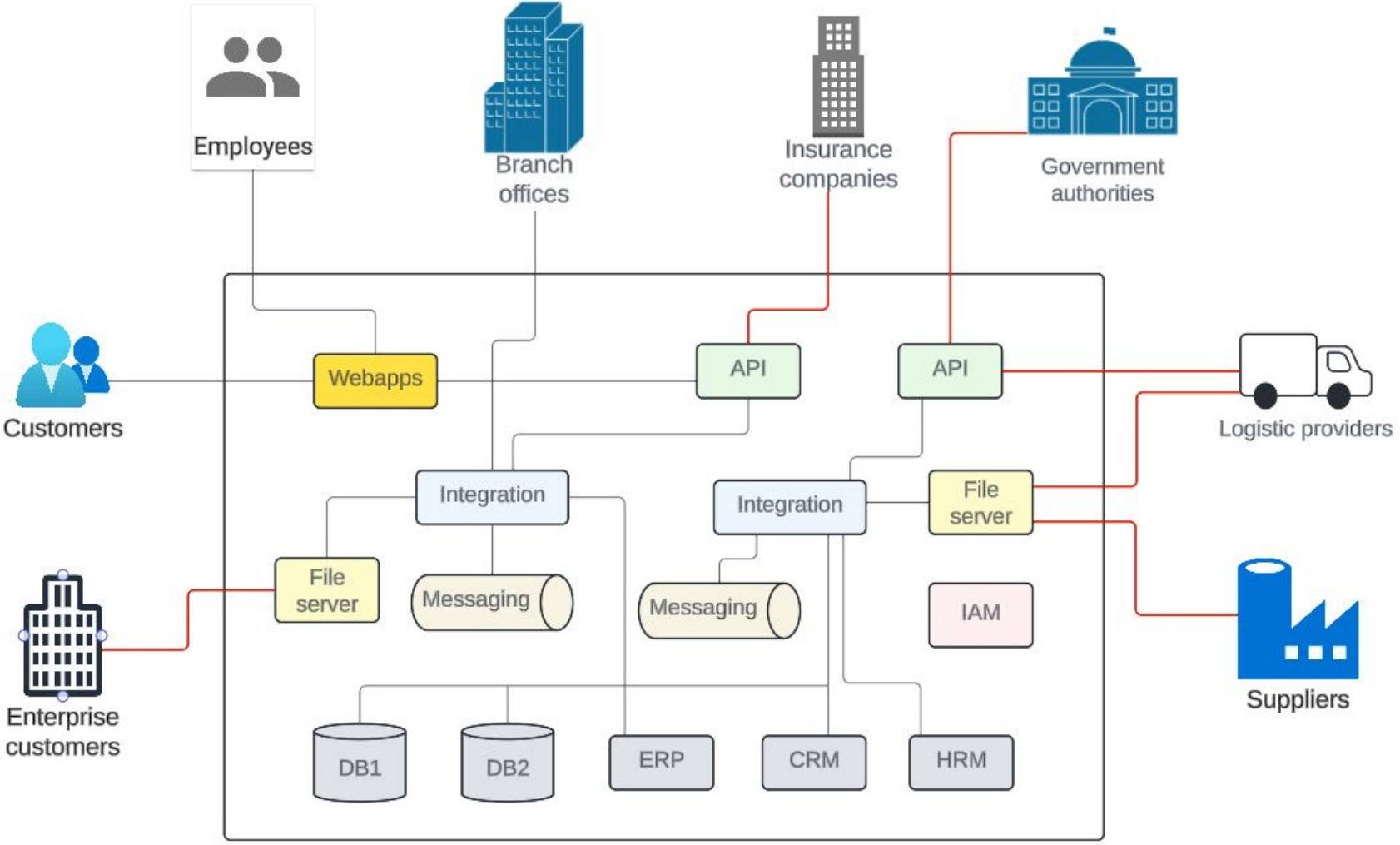
WSO2CON 2024

# Modernizing Legacy Systems using Ballerina

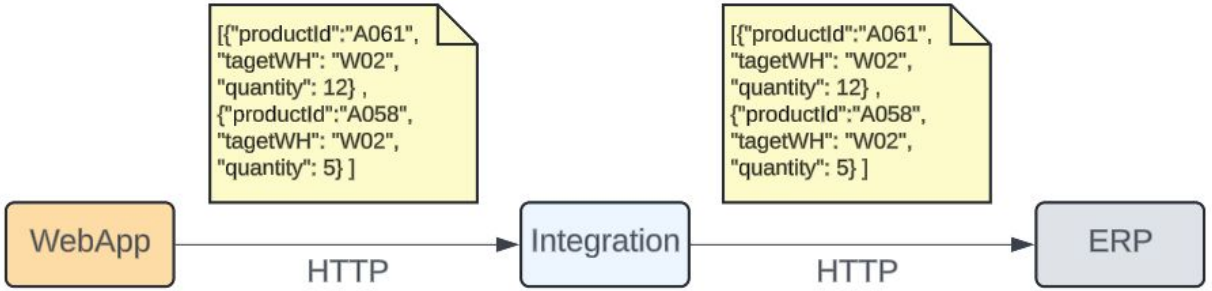


Chathura Ekanayake
Software Architect
WSO2

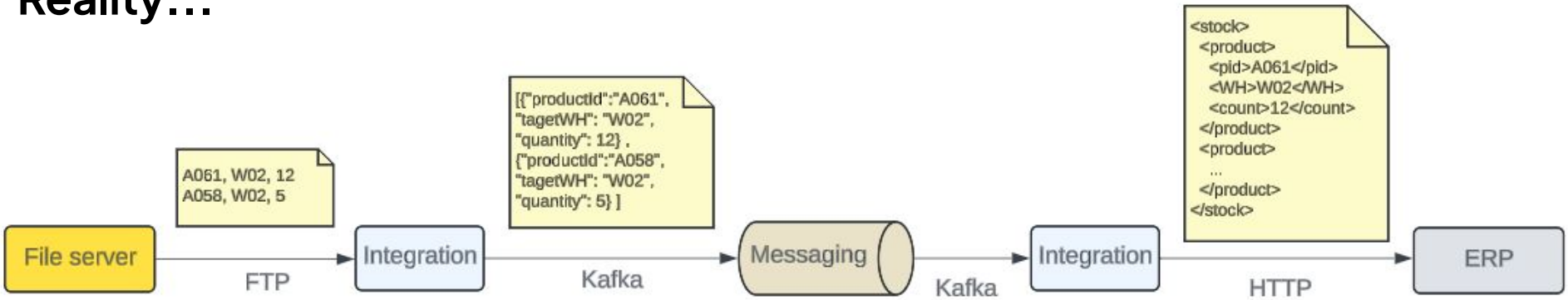
# A look at a typical enterprise…


Employees
Branch offices
Insurance companies
Government authorities
Customers
Webapps
API
API
Logistic providers
Integration
Integration
File server
File server
Messaging
Messaging
IAM
Enterprise customers
Suppliers
DB1
DB2
ERP
CRM
HRM

# Expectation…

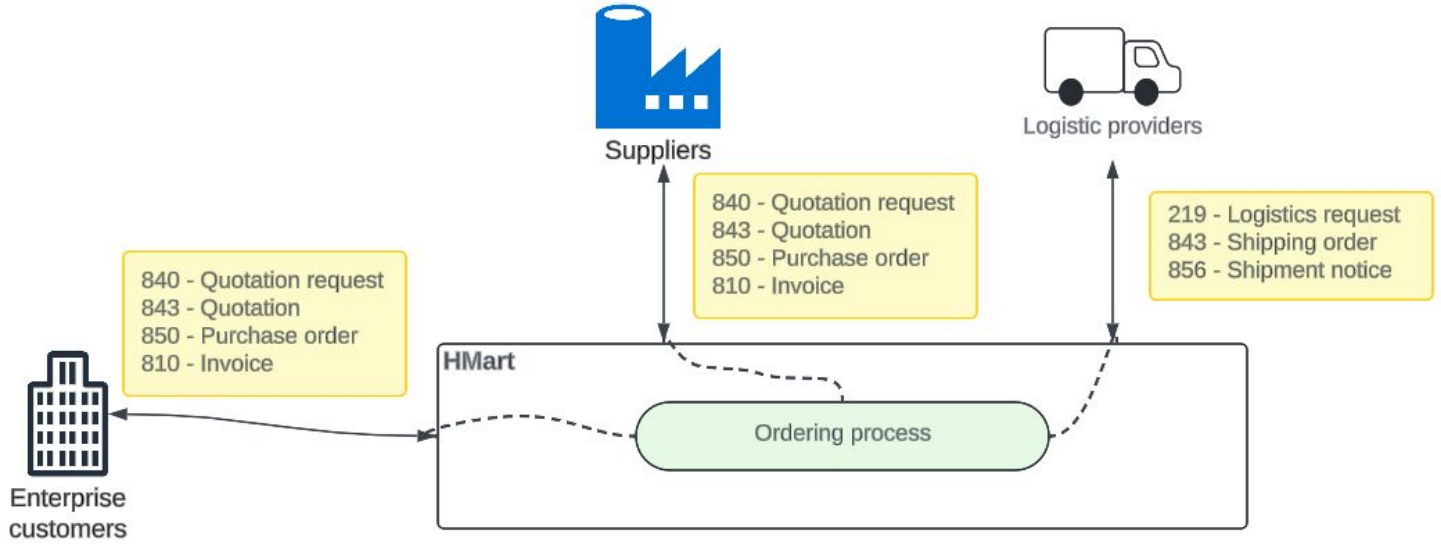



# Reality…

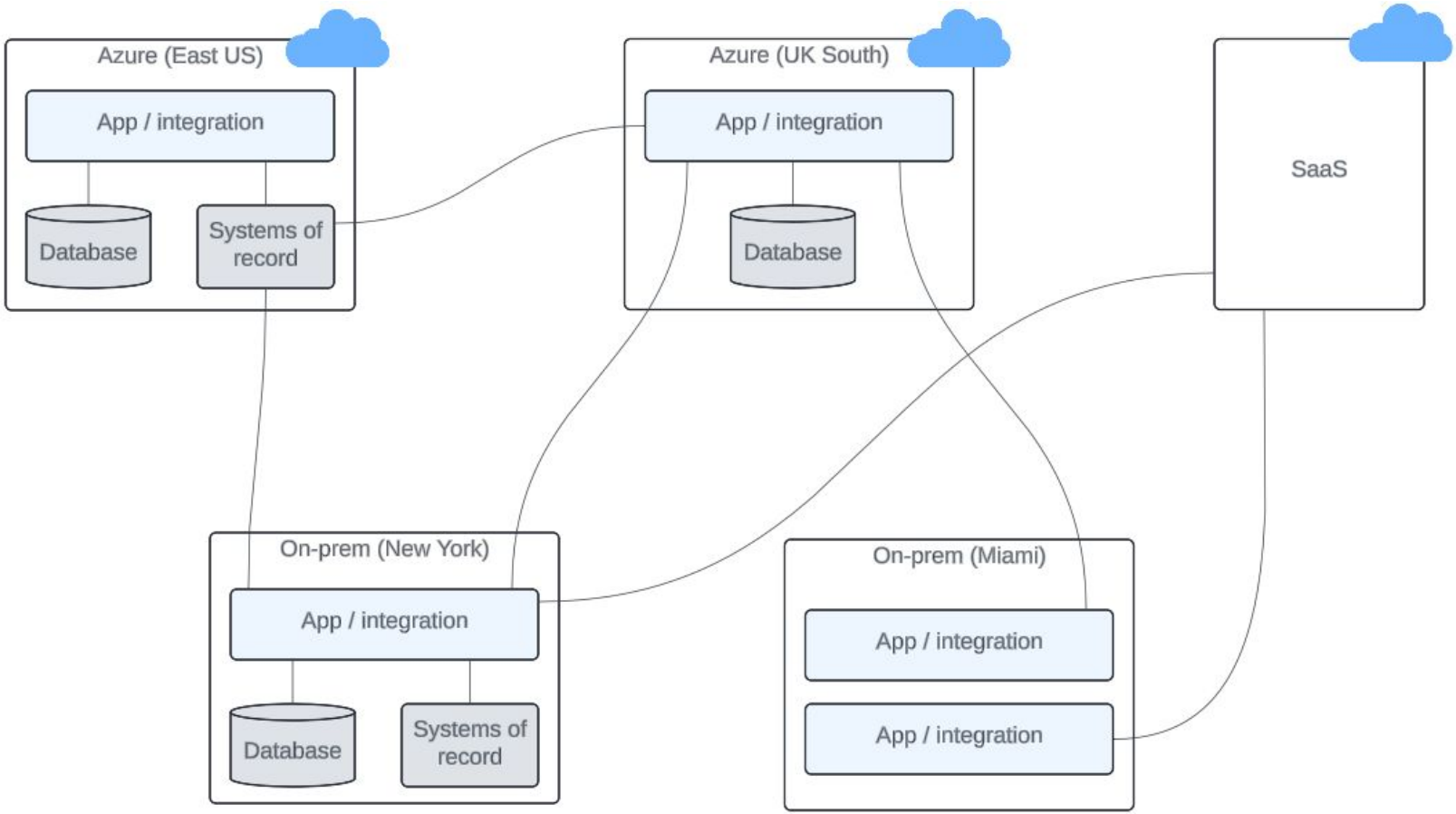


# Business-to-Business (B2B) interactions

- **Minimal control** over message formats, data and protocols
- **EDI** as a prominent message format
- Comprehensive standardizations, but often uses **variations**



# Deployed everywhere!



# A technology for integrating everything…

# Native support for JSON, XML and CSV

```
xml x = xml
    `<items>
        <!--Contents-->
        <book>
            <name>A Study in Scarlet</name>
            <author><name>Arthur Conan Doyle</name></author>
        </book>
        <planner>Daily Planner<kind>day</kind><pages>365</pages></planner>
        <book>
            <name>The Sign of Four</name>
            <author><name>Arthur Conan Doyle</name></author>
        </book>
        <pen><kind>marker</kind><color>blue</color></pen>
    </items>`;
```

```
json[] users = [
    {
        user: {
            name: {
                firstName: "John",
                lastname: "Smith"
            },
            age: 24
        }
    },
    null
];
```

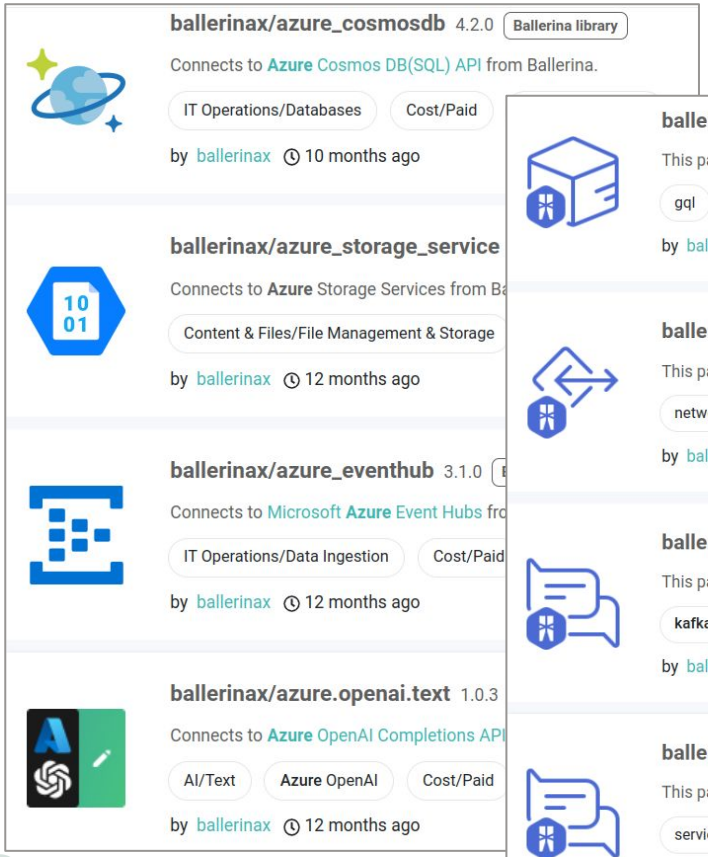
```
xml a = x.<items>;
```

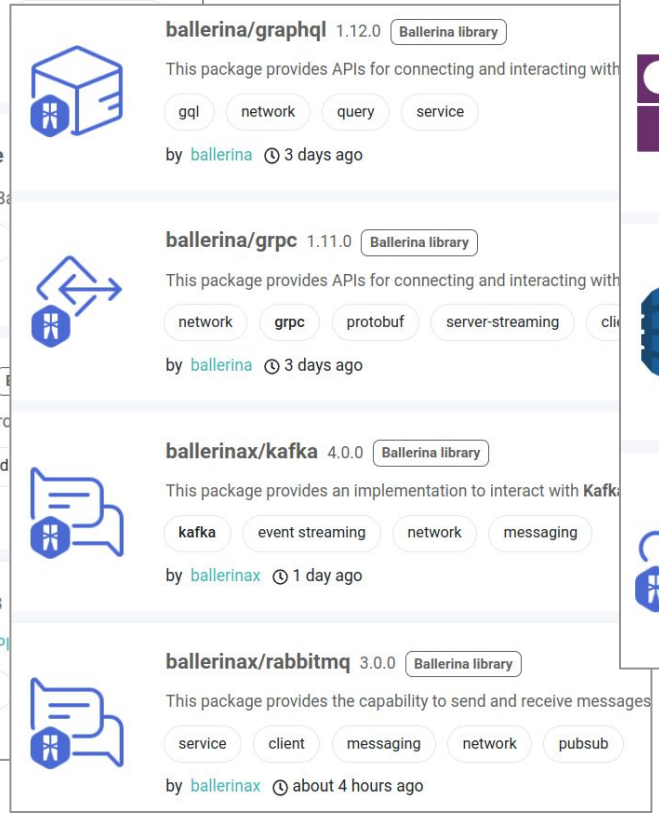
```
xml g = x/<book>[0];
```

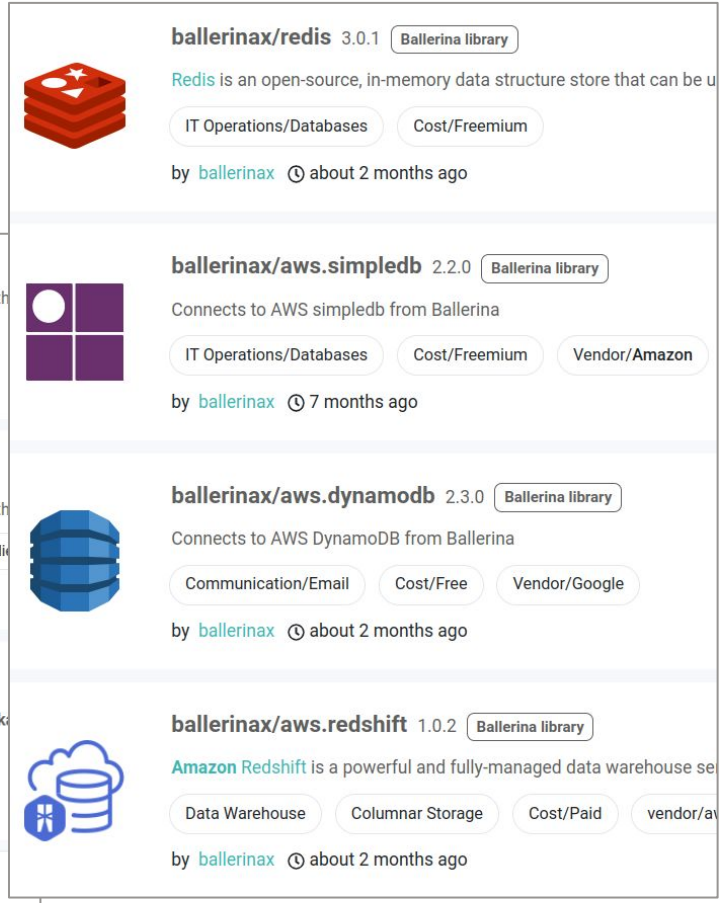
```
xml d = x/<planner|pen>;
```

```
json firstUserName = check users[0].user.name;
```

# Connect with any system



# Deployment flexibility

## Docker

```
[container.image]
repository="wso2inc"
name="hello"
tag="v0.1.0"
```

```
$ bal build --cloud="docker"
```

## Kubernetes

```
[cloud.deployment]
min_memory="100Mi"
max_memory="256Mi"
min_cpu="1000m"
max_cpu="1500m"

[cloud.deployment.autoscaling]
min_replicas=2
max_replicas=5
cpu=50
memory=80
```

```
$ bal build --cloud="k8s"
```

## AWS Lambda

```
import ballerina/io;
import ballerinax/aws.lambda;

// The `@lambda:Function` annotation marks a function
@lambda:Function
public function echo(lambda:Context ctx, json input)
    io:println(input.toJsonString());
    return input;
}
```

# B2B integrations with EDI

```
HDR*1201*2008-01-01~
ORG*P120*ABC Store*67, Park road:Brisbane:Australia*01 678 8908^04 732 3721~
ITM*A-250*12*~
ITM*A-45*100*Kevin:Urgent~
ITM*D-10*58*Smith:REF 10053^Steven:Pre orders~
...
```

- Used since early 70s
- Heavily adopted in transportation, insurance, finance, retail, and supply chain sectors.
- Comprehensive standardizations for specific industry sectors

# EDI with Ballerina *(in 3 steps)*

1. Prepare the schema:

```
"name" : "PartnerOrder",
"delimiters" : {"segment" : "~", "field" : "*", "component" : ":",
"preserveEmptyFields": true,
"includeSegmentCode": true,
"segments" : [
    {
        "code": "HDR",
        "tag" : "header",
        "fields" : [{"tag" : "code"}, {"tag" : "orderId"}, {"tag"
    },
    {
        "code": "ORG",
        "tag" : "organization",
        "fields" : [
            {"tag" : "code"},
            {"tag" : "partnerCode"},
            {"tag" : "name"},
            {"tag" : "address", "components" : [{"tag" : "streetAd
            {"tag" : "contact", "repeat" : true}]
    },
    {
        "code": "ITM",
        "tag" : "items",
        "maxOccurances" : -1,
        "truncatable" : false,
        "fields" : [
            {"tag" : "code"},
            {"tag" : "item"},
```

2. Generate code

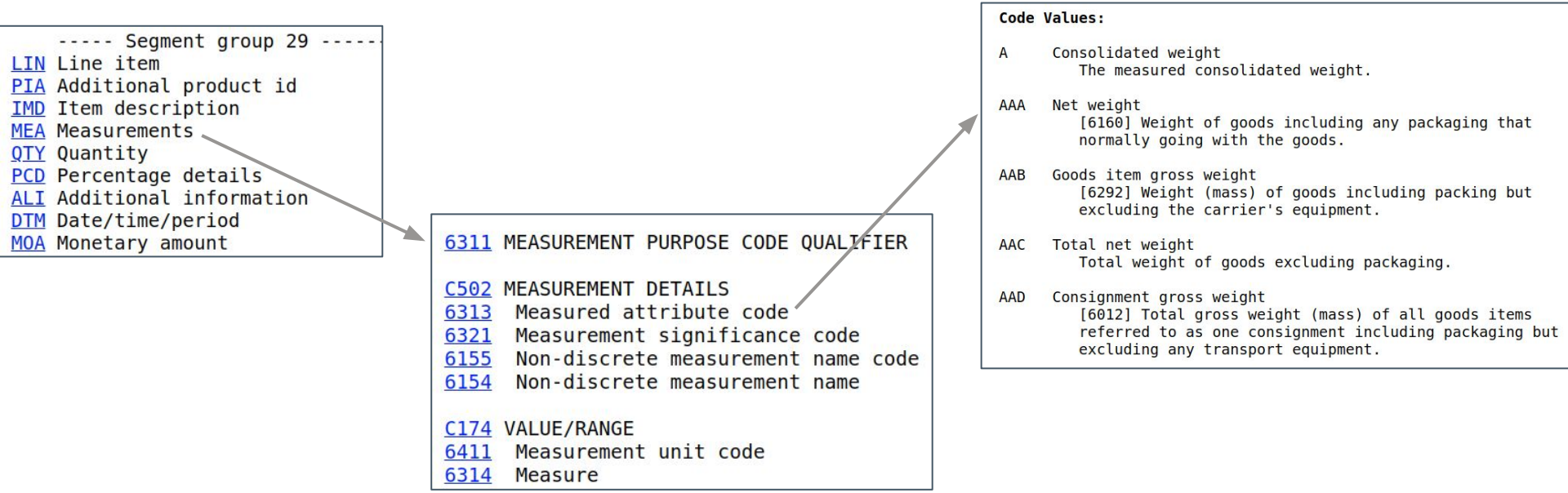
```
$ bal edi codegen -i porder.json -o border.bal
```

3. Process EDI messages:

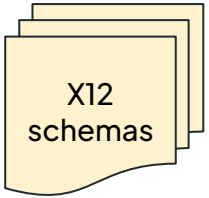
```
string ediText = check io:fileReadString("resources/order1.edi");
POrder pOrder = check fromEdiString(ediText);
io:println(pOrder.toJsonString());
```

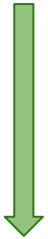
# Working with B2B EDI standards

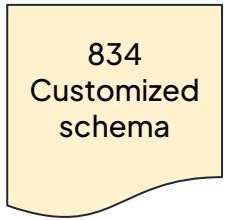
- Standardizations for B2B messages (X12, EDIFACT, HL7, etc)
  - X12 850 - Purchase order
  - X12 810 - Invoice
  - X12 837 - Healthcare claim



# Working with B2B EDI standards





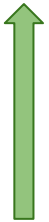






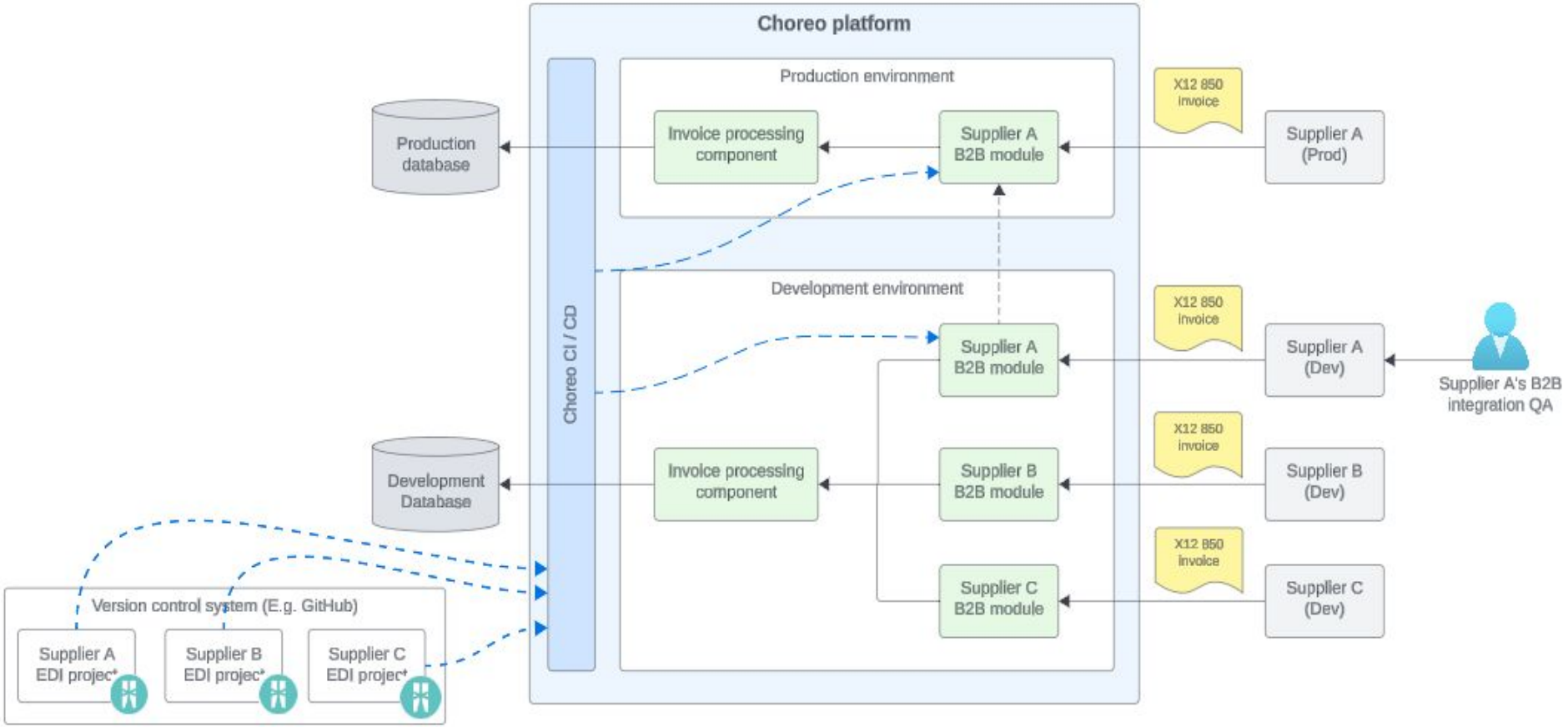


```
string ediText = check io:fileReadString("resources/834_1.edi");
m834:EnrollmentData enrollment =
        check abc:fromEdiString(ediText, abc:EDI_834).ensureType();
io:println(enrollment);
```



```
public type EDI_834_Benefit_Enrollment_and_Maintenance record {|
   Transaction_Set_Header_Type Transaction_Set_Header;
   Beginning_Segment_Type Beginning_Segment;
   Reference_Information_Type[] Reference_Information = [];
   Date_or_Time_or_Period_Type[] Date_or_Time_or_Period = [];
   Monetary_Amount_Information_Type[] Monetary_Amount_Information
   Quantity_Information_Type[] Quantity_Information = [];
   A_1000_Loop_GType[] A_1000_Loop = [];
   A_2000_Loop_GType[] A_2000_Loop = [];
   Transaction_Set_Trailer_Type Transaction_Set_Trailer;
|};
```

# B2B Integration architecture with Choreo



# Question Time!



# Thank You!