|  |  |
| --- | --- |
|  | |
|  | |
|  | |
| B2B Core Modules – Greenhouse application sample | |
| WP 200 | |
| Project Acronym & Number: | FIspace – 604 123 |
| Project Title: | FIspace: Future Internet Business Collaboration  Networks in Agri-Food, Transport and Logistics |
| Funding Scheme: | Collaborative Project - Large-scale Integrated Project (IP) |
| Latest version of Annex 1: | 2013-10-03 |
| Start date of the project: | 01.04.2013 |
| Duration: | 24 |
| Status: | Draft |
| Editor: | Said Rahma (ATOS) |
| Contributors (to the “R” part of the deliverable[[1]](#footnote-2); ordered by project partner) | **ATOS**: Said Rahma Rodriguez  **IBM**: Fabiana Fournier, Moti Nisenson, Inna Skarbovsky |
| Document Identifier: |  |
| Date: | 30.06.2015 |
| Revision: | 001 |
|  |  |
| Project website address: | <http://www.FIspace.eu> |

The FIspace Project

Leveraging on outcomes of two complementary Phase 1 use case projects (FInest & SmartAgriFood), aim of FIspace is to pioneer towards fundamental changes on how collaborative business networks will work in future. FIspace will develop a multi-domain Business Collaboration Space (short: FIspace) that employs FI technologies for enabling seamless collaboration in open, cross-organizational business networks, establish eight working Experimentation Sites in Europe where Pilot Applications are tested in Early Trials for Agri-Food, Transport & Logistics and prepare for industrial uptake by engaging with players & associations from relevant industry sectors and IT industry.

Project Summary

As a use case project in Phase 2 of the FI PPP, FIspace aims at developing and validating novel Future-Internet-enabled solutions to address the pressing challenges arising in collaborative business networks, focussing on use cases from the Agri-Food, Transport and Logistics industries. FIspace will focus on exploiting, incorporating and validating the Generic Enablers provided by the FI PPP Core Platform with the aim of realising an extensible collaboration service for business networks together with a set of innovative test applications that allow for radical improvements in how networked businesses can work in the future. Those solutions will be demonstrated and tested through early trials on experimentation sites across Europe. The project results will be open to the FI PPP program and the general public, and the pro-active engagement of larger user communities and external solution providers will foster innovation and industrial uptake planned for Phase 3 of the FI PPP.

Project Consortium

|  |  |
| --- | --- |
| DLO; Netherlands | Kühne + Nagel; Switzerland |
| ATB Bremen; Germany | University Duisburg Essen; Germany |
| IBM; Israel | ATOS; Spain |
| KocSistem; Turkey | The Open Group; United Kingdom |
| Aston University; United Kingdom | CentMa; Germany |
| ENoLL; Belgium | iMinds; Belgium |
| KTBL; Germany | Marintek; Norway |
| NKUA; Greece | University Politecnica Madrid; Spain |
| Wageningen University; Netherlands | Arcelik; Turkey |
| PlusFresc; Spain | EuroPoolSystem; Germany |
| FloriCode; Netherlands | GS1 Germany; Germany |
| Kverneland; Netherlands | Mieloo & Alexander; Netherlands |
| North Sea Container Line; Norway | OPEKEPE; Greece |
| LimeTri; Netherlands | Innovators; Greece |
| BO-MO; Slovenia | CIT; Spain |
| MOBICS; Greece | SDZ; Germany |
| Fraunhofer IML; Germany | Snoopmedia; Germany |
| Q-ray; Netherlands | EECC; Germany |
| FINCONS; Italy | CBT; Spain |

More Information

Harald Sundmaeker (coordinator) e-mail: [sundmaeker@atb-bremen.de](mailto:sundmaeker@atb-bremen.de)

Bert Vermeer (deputy coordinator) e-mail: [bert.vermeer@wur.nl](mailto:bert.vermeer@wur.nl)

Project Website Web link: <http://www.fispace.eu/>

Dissemination Level

|  |  |  |
| --- | --- | --- |
| PU | Public |  |
| PP | Restricted to other programme participants (including the Commission Services) | X |
| RE | Restricted to a group specified by the consortium (including the Commission Services) |  |
| CO | Confidential, only for members of the consortium (including the Commission Services) |  |

Change History

|  |  |  |
| --- | --- | --- |
| Version | Notes | Date |
| 001 | Creation of the document B2B Core Modules – Greenhouse application sample | 30.06.2015 |
|  |  |  |
|  |  |  |
|  |  |  |

Abbreviations

|  |  |
| --- | --- |
| AAA | Authentication, Authorisation, and Accounting |
| ACSI | Artifact-Centric Service Interoperation |
| AdvB | Advisory Board |
| AJAX | Asynchronous JavaScript + XML |
| API | Application Programming Interface |
| App | Software Application |
| B2B | Business-to-business |
| B2C | Business-to-Consumer |
| BCM | Business Collaboration Module in FIspace |
| BCO | Business Collaboration Objects in FIspace |
| BE | Business Entities |
| BPPC | Business Process Participant Configuration |
| BSS | Business Support Systems |
| CDR | Charging Detailed Records |
| CEP | Complex Event Processing |
| CSB | Cloud Service Bus |
| CSS | Cascading Style Sheets |
| CSV | Comma-Separated Values |
| D | Deliverable |
| DAO | Data Access Object |
| DB | Database |
| DoW | Description of Work |
| EC | European Commission |
| EDI | Electronic Data Interchange |
| EE | Experimentation Environment |
| e.g. | Exempli gratia = for example |
| EPA | Event Processing Agent |
| EPM | Event Processing Module in FIspace |
| ESB | Enterprise Service Bus |
| EU | European Union |
| FIA | Future Internet Assembly |
| FI-PPP | Future Internet Public Private Partnership |
| FP7 | Framework Programme 7 |
| GA | Grant Agreement |
| GE | Generic Enabler |
| GUI | Graphical User Interface |
| HTML | HyperText Markup Language |
| IaaS | Infrastructure as a Service |
| ICT | Information and Communication Technology |
| IDE | Integrated Development Environment |
| IDM | Identity Management |
| i.e. | id est = that is to say |
| IE | Integration Environment |
| IEC | International Electrotechnical Commission |
| IETF | Internet Engineering Task Force |
| I/O | Input / Output |
| IoT | Internet of Things |
| IP | Intellectual Property |
| IP (protocol) | Internet Protocol |
| IPR | Intellectual Property Rights |
| IPsec | Internet Protocol Security |
| IT | Information Technology |
| ITU | International Telecommunication Union |
| ISO | International Standardization Organisation |
| J2SE | Java 2 Platform, Standard Edition |
| JDK | Java Development Kit |
| JDT | Related to Eclipse Java Development Tools |
| JMX | Java Management Extensions |
| JRE | Java Runtime Environment |
| JS | JavaScript |
| JSON | JavaScript Object Notation |
| JSP | Java Server Page |
| JVM | Java Virtual Machine |
| KPI | Key Performance Indicator |
| LPA | Logistics Planning Application |
| M | Month |
| MTBF | Mean Time Between Failures |
| MVC | Model–View–Controller |
| OASIS | Organization for the Advancement of Structured Information Standards |
| OAuth | Open standard Authentication protocol |
| OMG | Object Management Group |
| OSS | Operational Support Systems |
| P2P | Peer-to-peer |
| PaaS | Platform as a Service |
| PDE | Related to Eclipse Java Development Tools |
| PE | Production Environment |
| PIA | Product Information App |
| PIE | Preliminary Integration Environment |
| PKI | Public Key Infrastructure |
| PM | Person Month |
| POM | Project Object Model (used by maven tools) |
| Proton | IBM Proactive Technology Online |
| QoS | Quality of Service |
| RBAC | Role-Based Access Control |
| RCP | Rich Client Platform |
| REST | Representational State Transfer |
| RFC | Request for Comments |
| RSS | Revenue Sharing System |
| RTD | Research and Technological Development |
| SaaS | Software as a Service |
| SDI | System and Data Integration layer in FIspace |
| SDK | Software Development Kit |
| SME | Small and Medium Sized Enterprise |
| SOA | Service Oriented Architecture |
| SOAP | Simple Object Access Protocol |
| SOA-RM | (OASIS) Reference Model for Service Oriented Architecture |
| SPT | Security, Privacy and Trust Framework |
| SSH | Secure Shell |
| SSL | Secure Sockets Layer |
| SSO | Single Sign On |
| ST | Sub-Task |
| SWT | Standard Widget Toolkit |
| T | Task |
| TCP | Transmission Control Protocol |
| TIC | Tailored Information for Consumers |
| TLS | Transport Layer Security |
| TPM | Transport Planning Module |
| UAA | User Management, Authentication and Authorisation |
| UI | User Interface |
| UML | Unified Modeling Language |
| URI | Universal Resource Identifier |
| URL | Universal Resource Locator |
| USDL | Unified Service Description Language |
| VM | Virtual Machine |
| VPN | Virtual Private Network |
| W3C | World Wide Web Consortium |
| WADL | Web Application Description Language |
| WLAN | Wireless Local Area Network |
| WP | Work Package |
| WS | Web Service |
| WSDL | Web Services Description Language |
| XLS/XLSX | Microsoft Excel file Format |
| XML | eXtensible Markup Language |
| XSD | XML Schema Definition |
|  |  |
|  |  |

Table of Contents

[2.1 Overview 11](#_Toc427230508)

[3.1 Greenhouse scenario introduction 13](#_Toc427230509)

[3.2 Logical collaboration flow between EPM and BCM components 13](#_Toc427230510)

[3.3 Greenhouse sample application description 14](#_Toc427230511)

[3.4 EPM application definitions and flow with BCM 15](#_Toc427230512)

[3.5 BCM application definitions and flow 22](#_Toc427230513)

List of Figures

[Figure 1: Greenhouse sequence diagram 14](#_Toc427230514)

[Figure 2 - EPMGreenhouseSampleTemplate.JSON 15](#_Toc427230515)

[Figure 3: CSB producers definitions for the two customized applications 16](#_Toc427230516)

[Figure 4: Threshold values per business Process Id 11234 16](#_Toc427230517)

[Figure 5: Threshold values per business Process Id 45678 17](#_Toc427230518)

[Figure 6: SensorReadOutOfBoundaries EPA and its derived event, B2BOutOfBoundariesNotification 18](#_Toc427230519)

[Figure 7: CSB consumer definition 18](#_Toc427230520)

[Figure 8 : B2BContextTerminationNotification event structure 20](#_Toc427230521)

[Figure 9: Composite context definition 20](#_Toc427230522)

[Figure 10: Temporal context definition 21](#_Toc427230523)

[Figure 11: BCM – GSM model 22](#_Toc427230524)

[Figure 12: Greenhouse information model 22](#_Toc427230525)

[Figure 14 B2BOutOfBoundariesNotification Data Type 24](#_Toc427230526)

[Figure 15 B2BOutOfBoundariesNotification Event 24](#_Toc427230527)

[Figure 17 GSM Lifecycle 25](file:///D:\FIspace\B2B%20collaboration%20module\external%20documentation%20in%20wiki\FIspace-B2B_Greenhouse_sample%20v0.6-clean.docx#_Toc427230528)

[Figure 16 GreenhouseAdvice Information model 25](#_Toc427230529)

[Figure 18 Root stage for the GSM lifecycle 25](#_Toc427230530)

[Figure 19 RequestAdvice stage with associated task (reached by clicking … button next to Atomic Stage on the stage editor page). 26](#_Toc427230531)

[Figure 20 SendResourceAvailableNotification stage with associated task (reached by clicking … button next to Atomic Stage on the stage editor page). 26](#_Toc427230532)

[Figure 21 HandleAdvice parent stage 27](#_Toc427230533)

[Figure 22 CloseEpmContext stage with associated task 27](#_Toc427230534)

[Figure 23 Setting the data to be sent when invoking the close\_epm\_context external service (reachable by clicking the … button next to Task Type) 27](#_Toc427230535)

# Introduction

The aim of this document is to accompany the “*B2B Core Modules – The manual for business architects*” presented as a user guideline related to the business collaboration activities and provide an example application implementation involving the EPM and BCM components. The example application is built for the greenhouse scenario, executing business processes for monitoring of greenhouse-related activities.

In this document we focus more on the interaction between EPM and BCM modules and understanding logical dependencies between the components, and less on how to implement an application for each component, this is covered in the manual for business architects document.

**Online documentation for B2B Core Modules – The manual for business architects**: <http://dev.fispace.eu/doc/wiki/b2b>

# B2B Core Modules – EPM and BCM collaboration

## Overview

At the heart of the envisaged FIspace platform reside the Business-to-Business Core Modules. The B2B Core ensures that all information and status updates are provided to each involved stakeholder in real-time. The B2B core allows for the creation, management, execution, and monitoring of collaborative business processes in the FIspace platform. The B2B Core consists of two interrelated and complementary components:

* A Collaboration Engine that captures, in the form of Business Entities, the information to be exchanged among collaborating stakeholders along with status and control of the a collaborative business processes. The BCM component is responsible to orchestrate the different processes from different stakeholders and assure the correct sequence of the tasks execution;
* An Event Processing Engine that detects and analyses events coming from activities in the collaborative processes or from IoT devices. The Event Processing Module (EPM) component monitors events and detect situations of interest in real-time, i.e., situations that require appropriate reactions;

The BCM component is responsible to orchestrate the different processes from different stakeholders and assure the correct sequence of the tasks execution. The BCM is based on the entity-centric approach (for more details, please refer to the outcomes of the [ACSI project](http://www.acsi-project.eu/)). This approach relies on the notion of business entities (aka, as (dynamic/business) artefacts). These provide a holistic marriage of data and process, both treated as first-class citizens, as the basic building block for modelling, specifying, and implementing services and business processes. A (business) entity is a key conceptual concept that evolves as it moves through a business (or other) process. An entity type includes both a data schema and a lifecycle schema which are tightly linked. The data schema provides an end-to-end conceptual view of the key data for this entity type. The lifecycle schema of an entity type specifies the different ways that an entity instance might evolve as it moves through the overall process. In FIspace we apply the GSM (Guards, Stages, and Milestones) model to specify the lifecycle schema of the business entities.

The Event Processing Module (EPM) component monitors events and detect situations of interest, i.e. situations that require appropriate reactions. The events sources (aka events producers) can be the actual execution of the collaboration (i.e., the BCM), external systems, or sensors. The EPM processes these events and by applying pattern matching derives situations of interest. Examples of situations of interest can be: Missing documentation at a certain point in time, a sensor reading outside a permitted range, or a delay in a delivery. In general, we can distinct between situations that result from the actual execution of the process or collaboration and situations that result from external events (i.e., events coming from external systems or sensors). The EPM in FIspace supports two types of situation detection capabilities: reactive and proactive. Reactive rules analyse past events and derive situations by applying pattern matching over a single or a set of events over time. Proactive rules, on the other hand, relate to situations that are likely to happen in the (near) future. In general, we refer to proactive event-driven computing as the ability to mitigate or eliminate undesired states, or capitalize on predicted opportunities—in advance. This is accomplished through the online forecasting of future events, the analysis of events coming from many sources, and the application of online decision-making processes. EPM application can be a generic application, suitable for different business processes of the same business process template, or custom application, customized from some generic application created for the specific business processes template by a business architect.

The result of EPM processing, in the form of derived events, or situations, are streamed into the BCM engine for further processing. Further processing might include initiation of business process, update of existing state or termination of the business process, depending on the scenario and application logic. Additionally the BCM might also serve as a source of information for EPM and create events which might be correlated by EPM as part of its application logic.

These two modules are connected each to the other and to the other FIspace components of the platform via the [Component Service Bus (CSB)](http://dev.fispace.eu/doc/wiki/csb).

# Implementation of the Greenhouse scenario

## Greenhouse scenario introduction

Two main business actors which are involved in this scenario: the farmer/greenhouse manager and an advisory/expert system enterprise, which provides advisory services to the greenhouse based on the conditions inside the greenhouse. The idea is that the sensors’ values of the Greenhouse are forwarded to the Greenhouse Farm Management System (FMIS), where they are contextualized, and afterwards they are forwarded to FIspace. In case of events inside the greenhouse (i.e. sensor values detected out of pre-defined boundaries) a request for actions is sent to the advisory system. The end-user receives the actions from the expert system via the respective FIspace app.

## Logical collaboration flow between EPM and BCM components

The logical flow of this scenario is as follows (see Figure 1: Greenhouse sequence diagram): the EPM receives a ***ReceiveSensorValuesRequestMessage.*** This message contains information with different sensor readings, such as temperature, luminosity etc. Due to platform limitations, it is important that the lifecycle is singularly identifiable by just the business process id. This means that a business process should be created for each uniquely identifiable field – in this case, the combination of farmer, farm (farmId), and crop (cropFieldId) should have a unique business process. Thus, processing and state for different crops, even for the same farmer, have different business process ids and do not interfere with each other. EPM validates the readings inside the message versus specific thresholds defined for each reading type. If it detects a violation, it generates a ***B2BOutOfBoundariesNotification*** which is forwarded to the BCM system. BCM initiates a business process per the provided businessProcessId, and requests for advice. Until the advice is received, EPM will not continue additional notifications for out of bounds readings for this business process id, even though it might receive additional input messages where the values exceed the specified thresholds. This is done so that EPM doesn’t flood the advice system with request for advice, it will wait for notification that an action was taken. .Once it receives such notification, in the form of ***B2BContextTerminationNotification*** message, this will be an indication that this particular problem is considered to be taken care of, and therefore if additional ***ReceiveSensorValuesRequestMessage*** for this particular business process id are received which are out of bounds, it should again initiate a process for request for advice.

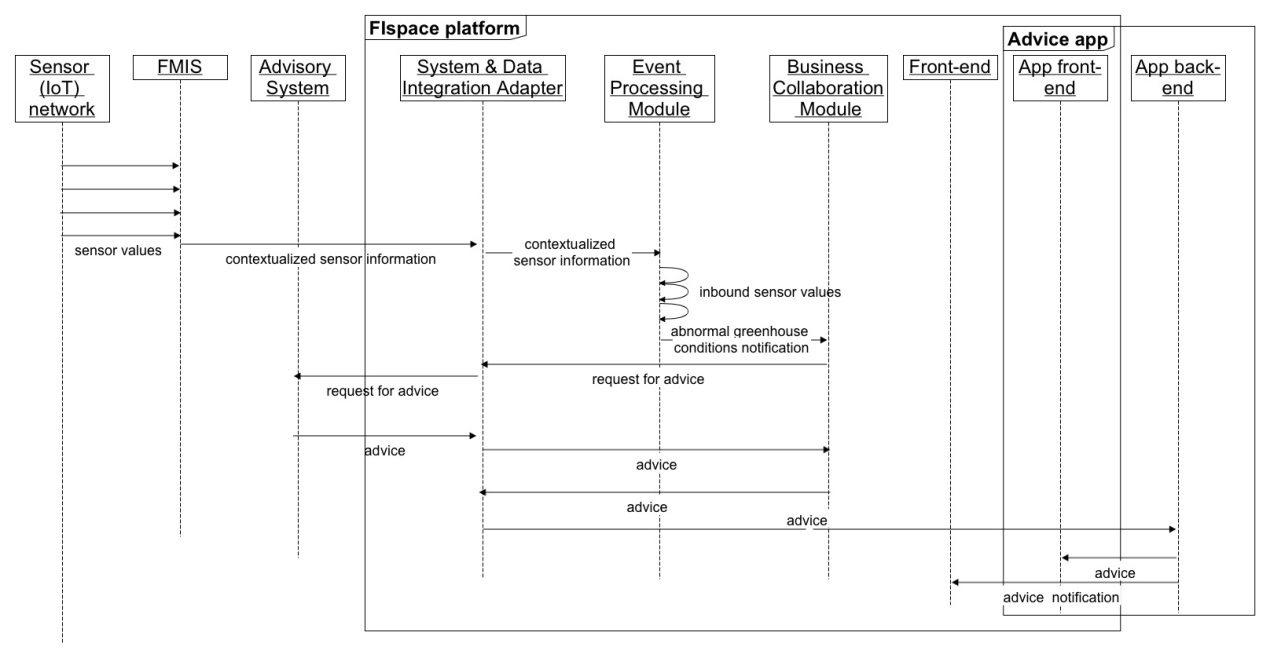


Figure 1: Greenhouse sequence diagram

## Greenhouse sample application description

EPM greenhouse application is defined per business process id. In case of greenhouse, business process id designates a specific farmer, farm, and crop. That means that for multiple crops of the same farmer, there exist multiple application definitions, created from some common template authored by business architect, but customized with specific threshold values matching the needs of each farmer. This is due to current platform limitation in information included in the messages, and not due to EPM limitations. In the future, once this limitation in messages is resolved, a business process can be created per farmer, and the farmId and cropId used in EPM segmentation context to run instance of the app for different crops in parallel. [[2]](#footnote-3)

For EPM we demonstrate two customized greenhouse application definitions for different business process ids (representing farmer+farmId+cropId). ***EPMGreenhouseSample11234.json*** is for business process id 11234. ***EMPGreenhouseSample45678.json*** is for business process id 45678. Those custom applications can be created from a sample template application authored by business architect by providing custom values for parameter thresholds (in this application’s case those are the threshold values for different sensor readings). Please note that those thresholds are different. For example, see the definition of threshold values in a EMPGreenhouseSampleTemplate.json (see Figure 2)

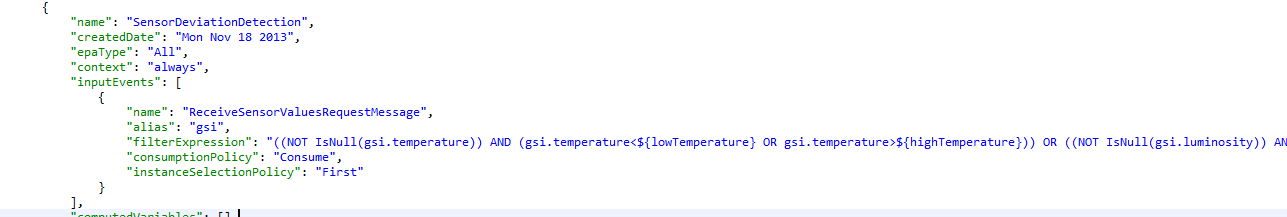


Figure 2 - EPMGreenhouseSampleTemplate.JSON

## EPM application definitions and flow with BCM

Each of the different EPM applications (per different business process ids) will listen on events on different CSB queues. Therefore, each of the applications will receive messages only related to the specific business process id. Figure 3: CSB producers definitions for the two customized applications demonstrates the CSB adapter definitions for the two applications. The first figure demonstrates a queue defined for business process id 11234, the right side demonstrates a queue for business process id 45678.Please not that this information is automatically generated for a business-process specific application during deployment, this is not something provided by the business architect.





Figure 3: CSB producers definitions for the two customized applications

Once the ***ReceiveSensorValuesRequestMessage*** arrives to one of those applications via those CSB queues, this sensor reading is processed against the threshold values specified in the ***SensorDeviationDetection*** EPA. Please note that the values are customized per application from the template application, as mentioned in Figure 2

For the businessProcessId 11234 those values are (Figure 4):



Figure 4: Threshold values per business Process Id 11234

**Allowed sensor values for this farmer are:**

* 10 ≤ temperature ≤ 33
* 5 ≤ luminosity ≤ 40
* 50 ≤ airHumidity ≤ 80
* 5 ≤ PH ≤ 7
* 1.5 ≤ EC ≤ 3.5
* 60 ≤ soilMoisture 90
* 200 ≤ CO2 ≤ 1000

On the other hand, for business process with id 45678 the threshold values specified are

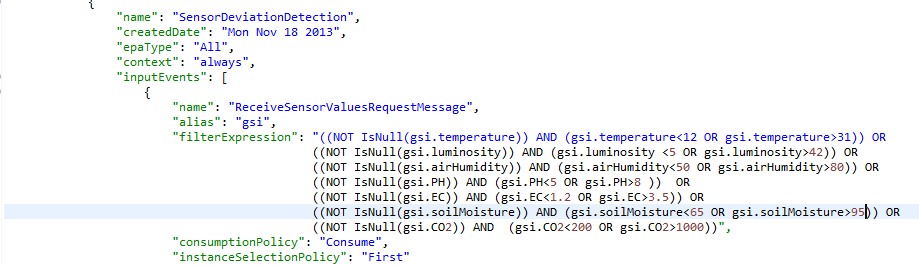


Figure 5: Threshold values per business Process Id 45678

**Allowed sensor values for this farmer are:**

* 12 ≤ temperature ≤ 31
* 5 ≤ luminosity ≤ 42
* 50 ≤ airHumidity ≤ 80
* 5 ≤ PH ≤ 8
* 1.2 ≤ EC ≤ 3.5
* 65 ≤ soilMoisture 95
* 200 ≤ CO2 ≤ 1000

Once a sensor values violation is detected for the first time based on those threshold values, EPM creates and send a **B2BOutOfBoundariesNotification** event to the BCM consumer (this is an outcome of the calculation) – see Figure 5.



Figure 6: SensorReadOutOfBoundaries EPA and its derived event, B2BOutOfBoundariesNotification

One thing to understand regarding the outbound events in general –since the communication is done via CSB, eventually the event is transformed into a POJO object which is then sent via the CSB mechanism. The definitions of how to transform the derived events and how to send them using CSB are placed in CSB consumer.(see Figure 7)



Figure 7: CSB consumer definition

One of the relevant definitions is the path to the jar file where the classes for such POJO objects are stored. The transformation from derived event to a POJO is done based on name matching – for this particular event, we will look inside the jar specified in the CSB consumer “jarPath” property, and will look for a class named “B2BOutOfBoundariesNotification”. We expect to find such a class with setters for fields which match the attribute names of the ***B2BOutOfBoundariesNotification*** event.

In case EPM cannot find such a custom POJO in the consumer jarPath, it will construct a generic message (**EpmBcmGenericMessage**) which will be transferred via CSB to BCM and BCM knows how to handle it.[[3]](#footnote-4) This generic message has the following built in information:

* **eventTypeName** – the name of the event type this message represents. In our sample case, it will hold “B2BOutOfBoundariesNotification” value
* **businessProcessId** – will hold the businessProcessId value
* **attributes** – is a map of String keys and Object values, this map with be populated with attribute names of the derived event (in our sample case, “airHumidity” for example, since B2BOutOfBoundariesNotification event type has such attribute) and their respective values.

Once the B2BOutOfBoundariesNotification message reaches BCM, it initiates a business process handling of the threshold violation. This involves sending a request for advice (GreenhouseAdviceRequest) and when receiving the AdviceResponse, in forwarding it, and closing the EPM context explicitly, using a context termination notification message. See the BCM application definitions and flow section for the details of configuring BCM for this scenario.

There exists a predefined context termination message which BCM can send to EPM in order to close the EPM context; this should usually be done once a business process is terminated, but for more advanced processes, it may be done earlier. The object representing this message is named **B2BContextTerminationNotification**. It has a **businessProcessId** field, and 10 String fields, named field1-field10. Those additional fields can represent any value which should be taken into account when terminating context. This event should be defined as part of EPM events definition, and should be defined as context termination event for the relevant context.

Let’s took event at this event’s definition in EPM configuration for greenhouse scenario.

This is the **B2BContextTerminationNotification** event definition in EPM (Figure 8). Please note that in this scenario we are only interested in the business process id, so we do not have to declare any of the ten String fields of the **B2BContextTerminationNotification** object here.[[4]](#footnote-5)



Figure 8 : B2BContextTerminationNotification event structure

In the greenhouse scenario, the **B2BContextTerminationNotification** is defined as a terminator of the temporal context within the composite **DeviationFarmIdFieldId** context. (see Figure 9 for composite context definition)



Figure 9: Composite context definition

The **FromDeviationToAdvice** temporal context is defined as: (Figure 10)



Figure 10: Temporal context definition

The B2BContextTerminationNotification event is defined as terminator event for the temporal context.

Given all definitions, a **B2BContextTerminationNotification** message will terminate the composite context instance of the matching ***business process id*** and will allow the request for advice process for this farm and crop to happen again in case of new threshold violations.

## BCM application definitions and flow

**Business Entity (BE)**: GreenhouseAdvice

**GSM model (\*)**:



Figure 11: BCM – GSM model

**Greenhouse Information model:**



Figure 12: Greenhouse information model

To begin configuration, one should use the BCM Configuration Wizard in the SDK. This includes choosing the domain (Agriculture), the name for the business entity (GreenhouseAdvice), and what messages should start the business process (eu.fispace.api.ag.GreenhouseAdviceRequest), and of course, selecting the previously created Business Process Template. Note that our business process can be started either by the farmer making an explicit request, or when the custom B2BOutOfBoundsNotification message arrives from EPM[[5]](#footnote-6). Custom messages always arrive to the bridge as an **EpmBcmGenericMessage**. The default handling in the CSB-BCM bridge when receiving instances of this message is to create a new business entity (and thus a new lifecycle), using the business process id to instantiate the entity. Furthermore, the message is sent as an XML event – the root element is the **eventTypeName** and its children elements are: **businessProcessId** and each key appearing in **attributes**, the values for each element are taken out from the **EpmBcmGenericMessage** as appropriate.

In order to receive the event in ACSI, it is necessary to define an additional data type (by right clicking on *Data Types* and selecting *Create new data type*). Enter the data-type name as B2BOutOfBoundariesNotification, and add each field as an attribute – the name should exactly match (e.g., *luminosity*), the type should match as well (e.g., *double*)[[6]](#footnote-7) and *Max Occurs* should be **n** (see Figure 14). Then, an *Event* should be added for it beneath the *Event Model*. Right-click on *Event Model* and select *Create Event*. Enter the *Event Name* as B2BOutOfBoundariesNotification, and select B2BOutOfBoundariesNotification for both the input message and output message (see Figure 15). As well, it is recommended to add an attribute called B2BOutOfBoundariesNotification of type B2BOutOfBoundariesNotification to the business entity’s information model. We will later have to define a correlation key for B2BOutOfBoundariesNotification, as it is a custom message.

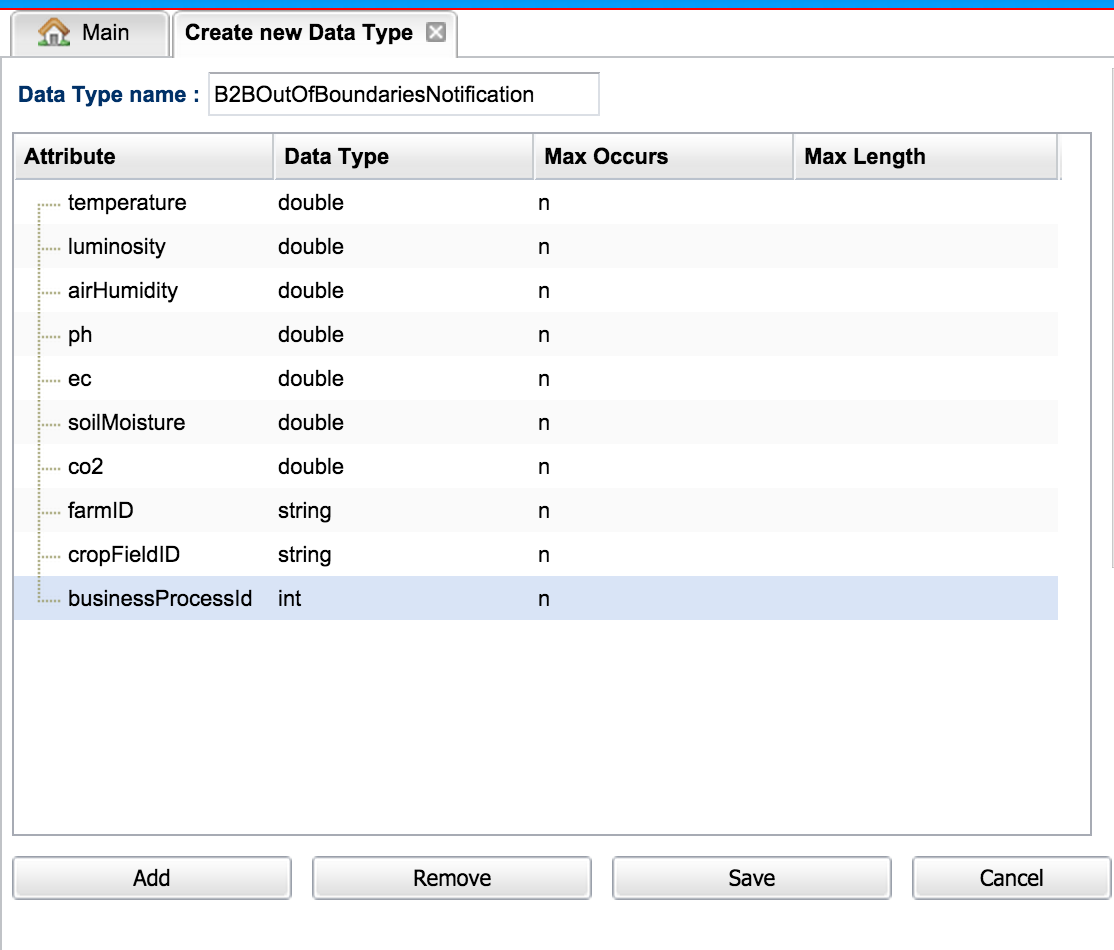


Figure 14 B2BOutOfBoundariesNotification Data Type

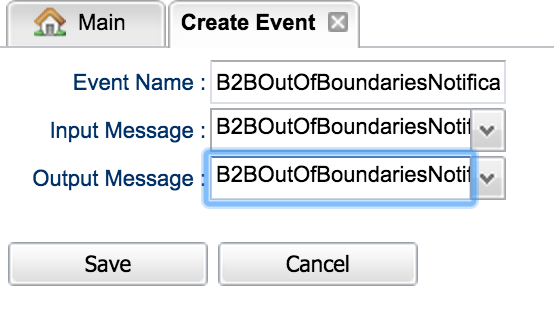


Figure 15 B2BOutOfBoundariesNotification Event

Now we are ready to define the lifecycle. Here, we define a root stage for the whole lifecyle, and then define sub-stages which deal with our events; what to do when we receive the B2BOutOfBoundariesNotification message and the resourceAvailableNotification.[[7]](#footnote-8) Figure 16 shows what the information model should look like. Figure 17 shows the GSM lifecycle stages. When we receive the B2BOutOfBoundariesNotification message we send out a request for advice (leveraging the external service for the capability type). When we receive the resourceAvailableNotification we likewise notify the farmer (again, using the appropriate external service), as well as closing the EPM context. The EPM context is closed by invoke the **close\_epm\_context** service and sending the predefined **B2BContextTerminationNotification** message. It is important when sending this message to populate the **businessProcessId** field from the business entity, or from the resourceAvailableNotification itself. See the figures below highlighting different aspects of the lifecycle (not all stages and tasks are shown below, click through the screens in the loaded sample to see how everything is set up).

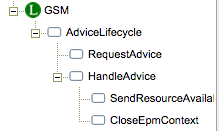
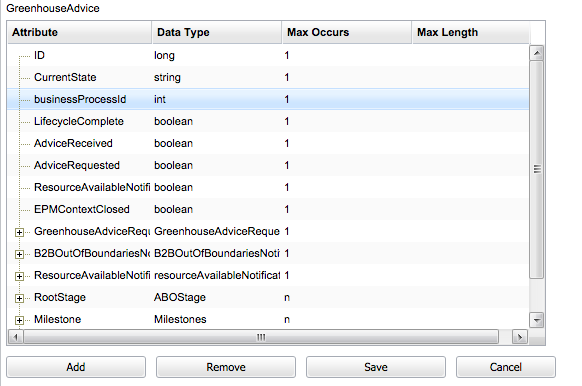
 

Figure 17 GSM Lifecycle

Figure 16 GreenhouseAdvice Information model

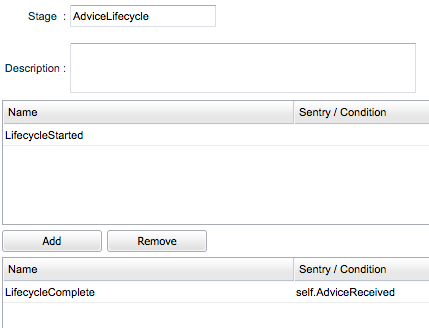


Figure 18 Root stage for the GSM lifecycle

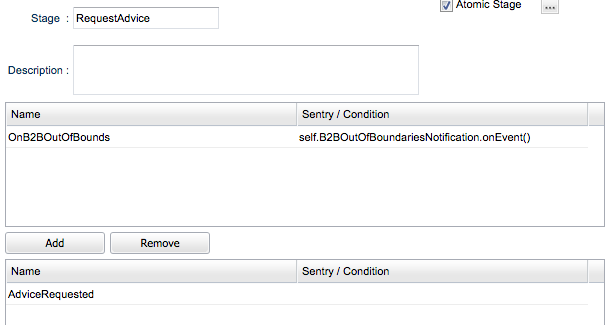
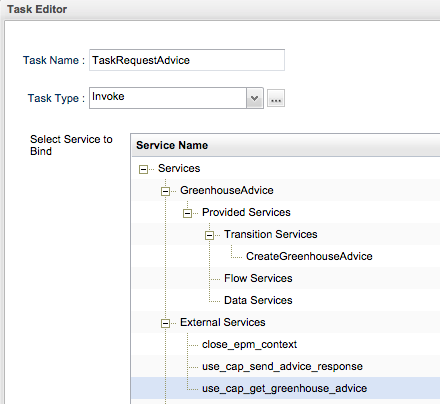
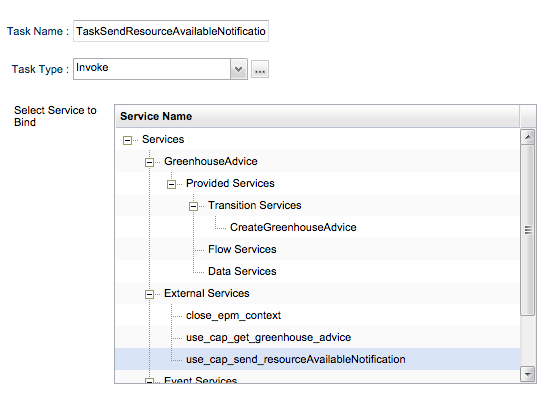


Figure 19 RequestAdvice stage with associated task (reached by clicking … button next to Atomic Stage on the stage editor page).



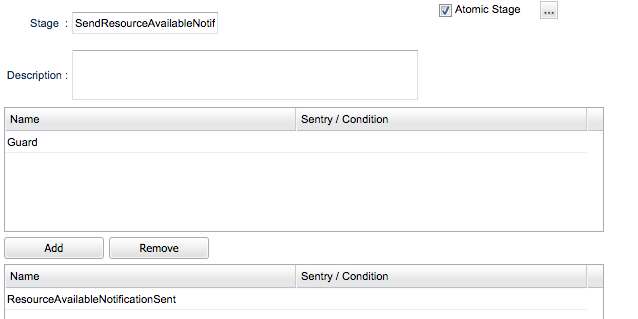


Figure 20 SendResourceAvailableNotification stage with associated task (reached by clicking … button next to Atomic Stage on the stage editor page).

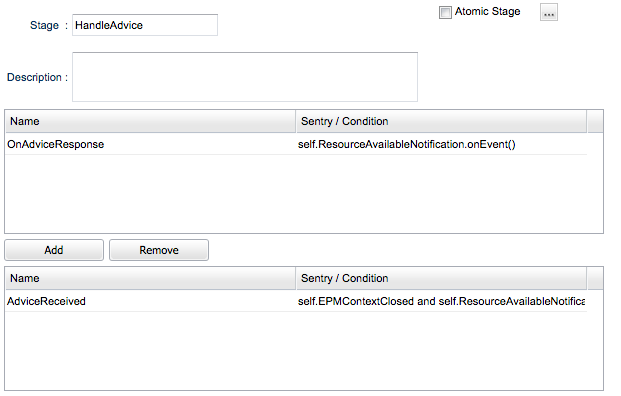


Figure 21 HandleAdvice parent stage

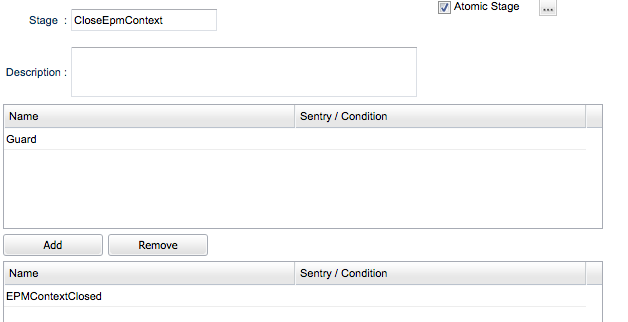
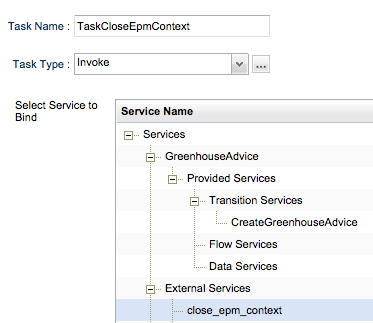


Figure 22 CloseEpmContext stage with associated task

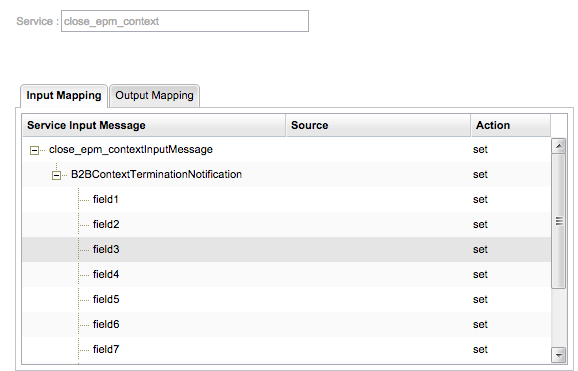
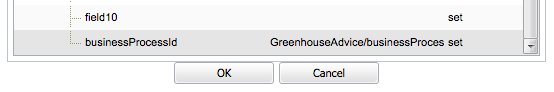


Figure 23 Setting the data to be sent when invoking the close\_epm\_context external service (reachable by clicking the … button next to Task Type)

1. Contributors to FIspace code (“P”) include ATB, UDE, IBM, ATOS, KOC, TOG, AST, NKUA, UPM and LimeTri; contributing persons are listed at <https://bitbucket.org/fispace/profile/members> [↑](#footnote-ref-2)
2. Once the relevant information regarding additional attributes are added to the core api messages, the EPM application can use a segmentation context, so that it monitors and treats sensor readings of each farm and crop separately by using an application instance instead of a new application, as it is currently done. In execution time it means that if it receives a notification of sensor readings per some farm and crop which exceeds threshold vales, and asks for advice, if it receives another message which also exceeds threshold values for a different crop within the same farm, or for the same crop in different farm, it will ask for advice as well. However if it receives additional notification for the same farm and crop for which it already reported readings out of bounds and asked for advice, it will not notify again until such an advice is received. [↑](#footnote-ref-3)
3. In any case, whether what transmitted to BCM is the generic message with the structure defined above, or a custom POJO, those objects will be wrapped in an envelope object, additional information from CSB consumer such as **businessProcessId** and **businessProcessTemplateId** (see Figure 6) will be added to this envelope message, and this will be transmitted to BCM [↑](#footnote-ref-4)
4. Had we used some fields to segment the context by, these should then be included in the **B2BContextTerminationNotification** message, which is sent from BCM to EPM (and the mapping of these fields to the message’s ***field1, field2,*** etc., should be identical in both BCM and EPM (since the business architect is the one building both the EPM and the BCM applications configurations she needs to remember which field signifies which value).

   For this particular scenario, what we would be interested in in order to terminate the context is which farm and crop is the termination related to, since all the processing is done per farm-crop level. So we would need to the farmId and cropFieldId values inside the termination event in order to terminate the context. [↑](#footnote-ref-5)
5. Consult the BCM Guide for full details on what to do when sending custom events from EPM to BCM. [↑](#footnote-ref-6)
6. When under the Data Type combo-box for an attribute, the list is alphabetically ordered and you can press the letter that starts the name of the data type, such as “d” for “double”, and it will go directly to types beginning with those letters. If you encounter a list which isn’t sorted, you can still press a letter to cycle through data types beginning with that letter. Note that there are some internally derived data types in the list, which begin with the \_\_T\_ prefix, these generally shouldn’t be selected by the user. [↑](#footnote-ref-7)
7. In general, the message ResourceAvailableNotification is used to indicate when data is available – this message doesn’t have any further identifying information, and thus should only be used when the business entity is only identifiable by the business process id. Note that the data type for ResourceAvailableNotification in ACSI is resourceAvailableNotification - this matches the element defined in the XSD schemas – there is also a \_\_T\_ResourceAvailableNotification data type. This data type is marked with the \_\_T\_ prefix to indicate that is for internal use – it represents the complex type ResourceAvailableNotification defined in the XSD schemas, but it should not be sent (as it doesn’t have a matching element definition in the XSD schemas – i.e., it is not declared to be a message type). [↑](#footnote-ref-8)