Bachelor's thesis

Information and Communications Technology

2018

Vili Kangas

CANONICAL DATA MODEL IN PRINCIPAL CONNECTIONS

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information and Communications Technology

2018 | 32

Vili Kangas

CANONICAL DATA MODEL IN PRINCIPAL CONNECTIONS

Business partners exchanging messages in highly globalized world introduces multiple combability issues. Businesses from different countries using different systems rarely use the exact same file formats or even support same transfer protocols. These issues can be overcome with system integration. System integration can be done using different architectures. Canonical data model is one of them and it is used to reduce the amount of needed translations in big integration projects. The purpose of this bachelor's thesis was to examine the benefits of canonical data model.

Canonical data model's benefits were examined first on theoretical basis using existing research data and field's literature. The data model's benefits were also examined in practice by implementing an interface using canonical data model. Interface was created using Axway B2Bi integration product that is a commercial integration product used by the company that commissioned this bachelor's thesis.

From theory's perspective the canonical data model reduced implementation and maintenance costs even in smaller system integration projects. Practical examination supported these findings and the data model's benefits were clear even in the initial stages of the project.

During this bachelor's thesis results showed that canonical data model can successfully reduce the amount of connections needed in a project involving multiple principals and systems with multiple interfaces.

KEYWORDS:

system integration, Axway B2Bi, canonical data model, messaging

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tieto- ja viestintätekniikan koulutus

2018 | 32

Vili Kangas

KANONINEN TIETOMALLI ASIAKASYHTEYKSISSÄ

Yrityskumppaneiden vaihtaessa sanomia vahvasti globalisoituneessa maailmassa tulee helposti vastaan useita yhteensopivuusongelmia. Yritykset eri maista käyttävät eri järjestelmiä ja harvoin käyttävät samoja tiedostoformaatteja, tai edes samoja tiedonsiirtoprotokollia. Nämä ongelmat voidaan ratkaista järjestelmäintegraatiolla. Integraatiota voidaan toteuttaa useilla eri arkkitehtuureilla, joista kanoninen tietomalli on yksi. Sitä käytetään, jotta voidaan pienentää käytettyjen sanomamuunnosten määrää isoissa integraatioprojekteissa. Opinnäytetyön tarkoituksena oli tutkia kanonisen tietomallin käyttöönoton hyötyjä järjestelmäintegraatio projekteissa.

Kanonisen tietomallin hyötyjä tarkasteltiin teoreettiselta pohjalta käyttäen hyväksi aiempia tutkimuksia sekä alan kirjallisuutta. Tietomallin hyötyjä tarkasteltiin myös käytännössä implementoimalla kanonista tietomallia hyödyntävä liittymä hyödyntäen Axwayn B2Bi integraatiotuotetta, joka on toimeksiantajalla käytössä oleva kaupallinen integraatiotuote.

Kanoninen tietomalli pienensi teorian pohjalta tarkastellessa jo pienemmissäkin järjestelmäintegraatio projekteissa ylläpitokustannuksia sekä kehityskustannuksia. Käytännön tarkastelu tuki havaintoja ja hyödyt tulivat esille jo alkuvaiheessa olevassa projektissa.

Opinnäytetyössä saadut tulokset näyttivät, että kanoninen tietomalli voi onnistuneesti vähentää vaadittujen yhteyksien määrää projekteissa, joihin liittyy useita asiakkaita ja järjestelmiä useilla rajapinnoilla.

ASIASANAT:

järjestelmäintegraatio, Axway B2Bi, kanoninen tietomalli, sanomaliikenne

CONTENT

CODES

PICTURES

LIST OF ABBREVIATIONS OR SYMBOLS

CMF	Common Middle Format
HTTP	Hypertext Transfer Protocol
SFTP	Secure File Transfer Protocol
EDIFACT	Electronic Data Interchange for Administration, Commerce and Transport
DESADV	Despatch Advice
EDI	Electronic Data Interchange
B2B	Business to Business
A2A	Application to Application
ERP	Enterprise Resource Planning
CDM	Canonical Data Model
API	Application Programming Interface
UI	User Interface
DMZ	Demilitarized Zone
LAN	Local Area Network

1 INTRODUCTION

File format standards are developed to make communication between different systems easy. However, every standard has many iterations that are not usually backwards compatible and usually different principals prefer different standards over others. Sometimes principals even want to use their own inhouse format that suits perfectly for their own needs. This causes an issue making communication between different systems hard. This can be solved by an integration product that will transform any input format to any output format desired.

In B2B integrations a simple point-to-point integration is the most common way of making an integration between two systems. It is popular way to design an integration because of its simple and straightforward approach. It is fast to implement and doesn't have many moving parts. One alternative to this is to do the integration using canonical data model.

Canonical data model means that to do a transformation from format A to format B you first transform the format A to a common format. This adds one extra step to the integration but in return it makes the transformations already created recyclable reducing the number of transformations needed in long run.

If a transformation for example from EDIFACT format to the common format is made once you can use that transformation whenever you need to transfer EDIFACT to something. After a while when transformations from most standard formats to the common format and transformations from common format to most standard formats are done implementing a new interface would require only using these transformations made earlier.

This thesis is divided into two sections. In first section I describe the tools and products used to implement the integration as well as file formats that are used in the implementation. I explain the key functions of the products involved and what they are capable of from principal connection perspective.

In the second part of this thesis I will implement one B2B integration using canonical data model. The interface I implement is an advanced shipping notice message from principal to the medical supplier. During this part I will go into detail how the products introduced in the first part of the thesis work.

After the implementation I evaluate the benefits of the canonical data model compared to more straightforward point-to-point integrations. I also evaluate the canonical data model's capability to reduce translation maintenance

The company that commissioned this thesis will be referred throughout the thesis as "the medical supplier"

2 CANONICAL DATA MODEL

Canonical data model or CDM is a data model used to represent data in a simple uniform form. It does so utilizing a data format that is generalized to define entities, their attributes and relations. (Technopedia 2018)

Best case scenario for messaging would be if every partner and system used the same data formats when exchanging data. However, this is not the case and with partners from different countries using different systems there is almost always the need to do message translations. Canonical data model is one of the ways to implement these message translations.

2.1 Benefits

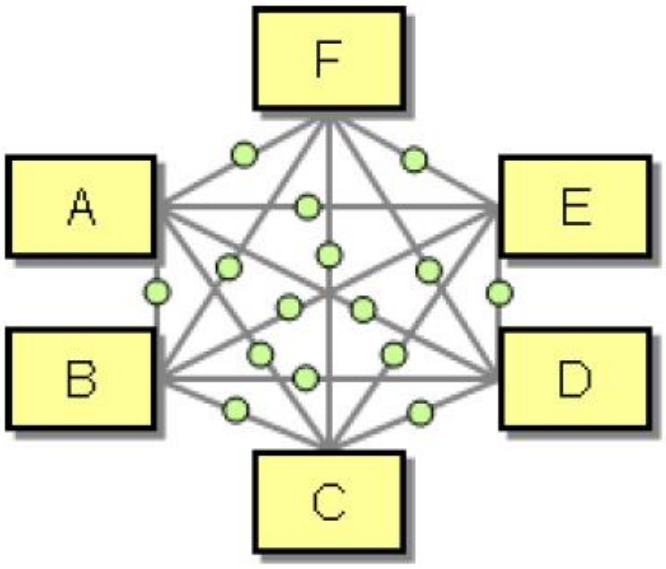
Canonical data model is not beneficial with organizations with small number of integrations. However, with big organizations the canonical data model quickly pays off when many interfaces are introduced to the system. The highest benefit comes from the development costs. After the initial development of one interface it can always be reused when another partner or system is introduced to the project. (Gregor H and Bobby W 2003)

Another benefit from the reusability of canonical data model is that systems can be replaced in the future. For example, if multiple systems are connected to a legacy system the effort to migrate to newer systems will be much less if canonical data model has been implemented. (Gregor H and Bobby W 2003)

There is always the need to evaluate the needs of the organization when considering canonical data model over point to point integrations. Since canonical data model always requires two translations for one message it will create some overhead and latency to the process. If organization prioritized performance over anything else point to point integrations might be the only possibility. This will however reduce maintainability of the integration platform. (Gregor H and Bobby W 2003)

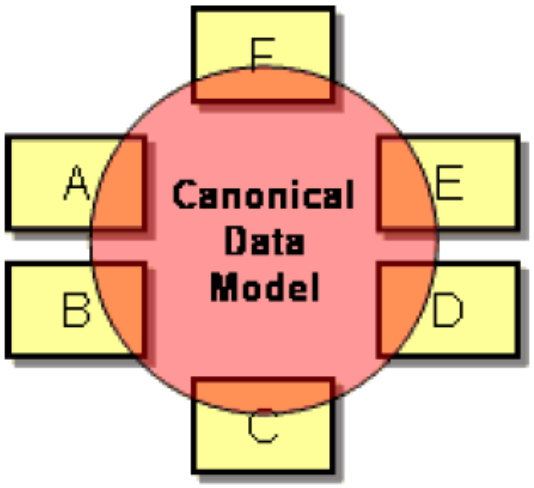
2.2 Examples

Benefits of canonical data model can easily be explained through examples. For small integration projects it is not worth using canonical data model because for example if organization only needs two interfaces connected it would need only two traditional point to point integrations. When using canonical data model in this scenario you would need four message translations. This is illustrated in picture 1. (Gregor H and Bobby W 2003)



Picture 1. Six partners with connections to each other (Gregor H and Bobby W 2003)

To explain the benefits of the canonical data model in bigger projects you only need six different partners or systems that all needs one interface between each other. With traditional integrations there would be a need of 30 different translations in total. However, with canonical data model this number can be reduces to only 12 different translations. For each partner or system one translation to the canonical data model and one translation from canonical data model to partner's own format. This is illustrated in picture 2. (Gregor H and Bobby W 2003)



Picture 2. Six partners with canonical data model (Gregor H and Bobby W 2003)

3 AXWAY B2BI

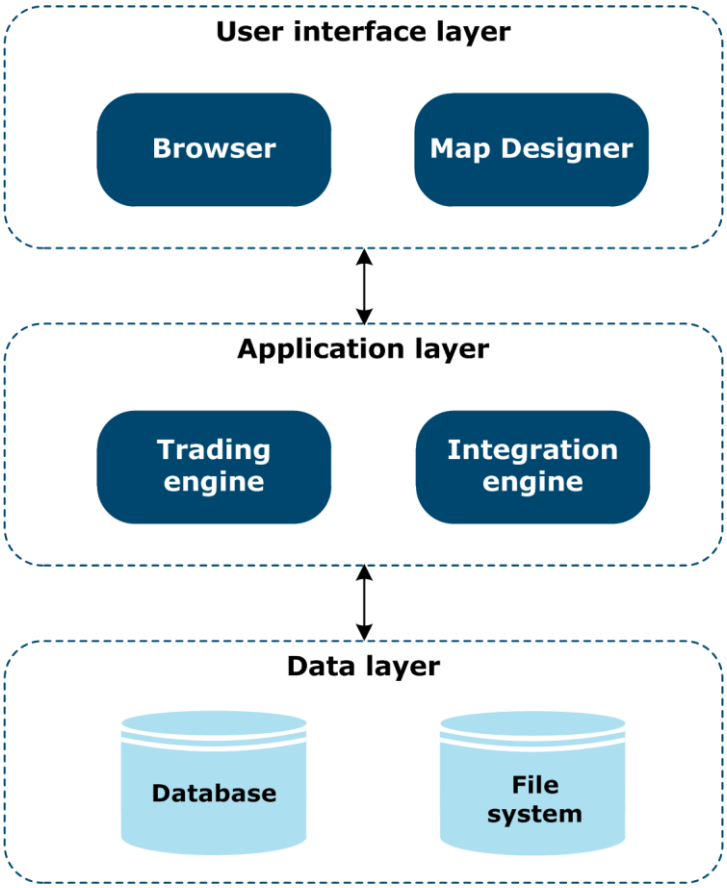
Axway B2Bi is an infrastructure used in automating message transformations from multiple different formats to any other supported formats and transferring them via various different messaging protocols. It is used to integrate different business partners want to continue using their own data formats or applications that do not support same messaging protocols. It can also be used as a middleware software to increase the security of message transfer. (Axway Software 2017)

There are many benefits to have an integration product that is one centralized hub to handle the communication and message transformations. In large organizations security can be enhanced greatly. For example, you only need to open firewalls from one business partner to B2Bi server once if organization has many systems that would otherwise have been connected straight to the business partner. (Axway Software 2017)

It is also possible to make message transformations in one of the endpoint of the transfer but centralizing every message transformation to one infrastructure makes the system easier for monitoring, error management and overall maintenance. (Axway Software 2017)

Products made especially for integration and message transfer purposes also offer much better tools for message transformations compared to tools provided by endpoint applications such as ERPs. (Axway Software 2017)

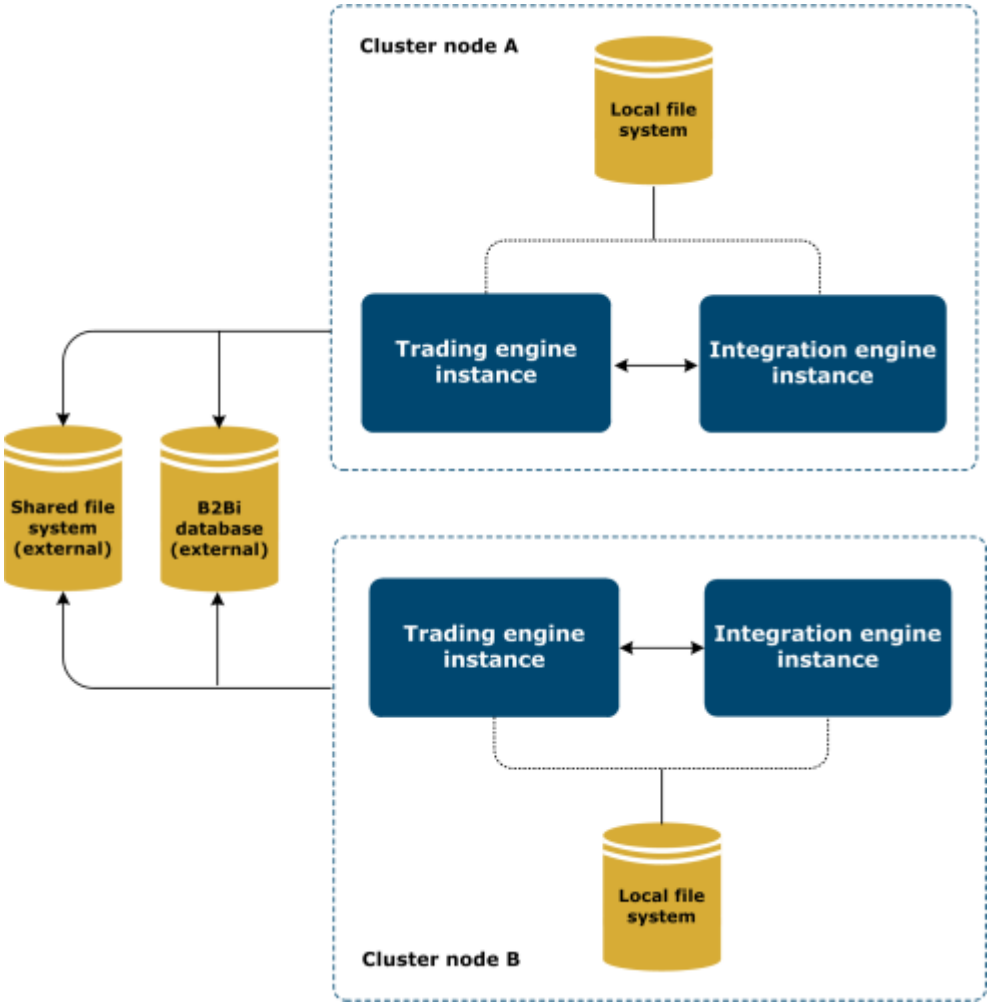
3.1 Architecture



Picture 3. Architecture of B2Bi (Axway Software 2018)

Architecture of B2Bi is structured in three functional layers. The layers are UI layer, application layer and data layer. The UI layer consists of regular browser that can be used to access interchange and Mapping Services that is used to develop and deploy mappings to integrator. The data layer contains B2Bi database and filesystem. The database is used to store all the configuration made in interchange and integrator. In the filesystem is used for configuration files and to store every message that has gone through B2Bi for backup purposes. (Axway Software 2018)

B2Bi can be installed in cluster using two or more installations of B2Bi in different servers to increase availability. In clustered installation the database and the filesystem are shared between the nodes as illustrated in picture 4. Both node still have their own local file systems for local data. (Axway Software 2018)



Picture 4. B2Bi installation in two node cluster (Axway Software 2018)

3.2 Mapping Services

Mapping Services is a standalone product for the tool that is used to design the transformations that are called mappings. In Mapping Services, the output and input formats will be described in components called business document. The business document is a logical expression of the structure of the input and output file. This will be explained in more detail in the application section of this thesis. (Axway Software 2014)

After making the business documents then a mapping component is created to set the rules based on which the file is transformed from input format to output format. This is also explained further in the application section of this thesis. (Axway Software 2014)

Lastly a mapping flow is created that describes the flow of the transformation connecting input to the mapping and the mapping to the output. In the mapping flow you can set up additional steps to the flow like splitting of the file etc. (Axway Software 2014)

3.3 Integrator

Integrator is a standalone product name for the integration engine of Axway B2Bi. It mainly handles the message format transformations and other back-end tasks. Integrator is based on Axway B2Bi's predecessor called XIB.

Integrator also provides some low-level monitoring tools. You can for example monitor each message that passes through the integration engine and observe each step including message detection, mapping and some possible post processing steps. Messages that are encrypted during transfer can also be read unencrypted in integrators message log. (Axway Software 2018)

Integrator supports multiple message format standards:

- ANSI X12
- EDIFACT
- ODETTE
- TRADACOMS
- EDIG@S
- VDA
- AAR/TDCC
- UCS/VICS
- MEC XML
- HL7 V3
- CCR
- CCD
- NACHA 2007, 2009, 2012 and 2014
- FIXML
- Fedwire
- BAI2
- SEPA
- XML (with full support of W3C XML Schema and DTDs)

- Database
- Flat file (CSV or fixed)
- SAP IDOC
- Lawson MEC XML
- Inhouse (Any variable and/or hierarchical format)
- COBOL (fixed/variable segments)

Transformations can be done between any of these formats. With inhouse configuration nearly any custom format can be transformed to some standard format or to another inhouse format. (Axway Software 2018)

Integrator can be installed as a cluster of two or more servers for high availability. This model provides additional robustness and maintainability for the system updates or hotfixes can easily be installed to one cluster node at once to prevent any downtime. Also, in case of server crashes or other problems in other node the other that is still running can automatically take over the process. Clustering the installation is also used for load balancing purposes. B2Bi can automatically distribute messages to either of the integration engine nodes based on current load. (Axway Software 2018)

3.4 Interchange

Interchange is a standalone product name for the trading engine of Axway B2Bi. It handles message transfers and routes them to the integration engine for message transformations. It also provides the graphical user interface to configure the flow of the integration. Like integrator it can be installed to multiple operating systems such as Windows Server or Linux. It also supports multiple databases like Microsoft SQL or MySQL. (Axway Software 2018)

For message transfers you can use multiple different message transfer protocols both modern secure protocols such as Web Services or legacy protocols such as FTP. It is also certified for interoperability for AS1, AS2, AS3 and ebXML. Interchange can support many other message protocols as well. Here is a list of all the message protocols and their corresponding transport protocols in brackets that supported:

- EDIINT AS1, AS2, AS3,
- AS4
- OFTP (V1 and V2) over TCP/IP, X.25,and ISDN

- RosettaNet RNIF (V1.1 and V2)
- ebXML/ebMS (V2)
- Web Services, SOAP and WSDL
- X.400 over X.25 and TCP/IP (X.420 and X.435 profiles)
- HTTP, HTTP/S, Staged HTTP
- Web Servlet
- Axway Transfer CFT, and PeSIT
- FTP, S/FTP, FTP/S
- JMS
- MLLP
- IBM WebSphere MQ
- cXML
- WebDAV
- Secure email (via SMTP)
- SDK for custom protocols

Like integration engine trading engine can be installed as a cluster to provide better availability. Should the other interchange server be offline for any reason the other would still be able to continue data transfers uninterrupted. This can be crucial for the medical supplier since even few hours of downtime would mean that orders from local pharmacies are failed to be delivered in time. A cluster installation will also ease maintenance activities. You can install updated or reboot servers one by one while still being able to receive, process and send messages. (Axway Software 2018)

Interchange also provides some high-level monitoring tools which can be used to monitor message transfers to and from the system.

4 FILE FORMATS

4.1 DESADV

DESADV or dispatch advice message is a message standard defined by the UN based on EDIFACT syntax. DESADV is used between trading partners to send information about goods that either dispatched or ready to be dispatched under agreed conditions. It is a universal format that can be used globally and isn't dependent on any specific industry or form of business. (UNECE 1990)

DESADV is defined to have 37 different possible segments. The segments and their descriptions in alphabetical order are: (UNECE 2000)

- ALI Additional information
- BGM Beginning of message
- CNT Control total
- COD Component details
- COM Communication contact
- CPS Consignment packing sequence
- CTA Contact information
- CUX Currencies
- DGS Dangerous goods
- DLM Delivery limitations
- DTM Date/time/period
- EQA Attached equipment
- EQD Equipment details
- FTX Free text
- GIN Goods identity number
- GIR Related identification numbers
- HAN Handling instructions
- IMD Item description
- LIN Line item
- LOC Place/location identification
- MEA Measurements
- MOA Monetary amount

- NAD Name and address
- PAC Package
- PCD Percentage details
- PCI Package identification
- PIA Additional product id
- QTY Quantity
- QVR Quantity variances
- RFF Reference
- SEL Seal number
- SGP Split goods placement
- TDT Details of transport
- TMD Transport movement details
- TOD Terms of delivery or transport
- UNH Message header
- UNT Message trailer

DESADV is the format that the principal is using to send the advanced shipping notice message to the medical supplier. Local ERP however wants the file in DEASN inhouse format, so a transformation is needed in Bridge. (UNECE 2000)

4.2 CMF

CMF or Custom Middle Format is the format that file is transformed when it is sent to Bridge regardless of the direction of the flow. CMF is an XML file that is based on an XSD schema defined by the medical supplier's IT architects. Each interface has its own XSD schema and thus its own CMF.

The CMF file is defined in XSD schema in code 1. The schema defines the structure and XML tags and their properties such as the datatype of the tag and whether the tags are mandatory or not.

```
<?xml version="1.0" encoding="utf-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:phub="urn:schema:phub:cmf:advancedshippingnotice:1"
xmlns:typ="urn:schema:phub:cmf:common:types:1"
    targetNamespace="urn:schema:phub:cmf:advancedshippingnotice:1"
```

```
    elementFormDefault="qualified"
attributeFormDefault="unqualified"
    version="1.0">
    <xsd:import namespace="urn:schema:phub:cmf:common:types:1"
        schemaLocation="./common/v1/Types.xsd" />
    <xsd:element name="advancedShippingNotice"
type="phub:AdvancedShippingNoticeType">
    </xsd:element>
    <xsd:complexType name="AdvancedShippingNoticeType">
        <xsd:sequence>
            <xsd:element name="sender" type="typ:PartyType">
            </xsd:element>
            <xsd:element name="receiver" type="typ:PartyType">
            </xsd:element>
            <xsd:element name="shipFrom" type="typ:PartyType"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="principalsShipmentID"
type="typ:IDType">
            </xsd:element>
            <xsd:element name="principalsPurchaseOrderID"
type="typ:IDType"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="theMedicalSupplierPurchaseOrderID"
type="typ:IDType"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="shipmentTrackingID" type="typ:IDType"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="advancedShippingNoteDate"
type="xsd:date"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="goodsReceiptDateOfAdviced"
type="xsd:date">
            </xsd:element>
            <xsd:element name="planedTotalNumberOfPallets"
type="typ:QuantityType"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="weightPerTotalDelivery"
type="typ:WeightType"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="truckLicenceID" type="xsd:string"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="carrierName" type="xsd:string"
            minOccurs="0">
            </xsd:element>
            <xsd:element name="advancedShippingNoticeLineItem"
            type="phub:AdvancedShippingNoticeLineItemType"
            maxOccurs="unbounded">
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="identification" type="xsd:string"
            use="required">
```

```
        </xsd:attribute>
        <xsd:attribute name="messageID" type="xsd:string"
            use="optional">
        </xsd:attribute>
        <xsd:attribute name="creationTimestamp" type="xsd:dateTime"
            use="required">
        </xsd:attribute>
    </xsd:complexType>
    <xsd:complexType name="AdvancedShippingNoticeLineItemType">
        <xsd:sequence>
            <xsd:element name="principalsShippmentLineID"
                type="xsd:positiveInteger" minOccurs="0">
            </xsd:element>
            <xsd:element name="principalsDeliveyNoteLineID"
                type="xsd:positiveInteger" maxOccurs="1"
minOccurs="0">
            </xsd:element>
            <xsd:element name="asnLineID" type="xsd:positiveInteger"
minOccurs="0">
            </xsd:element>
            <xsd:element name="principalsProductCode"
                type="xsd:string">
            </xsd:element>
            <xsd:element name="theMedicalSupplierProductCode"
type="xsd:string"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="productName" type="xsd:string"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="gtin14" type="typ:GTIN14Type"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="gtin8" type="typ:GTIN8Type"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="officialPharmaCode" type="xsd:string"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="batchCode1" type="xsd:string"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="batchCode2" type="xsd:string"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="batchExpiration" type="xsd:date"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="batchDateOfManufacturing"
type="xsd:date"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="batchDateOfFreshness" type="xsd:date"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="advicedQuantity"
                type="typ:QuantityType">
            </xsd:element>
            <xsd:element name="stockStatusCurrent"
```

```
                type="typ:StockStatusType" minOccurs="0">
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:schema>
```

Code 1. Advanced shipping notice CMF schema

4.3 DEASN

DEASN is an inhouse format defined by the medical supplier's local ERP's IT team. It is based on the interface specifications provided by the medical supplier. The file is fixed length and its rows are defined to be 1000 bytes long. The file has first one header row after which the rest of the lines are content lines.

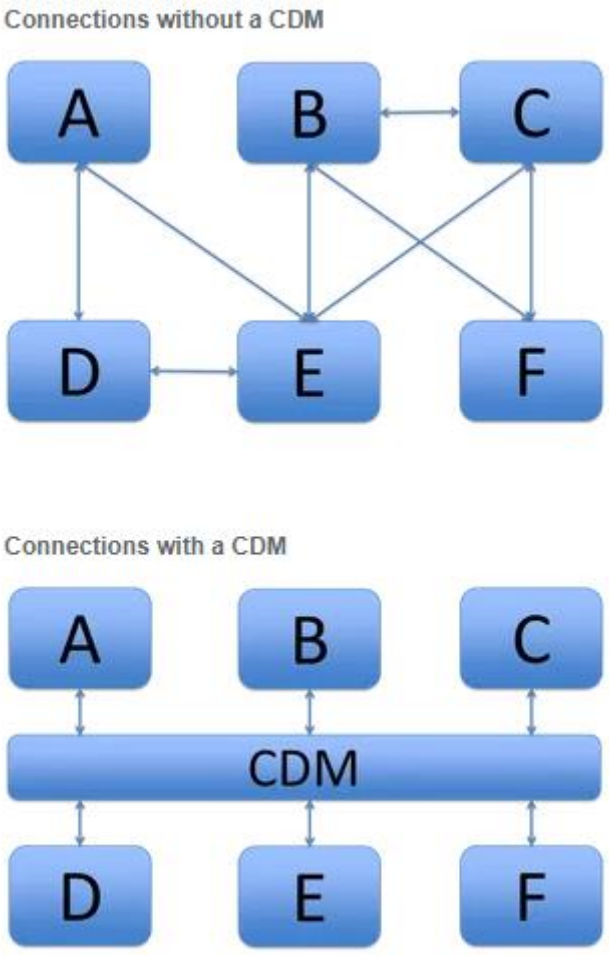
All fields are mandatory in terms of keeping file structure. The fields are separated by semicolon. While every field is defined to be required not every field is required to have data. If a field doesn't have data it will be filled with whitespace characters or zeroes depending on if the file is defined to be numeric field, string field or date field to maintain the required length of 1000 bytes.

Numeric fields will be filled with leading zeroes should the value be shorter than what is defined. String fields are filled with trailing whitespace characters. Numeric fields are always defined to be integers and all real numbers will be converted to integers.

The interface specifications also specify some fields to have default values or static values that will always be the same.

5 BRIDGE

Bridge is an architecture that provides centralized B2B communication between specialized local ERP systems and principal's systems. It utilizes canonical data model to reduce effort for B2B communication. The implementation of canonical data model in Bridge architecture is the CMF.



Picture 5. Transformations with and without CDM

With canonical data model you can significantly reduce the number of translations done between different formats. In the example in picture 5 you would need 16 transformations when not using CDM and only 12 when using CMD. The benefit is increased the bigger the system gets.

5.1 Purpose of Bridge

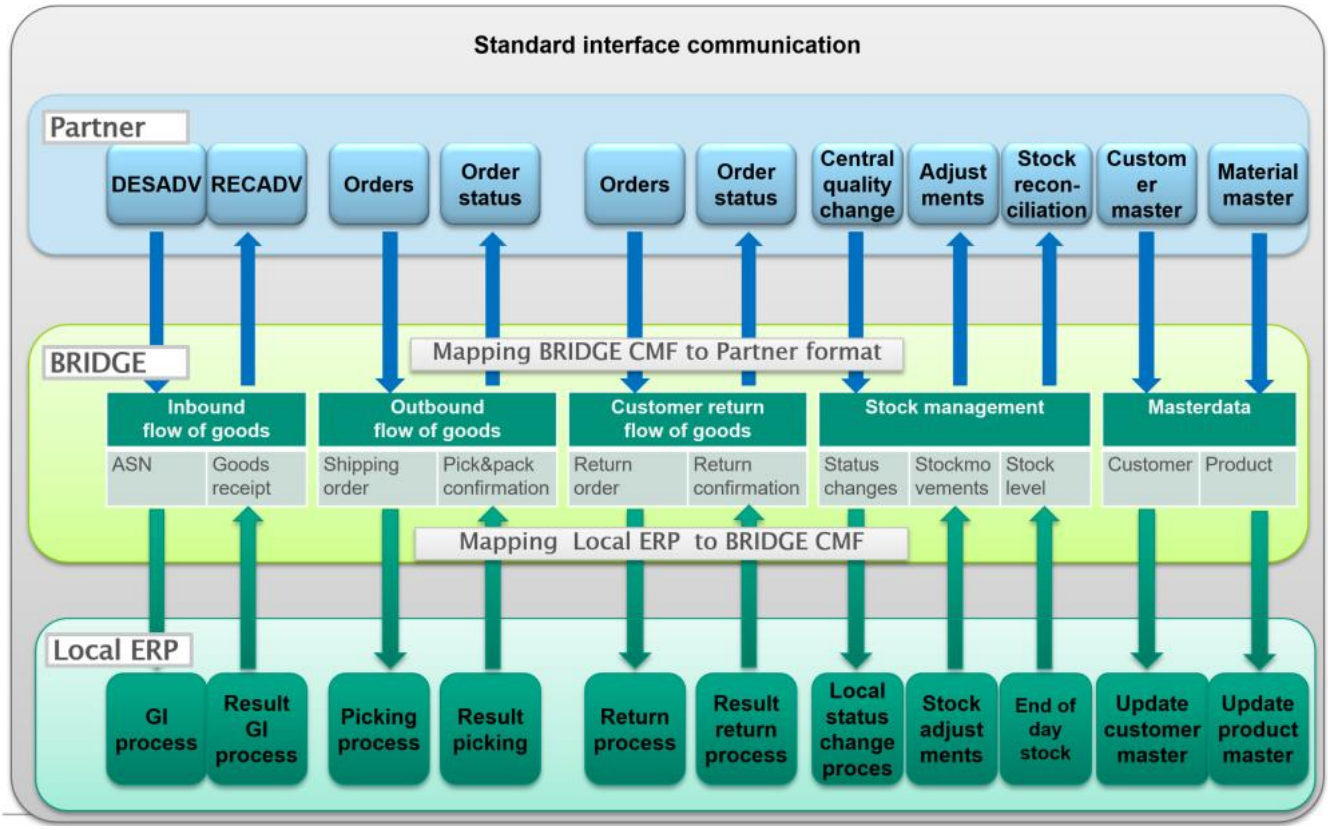
Bridge was designed to provide flexibility for principal specific needs. With nine different ERP systems and tens of principals in different countries there are lots of benefits when making message transformation in one centralized location instead of making then for example inside the local ERP systems. This makes maintenance, error monitoring and administration easier.

Bridge was also designed to reduce development effort to create new principal connections. This can be achieved by reducing redundancy in development work with canonical data model. If for example transformation from DESADV file format to CMF file format is made once every future principal connections using DESADV format for advance shipping notice messages can use the same interface.

Interfaces are developed to be versatile for multiple principals. This can be achieved with cross reference lookup database tables. When implementing a new principal that can utilize an interface that has already been developed you only need to add new database values for the new cross reference lookup values.

5.2 Bridge standard interfaces

In the Bridge standard interface communication design there are 11 different possible interfaces. With some principal connections some of the interfaces might not be implemented or some additional non-standard interfaces might be implemented. The standard interfaces and the direction of the flow of data is illustrated in picture 6. With the principal I am working with all interfaces except customer master data and material master data are implemented. During this thesis the implementation of the inbound ASN interface will be examined in detail.



Picture 6. Bridge standard interface communication design

6 APPLICATION

In this section the implementation of the advanced shipping notice interface will be examined in close detail. The purpose of the interface is to send information about an upcoming delivery to the medical supplier's warehouse. The direction of the interface's flow is inbound. It is received with AS2 HTTP protocol encrypted and signed using appropriate certificates to ensure security of transport.

The ASN message is needed to update local ERP for upcoming goods receipt. It is used to create the goods receipt confirmation message that is an outbound message sent to principal confirming that goods are received by the warehouse so that principal's ERP system can update stock information.

Transformation and transportation of the file is done using Axway's B2Bi integration product. AS2 connectivity including decrypting the message is handled by interchange. The actual transformation is handled by integrator.

The DESADV message coming from principal is first transformed to CMF. From CMF the file is then converted to DEASN format. Conversion between DESADV to CMF and CMF to DEASN are kept separate to make it easier to detect the stage is failure for support process after go-live.

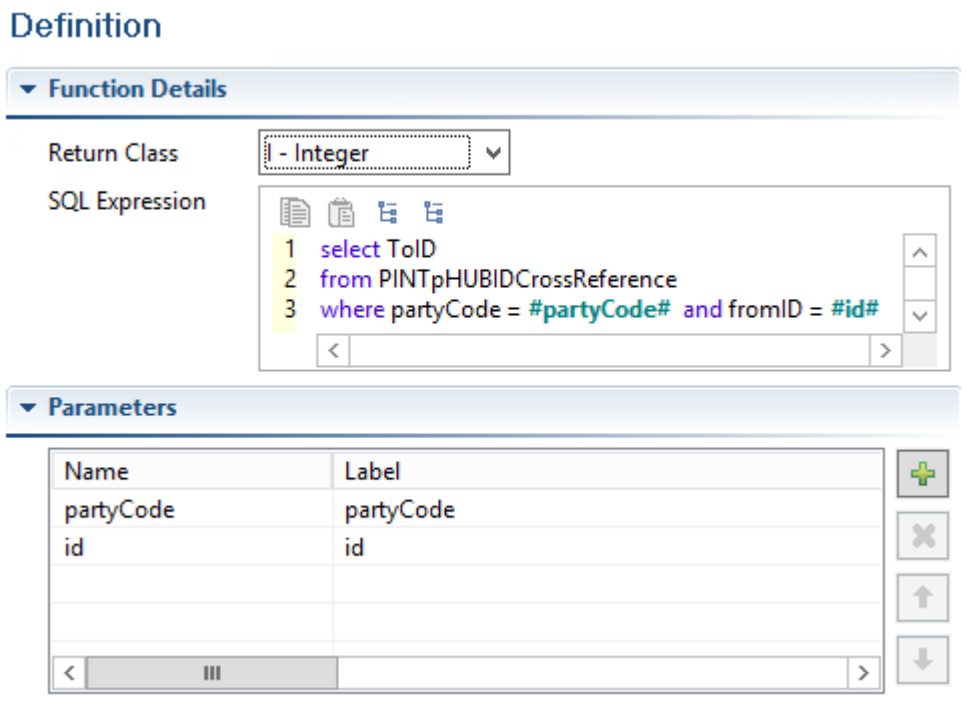
After conversion the file is then delivered to local ERP's FTP server. File transfer between Bridge and local ERP can be done using FTP because Bridge is located in DMZ and local ERP's FTP server is located in LAN. This way files never goes outside the medical supplier's internal network and is safe to send unencrypted. File is then loaded from FTP to local ERP which in this case is SAP.

6.1 Mapping from principal to Bridge

Transformation from DESADV to CMF follows rules defined in mapping specifications defined by the medical supplier's business team. The mapping specifications define which fields are to be mapped to which fields and should the data in the field need any alterations.

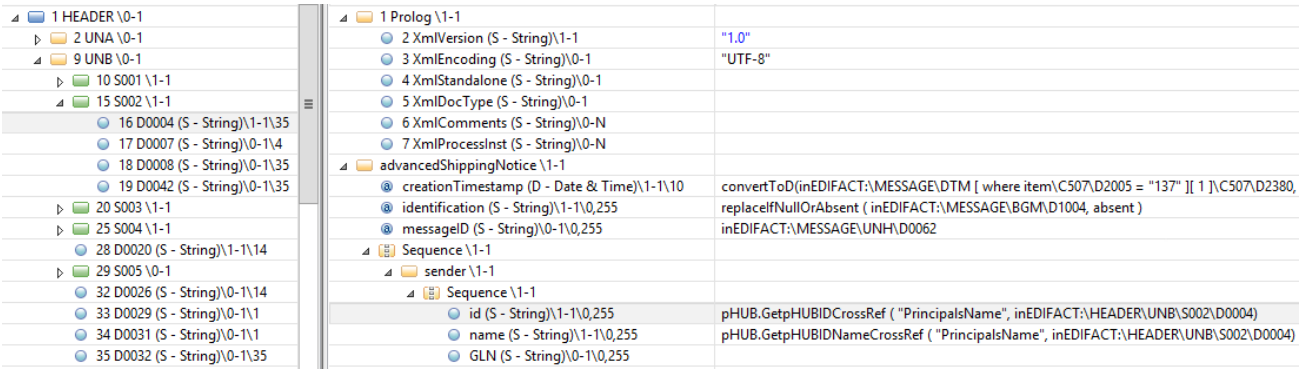
Mapping specifications also specify the cross-reference table that is used to convert certain values according to pre-defined rules. In stock management interfaces this means more complicated rules for transforming stock statuses but in this ASN interface the only fields that needs conversion with cross-reference tables are sender's and receiver's ID and name fields. This is because both the principal and the medical supplier has their own definitions in their ERP systems.

Cross-reference tables are created using Mapping Services's ability to connect to database and execute SQL queries using custom defined functions. For the ASN interface for example custom function defined in picture 7 is used to get the sender or received ID used in local ERP. The function takes the partners name and the ID used in principal's ERP as parameters and inserts them into the SQL query.



Picture 7. Custom function for cross-reference table

Fields are mapped by writing expressions in Mapping Services. In picture 8 is an example how sender ID field is mapped. The custom function defined in picture 7 is used in this field to use the cross-reference table. The function takes in the value from sender ID field found in the UNB segment of the DESADV. Also, principal's name is given as a static value because the cross-reference database table contains values for multiple principals.



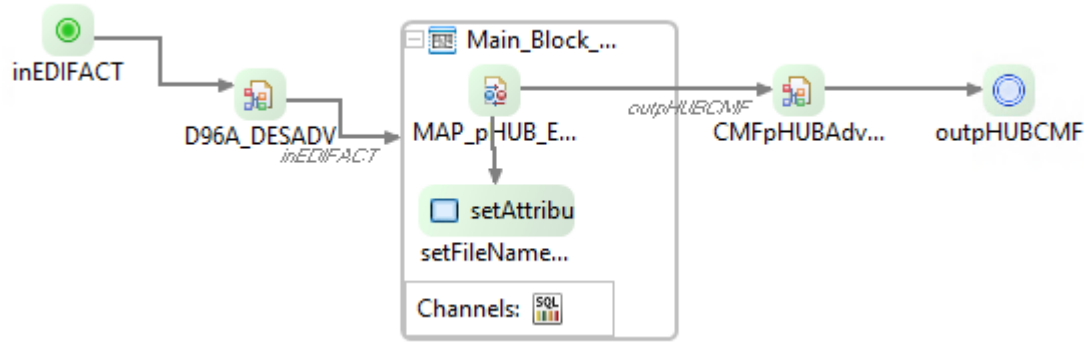
Picture 8. Example of mapping the ID field

After mapping of every field is done we can create a mapping flow based on the mapping. In mapping flow the flow of the transformation is defined. In picture 9 is the mapping flow for the B2B side of this integration. It defines that the integration has one input and output and connects the business documents to the mapping block.

In the mapping block additional configuration is necessary. It has a setAttribute block that contains several expressions defining attributes for the message. For example, here the filename for the message is set. Some attributes are added to make support process easier in the future. These attributes are for example integration ID, document type and document number. These attributes can be used to easily identify the integration or search certain documents from logs.

In the mapping block the database connection is also defined. This is just a expression that states that there is a database connection and it is used by the integration but the actual connection parameters are defined in the interchange.

The mapping flow can be simulated in the Mapping Services, so the transformation can be validated before creating the interchange configuration.



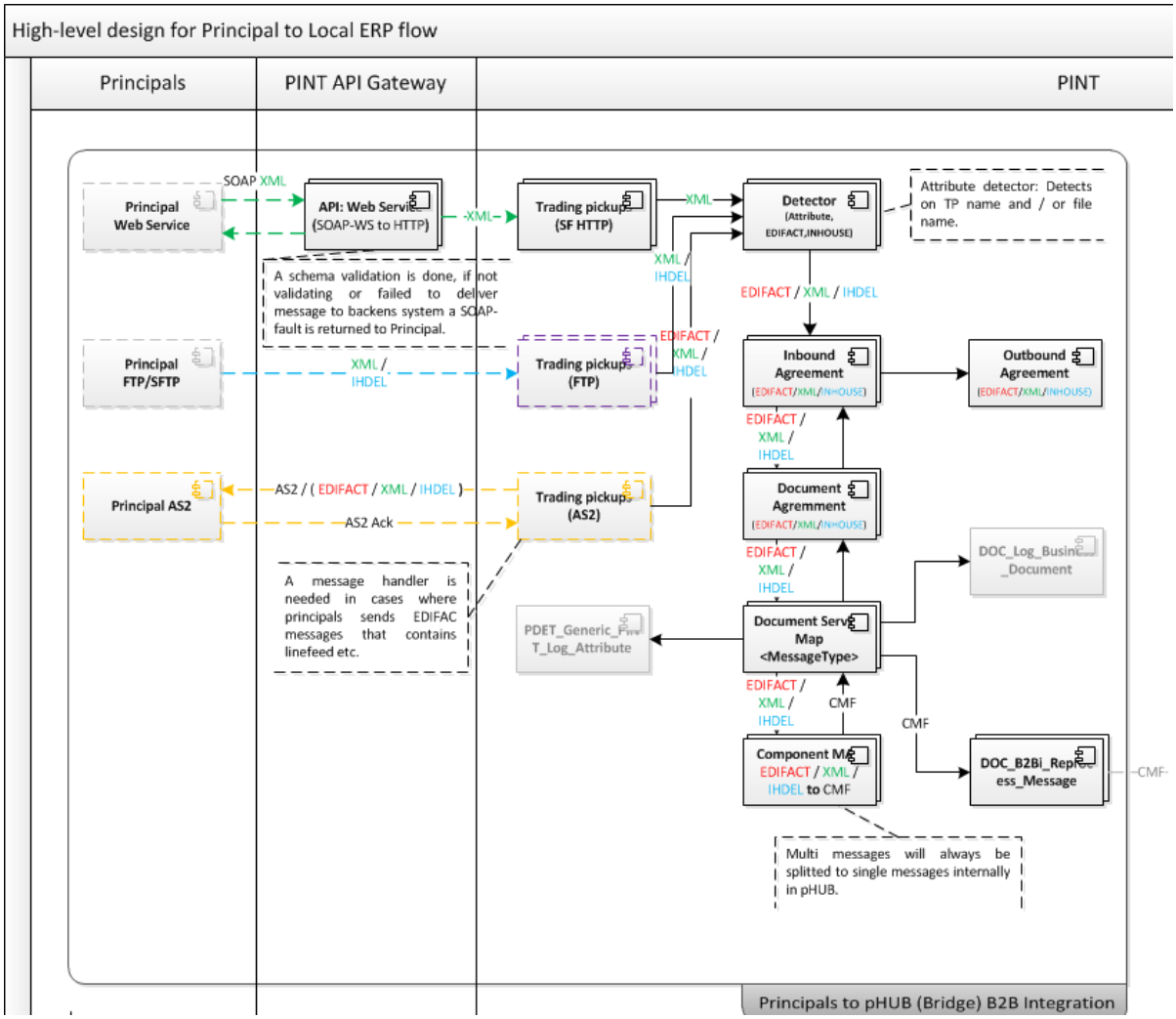
Picture 9. Mapping flow in B2B transformation

6.2 Trading engine configuration to Bridge

The high-level design in picture 10 defines the flow of the integration in interchange. It is a common design, so it has every possible input method illustrated. While B2Bi is capable of using many more different transport protocols it was defined by the medical supplier's IT architect that only allowed transport protocols are AS2, SFTP or Web Service. For this particular principal AS2 was chosen to be the transport protocol.

The message is first delivered to B2Bi with trading pickup. With AS2 a common trading pickup is used that specifies a HTTP URL that principal will use to deliver the file using HTTP POST method encrypted and signed using the medical supplier's AS2 certificate.

The file is then detected by a detector component. Since the principal is sending DESADV message in EDIFACT format an EDIFACT detector can be used. The detector checks the sender's and receiver's IDs and connects the message to an agreement that has the corresponding IDs defined.



Picture 10. High level design of B2B side of the integration

The agreement is used to connect everything from sender A to receiver B. This is defined using messaging IDs that are configured for each principal in the interchange. The agreements also specify the format of the message which in this case is EDIFACT. For every different message type and for every principal a new agreement must be created.

In the agreements we then specify a document agreement that further specifies the document's type which in this case is DESADV. The document agreement then specifies the service artifact for the integration, so the message can now be connected to the right mapping.

Document agreements

Document name	Document version	Document type	Service
DESADV_D96A_BGM351	D96A	DESADV	SP_P_pHUB_EDIFACT_DESADV_D96A_PrincipalName_to_pHUBCMF_AdvancedShippingNotice

Picture 11. Document agreement for the ASN

In the service a component artifact is then specified. The component is the artifact that specifies the mapping for the flow. In the service some additional processing components can be specified. In this case we use a post detection component that adds the messages attributes that was set in the mapping in logs for easier monitoring. After the mapping component there are two document type components that are used. The first adds a row to log that helps finding the message. The other reprocesses the message. This reprocessing component is used as a delivery method when the file is coming to Bridge. The reprocessed message will go through the same steps as before, but it will now be in CMF.

Service

Name:* SP_P_pHUB_EDIFACT_DESADV_D96A_PrincipalsName_to_pHUBCMF

Type: Partner
Document format: EDIFACT
Document version: D96A
Document type: DESADV
Post detection component: PDET_Generic_PINT_Log_Attribute
Mapping component: MAP_pHUB_EDIFACT_DESADV_D96A_PrincipalsName_to_pHUBCMF_AdvancedShippingNotice

outpHUBCMF | Service attributes

Document

Type	Name	Component		
Document	DOC_Log_Business_Document	B2BX Application/PINT Log Business Document		Delete
Document	DOC_B2Bi_Reprocess_Message	B2BX Application/B2Bi Reprocess Message		Delete

Picture 12. Service's configuration.

The component artifact just specifies which mapping flow is used and if the mapping flow has some configuration needed it will also be specified there. In this case the database configuration for the cross-reference lookup table is specified. The configuration needs a JDBC connection string as well as credentials for the database.

6.3 Mapping from Bridge to local ERP

The mapping for the A2A side of the integration is also based on mapping specifications provided by the business side of the medical supplier. The A2A side of the integration also contains a cross-reference table that will be used to convert the sender's and receiver's ID and name fields from Bridge values to values expected by local ERP.

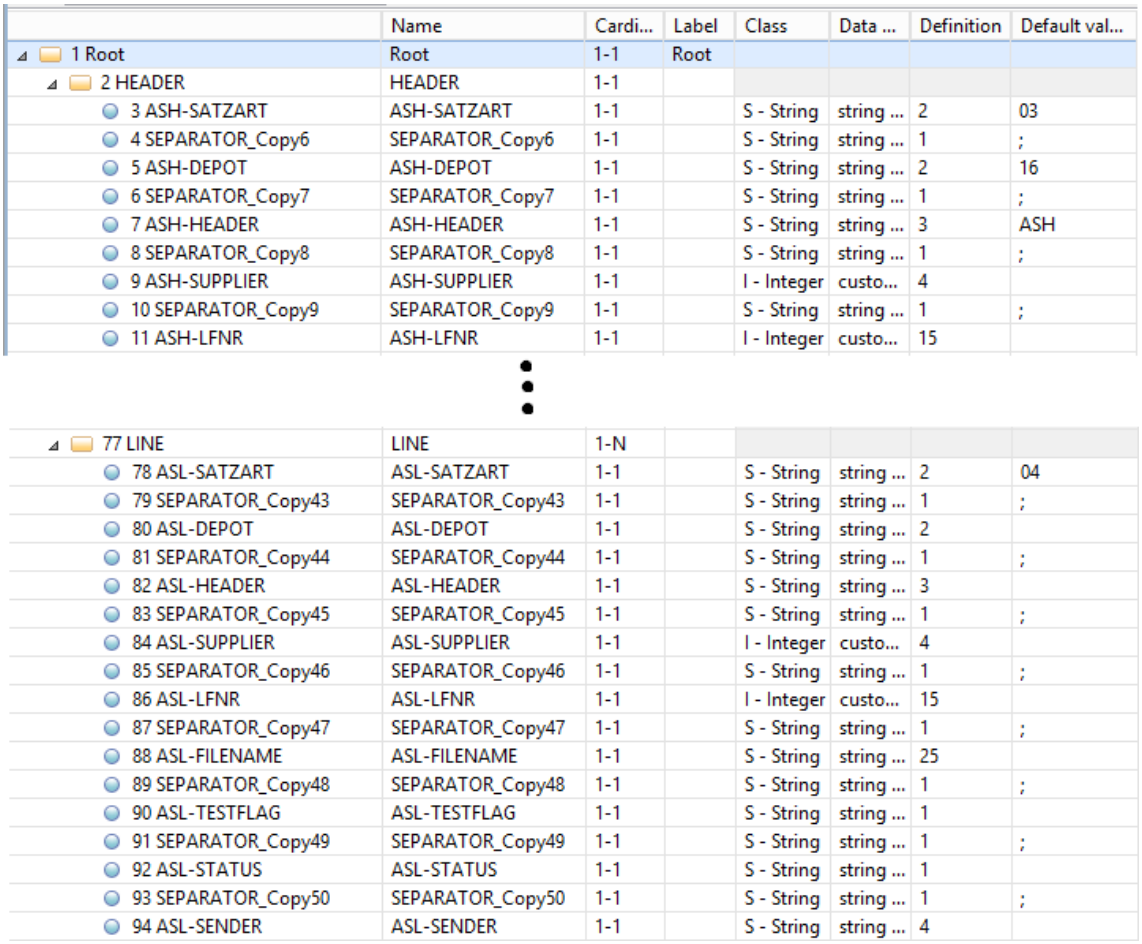
In this side of the mapping some limitations of the B2Bi forced to make some modifications to local ERP's initial wishes. Local ERP would have wanted that messages would be delivered even when faulty data is provided from the principal violating the length requirements. This however was not possible to do in B2Bi. The fields need to be specified to be fixed length in order to get leading zeroes and trailing whitespace characters that were defined in the DEASN file format definitions. However, when a field is defined to be fixed it will fail if too long data is inserted in the field. Unfortunately, this feature cannot be turned off making messages with faulty data fail. The issue will be overcome by defining a sufficient support process before go-live date.

The mapping process is very similar in A2A side that it was on B2B side. The main difference is the file formats. In this side of the integration the input format is CMF, and the output format is DEASN.

One key difference in this side is also that one of the formats is not a standardized format. When dealing with inhouse file formats a custom business document needs to be created. With standardized formats like XML you can create the business document automatically by exporting XSD schema.

Inhouse business documents are created by defining the structure of the file by using parent nodes and leaf nodes. Parent nodes are representations of sequences or individual lines. In picture 13 the header parent is an example of individual line and the line parent is an example of sequence. The cardinality defines whether the parent appears only once, many times or can it be empty.

The leaf nodes specify individual fields containing data. The cardinality for these can be either 1-1 or 0-1 where the first means the field is mandatory and the latter that its optional.



	Name	Cardi...	Label	Class	Data ...	Definition	Default val...
1 Root	Root	1-1	Root				
2 HEADER	HEADER	1-1					
3 ASH-SATZART	ASH-SATZART	1-1		S - String	string ...	2	03
4 SEPARATOR_Copy6	SEPARATOR_Copy6	1-1		S - String	string ...	1	;
5 ASH-DEPOT	ASH-DEPOT	1-1		S - String	string ...	2	16
6 SEPARATOR_Copy7	SEPARATOR_Copy7	1-1		S - String	string ...	1	;
7 ASH-HEADER	ASH-HEADER	1-1		S - String	string ...	3	ASH
8 SEPARATOR_Copy8	SEPARATOR_Copy8	1-1		S - String	string ...	1	;
9 ASH-SUPPLIER	ASH-SUPPLIER	1-1		I - Integer	custo...	4	
10 SEPARATOR_Copy9	SEPARATOR_Copy9	1-1		S - String	string ...	1	;
11 ASH-LFNR	ASH-LFNR	1-1		I - Integer	custo...	15	
77 LINE	LINE	1-N					
78 ASL-SATZART	ASL-SATZART	1-1		S - String	string ...	2	04
79 SEPARATOR_Copy43	SEPARATOR_Copy43	1-1		S - String	string ...	1	;
80 ASL-DEPOT	ASL-DEPOT	1-1		S - String	string ...	2	
81 SEPARATOR_Copy44	SEPARATOR_Copy44	1-1		S - String	string ...	1	;
82 ASL-HEADER	ASL-HEADER	1-1		S - String	string ...	3	
83 SEPARATOR_Copy45	SEPARATOR_Copy45	1-1		S - String	string ...	1	;
84 ASL-SUPPLIER	ASL-SUPPLIER	1-1		I - Integer	custo...	4	
85 SEPARATOR_Copy46	SEPARATOR_Copy46	1-1		S - String	string ...	1	;
86 ASL-LFNR	ASL-LFNR	1-1		I - Integer	custo...	15	
87 SEPARATOR_Copy47	SEPARATOR_Copy47	1-1		S - String	string ...	1	;
88 ASL-FILENAME	ASL-FILENAME	1-1		S - String	string ...	25	
89 SEPARATOR_Copy48	SEPARATOR_Copy48	1-1		S - String	string ...	1	;
90 ASL-TESTFLAG	ASL-TESTFLAG	1-1		S - String	string ...	1	
91 SEPARATOR_Copy49	SEPARATOR_Copy49	1-1		S - String	string ...	1	;
92 ASL-STATUS	ASL-STATUS	1-1		S - String	string ...	1	
93 SEPARATOR_Copy50	SEPARATOR_Copy50	1-1		S - String	string ...	1	;
94 ASL-SENDER	ASL-SENDER	1-1		S - String	string ...	4	

Picture 13. DEASN inhouse business document

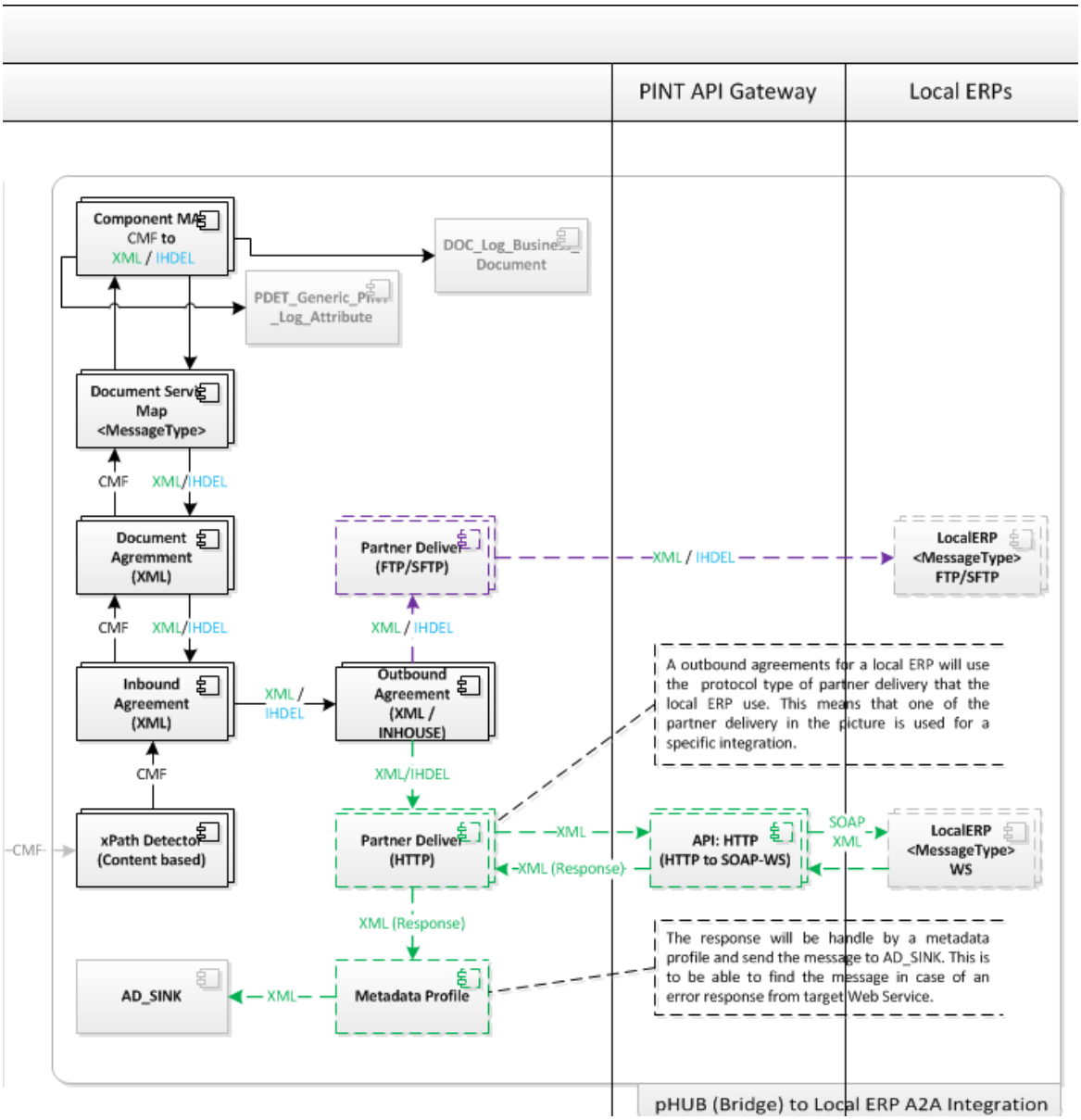
Each leaf node is specified to have a datatype. In this interface the allowed datatypes are string, integer or date. The definition field contains the length definition for the field. the field is set to be fixed and will be filled to the required length. If field has some default value it can be specified in the default column.

The rest of the mapping will be identical process as the B2B side was. Mapping is done based on the mapping specifications by specifying expression for each output field. Cross-reference tables are once again used to convert name and ID values for sender and receiver.

Mapping flow will also be created identically as with B2B side. Attributes are added to make support process easier in the future. Database connection is specified for the cross-reference lookup tables.

6.4 Trading engine configuration to local ERP

The A2A side of the integration is quite similar to the B2B side with minor differences. Because the B2B side of the integration uses the reprocess message component as a delivery method this side of the integration doesn't need a pickup component for getting messages. Messages are already in the B2Bi and ready to be detected.



Picture 14. High level design of A2A side of the integration

As the file format for CMF is XML we can use an XPath detector to detect the message and connect it to corresponding agreement. XPath detector is configured to check the

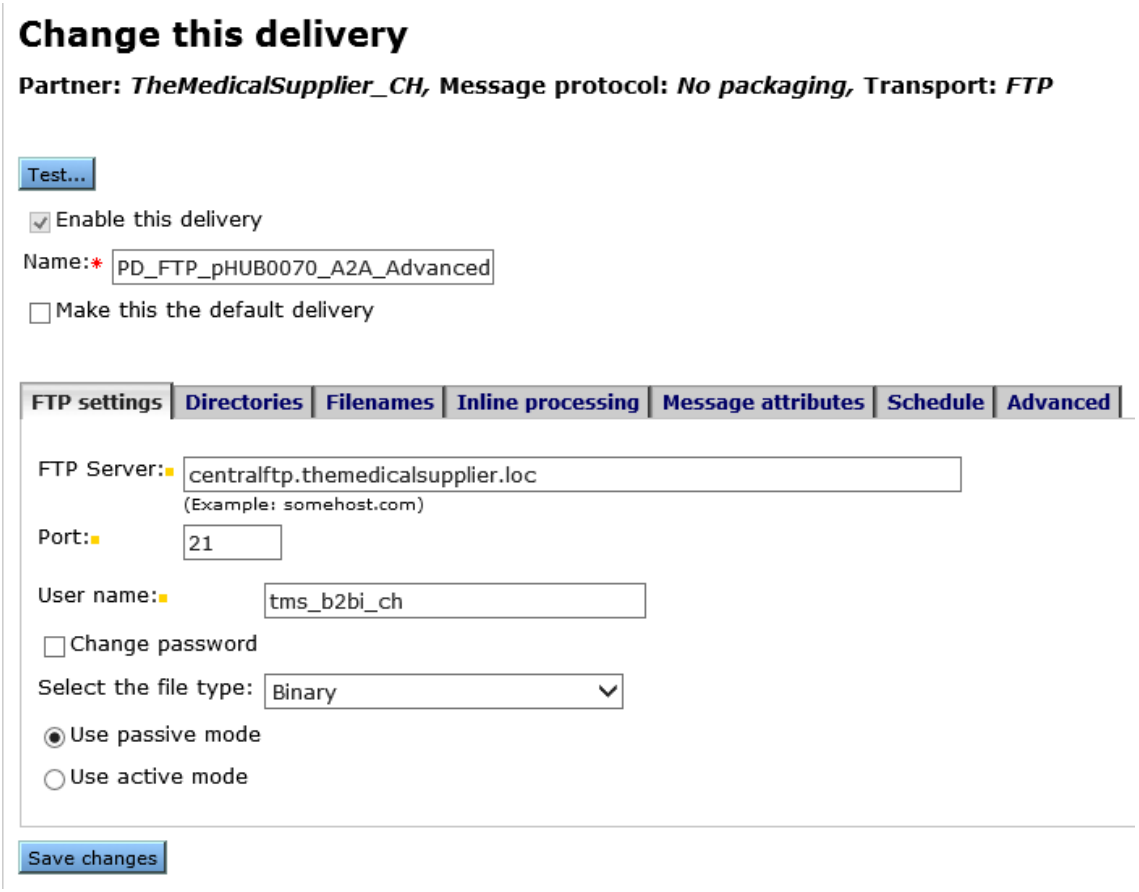
recipient's ID from the CMF file by providing XPath statement for the corresponding field. This is a common component and can be used with every interface regardless of the direction of the interface. Sender identifier is given as a static value because when detecting CMF files, the file is always coming from the Bridge. XPath detector also takes in document standard and type for matching the file to corresponding document agreement.

Picture 15. XPath detector configuration for A2A side of the integration

After detecting the message, it is delivered to agreement that has been configured to have matching messaging ID's that comes from the CMF's ID field. As well as in B2B side there will be new agreement for every principal and local ERP combination. Each interface will have its own document agreement configured to have the document type and standard.

In A2A side also the service is configured to have additional post detection component for logging attributes for easier support process. Service also houses the mapping component which is used to make the actual transformation from CMF to DEASN inhouse format required by the local ERP. Component takes in the database connection configuration the is used for the lookup table.

One difference in the A2A side is that the message is now delivered outside of B2Bi. Since the connection is still inside the medical supplier's private network, otherwise less secure FTP connection can be used. From the FTP server local ERP then transfers the message to their ERP's catalogue.



Picture 16. FTP delivery configuration to local ERP

7 CONCLUSION

The purpose of this bachelor's thesis was to examine the benefits of using canonical data model in integrations instead of traditional point to point integrations. The other purpose of this bachelor's thesis was to study how to implement the canonical data model in Axway's B2Bi integration product. The thesis was divided into theory section where the products, architectures and message standards are introduced and to a practical section where the canonical data model will be implemented as an interface for the medical supplier that commissioned this thesis.

The theory section was divided into three chapters. In the first chapter the Axway B2Bi integration product was introduced in detail. The second part gave explanations about the message formats and standards involved. With the canonical data model format defined for these interfaces were explained in more detail by providing the XSD schema definition of the format. The third part of the theory section described the Bridge architecture which is the implementation of the canonical data model for the medical supplier.

The canonical data model was examined in the application chapter by implementing one interface from the Bridge architecture. The application was done using Axway's B2Bi integration product that was introduced in earlier chapters. The interface chosen was the advanced shipping notice interface where DESADV formatted message will be transformed to the CMF and then to DEASN file format. File transfers were done using AS2 in the B2B side and with FTP in the A2A side.

Canonical data model proves to be very efficient way of doing integrations when the project is quite large in size. It can reduce the number of interfaces needed to be developed when the project grows. In the beginning though the canonical data model can be more labor intensive when for single connection two interfaces are needed. This will make the beginning of the development harder and more expensive. After this point of the development has been overcome the investment in the architecture will eventually pay itself back.

Traditional point-to-point integrations can still be valid option for many range of projects especially if they involve lots of inhouse formats. When dealing with lots of standardized formats such as the formats based on EDIFACT syntax or something similar canonical

data model is well worth considering as an alternative. After the initial design and implementation, it will offer reduction in cost and time used in the development.

Valuating the usefulness of the canonical data model could easily be continued based on this thesis. After implementing tens of interfaces for multiple countries ERPs it would be much easier to see the effects.

REFERENCES

Axway Software. 2017. Axway B2Bi Referred 26.2.2018 https://www.axway.com/sites/default/files/datasheet_files/axway_datasheet_b2bi_en.pdf

AMIS Tehcnology Blog. 2016. Benefits of canonical data model. Referred 24.3.2018 https://technology.amis.nl/2016/08/08/soa-benefits-of-a-canonical-data-model/

Axway Software. 2018. B2Bi architecture. Referred 24.3.2018 https://docs.axway.com/bundle/B2Bi_221_AdministratorsGuide_allOS_en_HTML5/page/Content/Operators_Guide/ops_guide_archi.htm

Axway Software. 2018. B2Bi cluster features. Referred 8.4.2018 https://docs.axway.com/bundle/B2Bi_221_AdministratorsGuide_allOS_en_HTML5/page/Content/B2Bi/cluster/concepts/b2b_AA_features.htm

Stylus Studio. 2000. Despatch advice message. Referred 5.3.2018 https://www.stylusstudio.com/edifact/D00A/DESADV.htm

United Nations Economic Commission for Europe (UNECE). 2000. Despatch advice message. Referred 1.3.2018 https://www.unece.org/trade/untdid/d00a/trmd/desadv_c.htm

United Nations Economic Commission for Europe (UNECE). 1990. Electronic data interchange for administration, commerce and transport (EDIFACT) - Application level syntax rules. Referred 1.3.2018 fhttp://www.unece.org/trade/untdid/texts/d422_d.htm

Axway Software. 2014. B2Bi 2.0.2 Developer Guide package. Referred 17.4.2018 https://support.axway.com/en/documents/document-details/id/1428131

Technopedia. 2018. Canonical Data Model (CDM). Referred 3.5.2018 https://www.techopedia.com/definition/30598/canonical-data-model-cdm

Gregor H and Bobby W. 2003. Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Referenced 3.5.2018