Updating the current Jolie microservices based DMS solution to include electronic invoicing

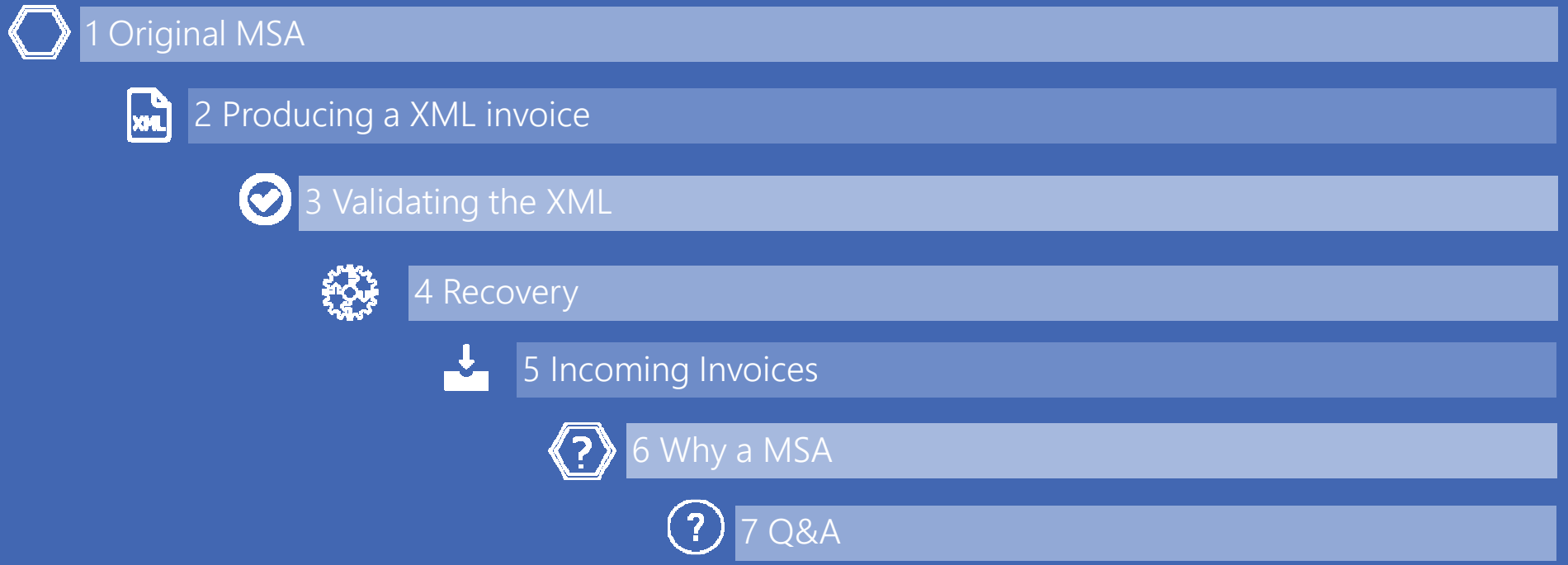
Balint Maschio
Monrif S.p.A

Microservices 2019

monrif group

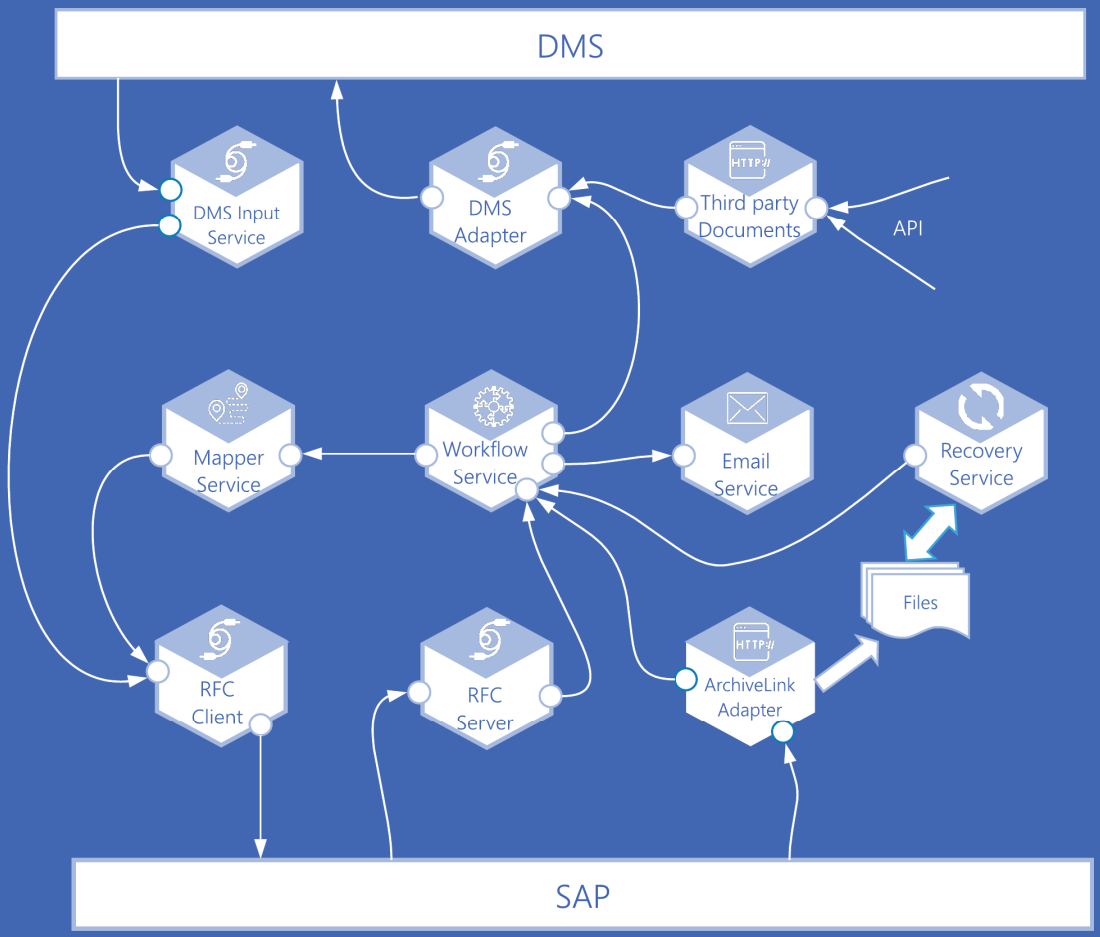
INTRODUCTION

Presentation agenda



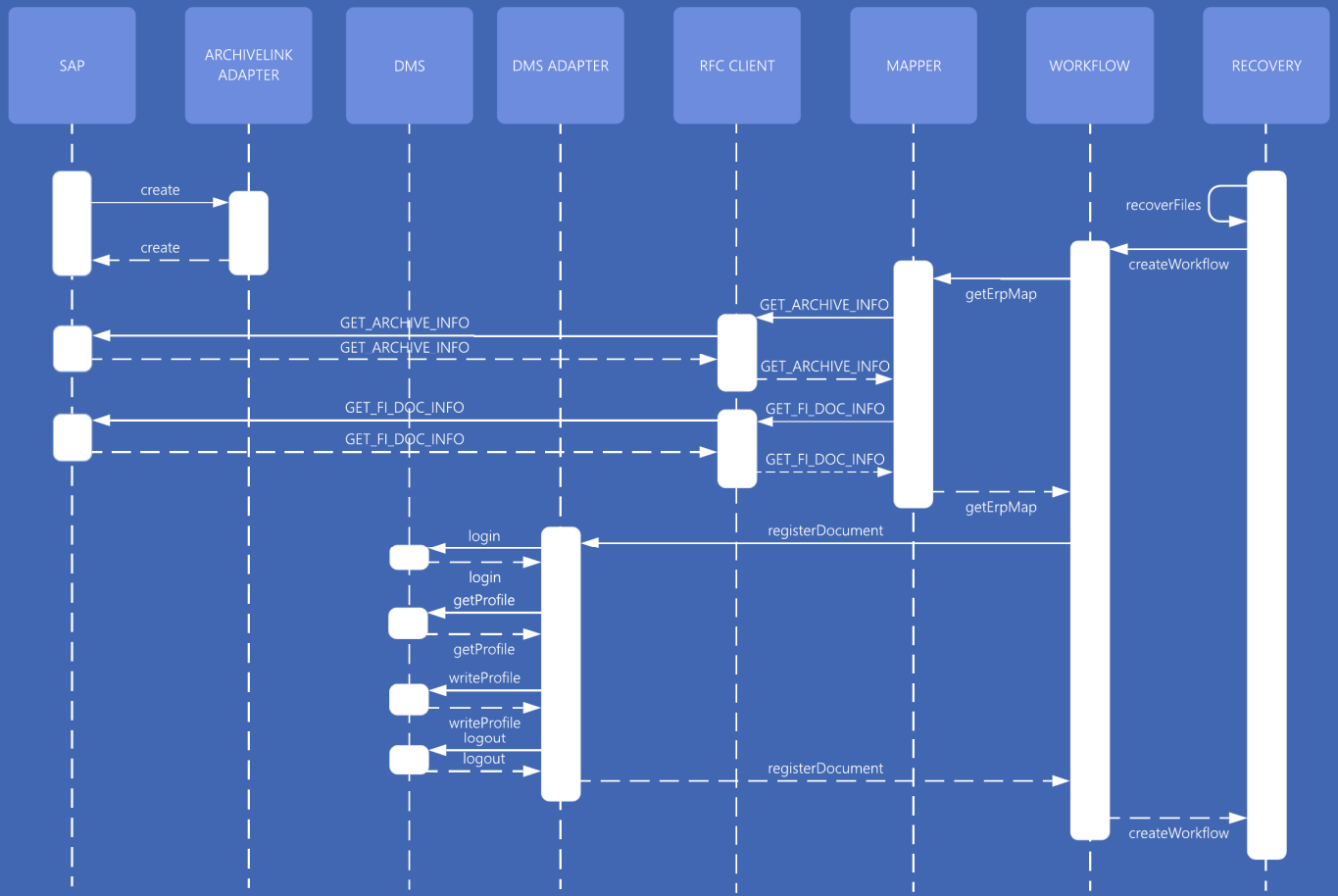
ORIGINAL MSA

Architectural Diagram



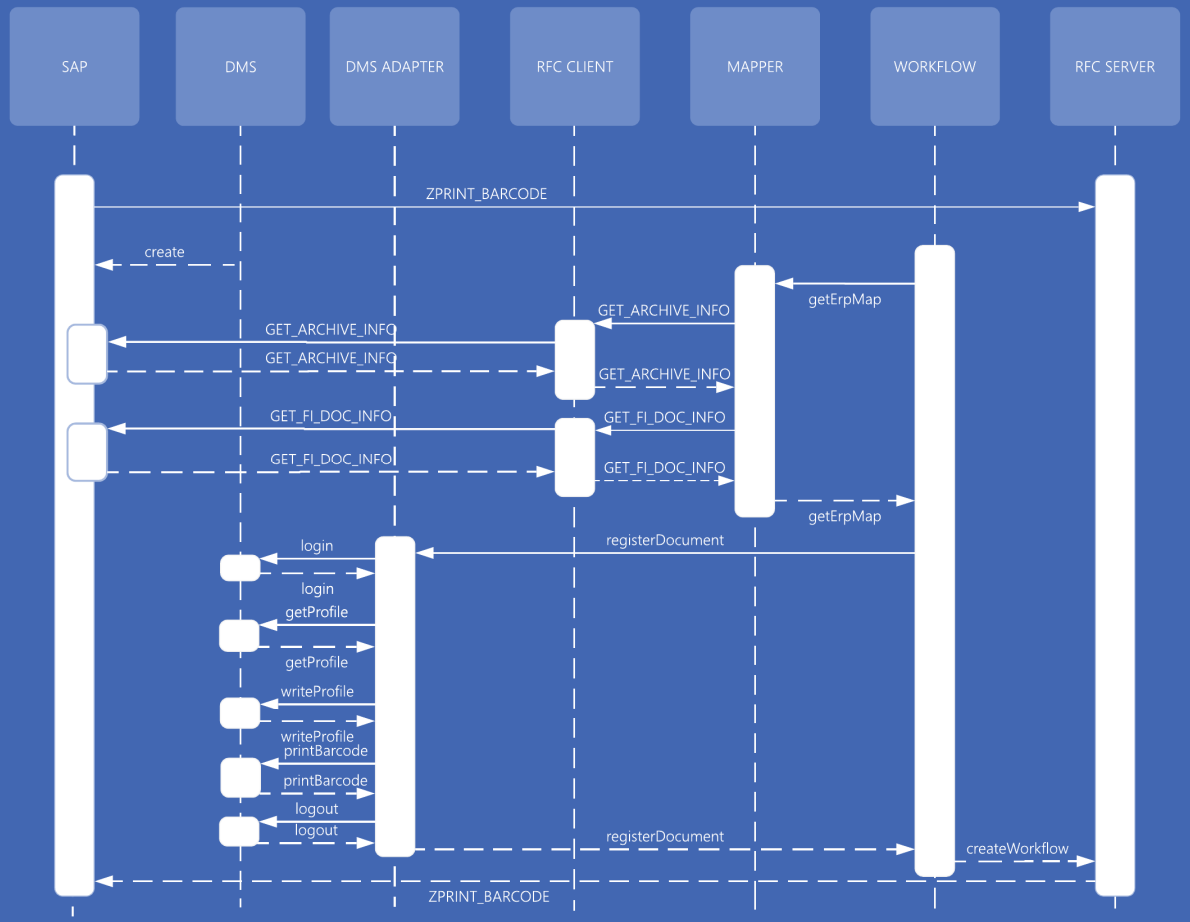
ORIGINAL MSA

Active Cycle



ORIGINAL MSA

Passive Cycle (ERLA)



PRODUCING AN XML INVOICE

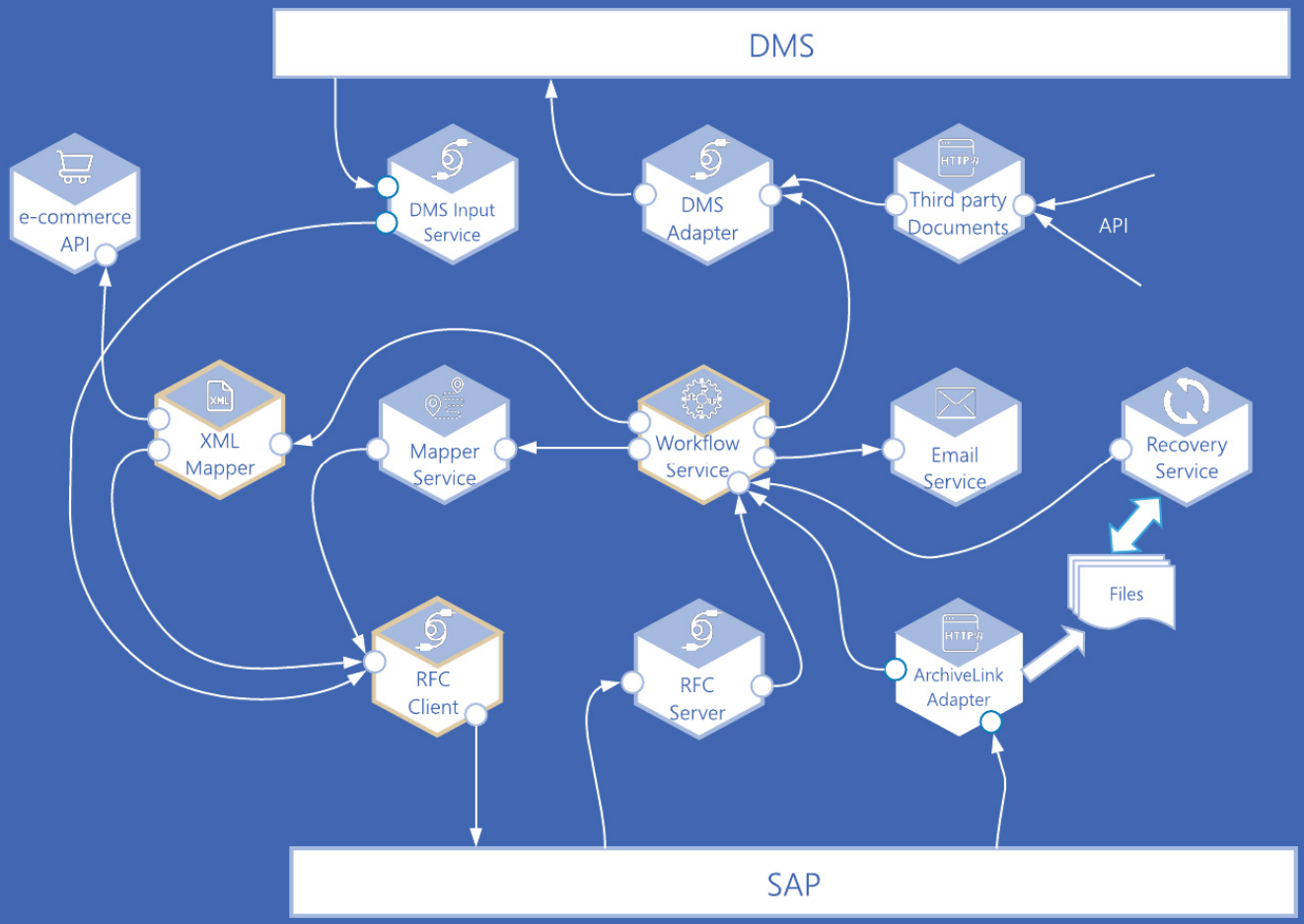
Introducing active XML invoicing

The project had the following limitations:

1. Our current SAP installation , due to packages restriction, does not allow the implementation of the standard electronic invoicing package
2. We wanted to maintain the current invoicing registration operation as unchanged as possible.
3. Some of the data used in the new invoice workflow were not easily transferable into SAP
4. We wanted to preserve as much as possible of the original microservices architecture
5. The new implementation had to be capable to adapt to new XML documents

PRODUCING AN XML INVOICE

Architectural Diagram



VALIDATING THE XML

Using Jolie runtime typecheck to validate the XML document

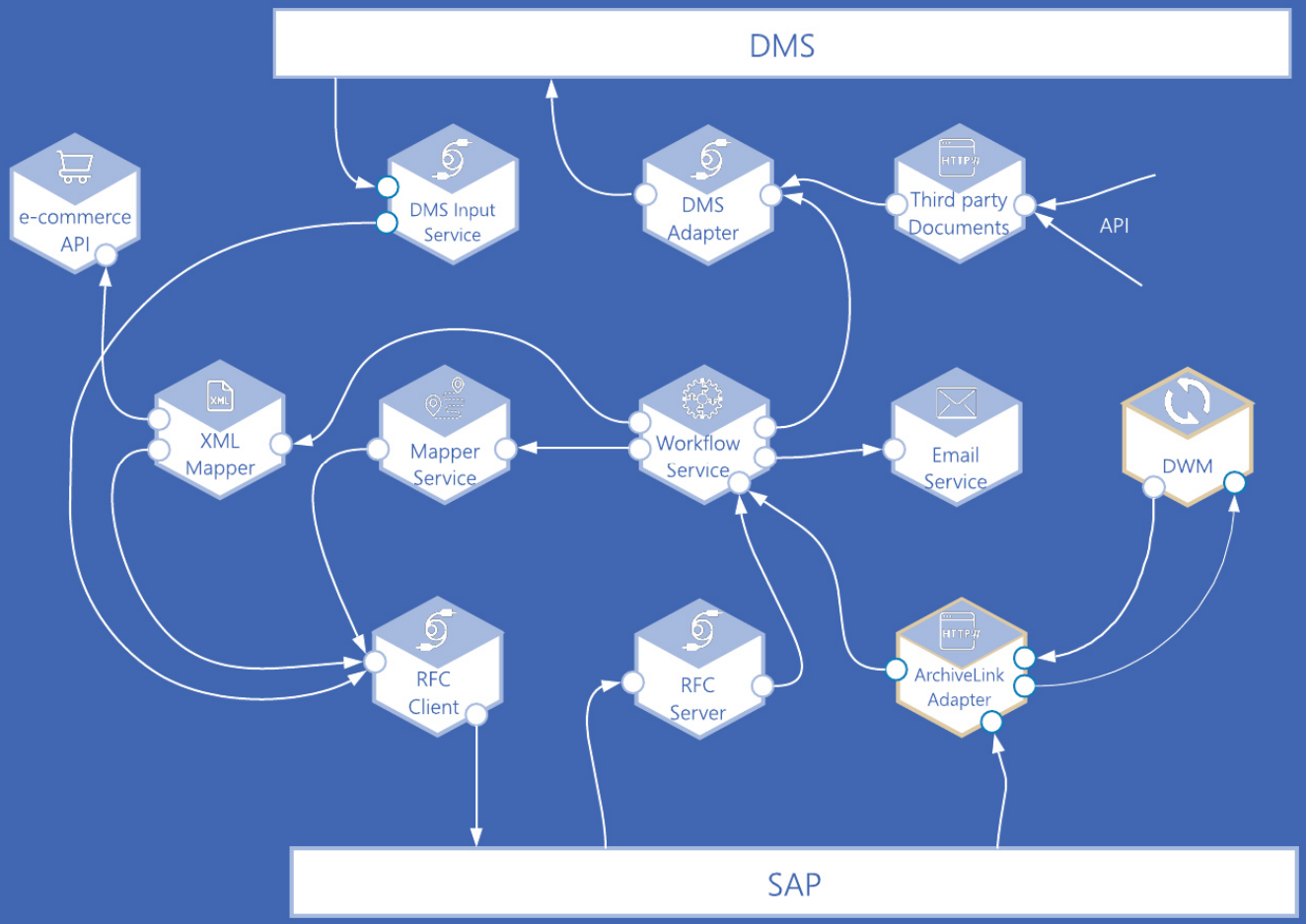
1. Using a conversion tool we have generated a Jolie type representing the electronic invoice type
2. We can use now the generate type as message for a check operation that will implemented by an internal service
3. The check operation is called on the end of mapping operation within a defined scope and with the TypeMismatch compensation
4. The check operation can be expanded to implement other document checks

```
interface CheckFatturaTypeInterface {
    RequestResponse:
        checkFattura( FatturaElettronicaType )( void ) throws DocumentCheckError (DocumentCheckError)
}

service CheckService {
    Interfaces: CheckFatturaTypeInterface
    main {
            checkFattura( request )() {
                //other check

            }
    }
}
```

```
            scope( check ) {
                install( TypeMismatch =>

                        requestIdentfyErrorSpit = check.TypeMismatch;
                        requestIdentfyErrorSpit.regex="#";
                        split@StringUtils(requestIdentfyErrorSpit)(responseIdentfyErrorSpit);
                        requestIdentfyErrorField =responseIdentfyErrorSpit.result[1];
                        requestIdentfyErrorField.regex="\\.";
                        split@StringUtils(requestIdentfyErrorField)(responseIdentfyErrorField);
                        println@Console(#responseIdentfyErrorField.result)();
                        lengthResults =  #responseIdentfyErrorField.result -1 ;
                        fault.fieldName = responseIdentfyErrorField.result[lengthResults];
                        fault.invoiceNumber = invoice_doc_number;
                        fault.compCode = comp_code;
                        fault.fiscalYear = fiscal_year;
                        throw( FatturaTypeError, fault )
                    );
                    install (DocumentCheckError=>
                      fault.fieldName = check.DocumentCheckError.fieldName;
                      fault.errorCode = check.DocumentCheckError.errorCode;
                      fault.invoiceNumber = invoice_doc_number;
                      fault.compCode = comp_code;
                      fault.fiscalYear = fiscal_year;
                      throw( FatturaTypeError, fault )
                    );
                    checkFattura@CheckService( invoice )()
                }
```

RECOVERY

Architectural Diagram



RECOVERY

Replacing the recovery service

In the architectural diagram I have shown a recovery service based on the storing of the documents on the file system

1. This approach limits the “recovery” to “all or nothing”
2. We need to have a more state machine approach to separate in different states any document workflow
3. The microservice that “owns” the document process registers the document to the DocumentWorkflowManager (italianaSoftware).
4. The behavior for each state is defined in a separated Jolie microservice that is embedded by the original process owner
5. All of the “state” microservices need to implement a single operation that is called by DWM

```
with( meta ) {
    .workflow_name = "ArxivarDocument";
    .automatic_deletion_of_history = true;
    with( .status_list[ 0 ] ) {
        .label = "Recovery";
        .branch= "main";
        .description = "status0 description";
        .next_status = -1;
        .mode = "time_dependent";
        .mode.process.location = locations.Locations.TimeDependentStatusProcessPort;
        with( .mode.scheduling ) {
            .second = "10";
            .minute = "*";
            .hour = "*";
            .dayOfMonth = "1/1";
            .month = "*";
            .dayOfWeek = "?"
        }
    }
};
```

INCOMMING INVOICE

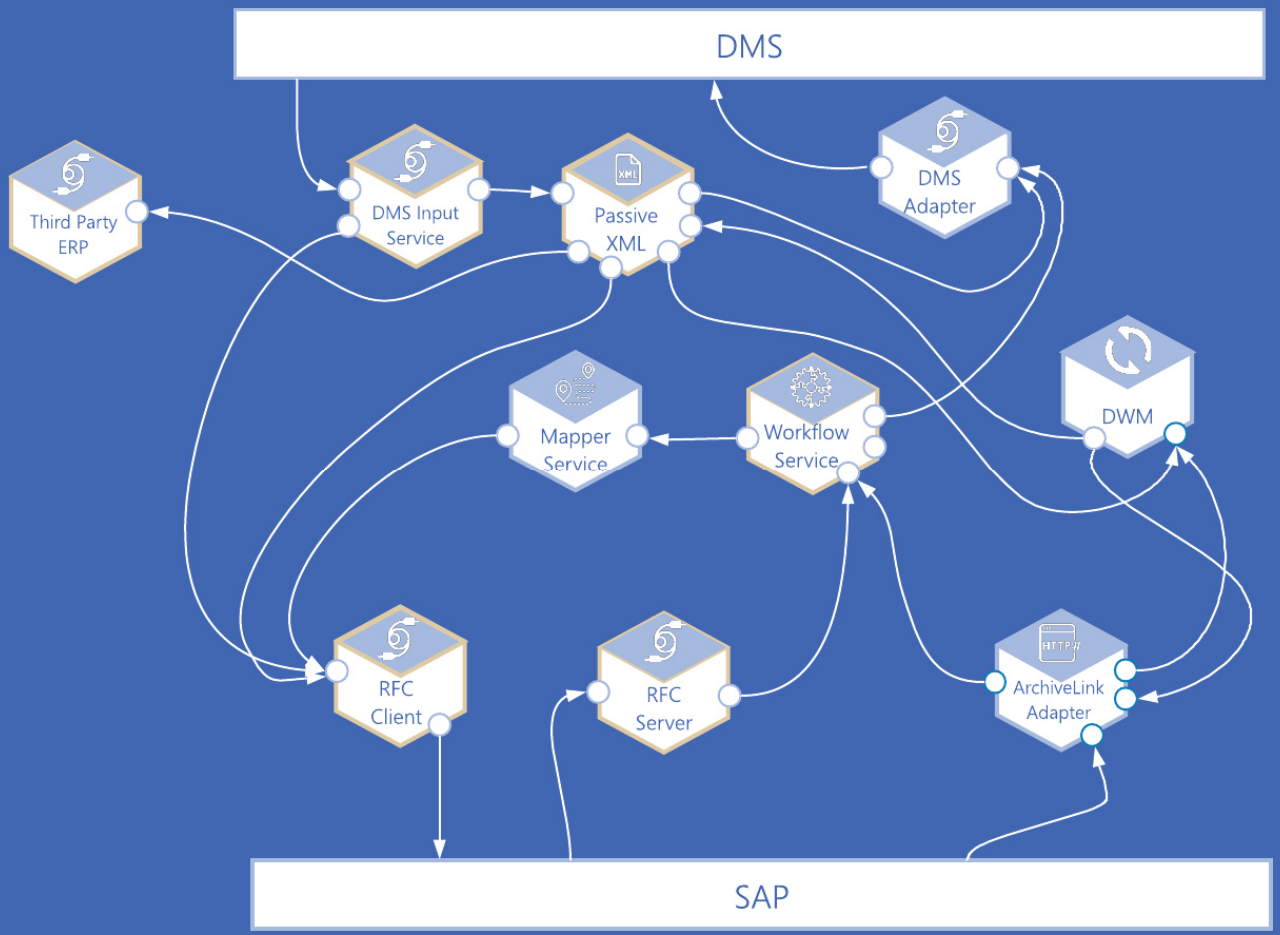
Processing incoming XML invoice

The project had the following objectives:

1. We needed to introduce the early archiving approach
2. We needed to extract and validate data from the XML document to created e pre registered invoice in SAP
3. The invoice document had to visible inside SAP via ArchiveLink
4. We needed to direct some of the invoices to an other group ERP
5. Once the operator concluded the registration the DMS profile had to be enriched with the SAP data.

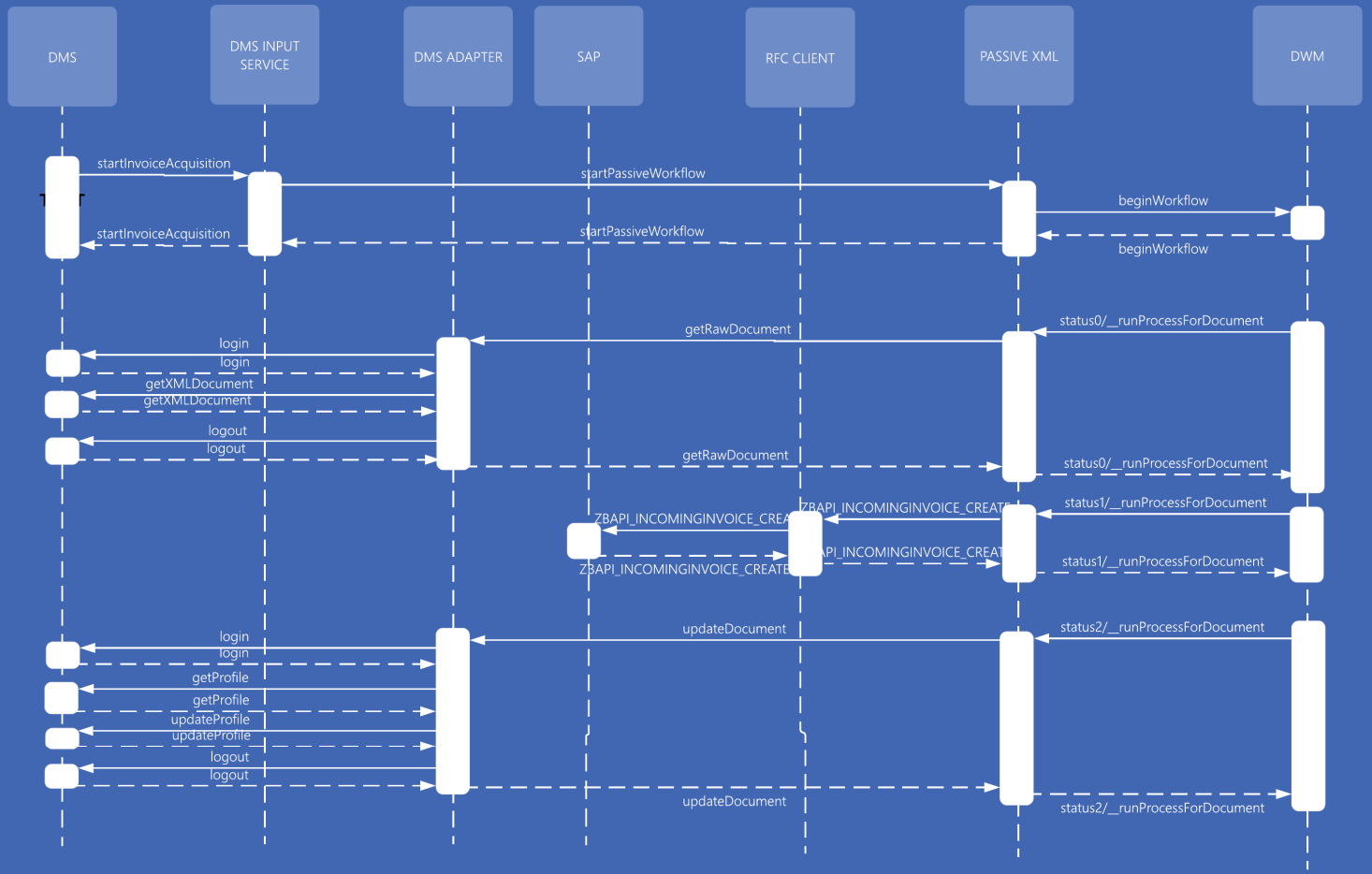
INCOMCING INVOICE

Architectural Diagram



INCOMING INVOICE

Passive Cycle (EALR)



Putting things together

The project has shown that

1. MSA has the potential to be an effective paradigm for system integration that to its intact modularity
2. We have been able to update the previous implementation by refactoring some of the microservices and introducing some specialized microservices
3. The project carried out almost entirely by internal resources
4. The current architecture has maintained the desired level of flexibility , we feel that will be perfectible capable to be adapted for other xml document management

Yet when working with a MSA

1. There is a risk to over complicate the solution by adding unnecessary complexity
2. Deciding the boundary of a single microservice is not always a straight forward as it may look
3. Would be useful to have a MSA normalization strategy/tool

Q&A

Any questions?