What is and Why federation in UIV?

- It is common to have multiple inventory systems which are specialized to store certain kinds of inventories eg: Physical Inventory.
- Federation is a way to fetch data from multiple inventory systems into a single inventory system so that the north bound system consuming the inventory does not know about the complexities on various inventory systems and interfaces with only one inventory system.
- UIV provides 2 modes of federation

Load external inventory data in UIV database using D&R framework

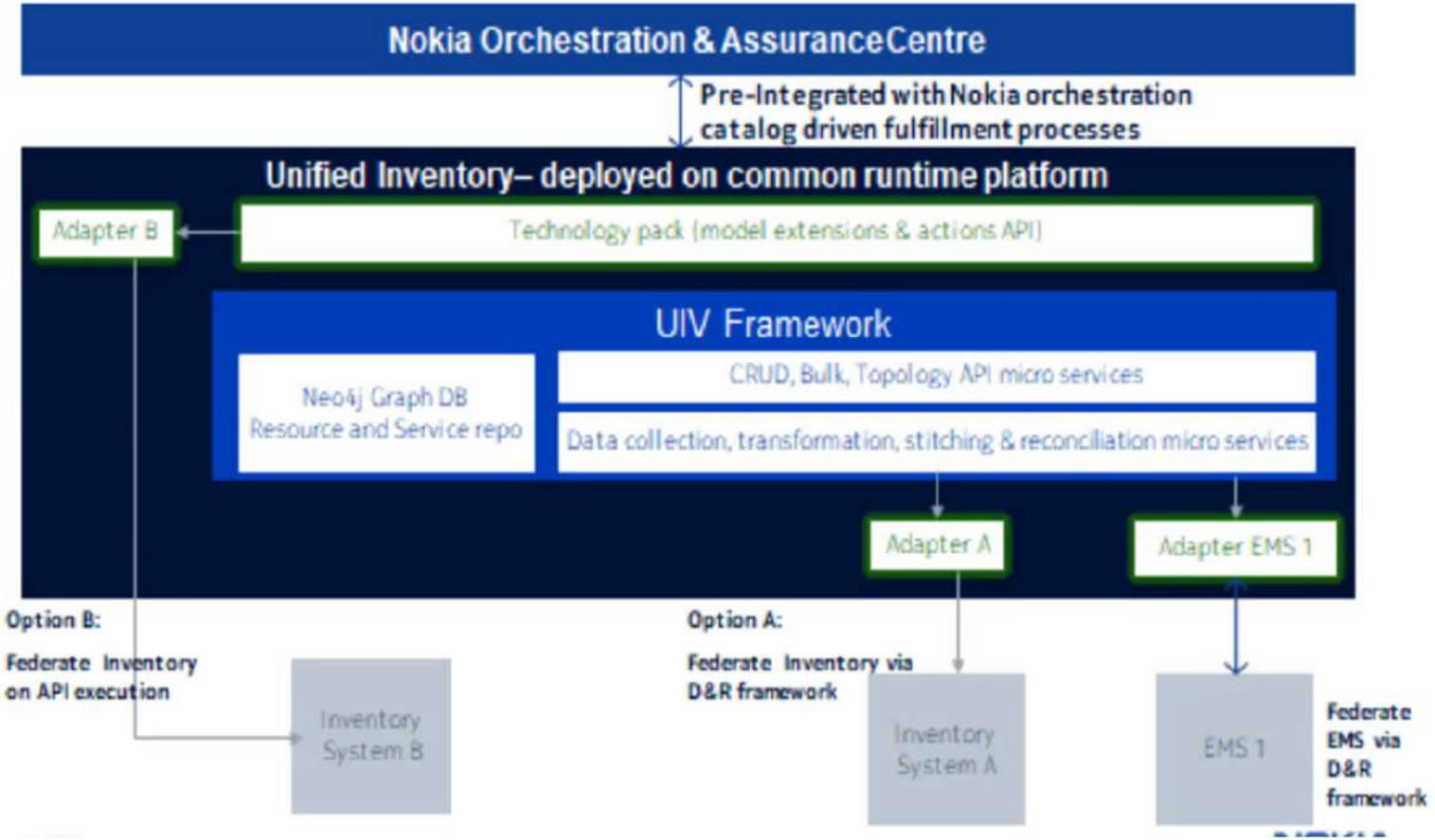
Fetch data on demand from external inventory systems and do not store in UIV database – API federation

What is and Why federation in UIV?

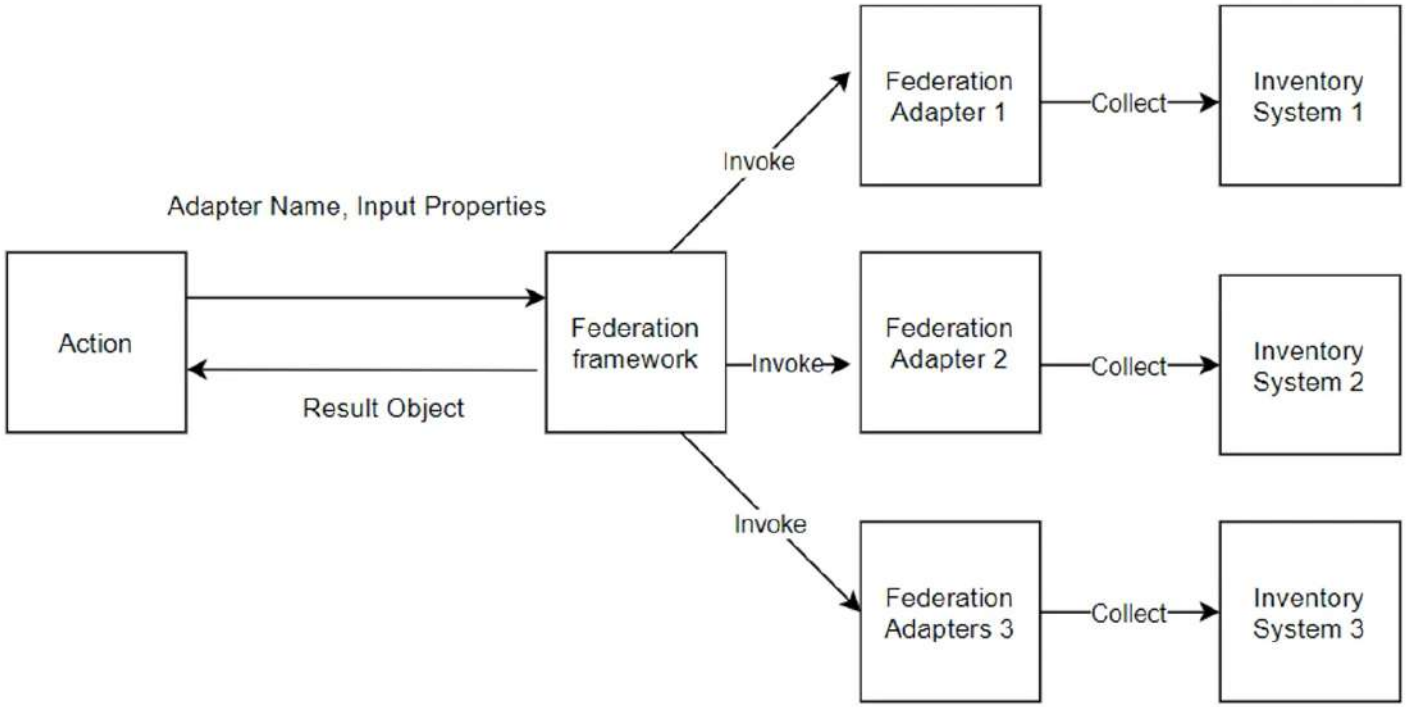
- It is common to have multiple inventory systems which are specialized to store certain kinds of inventories eg: Physical Inventory.
- Federation is a way to fetch data from multiple inventory systems into a single inventory system so that the north bound system consuming the inventory does not know about the complexities on various inventory systems and interfaces with only one inventory system.
- UIV provides 2 modes of federation

Load external inventory data in UIV database using D&R framework

Fetch data on demand from external inventory systems and do not store in UIV database – API federation



API Federation Architecture



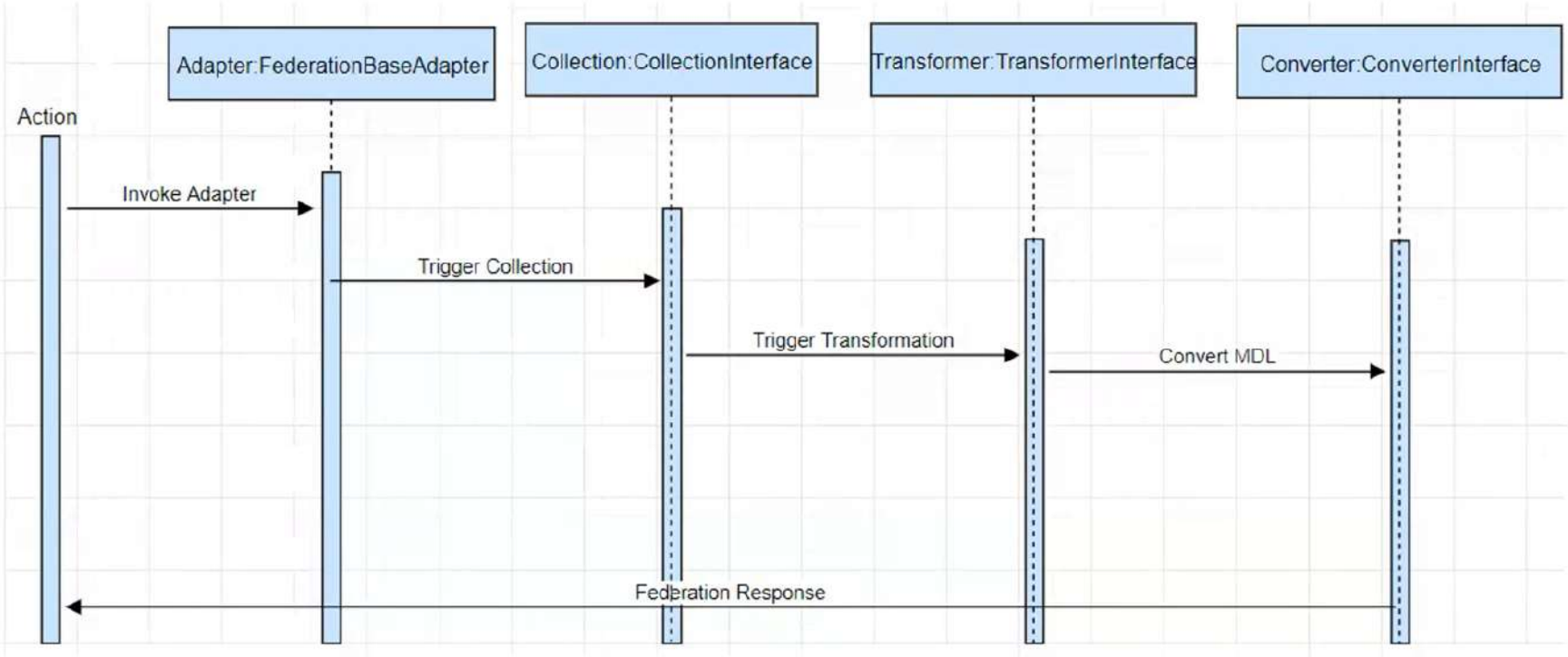
Federation Adapter Framework

- Federation Adapter framework is an extension of the UIV adapter framework.
- It continues the concept of Collection and Transformation pipeline stages from the UIV adapter framework.
- It replaces Streamer stage in UIV adapter framework with a Converter stage.
- Converter is the pipeline stage which converts the MDL created in Transformation stage into UIV technology pack/ base model defined objects

Federation Adapter Framework

- Federation Adapter framework is an extension of the UIV adapter framework.
- It continues the concept of Collection and Transformation pipeline stages from the UIV adapter framework.
- It replaces Streamer stage in UIV adapter framework with a Converter stage.
- Converter is the pipeline stage which converts the MDL created in Transformation stage into UIV technology pack/ base model defined objects

Federation adapter pipeline stages



Creating a Federation Adapter project

- Federation adapter can be created using uiv-cli

  -Windows

  java –jar uiv-cli -a

  -Linux

  ./uiv-cli –a

  Select option 4 in the multiple select menu,

```
Adapter Types
1. Floe Adapter
2. Event Adapter
3. Stitching Adapter
4. Federation Adapter
Select one of the Adapter type:
```

- Input the required maven parameters when prompted,

Prompt	Sample Values
Enter groupId	com.nokia This is the groupId of the federation adapter maven project
Enter artifactId	sure This is the artifactId of the federation adapter maven project
Enter version	21..0.0-SNAPSHOT This is the version of the federation adapter maven project
Enter adapterName	Sure This is the name of the adapter that is being created

Creating a Federation Adapter project

- Federation adapter can be created using uiv-cli

  -Windows

  java –jar uiv-cli -a

  -Linux

  ./uiv-cli –a

  Select option 4 in the multiple select menu,

```
Adapter Types
1. Floe Adapter
2. Event Adapter
3. Stitching Adapter
4. Federation Adapter
Select one of the Adapter type:
```

- Input the required maven parameters when prompted,

Prompt	Sample Values
Enter groupId	com.nokia This is the groupId of the federation adapter maven project
Enter artifactId	sure This is the artifactId of the federation adapter maven project
Enter version	21..0.0-SNAPSHOT This is the version of the federation adapter maven project
Enter adapterName	Sure This is the name of the adapter that is being created

Adapter Entry Point

- The Entry point class is generated when Federation adapter is created using uiv-cli
- The Entry point class must extend from class:

```
com.nokia.oss.sure.adapter.intf.FederationBaseAdapter;
```

- The Entry Point class must be a spring Component

```
@Component
```

- The Entry point class must have the spring profile "FederationAdapters"

```
@Profile("FederationAdapters")
```

Sample Entry Point Class

```java
package com.nokia.adapter;

import org.springframework.stereotype.Component;
import org.springframework.context.annotation.Profile;


import com.nokia.nsw.uiv.adapterworkflow.metadata.WorkflowData;
import com.nokia.oss.sure.adapter.intf.FederationBaseAdapter;


@Component
@Profile("FederationAdapters")
public class Sure extends FederationBaseAdapter{

        @Override
        public void execute(WorkflowData workflowData) throws Exception {
                //Entry point for the adapter
                super.execute(workflowData);
        }
}
```

Parameters passed to Entry Point

- UIV adapter framework used the SEED features to pass required parameters into the adapter.
- Federation adapter framework passes the required parameters into the federation adapters using the WorkflowData object.
- The input parameters can be accessed by Federation Adapters collector, transformer and converter using,

```
Map <String,Object> commarea  = workflowData.getCommarea();
```

Collector and Transformer

- Federation Adapter Collector and Transformer are the borrowed from the UIV adapter framework, all functionality provided by UIV adapter framework for collector and transformer are available.

Converter

- Converter is the pipeline stage which converts the MDL created in Transformation stage into UIV technology pack/ base model defined objects
- To aid in this conversion Federation adapter framework has provided a generic implementation in class,

```
com.nokia.nsw.uiv.federation.converter.FederationNetworkEntityConverter;
```

Modifier and Type	Method and Description
`static java.util.Set<Neo4jDomainObject>`	`getNetworkEntityObject(java.lang.String ne)`

- The getNetworkEntityObject method takes the MDL file as an input and provides a Set of Neo4jDomainObejcts.
- The result of the converter phase has to be set int the WorkFlowData Objects commArea map with key *"federationAdapterResponse"*

```
workflowData.getCommarea().put("federationAdapterResponse", entities);
```

Sample Converter

```
@Profile("FederationAdapters")
public class SureConverter implements ConverterInterface {


    public Set<Neo4jDomainObject> convert(WorkflowData workflowData) throws Exception{
        log.info("Inside SureConverter :: convert");
        try {
            String sourceFile =  workflowData.getFile().getAbsolutePath();
            String fileString = decompressGzip(sourceFile);
            Set<Neo4jDomainObject> entities = FederationNetworkEntityConverter.getNetworkEntityObject(fileString);
            log.info("SureConverter :: convert entities count=" + entities.size());
            workflowData.getCommarea().put("federationAdapterResponse", entities);
            return entities;

        } catch (IOException e) {
            log.error("SureConverter error while processing." + e.getMessage());
            throw new Exception(e.getMessage(), e);
        }
    }

    public static String decompressGzip(String sourceFile) throws IOException {
        Path source = Paths.get(sourceFile);
        if(sourceFile.endsWith(".gz")) {
            try(GZIPInputStream gis = new GZIPInputStream(new FileInputStream(source.toFile()))) {
                return IOUtils.toString(gis, StandardCharsets.UTF_8);
            }
        }else {
            return new String(Files.readAllBytes(source), StandardCharsets.UTF_8);
        }
    }
}
```

Federation Framework

- Federation framework has been provided so that actions can invoke federation adapters.
- The Federation Framework contains a Spring Component called FederationAdapterInvoker which can be used to invoke federation adapters,

```
com.nokia.nsw.uiv.Federation.invoker.FederationAdapterInvoker;
```

Modifier and Type	Method and Description
`java.lang.Object`	`invoke(java.lang.String adapter, java.util.Map<java.lang.String,java.lang.Object> request)`

ParameterName	Description
String adapter	is the name of the Adapter to be invoked
Map<String,Object> request	is a map holding the input parameters to the adapter which are passed via the WorkflowData object

Federation Framework

- Federation framework has been provided so that actions can invoke federation adapters.
- The Federation Framework contains a Spring Component called FederationAdapterInvoker which can be used to invoke federation adapters,

```
com.nokia.nsw.uiv.Federation.invoker.FederationAdapterInvoker;
```

Modifier and Type	Method and Description
`java.lang.Object`	`invoke(java.lang.String adapter, java.util.Map<java.lang.String,java.lang.Object> request)`

ParameterName	Description
String adapter	is the name of the Adapter to be invoked
Map<String,Object> request	is a map holding the input parameters to the adapter which are passed via the WorkflowData object

Sample Action

```java
@Action
public class DemoFederation implements HttpAction{

    @Autowired
    FederationAdapterInvoker adapterInvoker;

    @Override
    public Class getActionClass() {
        return FederationInput.class;
    }

    @Override
    public Object doPost(ActionContext actionContext) throws Exception
    {
        Map<String,Object> params = new HashMap<>();
        FederationInput input = (FederationInput)actionContext.getObject();
        params.put("federationRequest", input.federationRequest);
        Set<Neo4jDomainObject> obj  = (Set<Neo4jDomainObject>) adapterInvoker.invoke("Sure", params);
        return obj;
    }
}
```

```java
public class FederationInput {
    public String federationRequest;
}
```