D7.11 – On-board Integration Platform 1

**Due date of deliverable:** 31/03/2016  
**Actual submission date:** 30/03/2016  
**Organisation name of lead contractor for this deliverable:** SAG  
**Revision:** 1.0

**Grant Agreement Nº:** 607567  
**Project Acronym:** IPATCH  
**Project Title:** Intelligent Piracy Avoidance using Threat detection and Countermeasure Heuristics  
**Funding Scheme:** SEC-2013.2.4-2  
**Start date of project:** 01/04/2014  
**Duration:** 36M

<table>
<thead>
<tr>
<th>Dissemination Level</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PU</strong> Public</td>
<td></td>
</tr>
<tr>
<td><strong>RE</strong> Restricted to a group specified by the consortium (including the Commission)</td>
<td>✓</td>
</tr>
</tbody>
</table>
# Table of Contents

Executive Summary ........................................................................................................ 6

1 Introduction .................................................................................................................. 7
   1.1 The IPATCH Project ................................................................................................. 7
   1.2 Deliverable overview .............................................................................................. 7

2 Integration Platform ...................................................................................................... 9
   2.1 Overall design concept .......................................................................................... 9
   2.2 Status ..................................................................................................................... 11

3 Evolution of ARENA prototype .................................................................................... 12
   3.1 Topics .................................................................................................................... 12
   3.2 Roles ..................................................................................................................... 12
   3.3 Notification ........................................................................................................... 12
   3.4 Negotiation ........................................................................................................... 13
   3.5 Data model ........................................................................................................... 13
   3.6 Tools .................................................................................................................... 13

4 Conclusions and future work ...................................................................................... 15

Appendix A: Demonstration of basic functionality ......................................................... 16
Document Summary Information

Authors and Contributors

<table>
<thead>
<tr>
<th>Initials</th>
<th>Name</th>
<th>Organisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP</td>
<td>Romuald Périnelle</td>
<td>SAG</td>
</tr>
</tbody>
</table>

Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Initials</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>02/03/2016</td>
<td>RP</td>
<td>Initial version</td>
</tr>
<tr>
<td>0.2</td>
<td>22/03/2016</td>
<td>GS</td>
<td>Peer review</td>
</tr>
<tr>
<td>1.0</td>
<td>30/03/2016</td>
<td>TC</td>
<td>Final review before submission</td>
</tr>
</tbody>
</table>

Quality Control

<table>
<thead>
<tr>
<th>Role</th>
<th>Name</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer Review</td>
<td>Genevieve Sella</td>
<td>22/03/2016</td>
</tr>
<tr>
<td>Work Package Leader</td>
<td>Grzegorz Taberski (ITTI)</td>
<td>30/03/2016</td>
</tr>
<tr>
<td>Project Manager</td>
<td>Tom Cane</td>
<td>30/03/2016</td>
</tr>
</tbody>
</table>

Security Scrutiny Committee Review

<table>
<thead>
<tr>
<th>Comments</th>
<th>Recommended Distribution</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>No comments</td>
<td>Consortium and Commission services</td>
<td>30/03/2016</td>
</tr>
</tbody>
</table>
Disclaimer

The content of the publication herein is the sole responsibility of the publishers and it does not necessarily represent the views expressed by the European Commission or its services.

While the information contained in the documents is believed to be accurate, the authors(s) or any other participant in the IPATCH consortium make no warranty of any kind with regard to this material including, but not limited to the implied warranties of merchantability and fitness for a particular purpose.

Neither the IPATCH Consortium nor any of its members, their officers, employees or agents shall be responsible or liable in negligence or otherwise howsoever in respect of any inaccuracy or omission herein.

Without derogating from the generality of the foregoing, neither the IPATCH Consortium nor any of its members, their officers, employees or agents shall be liable for any direct or indirect or consequential loss or damage caused by or arising from any information advice or inaccuracy or omission herein.
List of Figures

Figure 1: Integration Platform architecture overview ................................................................. 9
Figure 2: IP basic functionality: publication .................................................................................. 10
Figure 3: IP basic functionality: events ......................................................................................... 10
Figure 4: IP basic functionality: getting data .................................................................................. 11
Figure 5: GUI Modules list ............................................................................................................. 16
Figure 6: GUI Message list ............................................................................................................. 17
Figure 7: GUI Event list .................................................................................................................. 17
Figure 8: GUI Data request tab ...................................................................................................... 18

List of Tables

Table 1: Supported platforms ......................................................................................................... 11
Table 2: Auxiliary software tools .................................................................................................. 14

List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP</td>
<td>Integration Platform</td>
</tr>
<tr>
<td>ZMQ</td>
<td>Name of a library used that provide intelligent sockets.</td>
</tr>
<tr>
<td>SQLite</td>
<td>Name of a library providing standard SQL interface including JDBC for a local file database.</td>
</tr>
<tr>
<td>SQL</td>
<td>Simple Query Language – standard for querying databases</td>
</tr>
<tr>
<td>JDBC</td>
<td>Java Data Base Connector – Java standard application programmable interface to access to databases.</td>
</tr>
<tr>
<td>WP</td>
<td>Work Package</td>
</tr>
<tr>
<td>ID</td>
<td>Identifier – unique name of a computer managed entity.</td>
</tr>
</tbody>
</table>
Executive Summary

The Integration Platform for IPATCH is an evolution of Integration Platform developed within the FP7 project ARENA. The architectural evolutions were described in IPATCH deliverable D2.3 and the implementation and refinements have been carried out in task 7.1. The main evolutions contain, besides the data model, the inclusion of topics in the publish/subscribe architecture, the clarification of modules’ roles, and a rewriting of the C++ library.

The first phase of task 7.1 succeeded in providing the evolutions required for the IPATCH use case. The next phase will consist of code convergence, bug fixing, and better documentation, as well as adapting to any new requirements which arise as a result of the first field trials in WP8.
1 Introduction

1.1 The IPATCH Project

Funded by the European 7th Framework Programme, the IPATCH project addresses Security Topic SEC-2013.2.4-2: Non-military protection measures for merchant shipping against piracy. The goal of the IPATCH project is three-fold:

1. To perform an in-depth analysis of the legal, ethical, economic and societal implications of existing counter piracy measures.
2. To produce well-founded recommendations to the industry in the form of a manual, extending and complementing the Best Management Practices document and to support the use and further development of countermeasures.
3. To develop an on-board automated surveillance and decision support system providing early detection and classification of piracy threats and supporting the captain and crew in selecting the most appropriate countermeasures against a given piracy threat.

The analysis performed under (1) will lead to recommendations for the use of countermeasures in a range of scenarios, structured as a manual (2), and development and implementation of a proactive surveillance system forming part of the system developed in (3). The situational awareness system will robustly monitor the area around maritime vessels, providing early warning to crew members if piracy threats are detected. A low false alarm rate due to environmental or other innocuous events, combined with high threat detection sensitivity are central ambitions of the project.

To achieve these goals, a multispectral sensor suite comprising both passive and active sensors is envisaged, i.e., a system based on radar, visual and thermal sensors. The sensor suite will be complemented with advanced algorithms for information fusion, object detection and classification, and high level modelling of intent and behaviour analysis. The IPATCH project is strongly user-driven and demonstration of the developed surveillance system will be conducted in two different maritime environments.

1.2 Deliverable overview

This report describes the design and development of the integration platform component in task 7.1 of the IPATCH project. According to the DoW:

This task will focus on the design and implementation of the middleware and hardware platform which will be used to integrate the different sensor and algorithm components of the on-board system. The Generic Architecture from the ARENA project will be analysed to assess how well it meets the specific requirements of IPATCH. Subsequently, the architecture will be modified according to the requirements specified in WP2 and additional requirements from WP3-6. The task will specify the interfaces for the Early Detection Module (WP5), the Threat Recognition and Classification Module (WP6), the Decision Support Module (T7.2), a common database, integration with other on-board systems, and the HMI (T7.3). Based on the detailed design specification, the On-board Integration Platform will be implemented, tested and refined.

Description of the deliverable: The On-board Integration Platform will be a technological solution for bringing together the different hardware and software components which were developed in the
The Integration Platform is thus a critical component of the IPATCH on-board system. Although it does not play a role in the processing of data, it facilitates the flow from data sources to data consumers. Thus, the main constraints on the integration platform are that it shall be easy to use (from the perspective of the algorithm developers), and it shall provide debug, replay, and other features that are necessary to support the development of a complex, multi-partner system.

This document begins with a description of how the integration platform design has been adapted from the first ARENA version to be more suited for the IPATCH use case. It then gives an overview of the current status of the platform and explains each of the evolutions in more detail. Finally, the conclusions outlines the anticipated further work over the second phase of task 7.1.

To demonstrate the functionality of the Integration Platform, Appendix A provides a brief overview of the basic operation, accompanied by screenshots.
2 Integration Platform

2.1 Overall design concept

The Integration Platform (IP) allows all IPATCH modules to communicate using a common data model and publish/subscribe mechanisms. To limit specific developments, and since the ARENA integration platform gave good results, it has been decided to continue with the overall design concept and to reuse code from the ARENA project.

The global architecture is still the same, as explained in IPATCH deliverable D2.3. There are still Java and C/C++ libraries that communicate with each other and with a central server application, as shown on the schema below. The IP provided elements are shown in blue and the application specific developments are shown in violet. Arrows show what sort of messages go through the server and which ones are exchanged directly between applications.

![Integration Platform architecture overview](image)

While the architecture did not change, the use has evolved. The logical communication is still a pipeline, (or, to be more precise, an oriented graph) and the physical communications still go through the server. However, the message exchange is now based on a publish/subscribe pattern supported by a data repository.
Published messages (Ipatch : by topics)

**Figure 2: IP basic functionality: publication**

In the figure above, the messages are sent to the Server, which publishes them to the Repository. Then, in the next figure, the Repository notifies (still through the Server) each module for the changes using events:

Published messages (Ipatch : by topics)

**Figure 3: IP basic functionality: events**
Once the modules have received the event, they are free to ask the Repository for any object. This communication does not use the Server, thus keeping message bandwidth to a minimum:

![Diagram](image)

**Figure 4: IP basic functionality: getting data**

### 2.2 Status

The first Java version was available in September 2015. At that time, the data model was in its first version. By the end of February 2016, a first version of the C/C++ was completed, but untested. The bug fixing and integration of Java and C++ versions was done in early March. By mid-March, the C++ version is available for both Windows 64-bit and Linux 64-bit architecture. The work is now in the final convergence and maintenance phase. The platform compatibilities are summarised in the table below:

<table>
<thead>
<tr>
<th>Language</th>
<th>OS</th>
<th>Compiler version</th>
<th>Dependencies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java</td>
<td>32 bits JVM</td>
<td>Java 7</td>
<td>ZMQ 3.2.2, JZMQ, SQLite 3.8.10.1 (Repository only)</td>
</tr>
<tr>
<td>Java</td>
<td>64 bits JVM</td>
<td>Java 7</td>
<td>ZMQ 4, JZMQ, SQLite 3.8.10.1 (Repository only)</td>
</tr>
<tr>
<td>C++</td>
<td>Windows 64 bits</td>
<td>Visual 2013 (vc120)</td>
<td>Boost 1.60, ZMQ 4.0.4, Xerces-c: 3.1.3, XSD: 4.0.0, Pthread-w32:2.9.1</td>
</tr>
<tr>
<td>C++</td>
<td>Linux 64 bits</td>
<td>G++ 4.8.4</td>
<td>Boost 1.54, ZMQ 4.0.4, Xerces-c: 3.1.1, XSD: 3.3.0, Pthread: 2.19</td>
</tr>
</tbody>
</table>
3 Evolution of ARENA prototype

Starting with the ARENA prototype, the main improvements of the new Integration Platform in IPATCH are:

- Use of topics to specialize communications and lower bandwidth usage
- Clarification of roles: layer 2 roles (sender, receiver, etc.) are now separated from the layer 3 roles (object assessment, situation assessment, …)
- Change in notification: from distributing object in ARENA, we go to storing objects on a database and give them on request, and to publish the Create / Update / Delete events to registered modules
- Negotiation between modules has been discontinued
- Rework of C/C++ version for performance reasons

Besides that, a big difference is that the ARENA prototype was not designed to be run in real-time and thus needed a scheduling mechanism to synchronize the replays and a frame-by-frame distribution of video data. The IPATCH system aims for real-time operation with asynchronous processing, and thus uses real video streams and no scheduling.

The main evolutions are described further in the following sections.

3.1 Topics

The introduction of topics prompted some changes in the data model:

- The main class for communication is now AbstractTopicObject instead of AbstractDataModel
- A short name of the topic is added at header before the XML message
- ZMQ is asked to use that prefix as register filter, thus implementing the source filtering

3.2 Roles

The clarification of roles in the system led to an update in the module registering method signatures as well as the module classes and messages.

3.3 Notification

The new notification paradigm was implemented as follows:

- A new version of the Data Repository is implemented, using SQLite
- The Data Repository is registered as DataConsumer (old way of ARENA), and should be the only one doing so (besides debug tools)
- Other modules are registered as DataProducer and/or DataEventConsumer
- When data is created/updated/deleted, modules send the new object, the Data Repository receives it, records it, and sends events
DataEventConsumers receive the events, and, if they need the real data, they ask the repository for it.

Given the keys on which the data are requested, a single table has been used in the Data Repository. This could be easily changed in the future, if necessary. The advantage of the current implementation is that it is greatly independent of the data model, and thus the repository does not have to be updated every time there is a change to the data model.

### 3.4 Negotiation

In WP2, it was foreseen to implement dynamic negotiation between data providers and data consumers, based on sets of properties.

At the time of the development of the Integration Platform, the studies on that subject have not converged to a stable solution, because of the following issues:

- The list of parameters on which the negotiation had to be done was not known. The list need not be exhaustive, but it was not long enough to ensure that most of difficulties could be addressed.
- The optimal strategy was not unique: for some parameters, the higher the better; for others, the lower the better; for others, more complex rules had to be set up. This implies that the strategies have to be listed in an extensible way so that they can be referenced in the data model, and expanded when new strategies have to be set up for new configuration data.

Furthermore, it became clear as the project progressed that the feature was not critical for the performance of algorithms and modules. It was therefore decided to discontinue this line of development in IPATCH, but retaining the possibility of adding it as an extra feature in the future.

### 3.5 Data model

As in ARENA, the Integration Platform is largely independent of the Data Model. The few dependencies are:

- The main class of the transmitted data needs to be known and contain data ID and timestamp
- The header of configuration messages needs to be known

Thus, the interfaces between specific modules have no impact on the Integration Platform as long as they are compatible with the dynamics defined in WP2. In integration testing so far, this has proved to be the case.

### 3.6 Tools

In ARENA, a number of auxiliary software tools were developed to support the testing and debugging of the whole on-board system. Because of the changes described in the previous sections, these tools are also being updated and extended to support the testing and further development of the IPATCH system. The current status is summarised in Table 2.

To demonstrate the functionality of the Integration Platform and some of the tools, Appendix A provides a brief overview of the basic operation, accompanied by screenshots.
<table>
<thead>
<tr>
<th>Tool name</th>
<th>ARENA Availability</th>
<th>IPATCH Availability</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUISensorManager</td>
<td>Yes</td>
<td>Not yet</td>
<td>List and pilot Sagem sensor managers</td>
</tr>
<tr>
<td>DataEventSubscriberStub</td>
<td>No</td>
<td>Yes</td>
<td>Write summaries of events on console, filtered by topics.</td>
</tr>
<tr>
<td>DataProviderStubService</td>
<td>Yes</td>
<td>Yes</td>
<td>Send objects, either a predefined list, or from a directory of recorded XML messages.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>List registered modules, sent messages, sent events.</td>
</tr>
<tr>
<td>GUIStubFrame</td>
<td>Yes</td>
<td>Yes</td>
<td>Allow to request data from repository.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Allow to request image from sensor manager (obsolete in IPTACH), records messages in XML format for further replay.</td>
</tr>
<tr>
<td>RepositoryStubService</td>
<td>Yes</td>
<td>No</td>
<td>Obsoleted in IPATCH, replaced by RepositorySQLiteService.</td>
</tr>
<tr>
<td>RepositorySQLiteService</td>
<td>No</td>
<td>Yes</td>
<td>Store every sent message into a database. Respond to getData() requests.</td>
</tr>
<tr>
<td>SensorManagerStub</td>
<td>Yes</td>
<td>No</td>
<td>Provide Sensor manager like features from a directory of images instead of a camera.</td>
</tr>
<tr>
<td>SchedulerGUI</td>
<td>Yes</td>
<td>No</td>
<td>Scheduler to synchronize modules in a non-real time system.</td>
</tr>
<tr>
<td>TimeSteppedStub</td>
<td>Yes</td>
<td>No</td>
<td>Dummy module to synchronize on a given duration by step or to provide data from a directory in a synchronized way.</td>
</tr>
</tbody>
</table>
4 Conclusions and future work

The first phase of task 7.1 has successfully implemented the modifications and updates needed to migrate from the ARENA prototype to a working integration platform tailored for the IPATCH use case. The main aim was to support real-time operation, which was achieved by moving to a publish/subscribe message exchange pattern, whilst retaining the logical pipeline model for the data processing chain. The scheduling mechanism has been removed and data access is now managed by a repository.

The second phase of task 7.1 will continue to update and maintain the integration platform components and auxiliary tools to support the further development and testing of other modules in the on-board system. This work, and any further extensions, will be reported in deliverable D7.12. This deliverable will also provide documentation for the components and instructions on how the Integration Platform should be installed and configured.
Appendix A: Demonstration of basic functionality

For illustrating the Integration Platform, we ran the three basic programs:

- The Server
- The RepositorySQLite
- The GUIStub

To have details of events, we start also the DataEventReceiverStub, which gives the received events on the console.

Once this is done, no data messages have been exchanged. To have some, we start aDataProviderStubService.

At this time, the GUI first tab gives the list of registered modules:

![Figure 5: GUI Modules list](image)

On the second tab, we get a list of sent data. Only the Repository and the GUI test tool receive them:
The other modules subscribe to the events. These events are listed in tab “Events”:

**Figure 6: GUI Message list**

**Figure 7: GUI Event list**
In this last tab, there is a trick: the End validity is not given in the event. Actually, the GUI did request the repository to get this information. All other columns are from the event.

The GUI allows also requesting recorded data, either as object ("Send request" button) or as Id list ("Send request Ids" button). In the example below we asked for Threats as objects. We got threats that were recorded in a previous session. As these are objects and not only Ids, all the columns are filled.

![GUI Data request tab](image)

**Figure 8:** GUI Data request tab