Quest Journals
Journal of Software Engineering and Simulation
Volume 11 ~ Issue 5 (May 2025) pp: 30-34
ISSN(Online) :2321-3795 ISSN (Print):2321-3809
www.questjournals.org

Research Paper

Framework for Data driven EDI Generation and Reading

Sachin Shridhar Padhye
Katy, Texas, U.S

***Abstract*—** *Electronic Data Interchange (EDI) is a technology that enables business to exchange business documents electronically in standardize format. It is electronic data in predefined format that is transferred from one computer application to another over communication link. There are several ways to represent data in any computer system e.g. XML , JSON , Objects and Map, in either of the representation data is tied with a key which tells what underlying data represents. The EDI contains only data and what the data is maintained separately. This make EDI unreadable, it must be integrated with it's specification to make it readable. This is integration framework uses best of both worlds, i.e. specification of an EDI will be integrated with the EDI at run time. This framework is used to generate EDI from data objects like JSON and read EDI to generate data object like JSON at runtime.*

***Keywords*—** *EDI , JSON, CBP*

I. INTRODUCTION

Many computers application system uses Electronic Data Interchange (EDI) to send and receive business documents. Computer applications are developed to collect data from the end user to construct business document. This data comes is saved in various formats e.g. Relational data base is used to save data in tabular format, NOSQL data base is used to save data in JSON document format. In all the cases data is saved with its definition. E.g. Person's information can be saved in table structure as below.

Fig.1– Tabular Representation of Data

PERSON_INFORMATION				
Id	***First Name***	***Last Name***	***Gender***	***Age***
123	John	Doe	Male	23

Similar Data can be represented in JSON format as below.

```
{
 "id":"123",
 "First Name":"John",
 "Last Name":"Doe",
 "Gender":"MALE",
 "AGE":23
    }
```

Fig.2– JSON Representation of Data

Just looking at the record it is easily identifiable that Person's first name is "John", last name is "Doe" , age is 23 , gender is Male and id is 123. This same record can be represented in EDI as

```
PR01JOHN      DOEM12345672323
```

Fig.3– Sample EDI Line

From above EDI it is difficult to read person's information, it is impossible to read first name, last name, age, phone number, gender. To make EDI readable, it needs to be translated into JSON object. Another aspect of the framework is to validate the EDI as per the specifications and expose an API for Create, Read, Update, Delete (CRUD) operations.

This Paper describes the framework developed to generate EDI from user data i.e. from JSON and generate JSON object from EDI. This paper refers supply chain domain for sample EDI. This EDI specifications are published by U.S customs and border protection (CBP) and available for public on internet.

The U.S customs and border protection (CBP) is the nation's largest federal law enforcement agency, incharged with securing the nation's borders and facilitating international travel and trade. One of the responsibilities of CBP is ensuring smooth and effective flow of the legitimate trade, enforcing the laws related to trade. CBP has developed an electronic platform called Automated Commercial Environment (ACE) for all trade processing. One of the channel trade users can use ACE by EDI. [1]. The trade user needs an ability to create EDI from user data to transmit it to CBP and Read EDI received from CBP and convert it in user readable data format.

II. ARCHITECTURE

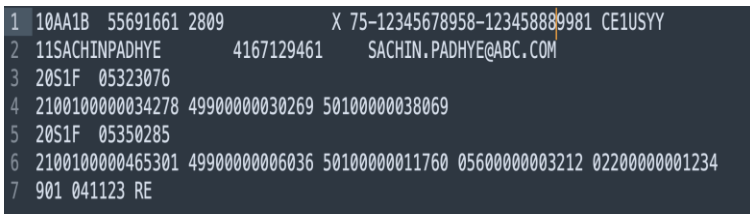
The Architecture of the framework consists of three parts.
1) Defining Sematic of an Edi 2) Implementation and Data flow 3) API.

A. Semantic Of an EDI

One of the properties of the EDI is that EDI just represents series of data, it does not tell what data is, i.e Just looking at the EDI it is difficult tell given data is what. For an example in Fig.3 where is the phone number of the person, what is the name of the person. This specification is maintained separately, i.e in Fig 3, position 5 to position 10 is first name of the person, it should contain only characters, no special characters allowed is maintained in separately.

First component of an architecture is to define Semantic for EDI. Basic properties of an EDI document are as follow. 1) EDI is text document, which contains multiple lines of fixed length data. 2) Each line of the data can is called Record, first few characters of the record define the record name. 3) Documents consist of multiple Records. 4) Records appear in documents in specific order which can be called as a hierarchy of records. This EDI structure can be represented in the JSON format.

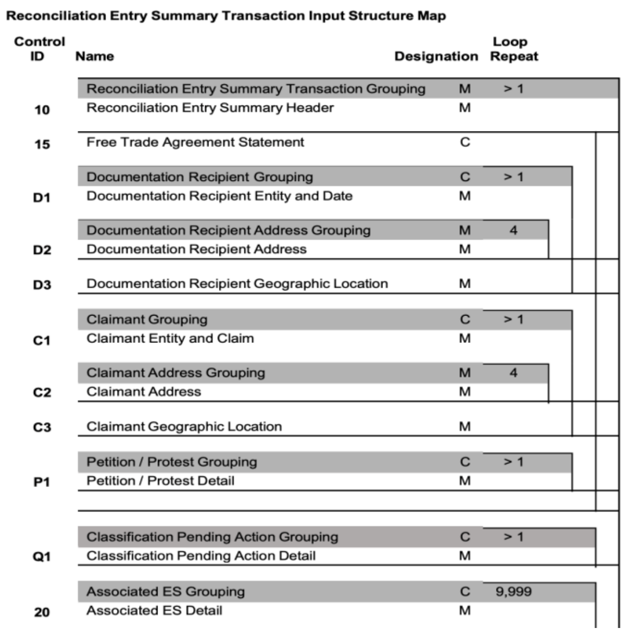
In this paper, we will consider ACE recon entry summary EDI. Below Figure 1 is the sample Recon entry summary EDI.



```
1 10AA1B  55691661 2809           X 75-12345678958-123458889981 CE1USYY
2 11SACHINPADHYE       4167129461     SACHIN.PADHYE@ABC.COM
3 20S1F  05323076
4 2100100000034278 49900000030269 50100000038069
5 20S1F  05350285
6 2100100000465301 49900000006036 50100000011760 05600000003212 02200000001234
7 901 041123 RE
```

Fig.4– Sample EDI document

As shown in above diagram Each line is 80 characters long, each line starts with either 10 , 11 , 20, 21,90. These are record name so line 1 is record 10, line 2 is record 11 , line 3 is record 20 , etc. The records appear in specific order i.e. record 10,11, 20, 21, 90. Also, record 20 appears multiple times, and each 20 record is followed by record 21, this will be called Record hierarchy. Generally, EDI specification defines records hierarchy, below diagram shows Sample Record hierarchy for recon entry summary EDI.



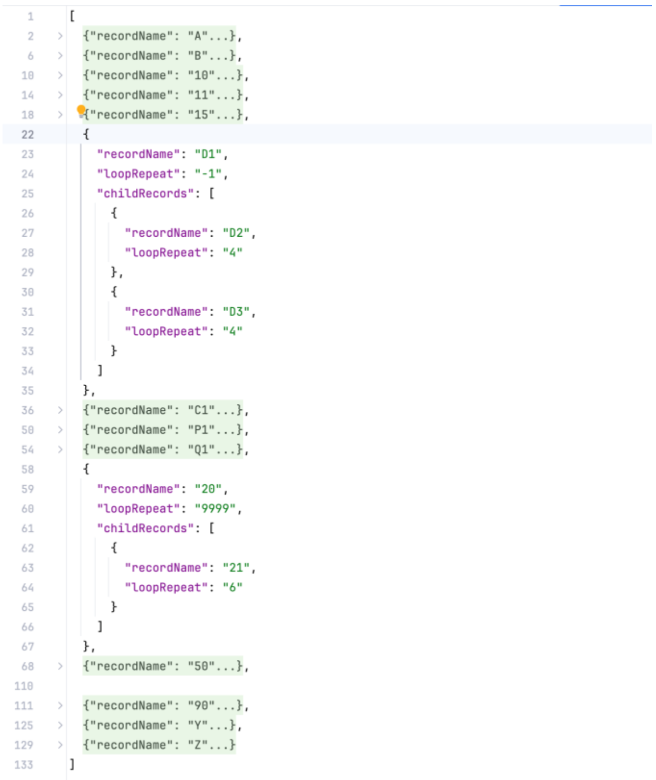
Reconciliation Entry Summary Transaction Input Structure Map

Control ID	Name	Designation	Loop Repeat
	Reconciliation Entry Summary Transaction Grouping	M	> 1
10	Reconciliation Entry Summary Header	M	
15	Free Trade Agreement Statement	C	
	Documentation Recipient Grouping	C	> 1
D1	Documentation Recipient Entity and Date	M	
	Documentation Recipient Address Grouping	M	4
D2	Documentation Recipient Address	M	
D3	Documentation Recipient Geographic Location	M	
	Claimant Grouping	C	> 1
C1	Claimant Entity and Claim	M	
	Claimant Address Grouping	M	4
C2	Claimant Address	M	
C3	Claimant Geographic Location	M	
	Petition / Protest Grouping	C	> 1
P1	Petition / Protest Detail	M	
	Classification Pending Action Grouping	C	> 1
Q1	Classification Pending Action Detail	M	
	Associated ES Grouping	C	9,999
20	Associated ES Detail	M	

Fig.5– EDI Record Hierachy

Above diagram show the hierarchy published by the CBP for Recon entry summary. It shows record 10 will be followed by 15 ,D1 ,D2, D3. D1 record can be appeared more than 1 and each D1 records record can be followed by at most four D2 records and one D3 record. This can be interpreted as D1 record has child records D2 and D3. D2 can appear at most four times and D3 can be appeared 1 time.

This EDI structure can be described in JSON format as below

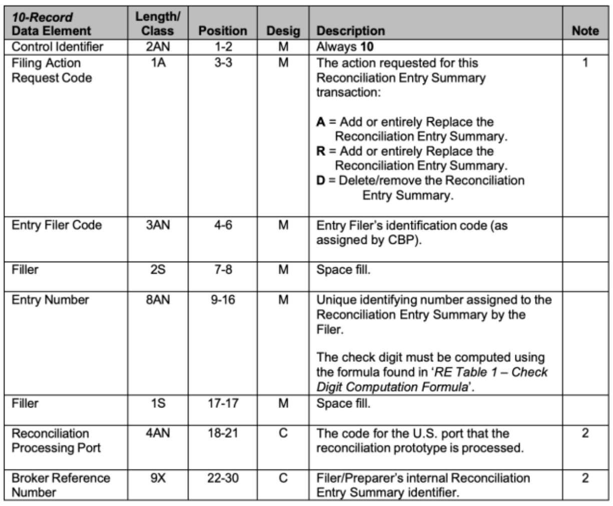


```
[
  {"recordName": "A"...},
  {"recordName": "B"...},
  {"recordName": "10"...},
  {"recordName": "11"...},
  {"recordName": "15"...},
  {
    "recordName": "D1",
    "loopRepeat": "-1",
    "childRecords": [
      {
        "recordName": "D2",
        "loopRepeat": "4"
      },
      {
        "recordName": "D3",
        "loopRepeat": "4"
      }
    ]
  },
  {"recordName": "C1"...},
  {"recordName": "P1"...},
  {"recordName": "Q1"...},
  {
    "recordName": "20",
    "loopRepeat": "9999",
    "childRecords": [
      {
        "recordName": "21",
        "loopRepeat": "6"
      }
    ]
  },
  {"recordName": "50"...},

  {"recordName": "90"...},
  {"recordName": "Y"...},
  {"recordName": "Z"...}
]
```

Fig.6– JSON representation of the EDI document.

EDI specifications describe how each record is structured. Below table is the sample specification published by CBP for 10-record for reconciliation entry summary. It shows first 2characters of the EDI is "Control Identifier" same as Record Name, it is mandatory in EDI line and can contain alphanumeric data.



10-Record **Data Element**	**Length/ Class**	**Position**	**Desig**	**Description**	**Note**
Control Identifier	2AN	1-2	M	Always **10**	
Filing Action Request Code	1A	3-3	M	The action requested for this Reconciliation Entry Summary transaction: **A** = Add or entirely Replace the Reconciliation Entry Summary. **R** = Add or entirely Replace the Reconciliation Entry Summary. **D** = Delete/remove the Reconciliation Entry Summary.	1
Entry Filer Code	3AN	4-6	M	Entry Filer's identification code (as assigned by CBP).	
Filler	2S	7-8	M	Space fill.	
Entry Number	8AN	9-16	M	Unique identifying number assigned to the Reconciliation Entry Summary by the Filer. The check digit must be computed using the formula found in '*RE Table 1 – Check Digit Computation Formula*'.	
Filler	1S	17-17	M	Space fill.	
Reconciliation Processing Port	4AN	18-21	C	The code for the U.S. port that the reconciliation prototype is processed.	2
Broker Reference Number	9X	22-30	C	Filer/Preparer's internal Reconciliation Entry Summary identifier.	2

Fig.7– 10-Record Specifications

Below is the ER diagram to store semantic of the EDI document in relational database. This way EDI specification is separated out from the code and user can manage the EDI configuration without development team dependency. UI can be built to manage different types of the EDI document

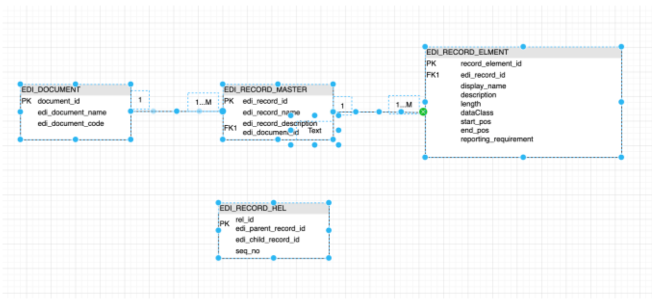


Fig. 8 – Entity Relationship Diagram

B. Implementation and Data Flow

There are two part of the implementation a) EDIGenerator b) EDIParser. Both RecordGenerator and EdiReader uses the EDI definition file in JSON format which describes the EDI semantic.

a. **EDI Generator** - There could be N different ways user data can be represented; hence each user Object requires specific converter which converts the data in JSON format as per EDI specification. Once JSON object is created then it will be parsed and EDI in text format will be generated.

b. **EDI Reader –** EDI Reader converts EDI in textual format to JSON Object as per the EDI semantics. EDI semantics contains start and end position of each data element in record. This is used to fetch data from the record and set it in appropriate JSON object.

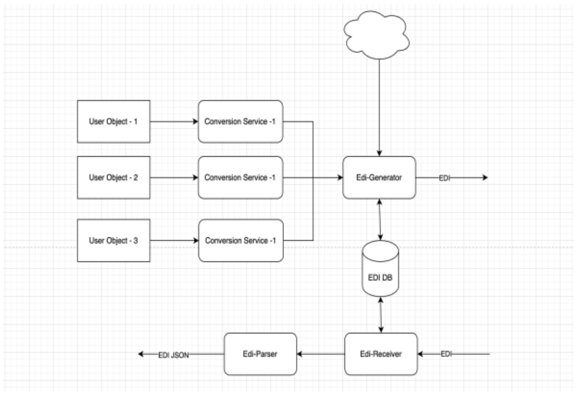


Fig. 9 - EDI Generator and Parser Data flow

C. API

Third component of the architecture is the Create Read Update Delete (CRUD) APIs to consume the services exposed by the framework. Below table contains all the API exposed by the framework.

API Name	Description
create-edi-document	Creates EDI documents in semantic format
Update-edi-document	Updates existing EDI document
Delete-edi-document	Delete EDI document
Get-edi-document	Get EDI document
create-record-element	Create record element in EDI document
update-record-element	Update record element in EDI document
delete-record-element	Delete record element in EDI document
get-record-elements	Get record element in EDI document
generate-edi	Generate EDI from user data
read-edi	Read EDI and convert it to JSON

D. Conclusion and Future work

This paper describes the data driven framework for transforming EDI to JSON object. Aim of the framework is to automate the transaction between EDI to JSON and vice versa and make it data driven. EDI sematic is disintegrated from the application code, so it can be managed independently by subject matter expert without technical dependency. This reduces the frequency of the code update in production. All the changes in the EDI specifications are data driven so updating the specification is trivial and can be managed independently in it's own cycle. EDI specification can be managed though rich user interface, also EDI validation and verification can be implemented by providing mock data transformation.

Currently, EDI specification is maintained manually but using AI / ML it can be generated automatically. Future research can be done on writing AI agents to read EDI and derive EDI specifications. Or Given specification document AI agent can be created to define EDI specification and validation rules.

REFERENCES

[1]. Yinsheng Li et al., "An XML-based data interchange protocol and supporting systems for online customs declaration", 2009 IEEE International Conference on Systems, Man and Cybernetics

[2]. Roland Hellberg, Ragnvald Sannes "Customs clearance and electronic data interchange — A study of Norwegian freight forwarders using EDI", International Journal of Production Economics Volume 24, Issues 1–2, November 1991, Pages 91-101.

[3]. Kavita Choudhary et al., "Electronic Data Interchange: A Review", 2011 Third International Conference on Computational Intelligence, Communication Systems and Networks

[4]. Gerrit K. JANSSENS, "ELECTRONIC DATA INTERCHANGE: FROM ITS BIRTH TO ITS NEW ROLE IN LOGISTICS INFORMATION SYSTEMS" International Journal on Information Technologies and Security (ISSN 1313-8251) 3 (vol. 3), 2011, pp.45-56.

[5]. Francois Bergeron and Louis Raymond, "The Advantages of Electronic Data Interchange" ACM SIGMIS Database: the DATABASE for Advances in Information Systems, Volume 23, Issue 4 Pages 19 - 31

[6]. Dunlu PENG et al. "Using JSON for Data Exchanging in Web Service Applications", Journal of Computational Information Systems 7: 16 (2011) 5883-5890

[7]. Ben Smith, "Beginning JSON" ISBN:9781484202029, 1484202023

[8]. Teng Lv, Ping Yan and Weimin He, "Survey on JSON Data Modelling" ,3rd Annual International Conference on Information System and Artificial Intelligence (ISAI2018) 22–24 June 2018, Suzhou

[9]. Sai Kumar Reddy Thumburu, Innovative Approaches in EDI X12 Retail Transaction Management,Vol. 3 No. 1 (2020): ACS 2020

[10]. ACE Automated Broker Interface (ABI) CBP and Trade Automated Interface Requirements (CATAIR), https://www.cbp.gov/trade/automated/catair