of data-sources such as Triplestore, XML, and NoSQL databases and others. DONSFed is data-store transparent. When a user enters a query, DONSFed breaks it into sub-queries and submits each sub-query to the relevant data-source. It then collects the result of each sub-query, aggregates them and delivers them to the user.

The rest of the article is organized as follows. Section II

presents a summary of data integration techniques while Section III discusses in details the proposed framework. Section IV highlights our conclusions and envisions our directions for future work.

II. Data Integration Taxonomy

Data integration techniques can be classified into five categories as shown in Figure 1. The first technique is the link integration [6], [12]. In this technique, the search begins from the first resource via hyperlinks to get related information. However, the drawbacks of this technique are instability of hyperlinks, ambiguities, and the vulnerability of naming conflicts [6].

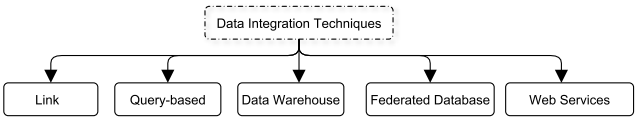

Fig. 1. Data Integration Classification

The second technique is query-based integration [13]. Even though it allows the user to query and retrieve data from different sources by a single query, the query is complex and it lacks the transparency of data location and integration to users.

In the data warehouse integration technique [14], [15], the system queries and retrieves data from different sources to a unified and central repository. The advantages of this technique are improving the performance and increasing data consistency. On the other hand, the disadvantages of this method include keeping an up-to-date central repository, supporting scalability, and maintaining privacy.

The federated database integration provides a uniform and central access to query and retrieve data [13]. This technique is more scalable and flexible than previous techniques [16] since there is no need for a centralized repository. Hence, data replication is not required, and this leads to enhance data privacy and scalability support. This technique is utilized by many bioinformatics systems such as Entrez [17], BioMart [18] and EuPathDB [16].

Web service integration provides extensibility and flexibility features for data integration. Nowadays, this technique is used by many Bioinformatics databases [13], [19], [20]. For example, the National Center for Biotechnology Information (NCBI) [21], European Bioinformatics Institute (EBI) [22], DNA Data Bank of Japan (DDJB) [23], BioMOBY [24], and PathPort [25] use web services techniques to collect and integrate data from their data-sources.

In summary, the federated database and web services techniques are prominent due to their advantages including minimizing the interference of existing operations, managing heterogeneity, preserving local autonomy of constituent systems and supporting scalability. Combining these techniques could be the key to ensure the advantages of both. This research combines federated database and web services integration techniques to build a DONS framework to connect different data-sources together internally and introduce unified access to the data offered by these data-sources.

III. DONSFed Framework

In this section, a framework for DONS is presented that consists of a federation of databases supported by web services. Our proposed framework, DONSFed, includes federation services and component web services. Using an advanced design and an XML node-labeling technique [11] patented in the US in 2016 by our team, the framework was tested and validated as part of this work. The framework allows the use of a portal to query databases in real-time. Such a query is usually split into pieces and then sent across to the target component systems through web services. The query is then processed to retrieve the required data and results are aggregated and returned to the requesting entity. The administrators of the federated services system are empowered to design and implement the required federation services. The component systems' administrators ensure their systems are connected and available. The component systems must maintain high availability because the federated system mainly relies on it for responding to user queries. An abstraction layer, to hide the major differences among the participating systems, is necessary to make the access consistent across the entire framework.

Thus, the DONSFed design consists of the following core elements: the framework layers, the framework workflow, and the environment setup. We have reviewed various approaches that ensure web services integration and offer substantial abstraction among the specific component systems that constitute the federation. Based on the detailed study and analysis of these approaches, we identified and categorized the web services and the required operations for each identified service in our framework. Each web service consists of its description and specifies the necessary input parameters that are needed to invoke its operations. A dedicated web service is available with every component that supports the connection to the portal. Moreover, many advanced features to support changes to the web service operations have been implemented in order to reduce the maintenance required.

A. Framework Architecture

III Fig. 2 presents the DONSFed framework architecture which consists of five layers namely: DONS Federation, Adaptation, Component Systems, Query Processing, and Interface. In the DONS federation layer, the federated services connect to different database systems that participate in the federation. The DONS federation layer consists of several federated services with each service responsible for processing predefined requests upon demand. A query triggers the corresponding federated service which may initiate selection of the available web services in the component systems layer.

The adaptation layer maintains an updated directory of web services available from each component database. It supports non-canonical databases, which do not provide web services natively. This is accomplished by generating web services in a compatible format. In addition, the adaptation layer takes care of the communication between the federation layer and the component systems.

The component systems layer supports heterogeneous data sources. These data sources may have native support for web services. If not, non-canonical data sources will work with

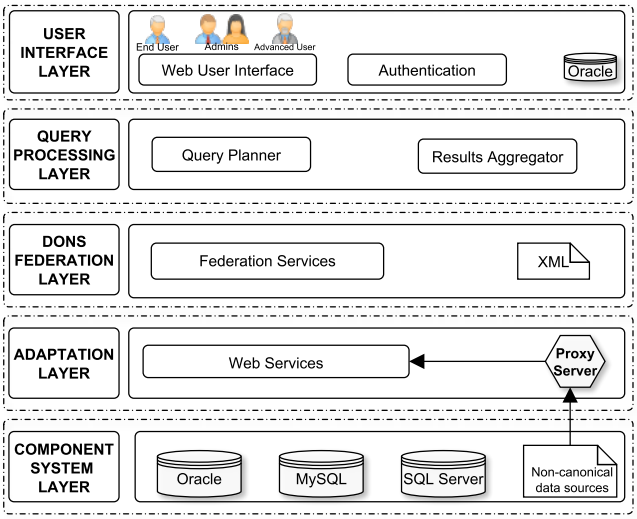

Fig. 2. DONSFed Framework Architecture

a proxy server to generate the required web services in the compatible format. Thus, the component systems layer delivers the required data from the data sources to answer a particular query or sub-query.

The requested data is retrieved from various data sources in XML formats and sent to the results aggregator module which aggregates them into a global result in a suitable format to be delivered to the requesting application or user. To process XML data in XML data sources and to efficiently integrate and process XML data that is generated by the component databases, we developed XML data labeling scheme called Dynamic XDAS. Nearly all the existing node labeling schemes are not updated friendly. We chose to use Dynamic XDAS because it is fast, dynamic, and requires less storage space. It is fast because it computes parent-child, ancestor-decedent, and sibling relationships between XML data using logical operators. It is dynamic because, unlike nearly all the existing schemes, relabeling of XML data is not required during updates. For example, in the popular Dewey node labeling system, insertion of a new sibling node between its siblings labeled n and n+1 is impossible. In the worst case, the whole XML data in the corresponding data source must be relabeled. In Dynamic XDAS that is not required. Any node can be inserted without relabeling any other node. For example and as shown in Fig. 3, The sub-tree labeled $1,011.01$ (colored red) was inserted between the nodes labeled $1,011.01.01$ and $1,011.011$ without relabeling any existing node.

The federated services are described using the Web Service Definition Language (WSDL). The user queries are maintained in a natural language format as questions, with the provision that allows users to choose those questions. Users identify the disease or the category of the reported cases that they need to search and also provide the parameter values related to the selected question. The query planner module transforms the question into sub-queries. Web services are maintained in the Representational State Transfer (RESTful) design. The DONS federation service is passed the web URL of the required web service with the necessary parameters to properly route the

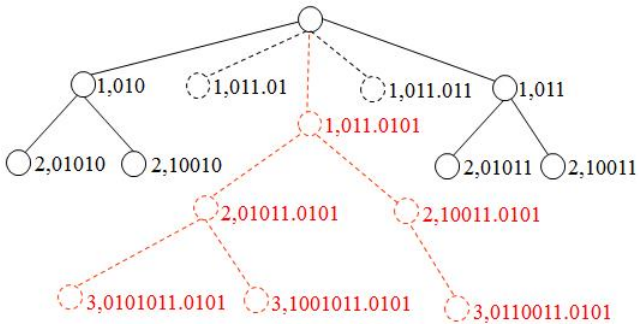

Fig. 3. An Example of Insertion Process in Dynamic XDAS

query to the component system.

The invocation of a RESTful web service with the required parameters determines which component system should be included. The participating component systems return data in XML format and the DONS federation service parses the XML data to combine the results into a single XML document using Dynamic XDAS result as an array of strings. The result is returned to the user in any format that he requires. a tabular format with respective columns for each request to ensure a semantically meaningful result.

The user interface layer provides an interface for authentication service to login to the portal. The authentication service is not only used to verify the user but also grant authorization to all required federation services. The resources across the network can be accessed based on the identified role of end-users during authentication which includes roles such as applications, administrators, advanced-users or end-users. The portal is designed to allow users or applications to select from several categories that contain a set of questions. Users can use the predefined question templates to select their queries to the system and provide the needed parameters. The query service will process and decompose the user query into a set of sub-queries. The results are then delivered to the component systems through the DONS federation layer using the appropriate web service.

B. Framework Workflow

In this section, the workflow that is initiated by a user through the submission of a query into DONSFed is presented. The term workflow, by our definition here, is a set of steps that outline the interactions between a user and the system. The workflow ensures the processing and return of the required results of the user inquiries.

The proposed framework has been designed to return the results of a distributed query in real-time. As mentioned in the previous section, each component system participating on DONSFed has web services which can be used to execute a single or multiple questions (numbered Q_1 to Q_n) and generated using a question template. The portal interface consists of a set of federated services that are designed and deployed by DONSFed administrators. Each federated service is defined as a set of questions that can be selected as workload by either the end user, application or administrator. The selected federated service will list the instructions on how to map the selected queries (Q_i) to various web services to retrieve data through

those web services. The framework is highly flexible and can adapt to demands of new heterogeneous and distributed systems. These systems can join DONSFed by configuring the set of questions and deploying the required web services.

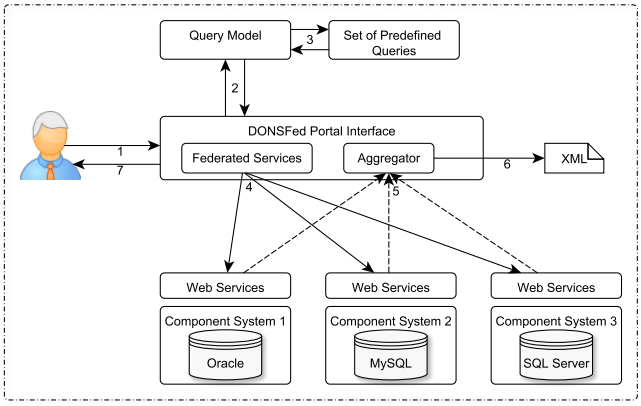

Fig. 4. DONSFed Framework Workflow

Fig. 4 illustrates the workflow approach in practice: 1) the authorized user identifies and selects a specific federation service from an available pool of federation services by accessing the portal; 2) the federation service generates the consolidated question based on the parameters identified by the user; 3) the query module decomposes the consolidated query into sub-queries by mapping each sub-query to one of the questions in the consolidated user-developed question and returns a batch of queries to the federation service; 4) the federated service then invokes a set of different web services of each component system linked to the sub-query; 5) each web service will generate the results and deliver it to the aggregator for a consolidated output; 6) the aggregated results are routed to the local server; 7) The results are displayed to the user in a tabular format as an HTML page.

Fig. 5 compares the execution of a request that generates multiple sub-queries with a straightforward single request. As illustrated, the partitioning, routing and merging of a complex and parallel fetching query using component systems are executed with considerable ease.

The design of the DONSFed framework resolves two major issues that are routinely encountered in a database federation environment. The autonomy provided to the participating systems with adequate provisions for the maintenance of this autonomy is a major challenge for architectures such as ours. The DONSFed services mitigate this issue by applying a sufficiently strong abstraction layer for the affected operations. Furthermore, the DONSFed design ensures that changes are rare to the services layer which guarantees lower maintenance. In order to maintain autonomy of the participating systems, the DONSFed service does not require control over the connected components.

The second issue is the support for heterogeneous data sources that participate from the component repositories. The web services approach allows an abstraction layer that, in turn, supports structural heterogeneity. Heterogeneity in the data tier is generally considered to be a difficult issue to resolve. However, in the DONSFed framework, it is not a major

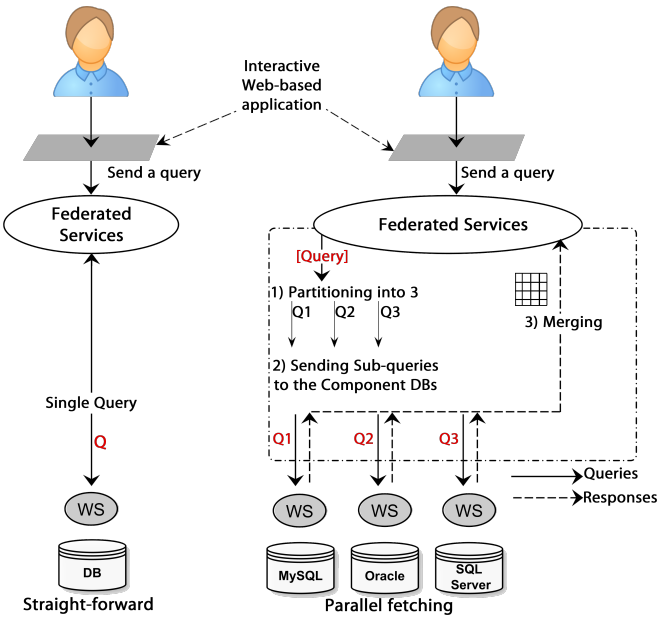

Fig. 5. DONSFed Query Partitioning

problem since the component systems yield mostly similar types of data for diseases, cases and outbreaks. DONSFed addresses the issues of data heterogeneity and data matching thoroughly, thereby, reducing the need for the component systems to modify the data sources. Further, DONSFed encourages the use of similar naming conventions across the network. The optimized federation layer using our patented XML technique [11] and web services makes the DONSFed a highly scalable and efficient framework. The scalability of the proposed framework is supported by the building blocks, a flexible and optimized federation layer with a patented design, RESTful web services and enforcement of standards across the network based on best practices. A new component system joining DONSFed needs to design and deploy the required web services that adhere to the framework guidelines. This is followed by necessary actions on part of DONSFed administrators to add the component system to the federation layer.

C. Prototype Deployment Architecture

In this section, the prototype architecture is described in detail with respect to heterogeneous federated databases and web services that are used to validate the proof-of-concept implementation.

The prototype is a cloud-based and geographically spread implementation that spans multiple heterogeneous platforms across three tiers. The first tier is the presentation tier that represents the user interface. Typically, this involves the use of browser-based graphical user interface for smart client interaction. As shown in Fig. 6, the DONSFed browser-based interfaces for data entry, data aggregation, and data integration aid the main stakeholders including primary health centers, experts and healthcare practitioners in operational and decision-making roles. The external databases such as World Health Organization (WHO) databases and others may also

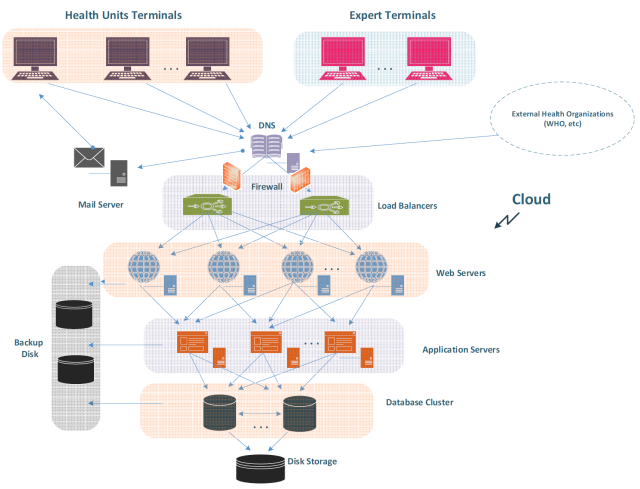

Fig. 6. DONSFed Prototype System Architecture

be connected through this layer for data transmission and retrieval.

The second tier is the application and logic tier where federated services are built to address the functional specifications in terms of federated queries and services based on the stakeholder requirements. Finally, the data tier consists of various heterogeneous database servers. This tier can be accessed through the business services layer and on occasion by the user services layer. Here, information is stored and retrieved and hence this tier keeps data neutral and independent from application servers or business logic while improving scalability and performance.

The different tiers communicate amongst themselves through standard interfaces and protocols. Incoming HTTP requests from users are first sent to the DNS server, where the load balancer routes the requests to web servers with the least load. Web servers directly interact with the appropriate application server to process the requests and receive a proper response. In the implementation, the different component systems were deployed with each one hosted on different virtual machines in a cloud setup using web services middleware in service-oriented architecture design.

Fig. 7 illustrates the high-level view that visualizes the hardware, the middleware and the software used in the prototype implementation as a proof-of-concept deployment. The deployed model consists of multiple tiers including the application and data tier components such as web servers, clients, data sources, and integration links.

D. Prototype Data Tier

In the data tier of the prototype implementation, three autonomous, heterogeneous and distributed databases are connected. These databases were selected based on diverse geographical locations and their database repositories were migrated to our cloud platform. These databases with different schemas and semantics were evaluated as suitable for testing the proposed federation framework. The first database which formed part of the prototype deployment is the KSA DONS system which is an Oracle cloud-based database [26].

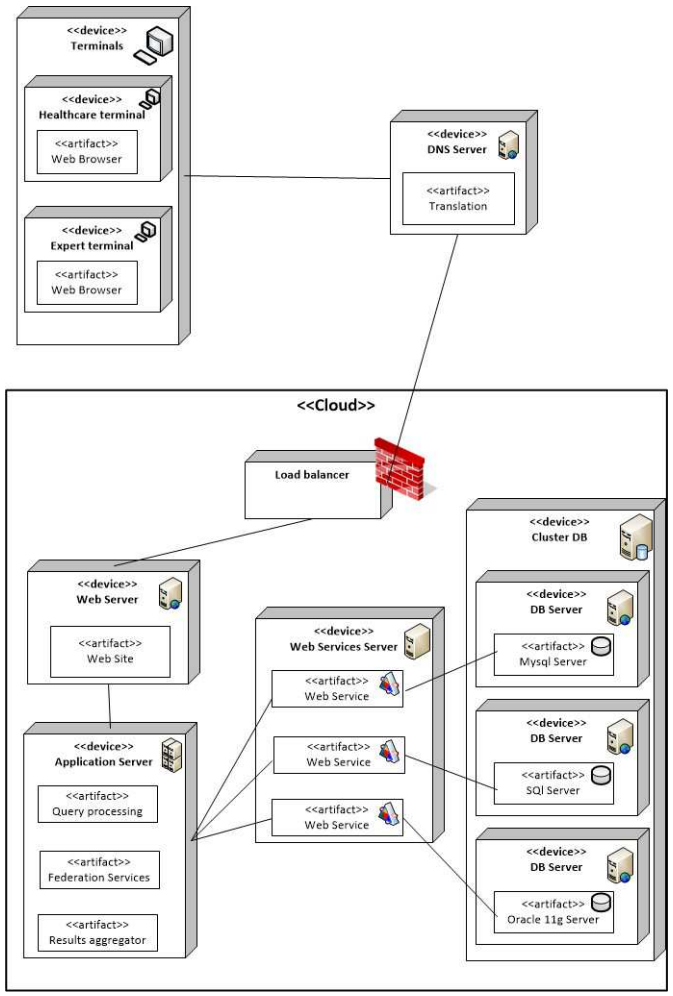

Fig. 7. DONSFed Deployment Model

The KSA DONS database server sits on our university private cloud called KLOUD (KFUPM Cloud) virtual machine with Red Hat Linux 6.4 as its operating system. The Oracle server and client software were configured on all the servers and clients in the KSA DONS architecture. This configuration helped in establishing communication amongst all components of the KSA DONS system including the database server. As shown in Fig. 8, the database schema consists of 19 tables along with stored procedures, triggers, and views.

The second database is a MySQL database from the CASE system in Sweden. This system was developed at the Swedish Institute for Communicable Disease Control (SMI). The system acquires data from the database that collects notifiable diseases in Sweden (SmiNet). The system is currently active and performs daily surveillance. This is an open source software without the personal identification of patients. The available data includes selected variables from the CASE database [4]. The CASE database schema is illustrated in Fig. 9.

In order to further validate our approach that spans a federated database, constituent and actively participation systems, and integration using web services, an additional data source is added. The third database sourced the data again from the CASE database. The entire database was successfully migrated

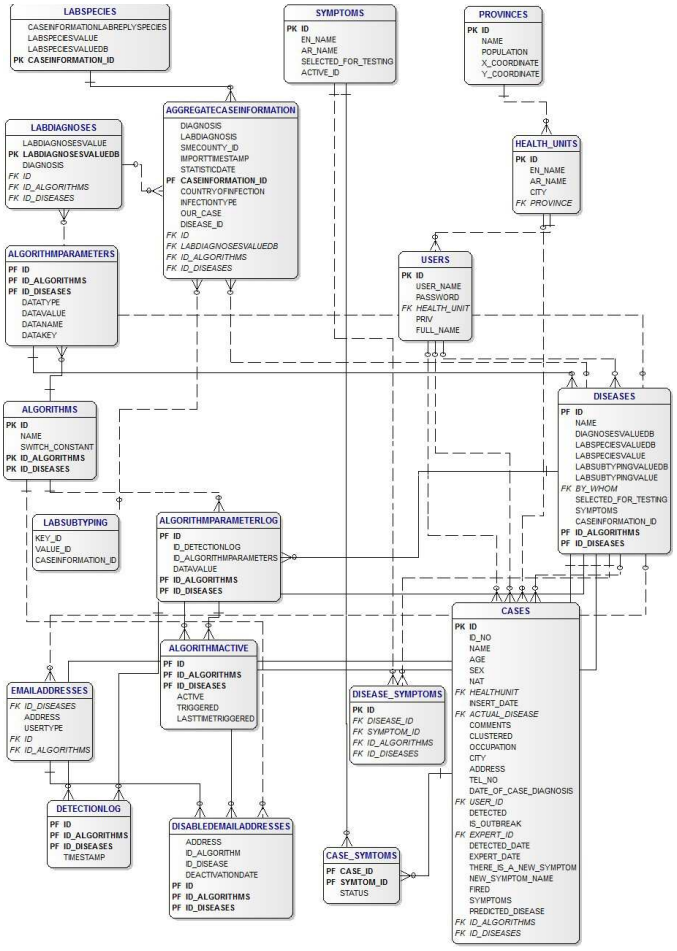

Fig. 8. KSA DONS Database Schema (Oracle)

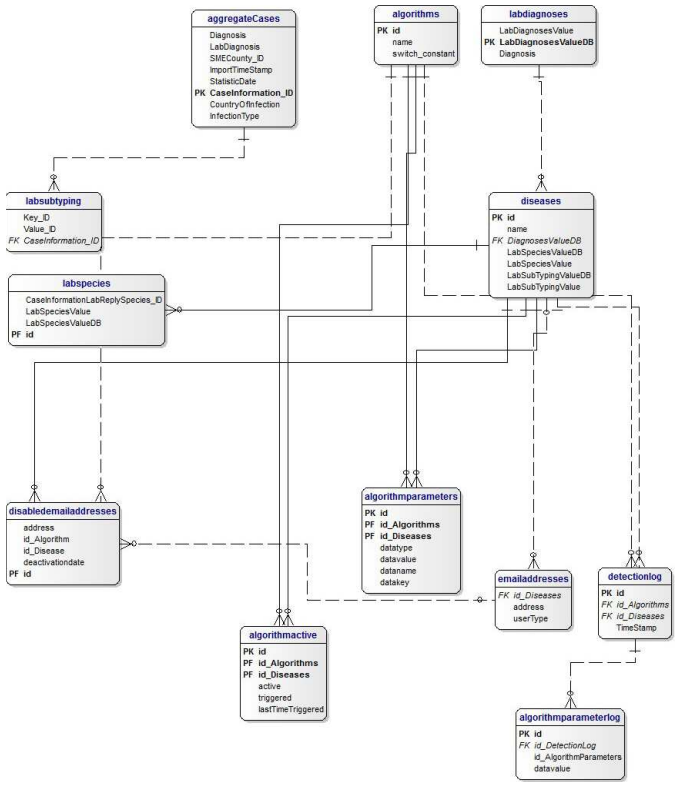

Fig. 9. CASE Database Schema (MySQL)

with all the associated objects including the database schema, stored procedures, triggers etc. The migration to Microsoft SQL server database platform was performed in order to ensure additional heterogeneity to the proposed deployment model. The tools used for migration included SQL Server Migration Assistant (SSMA) utility. The SSMA, which has built-in migration support, aided in the migration of database objects and data from our source MySQL database. The process involved configuring project-level options to convert objects, accurately map source data types to target data types, migrate the data, and ensure all configuration options are compatible with the proposed framework specifications. The migrated database schema consists of 12 base tables and 8 data views with the correctly mapped primary keys and indexes. The stored procedures and triggers were also migrated. The DONS database schema on the SQL server platform is presented in Fig. 10.

E. Prototype Presentation Tier

A cloud-based system with geographically spread component DONS is developed which consists of heterogeneous application and data layers communicating with the DONSFed federation layer. A portal interface is used to allow users to connect to the DONSFed. Typically, the user connects using a browser-based graphical user interface. The DONSFed interface layer is the presentation tier for data entry, aggregation and integration, as shown in Fig. 2, helps the major stakeholders

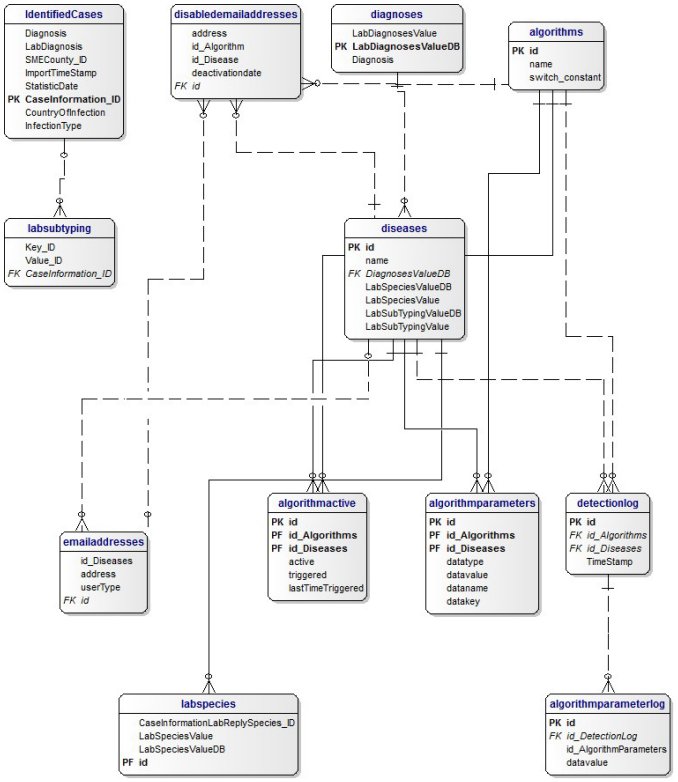

Fig. 10. DONS Database Schema (SQL Server)

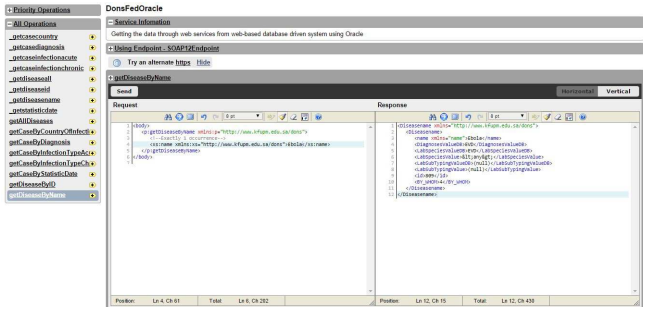

Fig. 11. DONSFed Oracle Data Service

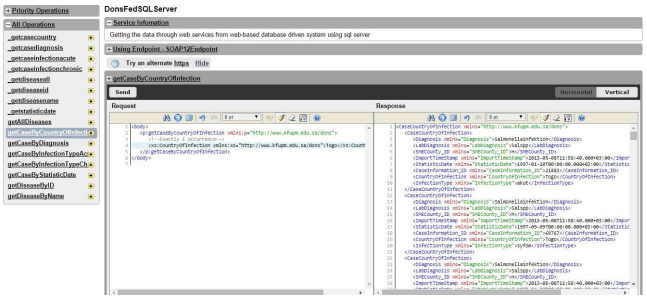

Fig. 12. DONSFed SQLServer Data Service

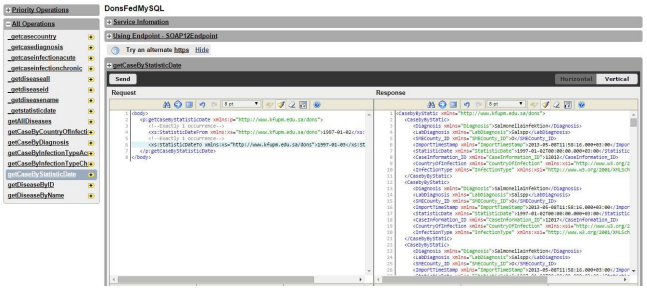

Fig. 13. DONSFed MySQL Data Service

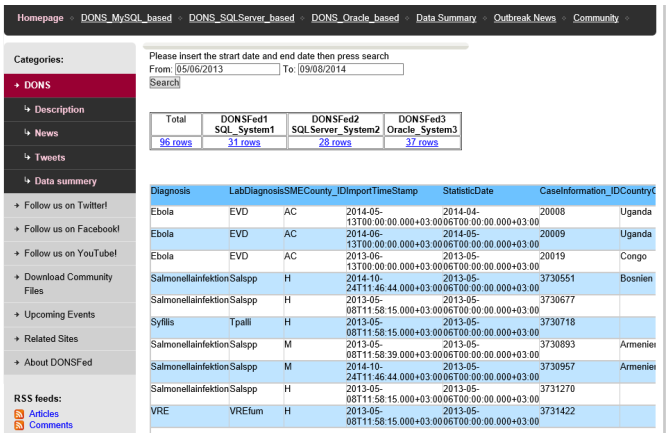

Fig. 14. DONSFed Federated Service

such as primary health centers, healthcare consultants and practitioners to interact with the system. There is provision to connect external databases such as WHO database directly, through an interface in this layer, for data transmission and retrieval. In the application and logic tiers, we maintain the federated services that are designed according to the functional requirements and specifications to support federated queries and services. As mentioned earlier, the data tier consists of heterogeneous database servers participating in the DONSFed. The data tier can be accessed, if needed, through the tier directly using web services. This tier maintains data independent and neutral from application servers or business logic.

As part of the prototype implementation, authors deployed several web services using a data services server that connects to heterogeneous databases through a service-oriented architecture and offers uniform access to autonomous and heterogeneous data sources. Using data masking techniques, the heterogeneity between the data sources, including databases, spreadsheets, or files, is hidden. The web services supported include SOAP and RESTful services. A web service that originates from a DONS federation service and connected to an Oracle database is shown in Fig. 11. The service supports several operations using a WSO2 data services server [1]. This service generates a request in XML format through a request window. After proper parameters are supplied, it will deliver the results in XML format as shown in Fig. 11.

The DONSFed portal offers quick and easy access to users by providing links to specific component databases sites and to the federated services. From the portal page, a user can query to determine which disease is an outbreak. The detection can be queried based on time and location and restricted to registered cases from all component databases. The second web service that originates from a DONS federation service and connected to an SQL server database is shown in Fig. 12. The third web service that originates from a DONS federation service and connected to MySQL database is shown in Fig. 13.

All the results are collected as datasets and formatted into a tabular representation. Fig. 14 shows a federation service that collects the registered cases on all component databases based on a specified date range which is defined as a parameter to that service. The aggregator service module receives and parses the XML output and generates tabular results as shown in the figure. In this particular result, the output presents the number of cases found in each of the participating data sources with the cumulative total.

The HTML output further provides a drill down feature where the user can click on the active hyperlinks to explore the data from each data source. Fig. 14 presents the results of a query as follows: MySQL database produced 31 cases, the SQL server database listed 28 cases, and the Oracle database came up with 37 cases. Several federation services that authors have tested were implemented similar fashion.

IV. Conclusion and Future Work

The proposed approach in the design of a framework has proved successful. The advanced design and patented XML technique ensured that the proposed framework for disease outbreak notification systems is unique. The use of web services for implementing database federation has ensured that the components of the federated system can be added and removed without any impact on the overall federation system

[1] http://wso2.com/products/data-services-server/

while guaranteeing the access, sharing, and retrieval of data from each participating system. The structure of the constituent databases is abstracted using XML. The flexibility introduced through the creation of a federation of databases enables maintaining and supporting autonomous and heterogeneous component systems. The need for local to global schema translation is eliminated through this design. Compliant and non-compliant databases are supported through direct web services or through a proxy setup. The proxy server generates web services in supported formats. Finally, we ensure that the local autonomy of constituent databases is maintained. The proof-of-concept prototype implementation of the proposed framework was successfully deployed. Three different autonomous and distributed databases were used, the KSA DONS system which is an Oracle cloud-based database, the second is a CASE system MySQL database, while the third database is based on Microsoft SQL server. These databases are located at different venues with different schemas and semantics proved suitable for testing our implementation. As part of our future work, authors intend to make the DONSFed framework further compatible for component systems by developing annotations. The federated and constituent systems must concur on the developed ontology to decrease any ambiguity in semantics. These annotations can be used to describe, in a compatible manner, the functionality of each operation, inputs, and outputs of a web service.

ACKNOWLEDGMENT

The authors would like to acknowledge the support provided by the Deanship of Scientific Research at King Fahd University of Petroleum & Minerals (KFUPM). This project is funded by King Abdulaziz City for Science and Technology (KACST) under the National Science, Technology, and Innovation Plan (project number 11-INF1657-04). This work is part of the MSc. Thesis of Ghaleb Mustafa, presented at the Information & Computer Science Department, KFUPM [10].

REFERENCES

[1] T. Millard, S. Dodson, K. McDonald, K. M. Klassen, R. H. Osborne, M. W. Battersby, C. K. Fairley, and J. H. Elliott, "The systematic development of a complex intervention: HealthMap, an online self-management support program for people with HIV," *BMC infectious diseases*, vol. 18, no. 1, p. 615, 2018.

[2] T. Mayo, M. Coletta, S. Crossen, and K. Oliver, "Enhancing Surveillance on the BioSense Platform through Improved Onboarding Processes," *Online Journal of Public Health Informatics*, vol. 10, no. 1, 2018.

[3] J. S. Brownstein, C. C. Freifeld, B. Y. Reis, and K. D. Mandl, "Surveillance Sans Frontieres: Internet-based emerging infectious disease intelligence and the HealthMap project," *PLoS Med*, vol. 5, no. 7, p. 151, 2008.

[4] B. Cakici, K. Hebing, M. Grünewald, P. Saretok, and A. Hulth, "CASE: a framework for computer supported outbreak detection," *BMC medical informatics and decision making*, vol. 10, no. 1, p. 14, 2010.

[5] C. Swaan, A. van den Broek, M. Kretzschmar, and J. H. Richardus, "Timeliness of notification systems for infectious diseases: A systematic literature review," *PloS one*, vol. 13, no. 6, p. e0198845, 2018.

[6] T. Lengauer, *Bioinformatics-From Genomes to Therapies*. Wiley Online Library, 2007.

[7] P. Kumar, "An overviewof architectures and techniques for integrated data systems (IDS) implementation," 2012.

[8] S. Hellmann, J. Lehmann, S. Auer, and M. Brümmer, "Integrating NLP using linked data," in *The Semantic Web–ISWC 2013*. Springer, 2013, pp. 98–113.

[9] B. Zhou, "Data integration as a service for Applications Deployment on the SaaS Platform," in *Biomedical Engineering and Informatics (BMEI), 2013 6th International Conference on*. IEEE, 2013, pp. 672–676.

[10] M. Ghaleb, "Federated Database Framework for Disease Outbreak Information and Notification Systems: A Web Service Approach," Master's thesis, King Fahd University of Petroleum and Minerals (Saudi Arabia), 2014.

[11] T. A. Ghaleb and S. A. Mohammed, "XML node labeling and querying using logical operators," Patent, 2016.

[12] A. Ayton, "Computing for History Undergraduates: A Strategy for Database Integration," *Historical Social Research/Historische Sozialforschung*, vol. 14, no. 4 (52, pp. 46–51, 1989.

[13] J. Wang, Z. Miao, Y. Zhang, and B. Zhou, "Querying heterogeneous relational database using SPARQL," in *Computer and Information Science, 2009. ICIS 2009. Eighth IEEE/ACIS International Conference on*. IEEE, 2009, pp. 475–480.

[14] S. Philippi, "Light-weight integration of molecular biological databases," *Bioinformatics*, vol. 20, no. 1, pp. 51–57, 2004.

[15] C. Schönbach, P. Kowalski-Saunders, and V. Brusic, "Data warehousing in molecular biology," *Briefings in Bioinformatics*, vol. 1, no. 2, pp. 190–198, 2000.

[16] C. Aurrecoechea, A. Barreto, E. Y. Basenko, J. Brestelli, B. P. Brunk, S. Cade, K. Crouch, R. Doherty, D. Falke, S. Fischer, and Others, "EuPathDB: the eukaryotic pathogen genomics database resource," *Nucleic acids research*, vol. 45, no. 1, pp. 581–591, 2016.

[17] D. Maglott, J. Ostell, K. D. Pruitt, and T. Tatusova, "Entrez Gene: gene-centered information at NCBI," *Nucleic acids research*, vol. 39, no. 1, pp. 52–57, 2010.

[18] D. Smedley, S. Haider, S. Durinck, L. Pandini, P. Provero, J. Allen, O. Arnaiz, M. H. Awedh, R. Baldock, G. Barbiera, and Others, "The BioMart community portal: an innovative alternative to large, centralized data repositories," *Nucleic acids research*, vol. 43, no. 1, pp. 589–598, 2015.

[19] "Web Services Based Integration Tool for Heterogeneous Databases," *International Journal of Research in Engineering and Science*, vol. 1, no. 3, pp. 16–26, 2013.

[20] D. Benslimane, M. Barhamgi, F. Cuppens, F. Morvan, B. Defude, and E. Nageba, "PAIRSE: a privacy-preserving service-oriented data integration system," *ACM SIGMOD Record*, vol. 42, no. 3, pp. 42–47, 2013.

[21] N. R. Coordinators, "Database resources of the national center for biotechnology information," *Nucleic acids research*, vol. 45, no. Database issue, p. 12, 2017.

[22] C. E. Cook, M. T. Bergman, G. Cochrane, R. Apweiler, and E. Birney, "The European Bioinformatics Institute in 2017: data coordination and integration," *Nucleic acids research*, vol. 46, no. 1, pp. 21–29, 2017.

[23] S. Miyazawa, "DNA Data Bank of Japan: Present Status and Future Plans," in *Computers and DNA*. Routledge, 2018, pp. 47–61.

[24] B. Consortium and Others, "Interoperability with Moby 1.0—it's better than sharing your toothbrush!." *Briefings in bioinformatics*, vol. 9, no. 3, pp. 220–231, 2008.

[25] B. Yang, T. Xue, J. Zhao, C. Kommidi, J. Soneja, J. Li, R. Will, B. Sharp, R. Kenyon, O. Crasta, and Others, "Bioinformatics Web Services." in *BIOCOMP*. Citeseer, 2006, pp. 258–264.

[26] F. Azzedin, J. Yazdani, and M. Ghaleb, "A Generic MODEL FOR DISEASE OUTBREAK NOTIFICATION SYSTEMS," *International Journal of Computer Science & Information Technology*, vol. 6, no. 4, pp. 137–154, 2014.