Semantic EDI

Kristina Enes
Fardis Amouzadeh Araei
Negar Batenipour
/ Mentor: Niklas Petersen

Introduction

What is Semantic EDI?

- EDI means Electronic Data Interchange

E.g. the virtual exchange of data or business documents in electronic format between trading partners.

- messages
- ordering
- delivery notification
- ...

- EDI standards: **EDIFACT**,X12,ODETTE,...

Goal of the Project:

Translating EDI messages into “semantic” forms in which the statements encoded in the compact forms are explicitly expressed.

- **ORDERS (Purchase Order Message)**
- INVOICE (Invoice Message)
- ….

Problem

- Comparison of EDI and Paper-based interchange

  Pros: less human interventions
  less human error
  less manual entries ⟶ less labor hours
  better B2B relationship ⟶ mutual cost saving
  self-explanatory order

- Comparison of Semantic EDI and EDI

  Cons: high processing time, sequenced-format
  increased errors
  slow access time for retrieving documents


Order processing without EDI
Buyer generates the purchase order
Buyer sends to the supplier
Supplier receives order
Supplier enters the invoice into their system to
Invoice is created
Invoice is sent to the buyer
Buyer enters invoice into their system


Order processing with EDI
Purchase Order
pacific
commerce
Invoice
Buyers internal system/ERP
Suppliers internal system/ERP

EDI and Paper-based Interchange

Purchase Order

AutoCompany
123 Main Street
Fairview, CA 94168

PO Number: 4768

PO Date: 6/10/2018

Item No.	Quantity	Unit of Measure	Price	Product ID
1	1	EA	1290	331896-42

Total Items: 1

Total Quality: 100

```
UNH+SSDD1+ORDERS:D:03B:UN:EAN008'
BGM+220+4768+9'
DTM+137:20180610:102'
NAD+BY+5412345000176::9++AutoCompany+123Main Street+Fairview+CA+94168+US'
LIN+1+1+331896-42:VN'
QTY+1:1:EA'
PRI+AAA:1290'
UNS+S'
CNT+2:1'
UNT+10+SSDD1'
```

segment

element

Component

Relevance and Importance

Improving the communication between business parties.
e.g. order message

- One company orders the component from different company. Company receives an order, and if it isn't available, produces a component.

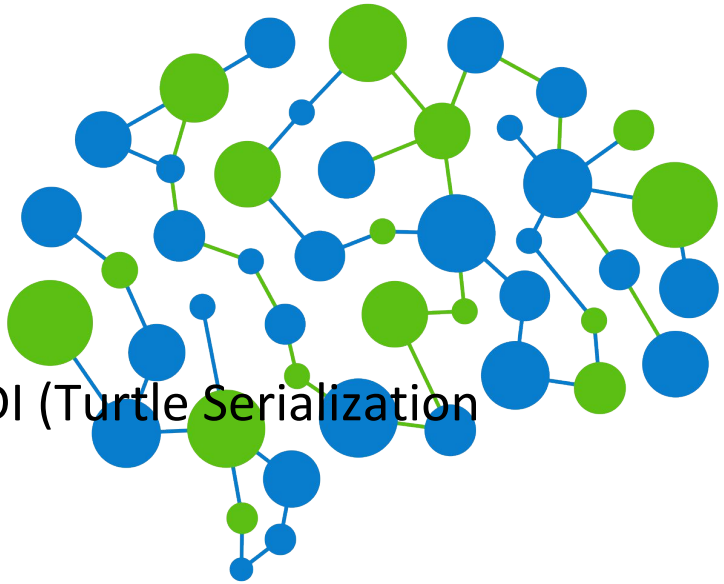
Goal: We can improve this ordering job with Semantic EDI.
Advantages: save time, effort, cost,...

Challenges

- Creating EDI .txt files, .TTL files as inputs

- Designing Translator :
  Convert .EDI ordering messages to .TTL
  ⟶ **Semantic EDI**
- Comparing two different versions of EDI order messages
  -D.03B, D.16B
- Designing Smart Client :
  Use SPARQL and make query on turtle files
  ⟶ **Check the usability of our Translator**

Proposed Solution



Translate EDI order messages to Semantic EDI (Turtle Serialization format):

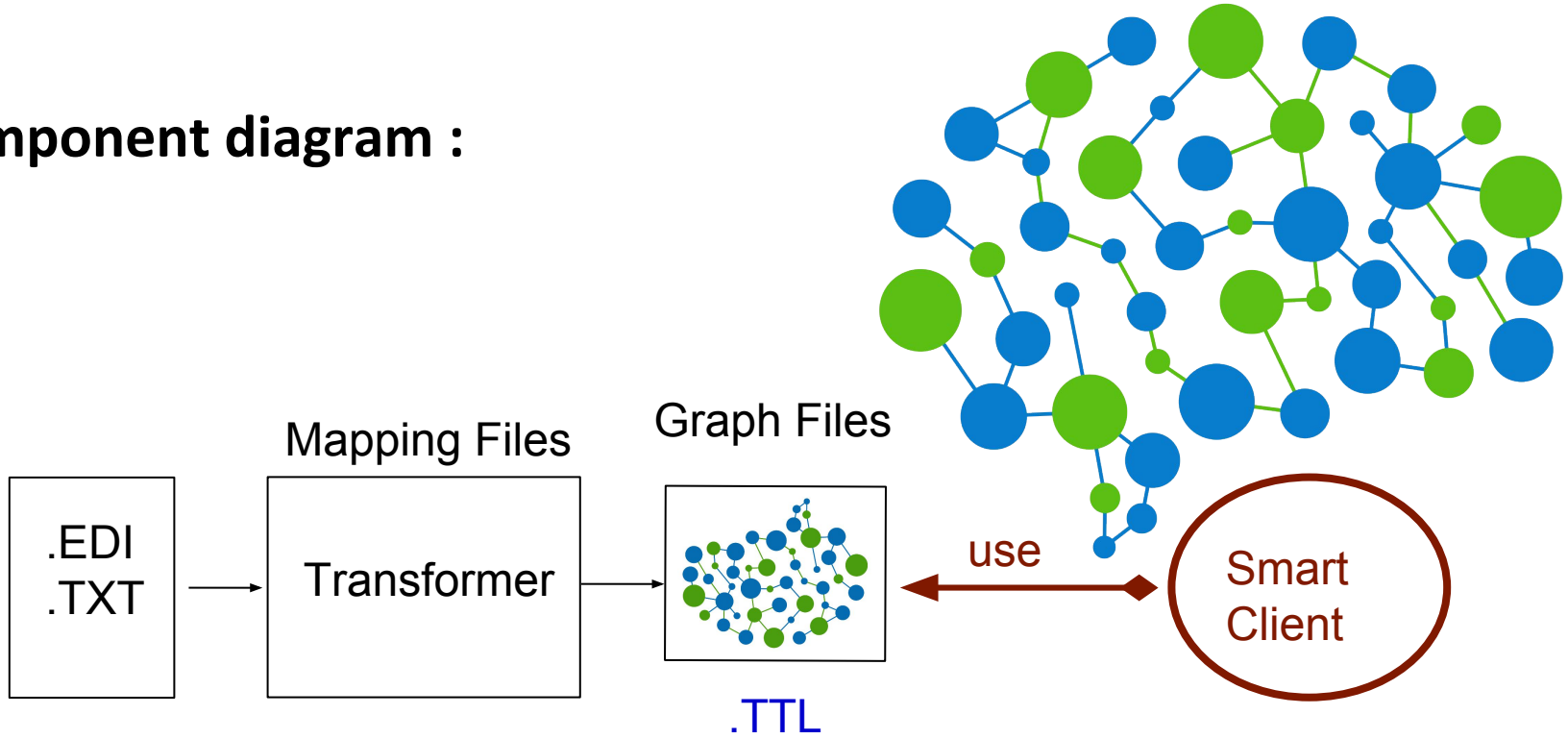
Properties:

-machine-interpretable format,

-decreased processing time

-faster access time

Component diagram :



Implementation

- https://www.truugo.com/edifact/d16b/orders/
- Written .EDI Ordering Message
- Written code in Python.

- Saved outputs in .TTL files, to make use of them in Smart Client.
- Completed our code for two versions of EDI
  -D.03B (2013), D.16B(2016)

Truugo ORDERS Message :

UNH MESSAGE HEADER M 1

A service segment starting and uniquely identifying a message. The message type code for the Purchase order message is ORDERS. Mandatory | Max repeat: 1

0062 MESSAGE REFERENCE NUMBER
Mandatory | Type: an | Length: 0..14
Unique message reference assigned by the sender.

S009 MESSAGE IDENTIFIER
Mandatory
Identification of the type, version etc. of the message being interchanged.

- **0065** Message type identifier
  Mandatory | Type: an | Length: 0..6
  Code identifying a type of message and assigned by its controlling agency.
  Show all standard codes
- **0052** Message type version number
  Mandatory | Type: an | Length: 0..3
  Version number of a message type.
- **0054** Message type release number
  Mandatory | Type: an | Length: 0..3
  Release number within the current message type version number (0052).
- **0051** Controlling agency
  Mandatory | Type: an | Length: 0..2
  Code identifying the agency controlling the specification, maintenance and publication of the message type.
  Show all standard codes
- **0057** Association assigned code
  Conditional | Type: an | Length: 0..6
  Code, assigned by the association responsible for the design and maintenance of the message type concerned, which further identifies the message.

EDIFACT Validation | EDIFACT Subset | EDIFACT to XML | EDIFACT Browser | EDIFACT to CSV | Contact

TRUUGO

EDIFACT Directories / EDIFACT D.16B / Messages

Log in | Sign up

Try demo

D.16B ORDERS

Purchase order message

Version history | Message definition

UNH	MESSAGE HEADER	M 1
BGM	BEGINNING OF MESSAGE	M 1
DTM	DATE/TIME/PERIOD	M 35
PAI	PAYMENT INSTRUCTIONS	C 1
ALI	ADDITIONAL INFORMATION	C 5
IMD	ITEM DESCRIPTION	C 999
FTX	FREE TEXT	C 99
GIR	RELATED IDENTIFICATION NUMBERS	C 10
GRP1	RFF DTM	C 9999
GRP2	NAD LOC FII GRP3 GRP4 GRP5	C 99
GRP6	TAX MOA LOC	C 5
GRP7	CUX PCD DTM	C 5
GRP8	PYT DTM PCD GRP9	C 10
GRP10	TDT GRP11	C 10
GRP12	TOD LOC	C 5
GRP13	PAC MEA GRP14	C 99
GRP15	EQD HAN MEA FTX	C 10
GRP16	SCC FTX RFF GRP17	C 10

Written Code in Python:

EDIOrderingMessage2016-Order1.py ×

Match Case Regex Words

```
1121            if PYTElements >= 6:
1122                g.add((Order1, n.PeriodTypeCode, Literal(PYT[5][0])))
1123            if PYTElements >= 7:
1124                g.add((Order1, n.PeriodCountQuantity, Literal(PYT[6][0])))
1125
1126            #DTM
1127
1128            if DTMElements >= 2:
1129                g.add((Order1, n.DateOrTimeOrPeriodFunctionCodeQualifier, Literal(DTM[1][0])))
1130                if DTMComponents[1] >= 2:
1131                    g.add((Order1, n.DateOrTimeOrPeriodText, Literal(DTM[1][1])))
1132                    if DTMComponents[1] >= 3:
1133                        g.add((Order1, n.DateOrTimeOrPeriodFormatCode, Literal(DTM[1][2])))
1134
1135            #FTX
1136            if FTXElements >= 2:
1137                g.add((Order1, n.TextSubjectCodeQualifier, Literal(FTX[1][0])))
1138            if FTXElements >= 3:
1139                g.add((Order1, n.FreeTextFunctionCode, Literal(FTX[2][0])))
1140            if FTXElements >= 4:
1141                g.add((Order1, n.FreeTextDescriptionCode, Literal(FTX[3][0])))
1142                if FTXComponents[3] >= 2:
1143                    g.add((Order1, n.CodeListIdentificationCode, Literal(FTX[3][1])))
1144                    if FTXComponents[3] >= 3:
1145                        g.add((Order1, n.CodeListResponsibleAgencyCode, Literal(FTX[3][2])))
1146            if FTXElements >= 5:
1147                g.add((Order1, n.FreeText, Literal(FTX[4][0])))
1148                if FTXComponents[4] >= 2:
1149                    g.add((Order1, n.FreeText, Literal(FTX[4][1])))
1150                    if FTXComponents[4] >= 3:
1151                        g.add((Order1, n.FreeText, Literal(FTX[4][2])))
1152                        if FTXComponents[4] >= 4:
1153                            g.add((Order1, n.FreeText, Literal(FTX[4][3])))
1154                            if FTXComponents[4] >= 5:
1155                                g.add((Order1, n.FreeText, Literal(FTX[4][4])))
```

Run EDIOrderingMessage2016-Order1

```
/usr/bin/python2.7 "/home/negar/SDWT Lab/Final Presentation/SmartClient new/FInal Final/EDIOrderingMessage2016-Order1.py"

Process finished with exit code 0
```

Written .EDI Order Message:

```
UNH+SSDD1+ORDERS:D:03B:UN:EAN008'
BGM+220:1:1+4768+9'
DTM+137:20181110:102'
PAI+1:::1:2'
ALI+001+8+3+6+4'
GIR+3+::1'
FTX+AAA+++Free Text One:Free Text Two:Free Text Three'
NAD+BY+5412345000176::9++Auto Company+123 Main Street+Fairview+CA+94168+US'
LIN+1+1+331896-42:VN'
IMD+A+2:1:1+::::Item description one:Item description two'
IMD+F+2:1:1+::::Item description three:Item description four'
RFF+1018:::'
LOC+3160+::5:'
FII+AI'
DOC+220::5:'
CTA+AA+15'
COM+512:AD'
TAX+6+AAE:::'
MOA+256:1290::18:113'
CUX+3::18+++CIE'
PCD+5044::1::5+1'
PYT+72+5::5+9+11+6M'
RJL+FF::5:Object Ordered+1::5'
TDT+34+Airplane+22:Aerial++::5:Lufthansa Airline+SB+++3'
PAC+1+3:4:15+::5:+S::AA+ZZZ:10'
MEA+AAB+AAC:9:19:++AF'
PCI+23+Do Not Drop-Fragile'
GIN+AB+34567-12'
EQD+AK+::5:US+12::5:90*120+1+2+5+17'
HAN+2::5:It Should be in dry environment+::5:'
SCC+12+CD+B:5:F'
APR+AD+23:CSD'
RNG+13'
ALC+H+:6+13+8+AAE::5'
RTE+7:60:30:+1'
RCS+4+3::5:+1+001'
DGS+ADR+66+++2'
EFI+order.txt:Ordering Message+text(.txt):2018:ASCII'
CED+3+::5:Ubuntu:16.04:2016:34567-12'
PIA+5'
GEI+2+6::5:+8'
QVR+:7+AA+AB::5'
MTD+9+1'
CCI+11+AAC:9:7:+34567-12::5::+2'
CAV+3::5'
STS+7::5+1::5:To be Done'
TOD+5+AD+1::5'
STG+1+2+4'
QTY+1:5'
PRI+AAA:1290:Euro:::Euro'
UNS+S'
CNT+2:1'
UNT+10+SSDD1'
```

Output.ttl :

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix ns1: <http://example.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

[] ns1:AccountingEntryTypeNameCode "1" ;
    ns1:AccountingJournalIdentifier "FF" ;
    ns1:AccountingJournalName "Object Ordered" ;
    ns1:ActionCode "1" ;
    ns1:AllowanceOrChargeCodeQualifier "H" ;
    ns1:AllowanceOrChargeIdentificationCode "6" ;
    ns1:AllowanceOrChargeIdentifier "" ;
    ns1:AssociationAssignedCode "EAN008" ;
    ns1:CalculationSequenceCode "8" ;
    ns1:CarrierIdentification "" ;
    ns1:CarrierName "Lufthansa Airline" ;
    ns1:ChangeReasonDescriptionCode "AB" ;
    ns1:CharacteristicDescription "" ;
    ns1:CharacteristicDescriptionCode "34567-12" ;
    ns1:CharacteristicRelevanceCode "2" ;
    ns1:CityName "Fairview" ;
    ns1:ClassTypeCode "11" ;
    ns1:CodeListIdentification "" ;
    ns1:CodeListIdentificationCode "",
        "1" ;
    ns1:CodeListResponsibleAagencyCode "5" ;
    ns1:CodeListResponsibleAgencyCode "",
        "1",
        "2",
        "5",
        "9" ;
```

Differences between versions

- Differ in the number of segments, elements and components
  - D.03B: 49 segments, 194 elements, 364 components
  - D.16B: 52 segments, 214 elements, 412 components
- Differ in element and component names
- Example version 2003 vs 2016:
  - Differ in number of elements and components:
    BGM,TAX,TDT,EQD,DGS
  - Differ in name of element and component:
    LIN,RCS,PCI,RFF,NAD,FII,CTA,COM
  - New added segments:
    EFI,CED,STS

Evaluation

- **Designing Smart Client :**
  make SPARQL query on turtle files
  -output1.ttl,output2.ttl,output3.ttl,data.ttl
  check the usability of Translator

Result : - item Identifier, price, quantity of three products
- availability of products
- production time, readiness of product

Smart Client:

Smart Client.py ×

```
g1.parse("data.ttl", format = "n3")
g1.parse("outputfile1.ttl", format = "n3")

qres1 = g1.query(
    """PREFIX ns1: <http://example.org/>
       PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
       SELECT DISTINCT ?Product ?ItemIdentifier ?Price ?Quantity ?InStock ?ReadyForTransport ?ProductsToProduce ?ProductionTime ?NewProductsReadyIn
       WHERE {
          ?order ns1:ItemIdentifier ?ItemIdentifier.
          ?Product ns1:Identifier ?ItemIdentifier.
          ?order ns1:PriceAmount ?Price.
          ?order ns1:Quantity ?Quantity.
          ?Product ns1:AvailableInStock ?InStock.
          bind(if(xsd:integer(?InStock) >= xsd:integer(?Quantity),"True", "False") as ?ReadyForTransport).
          bind(if(xsd:integer(?InStock) >= xsd:integer(?Quantity),"0", xsd:integer(?Quantity) - xsd:integer(?InStock)) as ?ProductsToProduce).
          ?Product ns1:ProductionProcess ?Process.
          ?Process ns1:ProductionLine ?ProductionLine.
          ?ProductionLine ns1:ProductionTimePerItem ?TimePerItem.
          ?ProductionLine ns1:Machines ?Ms.
          ?Ms ns1:FreeIn ?MachineFreeIn.
          bind(if(xsd:integer(?ProductsToProduce) = 0, "",  xsd:integer(?TimePerItem) * xsd:integer(?ProductsToProduce)) as ?ProductionTime).
          bind(if(xsd:integer(?ProductsToProduce) = 0, "",  xsd:integer(?ProductionTime) + xsd:integer(?MachineFreeIn)) as ?NewProductsReadyIn).
```

Run Smart Client

```
/usr/bin/python2.7 "/home/negar/SDWT Lab/Final Presentation/SmartClient new/FInal Final/Smart Client.py"
Product                 ItemIdentifier   Price   Quantity   AvailableInStock   ReadyForTransport   ProductsToProduce   ProductionTime   NewProductsReadyIn
http://example.org/Product1   331896-42   1290   5          1000               True                0
http://example.org/Product2   331896-43   1000   3          20                 True                0
http://example.org/Product3   331896-44   900    6          0                  False               6                   18               28

Process finished with exit code 0
```

Use Cases

- One company orders a product from different companies.
- The company receives the order and checks its availability:
  - if the product is in stock the order will be processed
  - otherwise the production time should be calculated

E.g. : Volkswagen company orders the component of the car from different companies.

Demonstration

- **GitHub:**
https://github.com/NegarBatenipour/Semantic-EDI/

- **Video Representation**

Lessons Learned

- Importance of communication between business parties
  - -to save time, cost, effort,...
- Convert EDI to Semantic EDI
  - - create decent code for Translator
  - - consider all segments, elements, and components
    - → mandatory, conditional
  - - compare two versions: D.03B, D.16B
- Create EDI order message for both versions

- Design Smart Client
  - - decent SPARQL code
  - - create data.ttl

Conclusion and Future Work

- Translate EDI to Semantic EDI (Translator)
  - D.03B, D.16B
- Design Smart Client
  - Make use of translator

In Future:

- extended Translator ⟶ different versions of EDI order message
- work with different types of messages ⟶ 195 messages in D.16B
- different standards of EDI
  - e.g. X12,...

Final Slide