Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 6 – No.8, March 2017 – www.caeaccess.org

An Ideal Design and Way Out for Various Service Interoperability and Integration in Open Cloud Environment

L. Rahunathan
Assistant Professor,
Department of Computer Applications,
Kongu Engineering College,
Perundurai, Erode – 638052,
Tamilnadu

A. Tamilarasi, PhD
Professor & Head,
Department of Computer Applications,
Kongu Engineering College,
Perundurai, Erode – 638052,
Tamilnadu

D. Sivabalaselvamani
Assistant Professor,
Department of Computer Applications,
Kongu Engineering College,
Perundurai, Erode – 638052,
Tamilnadu

ABSTRACT
Docker hub is cloud based repository which might include the integration of the activities like code repositories, create images on own and store manually created images into the docker cloud likewise Freight is a fully distributed application environment designed from the ground up to specialize in connecting existing cloud services. End Users will mostly experience Freight through prebuilt solutions that make cloud integration problems vanish. Power Users will mostly experience Freight through the point-and-click Plan Builder, the easiest and most powerful tool for assembling cloud business processes. For developers, though, there's a vast subterranean world to benefit from. Freight is the first cloud integration system that is built from the ground up for fully distributed operation. Each Activity in a Plan can be hosted, operated, upgraded, modified, and secured by a different Terminal running on a different platform in a different physical location and written in a different language. All communication is done through simple HTTP endpoints, and all data is JSON. This creates a system with many advantages.

Keywords
Cloud Computing, Service Oriented Architecture, Interoperability, JSON, and Web Semantics.

1. INTRODUCTION
As the demand of the e-business grows one step higher in day today life, many of the activities are in separable way of communication which leads to accessing the individual activity in delay processing and hence it must have a common loop where it get executes later with certain interoperability. Since the data from various activities are coming in the form of different physical location, with different languages which should have common communication media and integration languages. Here it comes across with certain design and workflow to integrate the tasks as single unit with common interoperability. As the data travelling among these sectors in a loosely coupled environment it requires certain security among the data transaction, considering all the key factors and to get through the solution for the issues raised and approached the concept of Service oriented Architecture with its interoperable mechanism. This paper proposes an interoperable data exchange between different activities in open cloud integration. The three individual activities are document signature, sales force, plan directory which is going to have single unit. So freight provides a solution for three different audiences like developers, power users and consumer end users. For Developer, easy ways to integrate around the technology with the cloud of the business process level. Power users, who want to point and click environment easily assembling application flows, Consumer end users who want to see how the messages are made.

2. RELATED WORK
Information Technology has a significant part in Public service sector, it influence their task of exchanging the information data, transporting from heterogeneous environment with various obstructions. Usha Batraa et al. [1] proposed the interoperability among various health care and the information source from heterogeneous environment which challenges the variety of standards and structures in to a common illustration. Luciana Cardosoet al. [2] proposed a platform AIDA (Archive of Medical Information) and MAS (Multi-Agent System) allow the interoperability and integration through technology like SOA. Reza Rezaei et al. [3]described ERS are operate up on ultra large scale system so the maturity model was proposed which will assed and the maturity level from interoperability can be identified. Carlos Agostinho et al. [4] proposed to create a system that maintains a self-supporting and interoperable frame work that is able to realign information and accommodating the changes of the devices. Secundino et al. [5] suggested a model on how to interact and interoperate between two or more organization which holds the same properties so web service offers the common thematic. C. Coutinho et al. [6] proposed that ESA-CDF are improving the interoperability by adding negotiation mechanism which facilitate the interoperability solutions leads to better results, capabilities and relationships, thus contributing to reduce the risk of losing interoperability. M.B. Doumbouya et al. [7] proposed the levels of interoperability that follows standards for Identifiers, Messaging/Information exchange, structure and content. Diana et al. [8] proposed a model enabling the interoperability between PLM and external engineering products, with respect to coherence concern. Maya et al. [9] proposed a heterogeneous node of Meta model container and it associated with nine tuple node information and description structure, deigning the metadata extension mechanism in order to support common attributes and discrepant information. An architecture for SLA-based service virtualization and hence provides an easy interoperable service execution in a diverse, distributed and heterogeneous world of services [10]. A P/S framework with

the layer that allows us to easily interface the various communication technology and show how to integrate the heterogeneous technologies and enables scenario configuration for every runtime [11]. Petr Novak et al. [12] provide a valuable and efficient integration of heterogeneous data source and component as a knowledge basis to support variable simulation in industries. Fengming Liu et al. [13] proposed a trust evaluation model with an amendatory subjective logic, the model outperforms in terms of both detection capability and stability. The system guarantees automatic collaboration between the services and grant access to processing the resources from the heterogeneous process logic[14]. The method works based on the principal of encapsulation several hierarchies be creating patterns not only for the incorporation of security features but also for modeling the threats and part of its process[15]. Rahunathan et al. [16] proposed the concept of web services with the framework of SOA and performance evaluation process algebra for the composition of services. L.Rahunathan et al. [17] suggested the concept of web services with framework of Service Oriented Architecture (SOA) and data interchange agent that migrates data coming from wide-ranging Emergency Response Service providers into a standard data interchange format and it could be moved in to the data repository

3. OBJECTIVE OF STUDY

3.1 Freight is portable, since it has given access to many peoples to login their hub to fulfill their plans also get publishing the Freight Hub spec and the things are deliberately kept it as simple as possible to make it easy to process Activities and Containers

3.2 Freight is open, Once a Terminal is up and running, the Actions that it exposes can be made public and it can be captivated into obtainable Hubs. This allows the Terminal builder to integrate their own Actions with the publicly available Actions. Put another way, if you expose some of the system's API using a Freight Terminal, you get to automatically tie in everything that can be done on Freight, ranging from sending notifications to carrying out middleware crunching of data, to more.

3.3 Freight is extensible, The Freight Hub has no knowledge about the Freight Terminals it works with, beyond what it discovers when it starts up. That means that individual Terminals and their Activities can be upgraded, patched, and extended independently of the Hubs and client code

4. DESIGN AND WORKFLOW OF FREIGHT

4.1 Hubs and Terminals

Freight Plans are assembled out of individual Activities. The Web Services that host and process Freight Plans are called Fr8 Hubs. The Freight operates a major Hub at an open cloud, but anyone can run their own Hub by creating a web service that supports the hub specification. When a Hub is ready for the next Activity in the Plan to be executed, it makes a call to the Activity's Freight Terminal. Terminals carry out Activity processing. Many Terminals focus on a single web service. For example, in the Plan above, the Activity Get Google Sheet Data will be executed by the Freight Google Terminal, the Activity Send Document Signature Envelope will be executed by the Freight Document Signature Terminal, and the Activity Loop will be executing by the Freight Core Terminal.

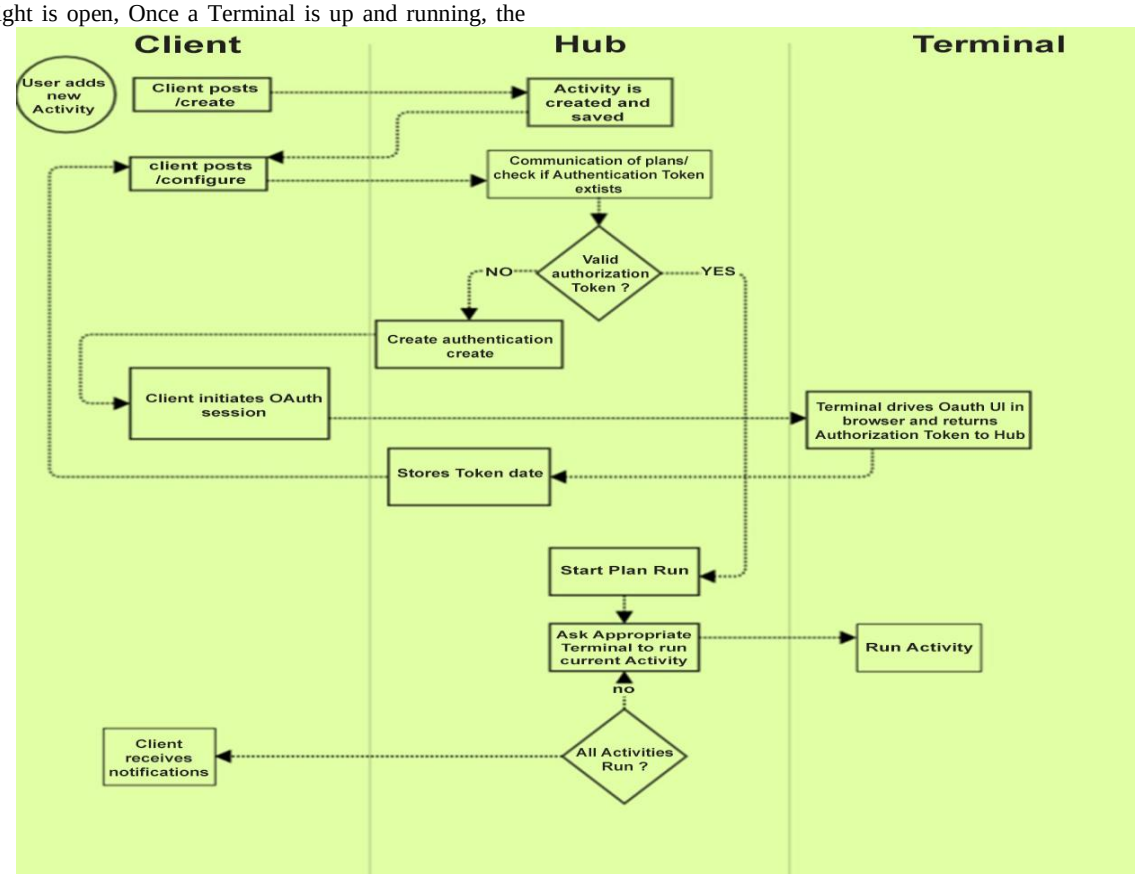

Fig 1: Design and Workflow of Freight Model with swim lane activity

4.2 Crates and Containers: Where Data Lives

Freight does not use a central database. Instead, data lives in Crates, which are stored in Containers. When a Hub executes a Plan, it creates a Container to hold the user's data and routes the Container as it travels from Activity to Activity. The Container is handed off to the next Terminal, which carries out processing, updates the Crates in the Container, and returns the Container to the Hub. All Crate and Container data is represented in JSON. Here's an example of a Crate containing a single DocumentSignatureEnvelope. The Manifest information refers to a shared registry of Crate specifications. This allows Activity A to store data in a Crate using a particular manifest, and know that unknown Activity B, downstream, will be able to look for that Crate and effectively process it

4.3 Crates

Data is transported throughout the Freight network in Crates. A Crate is simply a JSON container for storing data. Crates are currently generated primarily for two specific purposes:

4.3.1 When a user configures a Plan and its Activities in the Plan Builder (Generally call this "Design-Time"), information about the Activities and settings is stored in Crates that are then put into the Crate Storage of a Plan Container.

4.3.2 When a Plan gets run at Run-Time, a Payload Container is created, and as the different Terminals carry out their Activity processing, data is generated, crated, and stored in the Payload Container


```
{
  "manifestType": "Docusign Envelope",
  "manifestId": 15,
  "manufacturer": null,
  "manifestRegistrar": "www.fr8.co/registry",
  "id": "d75f9587-d314-4e20-bd6b-91a521250ce0",
  "label": "DocuSign Envelope",
  "contents": {
    "Status": "Sent",
    "CreateDate": "2016-04-14T15:29:18",
    "SentDate": "2016-04-14T16:11:07",
    "DeliveredDate": null,
    "CompletedDate": null,
    "EnvelopeId": "3237f8f7-06d6-4027-b9f2-23e39d9cb4ae",
    "ExternalAccountId": "docusign_developer@dockyard.company",
    "Name": null,
    "Subject": null,
    "OwnerName": null,
    "SenderName": null,
    "SenderEmail": null,
    "Shared": null,
    "StatusChangedDateTime": null
  },
  "parentCrateId": null,
  "createTime": "",
  "availability": null
},
```


Fig 2: An example of a Crate containing a single Document Signature Envelope

5. STRUCTURE

In the real world, a Crate consists of the stuff inside the Crate (the "Storage"), and some amount of metadata about the Crate (such as receipts, logs, manifests, and directives) that is typically found taped, stapled, nailed, or painted onto the outside of the Crate. Freight Crates are similar. Crate properties can be thought of as the "inside" of the Crate, which is stored in a JSON element called contents, and the "outside" of the Crate, which is generally referred to as the Manifest of the Crate, and which includes a bunch of individual pieces of metadata about the Crate, that provides a

description of the structure of the data in the Crate Storage.

5.1 Crate JSON Definition

A Crate is a JSON element that can contain arbitrary data. It creates a standardized way to store different kinds of data in an organized way

5.1.1 Crates are stored inside of Activities (where they're used to store design-time information used to drive the client user experience) and Containers (where they contain run-time payload, the "real information" that the user wants to manipulate.

5.1.2 Crates are stored in an array inside the crate storage element of these parent elements.

Crate Elements

Name	Type	Notes
Id	GUID	not required and not currently used as part of processing.
Label	string	an arbitrary name useful for distinguishing crates from each other.
Contents	string (JSON)	The actual data inside of the crate. Everything else can be thought of as metadata taped to the outside of the crate. This distinction is important from a security point of view. Generally, properties other than Contents should be assumed to be publicly visible and insecure. On the other hand, by encrypting the Contents of a crate, it can be considered to be data secure.
ParentCrateId	string	Crates can be put inside of other Crates
ManifestType	string	A friendly name for different types of Manifests. Not really necessary, but storing it here makes the json more readable
ManifestId	int	An optional way to signal the schema of the contents.
Manufacturer	ManufacturerDTO	A data structure identifying the Terminal that created the crate
Mode	string	choices "Design-Time", "Run-Time". [Review for need]

Table 1: Represents Properties of Crate Elements

5.2 Signaling of Crate Data

Signaling is a process that takes place in Design mode. The Terminal signals the types of data that an Activity will generate when executed in Run mode, so that the user can configure connections between Activities. For example, suppose a User is building a Plan that will notify him via Slack message every time certain Google Form is submitted. Here it can see how crate signaling is performed by Monitor Google Form activity. After user selects a specific form in Monitor Google Form activity UI, the following set of sequence might happens:

5.2.1 The Hub sends /configure request for Monitor Google Form Activity

5.2.2 Monitor Google Form connects to Google reads the field name of the selected Google form and puts them into a crate of Crate Description and add this crate to activity's crate storage.

It very important to understand that all this process is happening during the Design-Time. No activity has been executed yet. The selected Google Form hasn't been filled and submitted by someone. The only information have at this point is how many fields the selected form has and what are names of these fields.

```
"Crate Storage": {
            "Crates": [
              {
                "id": "a3d8aec5-2f72-4cbd-b83f-13d4163c6c31",
                "label": "Configuration Controls",
                "contents": {
                  "Controls": [
                    {
                      "isReadOnly": true,
                      "name": null,
                      "required": false,
                      "value": "<img height=\"30px\" src=\"/Content/icons/web services/DocuSign-
Logo.png\"><p>You will be asked to select a DocuSign Template.</p><p>Each time a related DocuSign Envelope
is completed, we'll extract the data for you.</p>",
                      "label": "",
                      "type": "Text Area",
                      "selected": false,
                      "events": null,
                      "source": null,
                      "help": null,
                      "error Message": null
                    },
                    {
                      "list Items": [
                        {
                          "key": "Write to Azure Sql Server",
                          "value": "88",
                          "tags": null,
                          "availability": 1,
                          "sourceCrateManifest": {
                            "Type": null,
                            "Id": 0
                          },
                          "sourceCrateLabel": null
                        },
                        { }]
                    }
```


Fig 3: Sample of Crates inside of an Activity's Crate Storage

6. SERVICES AND METHODOLOGY

6.1 Mail Merge into Document Signature

Freight provides a solution for Document signature (i.e.) it extract data from envelopes, which is powerful report generator extends the capabilities of the standard Document Signature reporting tools. Search by Recipient or Template and build powerful queries with a few mouse clicks. Mail merge into Document Signature given a solution to take data from any table-like source which supports basically Microsoft Excel and Google Sheets where it create and send Document Signature Envelopes. A Document Signature Template is used to generate the envelopes, and Freight makes it easy to map data from the sources to the Document Signature Template for automatic insertion. This Activity also highlights the use of the Loop activity, which can process any amount of table data, one row at a time. Track Document Signature Recipients, which Link your important outgoing envelopes to Freights powerful activities, which allow you to receive notification in the form of emails, receive posts or SMS notices, to popular tracking systems. Get notified when recipients take too long to sign

6.2 Mail Merge from sales force

Pull data from a variety of sources, including Excel files, Google Sheets, and databases, and merge the data into your Sales force template. You can link specific fields from your source data to Sales force fields.

6.3 Plan Directory

Plans are created by the Hub in response to client requests. Initially, Plans are inactive. A run call might be generated as the result of a user activity (i.e.) clicking Run in the Plan Dashboard or in response to an external trigger event. When a plan is run, the Hub creates a JSON structure called a Container sometimes Payload Container. The Hub works its way down the Plan, calling each activity's function and passing it the Payload Container. This container is often initially empty, although if the Plan is triggered by an external event, it will usually contain a Crate of data about the event. The ordering is depth-first: children activities are run before sibling activities. Activities are expected to process the incoming data and add to the Payload Container any new data. Depending on the situation, an Activity might want to modify an existing Crate of data or create one or more new Crates. Then the modified Payload Container is passed back to the Hub

7. SECURITY OVERVIEW

Freight access control uses the Sales force architecture and is based on User Profiles, Roles and Permissions. Their primary purpose is to give an easy way to manage access rules for group of users, restrict a set of web pages for certain users, and allow certain action to specific objects. Freight has support for registering accounts as individual users or users that are part of some Organization. System permissions grant access to different objects which apply to the entire Freight environment. Those permissions are grouped in sets which are linked to Profiles. Every user can have only one Profile assigned to him. Freight supports two levels of security principles for accessing its data – Object Based Permissions – Record Based Permissions.

7.1 Object –Type Permissions

These permissions let administrators revoke access for all objects from interaction, or add a possibility for CRUD operations on all group of objects.

7.2 Record-Based Permissions

Determines the ability to grant access to individual object instances, or records. Record based permissions are helpful in the structure of sharing a private object with a group of users, has higher priority from Object Based Security. So the security system first checks to see if an object contains some record-based security.

8. CONCLUSION

This paper has given a better model for the interoperability between the different documents sharing among the different activities with certain execution plan and each plan has a sub plan where the integration of the data is happening, each activity has its own independent location and language which is looped by means of JSON as the communication media. Also SOA architecture helps us in transferring data with different data formats and as a storage aspect the open cloud roots its desired path. Since by making three to four activities in to a single activity will improve the best data sharing component through this freight architecture. The same model can be applied to social networks like whatsApp-Hangouts-Telegram which enables the reliable service between the vendors.

9. REFERENCES

[1] Usha Batraa, Shelly Sachdevab, Saurabh Mukherjeec, "Implementing healthcare interoperability utilizing SOA and data interchange agent": Health Policy and Technology, Volume 4, Issue 3, September 2015, Pages 241-255.

[2] Luciana Cardoso, Fernando Marins, Antonio Abelha, Jose Machado: 'Healthcare Interoperability through Intelligent Agent Technology', International Conference on Health and Social Care Information Systems and Technologies, Procedia Technology 16 (2014), 1334 – 1341.

[3] Reza Rezaei, Thiam Kian Chiew, Sai Peck Lee," An interoperability model for ultra large scale systems", Journal of Advances in Engineering Software, Vol.67, January 2014, pages.22–46.

[4] Carlos Agostinho, Jose Ferreira, and Ricardo Jardim-Goncalves, "Information Realignment in Pursuit of Self-Sustainable Interoperability at the Digital and Sensing Enterprise", IFAC-Papers Online .Vol.48,Issue-3, (2015), pages. 038–045Tavel, P. 2007 Modeling and Simulation Design. AK Peters Ltd.

[5] Secundino Jose Gonzalez Perez, Benjamin Lopez Perez, Daniel Machado Fernandez, José Emilio Labra Gayo, "Interoperability between platforms without a defined referential model: A semi-automatic learning system for structural pairing", Computers in Human Behavior 51 (2015) 1351–1358.

[6] Carlos Coutinhoa, Adina Cretanb, Ricardo Jardim-Goncalvesa, "Sustainable interoperability on space mission feasibility studies", Computers in Industry: Volume 64, Issue 8, October 2013, Pages 925–937.

[7] Mamadou Bilo Doumbouya, Bernard Kamsu-Foguem, Hugues Kenfack, Clovis Foguem, "Telemedicine using mobile telecommunication: Towards syntactic interoperability in teleexpertise": Telematics and Informatics, 31 (2014) 648–659.

[8] Diana Penciuca, Alexandre Durupta, Farouk Belkadib, Benoît Eynarda, Harvey Rowson, "Towards a PLM interoperability for a collaborative design support system", Procedia CIRP 25 (2014), 369 – 376.

[9] Maya Souilah Benabdelhafid, Mahmoud Boufaida, "Toward a better Interoperability of Enterprise Information Systems: A CPNs and Timed CPNs -based Web Service Interoperability Verification in a Choreography", Procedia Technology 16 (2014) , 269 – 278.

[10] A. Kertesz, G. Kecskemeti, I. Brandic, "An interoperable and self-adaptive approach for SLA-based service virtualization in heterogeneous Cloud environments", Future Generation Computer Systems 32 (2014), 54–68.

[11] A.C. Olivieri, Y. Bocchi, G. Rizzo, "Integration in the Internet of Things: A semantic middleware approach to seamless integration of heterogeneous technologies", A volume in Intelligent Data-Centric Systems 2016, Pages 97–128.

[12] Petr Novak, Estefania Serral, Richard Mordinyi, Radek Sindelar, "Integrating heterogeneous engineering knowledge and tools for efficient industrial simulation model support", Advanced Engineering Informatics 29 (2015) 575–590.

[13] Fengming Liu, Li Wang, Lei Gao, Haixia Li, Haifeng Zhao, Sok Khim Mend, "A Web Service trust evaluation model based on small-world networks", Knowledge-Based Systems 57 (2014) 161–167.

[14] Valentin Klimov, "Peculiarities of semantic web-services cloud runtime"; Procedia Computer Science,Volume 71, 2015, Pages 208–214.

[15] Anton V. Uzunov, Eduardo B. Fernandez, Katrina Falkner, "ASE: A comprehensive pattern-driven security methodology for distributed systems", Computer Standards & Interfaces 41 (2015) 112–137.

[16] Rahunathan.L,Sivabalaselvamani.D,Harishankher.A.S,Tamilarasi.A,"A Common Methodical Framework and Dynamic Model of Private Services Composition", International Journal of Advanced Research in Computer Science and Software Engineering,Volume:6,Issue:7,ISSN: 2277 128X,July 2016,Page:251-257.

[17] L.Rahunathan, Dr.A.Tamilarasi, D.Sivabalaselvamani ,"A Smooth System for Applying various Services Interoperability and Data Interchange Agent in SOA", Asian Journal of Research in Social Sciences and Humanities,Volume:6,Issue:9,ISSN:2249-7315,DOI: 10.5958/2249-7315.2016.00797.8,September 2016, page:293-305.